1992

Research and development of microcontroller experiment instructional units and their effectiveness with industrial technology, electronic technology, and electrical engineering technology majors

Steve Chih Hsiung
Iowa State University

Follow this and additional works at: https://lib.dr.iastate.edu/rtd

Part of the Communication Technology and New Media Commons, Electrical and Computer Engineering Commons, Engineering Education Commons, Instructional Media Design Commons, and the Other Education Commons

Recommended Citation
Hsiung, Steve Chih, "Research and development of microcontroller experiment instructional units and their effectiveness with industrial technology, electronic technology, and electrical engineering technology majors " (1992). Retrospective Theses and Dissertations. 10003.
https://lib.dr.iastate.edu/rtd/10003

This Dissertation is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Retrospective Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.
INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.
Research and development of microcontroller experiment instructional units and their effectiveness with industrial technology, electronic technology, and electrical engineering technology majors

Hsiung, Steve Chih, Ph.D.

Iowa State University, 1992
Research and development of microcontroller experiment instructional units and their effectiveness with industrial technology, electronic technology, and electrical engineering technology majors

by

Steve Chih Hsiung

A Dissertation Submitted to the Graduate Faculty in Partial Fulfillment of the Requirements for the Degree of DOCTOR OF PHILOSOPHY

Major: Industrial Education and Technology

Approved:

Signature was redacted for privacy.
In Charge of Major Work

Signature was redacted for privacy.
For the Major Department

Signature was redacted for privacy.
For the Graduate College

Iowa State University
Ames, Iowa

1992
## TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER I. INTRODUCTION</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background</td>
<td>1</td>
</tr>
<tr>
<td>Problem of the Study</td>
<td>5</td>
</tr>
<tr>
<td>Purpose of the Study</td>
<td>5</td>
</tr>
<tr>
<td>Objectives of the Study</td>
<td>6</td>
</tr>
<tr>
<td>Assumptions of the Study</td>
<td>7</td>
</tr>
<tr>
<td>Delimitations of the Study</td>
<td>7</td>
</tr>
<tr>
<td>Procedures of the Study</td>
<td>8</td>
</tr>
<tr>
<td>Definition of Terms</td>
<td>9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CHAPTER II. REVIEW OF LITERATURE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microprocessor High Technology Impact</td>
<td>13</td>
</tr>
<tr>
<td>Historical Development of Microprocessors</td>
<td>16</td>
</tr>
<tr>
<td>The 8 Bit Microprocessor World</td>
<td>18</td>
</tr>
<tr>
<td>Intel</td>
<td>20</td>
</tr>
<tr>
<td>Motorola</td>
<td>22</td>
</tr>
<tr>
<td>Zilog</td>
<td>22</td>
</tr>
<tr>
<td>Commodore</td>
<td>22</td>
</tr>
<tr>
<td>National Semiconductor</td>
<td>23</td>
</tr>
<tr>
<td>Fairchild</td>
<td>23</td>
</tr>
<tr>
<td>Microprocessor Curriculum Efforts</td>
<td>23</td>
</tr>
<tr>
<td>Differences between Microprocessors and Microcontrollers</td>
<td>33</td>
</tr>
<tr>
<td>General Description of Microcontroller 68HC11 EVB and EVM</td>
<td>34</td>
</tr>
<tr>
<td>Summary of Literature Review</td>
<td>39</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CHAPTER III. METHODOLOGY</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design and Development of the Seven Explanatory Laboratory Experiments</td>
<td>42</td>
</tr>
<tr>
<td>Hypotheses of the Study</td>
<td>43</td>
</tr>
<tr>
<td>Sampling Subjects</td>
<td>44</td>
</tr>
<tr>
<td>Research Design</td>
<td>45</td>
</tr>
<tr>
<td>Data Collection Instrument</td>
<td>47</td>
</tr>
<tr>
<td>Statistical Analysis of the Data</td>
<td>47</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CHAPTER IV. RESULTS AND FINDINGS</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Presentation of the Seven Explanatory Laboratory Experiments</td>
<td>49</td>
</tr>
<tr>
<td>Experiment 1</td>
<td>49</td>
</tr>
<tr>
<td>Experiment 2</td>
<td>54</td>
</tr>
<tr>
<td>Experiment 3</td>
<td>55</td>
</tr>
<tr>
<td>Experiment 4</td>
<td>60</td>
</tr>
<tr>
<td>Experiment 5</td>
<td>70</td>
</tr>
<tr>
<td>Experiment 6</td>
<td>71</td>
</tr>
</tbody>
</table>
**LIST OF FIGURES**

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Motorola microprocessor family evolution</td>
<td>21</td>
</tr>
<tr>
<td>2</td>
<td>Family tree of microprocessors and microcontrollers</td>
<td>37</td>
</tr>
<tr>
<td>3</td>
<td>The standard microprocessor</td>
<td>38</td>
</tr>
<tr>
<td>4</td>
<td>The 68HC11 microcontroller</td>
<td>38</td>
</tr>
<tr>
<td>5</td>
<td>Stepper motors control circuit</td>
<td>51</td>
</tr>
<tr>
<td>6</td>
<td>Flow chart of stepper motors control</td>
<td>52</td>
</tr>
<tr>
<td>7</td>
<td>Stop lights control circuit</td>
<td>56</td>
</tr>
<tr>
<td>8</td>
<td>Flow chart of street lights control</td>
<td>57</td>
</tr>
<tr>
<td>9</td>
<td>Software debouncing keypad encoding circuit</td>
<td>61</td>
</tr>
<tr>
<td>10</td>
<td>Hardware debouncing with 74C922 keypad encoding circuit</td>
<td>62</td>
</tr>
<tr>
<td>11</td>
<td>Flow chart of software debouncing control</td>
<td>63</td>
</tr>
<tr>
<td>12</td>
<td>Flow chart of hardware debouncing control</td>
<td>64</td>
</tr>
<tr>
<td>13</td>
<td>LCD module parallel control circuit</td>
<td>66</td>
</tr>
<tr>
<td>14</td>
<td>LCD module serial control circuit</td>
<td>67</td>
</tr>
<tr>
<td>15</td>
<td>Flow chart of LCD module parallel control</td>
<td>68</td>
</tr>
<tr>
<td>16</td>
<td>Flow chart of LCD module serial control</td>
<td>69</td>
</tr>
<tr>
<td>17</td>
<td>Keypad &amp; LCD module with hardware debouncing control circuit</td>
<td>73</td>
</tr>
<tr>
<td>18</td>
<td>Keypad &amp; LCD module with software debouncing control circuit</td>
<td>74</td>
</tr>
<tr>
<td>19</td>
<td>Flow chart of hardware debouncing in keypad &amp; LCD module control</td>
<td>75</td>
</tr>
<tr>
<td>20</td>
<td>Flow chart of software debouncing in keypad &amp; LCD module control</td>
<td>79</td>
</tr>
</tbody>
</table>
Figure 21. Electronic keys security system control circuit 84

Figure 22. Flow chart of electronic keys security system control 85

Figure 23. DC motor control circuit 90

Figure 24. Flow chart of DC motor control 91

Figure 25. The experimental and the control groups' pretest & posttest scores chart 95
### LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1</td>
<td>Historical development of microprocessors</td>
<td>19</td>
</tr>
<tr>
<td>Table 2</td>
<td>Comparison among 8 bit microprocessors</td>
<td>24</td>
</tr>
<tr>
<td>Table 3</td>
<td>Historical development of microcontrollers</td>
<td>35</td>
</tr>
<tr>
<td>Table 4</td>
<td>The analysis of variance of pretest scores</td>
<td>96</td>
</tr>
<tr>
<td>Table 5</td>
<td>The analysis of variance of posttest scores</td>
<td>97</td>
</tr>
<tr>
<td>Table 6</td>
<td>Pearson correlation coefficient of the pretest and the posttest scores</td>
<td>98</td>
</tr>
<tr>
<td>Table 7</td>
<td>The analysis of covariance of posttest scores</td>
<td>98</td>
</tr>
<tr>
<td>Table 8</td>
<td>Pretest and posttest statistics</td>
<td>99</td>
</tr>
<tr>
<td>Table 9</td>
<td>The analysis of covariance with interaction</td>
<td>100</td>
</tr>
<tr>
<td>Table 10</td>
<td>The analysis of variance of the scores differences</td>
<td>101</td>
</tr>
<tr>
<td>Table 11</td>
<td>The learning achievement of the statistics of the two groups</td>
<td>102</td>
</tr>
<tr>
<td>Table 12</td>
<td>The average days needed for the experimental and control groups</td>
<td>103</td>
</tr>
<tr>
<td>Table 13</td>
<td>The t-test of average days needed for completing a project</td>
<td>104</td>
</tr>
<tr>
<td>Table 14</td>
<td>The preference of the 68HC11 EVB in the two groups</td>
<td>105</td>
</tr>
</tbody>
</table>
CHAPTER I. INTRODUCTION

Motorola Inc., one of the largest semiconductor manufacturers in the world, has claimed that its achievement is "deep in the heart of the state of the art" of semiconductor developments. Its microcontroller section has developed several modules which are very convenient for industry's design applications. Educators teaching students about the high-tech world have an obligation to adopt new devices and up-dated lab experiments into technology programs as early as possible. In other words, it is the responsibility of educators to get students acquainted with what "state-of-the-art" technology really means. Thus, the design and development of the experimental projects on the Motorola MC68HC11 microcontroller EVB (Evaluation Board) and EVM (Evaluation Module) are urgently needed (Lipovski, 1988).

Background

When reviewing western history, it is clear that there were three drastic changes in tool implementation. According to Toffler's (1980) definition, the first change of human history was the agricultural revolution, the second change was the
industrial revolution, and the third change was the information revolution, computer revolution, or the so-called "Third Wave" (Toffler, 1980). The Third Wave has changed western society dramatically. The most important factor that makes the Third Wave so astonishing is the use of computers as tools for calculation, information storage and retrieval, process controlling, and industrial automation and robotics. Not only is the computer fast, but it is also accurate and can complete tedious jobs without becoming bored (Driscoll, 1987).

The application of computers makes a path for the Third Wave. The key point is that the application of the microcomputer, which is accessible to almost everyone and affordable to every interested individual, has radically changed our society. As a result, industrial educators must be prepared to utilize the high technology features, now available in a single integrated circuit chip which is called the microprocessor and which comprises the heart of the microcomputer. At the same time, it can be said that without the help of the microprocessor, there would be no microcomputer. Generally speaking, a microcomputer is a multipurpose, programmable logic device. The microprocessor within the microcomputer is the brain of this device. In order to introduce high technology to the students, industrial educators have to explore the major concepts about how the microprocessor really functions. So, an effective way to teach the students the process and control applications of the microcomputer is to
Learning microprocessors for an engineer and for a technologist requires two different approaches. The approach for an engineer is to design, invent, and improve the state-of-the-art technology. But for the technologist, the approach is to apply, implement, and manage such technology. From an educational point of view, the program in teaching microprocessors, particularly for the Industrial Technologist, needs to provide students with hands-on practical experience.

However, the microcomputer is designed as a general purpose machine and the microprocessor chip cannot function independently. It must be supported with several peripheral devices (Zaks & Wolfe, 1987). Therefore, the standard way of teaching students about microprocessor concepts in the technology fields is to purchase some kind of trainer as a teaching tool or to use a microcomputer and design a instructional suitable interface. Normally, the trainers are expensive, have certain limitations, and cannot give students who major in Technology the overall understanding of controlling concepts. If the microcomputer is chosen, building a suitable interface for it is time consuming and costly too. Also, different trainers or microcomputers do provide completely different specifications. When interfacing, developing communications, and timing designs for a project, students need to implement a lot of decoding and wiring (Zaks & Wolfe). In
addition to the complexity of implementing projects, several kinds of other supportive chips are still needed. In consideration of time, economy, and efficiency, choosing a trainer or microcomputer as the teaching tool is not the best approach for instructing technology students. In contrast, if industrial educators choose to teach the students to design their own microprocessor-based systems without using the trainers or microcomputers, this approach is more appropriate for engineers rather than for technologists.

Fortunately, Motorola has developed, a single chip 8-bit microcontroller that contains almost all the necessary devices for a microprocessor based control system. This microcontroller chip (MC68HC11) has memories (ROM, RAM, EEPROM), Input/output pins, ADC, UART, and Timers (MC68HC11 EVB user's manual, 1986; MC68HC11 reference manual, 1989). The MC68HC11 itself could be called a single chip microcomputer that can perform several simple tasks. The MC68HC11 is one of the well-designed powerful 8-bit machines, easily taught and learned (Lipovski, 1988). In the industrial applications of the MC68HC11, there are several examples, such as GM automobile control, SONY video camera, hard disk drive controller, grocery scanning machine, etc., where these chips are widely used.

In addition to those mentioned above, Motorola has just marketed its own design modules for the MC68HC11. These modules are called EVB (Evaluation Board) and EVM (Evaluation Module) which are the ideal tools for teaching students about the
microprocessor applications concepts (Motorola Inc., 1989a). However, there have not been many published articles describing applications concerning EVB and EVM, due to their recent appearance in the market, so that technology major students are unable to see the real design/application usage of the EVB and EVM. The purpose of this research was to design, develop, and test sample explanatory experiments for Industrial Technology programs. For keeping students current in new microprocessor technology, the proposed experiments using the Motorola MC68HC11 EVB within the programs are considered essential.

**Problem of the Study**

The problem of this study was to design, develop, and test seven sample explanatory experiments, by using Motorola MC68HC11 microcontroller EVB to improve the effectiveness, efficiency, and to update curriculum content about microprocessor applications in Industrial Technology programs.

**Purpose of the Study**

The purpose of this study is:

1. To develop seven pilot or demonstration experiments that can be useful in teaching microprocessor applications in
Industrial Technology programs.

2. To explore the MC68HC11 microcontroller state-of-the-art contents that are applicable in the technical education curriculum.

3. To evaluate the effectiveness of applying the high technology concepts into experimental laboratory activities.

Objectives of the Study

This research, which identified a plan to design, develop, and test the explanatory experiments that were implemented by the MC68HC11 EVB for Industrial Technology programs, had the following objectives:

1. Ascertained the content that is necessary for teaching the students microprocessor applications.

2. Select and develop sample application projects by using MC68HC11 EVB.

3. Develop teaching materials for the implementation and application of hardware and software in microprocessor applications.

4. Set-up the apparatus and trial-test the experiments with sample of technology students.

5. Appraise the effectiveness in learning of the explanatory experiments using MC68HC11 EVB.
Assumptions of the Study

This study was based on the following assumptions:

1. Microprocessor control applications are an important concept for students majoring in Industrial Technology with the concentration of electrical controls and manufacturing.
2. The students who were involved in this study have already had the basic linear and digital electronics theory background.
3. The students had already learned the basic concepts of programming skills (such as BASIC or PASCAL or C languages).
4. The procedure for selecting the research subjects was valid and the results can be generalized to the general population.
5. Any uncontrolled variables of the study were uniformly distributed over the entire sample.

Delimitations of the Study

This study is limited to:

1. The available equipment in the Department of Industry and Technology at the California University of Pennsylvania.
2. The microprocessor controls related to teaching content in the Industrial Technology program which has functions similar to those found in industrial applications.
3. The students who are majoring in Industrial Technology, Electronic Technology, Electrical Engineering Technology at the California University of Pennsylvania.

4. The uncontrollable interactions between experimental and control group subjects.

**Procedures of the Study**

The method of this study consisted of the following steps:

1. Reviewed the related literature and technology concerning the 8-bit microprocessors and microcontroller applications.

2. Reviewed the current laboratory activities of the microprocessor curriculum in Industrial Technology/Electrical Engineering Technology/Electronic Technology programs.

3. Identified sample explanatory experiments of the microprocessors control applications.

4. Developed the achievement test instrument.

5. Developed seven experimental projects by using Motorola MC68HC11 EVB.

6. Administered the pretest.

7. Field-tested the explanatory experiments with students.

8. Administered the posttest.

9. Gathered and analyzed the data.
10. Provided research documentation of the experimental projects and written summary, conclusions, and recommendations.

**Definition of Terms**

**ADC:** Analog to digital conversion, a process of converting analog to digital signals, typically by reading the analog signal of voltage or current and converts to its digital quantity which is proportional to the amplitude of the input (Cahill, 1987).

**Decoding:** implementing of a digital circuit that can take an n-bit number as input and uses it to select exactly one of the 2^n output line. This technique is essential for the microprocessor to recognize the outside world signals (Tanenbaum, 1984).

**EEPROM:** the Electrically Erasable Programmable Read Only Memory. It is electrically erasable by the high pulse voltage; so, information can be directly and dynamically read/write electrically under program control (Prince and Gundersen 1983).

**EVB:** MC68HC11 Evaluation Board, a Motorola product which contains the MC68HC11 (microcontroller), RS232 communication ports, and monitor program encoded in the
memory called "BUFFALO". This is a good design tool by implementing the 8-bit microcontroller.

EVM: MC68HC11 Evaluation Module, also a Motorola product which holds all the features that EVB has, and the EEPROM microcontroller programmer sockets.

High/low level language: computer programming languages, low level refers to the Assembly language or machine codes, high level refers to BASIC, PASCAL, FORTRAN, PL1, C etc., but C usually considered as a computer language lies in-between high and low levels.

Input/Output: the microprocessor communication device which the microprocessor can talk with outside world and exchange the digital information.

Interfacing: A device which accepts signals from one device and converts them to a form acceptable by another. When referring to the microprocessor, the interfacing involves the decoding, buffering of the memory signal and data signal which can be recognized by the microprocessor.

MC68HC11: Motorola product of a microcontroller whose function as a stand-alone single-chip microcomputer and as a microprocessor in a multiple-chip microcomputer, such as a personal computer or a multiple-board computer (Lipovski).

Microcontroller: a single-chip microcomputer or called as microcomputer unit (MCU) as opposed to microprocessor, called as microprocessor unit (MPU), includes memories,
input/output, and other peripheral devices on a single chip (Slater, 1989).

Microprocessor: an integrate circuit component which implements the functions of an arithmetic-logic-unit and some logic control unit in a single chip. The basic microprocessor can be classified as 4-bit, 8-bit, 16-bit, and 32-bit logic processing machines (Zaks, 1977).

Module: An assembled printed circuit with a self contained operating program can perform a variety of functions (ERA, 1979).

RAM: Random Access Memory, primary or main memory which can be read/write. It is nonvolatile memories as long as power is applied (Cheung & Bredeson, 1990).

ROM: Read Only Memory, primary or main memory which can only be read, indicating that information is programmed by the mask at fabrication time. The information thus encoded cannot be changed at all (Cheung & Bredeson).

State-of-the-Art: The current high technology concept which is a changing subject depending on the new technology invention.

Trainer: A microprocessor teaching instrument which is a pre-design prototype machine, it usually provides monitor program and input/output devices, and can only accept hexadecimal machine codes.
UART: Universal Asynchronous Receiver Transmitter, a serial standard asynchronous formats for computer communications (Cahill).
CHAPTER II. REVIEW OF LITERATURE

Before developing the explanatory experiments for microcontroller applications, the related literature which has been previously published by researchers and suggested for further study is worth investigation. The relevant literature review has been arranged into seven categories, listed as follows: (1) microprocessor high technology impact, (2) historical development of microprocessors, (3) the 8 bit microprocessor world, (4) microprocessor curriculum efforts, (5) differences between microprocessors and microcontrollers, (6) general description of microcontroller 68HC11 EVB and EVM, and (7) summary of literature review.

Microprocessor High Technology Impact

The emergence of working transistors can be traced to the period following World War II. Following the transistors in the late 1950's and early 1960's, the integrated circuit or I.C. was developed. In 1970, the first microprocessors chip was produced (Connecticut, 1986).

The microprocessor's flexibility as a result of being controlled by a program rather than hard-wired logic, its lower
price compared with the available alternatives, and the reduced convenient size, all these characteristics make the microprocessor-based applications almost endless. During the past two decades, numerous significant technological changes have taken place in the industrial workplace as well as our daily living activities. The most recognizable facts among the industrial workplace are the use of microprocessor-based computers, computer-aided design and drafting, and computer-aided manufacturing. The applications in appliances such as microwave oven, washing machine, air-conditioner, automobile, telephone services, fax machine, and the affordable/powerful personal computer are common devices in our daily life.

In the manufacturing process, industrial robots have been in the workplace since the early 1960, and today they have become an integral part of automated systems within existing manufacturing processes. The coupling of microprocessors and computers to robots has made the modern application of robotics possible. This introduction of high technology into the workplace is being referred to as the dawning of the second industrial revolution (Louisiana, 1987).

With the technology advancement, the cost of computer logic devices is falling at the rate of 25 percent per year and also the cost of computer memory at the rate of 40 percent per year. Computational speed has increased by a factor of 200 in twenty-five years. In the same period the cost, the energy consumption, and size of computers of comparable power have
decreased by a factor of 10,000 (Toong & Gupta, 1982). We, now, have the opportunity to create microprocessor-based devices of greater power at a cost so low that we can afford to use them to control industry automation process, the engine of a car, washing machine cycles in the home, a robot performances in the factory, or on the battlefield.

As the high technology continues to expand and gain entry into every facet of human existence, it is inevitable that all of us are going to use microprocessors in our everyday work. This inevitability is not something to worry about. In fact, it is something to look forward in our future as we will stand to gain by using microprocessors (Standing, 1982). Nowadays, the microprocessor has made it possible to produce extremely powerful computers on a single silicon chip or the combination of chips. The microprocessor high technology has brought about great changes in our society. It induced dramatic cost reduction and created a wide range of industrial applications. This diversity of applications for microprocessor technology has led futurists to predict a social transformation comparable with the agriculture revolution that began about 10,000 years ago, and with the industrial revolution (Levitan & Johnson, 1982). In the near future, the microprocessor technology will be even more powerful, making the machines totally automated and possessing their own thinking capabilities.

These automation and society transformations definitely produce dislocations and unemployment. The resistance to change
is a major factor influencing acceptance of new technology. As Rees (1973) stated, "Union resistance to labor-saving technical change within an industry can often be moderated by careful management of change, which will minimize its effect in creating unemployment" (p. 137). Technology advances are inevitable. Obviously, there are more and more jobs being done by machine, which leads to unemployment in one sense but creates new opportunities in other fields. As a consequence, there will be many more jobs created by all the new technical possibilities (Cuthbert, 1981). The proper management of change is the best policy for society to face these changes in formats as longer training and extended education, more re-training and re-education, shorter work weeks, more vacation time, earlier retirement, welfare, and unemployment payments become viable options (Levitan & Johnson, 1982).

**Historical Development of Microprocessors**

The microprocessor resulted from a merging of integrated circuit (I.C.) technology and computer architecture. The I.C. technology plays a dominant role in the development of microprocessors. Integrated circuits are commonly divided into classes, which are based on the number of transistors per chip (Slater, 1989):
Small-Scale Integration (SSI): 1 to 100 transistors per chip

Medium-Scale Integration (MSI): 100 to 1,000 transistors per chip

Large-Scale Integration (LSI): 1,000 to 10,000 transistors per chip

Very Large Scale Integration (VLSI): 10,000 to 100,000 transistors per chip

Super Large Scale Integration (SLSI): over 100,000 transistors per chip

The first invention of a commercial microprocessor with the LSI technology was introduced in late 1971 by Intel. Named the 4004, it has integrated 2,300 transistors on a single silicon chip (Cahill, 1987). In 1972, Intel introduced the first general purpose 8 bit microprocessor, called the 8008. Two years later, the second generation 8 bit microprocessor, Intel 8080, used 5,000 transistors in its fabrication was put on the market as the successor to the 8008. Within these two years, all the main standard 8 bit microprocessors were introduced, most of them inspired by the early design of the 8008.

After Intel 8080's introduction, the 8 bit microprocessor continued to be refined through the decade. In 1974 (shortly after Intel 8080), Motorola introduced its first microprocessor 6800 to join the market competition. Similar microprocessors,
such as Rockwell the 6500, Signetic the 2650, and Commodore the 6502, were also introduced during the same period (Zaks & Wolfe, 1987).

The third generation of 8 bit microprocessors, the successors to 8080 and 6800 after 1974, were Z80 from Zilog, 8085 from Intel, and 6809 from Motorola. In 1977, the 16 bit microprocessors had been put on the market, these were Intel 8086, Motorola 68000, Zilog Z8000, Texas Instrument 9900, etc.

The current microprocessor design generation began in 1983 with the introduction of the first 32 bit microprocessor, National Semiconductor's 32032. This was followed by Intel 80386, Motorola 68020, and Zilog Z80000.

The overall historical evolution of the microprocessors' designs can be summarized in table 1.

Motorola in particular, its microprocessor family evolution which is shown in Figure 1, provides us a general idea of the microprocessor developing trend (Motorola, 1990).

The 8 Bit Microprocessor World

Early in 1974, the 8 bit microprocessor had been introduced on the market. The reason why an 8 bit microprocessor can still coexist with a 16 bit microprocessor or even a 32 bit microprocessor today is because of the 8 bit designs that have more than adequate power for many applications and the advantage
Table 1. Historical development of microprocessors

<table>
<thead>
<tr>
<th>YEAR</th>
<th>COMPANY</th>
<th>MODEL</th>
<th>BIT-SIZE</th>
<th>I.C.-TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1971</td>
<td>INTEL</td>
<td>4004</td>
<td>4</td>
<td>LSI</td>
</tr>
<tr>
<td>1971</td>
<td>NATIONAL SEMICONDUCTOR</td>
<td>COP400</td>
<td>4</td>
<td>LSI</td>
</tr>
<tr>
<td>1971</td>
<td>NEC</td>
<td>uPD75XX</td>
<td>4</td>
<td>LSI</td>
</tr>
<tr>
<td>1972</td>
<td>INTEL</td>
<td>8008</td>
<td>8</td>
<td>LSI</td>
</tr>
<tr>
<td>1974</td>
<td>INTEL</td>
<td>8080</td>
<td>8</td>
<td>LSI</td>
</tr>
<tr>
<td>1974</td>
<td>MOTOROLA</td>
<td>6800</td>
<td>8</td>
<td>LSI</td>
</tr>
<tr>
<td>1974</td>
<td>RCA</td>
<td>CDP1800</td>
<td>8</td>
<td>LSI</td>
</tr>
<tr>
<td>1974</td>
<td>ROCKWELL</td>
<td>R6500</td>
<td>8</td>
<td>LSI</td>
</tr>
<tr>
<td>1975</td>
<td>COMMODORE</td>
<td>6502</td>
<td>8</td>
<td>LSI</td>
</tr>
<tr>
<td>1975</td>
<td>SEGNETIC</td>
<td>2650</td>
<td>8</td>
<td>LSI</td>
</tr>
<tr>
<td>1975</td>
<td>ZILOG</td>
<td>280</td>
<td>8</td>
<td>LSI</td>
</tr>
<tr>
<td>1975</td>
<td>FAIRCHILD</td>
<td>F8</td>
<td>8</td>
<td>LSI</td>
</tr>
<tr>
<td>1975</td>
<td>INTEL</td>
<td>8085</td>
<td>8</td>
<td>LSI</td>
</tr>
<tr>
<td>1979</td>
<td>MOTOROLA</td>
<td>6809</td>
<td>8</td>
<td>LSI</td>
</tr>
<tr>
<td>1977</td>
<td>TEXAS INSTRUMENTS</td>
<td>9900</td>
<td>16</td>
<td>VLSI</td>
</tr>
<tr>
<td>1978</td>
<td>INTEL</td>
<td>8086</td>
<td>16</td>
<td>VLSI</td>
</tr>
<tr>
<td>1978</td>
<td>ZILOG</td>
<td>28000</td>
<td>16</td>
<td>VLSI</td>
</tr>
<tr>
<td>1979</td>
<td>MOTOROLA</td>
<td>68000</td>
<td>16</td>
<td>VLSI</td>
</tr>
<tr>
<td>1983</td>
<td>NATIONAL SEMICONDUCTOR</td>
<td>32032</td>
<td>32</td>
<td>VLSI</td>
</tr>
<tr>
<td>1984</td>
<td>INTEL</td>
<td>80386</td>
<td>32</td>
<td>VLSI</td>
</tr>
</tbody>
</table>
Table 1. (continued)

<table>
<thead>
<tr>
<th>YEAR</th>
<th>COMPANY</th>
<th>MODEL</th>
<th>BIT-SIZE</th>
<th>I.C.-TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1984</td>
<td>MOTOROLA</td>
<td>68020</td>
<td>32</td>
<td>VLSI</td>
</tr>
<tr>
<td>1986</td>
<td>ZILOG</td>
<td>280000</td>
<td>32</td>
<td>VLSI</td>
</tr>
<tr>
<td>1987</td>
<td>MOTOROLA</td>
<td>68030</td>
<td>32</td>
<td>VLSI</td>
</tr>
<tr>
<td>1989</td>
<td>MOTOROLA</td>
<td>68040</td>
<td>32</td>
<td>SLSI</td>
</tr>
<tr>
<td>1989</td>
<td>INTEL</td>
<td>80486</td>
<td>32</td>
<td>SLSI</td>
</tr>
</tbody>
</table>

of the lower cost (Zaks & Wolfe, 1987). Due to the flexibility, complexity, applicability, popularity, and cost consideration, only the 8 bit microprocessor will be used in this experimental project. Accordingly, the related features of various 8 bit microprocessors were reviewed and compared.

**Intel**

It should be noted that 8008 was the first 8 bit microprocessor produced by Intel -- a rather difficult processor to use. It requires a large number of standard logic circuits to make it work and it is rather elementary in its instruction (ERA, 1979). The 8080 was designed as a successor to 8008 and was the first powerful microprocessor introduced on the market. It was also the first relatively convenient single chip microprocessor to use in numerous applications (Zaks & Wolfe, 1987). A later version 8085 accounts for roughly one of every four 8 bit microprocessors sold, which is a slightly improved
Figure 1. Motorola microprocessor family evolution
design of 8080.

**Motorola**

The 6800 was introduced as a direct competitor to the 8080, which in many ways is similar to the 8080. However, the 6802 is a new version of the 6800, which contains an internal clock generator and 128 bytes of RAM. The 6809 was designed for high performance. It has 16 bit internal buses and can be considered a device between 8 bit and truly 16 bit microprocessors. The 6800/6809 family was the third most popular 8 bit microprocessor family after the Z80 and the 8080/8085 families.

**Zilog**

"Three of the designers of the 8080 left Intel and created their own company, Zilog, in Los Altos. Zilog is an affiliate of Exxon, the oil corporation" (Zaks & Wolfe, 1987, p. 212). So, the instruction set of Z80 contains all those of the 8080 and some additional ones which make it more powerful. It should be noted that the 8080 programs can run in a Z80 system without modification. Z80 is the second most popular 8 bit microprocessor which was sold roughly one in every three devices.

**Commodore**

The 6502 is one of the most popular 8 bit microprocessor for use in video games and low-cost early home computers. "The 6502 was designed by former Motorola engineers who were a part of the 6800 design team. As a result of this background, the 6502 has much in common with the 6800. While it is faster, its
bus organization, internal registers, and instruction set are all very close to 6800" (Zaks & Wolfe, 1987, p. 211).  

**National Semiconductor**

NSC800, a Z80 compatible microprocessor acts as CPU and works in conjunction with a line of 8085 compatible peripheral chips. The instructions are very basic and are not very efficient in terms of execution time or memory requirement (ERA, 1979).

**Fairchild**

The F8 was introduced as a two chip microcomputer, with the partitioning onto the two chips that the ROM was basically on one chip and everything else on the other. It offers a complete microcomputer on two chips and does not need many additional hardware devices to make a working system (ERA, 1979). So, the F8 can be considered as the first microcontroller implementation device.

The detailed comparison of those 8 bit microprocessors mentioned above can be summarized in Table 2 as presented by Zaks and Wolfe (1987, p. 203-204).

**Microprocessor Curriculum Efforts**

It is clear that we are in the century of computer revolution or information revolution. The computer has changed the face of business and industry both large and small. The
Table 2. Comparison among 8 bit microprocessors

<table>
<thead>
<tr>
<th>Company</th>
<th>Model</th>
<th>Commodore 6502</th>
<th>Fairchild F8</th>
<th>Intel 8080</th>
<th>Intel 8085</th>
<th>Motorola 6800</th>
<th>Motorola 6809</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technology</td>
<td>NMOS/CMOS</td>
<td>NMOS</td>
<td>NMOS</td>
<td>NMOS</td>
<td>NMOS</td>
<td>NMOS</td>
<td>NMOS</td>
</tr>
<tr>
<td>No. of Instructions</td>
<td>57/68</td>
<td>70</td>
<td>69</td>
<td>71</td>
<td>71</td>
<td>71+</td>
<td>8+8</td>
</tr>
<tr>
<td>Cycle Time (μS)</td>
<td>1.5/3.0</td>
<td>2-13</td>
<td>1.3,1.5,</td>
<td>2.0</td>
<td>1.3</td>
<td>2.5</td>
<td>1.3</td>
</tr>
<tr>
<td>Direct Addr. (Bits)</td>
<td>12-16</td>
<td>16</td>
<td>16</td>
<td>8+8</td>
<td>16</td>
<td>8+8</td>
<td>8+8</td>
</tr>
<tr>
<td>Registers</td>
<td>3</td>
<td>1+RAM</td>
<td>7</td>
<td>7</td>
<td>4</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>Stacks</td>
<td>Soft</td>
<td>External</td>
<td>Soft</td>
<td>Soft</td>
<td>Soft</td>
<td>Soft</td>
<td></td>
</tr>
<tr>
<td>Interrupts</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>On-Chip Clock</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>ROM</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>RAM</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Timer</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>I/O Lines</td>
<td>N</td>
<td>16</td>
<td>N</td>
<td>2</td>
<td>N</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>PK(Pins)</td>
<td>28/40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>Power(V)</td>
<td>5</td>
<td>5,12</td>
<td>5, -5,12</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>
Table 2. (continued)

<table>
<thead>
<tr>
<th>Company Model</th>
<th>National NSC800</th>
<th>National 32008</th>
<th>Zilog Z80</th>
<th>Zilog Z280</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Standard</td>
<td>Standard</td>
<td>Standard</td>
<td>Standard</td>
</tr>
<tr>
<td>Technology</td>
<td>CMOS</td>
<td>NMOS</td>
<td>CMOS, NMOS</td>
<td>CMOS</td>
</tr>
<tr>
<td>No. of Instructions</td>
<td>158</td>
<td>170</td>
<td>69+</td>
<td>N</td>
</tr>
<tr>
<td>Cycle Time (uS)</td>
<td>1.6, 1.0</td>
<td>0.1, 0.17, 0.3</td>
<td>1.3, 1.6</td>
<td>0.1</td>
</tr>
<tr>
<td>Direct Addr. (Bits)</td>
<td>16</td>
<td>24</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>Registers</td>
<td>22</td>
<td>8</td>
<td>17</td>
<td>N</td>
</tr>
<tr>
<td>Stacks</td>
<td>Soft</td>
<td>Soft</td>
<td>Soft</td>
<td>Soft</td>
</tr>
<tr>
<td>Interrupts</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>N</td>
</tr>
<tr>
<td>On-Chip Clock</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>ROM</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>RAM</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Timer</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>I/O Lines</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>PK(Pins)</td>
<td>40</td>
<td>48</td>
<td>40</td>
<td>68</td>
</tr>
<tr>
<td>Power (V)</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>
advent of the microelectronics has pushed the microcomputer applications to even more astonished levels, and the impact of the changes almost reaches the point where what we did yesterday is already obsolete (Goettmann, 1984). It is the microprocessor and the associated products of very large scale integration (VLSI) or super large scale integration (SLSI) of electronic circuit which make the biggest impact on our living (Summers, 1982).

Accompanying the variety of applications using microprocessor based devices in today's society is a fast growing demand for trained personnel to manage, build, test, or service this equipments (Pasahow, 1981). Considering the importance of the microprocessor to Industrial Technology, Oscarson (1982) stated "for technical and non-technical managers and instructional personnel, the advent of the microprocessor provided new dimensions for training, development, and decision-making" (p. 30). The above statement can reveal the strength. Industrial technology educators, who are responsible for these trained personnel, should understand the promise of the microprocessor and the microcomputer; moreover, they can play a key role in orienting their programs to this challenging time in our society.

In analyzing the current status of microprocessor curriculum efforts, it appears that many regions in the United States are ready to begin the exploration of the microprocessor technology. The Connecticut State Department of Education has
developed a set of microprocessor curriculum guides which are listed as follows:

Microprocessors

A. Concepts

1. Definition

2. Role of the microprocessor in relation to the computer

3. Types of microprocessors
   a. Types of busses
   b. Information handling capabilities
   c. Series/Classification

B. Goals

1. Computer Architecture

2. System Architecture

3. Storage

4. Processing

5. I/O Communication

6. Digital Logic

7. Microcomputer Interfacing
   a. Cabling and connectors
   b. Modems
   c. Series and parallel paths

8. Applications
   a. All computers of every type
   b. Robotics
   c. CNC machines
d. Automotive systems
e. Military systems
f. Photography
g. Environmental control

9. Hardware
a. Chip types
b. Support chips
c. Microprocessor trainers

(Connecticut, 1986, p. 64-73)

Also, the Louisiana State Department of Education produced the microprocessor and robotics curriculum development bulletin 1803, whose circulation includes the central states of North Dakota, South Dakota, Nebraska, Iowa, Colorado, Kansas, Missouri, Oklahoma, Arizona, Louisiana, and Texas. This bulletin lists competency based courses of microprocessor as follows:

1. Understand the basic architecture of a hypothetical microprocessor.
2. Write a simple program to control the operation of the microprocessor.
3. Understand the structure of basic memory systems available to the microprocessor.

The courses outlines are:

I. Hypothetical 8 bit microprocessor.
II. Programming Basics
III. Memory Systems

(Louisiana, 1987, p. 57-84)

In Texas, the education agency in Austin produced a list of instructional units of microprocessor and interfacing which is based on performance objectives. These units are listed as follows:

Microprocessors and Interfacing

1. Microprocessor Architecture
2. Input/Output
3. Standard Interfaces
4. Analog/Digital Interfaces and Peripheral Equipment
5. Troubleshooting Microprocessors

(Texas, 1984, p. 850-1011)

In a manual of the curriculum consortium for microprocessor training of service technicians written by Brown and Fulkerson, the authors provide the following training guides:

A. Advanced Microcomputer Service Technician

1. Fundamentals of Microprocessors
2. System Architecture
3. Microprocessor Architecture and Timing
4. Tools and Test Equipment
5. Busses, Protocols, and Handshakes
6. Generic Troubleshooting

B. Equipment: Microprocessor trainers
(Brown & Fulkerson, 1985)

The Mississippi State Department of Education also provide the curriculum guide on microprocessor courses. This curriculum guide to the teaching of microprocessor courses gives the following outline:

Introduction to Microprocessors

1. Architecture
2. Language
   a. Binary
   b. Machine
   c. Basic
   d. Others
3. Memory
   a. ROM (Read Only Memory)
      (1) PROM (Programmable Read Only Memory)
      (2) EPROM (Erasable Programmable Read Only Memory)
   b. RAM (Random Access Memory)
4. Applications of Microprocessors
(Mississippi, 1983, p. 37-40)

In an article entitled "Microprocessors and School Physics", the author Cuthbert (1981) describes what skills the candidates should process after the completion of studying the microprocessor system.

Microprocessor Systems

1. Draw a block diagram of a microprocessor system
consisting of the following blocks:

a. Processor
b. Clock
c. Memory input/output interface

2. Describe the function of the following types or peripheral:

a. Teletype
b. Alphanumeric video cassette tape recorder as the simplest form of backing store.

3. Describe the differences in use between the following type of memory:

a. RAM
b. ROM
c. PROM
d. EPROM

4. Explain how the

a. Data bus
b. Address bus
c. Read/write line

are used in a microprocessor

Regarding the necessary equipment, he lists three choices:

1. Build a system from integrated circuits.
2. Buy a microcomputer and build a suitable interface to the system being controlled.
3. Buy a microprocessor trainer.
Morris (1981), in his article "Microelectronics in Scottish Education", states a general idea of how the microelectronics should be implemented in the education system. The stated objectives are:

1. Creating awareness of microelectronics in society at large.
2. The use of microcomputers in school and college administration.
3. Production of curricular materials and learning packages to be run on microcomputers.
4. Programs for assessment.
5. Learning materials and control systems to aid all forms of handicap.

In Summers's (1982) two consecutive articles, entitled "A Simple Microprocessor Breadboard System for Use in Microelectronics or Computer Education-Part 1 & Part 2", he presents a description and sample circuit/program for building a microprocessor based system. This presentation shows the details of how to design a basic working system which is a different approach for the teaching of microprocessors.
Differences between Microprocessors and Microcontrollers

Cuthbert (1981) defined the term microprocessor as: "an integrated circuit that acts as the main control and processing device in a microcomputer. The microprocessor is not a computer in its own right and needs other integrated circuits to enable it to perform any useful function" (p. 136). Wylie and Takushi (1980) gave an even more broad description of a microprocessor -- "any device that utilizes digital technology and microelectronics to process (alter, combine, analyze, store and retrieve, and so forth) information in a specific, basically unalterable way" (Wylie & Takushi, 1980, p. 88). A microprocessor itself is just an important element of a workable system; on the other hand, it can not function independently. If the microprocessor must communicate/control other devices in parallel format, it needs some parallel peripheral interface chips. If serial communication/control is required, the microprocessor has to have some serial interface chips to fulfill the objectives. Furthermore, storage of control criteria, sequential timing events, and digital-to-analog conversion (DAC)/analog-to-digital conversion (ADC) will all need some special peripheral devices such as RAM/ROM, timer, DAC/ADC chips to support the microprocessor to accomplish the required objectives of a system performance. Therefore, a microprocessor cannot form a workable control system without the help of other peripheral devices.
Unlike microprocessors, which are designed for a broad range of applications, microcontrollers are generally designed with a specific application in mind. A microcontroller has all the peripheral features needed to implement the computer portion of an embedded application (Vaglica & Gilmour, 1990). Usually, the microcontroller is also called a single-chip microcomputer, and includes a microprocessor, memory, and I/O facilities on the same chip (Slater, 1989). This device has a historical development that is similar to that of the microprocessor. The year, manufacturer, and model of microcontrollers along with bit-size are listed in Table 3.

Slater's (1989) statement about microprocessors and microcontrollers family tree can give us an overall picture of the comprehensive development in the last two decades. The family tree is listed in Figure 2 (p. 47).

General Description of Microcontroller
68HC11 EVB and EVM

The experimental project makes use of the Motorola microcontroller 68HC11 EVB. The related features of these devices justify some discussions. The Motorola MC68HC11 is a single chip microcomputer, called a MCU (Microcomputer Unit); sometimes, it is also called a Microcontroller. It processes all the features that the Microprocessor can have and all the interfacing functions that a general microprocessor
cannot perform without the help of other peripheral devices. Figure 2 and 3 illustrate the major differences between microprocessor and microcontroller (Zaks & Wolfe, 1987).

Table 3. Historical development of microcontrollers

<table>
<thead>
<tr>
<th>YEAR</th>
<th>COMPANY</th>
<th>MODEL</th>
<th>BIT-SIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1976</td>
<td>Intel</td>
<td>8048/8049</td>
<td>8</td>
</tr>
<tr>
<td>1979</td>
<td>Zilog</td>
<td>28</td>
<td>8</td>
</tr>
<tr>
<td>1980</td>
<td>Motorola</td>
<td>6805</td>
<td>8</td>
</tr>
<tr>
<td>1981</td>
<td>Intel</td>
<td>8051</td>
<td>8</td>
</tr>
<tr>
<td>1982</td>
<td>Intel</td>
<td>8096</td>
<td>16</td>
</tr>
<tr>
<td>1984</td>
<td>Intel</td>
<td>8052</td>
<td>16</td>
</tr>
<tr>
<td>1985</td>
<td>Motorola</td>
<td>68HC11</td>
<td>8</td>
</tr>
<tr>
<td>1988</td>
<td>Motorola</td>
<td>68332</td>
<td>32</td>
</tr>
<tr>
<td>1990</td>
<td>Motorola</td>
<td>68HC16</td>
<td>16</td>
</tr>
</tbody>
</table>

In addition to the MC68HC11 microcontroller chip, Motorola also developed the EVB (Evaluation Board) and EVM (Evaluation Module) system for the MC68HC11 chip. The EVB has a "BUFFALO" (Bit Users Fast Friendly Aid to Logic Operation) monitor program that has burned into an EPROM and takes care of all the necessary communication protocol. This enables the EVB to communicate with any host computer (Macintosh or IBMPC) or network terminal by using the public domain software called...
"Kermit" through its RS232 ports. The EVB has 60 lines for its control interfacing and two RS232 ports for its host computer or terminal communications. Having these features, one can design his/her hardware, when connected to the 60 lines of the EVB, and write the control software accordingly through the host computer or terminal (Motorola Inc., 1986b). He/She can use Motorola free cross assembler, running on the host computer or terminal, to generate the MC68HC11 machine codes. Later on, these machine codes could be downloaded into the EVB RAM or EEPROM to perform the actual design control functions.

The EVB provides the flexibility, efficiency, economy, and simplicity for the designing job of the microprocessor control applications. Similar to the EVB, the EVM provides two additional sockets (one for the DIP (dual in line package), one for the PLCC (plastic leaded chip carrier) package) for putting the workable version of the control program into the other MC68HC11 chips (Motorola Inc., 1989b). This feature can readily match the demand of real world designs. The actual application procedures are: first, one can test his/her control program on the module; then, second, one can put his/her error-free control program into another MC68HC11 chip and take this chip to wherever the design criteria are needed to preform the desired functions.
Figure 2. Family tree of microprocessors and microcontrollers
Figure 3. The standard microprocessor

Figure 4. The 68HC11 microcontroller
Summary of Literature Review

Microprocessor and microelectronic technology permit the capability of a large computer to be concentrated into a small space, with the resultant advantage of using less energy, producing less heat, needing seldom elaborate air conditioning, and costing extremely less. They enable us to do what we already did more quickly, more inexpensively, and more accurately (Cherns, 1980). With microprocessors dominating modern digital applications, it has introduced our society to the computer age. The potential impact of microprocessors is heightened by their seemingly endless number of applications. Therefore, the career prospect of the youth will definitely be influenced by this impact (Levitan & Johnson, 1982).

The introduction of the first commercial microprocessor was made by Intel's 4004 in 1971. After the 4004, there are National Semiconductor's COP400 and NEC's uPD75xx series, which are all classified as 4 bit microprocessors used in calculator applications. Between 1974 - 1979, this period was considered as 8-bit microprocessor years and the microprocessor's standard was set in that period. The companies involved in 8 bit microprocessors development were Intel 8008, 8080, Motorola 6800, 6802, 6809, Commodore 6502, Rockwell R6500, RCA CDP1800, and Fairchild F8. After 1977, the 16 bit microprocessor development started with the higher microelectronics technology called VLSI. The leading companies in this 16-bit
microprocessor business were Intel 8086, 80286, Motorola 68000, 68020, Zilog Z8000. Beginning from 1983, the microprocessor development got into 32 bit which makes the microprocessor more powerful, faster, and more efficient. Intel 80386, Motorola 68030, and Zilog Z80000 were the representatives. In 1989, Motorola and Intel had just introduced their most recent products of 68040 and 80486 respectively. The in-coming designs and applications in hardware and software of these new products are envisioned.

The 8-bit microprocessor is flexible, powerful, and easy to manipulate, and is able to perform all the basic microprocessor based control functions. It is difficult to determine which manufacture's microprocessor is the best. The choice is dependent on the user's preference, project requirements, and availability of the devices (microprocessor itself and peripheral chips).

The curricula which have been reviewed in this document serve as general guide lines. They only specified what content path to follow. The detailed implementation of the lab or project activities is still missing. In Summers (1982) presentation, the ideal condition is far different from those aforementioned guidelines about the microprocessor, but even Summers proposal is still not suitable for the practical approach to Industrial Technology. Besides, the comparison of the industry's implementation of the state-of-the-art know-how and the Industrial Technology curricula in microprocessors, the
latter is still far behind the needs of real world. The design and development of a series of practical lessons and programs, which can encourage and inspire the students by using the state-of-the-art technology, are an urgent task.

A microcontroller is actually a member of the microprocessor family. It has been designed with the latest microelectronic technology which has an embedded necessary memory, input/output, timer, communication, and ADC/DAC peripheral devices with a microprocessor into a single chip. These features give a microcontroller far more superior capabilities than a microprocessor in designing specific control functions. They make the design work easier, the space smaller, the implementation faster, and the cost lower.

Especially the Motorola microcontroller 68HC11 and its design support module EVB/EVM, which make the use of a microcontroller in various applications too hard to reject. Considering any aspect of these devices, it is apparent that they should be quickly adopted to our Industrial Technology programs. Continued research must be conducted to discover better results in students' achievements of learning about emerging new technologies and their capacity of being competent to use such technologies.
CHAPTER III. METHODOLOGY

This chapter describes the procedures used to conduct the study. The procedures can be divided into the following sections: (1) design and development of the seven explanatory laboratory experiments, (2) hypotheses of the study, (3) sampling subjects, (4) research design, (5) data collection instrument, and (6) statistical analysis of the data.

Design and Development of the Seven Explanatory Laboratory Experiments

The concept of the explanatory experiments is chosen according to the literature examined in the preceding chapter which provides information about the features of a microprocessor and the applications of a microprocessor. The choices were made on the basis of generality, commonality, and students' learning ability considerations. The electronic device used for the explanatory experiments was Motorola 68HC11 microcontroller EVB (Evaluation Board). The experimental concept can be applied to any Motorola 8 bit microprocessor chip, such as the Motorola 6800 or 6809 microprocessor in TRS-80 microcomputers which were used by the control group.
The seven explanatory experiments include: (1) stepper motors control, (2) stop lights with pedestrian interrupt control, (3) keypad debouncing encoding, (4) LCD display module control, (5) keypad and LCD display module control, (6) electronic keys security system, and (7) DC motor control.

The elements of each experiment include: (1) title, (2) objectives, (3) performance requirements, (4) equipment needed, (5) apparatus set-up (schematic or circuit diagram illustration), and (6) control program (flow chart illustration).

**Hypotheses of the Study**

The study tested the two null hypotheses which stated:

**H₀₁**: There is no significant learning achievement difference between the experimental group students who used the Motorola 68HC11 microcontroller EVB and the control group students who used the traditional microprocessor as their project design tool.

and

**H₀₂**: There is no significant difference in time required to complete the assigned projects between the two groups.
Sampling Subjects

The subjects who participated in this study were the students majoring in Industrial Technology, Electronic Technology, or Electrical Engineering Technology programs in the department of Industry and Technology at the California University of Pennsylvania, California, Pennsylvania. All of the students were either in their freshman (including the transfer students), sophomore, or junior level of their four-year program of study. Due to two students dropped out of the class, a total of twenty students were selected from the department of Industry and Technology at California University of Pennsylvania. They were divided into two groups with matched pairs based on the use of the pretest scores as the selection criteria. There were ten students in each group. All the sampling methods and procedures planned for this study were approved by the Human Subject Review Committee at Iowa State University. The approved document is located in Appendix A.

These two groups were named the experimental group and the control group. They were organized and treated as follows:

Experimental Group: Ten students used the Motorola 68HC11 microcontroller EVB as their project design device after having been taught the general microprocessor concepts and specific features of the 68HC11 microcontroller.
Control Group: Ten students used the Radio Shack TRS-80 microcomputer as their project design device after having been taught the general microprocessor concepts and specific features of the 6809 microprocessor.

Research Design

The investigation focused on the achievement test scores of the subjects who were exposed to the same lectures about the general microprocessors but used different devices to implement the design project experiments in the laboratory activities.

The research design in this study employed a "Pretest-Posttest Control-Group Design with Matching" by Borg and Gall (1989, p. 674). The steps in carrying out this study were as follows:

(1) Administered the pretest for measuring the dependent variables to all the subjects of their background knowledge.

(2) Assigned subjects to matched pairs on the basis of their pretest scores.

(3) Randomly assigned one member of each pair to the experimental group and the other to the control group.

(4) Exposed both groups to the series of lectures but let
the experimental group use the 68HC11 microcontroller and the control group used the 6809 microprocessor during their design and laboratory activities.

(5) Administered the posttest for collecting achievement data measuring the dependent variables to the experimental and control groups.

(6) Compared the performance of the experimental and control group on the posttests to ascertain if statistical significance were present.

(Borg & Gall, 1989)

This design can also be illustrated schematically as follows:

Experimental Group

\[ R \quad O \quad X_E \quad O \]

Control Group

\[ R \quad O \quad X_C \quad O \]

R represents the simple random selection of subjects
O represents the pretest and posttest of measuring the dependent variables.

\[ X_E \] represents the independent variable, experimental treatment of using 68HC11 microcontroller.

\[ X_C \] represents the independent variable, control treatment of using 6809 microprocessor.
Data Collection Instrument

The achievement test instrument was designed to measure how well the subjects had learned the applications in a microprocessor control. The knowledge and skills included:

1. Understanding the microprocessor structures and programming.
2. Understanding the potential available features of the microprocessor.
3. Applying the microprocessor to various practical control functions.
4. Designing the microprocessor in the useful automatic systems.

In order to control the extraneous variables, the pretest and posttest questionnaires were the same and each one was administered at the same time to the two group of subjects. The achievement test instruments are located in Appendix B.

Statistical Analysis of the Data

To assure the initial equivalence of the experimental and control group, the random assignment of subjects to groups and the matching criteria were used. The analysis of covariance was used to further compensate and adjust the initial differences
between groups. The following procedures were used to analyze
the data in this study:

1. An analysis of variance using pretest/posttest scores
as the dependent variable was used to distinguish the
significant differences on the pretest/posttest scores
of the two groups.

2. An analysis of covariance using the pretest as the
covariate and the posttest as the dependent
variable were used to find whether significant
differences on the posttest scores of the two
groups existed or not.

3. An analysis of covariance using the pretest as the
covariate and the posttest as the dependent variable
were used to find significant differences of the
interaction between the two groups and pretest scores.

4. An analysis of variance using the difference between
posttest and pretest scores as the dependent variable
was used to test the significant differences between
the achievements of the two groups.

5. A t-test was used to test the difference in the time
needed for each student to complete one's projects
between the two groups.

6. A Chi-Square was used to test the independence between
the two groups of the students responses to the
question of Motorola MC68HC11 device.
CHAPTER IV. RESULTS AND FINDINGS

This study was conducted in the Department of Industry and Technology at the California University of Pennsylvania, California, Pennsylvania.

The primary purpose of this study was to design, develop, and evaluate the explanatory experiments on Motorola MC68HC11 microcontroller EVB. A set of seven explanatory laboratory experiments was designed and developed. The detailing of the seven experiments are reported in this chapter. To fulfill the second purpose, a sequence of statistical evaluation procedures was conducted, and test instruments were employed to collect relevant data for analysis. The findings from statistical analysis of data are also reported in this chapter.

Presentation of the Seven Explanatory Laboratory Experiments

Experiment 1

Title: Stepper Motors Control

Objectives:

1. Provide information and experience in input/output ports manipulation of the MC68HC11.
2. Provide information and experience in stepper motors operations.

3. Create a structured assemble program by using a text editor.

4. Use Cross Assembler to generate Motorola S-record for MC68HC11 microcontroller.

**Performance Requirements:**

1. Generate a set of four square waveforms with different phase shifts which are required in rotating a stepper motor shaft with four stationary coils.

2. Write a control program that can read control selections from a dual in line package (DIP) switch to control two stepper motors': (a) speed (b) clockwise rotation (c) counter clockwise rotation (d) one clockwise rotation & one counter clockwise rotation (e) one counter clockwise rotation & one clockwise rotation.

**Equipment Needed:**

IBM PC or Compatible, MS DOS EDLINE text editor or any word processing text editor, Motorola "AS11" Cross Assembler, MC68HC11 EVB, DIP Switch, Stepper Motor * 2, Breadboard.

**Apparatus Set-Up:**

The actual hardware schematic diagram of this experiment is presented in Figure 5.

**Control Program:**

The assembler source codes named as "STEP.EXE" is presented in Appendix C. The algorithm of the control program in a flow chart format was implemented in Figure 6.
Figure 5. Stepper motors control circuit
START

CHECK IS THE "PC1" KEY Hi ?

Y

JUMP TO STOP ALL ROUTINE

N

CHECK IS THE "PC2" KEY Hi ?

Y

JUMP TO RIGHT TURN ROUTINE

N

JUMP TO LEFT TURN ROUTINE

CHECK IS THE "PC3" KEY Hi ?

Y

JUMP TO FORWARD ROUTINE

N

JUMP TO BACKWARD ROUTINE

STOP ALL ROUTINE:

TURN OFF EVERY BIT AT PORT B

RTS

TIMING ROUTINE:

CHECK IS THE "PC1" KEY Hi ?

Y

LOAD SMALL NUMBER FOR HIGH SPEED TIMING

N

LOAD LARGE NUMBER FOR LOW SPEED TIMING

RTS

RTS

Figure 6. Flow chart of stepper motors control
RIGHT TURN ROUTINE:

TURN ON RIGHT
TURN COILS THRU
PORT B

JUMP TO TIMING
DELAY ROUTINE

REPEAT FOR EVERY
COIL AS ABOVE

RTS

LEFT TURN ROUTINE:

TURN ON LEFT
TURN COILS THRU
PORT B

JUMP TO TIMING
DELAY ROUTINE

REPEAT FOR EVERY
COIL AS ABOVE

RTS

FORWARD MOVE ROUTINE:

TURN ON FORWARD
MOVE COILS THRU
PORT B

JUMP TO TIMING
DELAY ROUTINE

REPEAT FOR EVERY
COIL AS ABOVE

RTS

BACKWARD MOVE ROUTINE:

TURN ON BACKWARD
MOVE COILS THRU
PORT B

JUMP TO TIMING
DELAY ROUTINE

REPEAT FOR EVERY
COIL AS ABOVE

RTS

Figure 6. (continued)
Experiment 2

Title: Stop Lights with Pedestrian Interrupt Control

Objectives:
1. Define and describe the regular stop lights changing sequences.
2. Determine when and where is the pedestrian crossing interrupt best fit into the control program.
3. Use the "Strobe A" interrupt of the MC68HC11 microcontroller.
4. Build appropriate hardware to accommodate the "Strobe A" interrupt.

Performance Requirements:
1. Control of four LEDs, red, green, yellow, and green (left turn) to simulate the stop lights in controlling the regular traffic flows.
2. Implement an appropriate pedestrian crossing interrupt function into the regular stop lights sequences.

Equipment Needed:
IBM PC or Compatible, MS DOS EDLINE text editor or any word processing text editor, Motorola "AS11" Cross Assembler, MC68HC11 EVB, LEDs (red * 4, green * 8, yellow * 4), Push Bottom Switch * 4, Breadboard.

Apparatus Set-Up:
The actual hardware schematic diagram of this experiment is presented in Figure 7.

Control Program:
The assembler source codes named as "STOPLITE.EXE" is presented in Appendix C. The algorithm of the control program in a flow chart format was implemented in Figure 8.

**Experiment 3**

**Title:** Keypad Debouncing Encoding

**Objectives:**

1. Distinguish and experience the software debouncing and hardware debouncing of the key switches in a commercial used keypad.

2. Implement the software debouncing and store information from a keypad into memory locations of the MC68HC11 EVB.

3. Implement the hardware debouncing and store information from a keypad into memory locations of the MC68HC11 EVB.

4. Identify the complexity and cost-effect difference of the software debounced and hardware debounced controls.

**Performance Requirements:**

1. Use pure software to handle the mechanical bouncing key switches and store the key information into consecutive memory locations.

2. Use the 74C922 16 key encoder to handle the mechanical bouncing key switches and store the key information into consecutive memory locations.

3. Write an interrupt service routine along with the 74C922 encoder to accomplish the hardware debounced control.
Figure 7. Stop lights control circuit
Figure 8. Flow chart of street lights control.
**N/S CHECK ROUTINE:**

- **GET PED PUSH BUTTON INFO**
  - **IS THERE ANY PUSH BUTTON Pressed?**
    - **N** → **RTS**
    - **Y** → **CLEAR THE D LATCHES**
  - **IS THERE N/S SIDE PLED CROSS?**
    - **Y** → **DUMMY INSTRUCTION TO CLEAR STRA FLAG**
    - **N** → **DUMMY INSTRUCTION TO CLEAR STRA FLAG**
  - **YELLOW ON N/S RED ON W/E SHORT TIME**
  - **JUMP TO STOP ALL TRAFFIC ROUTINE**
  - **JUMP TO GREEN ON W/E RED ON N/S IN MAIN PROGRAM**

**Figure 8. (continued)**
W/E CHECK ROUTINE:

GET PED PUSH BUTTON INFO

IS THERE ANY PUSH BUTTON PRESSED?
N → RTS
Y

CLEAR THE D LATCHES

IS THERE W/E SIDE PED CROSS?
Y → DUMMY INSTRUCTION TO CLEAR STRA FLAG
N

DUMMY INSTRUCTION TO CLEAR STRA FLAG

YELLOW ON N/S RED ON N/S SHORT TIME

JUMP TO STOP ALL TRAFFIC ROUTINE

JUMP TO GREEN ON N/S RED ON W/E IN MAIN PROGRAM

INCREASE COUNTER FOR 4 LOOPS

IS THIS THE END OF COUNTER?
N → INCREASE THE COUNTER 2
Y → RTS

JUMP TO GREEN ON W/E RED ON N/S IN MAIN PROGRAM

Figure 8. (continued)
Equipment Needed:

IBM PC or Compatible, MS DOS EDLINE text editor or any word processing text editor, Motorola "AS11" Cross Assembler, MC68HC11 EVB, 16 Key Keypad, 74C922 16 Key Encoder, Breadboard.

Apparatus Set-Up:

The actual hardware schematic diagrams for software debounced control and hardware debounced controls of this experiment are presented in Figure 9 and 10 respectively.

Control Program:

The assembler source codes named as "KEY2.EXE" for software debounced & "HARDKEY.EXE" for hardware debounced are presented in Appendix C. The algorithm of the control program in flow charts format of the software debouncing and hardware debouncing were implemented in Figure 11 and 12 respectively.

Experiment 4

Title: LCD Display Module Control

Objectives:

1. Provide information and experience in LCD display module characteristics.
2. Distinguish the parallel communication and serial communication between the LCD module and the MC68HC11 EVB.
3. Use the MC68HC11 EVB serial peripheral interface (SPI) to communicate with LCD module serially.
4. Identify the difference between direct parallel and serial communications.
Figure 9. Software debouncing keypad encoding circuit
Figure 10. Hardware debouncing with 74C922 keypad encoding circuit
Figure 11. Flow chart of software debouncing control
Figure 12. Flow chart of hardware debouncing control
Performance Requirements:

1. Utilize all the display functions of a LCD module such as shifting (left & right), blinking, first line display, second line display, and dual line display.

2. Write the control programs and implement hardware for the MC68HC11 EVB to control a LCD module in direct parallel format and serial format by using 74164 serial-in parallel-out shift register.

Equipment Needed:

IBM PC or Compatible, MS DOS EDLINE text editor or any word processing text editor, Motorola "ASll" Cross Assembler, MC68HC11 EVB, Philips LTN242 LCD Module, 74164 8 bit Serial-in Parallel-out Shift Register, Breadboard.

Apparatus Set-Up:

The actual hardware schematic diagrams for parallel format control and serial format control of this experiment are presented in Figure 13 and 14 respectively.

Control Program:

The assembler source codes named as "LCD.EXE" for parallel format control & "LCDWSPI.EXE" for serial format control are presented in Appendix C. The algorithm of the control program in flow charts format of parallel communication and serial communication were implemented in Figure 15 and 16 respectively.
Figure 13. LCD module parallel control circuit
Figure 14. LCD module with SPI serial control circuit
Figure 15. Flow chart of LCD module parallel control
Figure 16. Flow chart of LCD module serial control
Experiment 5

Title: Keypad and LCD Display Module Control System

Objectives:
1. Construct a useful display system with the combination of keypad and LCD module.
2. Compare the difference between the "Strobe A" interrupt and "real time" interrupt.
3. Implement both the software debounced and hardware debounced controls into the display system.

Performance Requirements:
1. Pre-store 16 different messages into memory of the MC68HC11 EVB.
2. Use keypad to select any one of the 16 messages and display it on the LCD module thru serial communication.
3. Use keypad switches 1 to 5 to choose the display format of the chosen displayed message.
4. Implement "Strobe A" interrupt into hardware debouncing of the keypad to encode the key information for this system.
5. Implement "real time" interrupt and software debouncing of the keypad to encode the key information for this system.

Equipment Needed:
IBM PC or Compatible, MS DOS EDLINE text editor or any word processing text editor, Motorola "AS11" Cross Assembler, MC68HC11 EVB, Philips LTN242 LCD Module, 74164 8 bit Serial-in Parallel-out Shift Register, 74C922 16 Key Encoder, Breadboard.

Apparatus Set-Up:
The actual hardware schematic diagrams for hardware
debouncing and software debouncing of this experiment are
presented in Figure 17 and 18 respectively.

Control Program:

The assembler source codes named as "HKYLDSPI.EXE" for
hardware debouncing and "KYLCDSPI.EXE" for software debouncing
are presented in Appendix C. The algorithm of the control
program in flow charts format of hardware debouncing and
software debouncing were implemented in Figure 19 and 20
respectively.

Experiment 6

Title: Electronic Keys Security System

Objectives:
1. Introduce the structured programming and applications.
2. Apply former explanatory experiments into a useful real
   world security system.
3. Exercise and organize serval subroutines, which were
   developed in previous experiments, in a logic flow pattern
   controlled by a simple main program.

Performance Requirements:
1. Pre-store messages into a table of information in the memory
   of MC68HC11 EVB.
2. Use keypad to accept the secret codes from the users.
3. Implement the LCD module to interact with the users thru the
   keypad to direct the operating procedures.
4. Simulate the security performance by turning on the green LED while an opened gate function is needed and turning on the red LED with buzzer while a closed gate procedure is reached.

**Equipment Needed:**

- IBM PC or Compatible, MS DOS EDLINE text editor or any word processing text editor, Motorola "AS11" Cross Assembler, MC68HC11 EVB, Philips LTN242 LCD Module, 74164 8 bit Serial-in Parallel-out Shift Register, LEDs (green & red), Buzzer, Breadboard.

**Apparatus Set-Up:**

The actual hardware schematic diagram of this experiment is presented in Figure 21.

**Control Program:**

The assembler source codes named as "EEKEYSPI.EXE" is presented in Appendix C. The algorithm of the control program in a flow chart format was implemented in Figure 22.

**Experiment 7**

**Title:** DC Motor Control

**Objectives:**

1. Interact the keypad and DC motor in an open loop motor control system.
2. Understand and experience the useful pulse width modulation (PWM) of the MC68HC11.
3. Implement the PWM into a DC motor control application.
Figure 17. Keypad & LCD module with hardware debouncing control circuit
Figure 18. Keypad & LCD module with software debouncing control circuit
Figure 19. Flow chart of hardware debouncing module control in keypad & LCD.
Figure 19. (continued)
DISPLAY FUNCTIONS
Routine:

CHECK IS THE "1" KEY PRESSED? Y
INITIALIZE THE LCD MODULE

N

RETURN

CHECK IS THE "2" KEY PRESSED? Y
DISPLAY 1ST LINE OF 40 LETTERS

N

RETURN

CHECK IS THE "3" KEY PRESSED? Y
SHIFT THE ENTIRE DISPLAY LEFT 80 POSITIONS

N

RETURN

CHECK IS THE "4" KEY PRESSED? Y

N

RETURN

CHECK IS THE "5" KEY PRESSED? Y
BLINK THE ENTIRE DISPLAY 10 TIMES

N

RTS

RETURN

Figure 19. (continued)
DISPLAY MESSAGE ROUTINE:

1. INITIALIZE THE LCD MODULE
2. WRITE TO DISPLAY DATA REGISTER
3. DISPLAY THE INDIVIDUAL LETTER BY WRITING TO DATA REGISTER
4. IS THIS THE LAST LETTER TO BE DISPLAYED?
   - NO: Go back to step 2
   - YES: RTS

STROBE A INTERRUPT SERVICE ROUTINE:

1. LOAD INFORMATION FROM KEYPAD THRU PORT C
2. USE DUMMY INSTRUCTION TO CLEAR STROBE A INTERRUPT
3. RTI

Figure 19. (continued)
Figure 20. Flow chart of software debouncing in keypad & LCD module control
Figure 20. (continued)
DISPLAY FUNCTIONS
ROUTINE:

CHECK IS THE "1" KEY PRESSED?  Y  INITIALIZE THE LCD MODULE
N  RETURN

DISPLAY 1ST LINE OF 40 LETTERS
Y  RETURN

CHECK IS THE "2" KEY PRESSED?  Y  DISPLAY 2ND LINE OF 40 LETTERS
N  RETURN

CHECK IS THE "3" KEY PRESSED?  Y  BLINK THE ENTIRE DISPLAY 10 TIMES
N  RTS

SHIFT THE ENTIRE DISPLAY LEFT 80 POSITIONS
Y  RETURN

CHECK IS THE "4" KEY PRESSED?  Y  BLINK THE ENTIRE DISPLAY 10 TIMES
N  RTS

CHECK IS THE "5" KEY PRESSED?  Y  BLINK THE ENTIRE DISPLAY 10 TIMES
N  RTS

Figure 20. (continued)
DISPLAY MESSAGE ROUTINE:

1. INITIALIZE THE LCD MODULE
2. WRITE TO DISPLAY DATA REGISTER
3. DISPLAY THE INDIVIDUAL LETTER BY WRITING TO DATA REGISTER
4. IS THIS THE LAST LETTER TO BE DISPLAYED?
   - N: Go back to step 2
   - Y: RTS

Figure 20. (continued)
REAL TIME INTERRUPT SERVICE ROUTINE:

1. IS THERE ANY KEY Pressed?
   - Y: Time delay for key debouncing
   - N: Dummy instruction to clear real-time interrupt flag

2. IS THE KEY CODE MATCHES WITH TABLE?
   - Y: Decode the key according to the code position in the table
     - Store the key code information into memory
     - Advance the pointer in memory for next key code information
     - Dummy instruction to clear real-time interrupt flag
     - RTI
   - N: Does it match with next in table?
     - N: Dummy instruction to clear real-time interrupt flag
     - Y: Return to step 1

Figure 20. (continued)
Figure 21. Electronic keys security system control circuit
START

SET UP THE MESSAGE TABLE

INITIALIZE THE SPI COMMUNICATION THRU 74164 PARALLEL TO SERIAL CONVERSION

SET THE 3 TIMES TRIAL COUNTER

CLEAR THE KEYCODE STORAGE MEMORY

JUMP TO DISPLAY MESSAGE ROUTINE

CHECK THE 1ST PAIR KEY CODE SAME?

N  JUMP TO THE CLOSE DOOR ROUTINE

Y  CHECK THE 2ND PAIR KEY CODE SAME?

N  JUMP TO THE CLOSE DOOR ROUTINE

Y  JUMP TO THE OPEN DOOR ROUTINE

BRANCH ALWAYS

Figure 22. Flow chart of electronic keys security system control
GET KEY CODES ROUTINE:

1. IS THERE ANY KEY Pressed?
   - Y: TIME DELAY FOR KEY DEBOUNCING
   - N: BRANCH ALWAYS

2. IS THE KEY CODE MATCH WITH TABLE?
   - Y: DECODE THE KEY ACCORDING TO THE CODE POSITION IN THE TABLE
   - N: DOES IT MATCH WITH NEXT IN TABLE?

3. STORE THE KEY CODE INFORMATION INTO MEMORY
   - Y: ADVANCE THE POINTER IN MEMORY FOR NEXT KEY CODE INFORMATION
   - N: INCREASE COUNTER & STORE IN TEMP MEMORY

4. IS IT THE FOURTH TIME OF KEY ENTRY?
   - Y: RTS
   - N: BRANCH ALWAYS

Figure 22. (continued)
CLOSE DOOR ROUTINE:

START

INCREASE THE 3 TIMES TRIAL COUNTER

DISPLAY WARNING MESSAGE BY JUMPING TO DISPLAY MESSAGE ROUTINE

CHECK THE TRIAL COUNTER = 3 ?

Y

DISPLAY ALARM MESSAGE BY JUMPING TO DISPLAY MESSAGE ROUTINE

ACTIVATE THE ALARM SIGNAL

DELAY THE ALARM FOR 5 MINUTES

CLEAR THE 3 TIMES TRIAL COUNTER

RTS

N

RTS

Figure 22. (continued)
DISPLAY MESSAGE ROUTINE:

1. Initialize the LCD module
2. Write to display data register
3. Display the individual letter by writing to data register
4. Is this the last letter to be displayed?
   - Yes: RTS
   - No: Go back to step 3

OPEN DOOR ROUTINE:

1. Display welcome message
2. Activate the open door signal
3. Delay the open door for 6 minutes
4. Deactivate the open door signal
5. RTS

Figure 22. (continued)
Performance Requirements:

1. Use the MC68HC11 EVB to generate variable duty cycles of a PWM signal according to the keypad key switches information.
2. Drive a 1A - 2A DC motor by this PWM signal thru the help of Texas Instrument's L298KV full-H driver.
3. Write a program which can control the DC motor's speed, direction, and on/off according to the information encoded form the keypad.

Equipment Needed:

IBM PC or Compatible, MS DOS EDLINE text editor or any word processing text editor, Motorola "AS11" Cross Assembler, MC68HC11 EVB, 16 Key Keypad, 74C922 16 Key Encoder, 7404 Inverter Buffer, Texas Instrument L298KV Full-H Driver, 1A - 2A DC Motor, Breadboard.

Apparatus Set-Up:

The actual hardware schematic diagram of this experiment is presented in Figure 23.

Control Program:

The assembler source codes named as "DCMOTOR.EXE" is presented in Appendix C. The algorithm of the control program in a flow chart format was implemented in Figure 24.
Figure 23. DC motor control circuit
Figure 24. Flow chart of DC motor control
OUTPUT COMPARE INTERRUPT SERVICE ROUTINE:

1. Load free running counter information from TCNT
2. Add the HIOUT value from the user to the output compare register
3. Use dummy instruction to clear the output compare interrupt
   - RTI

STROBE A INTERRUPT SERVICE ROUTINE:

1. Load information from keypad thru port C
2. Use dummy instruction to clear STRA interrupt
   - RTI

Figure 24. (continued)
To achieve the equivalency between the control group and the experimental group, the sample with matching pairs were assigned to each group. To further minimize the original differences between the two groups, the pretest was administered to each group, and the pretest scores were used as covariates in the analysis of covariance of a General Linear Mode (GLM). The posttest test was administered at the end of the Fall 1991 semester after the students had submitted the reports of their design projects. The interpretations of the gathered data have followed the statistical procedures, which significant differences were tested at the alpha 0.05 level, are listed below:

1. Characteristics of the sample;
2. Analyses of variance of pretest and posttest scores;
3. Analysis of covariance of posttest scores;
4. Analysis of covariance with pretest and group interaction;
5. Analysis of variance of students learning achievements;
6. T-test of students' efficiency in completing a project.
**Characteristics of the Sample**

In the Department of Industry and Technology at the California University of Pennsylvania, twenty students who participated in this study were twelve juniors, six sophomores, and two freshmen, and their various majors included Electrical Engineering Technology, Electronic Technology, and Industrial Technology. Of the twenty students in the study, all were males. They were divided into two groups, and the assignment of each group member was based on one's major and pretest scores. The distribution of these two groups' scores is presented as follows:

<table>
<thead>
<tr>
<th>Control Group</th>
<th>Experimental Group</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Pretest</strong></td>
<td><strong>Posttest</strong></td>
</tr>
<tr>
<td>1. 74</td>
<td>102</td>
</tr>
<tr>
<td>2. 59</td>
<td>79</td>
</tr>
<tr>
<td>3. 56</td>
<td>109</td>
</tr>
<tr>
<td>4. 46</td>
<td>70</td>
</tr>
<tr>
<td>5. 45</td>
<td>62</td>
</tr>
<tr>
<td>6. 43</td>
<td>71</td>
</tr>
<tr>
<td>7. 43</td>
<td>62</td>
</tr>
<tr>
<td>8. 39</td>
<td>56</td>
</tr>
<tr>
<td>9. 35</td>
<td>40</td>
</tr>
<tr>
<td>10. 21</td>
<td>47</td>
</tr>
</tbody>
</table>

| Mean       | 46.1000 | 69.8000 | 52.9000 | 85.1000 |
| Std. Dev.  | 14.4025 | 22.0595 | 11.9949 | 15.1617 |

The chart of pretest and posttest scores of both control and experimental groups is shown in Figure 25.
Figure 25. The experimental and the control groups' pretest & posttest scores chart
From the two groups' pretest and posttest scores distribution chart, it shows that both groups have higher achievements after taking the course. It appears that the experimental group has higher learning achievement than the control group. In order to present a statistically significant difference between the two groups, the following analysis of retrieved data were used in this study.

**Analyses of Variance of Pretest and Posttest Scores**

The ANOVA tables of the pretest and posttest scores are presented in Table 4 and Table 5.

**Table 4. The analysis of variance of pretest scores**

<table>
<thead>
<tr>
<th>SCORE</th>
<th>DEGREE OF FREEDOM</th>
<th>SUM OF SQUARE</th>
<th>MEAN SQUARE</th>
<th>F-VALUE</th>
<th>PR&gt;F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>1</td>
<td>231.2000</td>
<td>231.2000</td>
<td>1.32</td>
<td>0.2663</td>
</tr>
<tr>
<td>Error</td>
<td>18</td>
<td>3161.8000</td>
<td>175.6556</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>19</td>
<td>3393.0000</td>
<td></td>
<td>R² = 0.0681</td>
<td></td>
</tr>
</tbody>
</table>

As shown in Table 4, the F value was 1.32 and the probability was 0.2663, which was much greater than the 0.05 level. This means that there was no significant difference in the pretest scores between the two groups.
According to the results of the analysis in Table 5, the F value was 3.27 and the probability of 0.0874, which was still greater than the 0.05 level. Therefore, there was no significant difference in the posttest scores between the two groups.

Table 5. The analysis of variance of posttest scores

<table>
<thead>
<tr>
<th></th>
<th>DEGREE OF FREEDOM</th>
<th>SUM OF SQUARE</th>
<th>MEAN SQUARE</th>
<th>F-VALUE</th>
<th>PR&gt;F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>1</td>
<td>1170.4500</td>
<td>1170.4500</td>
<td>3.27</td>
<td>0.0874</td>
</tr>
<tr>
<td>Error</td>
<td>18</td>
<td>6448.5000</td>
<td>358.2500</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>19</td>
<td>7618.9500</td>
<td></td>
<td>R² = 0.1536</td>
<td></td>
</tr>
</tbody>
</table>

Analysis of Covariance of Posttest Scores

The correlation coefficient between the pretest and posttest scores as reported in Table 6 was 0.7856, which means these two tests are strongly related. This leads to a necessity of analysis of covariance (ANCOVA) table of the posttest scores, using pretest as the covariate, is in Table 7.

Results of the analysis of covariance as reported in Table 7, indicate the F value was 16.12 and probability was 0.0001, which was much smaller than the 0.05 level or even 0.01 level. The significant effect on the two groups was due to the pretest
which was used as covariate to reduce the original difference between the two groups.

Table 6. Pearson correlation coefficient of the pretest and the posttest scores

<table>
<thead>
<tr>
<th></th>
<th>PRETEST</th>
<th>POSTTEST</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRETEST</td>
<td>1.0000</td>
<td>0.7856</td>
</tr>
<tr>
<td></td>
<td>0.0000</td>
<td>0.0001</td>
</tr>
<tr>
<td>POSTTEST</td>
<td>0.7856</td>
<td>1.0000</td>
</tr>
<tr>
<td></td>
<td>0.0001</td>
<td>0.0000</td>
</tr>
</tbody>
</table>

Table 7. The analysis of covariance of posttest scores

Dependent Variable: Posttest

<table>
<thead>
<tr>
<th>SCORE</th>
<th>DEGREE OF FREEDOM</th>
<th>SUM OF SQUARE</th>
<th>MEAN SQUARE</th>
<th>F-VALUE</th>
<th>PR&gt;F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>2</td>
<td>4988.1363</td>
<td>2492.0681</td>
<td>16.12</td>
<td>0.0001</td>
</tr>
<tr>
<td>Error</td>
<td>17</td>
<td>2630.8137</td>
<td>154.7537</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>19</td>
<td>7618.9500</td>
<td></td>
<td>R² = 0.6547</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SCORE</th>
<th>DEGREE OF FREEDOM</th>
<th>TYPE I S.S.</th>
<th>F-VALUE</th>
<th>PR&gt;F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pretest</td>
<td>1</td>
<td>4702.6320</td>
<td>30.39</td>
<td>0.0001</td>
</tr>
<tr>
<td>Group</td>
<td>1</td>
<td>285.5043</td>
<td>1.84</td>
<td>0.1921</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SCORE</th>
<th>DEGREE OF FREEDOM</th>
<th>TYPE III S.S.</th>
<th>F-VALUE</th>
<th>PR&gt;F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pretest</td>
<td>1</td>
<td>3817.6863</td>
<td>24.67</td>
<td>0.0001</td>
</tr>
<tr>
<td>Group</td>
<td>1</td>
<td>285.5043</td>
<td>1.84</td>
<td>0.1921</td>
</tr>
</tbody>
</table>
On the other hand, the covariate (pretest) did have the effects in reducing the errors on the two groups but not the posttest scores in testing the significant difference between the two groups.

The effect of the covariates (pretest scores) can be further expressed in Table 8.

Table 8. Pretest and posttest statistics

<table>
<thead>
<tr>
<th>GROUP</th>
<th>PRETEST</th>
<th>POSTTEST</th>
<th>ADJUSTED POSTTEST</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MEAN</td>
<td>STD. DEV.</td>
<td>MEAN</td>
</tr>
<tr>
<td>Experimental</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Group (N = 10)</td>
<td>52.9000</td>
<td>11.9949</td>
<td>85.1000</td>
</tr>
<tr>
<td>Control Group</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(N = 10)</td>
<td>46.4025</td>
<td>14.4025</td>
<td>69.8000</td>
</tr>
</tbody>
</table>

Note that in Table 8 the posttest mean for the experimental group had been adjusted downward, whereas the posttest mean for the control group had been adjusted upward. In this way, the pretest scores were used as covariate, and the inequality between the groups was compensated by the covariate.
Analysis of Covariate with Pretest and Group Interaction

The pretest and group interaction as well as ANCOVA are presented in Table 9.

Table 9. The analysis of covariance with interaction

<table>
<thead>
<tr>
<th>Dependent Variable: Posttest</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCORE</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>Model</td>
</tr>
<tr>
<td>Error</td>
</tr>
<tr>
<td>Total</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SCORE</th>
<th>DEGREE OF FREEDOM</th>
<th>TYPE I S.S.</th>
<th>F-VALUE</th>
<th>PR&gt;F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pretest</td>
<td>1</td>
<td>4702.6320</td>
<td>30.86</td>
<td>0.0001</td>
</tr>
<tr>
<td>Group</td>
<td>1</td>
<td>285.5043</td>
<td>1.87</td>
<td>0.1900</td>
</tr>
<tr>
<td>Pretest* Group</td>
<td>1</td>
<td>192.6408</td>
<td>1.26</td>
<td>0.2775</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SCORE</th>
<th>DEGREE OF FREEDOM</th>
<th>TYPE III S.S.</th>
<th>F-VALUE</th>
<th>PR&gt;F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pretest</td>
<td>1</td>
<td>393.8760</td>
<td>22.27</td>
<td>0.0002</td>
</tr>
<tr>
<td>Group</td>
<td>1</td>
<td>310.8759</td>
<td>2.04</td>
<td>0.1724</td>
</tr>
<tr>
<td>Pretest* Group</td>
<td>1</td>
<td>192.6408</td>
<td>1.26</td>
<td>0.2775</td>
</tr>
</tbody>
</table>

From Table 9, it can be noted that the significant F value was 0.0003 which was much smaller than the 0.05 level. But the only significant effect was the pretest scores which were used
as a covariate in this ANCOVA table. There was no significant effect on the pretest and groups interaction. This means the original two groups do not have a large difference and the original differences have been further reduced by the pretest scores.

Analysis of Variance of Students' Learning Achievements

The students' learning achievements can be measured through the difference between the pretest and posttest scores. The ANOVA table of the scores differences is presented in Table 10.

Table 10. The analysis of variance of the scores differences

<table>
<thead>
<tr>
<th>SCORE</th>
<th>DEGREE OF FREEDOM</th>
<th>SUM OF SQUARE</th>
<th>MEAN SQUARE</th>
<th>F-VALUE</th>
<th>PR&gt;F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>1</td>
<td>361.2500</td>
<td>361.2500</td>
<td>2.44</td>
<td>0.1355</td>
</tr>
<tr>
<td>Error</td>
<td>18</td>
<td>2661.7000</td>
<td>147.8722</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>19</td>
<td>3022.9500</td>
<td></td>
<td>$R^2 = 0.1195$</td>
<td></td>
</tr>
</tbody>
</table>

Based on the results of analysis of data as reported in Table 10, the F value was 2.44 and probability was 0.1355, which was larger than 0.05 level. As a result, there was no significant difference of students' learning achievements between the two groups.
From Table 7 the analysis of covariance of posttest scores and Table 10 the analysis of variance of scores differences, both give the same conclusion that there is no significant difference of the students' learning achievements. It is also verified that the two groups are independent.

In Table 11 and Figure 25, the students' learning achievements of the experimental group were higher than the control group, but based on the results of statistical analysis of data in Table 10 there was no evidence of significant difference in the posttest achievement between the two groups.

Table 11. The learning achievement of the statistics of the two groups

<table>
<thead>
<tr>
<th>GROUP</th>
<th>MEAN</th>
<th>STD. DEV.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Experimental Group (N = 10)</td>
<td>32.2000</td>
<td>11.9518</td>
</tr>
<tr>
<td>Control Group (N = 10)</td>
<td>23.7000</td>
<td>12.3653</td>
</tr>
</tbody>
</table>

Therefore, null hypothesis $H_{01}$ stated as: "There is no significant learning achievement difference between the experimental group students who used the Motorola 68HC11 microcontroller EVB and the control group students who used the traditional microprocessor as their project design tool." was not rejected.
T-test of Students' Efficiency in Completing a Project

Four projects criteria were given to students in the two groups, and the students had a choice to choose any three of the given four projects as a requirement to complete the course. The list of the average days for each student to finish a required project is presented in Table 12.

Table 12. The average days needed for the experimental and control groups

<table>
<thead>
<tr>
<th>Control Group</th>
<th>Average Days</th>
<th>Experimental Group</th>
<th>Average Days</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>32.6667</td>
<td>1.</td>
<td>14.0000</td>
</tr>
<tr>
<td>2.</td>
<td>29.0000</td>
<td>2.</td>
<td>14.0000</td>
</tr>
<tr>
<td>3.</td>
<td>18.3333</td>
<td>3.</td>
<td>25.0000</td>
</tr>
<tr>
<td>4.</td>
<td>17.0000</td>
<td>4.</td>
<td>25.0000</td>
</tr>
<tr>
<td>5.</td>
<td>34.0000</td>
<td>5.</td>
<td>11.6667</td>
</tr>
<tr>
<td>6.</td>
<td>34.0000</td>
<td>6.</td>
<td>9.3333</td>
</tr>
<tr>
<td>7.</td>
<td>35.0000</td>
<td>7.</td>
<td>9.3333</td>
</tr>
<tr>
<td>8.</td>
<td>11.6667</td>
<td>8.</td>
<td>10.0000</td>
</tr>
<tr>
<td>9.</td>
<td>11.6667</td>
<td>9.</td>
<td>10.6667</td>
</tr>
<tr>
<td>10.</td>
<td>11.6667</td>
<td>10.</td>
<td>11.3333</td>
</tr>
</tbody>
</table>

A t-test was used to test the significant difference of the time required for completing the projects between the two groups. The t-test results are presented as shown in Table 13.

According to the t-test results displayed in Table 13, the probability was 0.0219 which was less than the 0.05 level. So, a conclusion can be reached that experimental group students who used Motorola 68HC11 EVB did finish the required projects faster.
than the control group students who used Motorola 6809 microprocessor in TRS-80 microcomputer.

Table 13. The t-test of average days needed for completing a project

<table>
<thead>
<tr>
<th>GROUP</th>
<th>N</th>
<th>MEAN</th>
<th>STD.DEV.</th>
<th>T</th>
<th>PR&gt;T</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control</td>
<td>10</td>
<td>23.5000</td>
<td>10.3067</td>
<td>2.5090</td>
<td>0.0219</td>
</tr>
<tr>
<td>Experiment</td>
<td>10</td>
<td>14.0333</td>
<td>6.0112</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Findings

Based on the statistics presentation of the experimental and the control groups, the first null hypothesis $H_{01}$: "There is no significant learning achievements differences between the experimental group students who used the Motorola 68HC11 microcontroller EVB and the control group students who used the traditional microprocessor as their project design tool." was not rejected, and the second null hypothesis $H_{02}$: "There is no difference in time required to complete the assigned projects between the two groups." was rejected. However, the survey question presented in the end of the posttest has the following results (Table 14).

The Pearson Chi-Square test of independence, is
\( X^2 = \sum \left( \frac{(f_0 - f_e)^2}{f_e} \right) \), where \( f_0 \) is the observed frequency and \( f_e \) is the expected frequency \( f_e = \frac{(\text{Row Total} \times \text{Column Total})}{\text{Grand Total}} \).

By using this Chi-Square test for independence, the Chi-Square was 2.892 \( (X^2 = 2.892) \) and its probability was 0.235 which was not significant at the 0.05 level of confidence. Therefore, the conclusion drawn is that the students' opinions on the Motorola 68HC11 EVB device have no relation to the groups to which they belonged. Thirteen out of twenty students (65\%) in the experimental and control groups responded that they preferred to use the Motorola 68HC11 EVB in their project designs.

Table 14. The preference of the 68HC11 EVB in the two groups

<table>
<thead>
<tr>
<th>GROUP</th>
<th>OPPOSE 68HC11EVB</th>
<th>NO OPINION 68HC11EVB</th>
<th>FAVOR 68HC11EVB</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>Experiment</td>
<td>0</td>
<td>2</td>
<td>8</td>
<td>10</td>
</tr>
<tr>
<td>Total</td>
<td>2</td>
<td>5</td>
<td>13</td>
<td>20</td>
</tr>
</tbody>
</table>
This chapter is intended to summarize the purpose, methodology, and findings, to draw conclusions, and to make recommendations from the study. The summary of this study includes the background, purpose, and methodology. The conclusions are restricted by the limitations and assumptions used in the study, the actual findings and inferences are presented in the conclusions. The recommendations for further research are based on the actual findings and inferences.

Summary of the Study

Background

Due to the drastic advancement of high technology, it is difficult for educators to keep up with its new development without actually using the device. As far as can be determined from the related literature review, currently there are no academic applications of the microcontroller available in the educational programs of Industrial Technology. The Motorola MC68HC11 microcontroller is one of the semiconductor achievements that offers various facilities in microprocessor control applications with no need for excessive external support in hardwares. This new product provides the flexibility,
efficiency, economy, and simplicity for the designing job of the microprocessor control applications. This study was conducted to improve the current microprocessor teaching materials which are mostly out-of-date and expensive in conducting lab experiments, and to determine the relative effectiveness of the new microprocessor teaching materials. A set of seven explanatory lab experiments were designed and developed on the Motorola MC68HC11 microcontroller EVB (Evaluation Board) which were presented in Chapter IV, and students' learning achievement tests on implementing these set of experiments have been used to determine the effectiveness of the design as well as applications of the microcontroller. Also, the average days for completing a project were used to compute the efficiency difference in using different devices between the two groups.

Purpose

There were two purposes for this study. The first purpose was to investigate, design, and develop a set of seven sample explanatory experiments by using Motorola MC68HC11 EVB for Industrial Technology programs. The second purpose was to evaluate the effectiveness of applying the sample explanatory experiments into Industrial Technology microprocessor laboratory activities.

Research Design

Seven explanatory lab experiments were developed and designed according to the current literature reviews of microcontroller and several Industrial Technology microprocessor
courses guidelines. First four out of these seven experiments were chosen to be the students design projects samples. Students were asked to implement any three of the four experiments as their design project to be a part of the course requirements.

In order to evaluate the effectiveness of the explanatory experiments, a pretest and posttest with matching statistical model was used. Pretest and posttest instruments were developed for this model. First, the students who attended the EET 330/IND335 Advanced Microprocessors course were administered a pretest. Secondly, the students were then divided into two groups (experimental and control group), according to their academic major, potential knowledge background, and pretest scores. Both groups were given the same lectures and demonstrations during the full semester period. The experimental group was assigned to use MC68HC11 EVB microcontroller to implement the projects according to their creative thinking. The control group was assigned to use MC6809, a standard microprocessor, which resided in the TRS-80 personal computer, to design the projects. At the end of the semester, a posttest with a single survey question was administered to both groups. A set of average days for each student to complete the projects were retrieved near the end of the semester. The pretest and posttest scores were used in several statistical analysis to compute the differences between
the two groups. The average days in completing a project were used to compute the efficient difference between the two groups.

Conclusions

It has been proved that there is no significant learning achievement difference between the experimental group students who used the Motorola 68HC11 microcontroller EVB and the control group students who used the traditional microprocessor as their project design tool. There is a significant difference in time required to complete the assigned projects between the two groups.

Through statistical analysis, the study showed that there was no significant difference of students' learning achievement between the two groups who used MC68HC11 EVB or MC6809 microprocessor. The analysis of covariance, which used the pretest scores as a covariate, reduced the original difference between the two groups, and adjusted the means of the posttest scores. The conclusion of this study confirms that the statistical model, pretest posttest with matching is an effective technique compared with other experimental design methods, when used in analyzing the learning achievement difference between the two groups.

These two groups in the study are independent in their opinions about the preference of the MC68HC11 EVB device. The
t-test of the efficiency of completing a project between the two groups is significant and there is no conflict in the assumptions before the t-test analysis. It can also be concluded that the MC68HC11 did have an effect on the efficiency in saving the time required for completing a project.

The results of this study were based upon the research limitations and assumptions which were:

1. the limited number of subjects
2. the subjects who majored in Electrical Engineering Technology, Electronic Technology, Industrial Technology in the Department of Industry and Technology at the California University of Pennsylvania
3. the uncontrolled interactions between and within the two groups
4. the limited course period
5. the available support devices to the subjects
6. the assumed normal distribution in the subjects

The inference from the conclusion is that the number of subjects who attended the study are not large enough. Consequently, with a larger sample of subjects and the study period extended to two or more semesters, these factors may yield a statistically significant difference.

From the exploring of technical literatures of the MC68HC11 EVB as well as the MC6809 and the experiencing of the design as well as implementation of the explanatory experiments, the following conclusions were obtained:
1. The MC68HC11 EVB has more functions than 6809 microprocessor.

2. The cross assembler is more flexible and easier to use than the 6809 assembler.

3. The MC68HC11 EVB has versatile timer functions than any 6809 peripheral timer chip.

4. Using MC68HC11 EVB is more economical than 6809 microprocessor.

5. MC68HC11 EVB can still use the same peripheral chip as the 6809 does.

6. MC68HC11 EVB can be easily implemented in a multiple processor environment.

7. 6809 has more addressing control functions in its instruction sets than MC68HC11.

8. MC68HC11 is the best device for use in simple control application, while 6809 is a general purpose device for use in a microcomputer system.

**Recommendations**

Laboratory activities for microprocessor should be efficient. Especially, when a project design is required in the laboratory activities, the most efficient and advanced techniques should be made available to the students. The MC68HC11 EVB appears to be an efficient device as a design tool.
The design skills of an useful application should be closely related to the state-of-the-art technology.

Teaching and learning in the area of microprocessor applications are complex and challenging. More research is needed in this area to identify the new development of technologies. The following research studies are recommended:

1. Repeat the similar experiment, using all the sample explanatory experiments and involving a larger samples from different colleges and universities at various locations in the country.

2. Investigate the relative merits of other microcontroller application design and lecture content at different course levels.

3. Advance the 8 bit microcontroller to 16 bit or even 32 bit microcontrollers.

4. Apply the microcontroller functions to robotics controls, pneumatic or hydraulic servo control applications.

5. Design and develop the more complex experiments which utilize all the features of the MC68HC11 EVB.

6. Investigate all the state-of-the-art microcontrollers which are currently used in the industry applications and find their applicabilities to the Industrial Technology instructional programs.
BIBLIOGRAPHY


ACKNOWLEDGEMENTS

In conducting this study, a number of people have made very helpful contributions and suggestions. The author wishes to express his thanks to the committee members: Dr. Arthur Akers, Dr. Larry Bradshaw, Dr. David Johnson, Dr. David Stephenson, and Dr. Robert Strahan for their assistance, contributions, ideas, support, and encouragement.

A special thanks is extended to Dr. William Wolansky, the author's major professor, for his guidance, leadership, and support throughout this study. For his contribution, counsel, and direction of this study, the author is grateful to him.

In addition, my gratitude and appreciation is expressed to the Sigma Xi Grants-in-Aid for their partial financial support of this study.

Thanks are also due to author's wife Elaine and parents for their encouragement and moral support during the preparation of this study.
APPENDIX A. HUMAN SUBJECT REVIEW
Information for Review of Research Involving Human Subjects
Iowa State University
(Please type and use the attached instructions for completing this form)

Research and Development of Microcontroller Experiment Instructional Units and Their Effectiveness with IT/ET/EET Majors

1. Title of Project

2. I agree to provide the proper surveillance of this project to insure that the rights and welfare of the human subjects are protected. I will report any adverse reactions to the committee. Additions to or changes in research procedures after the project has been approved will be submitted to the committee for review. I agree to request renewal of approval for any project continuing more than one year.

Hsiung, Steve Chih
Typed Name of Principal Investigator

3. Signatures of other investigators

William D. Wolansky
Major Advisor

4. Principal Investigator(s) (check all that apply)

Faculty □ Staff □ Graduate Student □ Undergraduate Student

5. Project (check all that apply)

Research □ Thesis or dissertation □ Class project □ Independent Study (490, 590, Honors project)

6. Number of subjects (complete all that apply)

# Adults, non-students □ # ISU student □ # minors under 14 □ # minors 14 - 17 □ other (explain)

7. Brief description of proposed research involving human subjects: (See instructions, Item 7. Use an additional page if needed.)

Please see the attached sheets.

8. Informed Consent:

□ Signed informed consent will be obtained. (Attach a copy of your form.)

□ Modified informed consent will be obtained. (See instructions, item 8.)

□ Not applicable to this project.
9. Confidentiality of Data: Describe below the methods to be used to ensure the confidentiality of data obtained. (See instructions, item 9.)

1. The students' names and numbers will be used for identification purpose only.
2. All the test scores will be kept confidential and recorded for statistical analysis.
3. All the data will be reported in the form of group results.

10. What risks or discomfort will be part of the study? Will subjects in the research be placed at risk or incur discomfort? Describe any risks to the subjects and precautions that will be taken to minimize them. (The concept of risk goes beyond physical risk and includes risks to subjects' dignity and self-respect as well as psychological or emotional risk. See instructions, item 10.)

   No Risk.

11. CHECK ALL of the following that apply to your research:

   □ A. Medical clearance necessary before subjects can participate
   □ B. Samples (Blood, tissue, etc.) from subjects
   □ C. Administration of substances (foods, drugs, etc.) to subjects
   □ D. Physical exercise or conditioning for subjects
   □ E. Deception of subjects
   □ F. Subjects under 14 years of age and/or □ Subjects 14 - 17 years of age
   □ G. Subjects in institutions (nursing homes, prisons, etc.)
   □ H. Research must be approved by another institution or agency (Attach letters of approval)

   If you checked any of the items in 11, please complete the following in the space below (include any attachments):

   Items A - D Describe the procedures and note the safety precautions being taken.

   Item E Describe how subjects will be deceived; justify the deception; indicate the debriefing procedure, including the timing and information to be presented to subjects.

   Item F For subjects under the age of 14, indicate how informed consent from parents or legally authorized representatives as well as from subjects will be obtained.

   Items G & H Specify the agency or institution that must approve the project. If subjects in any outside agency or institution are involved, approval must be obtained prior to beginning the research, and the letter of approval should be filed.
Checklist for Attachments and Time Schedule

The following are attached (please check):

12. **X** Letter or written statement to subjects indicating clearly:
   a) purpose of the research
   b) the use of any identifier codes (names, #'s), how they will be used, and when they will be removed (see Item 17)
   c) an estimate of time needed for participation in the research and the place
   d) if applicable, location of the research activity
   e) how you will ensure confidentiality
   f) in a longitudinal study, note when and how you will contact subjects later
   g) participation is voluntary; nonparticipation will not affect evaluations of the subject

13. ■ Consent form (if applicable)

14. ■ Letter of approval for research from cooperating organizations or institutions (if applicable)

15. **X** Data-gathering instruments

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>16. Anticipated dates for contact with subjects:</td>
<td></td>
</tr>
<tr>
<td>First Contact</td>
<td>Last Contact</td>
</tr>
<tr>
<td>August 26, 1991</td>
<td>December 6, 1991</td>
</tr>
<tr>
<td>Month / Day / Year</td>
<td>Month / Day / Year</td>
</tr>
</tbody>
</table>

17. If applicable: anticipated date that identifiers will be removed from completed survey instruments and/or audio or visual tapes will be erased:

   May 30, 1992
   
   Month / Day / Year

18. Signature of Departmental Executive Officer  Date  Department or Administrative Unit

   John C. Rogers  5/18/91  Industrial Education & Technology

19. Decision of the University Human Subjects Review Committee:

   X  Project Approved  ■  Project Not Approved  ■  No Action Required

   Patricia M. Keith  5/1/91  Signature of Committee Chairperson

   Name of Committee Chairperson  Date  Signature of Committee Chairperson

GC: 1/90
APPENDIX B. WRITTEN STATEMENT AND TEST INSTRUMENTS
Written Statement to Subjects

The purpose of this test is used to determine the effectiveness of using Motorola MC68HC11 microcontroller EVB (Evaluation Board) in the microprocessor experiment laboratory learning activities. This activities involved in all the labs and projects of the EET 330/IND335 Advanced Microprocessors course.

All the research in gathering the data will be in the course period and no extra assignments or activities will be added in addition to the course requirements.

The use of your names and identification numbers will be kept completely confidential and for this research purpose only. The test scores will be used only in statistical analysis, no individual name will be revealed, and scores will not be applied in judging your final grade in the course.
Test Instruments

Questions for Pretest only

STUDENT INFORMATION SHEET

Name: ____________________________, ____________________________, ____________________________

(Last) (First) (Middle)

Soc. Sec. Number: ____________________________

Home Address: ____________________________

, Phone #: __________________

School Address: ____________________________

, Phone #: __________________

High School Attended: ____________________________

Year Graduated: ____________________________

If a College Transfer Student: (Name of College Attended)

Current Schedule

<table>
<thead>
<tr>
<th>Course</th>
<th>Cr.</th>
<th>M</th>
<th>T</th>
<th>W</th>
<th>TH</th>
<th>F</th>
<th>Instructor</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Questions for Pretest & Posttest

1. What is your major area? ____________________________
Directions: Please respond to each item by placing a check mark in the most appropriate space for each of the question.

2. Your gender is: ( ) Male ( ) Female

3. You are a: ( ) Freshman student ( ) Sophomore student ( ) Junior student ( ) Senior student.

4. What is your accumulated GPA?
   ( ) below 1.9 ( ) 2.0 - 2.5 ( ) 2.6 - 3.0 ( ) 3.1 - 3.5 ( ) 3.6 - 4.0

5. What is your SAT test score?
   ( ) 200 - 300 ( ) 301 - 400 ( ) 401 - 500 ( ) 501 - 600 ( ) 601 - 700 ( ) 701 above

Directions: Please respond to each item by placing a check mark in the most appropriate space for each of the question.

1. Can a microprocessor manipulate or calculate with analog signals directly?
   ( ) Yes ( ) No

2. Can a microprocessor manipulate or calculate with digital signals directly?
   ( ) Yes ( ) No

3. The difference between analog and digital signals is: the former signal can take on an infinite number of possible values, the latter signal has a finite number of possible values.
   ( ) Yes ( ) No

4. What is the number system used in microprocessor machine language?
   ( ) Binary Number ( ) Octal Number ( ) Decimal Number ( ) Hexadecimal Number

5. What is the code used in Assembly Language?
   ( ) Binary Number ( ) Octal Number ( ) Decimal Number ( ) Hexadecimal Number

6. Find the equivalent matching letter from the right column to the left column.
   ( ) (1) $53.9_{10}$ (a) $1110000_2$
   ( ) (2) $10101_2$ (b) $41_{\text{Hex}}$
   ( ) (3) $6FE_{16}$ (c) $3376_8$
   ( ) (4) $70_{\text{Hex}}$ AND $FF_{\text{Hex}}$ (d) $25_8$
7. What is the element that is not in a standard microprocessor?
   ( ) CPU ( ) Register ( ) Program Counter ( ) ADC

8. What are the four basic functions that a CPU can perform?
   (a) (b) (c) (d)

9. The fastest execution addressing mode is:
   ( ) Inherent Address ( ) Direct Address
   ( ) Indirect Address ( ) Index Address

10. The table look-up technique is best suitable for:
    ( ) Inherent Address ( ) Direct Address
     ( ) Indirect Address ( ) Index Address

11. The longest execution addressing mode is:
    ( ) Inherent Address ( ) Direct Address
     ( ) Indirect Address ( ) Index Address

12. How can you save the register contents when the program jumps to a subroutine?
    ( ) Use PC ( ) Use Index Registers & PUSH
     ( ) Use Stack Pointer & PUSH ( ) Use Interrupts

13. How do you set up the interrupt service routines?
    ( ) Set the stack pointer to the beginning address of the service routine.
    ( ) Set the program counter to the beginning address of the service routine.
    ( ) Set the specific vector to the beginning address of the service routine.
    ( ) Use jump statement to force program to go to service routine.

14. Which one is the most efficient way to ask the microprocessor to periodically perform the same task?
    ( ) Write a delay subroutine and task subroutine.
    ( ) Write the time and task routines wherever there is a need.
    ( ) Write interrupt service routine particularly for the periodic task.
    ( ) Use programmer timer device/function that combine with interrupt to perform the periodic task when it is needed.

15. What is the maximum range that an 8-bit microprocessor with 16 address lines can fully access?
    ( ) 16K ( ) 32K ( ) 64K ( ) 128K Bytes
16. Which register provides critical information for a microprocessor to perform the judging functions.
   ( ) Register A ( ) Register B ( ) Register X ( ) CCR

17. Which memory of the following needs to be refreshed periodically?
   ( ) ROM ( ) RAM ( ) Static Memory ( ) Dynamic Memory

18. Beginning of a routine program execution, which one of the follows will save all the registers contents automatically?
   ( ) Subroutine ( ) Interrupt Service Routine
   ( ) Both a & b ( ) None of the above

19. In question #18, the information (register content) is saved:
   ( ) According to the PC location
   ( ) According to the index register location
   ( ) According to where the stack pointer is pointing at
   ( ) According to the CCR status

Direction: Please use your own words to answer the following open ended questions.

20. List two characteristics to describe the differences between microprocessor & microcontroller.
   (1)
   
   (2)

21. List two characteristics to describe the differences between microprocessor & microcomputer.
   (1)
   
   (2)
22. List two practical applications of microprocessor digital I/O capability.

(1)

(2)

23. List two practical applications of microprocessor timer capability.

(1)

(2)

24. List two practical applications of microprocessor interrupt capability.

(1)

(2)

25. What are the minimum number of parts (peripheral devices), which exclude the microprocessor itself, needed to perform a single task control application (Assume the program is no larger than 200 bytes in length).

26. If a microprocessor system needs to communicate with a control center or another similar microprocessor system, state what hardware parts you need and how to implement this communication?

27. If a microprocessor system is implemented to control a DC servo-motor system (a closed loop control system), describe your plan in charts and sentences for your hardware and software.
28. What is(are) the difference(s) between RAM, ROM, EPROM, and EEPROM memories?

29. What is(are) the difference(s) between an 8-bit and 16-bit microprocessors.

30. List two characteristics that determine the speed of a microcomputer.

(1)

(2)

**Question for Posttest only**

31. What do you think of the 6809 in COCO and 68HC11 EVB (favor no opinion, or oppose)? Please state you opinions on (a) addressing modes (b) over all user friendly functions (c) interfacing capabilities (d) project implementations.
APPENDIX C. ASSEMBLER SOURCE CODES LIST
FOR CONTROL PROGRAMS
FILE NAME: STEP.EXE (Used in Experiment #1)  
FUNCTION: USE THE TIMEING LOOP AND SEQUENTIALLY TURN ON THE  
COILS IN THE TWO STEPPER MOTORS TO MAKE THEM ACTIVATE IN  
FOUR DIFFERENT ROTATIONS THE SELECTION OF THESE ROTATIONS*  
ARE DEPENDENT ON THE DIP SWITCHES SETTING  
AUTHOR: STEVE C. HSUONG  
DATE CREATED: 8/10/91  

THE ORIGINAL DATA SET UP *

ORG $DF00  
PORTA EQU $1000  
PORTB EQU $1004  
PORTC EQU $1003  
PORTD EQU $1008  
PORTE EQU $100A  
DDRC EQU $1007  
DDRD EQU $1009  

ORG $C000  
LDAA #$F0  
STAA DDRC  
AGAIN  
LDAA PORTC  
ANDA #$01  
CMPA #$01  
BEQ STOP  
LDAA PORTC  
ANDA #$04  
BEQ TURNL  
CMPA #$04  
BEQ TURNR  
ANDA #$08  
BEQ WARDB  
CMPA #$08  
BEQ WARDF  
BRA AGAIN  
 Turner JSR RTUN  
BRA AGAIN  
Turn NL JSR LTUN  
BRA AGAIN

THE MAIN PROGRAM STARTS HERE *

ORG $C000  
LDAA #$F0  
STAA DDRC  
AGAIN  
LDAA PORTC  
ANDA #$01  
CMPA #$01  
BEQ STOP  
LDAA PORTC  
ANDA #$04  
BEQ TURNL  
CMPA #$04  
BEQ TURNR  
ANDA #$08  
BEQ WARDB  
CMPA #$08  
BEQ WARDF  
BRA AGAIN  

RIGHT TURN SUBROUTINE
LOOP BACK
LEFT TURN SUBROUTINE
LOOP BACK
WARDF JSR FWAD ;FORWARD SUBROUTINE
BRA AGAIN ;LOOP BACK
WARDB JSR BWAD ;BACKWARD SUBROUTINE
BRA AGAIN ;LOOP BACK

* THE RIGHT TURN SUBROUTINE *
* ACTIVATE ONE MOTOR BY THE SEQUENCE OF 9 A 6 5 *
RTUN LDAA #$90
       STAA PORTB
       JSR TIME
       LDAA #$A0
       STAA PORTB
       JSR TIME
       LDAA #$60
       STAA PORTB
       JSR TIME
       LDAA #$50
       STAA PORTB
       JSR TIME
       RTS

* THE LEFT TURN SUBROUTINE *
* ACTIVATE ONE MOTOR BY THE SEQUENCE OF 9 5 6 A *
LTUN LDAA #$09
       STAA PORTB
       JSR TIME
       LDAA #$05
       STAA PORTB
       JSR TIME
       LDAA #$06
       STAA PORTB
       JSR TIME
       LDAA #$0A
       STAA PORTB
       JSR TIME
       RTS

* THE FORWARD SUBROUTINE, TWO STEPPER MOTORS TURN AT THE SAME *
* SPEED AND THE CAR GOES FORWARD *
* ACTIVATE TWO MOTORS BY THE SEQUENCE OF 9 A 6 5 AND 9 5 6 A *
FWAD LDAA #$99
       STAA PORTB
       JSR TIME
       LDAA #$A5
       STAA PORTB
       JSR TIME
       LDAA #$65
       STAA PORTB
       JSR TIME
       LDAA #$5A
       STAA PORTB
       JSR TIME
       RTS
THE BACKWARD SUBROUTINE, TWO STEPPER MOTORS TURN AT THE SAME SPEED BUT REVERSE AND THE CAR GOES BACKWARD
ACTIVATE TWO MOTORS BY THE SEQUENCE OF 9 5 6 A AND 9 A 6 5

BWDAD
LDAA #$99
STAA PORTB
JSR TIME
LDAA #$5A
STAA PORTB
JSR TIME
LDAA #$66
STAA PORTB
JSR TIME
LDAA #$A5
STAA PORTB
JSR TIME
RTS

STOP THE STEPPER MOTORS ACTION
STOP
LDAA #$00 ;DO NOT ENERGIZE ANY COIL IN THE
STAA PORTB ; STEPPER MOTORS
RTS

THE TIME DELAY ROUTINE
THERE ARE TWO DIFFERENT TIME DELAYS DEPEND ON THE SWITCH
SETTING WHICH IS CONNECTED TO PORTC BIT 1 (PC1)

LDAA PORTC ;GET THE SWITCHES SETTING
ANDA #$02 ;MASK OUT OTHER BIT SAVE BIT 1
BEQ LOSP ; IF BIT 1 CLEAR GO LOW SPEED
HISP
LDAB #$0F ;IF BIT 1 SET GO HIGH SPEED
BRA CONT
LOSP
LDAB #$FF
CONT
LDX #$OFAA
LP1
DEX
BNE LP1
DECB
BNE CONT
RTS

* THE ORIGINAL DATA SET UP *
ORG $DFOO ;SET THE STARTING ADDR. OF THE
* REAL KEY CODE TABLE
PORTA EQU $1000 ;PORT A FOR TIMER I/O SIGNAL
PORTB EQU $1004 ;PORT B FOR OUTPUT ONLY
PORTC EQU $1003 ;PORT C FOR IN/OUT PUT DEPENDS ON
; DDRC'S BITS VALUE
PORTD EQU $1008 ;PORT D FOR IN/OUT PUT FOR BIT 0 -
; 5 ONLY
PORTE EQU $100A ;PORT E FOR A/D ANALOG SIGNAL INPUT
DDRC EQU $1007 ;THE DATA DIRECTION REGISTER OF
; THE PORT C
DDRD EQU $1009 ;THE DATA DIRECTION REGISTER OF
; THE PORT D, BUT BIT 7,6 ARE
; NOT CONTROLLABLE
PIOC EQU $1002 ;PARALLEL CONTROL REGISTER FOR STRA
TEMP EQU $D000 ;TEMPORARY STORING MEMORY POSITION
COUNT1 EQU $D001 ;COUNTER 1 FOR N/S PED REPEAT INFO
COUNT2 EQU $D002 ;COUNTER 2 FOR W/E PED REPEAT INFO
PNTER EQU $DE00

* THE MAIN PROGRAM STARTS HERE *
ORG $C000 ;SET THE STARTING ADDR. OF THE MAIN
; PROGRAM
LDAA #$FF ;CONFIGURE PORT
LDAA #$50 ;CONFIGURE STROBE A FOR INTERRUPT
STAA PIOC ; ENABLE, FULL HANDSHAKE MODE, AND

* THE MAIN PROGRAM STARTS HERE *
ORG $C000 ;SET THE STARTING ADDR. OF THE MAIN
; PROGRAM
LDAA #$50 ;CONFIGURE STROBE A FOR INTERRUPT
STAA PIOC ; ENABLE, FULL HANDSHAKE MODE, AND
Hi TO Lo EDGE TRIGGER
LDAA #$F0 ;SET PORT C BIT 0-3 INPUT LINES AND
STAA DDRC ; BIT 4-7 OUTPUT LINES
STAA PORTC
CLC ;ENABLE ALL THE INTERRUPT
AGAIN
CLR COUNT1 ;INITIAL COUNTER 1
CLR COUNT2 ;INITIAL COUNTER 2
JSR STOPALL ;ALL THE LIGHTS ARE RED
LDAA #43 ;LEFT TURN FOR N/S AND RED FOR W/E
; RED
STAA PORTB
JSR TIME
NSCYC
LDAA #$12 ;GREEN FOR N/S AND RED FOR W/E
STAA PORTB
JSR TIME
JSR CHECKNS
NSCOUNT
LDAA #$06 ;YELLOW FOR N/S AND RED FOR W/E
STAA PORTB
JSR TIME
JSR STOPALL ;ALL THE LIGHTS ARE RED
LDAA #$83 ;LEFT TURN FOR W/E AND RED FOR N/S
STAA PORTB
JSR TIME

WECYC LDAA #$21 ;GREEN FOR W/E AND RED FOR N/S
STAA PORTB
JSR TIME
JSR CHECKWE

WECOUNT LDAA #$09 ;YELLOW FOR W/E AND RED FOR N/S
STAA PORTB
JSR TIME
BRA AGAIN

* REGULAR TIME DELAY ROUTINE *
TIME LDY #$FFFF
LO2 LDX #$00AF
LO1 DEX
BNE LO1
DEY
BNE LO2
RTS

* SHORT TIME DELAY ROUTINE FOR PEDESTRATION CROSSING *
STIME LDY #$FFFF
LO4 LDX #$005F
LO3 DEX
BNE LO3
DEY
BNE LO4
RTS

* STOP ALL THE TRAFFIC, ALL THE LIGHTS ARE RED *
STOPALL LDAA #$03
STAA PORTB
JSR TIME
RTS

* CHECK THE PEDESTRAIN INFORMATION IN THE NORTH/SOUTH SIDE *
* SEQUENCING FUNCTIONS *
CHECKNS LDAB PORTC ;CHECK THE PEDESTRAIN INFORMATION
BEQ END1 ; IF NONE THEN BACK NORMAL
STAB TEMP ;TEMPORARY STORE THE PED INFO
LDAA #$00 ;CLEAR THE D LATCH PED INFO
STAA PORTC
ANDB #$01 ;CHECK THE WEST/EAST SIDE PED INFO
CMPB #$01
BEQ WEBCRO1 ;GO TO THE W/E CHANGED SEQUENCE
LDAB TEMP ;CHECK THE NORT/SOUTH SIDE PED INFO
ANDB #$02
CMPB #$02
BEQ NSCRO1 ;GO TO THE N/S CHANGED SEQUENCE

WEBCRO1 LDAA PIOC ;DUMMY INSTRUCITON TO CLEAR STROBE
LDAA PORTCL ; A INTERRUPT FLAG
LDAA #$09 ;FINISH N/S YELLOW BUT SHORT TIME
STAA PORTB
JSR STIME
JSR STOPALL ; SKIP THE N/S LEFT TURN SIGNAL
JMP WECYC ; RESUME THE W/E GREEN CYCLE

NSCRO1
LDAA PIOC ; DUMMY INSTRUCTION TO CLEAR STROBE
LDAA PORTCL ; A INTERRUPT FLAG
INC COUNT1 ; INCREASE COUNTER 1 FOR LOOPTING 4

* 
LDAA COUNT1 ; CHECK THE COUNTER FOR 4
CMPA #$04
BEQ END1 ; LOOP 4 TIMES THEN CONTINUE ON

* 
JMP NSCYC ; RESUME THE N/S GREEN CYCLE

END1 RTS

* CHECK THE PEDESTRAIN INFORMATION IN THE WEST/EAST SIDE *
* SEQUENCING FUNCTIONS *

CHECKWE
LDAB PORTC ; CHECK THE PEDESTRAIN INFORMATION
BEQ END2 ; IF NONE THEN BACK NORMAL
STAB TEMP ; TEMPORARY STORE THE PED INFO
LDAA #$00 ; CLEAR THE D LATCH PED INFO
STAA PORTC
ANDB #$02 ; CHECK THE WEST/EAST SIDE PED INFO
CMPB #$02
BEQ NSCRO2 ; GO TO THE W/E CHANGED SEQUENCE
LDAB TEMP ; CHECK THE NORT/SOUTH SIDE PED INFO
ANDB #$01
CMPB #$01
BEQ WECR02 ; CLEAR THE D LATCH PED INFO
NSCRO2
LDAA PIOC ; DUMMY INSTRUCTION TO CLEAR STROBE
LDAA PORTCL ; A INTERRUPT FLAG
LDAA #$06 ; FINISH W/E YELLOW BUT SHORT TIME
STAA PORTB
JSR STIME
JSR STOPALL ; SKIP THE N/S LEFT TURN SIGNAL

WECR02
LDAA PIOC ; DUMMY INSTRUCTION TO CLEAR STROBE
LDAA PORTCL ; A INTERRUPT FLAG
JMP WECYC ; RESUME THE W/E GREEN CYCLE
INC COUNT2 ; INCREASE COUNTER 1 FOR LOOPTING 4

* 
LDAA COUNT1 ; CHECK THE COUNTER FOR 4
CMPA #$04
BEQ END4 ; LOOP 4 TIMES THEN CONTINUE ON

* 
JMP NSCYC ; RESUME THE N/S GREEN CYCLE

END2 RTS
FILE NAME: KEY2.EXE (Used in Experiment #3)
FUNCTION: USE THE SOFTWARE DEBOUNCING TO FIND THE KEY
PRESSED IN THE KEY PAD AND CONVERT TO A REAL NUMBER THEN
STORE INTO CONSECUTIVE MEMORY LOCATION START FROM CODE
ADDRESSES
AUTHOR: STEVE C. HSIUNG
DATE CAREATED: 7/21/91

THE ORIGINAL DATA SET UP

ORG $DF00 ;SET THE STARTING ADDR. OF THE
; REAL KEY CODE TABLE
PORTA EQU $1000 ;PORT A FOR TIMER I/O SIGNAL
PORTB EQU $1004 ;PORT B FOR OUTPUT ONLY
PORTC EQU $1003 ;PORT C FOR IN/OUT PUT DEPENDS ON
; DDRC'S BITS VALUE
PORTD EQU $1008 ;PORT D FOR IN/OUT PUT FOR BIT 0 -
; 5 ONLY
PORTE EQU $100A ;PORT E FOR A/D ANALOG SIGNAL INPUT
OPTION EQU $1039 ;SYSTEM CONFIGURATION OPTIONS
DDRC EQU $1007 ;THE DATA DIRECTION REGISTER OF
; THE PORT C
DDRD EQU $1009 ;THE DATA DIRECTION REGISTER OF THE
; PORT D, BUT BIT 7, 6 ARE NOT
; CONTROLLABLE
ADCTL EQU $30 ;A/D CONTROL/STATUS REGISTER
ADR1 EQU $31 ;A/D RESULT REGISTER 1
ADR2 EQU $32 ;A/D RESULT REGISTER 2
ADR3 EQU $33 ;A/D RESULT REGISTER 3
ADR4 EQU $34 ;A/D RESULT REGISTER 4
TMSK2 EQU $24 ;TIMER INTERRUPT MASK2
RTIME EQU $00EB ;REAL TIME INTERRUPT VECTOR
PACTL EQU $26 ;PULSE ACCUMULATOR CONTROL REGISTER
TFLG2 EQU $25 ;TIMER INTERRUPT FLAG2
CODE EQU $D010 ;THE MEMORY WHERE KEY CODE STORED
TEMP EQU $D000 ;TEMPORARY STORING MEMORY POSITION
PNTER EQU $DE00
KYTEBL FCB $EE, $DE, $BE, $7E, $ED, $DD, $BD, $7D
FCB $EB, $DB, $BB, $7B, $E7, $D7, $B7, $77

THE MAIN PROGRAM STARTS HERE

ORG $CO00 ;SET THE STARTING ADDR. OF THE MAIN
; PROGRAM
LDAA #$00 ;CONFIGURE ALL LINES OF PORTC AS
STAA DDRC ; INPUT PORT
CLC ; CLEAR THE BEGINNING C FLAG
LDY #CODE ;SET UP A POINTER POINTING TO THE
STY PNTER ; MEMORY CODE AND USE IT LATER FOR
; INDIRECT ADDRESSING
TRY       LDAA PORTC       ;READ KEYPAD INFOR FROM PORTC
         CMPA #$FF       ;CHECK IF ANY KEY PRESSED
         BCC TRY       ;IF NO KEY PRESSED CONTINUE WAITING
AGAIN     LDAA PORTC       ; THE KEY BOUNCING SETTLED DOWN
         DECB           ; THE DELAY LOOP IS ABOUT 20 mS
         BNE AGAIN
         STAA TEMP       ;STORE THE KEY INFOR TO A TEMPORARY
         *              ; MEMORY POSITION
         LDY PNTER       ;SET Y AS THE POINTER FOR THE
         *              ; CONSECUTIVE CODE STORAGE
         LDX #KYTBL      ;LOACTE THE KEY CODE IN KEYTABLE
FIND      LDAA 0,X        ;FIND THE MATCHED KEY CODE
         INX
         CMPA TEMP       ;IF MATCHED SAVE THE KEY INFOR
         BEQ SAVE       ; IF NOT MATCHED CHECK THE END OF
         *              ; TABLE
         CPX #$DF10      ;IF NOT MATCHED CHECK THE END OF
         BNE FIND       ; LOOP THRU THE ENTIRE TABLE
         LDAB #$FF       ;USE $FF AS THE SIGN OF THE END OF
         STAB CODE       ; TABLE
         BRA TRY
SAVE      XGDX           ;MOVE KEYTABLE POINTER TO D REG
         SUBD #KYTBL+1    ;CONVERT THE KEY CODE TO A REAL
         STAB 0,Y         ; EQUIVALENT NUMBER STORE IN
         *              ; POINTER MEMORY POSITON
         INY             ;ADVANCE THE POINTER FOR NEXT KEY
         *              ; CODE
         STY PNTER       ;RENEW THE POINTER POSITION
         LDY #$DF0C      ;SET UP A TIME DELAY FOR ABOUT 0.2S
LO        DEY             ; FOR THE BOUNCING KEY SETTLE DOWN
         BNE LO          ; THIS CAN BE REPLACED BY REAL
         *              ; TIME INTERRUPT TIME INTERVAL
         BRA TRY

****************************************************************************************************************************************
*                                                                                                                                         *
* FILE NAME: HARDKEY.EXE (Used in Experiment #3)                                                                                         *
* FUNCTION: USE THE HARDWARE DEBOUNCING WITH THE HELP OF 74C922 TO FIND THE KEY PRESSED IN THE KEY PAD THE HARDWARE WILL GENERATE AN  *
* INTERRUPT TO HC11 THE SERVICE ROUTINE WILL READ THE INPUT NUMBER FROM PORTC BIT 0-3 THEN STORE INTO MEMORY LOCATION NAMED "CODE" IT STORE *
* ONE NUMBER AT A TIME                                                                                                                  *
* AUTHOR: STEVE C. HSIUNG                                                                                                              *
* DATE CREATED: 8/25/91                                                                                                                *
****************************************************************************************************************************************
* THE ORIGINAL DATA SET UP *

<table>
<thead>
<tr>
<th>ORG</th>
<th>$DF00</th>
</tr>
</thead>
<tbody>
<tr>
<td>PORTA</td>
<td>$1000</td>
</tr>
<tr>
<td>PORTB</td>
<td>$1004</td>
</tr>
<tr>
<td>PORTC</td>
<td>$1003</td>
</tr>
<tr>
<td>PORTD</td>
<td>$1008</td>
</tr>
<tr>
<td>PORTE</td>
<td>$100A</td>
</tr>
<tr>
<td>OPTION</td>
<td>$1039</td>
</tr>
<tr>
<td>DDRC</td>
<td>$1007</td>
</tr>
<tr>
<td>DDRD</td>
<td>$1009</td>
</tr>
<tr>
<td>ADCTL</td>
<td>$30</td>
</tr>
<tr>
<td>ADR1</td>
<td>$31</td>
</tr>
<tr>
<td>ADR2</td>
<td>$32</td>
</tr>
<tr>
<td>ADR3</td>
<td>$33</td>
</tr>
<tr>
<td>ADR4</td>
<td>$34</td>
</tr>
<tr>
<td>TMSK2</td>
<td>$24</td>
</tr>
<tr>
<td>RTIME</td>
<td>$00EB</td>
</tr>
<tr>
<td>PACTL</td>
<td>$26</td>
</tr>
<tr>
<td>TFLG2</td>
<td>$25</td>
</tr>
<tr>
<td>IRQ</td>
<td>$00EE</td>
</tr>
<tr>
<td>PIOC</td>
<td>$02</td>
</tr>
<tr>
<td>PORTCL</td>
<td>$1005</td>
</tr>
<tr>
<td>CODE</td>
<td>$D010</td>
</tr>
<tr>
<td>TEMP</td>
<td>$D000</td>
</tr>
<tr>
<td>PNTER</td>
<td>$DE00</td>
</tr>
<tr>
<td>KYTBL</td>
<td>$EE,$DE,$BE,$7E,$ED,$BD,$7D</td>
</tr>
<tr>
<td></td>
<td>$EB,$DB,$BB,$7B,$E7,$D7,$B7,$77</td>
</tr>
</tbody>
</table>

* THE MAIN PROGRAM STARTS HERE *

<table>
<thead>
<tr>
<th>ORG</th>
<th>$C000</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDAA</td>
<td>#$00</td>
</tr>
<tr>
<td>STAA</td>
<td>DDRC</td>
</tr>
<tr>
<td>CLC</td>
<td></td>
</tr>
</tbody>
</table>

* SET UP THE IRQ INTERRUPT CONNECTIONS *

<table>
<thead>
<tr>
<th>LDS</th>
<th>#$0047</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDA</td>
<td>#$7E</td>
</tr>
<tr>
<td>STAA</td>
<td>IRQ</td>
</tr>
<tr>
<td>LDY</td>
<td>#$STRAIN</td>
</tr>
<tr>
<td>STY</td>
<td>IRQ+1</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>AGAIN</td>
<td>#$1000</td>
</tr>
<tr>
<td>BSET</td>
<td>PIOC,X #$D2</td>
</tr>
</tbody>
</table>

* THE MAIN PROGRAM STARTS HERE *
CLI ;ENABLE THE INTERRUPT
BRA AGAIN

* THE IRQ INTERRUPT SERVICE ROUTINE *
STRAIN LDAA PORTC ;READ THE DEBOUNCED KEY CODE AND
STAA TEMP ; STORE THE RESULT INTO MEMORY
STAA CODE
LDAA PIO,C,X ;DUMMY INSTRUCTIONS CLEAR THE STRA
LDAA PORTCL ; INTERRUPT FLAG IN THE FULL-INPUT
* ; HANDSHAKE MODE
END RTI

***************************************************************
* *
* FILE NAME: LCD.EXE (Used in Experiment #4) *
* FUNCTION: DISPLAY INFO ON LCD MODULE AND CONTROL VARIOUS *
* FUNCTIONS OF THE DISPLAYING: PORTB TO D7-D0, PORTD BIT 2- *
* TO E R/W RS RESPECTIVELY *
* AUTHOR: STEVE C. HSIUNG *
* DATE CREATED: 7/25/91 *
* *
***************************************************************

* THE ORIGINAL DATA SET UP *
PORTA EQU $1000 ;PORT A FOR TIMER I/O SIGNAL
PORTB EQU $1004 ;PORT B FOR OUTPUT ONLY
PORTC EQU $1003 ;PORT C FOR IN/OUT PUT DEPENDS ON *
* ; DDRC'S BITS VALUE
PORTD EQU $1008 ;PORT D FOR IN/OUT PUT FOR BIT 0 *
* ; -5 ONLY
PORTE EQU $100A ;PORT E FOR A/D ANALOG SIGNAL INPUT
OPTION EQU $1039 ;SYSTEM CONFIGURATION OPTIONS
DDRC EQU $1007 ;THE DATA DIRECTION REGISTER OF THE *
* ; PORT C
DDRD EQU $1009 ;THE DATA DIRECTION REGISTER OF THE *
* ; PORT D, BUT BIT 7, 6 ARE NOT *
* ; CONTROLLABLE
ADCTL EQU $1030 ;A/D CONTROL/STATUS REGISTER
ADR1 EQU $1031 ;A/D RESULT REGISTER 1
ADR2 EQU $1032 ;A/D RESULT REGISTER 2
ADR3 EQU $1033 ;A/D RESULT REGISTER 3
ADR4 EQU $1034 ;A/D RESULT REGISTER 4

* SET MESSAGE BANK STARTS AT $DE00 *
ORG $DE00
MESG1 FCC '*WELCOME TO THE HOUSE OF STEVE & ELAINE*'
MESG2 FCC '* Make Yourself at Home & Enjoy Staying*'
MESG3 FCC '* Come to Explore the Fun of uP World *'

* THE MAIN PROGRAM STARTS HERE THIS PROGRAM USES PORTB AND *
* PORTC BIT 0-2 AS OUTPUT LINES *
ORG $C000 ;SET THE STARTING ADDR. OF THE PROGRAM

LDAA #$FF ;SET PORTC AS OUTPUT LINES BUT ONLY USE BIT 0-2 LINES
STAA DDRC
STAA DDRD

*_INITIALIZATION OF LCD MODULE DISPLAY FUNCTIONS*

LDAA #$00 ;WRITE TO INSTRUCTION REGISTER
STAA PORTD
JSR TIME ;MAKE A DELY WAIT FOR LCD TO ACCEPT
LDAA #$38 ;FUNCTION SET 8 BIT DATA DUAL LINES
STAA PORTB
JSR PULSE ;PULSE THE E LINE FOR INSTRUCTION REGISTER
JSR TIME ;SHORT TIME DELAY
LDAA #$0C ;DISPLAY ON, NO CURSOR, NO BLINKING
STAA PORTB
JSR PULSE
JSR TIME
LDAA #$01 ;DISPLAY CLEAR
STAA PORTB
JSR PULSE
JSR TIME
LDAA #$06 ;ADDR COUNTER INCREMENT AFTER EACH DISPLAY
STAA PORTB ; DISPLAY
JSR PULSE
JSR TIME

* START TO DISPLAY MESSAGES *

AGAIN LDAA #$00 ;WRITE TO INSTRUCTION REGISTER
STAA PORTD
JSR TIME ;MAKE DELAY WAIT FOR LCD TO ACCEPT
LDAA #$01 ;CLEAR THE OLD DISPLAY
STAA PORTB
JSR PULSE
JSR TIME
LDAA #$02 ;SET ADDR COUNTER TO ZERO, MOVE CURSOR TO HOME POSITION
STAA PORTB
JSR PULSE
JSR TIME

* DISPLAY 1ST MESSAGES *

LDAA #$10 ;WRITE TO DISPLAY DATA REGISTER
STAA PORTD
JSR TIME ;MAKE DELAY WAIT FOR LCD TO ACCEPT
LDX #MESG1 ;GET 1ST DISPLAY
dspy1 LDAB 0,X ;DISPLAY THE FIRST MESSAGE
INX
STAB PORTB
JSR PULSE2 ;PULSE FOR DATA REGISTER
JSR TIME2 ;LONG TIME DELAY
CPX #MESG1+$50
BNE   DSPY1
JSR   SHFL
LDA 0C ;CONTINUOUS DELAY FOR STABLE
LOA
JSR   TIME2 ; DISPLAYING
DECA
BNE   LOA

* DISPLAY THE BLINKING MESSAGES *
LDA 0 ;WRITE TO INSTRUCTION REGISTER
STA  PORTD
JSR   TIME ;MAKE DELAY WAIT FOR LCD TO ACCEPT
LDA 01 ;CLEAR THE OLD DISPLAY
STA  PORTB
JSR   PULSE
JSR   TIME
LDA 02 ;SET ADDR COUNTER TO ZERO, MOVE
STA  PORTB ; CURSOR TO HOME POSITION
JSR   PULSE
JSR   TIME
LDA 10 ;WRITE TO DISPLAY DATA REGISTER
STA  PORTD
JSR   TIME ;MAKE DELAY WAIT FOR LCD TO ACCEPT
LDX  MESS2 ;GET 2ND DISPLAY
DSPY2
LDA 0,X ;DISPLAY THE 2ND MESSAGE
INX
STA  PORTB
JSR   PULSE2 ;PULSE FOR DATA REGISTER
JSR   TIME2 ;LONG TIME DELAY
CPX  MESS2+28
BNE   DSPY2
LDA 0 ;WRITE TO INSTRUCTION REGISTER
STA  PORTD
JSR   TIME ;MAKE DELAY WAIT FOR LCD TO ACCEPT
LDA 0 ;CHANGE TO 2ND FOR DISPLAYING
STA  PORTB
JSR   PULSE
JSR   TIME
LDA 10 ;WRITE TO DISPLAY DATA REGISTER
STA  PORTD
JSR   TIME ;MAKE DELAY WAIT FOR LCD TO ACCEPT
LDX  MESS3
DSPY3
LDA 0,X ;DISPLAY THE 3RD MESSAGE
INX
STA  PORTB
JSR   PULSE2 ;PULSE FOR DATA REGISTER
JSR   TIME2 ;LONG TIME DELAY
CPX  MESS3+27
BNE   DSPY3
JSR   BLINK
JMP   AGAIN

* ALL THE SUBROUTINES START HERE *
* SHIFT THE DISPLAY LEFT CONTINUOUSLY FOR EVERY LETTER *
SHFL LDX #$0050 ; CONTINUOUSLY SHIFT LEFT FOR 80
LO5 LDAA #$00 ; POSITIONS WHICH IS EQUIVALENT
STAA PORTD ; TO TWICE AS DISPLAY LENGTH
JSR TIME
LDAA #$18 ; SHIFT THE ENTIRE DISPLAY TO THE
STAA PORTB ; LEFT
JSR PULSE
JSR TIME
LDAA #$0C ; CONTINUOUS DELAY FOR STABLE
LO9 JSR TIME2 ; DISPLAYING
DECA
BNE LO9
DEX
BNE LO5
RTS

* BLINK THE DISPLAY FOR 10 TIMES *
BLINK LDX #$000A
LO8 LDAA #$00 ; WRITE TO INSTRUCTION REGISTER
. STAA PORTD
JSR TIME ; DELAY WAIT FOR LCD TO ACCEPT
LDAA #$08 ; DISPLAY OFF
STAA PORTB
JSR PULSE
JSR TIME
LDAA #$10
LO6 JSR TIME2
DECA
BNE LO6
LDAA #$0C ; RESUME DISPLAY ON, NO CURSOR,
STAA PORTB ; NO BLINKING
JSR PULSE
JSR TIME
LDAA #$23
LO7 JSR TIME2
DECA
BNE LO7
DEX
BNE LO8
RTS

* PULSE THE E LINE FOR INSTRUCTION REGISTER WRITING *
PULSE LDAB #$04
STAB PORTD
LDAB #$00
STAB PORTD
RTS

* PULSE FOR DISPLAY DATA REGISTER WRITING *
PULSE2 LDAB #$14
STAB PORTD
LDAB #$10
STAB PORTD
**SHORT TIME DELAY FOR PULSING E LINE**

```assembly
TIME    LDAB  #$FF
LO1    DECB
       BNE  LO1
       RTS
```

**LONG TIME DELAY FOR DISPLAYING**

```assembly
TIME2   LDAB  #$FF
LO4    LDY  #$0035
LO3    DEY
       BNE  LO3
       DECB
       BNE  LO4
       RTS
```

---

**FILE NAME:** LCDWspi.EXE (Used in Experiment #4)

**FUNCTION:** DISPLAY INFO ON LCD MODULE AND CONTROL VARIOUS

**FUNCTIONS OF THE DISPLAYING:** PORTD MISO (D3), SCK (D4), RESET

**PORTC BIT 0-2 TO E R/W RS RESPECTIVELY AND 74164 USED**

**FOR SPI**

**AUTHOR:** STEVE C. HSIUNG

**DATE CREATED:** 8/21/91

---

**THE ORIGINAL DATA SET UP**

```assembly
PORTA  EQU  $1000 ; PORT A FOR TIMER I/O SIGNAL
PORTB  EQU  $1004 ; PORT B FOR OUTPUT ONLY
PORTC  EQU  $1003 ; PORT C FOR IN/OUT PUT DEPENDS ON
       ; DDRC'S BITS VALUE
PORTD  EQU  $1008 ; PORT D FOR IN/OUT PUT FOR BIT 0 -
       ; 5 ONLY
PORTE  EQU  $100A ; PORT E FOR A/D ANALOG SIGNAL INPUT
OPTION EQU  $1039 ; SYSTEM CONFIGURATION OPTIONS
DDRC   EQU  $1007 ; THE DATA DIRECTION REGISTER OF THE
       ; PORT C
DDRD   EQU  $1009 ; THE DATA DIRECTION REGISTER OF THE
       ; PORT D, BUT BIT 7, 6 ARE NOT
       ; CONTROLLABLE
ADCTL  EQU  $1030 ; A/D CONTROL/STATUS REGISTER
ADR1   EQU  $1031 ; A/D RESULT REGISTER 1
ADR2   EQU  $1032 ; A/D RESULT REGISTER 2
ADR3   EQU  $1033 ; A/D RESULT REGISTER 3
ADR4   EQU  $1034 ; A/D RESULT REGISTER 4
SPCR   EQU  $1028 ; SPI CONTROL REGISTER
SPSR   EQU  $1029 ; SPI STATUS REGISTER
SPDR   EQU  $102A ; SPI DATA REGISTER
```
* SET MESSAGE BANK STARTS AT $DE00 *

ORG $DE00

MESG1 FCC 'WELCOME TO THE HOUSE OF STEVE & ELAINE''
MESG2 FCC 'Make Yourself at Home & Enjoy Staying''
MESG3 FCC 'Come to Explore the Fun of the World''

' Steve Law Said: Drink and Drive is "FUN"'

* THE MAIN PROGRAM STATS THERE *
* THIS PROGRAM USES PORTB AND PORTC BIT 0-2 AS OUTPUT LINES *

ORG $C000 ;SET THE STARTING ADDR. OF THE PROGRAM

LDAA #$FF ;SET PORTC AS OUTPUT LINES BUT ONLY
STAA DDRC ; USE BIT 0-2 LINES
LDAA #$2F ;SS-Hi, SCK-Lo, MOSI-Hi
STAA PORTD
LDAA #$38 ;SS, SCK, MOSI-OUTPUTS; MISO, TXD,
STAA DDRD ; RXD-INPUTS
LDAA #$5F ;SPI ON AS MASTER, CPHA=1, CPOL=1
STAA SPCR ; E/32 CLOCK RATE

* INITIALIZATION OF LCD MODULE DISPLAY FUNCTIONS *

LDAA #$00 ;WRITE TO INSTRUCTION REGISTER
STAA PORTC
JSR TIME ;MAKE A DELAY WAIT FOR LCD TO ACCEPT
LDAA #$38 ;FUNCTION SET 8 BIT DATA DUAL LINES
JSR DUMP ;DUMP DATA TO 74164 AND DISPLAY
JSR PULSE ;PULSE THE E LINE FOR INSTRUCTION REGISTER

JSR TIME ;SHORT TIME DELAY
LDAA #$0C ;DISPLAY ON, NO CURSOR, NO BLINKING
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$01 ;DISPLAY CLEAR
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$06 ;ADDR COUNTER INCREMENT AFTER EACH
JSR DUMP
JSR PULSE
JSR TIME

* START TO DISPLAY MESSAGES *

AGAIN LDAA #$00 ;WRITE TO INSTRUCTION REGISTER
STAA PORTC
JSR TIME ;MAKE DELAY WAIT FOR LCD TO ACCEPT
LDAA #$01 ;CLEAR THE OLD DISPLAY
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$02 ;SET ADDR COUNTER TO ZERO, MOVE
JSR DUMP
JSR PULSE
JSR TIME

* DISPLAY 1ST MESSAGES *
LDAA #$04 ;WRITE TO DISPLAY DATA REGISTER
STAA PORTC
JSR TIME ;MAKE DELAY WAIT FOR LCD TO ACCEPT
LDX #$MESG1 ;GET 1ST DISPLAY
DSPY1 LDAA 0,X ;DISPLAY THE FIRST MESSAGE
JSR DUMP
JSR PULSE2 ;PULSE FOR DATA REGISTER
JSR TIME2 ;LONG TIME DELAY
INX
CPX #$MESG1+$50
BNE DSPY1
JSR SHFL
LDAA #$0C ;CONTINUOUS DELAY FOR STABLE
LOA JSR TIME2 ; DISPLAYING
DECA
BNE LOA

* DISPLAY THE BLINKING MESSAGES *
LDAA #$00 ;WRITE TO INSTRUCTION REGISTER
STAA PORTC
JSR TIME ;MAKE DELAY WAIT FOR LCD TO ACCEPT
LDAA #$01 ;CLEAR THE OLD DISPLAY
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$02 ;SET ADDR COUNTER TO ZERO, MOVE
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$04 ;WRITE TO DISPLAY DATA REGISTER
STAA PORTC
JSR TIME ;MAKE DELAY WAIT FOR LCD TO ACCEPT
LDX #$MESG2 ;GET 2ND DISPLAY
DSPY2 LDAA 0,X ;DISPLAY THE 2ND MESSAGE
JSR DUMP
JSR PULSE2 ;PULSE FOR DATA REGISTER
JSR TIME2 ;LONG TIME DELAY
INX
CPX #$MESG2+$28
BNE DSPY2
LDAA #$00 ;WRITE TO INSTRUCTION REGISTER
STAA PORTC
JSR TIME ;MAKE DELAY WAIT FOR LCD TO ACCEPT
LDAA #$C0 ;CHANGE TO 2ND FOR DISPLAYING
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$04 ;WRITE TO DISPLAY DATA REGISTER
STAA PORTC
JSR TIME ; MAKE DELAY WAIT FOR LCD TO ACCEPT
LDX #MESG3
DSPY3 LDAA 0,X ; DISPLAY THE 3RD MESSAGE
JSR DUMP
JSR PULSE2 ; PULSE FOR DATA REGISTER
JSR TIME2 ; LONG TIME DELAY
INX
CPX #MESG3+$27
BNE DSPY3
JSR BLINK
JMP AGAIN

* ALL THE SUBROUTINES START HERE *
* SHIFT THE DISPLAY LEFT CONTINUOUSLY FOR EVERY LETTER *
SHFL LDX #$0050 ; CONTINUOUSLY SHIFT LEFT FOR 80
L05 LDAA #$00 ; POSITIONS WHICH IS EQUIVALENT
STAA PORTC ; TO TWICE AS DISPLAY LENGTH
JSR TIME
LDAA #$18 ; SHIFT THE ENTIRE DISPLAY TO THE
JSR DUMP ; LEFT
JSR PULSE
JSR TIME
LDAA #$60 ; CONTINUOUS DELAY FOR STABLE
L09 JSR TIME2 ; DISPLAYING
DECA
BNE L09
DEX
BNE L05
RTS

* BLINK THE DISPLAY FOR 10 TIMES *
BLINK LDX #$000A
L08 LDAA #$00 ; WRITE TO INSTRUCTION REGISTER
STAA PORTC
JSR TIME ; DELAY WAIT FOR LCD TO ACCEPT
LDAA #$08 ; DISPLAY OFF
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$10
L06 JSR TIME2
DECA
BNE L06
LDAA #$0C ; RESUME DISPLAY ON, NO CURSOR,
JSR DUMP ; NO BLINKING
JSR PULSE
JSR TIME
LDAA #$23
L07 JSR TIME2
DECA
BNE L07
DEX
BNE L08
RTS

* PULSE THE E LINE FOR INSTRUCTION REGISTER WRITING *
PULSE
LDAB #$01
STAB PORTC
LDAB #$00
STAB PORTC
RTS

* PULSE FOR DISPLAY DATA REGISTER WRITING *
PULSE2
LDAB #$05
STAB PORTC
LDAB #$04
STAB PORTC
RTS

* SHORT TIME DELAY FOR PULSING E LINE *
TIME
LDAB #$FF
LO1
DECB
BNE LO1
RTS

* LONG TIME DELAY FOR DISPLAYING *
TIME2
LDAB #$FF
LO4
LDY #$0035
LO3
DEY
BNE LO3
DECB
BNE LO4
RTS

* DUMP THE DATA TO 74164 SHIFT REGISTER AND WAIT FOR HC11 *
* ACCEPTANCE *
DUMP
STAA SPDR ;WRITE DATA SPI DATA REG TO XFER
WAIT
LDAA SPSR ;CHECK FOR SPIF FALG TO CHECK THE
BPL WAIT ; END OF THE XFER
RTS

***************************************************************
* *
* FILE NAME: HKYLDSPI.EXE (Used in Experiment #5) *
* FUNCTION: USE THE KEYPAD AND 74C922 TO INPUT THE CHOICES OF *
* CONTROLLING THE LCD MODULE DISPLAY FUNCTIONS PORTC BIT *
* 0-3 TO 74C922, PORTD MISO (D3), SCK(D4), RESET THRU SPI *
* TO 74164 AND PORTB BIT 0-2 TO E R/W RS RESPECTIVELY AS *
* THE OUTPUT LINES, PORTC AS THE INPUT LINES THRU 74C922 TO *
* GET THE CHOICES INFROMATION *
* AUTHOR: STEVE C. HSIUNG *
* DATE CREATED: 11/21/91 *
*
***************************************************************
* THE ORIGINAL DATA SET UP *
PORTA  EQU  $1000  ;PORT A FOR TIMER I/O SIGNAL
PORTB  EQU  $1004  ;PORT B FOR OUTPUT ONLY
PORTC  EQU  $1003  ;PORT C FOR IN/OUT PUT DEPENDS ON
*  ;DDRC'S BITS VALUE
PORTD  EQU  $1008  ;PORT D FOR IN/OUT PUT FOR BIT 0 -
*  5 ONLY
PORTE  EQU  $100A  ;PORT E FOR A/D ANALOG SIGNAL INPUT
OPTION  EQU  $1039  ;SYSTEM CONFIGURATION OPTIONS
DDRC  EQU  $1007  ;THE DATA DIRECTION REGISTER OF THE
*  ;PORT C
DDRD  EQU  $1009  ;THE DATA DIRECTION REGISTER OF THE
*  ;PORT D, BUT BIT 7,6 ARE NOT
*  CONTROLLABLE
ADCTL  EQU  $1030  ;A/D CONTROL/STATUS REGISTER
ADR1  EQU  $1031  ;A/D RESULT REGISTER 1
ADR2  EQU  $1032  ;A/D RESULT REGISTER 2
ADR3  EQU  $1033  ;A/D RESULT REGISTER 3
ADR4  EQU  $1034  ;A/D RESULT REGISTER 4
TMSK2  EQU  $24  ;TIMER INTERRUPT MASK2
RTIME  EQU  $00EB  ;REAL TIME INTERRUPT VECTOR
PACTL  EQU  $26  ;PULSE ACCUMULATOR CONTROL REGISTER
TFLG2  EQU  $25  ;TIMER INTERRUPT FLAG2
SPCR  EQU  $1028  ;SPI CONTROL REGISTER
SPSR  EQU  $1029  ;SPI STATUS REGISTER
SPDR  EQU  $102A  ;SPI DATA REGISTER
IRQ  EQU  $00EE  ;IRQ INTERRUPT VECTOR
PIOC  EQU  $02  ;PARALLEL I/O CONTROL REGISTER
PORTCL  EQU  $1005  ;PORTC LATCHED DATA REGISTER
CODE  EQU  $D010  ;THE MEMORY WHERE KEY CODE STORED
TEMP  EQU  $D000  ;TEMPORARY STORING MEMORY POSITION
TEMP2  EQU  $D002  ;TEMPORARY STORAGE FOR END OF
*  DISPLAY SIGNAL
MSG  EQU  $DC00  ;ADDR OF STARTING POINT

* SET MESSAGE BANK STARTS AT $DC02 *
ORG  $DC02
MSG1  FCC  ' *The 1ST key detected when "A" pressed*
  FCC  ' Prepare Ohms Law Wherever You Go '  
MSG2  FCC  ' *The 2ND key detected when "B" pressed*
  FCC  ' Should Born Rich Instead of Good Looking'  
MSG3  FCC  ' *The 3RD key detected when "C" pressed*
  FCC  ' Hay ! If It Works. Dont Fix It OK ! '  
MSG4  FCC  ' *The 4TH key detected when "D" pressed*
  FCC  ' With a Skill on Hands Worth Million $$'  
MSG5  FCC  ' *The 5TH key detected when "E" pressed*
  FCC  ' Never Say Its too Late to Start Learning'  
MSG6  FCC  ' *The 6TH key detected when "F" pressed*
  FCC  ' "TGIF" Yes ! It Is Party Time Go..Go..'  
ERR  FCC  ' * No Meaningful Key Pressed "Try Again"*'
  FCC  ' Choose A - F, then Press 0 - 5 Only '  
WELC  FCC  ' * Welcome to Ind & Tech Dept Open House *'
  FCC  ' Practical & Hands-on Experiences Are Fun'
THE MAIN PROGRAM STARTS HERE

This program uses PORTB and PORTD bit 2-4 as output lines.
It also uses PORTC bit 0-3 thru 74C922 as input lines to read the key pressed.

* ORG $C000 ; Set the starting addr. of the program
  * CLRA ; Set PORTC as input lines
  STAA DDRC
  LDAA #$2F ; SS-Hi, SCK-Lo, MOSI-Hi
  STAA PORTD
  LDAA #$38 ; SS, SCK, MOSI-outputs; MISO, TXD, RXD-inputs
  STAA DDRD
  LDAA #$5F ; SPI on as master, CPHA=1, CPOL=1
  STAA SPCR ; E/32 clock rate
  CLC

  LDS #$007E ; Set jump code in IRQ stra
  STAA IRQ ; Interrupt vectors
  LDY #STRAIN ; Set the addr of IRQ service
  STY IRQ+1 ; Routine

AGAIN
  BSET PIOC,X #$D2 ; Enable strobe A interrupt
  Jsr INIT

  LDX #$1000 ; The registers' base

  LDS #$0047 ; Top of user's stack on EVB
  LDAA #$7E ; Set jump code in IRQ stra
  STAA IRQ ; Interrupt vectors
  LDY #STRAIN ; Set the addr of IRQ service
  STY IRQ+1 ; Routine

LP1
  Jsr TIME2
  Deca
  Bne LP1

  LDAA CODE
  Cmpa #$FA ; Check the "A" key
  Bne CK5
  Ldx #$MSG1 ; First message
  Stx MSG
  Jsr DSP
  Jsr SELECT

CK5
  LDAA CODE
  Cmpa #$FB ; Check the "B" key
  Bne CK6
  Ldx #$MSG2 ; Second message
  Stx MSG
  Jsr DSP
  Jsr SELECT

CK6
  LDAA CODE
  Cmpa #$FC ; Check the "C" key
  Bne CK7
  Ldx #$MSG3 ; Third message
  Stx MSG
JSR DSP
JSR SELECT

CK7
LDAA CODE
CMPA #$FD ;CHECK THE "D" KEY
BNE CK8
LDX #MSG4 ;FOURTH MESSAGE
STX MSG
JSR DSP
JSR SELECT

CK8
LDAA CODE
CMPA #$FE ;CHECK THE "E" KEY
BNE CK9
LDX #MSG5 ;FIFTH MESSAGE
STX MSG
JSR DSP
JSR SELECT

CK9
LDAA CODE
CMPA #$FF ;CHECK THE "F" KEY
BNE CKA
LDX #MSG6 ;SIXTH MESSAGE
STX MSG
JSR DSP
JSR SELECT

CKA
LDX #ERR ;NO KEY PRESSED DISPLAY THE WARNING
STX MSG ;MESSAGE
JSR DSP
LDAA #$20 ;TIME DELAY FOR THE ERROR MESSAGE

LP2
JSR TIME2
DECA
BNE LP2
JMP AGAIN ;RESTART THE MAIN PROGRAM LOOP

* THE SUBROUTINES START HERE *
* SELECT THE FORMAT OF THE DISPLAY *

SELECT
LDAA CODE ;CHECK THE "0" KEY
CMPA #$F0
BEQ STOP2
CMPA #$F1 ;CHECK THE "1" KEY
BNE CK1
JSR INIT ;INITIALIZE THE LCD MODULE
RTS

CK1
LDAA CODE
CMPA #$F2 ;CHECK THE "2" KEY
BNE CK2
JSR DSLN1 ;DISPLAY 1ST LINE 40 LETTERS
RTS

CK2
LDAA CODE
CMPA #$F3 ;CHECK THE "3" KEY
BNE CK3
JSR SHFTR ;SHIFT RIGHT EVERY LETTER OF THE
RTS ;DISPLAY

CK3
LDAA CODE
CMPA #$F4 ;CHECK THE "4" KEY
BNE CK4
JSR SHFTL ; SHIFT LEFT EVERY LETTER OF THE DISPLAY

CK4
LDAA CODE
CMPA #$F5 ; CHECK THE "5" KEY
BNE SELECT
JSR BLNK ; BLINK THE DISPLAY FOR 10 TIMES

STOP2
RTS

* INITIALIZATION OF LCD MODULE DISPLAY FUNCTIONS *
INIT
LDAA #$00 ; WRITE TO INSTRUCTION REGISTER
STAA PORTB
JSR TIME ; MAKE A DELAY WAIT FOR LCD TO ACCEPT
LDAA #$38 ; FUNCTION SET 8 BIT DATA DUAL LINES
JSR DUMP ; DUMP DATA TO 74164 FOR DISPLAYING
JSR PULSE ; PULSE THE E LINE FOR INSTRUCTION REGISTER

JSR TIME ; SHORT TIME DELAY
LDAA #$0C ; DISPLAY ON, NO CURSOR, NO BLINKING
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$01 ; DISPLAY CLEAR
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$06 ; ADDR COUNTER INCREMENT AFTER EACH DISPLAY
JSR DUMP ; DISPLAY
JSR PULSE
JSR TIME
RTS

* DISPLAY 1ST LINE WITH 40 LETTERS *
DSLN1
LDAA #$00 ; WRITE TO INSTRUCTION REGISTER
STAA PORTB
JSR TIME ; MAKE A DELAY WAIT FOR LCD TO ACCEPT
LDAA #$01 ; CLEAR THE OLD DISPLAY
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$02 ; SET ADDR COUNTER TO ZERO, MOVE CURSOR TO HOME POSITION
JSR DUMP ; DISPLAY
JSR PULSE
JSR TIME
RTS

* DISPLAY 2ND LINE WITH 40 LETTERS *
DSLN2
LDAA #$00 ; WRITE TO INSTRUCTION REGISTER
STAA PORTB
JSR TIME ; MAKE A DELAY WAIT FOR LCD TO ACCEPT
LDAA #$C0 ; CHANGE TO 2ND FOR DISPLAYING
JSR DUMP
JSR PULSE
* SHIFT THE DISPLAY LEFT CONTINUOUSLY FOR EVERY LETTER *

SHFTL
LDX #$0050 ; CONTINUOUSLY SHIFT LEFT FOR 80
LO6
LDAA #$00 ; POSITIONS WHICH IS EQUIVALENT
STAA PORTB ; TO TWICE AS DISPLAY LENGTH
JSR TIME
LDAA #$18 ; SHIFT THE ENTIRE DISPLAY TO THE
JSR DUMP ; LEFT
JSR PULSE
JSR TIME
LDAA #$0A ; CONTINUOUS DELAY FOR STABLE
LO5
JSR TIME2 ; DISPLAYING
DECA
BNE LO5
DEX
BNE LO6
RTS

* SHIFT THE DISPLAY RIGHT CONTINUOUSLY FOR EVERY LETTER *

SHFTR
LDX #$0050 ; CONTINUOUSLY SHIFT RIGHT FOR 80
LP6
LDAA #$00 ; POSITIONS WHICH IS EQUIVALENT
STAA PORTB ; TO TWICE AS DISPLAY LENGTH
JSR TIME
LDAA #$1C ; SHIFT THE ENTIRE DISPLAY TO THE
JSR DUMP ; RIGHT
JSR PULSE
JSR TIME
LDAA #$08 ; CONTINUOUS DELAY FOR STABLE
LP5
JSR TIME2 ; DISPLAYING
DECA
BNE LP5
DEX
BNE LP6
RTS

* BLINK THE DISPLAY FOR 10 TIMES *

BLNK
LDX #$000A
LO9
LDAA CODE
BEQ STOP
LDAA #$00 ; WRITE TO INSTRUCTION REGISTER
STAA PORTB
JSR TIME ; DELAY WAIT FOR LCD TO ACCEPT
LDAA #$08 ; DIDDISPLAY OFF
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$10
LO7
JSR TIME2
DECA
BNE LO7
LDAA #$0C ; RESUME DISPLAY ON, NO CURSOR,
JSR  DUMP ; NO BLINKING
JSR  PULSE
JSR  TIME
LDAA  #$23
LO8
JSR  TIME2
DECA
BNE  LO8
DEX
BNE  LO9
STOP
RTS

* DISPLAY A 40 LETTER MESSAGE *
DSP
LDAA  #$04 ; WRITE TO DISPLAY DATA REGISTER
STAA  PORTB
JSR  TIME ; MAKE DELAY WAIT FOR LCD TO ACCEPT
LDX  MSG ; GET 1ST DISPLAY
LDY  MSG ; CALCULATE THE END OF MESSAGE WHICH
LDAB  #$51 ; IS 80 LETTER LONG TO TERMINATE
ABY ; THE DISPLAYING
STY  TEMP2
LOOP
LDAA  0,X ; DISPLAY THE FIRST MESSAGE
JSR  DUMP
JSR  PULSE2 ; PULSE FOR DATA REGISTER
JSR  TIME2 ; LONG TIME DELAY
INX
CPX  TEMP2 ; COMPARE THE END OF MESSAGE
BNE  LOOP
RTS

* PULSE THE E LINE FOR INSTRUCTION REGISTER WRITING *
PULSE
LDAB  #$01
STAB  PORTB
LDAB  #$00
STAB  PORTB
RTS

* PULSE FOR DISPLAY DATA REGISTER WRITING *
PULSE2
LDAB  #$05
STAB  PORTB
LDAB  #$04
STAB  PORTB
RTS

* SHORT TIME DELAY FOR PULSING E LINE *
TIME
LDAB  #$FF
LO1
DECB
BNE  LO1
RTS

* LONG TIME DELAY FOR DISPLAYING *
TIME2
LDAB  #$AF
LO4
LDY  #$0035
LO3
DEY
BNE LO3
DECB
BNE LO4
RTS

* DUMP THE DATA TO 74164 SHIFT REGISTER *
* AND WAIT FOR HC11 ACCEPTANCE *

DUMP
STAA SPDR ;WRITE DATA SPI DATA REG TO XFER
WAIT
LDAA SPSR ;CHECK FOR SPIF FLAG TO CHECK THE
BPL WAIT ; END OF THE XFER
RTS

* THE STROBE A IRQ INTERRUPT SERVICE ROUTINE *

STRAIN
LDAA PORTC ;READ THE DEBOUNCED KEY CODE AND
STAA TEMP ; STORE THE RESULT INTO MEMORY
STAA CODE
LDX #$1000 ;THE REGISTERS' BASE
LDAA PIOC,X ;DUMMY INSTRUCTIONS CLEAR THE STRA
LDAA PORTCL ; INTERRUPT FLAG IN THE FULL-INPUT
* ; HANDSHAKE MODE

END RTI

***************************************************************
* *
* FILE NAME: KYLCDSPI.EXE (Used in Experiment #5) *
* FUNCTION: USE THE KEYPAD TO INPUT THE CHOICES OF CONTROLLING*
* THE LCD MODULE DISPLAY FUNCTIONS, PORTD MISO(D3), SCK(D4), *
* RESPECTIVELY AS THE OUTPUT LINES, PORTB BIT 0-2 TO E R/W RS *
* LINES TO GET THE CHOICES INFORMATION. *
* AUTHOR: STEVE C. HSIUNG *
* DATE CREATED: 8/21/91 *
* *
***************************************************************

* THE ORIGINAL DATA SET UP *
PORTA EQU $1000 ;PORT A FOR TIMER I/O SIGNAL
PORTB EQU $1004 ;PORT B FOR OUTPUT ONLY
PORTC EQU $1003 ;PORT C FOR IN/OUT PUT DEPENDS ON
* ; DDRC'S BITS VALUE
PORTD EQU $1008 ;PORT D FOR IN/OUT PUT FOR BIT 0 -
* ; 5 ONLY
PORTE EQU $100A ;PORT E FOR A/D ANALOG SIGNAL INPUT
OPTION EQU $1039 ;SYSTEM CONFIGURATION OPTIONS
DDRC EQU $1007 ;THE DATA DIRECTION REGISTER OF THE
* ; PORT C
DDRD EQU $1009 ;THE DATA DIRECTION REGISTER OF THE
* ; PORT D, BUT BIT 7,6 ARE NOT
* ; CONTROLLABLE
ADCTL EQU $1030 ;A/D CONTROL/STATUS REGISTER
ADR1 EQU $1031 ;A/D RESULT REGISTER 1
ADR2 EQU $1032 ;A/D RESULT REGISTER 2
ADR3 EQU $1033 ; A/D RESULT REGISTER 3
ADR4 EQU $1034 ; A/D RESULT REGISTER 4
TMSK2 EQU $24 ; TIMER INTERRUPT MASK2
RTIME EQU $00EB ; REAL TIME INTERRUPT VECTOR
PACTL EQU $26 ; PULSE ACCUMULATOR CONTROL REGISTER
TFLG2 EQU $25 ; TIMER INTERRUPT FLAG2
SPCR EQU $1028 ; SPI CONTROL REGISTER
SPSR EQU $1029 ; SPI STATUS REGISTER
SPDR EQU $102A ; SPI DATA REGISTER
CODE EQU $D010 ; THE MEMORY WHERE KEY CODE STORED
TEMP EQU $D000 ; TEMPORARY STORING MEMORY POSITION
TEMP2 EQU $D002 ; TEMPORARY STORAGE FOR END OF
* ; DISPLAY SIGNAL
MSG EQU $DE00 ; ADDR OF STARTING POINT

* SET MESSAGE BANK STARTS AT $DE02 *

ORG $DE02
MSG1 FCC ' * The 1ST test message when "A" pressed*' 
MSG2 FCC ' * The 2ND test message when "B" pressed*' 
MSG3 FCC ' * The 3RD test message when "C" pressed*' 
MSG4 FCC ' * The 4TH test message when "D" pressed*' 
MSG5 FCC ' * The 5TH test message when "E" pressed*' 
MSG6 FCC ' * The 6TH test message when "F" pressed*' 
ERR FCC ' No meaningful key pressed "Try Again"* '

KYTBL FCB $EE,$DE,$BE,$7E,$ED,$DD,$BD,$7D
       FCB $EB,$DB,$BB,$7B,$E7,$D7,$B7,$77

* THE MAIN PROGRAM STARTS HERE *
* THIS PROGRAM USES PORTB AND PORTD BIT 2-4 AS OUTPUT LINES *
* IT ALSO USES PORTC AS INPUT LINES TO READ THE KEY PRESSED *

ORG $C000 ; SET THE STARTING ADDR. OF THE
* ; PROGRAM
CLRA ; SET PORTC AS INPUT LINES
STAA DDRC
LDAA #$2F ; SS-Hi, SCK-Lo, MOSI-Hi
STAA PORTD
LDAA #$38 ; SS, SCK, MOSI-OUTPUTS; MISO, TXD,
STAA DDRD ; RXD-INPUTS
LDAA #$5F ; SPI ON AS MASTER, CPHA=1, CPOL=1
STAA SPCR ; E/32 CLOCK RATE
CLC

LDX #$1000 ; THE REGISTERS' BASE
BSET TMSK2,X #$40 ; ENABLE THE RTI INTERRUPT
* ; THE SETTING SHOULD BE ACTIVATED
* ; IN EEPROM WITHIN 64 CYCLES AFTER
* ; THE RESET
BSET PACTL,X #$02 ; SET THE RTI INTERRUPT RATE OF
* ; 15.62mS
LDS #$0047 ; TOP OF USER'S STACK ON EVB
LDAA #$7E ; SET JUMP CODE IN RTI INTERRUPT
STAA RTIME ; VECTORS
LDY #TCLOCK ;SET THE ADDR OF RTI SERVICE
STY RTIME+1 ; TO 2ND & 3RD VECTORS
AGAIN JSR INIT
BCLR TFLG2,X #$BF ;CLEAR THE RTI FLAG BY
* CLI ;ENABLE THE INTERRUPT
LDAA CODE
CMPA #$0A ;CHECK THE "A" KEY
BNE CK5
LDX #MSG+$02 ;FIRST MESSAGE
STX MSG
JSR DSP
JSR SELECT
CK5 LDAA CODE
CMPA #$0B ;CHECK THE "B" KEY
BNE CK6
LDX #MSG+$2A ;SECOND MESSAGE
STX MSG
JSR DSP
JSR SELECT
CK6 LDAA CODE
CMPA #$0C ;CHECK THE "C" KEY
BNE CK7
LDX #MSG+$52 ;THIRD MESSAGE
STX MSG
JSR DSP
JSR SELECT
CK7 LDAA CODE
CMPA #$0D ;CHECK THE "D" KEY
BNE CK8
LDX #MSG+$7A ;FOURTH MESSAGE
STX MSG
JSR DSP
JSR SELECT
CK8 LDAA CODE
CMPA #$0E ;CHECK THE "E" KEY
BNE CK9
LDX #MSG+$A2 ;FIFTH MESSAGE
STX MSG
JSR DSP
JSR SELECT
CK9 LDAA CODE
CMPA #$0F ;CHECK THE "F" KEY
BNE CKA
LDX #MSG+$CA ;SIXTH MESSAGE
STX MSG
JSR DSP
JSR SELECT
CKA LDX #ERR ;NO KEY PRESSED DISPLAY THE WORNING
STX MSG ; MESSAGE
JSR DSP
JSR TIME2
JSR TIME2
JSR TIME2
JSR TIME2
JMP AGAIN ;RESTART THE MAIN PROGRAM LOOP

* THE SUBROUTINES START HERE *
* SELECT THE FORMAT OF THE DISPLAY *
SELECT LDAA CODE ;CHECK THE "0" KEY
BEQ STOP2
CMPA #$01 ;CHECK THE "1" KEY
BNE CK1
JSR INIT ;INITIALIZE THE LCD MODULE
RTS

CK1 LDAA CODE
CMPA #$02 ;CHECK THE "2" KEY
BNE CK2
JSR DSLN1 ;DISPLAY 1ST LINE 40 LETTERS
RTS

CK2 LDAA CODE
CMPA #$03 ;CHECK THE "3" KEY
BNE CK3
JSR DSLN2 ;DISPLAY 2ND LINE 40 LETTERS
RTS

CK3 LDAA CODE
CMPA #$04 ;CHECK THE "4" KEY
BNE CK4
JSR SHFT ;SHIFT EVERY LETTER OF THE DISPLAY
RTS

CK4 LDAA CODE
CMPA #$05 ;CHECK THE "5" KEY
BNE SELECT
JSR BLNK ;BLINK THE DISPLAY FOR 10 TIMES

STOP2 RTS

* INITIALIZATION OF LCD MODULE DISPLAY FUNCTIONS *
INIT LDAA #$00 ;WRITE TO INSTRUCTION REGISTER
STAA PORTB
JSR TIME ;MAKE A DELAY WAIT FOR LCD TO ACCEPT
LDAA #$38 ;FUNCTION SET 8 BIT DATA DUAL LINES
JSR DUMP ;DUMP DATA TO 74164 FOR DISPLAYING
JSR PULSE ;PULSE THE E LINE FOR INSTRUCTION

JSR TIME ;SHORT TIME DELAY
LDAA #$0C ;DISPLAY ON, NO CURSOR, NO BLINKING
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$01 ;DISPLAY CLEAR
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$06 ;ADDR COUNTER INCREMENT AFTER EACH
JSR DUMP ; DISPLAY
JSR PULSE
JSR TIME
RTS

* DISPLAY 1ST LINE WITH 40 LETTERS *

DSLN1
LDAA #$00 ;WRITE TO INSTRUCTION REGISTER
STAA PORTB
JSR TIME ;MAKE DELAY WAIT FOR LCD TO ACCEPT
LDAA #$01 ;CLEAR THE OLD DISPLAY
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$02 ;SET ADDR COUNTER TO ZERO, MOVE
JSR DUMP ; CURSOR TO HOME POSITION
JSR PULSE
JSR TIME
RTS

* DISPLAY 2ND LINE WITH 40 LETTERS *

DSLN2
LDAA #$00 ;WRITE TO INSTRUCTION REGISTER
STAA PORTB
JSR TIME ;MAKE DELAY WAIT FOR LCD TO ACCEPT
LDAA #$C0 ;CHANGE TO 2ND FOR DISPLAYING
JSR DUMP
JSR PULSE
JSR TIME
RTS

* SHIFT THE DISPLAY LEFT CONTINUOUSLY FOR EVERY LETTER *

SHFT
LDX #$0050 ;CONTINUOUSLY SHIFT LEFT FOR 80
LO6
LDAA #$00 ; POSITIONS WHICH IS EQUIVALENT
STAA PORTB ; TO TWICE AS DISPLAY LENGTH
JSR TIME
LDAA #$18 ;SHIFT THE ENTIRE DISPLAY TO THE
JSR DUMP ; LEFT
JSR PULSE
JSR TIME
LDAA #$0C ;CONTINUOUS DELAY FOR STABLE
LO5
JSR TIME2 ; DISPLAYING
DECA
BNE LO5
DEX
BNE LO6
RTS

* BLINK THE DISPLAY FOR 10 TIMES *

BLNK
LDX #$000A
LO9
LDAA CODE
BEQ STOP
LDAA #$00 ;WRITE TO INSTRUCTION REGISTER
STAA PORTB
JSR TIME ;DELAY WAIT FOR LCD TO ACCEPT
LDAA #$08 ;DISPLAY OFF
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$10
L07
JSR TIME2
DECA
BNE L07
LDAA #$0C ; RESUME DISPLAY ON, NO CURSOR,
JSR DUMP ; NO BLINKING
JSR PULSE
JSR TIME
LDAA #$23
L08
JSR TIME2
DECA
BNE L08
DEX
BNE L09
STOP RTS

* DISPLAY A 40 LETTER MESSAGE *

DSP LDAA #$04 ; WRITE TO DISPLAY DATA REGISTER
STAA PORTB
JSR TIME ; MAKE DELAY WAIT FOR LCD TO ACCEPT
LDX MSG ; GET 1ST DISPLAY
LDY MSG ; CALCULATE THE END OF MESSAGE WHICH
LDAB #$29 ; IS 40 LETTER LONG TO TERMINATE
ABY ; THE DISPLAYING
STY TEMP2
LOOP LDAA 0,X ; DISPLAY THE FIRST MESSAGE
JSR DUMP
JSR PULSE2 ; PULSE FOR DATA REGISTER
JSR TIME2 ; LONG TIME DELAY
INX
CPX TEMP2 ; COMPARE THE END OF MESSAGE
BNE LOOP
RTS

* PULSE THE E LINE FOR INSTRUCTION REGISTER WRITING *

PULSE LDAB #$01
STAB PORTB
LDAB #$00
STAB PORTB
RTS

* PULSE FOR DISPLAY DATA REGISTER WRITING *

PULSE2 LDAB #$05
STAB PORTB
LDAB #$04
STAB PORTB
RTS

* SHORT TIME DELAY FOR PULSING E LINE *

TIME LDAB #$FF
L01 DECB
BNE  LO1
RTS

* LONG TIME DELAY FOR DISPLAYING *
TIME2  LDAB  #$FF
LO4   LDY  #$0035
LO3  DEY
BNE  LO3
DECB
BNE  LO4
RTS

* DUMP THE DATA TO 74164 SHIFT REGISTER AND WAIT FOR HC11 *
* ACCEPTANCE *
DUMP  STAA  SPDR  ;WRITE DATA SPI DATA REG TO XFER
WAIT  LDAA  SPSR  ;CHECK FOR SPIF FALG TO CHECK THE
       BPL  WAIT  ; END OF THE XFER
       RTS

* REAL TIME INTERRUPT SERVICE ROUTINE WHICH PERIODICLY CHECKS *
* THE KEYPAD PRESSED INFORMATION AND SAVE THE KEY CODE IN CODE *
TCLOCK  LDAA  PORTC  ;READ KEYPAD INFOR FROM PORTC
       CMPA  #$FF  ;CHECK IF ANY KEY PressED
       BCC  END  ;IF NO KEY PressED CONTINUE WAITING
LOOP2  LDAA  PORTC  ; THE KEY BOUNCING SETTLED DOWN
       DECB  ; THE DELAY LOOP IS ABOUT 1 mS
       BNE  LOOP2
       STAA  TEMP  ;STORE THE KEY INFOR TO A TEMPORARY
       ; MEMORY POSITION
FIND  LDAA  0,X  ;FIND THE MATCHED KEY CODE
       INX
       CMPA  TEMP  ;IF MATCHED SAVE THE KEY INFOR
       BEQ  SAVE
       CPX  #$FF  ;USE $FF AS THE SIGN OF THE END OF
       ; OF TABLE
       BNE  FIND  ; LOOP THRU THE ENTIRE TABLE
       LDAB  #$FF  ; USE $FF AS THE SIGN OF THE END OF
       STAB  CODE  ; TABLE
END
SAVE  XGDX  ;MOVE KEYTABLE POINTER TO D REG
       SUBD  #$1000  ; CONVERT THE KEY CODE TO A REAL
       STAB  CODE  ; EQUIVALENT NUMBER STORE IN CODE
END
LDX  #$1000  ; MEMORY
BCLR  TFLG2,X #$BF ; CLEAR THE RTI FLAG BY WRITING
       ; 1 TO THE FLAG BIT BEFORE
       ; TURNING FROM RTI INTERRUPT
RTI
FILE NAME: EEKEYSPI.EXE (Used in Experiment #6)
FUNCTION: ELECTRONIC KEYS SECURITY SYSTEM. USE KEYPAD
CONNECTED TO PORTC LCD MODULE CONNECTED TO PORTD MOSI
(D3), SCK(D4), RESET AND PORTB BIT 0-2 TO E R/W RS, ALSO
USE PORTB BIT 3 AS OPEN GATE CONTROL USE PROTB BIT 7 AS
CLOSE GATE & ALARM CONTROL
AUTHOR: STEVE C. HSIUNG
DATE CREATED: 8/22/91

* THE ORIGINAL DATA SET UP *

ORG $DD00

SET THE STARTING ADDR. OF THE
REAL KEY CODE TABLE
PORTA EQU $1000 PORT A FOR TIMER I/O SIGNAL
PORTB EQU $1004 PORT B FOR OUTPUT ONLY
PORTC EQU $1003 PORT C FOR IN/OUT PUT DEPENDS ON
DDRC'S BITS VALUE
PORTD EQU $1008 PORT D FOR IN/OUT PUT FOR BIT 0 -
5 ONLY
PORTE EQU $100A PORT E FOR A/D ANALOG SIGNAL INPUT
OPTION EQU $1039 SYSTEM CONFIGURATION OPTIONS
DDRC EQU $1007 THE DATA DIRECTION REGISTER OF
PORT C
DDRD EQU $1009 THE DATA DIRECTION REGISTER OF
PORT D, BUT BIT 7, 6 ARE NOT
CONTROLLABLE
ADCTRL EQU $30 A/D CONTROL/STATUS REGISTER
ADR1 EQU $31 A/D RESULT REGISTER 1
ADR2 EQU $32 A/D RESULT REGISTER 2
ADR3 EQU $33 A/D RESULT REGISTER 3
ADR4 EQU $34 A/D RESULT REGISTER 4
TMASK2 EQU $24 TIMER INTERRUPT MASK2
RTIME EQU $00EB REAL TIME INTERRUPT VECTOR
PACTL EQU $26 PULSE ACCUMULATOR CONTROL REGISTER
TFLAG2 EQU $25 TIMER INTERRUPT FLAG2
SPCR EQU $1028 SPI CONTROL REGISTER
SPSR EQU $1029 SPI STATUS REGISTER
SPDR EQU $102A SPI DATA REGISTER
CODE EQU $D010 THE MEMORY WHERE KEY CODE STORED
TEMP EQU $D000 TEMPORARY STORING MEMORY POSITION
PINTER EQU $DC00 POINTER USED FOR MEMORY STORAGE
COUNT EQU $DC02 COUNTER USED FOR 4 TIMES INTERING
TEMP2 EQU $D008 TEMPORARY STORING MEMORY POSITION
SECNUM EQU $D002 4 SECURITY NUM STORED IN 4 MEMORY
LOCATIONS($D002-$D005)
TRIAL EQU $DC04 COUNTER USED FOR CHECKING 3 TRIALS
MSG1 FCC "*Welcome to the Electric Security system"
MSG2 FCC "*4 codes have been received. Please wait"
FCC 'for security checking. ................. '

MSG3 FCC '*The codes you entered are not acceptable'
FCC 'you have 3 chances to try OK! ....... '

MSG4 FCC '*Hay! You have entered the correct codes'
FCC 'Welcome to the home sweet home palace '

MSG5 FCC '*Warning! Danger! You only have total '
FCC 'three chances to try before the Alarming'

MSG6 FCC ' Ha! You are a "SOB" unwelcomed intruder'
FCC 'The alarm system has been activated now!

KYTBL FCB $EE,$DE,$BE,$7E,$6D,$6D,$BD,$7D
FCB $EB,$DB,$BB,$7B,$67,$D7,$B7,$77

* THE MAIN PROGRAM STARTS HERE *

ORG $C000 ;SET THE STARTING ADDR. OF THE MAIN

* PROGRAM

LDX #$1000 ;THE REGISTERS' BASE
BSET PACTL,X #$80 ;SET PORTA BIT 7 AS OUTPUT
CLR A ;SET PORTC AS INPUT LINES
STAA DDRC ;SS-Hi, SCK-Lo, MOSI-Hi
STAA PORTD
LDAA #$2F ;SS, SCK, MOSI-OUTPUTS; MISO, TXD,
STAA DDRD ;RXD-INPUTS
LDAA #$38 ;SPI ON AS MASTER, CPHA=1, CPOL=1
STAA SPCR ;E/32 CLOCK RATE
CLR C ;CLEAR THE BEGINNING C FLAG

AGAIN2 CLR TRIAL ;CLEAR THE TRIAL COUNTER
AGAIN1 JSR DSP1 ;DISPLAY THE 1ST MESSAGE
CLR COUNT ;CLEAR THE # OF KEY ENTER COUNTER
CLC
LDY #CODE ;SET UP A POINTER POINTING TO THE
STY PNTER ;MEMORY CODE AND USE IT LATER FOR

* INDIRECT ADDRESSING

JSR KEY ;DETECT THE KEY CODE ENTERY
LDX CODE ;GET THE KEY CODES AND CHECK WITH
CPX SECNUM ;THE SECURITY NUMBERS
BEQ NEXT1 ;CONTINUE CHECK THE SECOND PAIR

* ;OF KEY CODES AND SECURITY NUM

JSR CLOSE ;WRONG KEY CODES ROUTINE
BRA AGAIN1

NEXT1 LDX CODE+$02
CPX SECNUM+$02
BEQ NEXT2
JSR CLOSE
BRA AGAIN1

NEXT2 JSR OPEN ;CORRECT KEY CODES ROUTINE
BRA AGAIN2

* DISPLAY THE FIRST MESSAGE SUBROUTINE *

DSP1 JSR INIT
LDAA #$04 ;WRITE TO DISPLAY DATA REGISTER
STAA PORTB
JSR TIME ;MAKE DELAY WAIT FOR LCD TO ACCEPT
LDX #MSG1 ;GET 1ST DISPLAY

LOOP1
LDAA 0,X ;DISPLAY THE FIRST MESSAGE
JSR DUMP ;DUMP DATA TO 74164 FOR DISPLAYING
JSR PULSE2 ;PULSE FOR DATA REGISTER
JSR TIME2 ;LONG TIME DELAY
INX
CPX #MSG1+$51 ;COMPARE THE END OF MESSAGE
BNE LOOP1
RTS

* DISPLAY THE SECOND MESSAGE SUBROUTINE *

DSP2
JSR INIT
LDAA #$04 ;WRITE TO DISPLAY DATA REGISTER
STAA PORTB
JSR TIME ;MAKE DELAY WAIT FOR LCD TO ACCEPT
LDX #MSG2 ;GET 2ND DISPLAY

LOOP2
LDAA 0,X ;DISPLAY THE SECOND MESSAGE
JSR DUMP ;DUMP DATA TO 74164 FOR DISPLAYING
JSR PULSE2 ;PULSE FOR DATA REGISTER
JSR TIME2 ;LONG TIME DELAY
INX
CPX #MSG2+$51 ;COMPARE THE END OF MESSAGE
BNE LOOP2
RTS

* DISPLAY THE THIRD MESSAGE SUBROUTINE *

DSP3
JSR INIT
LDAA #$04 ;WRITE TO DISPLAY DATA REGISTER
STAA PORTB
JSR TIME ;MAKE DELAY WAIT FOR LCD TO ACCEPT
LDX #MSG3 ;GET 3RD DISPLAY

LOOP3
LDAA 0,X ;DISPLAY THE THIRD MESSAGE
JSR DUMP ;DUMP DATA TO 74164 FOR DISPLAYING
JSR PULSE2 ;PULSE FOR DATA REGISTER
JSR TIME2 ;LONG TIME DELAY
INX
CPX #MSG3+$51 ;COMPARE THE END OF MESSAGE
BNE LOOP3
RTS

* INITIALIZATION OF LCD MODULE DISPLAY FUNCTIONS *

INIT
LDAA #$00 ;WRITE TO INSTRUCTION REGISTER
STAA PORTB
JSR TIME ;MAKE A DELY WAIT FOR LCD TO ACCEPT
LDAA #$38 ;FUNCTION SET 8 BIT DATA DUAL LINES
JSR DUMP ;DUMP DATA TO 74164 FOR DISPLAYING
JSR PULSE ;PULSE THE E LINE FOR INSTRUCTION

* REGISTERS *
JSR TIME ;SHORT TIME DELAY
LDAA #$50C ;DISPLAY ON, NO CURSOR, NO BLINKING
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$01 ; DISPLAY CLEAR
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$06 ; ADDR COUNTER INCREMENT AFTER EACH
JSR DUMP ; DISPLAY
JSR PULSE
JSR TIME
RTS

* PULSE THE E LINE FOR INSTRUCTION REGISTER WRITING *
PULSE    LDAB #$01
STAB PORTB
LDAB #$00
STAB PORTB
RTS

* PULSE FOR DISPLAY DATA REGISTER WRITING *
PULSE2   LDAB #$05
STAB PORTB
LDAB #$04
STAB PORTB
RTS

* SHORT TIME DELAY FOR PULSING E LINE *
TIME    LDAB #$FF
L01      DECB
         BNE L01
RTS

* LONG TIME DELAY FOR DISPLAYING *
TIME2   LDAB #$BF
L04      LDY #$0035
L03      DEY
         BNE L03
         DECB
         BNE L04
RTS

* THE LONGEST TIME DELAY FOR DISPLAYING *
TIME3   LDAB #$FF
LS4      LDY #$05FF
LS3      DEY
         BNE LS3
         DECB
         BNE LS4
RTS

* DUMP THE DATA TO 74164 SHIFT REGISTER AND WAIT FOR HC11 *
* ACCEPTANCE *
DUMP    STAA SPDR ; WRITE DATA SPI DATA REG TO XFER
WAIT    LDAA SPSR ; CHECK FOR SPIF FALG TO CHECK THE
         BPL WAIT ; END OF THE XFER
RTS

* KEYPAD MONITOR 4 KEY CODES INPUT ROUTINE *
KEY  
LDAA PORTC  ; READ KEYPAD INFO FROM PORTC
CMPA #$FF  ; CHECK IF ANY KEY PRESSED
BCC KEY  ; IF NO KEY PRESSED CONTINUE WAITING

DELAY1  
LDAA #$FF  ; SET THE TIME DELAY LOOP WAIT FOR
DEC PORTC  ; THE KEY BOUNCING SETTLED DOWN
BNE DELAY1  ; THE DELAY LOOP IS ABOUT 20 mS

STAA TEMP  ; STORE THE KEY INFO TO A TEMPORARY
*
LDY PNTER  ; SET Y AS THE POINTER FOR THE
*
LDX #KYTBL  ; LOCATE THE KEY CODE IN KEYTABLE
FIND  
LDAA 0,X  ; FIND THE MATCHED KEY CODE
INX
CMPA TEMP  ; IF MATCHED SAVE THE KEY INFO
BEQ SAVE  ; OF TABLE
*
CPX #KYTBL+$10  ; IF NOT MATCHED CHECK THE END
*
BNE FIND  ; LOOP THRU THE ENTIRE TABLE
LDAB #$FF  ; USE $FF AS THE SIGN OF THE END OF
STAB CODE  ; TABLE
BRA KEY

SAVE  
XGDX  ; MOVE KEYTABLE POINTER TO D REG
SUBD #KYTBL+1  ; CONVERT THE KEY CODE TO A REAL
STAB 0,Y  ; EQUIVALENT NUMBER STORE IN
*
INY  ; ADVANCE THE POINTER FOR NEXT KEY
*
STY PNTER  ; RENEW THE POINTER POSITION
LDY #$DF0C  ; SET UP A TIME DELAY FOR ABOUT 0.2S
DELAY2  
DEY  ; FOR THE BOUNCING KEY SETTLE DOWN
BNE DELAY2  ; THIS CAN BE REPLACED BY REAL
*
INC COUNT
LDAA COUNT
CMPA #$04
BEQ END
BRA KEY

END  
JSR DSP2
JSR TIME3
RTS

* THE KEY CODES ENTERED ARE CORRECT *
OPEN  
JSR INIT
LDAA #$04  ; WRITE TO DISPLAY DATA REGISTER
STAA PORTB
JSR TIME  ; MAKE DELAY WAIT FOR LCD TO ACCEPT
LDX #MSG4  ; GET 4TH DISPLAY
LOOP4  LDAA 0,X ;DISPLAY THE FOURTH MESSAGE
JSR DUMP ;DUMP DATA TO 74164 FOR DISPLAYING
JSR PULSE2 ;PULSE FOR DATA REGISTER
JSR TIME2 ;LONG TIME DELAY
INX
CPX #MSG4+$51 ;COMPARE THE END OF MESSAGE
BNE LOOP4
LDAA #$0A ;MAKE THE SIGNAL SWITCHING 10 TIMES
STA TEMP2

LOP1
LDAA #$08 ;TURN PORTB BIT 3 ON
STAA PORTB
JSR TIME2
CLRA ;TURN PORTB BIT 3 OFF
STAA PORTB
JSR TIME2
JSR TIME2
DEC TEMP2
BNE LOP1
RTS

* THE KEY CODES ENTERED ARE INCORRECT *
CLOSE  INC TRIAL
JSR INIT
LDA #$04 ;WRITE TO DISPLAY DATA REGISTER
STA PORTB
JSR TIME ;MAKE DELAY WAIT FOR LCD TO ACCEPT
LDX #MSG5 ;GET 5TH DISPLAY

LOOP5
LDAA 0,X ;DISPLAY THE FIFTH MESSAGE
JSR DUMP ;DUMP DATA TO 74164 FOR DISPLAYING
JSR PULSE2 ;PULSE FOR DATA REGISTER
JSR TIME2 ;LONG TIME DELAY
INX
CPX #MSG5+$51 ;COMPARE THE END OF MESSAGE
BNE LOOP5
JSR BLNK
LDA TRIAL
CMPA #$03
BEQ ALAM
RTS

ALAM  JSR INIT
LDA #$04 ;WRITE TO DISPLAY DATA REGISTER
STA PORTB
JSR TIME
LDX #MSG6 ;GET 6TH DISPLAY

LOOP6
LDA 0,X ;DISPLAY THE SIXTH MESSAGE
JSR DUMP ;DUMP DATA TO 74164 FOR DISPLAYING
JSR PULSE2 ;PULSE FOR DATA REGISTER
JSR TIME2 ;LONG TIME DELAY
INX
CPX #MSG6+$51 ;COMPARE THE END OF MESSAGE
BNE LOOP6
JSR TIME3
CLR TRIAL
LDAA #$0A ;MAKE THE SIGNAL SWITCHING 10 TIMES
STAA TEMP2

LOP2
LDAA #$10 ;TURN PORTB BIT 4 ON WHICH IS AN
STAA PORTB ; ALARM SYSTEM
JSR TIME3
CLRA ;TURN PORTB BIT 4 OFF
STAA PORTB
JSR TIME2
DEC TEMP2
BNE LOP2

* BLINK THE DISPLAY FOR 4 TIMES *

BLNK LDX #$0004

LO9
LDAA #$00 ;WRITE TO INSTRUCTION REGISTER
STAA PORTB
JSR TIME ;DELAY WAIT FOR LCD TO ACCEPT
LDAA #$08 ;DISPLAY OFF
JSR DUMP
JSR PULSE
JSR TIME
LDAA #$09

LO7
JSR TIME2
DECA
BNE LO7

LDAA #$0C ;RESUME DISPLAY ON, NO CURSOR,
JSR DUMP ; NO BLINKING
JSR PULSE
JSR TIME
LDAA #$1E

LO8
JSR TIME2
DECA
BNE LO8
DEX
BNE LO9

RTS

*******************************************************************************
FILE NAME: DCMOTOR.EXE (Used in Experiment #7)
FUNCTION: THE OUTPUT COMPARE AND STRA INTERRUPTS ARE USED IN*
THE EXPERIMENT THE OC1 IS USED TO CONTROL THE PERIOD OF *
THE SIGNAL GENERATED IN OC2 AT PA6, WHEN TOC1 = TCNT AND *
THE DUTY CYCLE IS CONTROLLED BY THE TOC2 SETTING FROM *
KEYPAD INPUT, THE PULSE WIDTH MODULATED SIGNAL IS TO USED*
TO CONTROL A DC MOTOR THRU THE HELP OF TI L298KV FULL-H *
DRIVER CONTROLLER CHIP *
AUTHOR: STEVE C. HSIUNG *
DATE CREATED: 9/14/91 *
*******************************************************************************
* THE ORIGINAL DATA SET UP *

ORG $DF00 ; SET THE STARTING ADDR. OF THE PROGRAM

PORTA EQU $1000 ; PORT A FOR TIMER I/O SIGNAL
PORTB EQU $1004 ; PORT B FOR OUTPUT ONLY
PORTC EQU $1003 ; PORT C FOR IN/OUT PUT DEPENDS ON DDRC'S BITS VALUE
PORTD EQU $1008 ; PORT D FOR IN/OUT PUT FOR BIT 0 - 5 ONLY
PORTE EQU $100A ; PORT E FOR A/D ANALOG SIGNAL INPUT
OPTION EQU $1039 ; SYSTEM CONFIGURATION OPTIONS
DDRC EQU $1007 ; THE DATA DIRECTION REGISTER OF THE PORT C
DDRD EQU $1009 ; THE DATA DIRECTION REGISTER OF THE PORT D, BUT BIT 7,6 ARE NOT CONTROLLABLE

TMSK2 EQU $24 ; TIMER INTERRUPT MASK2
RTIME EQU $00EB ; REAL TIME INTERRUPT VECTOR
PACTL EQU $25 ; PULSE ACCUMULATOR CONTROL REGISTER
TFLG2 EQU $25 ; TIMER INTERRUPT FLAG2
TCNT EQU $0E ; FREE RUNNING COUNTER 16 BIT READ ONLY

TOC1 EQU $16 ; TIMER OUTPUT COMPARE REG1 16 BIT
TOC2 EQU $18 ; TIMER OUTPUT COMPARE REG2 16 BIT
TOC3 EQU $1A ; TIMER OUTPUT COMPARE REG3 16 BIT
TOC4 EQU $1C ; TIMER OUTPUT COMPARE REG4 16 BIT
TOC5 EQU $1E ; TIMER OUTPUT COMPARE REG5 16 BIT
TMSK1 EQU $22 ; TIMER INTERRUPT MASK1
TFLG1 EQU $23 ; TIMER INTERRUPT FLAG
TCTL1 EQU $20 ; TIMER OUTPUT SIGNALS CONTROL

OUTCOM1 EQU $00DF ; OC1 INTERRUPT VECTORS
OUTCOM2 EQU $00DC ; OC2 INTERRUPT VECTORS
OUTCOM3 EQU $00D9 ; OC3 INTERRUPT VECTORS
OUTCOM4 EQU $00D6 ; OC4 INTERRUPT VECTORS
OUTCOM5 EQU $00D3 ; OC5 INTERRUPT VECTORS
OC1D EQU $0D ; OUTPUT COMPARE 1 DATA REG
OC1M EQU $0C ; OUTPUT COMPARE 1 MASK REG
IRQ EQU $00EE ; IRQ INTERRUPT VECTOR
PIOC EQU $02 ; PARALLEL I/O CONTROL REGISTER
PORTCL EQU $1005 ; PORT C LATCHED DATA REGISTER
CODE EQU $D010 ; THE MEMORY WHERE KEY CODE STORED
TEMP EQU $D000 ; TEMPORARY STORING MEMORY POSITION
HIOUT EQU $DE00 ; STORAGE PLACE FOR HI PERIOD OUTPUT
RATE EQU $DE02 ; TEMPORARY STORAGE FOR Hi OF THE HI PERIOD WHICH IS 16 BIT LONG

KEY EQU $D002 ; TEMPORARY POSITION FOR KEY INFOR

TABLE FCB $00,$00,$02,$9A,$05,$34,$07,$CE,$0A,$68,$0D,$02
      FCB $0F,$9C,$12,$36,$14,$D0,$17,$6A,$1A,$04,$1C,$9E
      FCB $1F,$38,$21,$D2,$24,$6C,$27,$06

* THE MAIN PROGRAM STARTS HERE *

ORG $C000 ; SET THE STARTING ADDR. OF THE MAIN PROGRAM
* PROGRAM
  LDAA #$00 ; CONFIGURE ALL LINES OF PORTC AS INPUT PORT

* SET UP THE OUTPUT COMPARE PWN INTERRUPT CONNECTIONS *
LDS #$0047 ; TOP OF USER'S STACK ON EVB
LDAA #$7E ; SET JUMP CODE IN OUTPUT COMPARE
STAA OUTCOM1 ; OC1 INTERRUPT VECTORS
LDY #$FIVSQV ; SET THE ADDR OF OUTPUT COMPARE
STY OUTCOM1+1 ; SERVICE TO 2ND & 3RD VECTORS
STAA IRQ ; IRQ INTERRUPT VECTORS
LDY #$STRAIN ; SET THE ADDR OF STRA INTERRUPT
STY IRQ+1 ; SERVICE ROUTINE
LDX #$1000 ; THE REGISTERS' BASE
BSET PACTL,X #$80 ; SET PORT A BIT 7 AS OUTPUT LINE
  WHICH IS THE PIN OF OC1
BSET OC1M,X #$F8 ; SET ALL THE OUTPUT COMPARE MASK
BSET OC1D,X #$00 ; MAKE ALL OUTPUT Lo (OC1 OUTPUT)
BSET TCTL1,X #$FF ; SET OC2-OC5 COMPARE LINES Hi
LDD TCNT,X ; DELAY ANY OC1, OC2 COMPARE
STD TOC1,X ; FUNCTION TO BEGIN
STD TOC2,X
BSET PIOC,X #$D2 ; ENABLE STROBE A INTERRUPT
  SENSE Lo-Hi, FLAG ENABLE, FULL-
  INPUT HANDSHAKE MODE
AGAIN BCLR TFLG1,X #$07 ; CLEAR THE OC1F OC2F FLAGS BY
  WRITING 1S TO BIT 6 & 7
BSET TMSK1,X #$80 ; ENABLE THE OC1 INTERRUPT
CLI ; ENABLE THE INTERRUPT
CLR HIOUT ; STOP ANY ACTION ON MOTOR
WAIT0 LDAA KEY ; CHECK FOR ON/OFF SWITCH ON "0" KEY
CMPA #$00 ; IF OFF THEN LOOP HERE
BNE WAIT0
WAIT1 LDAA KEY ; CHECK FOR BACKWARD ROTATION
CMPA #$0B ; COMMAND INFORMATION
BEQ SETB ; IF "B" KEY PRESSED GO BACKWARD
LDAA KEY ; CHECK FOR FORWARD ROTATION COMMAND
CMPA #$0F ; INFORMATION
BNE WAIT1 ; IF NOTHING PRESSED LOOP AGAIN WAIT
SETF LDAB #$01 ; WAIT FOR "B" OR "F" KEY
STAB PORTB ; FORWARD INFOR FOR PORT B
BRA START
SETB LDAB #$02 ; BACKWARD INFOR FOR PORT B
STAB PORTB
START LDD #$246C ; SET THE INITIAL STARTING TORQUE
STD HIOUT ; TO MAKE THE MOTOR ROTATE
LDY #$05FF ; GIVE SOME TIME DELAY FOR THE START
LO1 DEY ; ING TORQUE STABLE
BNE LO1
CHGSPD LDAB KEY
ROLB ; MULTIPLY NUMBER BY 2 TO GET THE
LDX #TABLE ; EQUIVALENT NUMBER IN THE TABLE
ABX ; LOOK UP

; TOP OF USER'S STACK ON EVB
; SET JUMP CODE IN OUTPUT COMPARE
; OC1 INTERRUPT VECTORS
; SET THE ADDR OF OUTPUT COMPARE
; SERVICE TO 2ND & 3RD VECTORS
; IRQ INTERRUPT VECTORS
; SET THE ADDR OF STRA INTERRUPT
; SERVICE ROUTINE
; THE REGISTERS' BASE
; SET PORT A BIT 7 AS OUTPUT LINE
; WHICH IS THE PIN OF OC1
; SET ALL THE OUTPUT COMPARE MASK
; MAKE ALL OUTPUT Lo (OC1 OUTPUT)
; SET OC2-OC5 COMPARE LINES Hi
; DELAY ANY OC1, OC2 COMPARE
; FUNCTION TO BEGIN
; ENABLE STROBE A INTERRUPT
; SENSE Lo-Hi, FLAG ENABLE, FULL-
; INPUT HANDSHAKE MODE
; CLEAR THE OC1F OC2F FLAGS BY
; WRITING 1S TO BIT 6 & 7
; ENABLE THE OC1 INTERRUPT
; ENABLE THE INTERRUPT
; STOP ANY ACTION ON MOTOR
; CHECK FOR ON/OFF SWITCH ON "0" KEY
; IF OFF THEN LOOP HERE
; CHECK FOR BACKWARD ROTATION
; COMMAND INFORMATION
; IF "B" KEY PRESSED GO BACKWARD
; CHECK FOR FORWARD ROTATION COMMAND
; INFORMATION
; IF NOTHING PRESSED LOOP AGAIN WAIT
; WAIT FOR "B" OR "F" KEY
; FORWARD INFOR FOR PORT B
; BACKWARD INFOR FOR PORT B
; SET THE INITIAL STARTING TORQUE
; TO MAKE THE MOTOR ROTATE
; GIVE SOME TIME DELAY FOR THE START
; ING TORQUE STABLE
; MULTIPLY NUMBER BY 2 TO GET THE
; EQUIVALENT NUMBER IN THE TABLE
; LOOK UP
LDD 0,X ; GET THE DESIRED VALUES IN THE TABLE ACCORDING TO THE OFFSET
STD HIOUT ; FROM THE KEYPAD INPUT
LDAA KEY ; STORE THE HI OUTPUT PERIOD IN TEMPORARY MEMORY POSITION
CMFA #$00 ; CHECK THE ON/OFF KEY AGAIN
BEQ CHGSPD
BRA AGAIN ; LOOP BACK TO THE BEGINNING

* OUTPUT COMPARE INTERRUPT SERVICE ROUTINE, PWM SIGNAL PRODUCED *
* TO GENERATE THE VARIABLE DUTY CYCLE SQUARE WAVE *
FIVSQV LDX #$1000 ; THE BASE REGISTER
LDD TCNT,X ; SET THE SIGNAL PERIOD IN OC1
ADDD #$10000 ; THE PERIOD IS 10 mS
STD TOC1,X
LDD TCNT,X
ADDD HIOUT ; THE OC2 HAS VARIABLE HI OUTPUT
STD TOC2,X
BCLR TFLG1,X #$07 ; CLEAR THE OC1F-OC5F FLAGS

* ; BEFORE RETURNING FROM SERVICE ROUTINE
RTI

* THE IRQ INTERRUPT SERVICE ROUTINE *
* TO READ THE KEYPAD INPUT INFORMATION WHEN THE KEY PRESSED *
STRAIN LDAB PORTC ; READ THE DEBOUNCED KEY CODE AND
SUBB #$F0 ; ADJUST FOR THE CORRECT NUMBER
STAB KEY ; STORE THE KEYPAD INFORMATION INTO KEY
LDX #$1000 ; GET THE REGISTERS' BASE
LDAA PICC,X ; DUMMY INSTRUCTIONS CLEAR THE STRA
LDAA PORTCL ; INTERRUPT FLAG IN THE FULL-INPUT

END RTI