An 8-bit microcomputer system processor, memory and I/O
by Darrell Boots Irvin

A thesis submitted in partial fulfillment of the requirements for the degree of MASTER OF SCIENCE in Electrical Engineering
Montana State University

Abstract:
This thesis presents an 8-bit microcomputer based on the INTEL 8008 microprocessor Suitable applications for the system include data acquisition and digital control systems formerly realized with dedicated hardware or larger and more expensive minicomputer systems.

Processor, memory and I/O subsystems are first described from an architectural viewpoint with the aid of block diagram. The instruction set for the system is introduced in an informal manner within the main text while Appendix B formalizes the instruction definitions. Finally, each circuit card used in the prototype system is documentated with schematics and operational descriptions of the logic used on each circuit card.

STATEMENT OF PERMISSION TO COPY

In presenting this thesis in partial fulfillment of the requirements for an advanced degree at Montana State University, I agree that the Library shall make it freely available for inspection. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by my major professor, or, in his absence, by the Director of Libraries. It is understood that any copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Signature Darrell Boots Irwin

Date July 21, 1976

AN 8-BIT MICROCOMPUTER SYSTEM

PROCESSOR, MEMORY AND I/O

by

DARRELL BOOTS IRVIN

A thesis submitted in partial fulfillment
of the requirements for the degree

of

MASTER OF SCIENCE

in

Electrical Engineering

Approved:

D. R. Bjk
Chairperson, Graduate Committee

Paul E. Dehlich
Head, Major Department

Henry L Parsons
Graduate Dean

MONTANA STATE UNIVERSITY
Bozeman, Montana

July, 1976

ACKNOWLEDGEMENT

The development and construction of a prototype system of the size and complexity represented by this thesis requires the cooperation and creative talent of many individuals. The author wishes to thank the U.S. Forest Service, Forest and Range Experiment Station, Bozeman, Montana, and in particular Mr. William Hartsog for supplying the funding which made this thesis possible.

Finally, the author is especially grateful to Mr. Donald Bjork for his support of the project and a special thanks is due Mr. John Somppi for his invaluable assistance in constructing and debugging the prototype.

D.B.I.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

This thesis presents an 8-bit microcomputer based on the INTEL 8008 microprocessor. Suitable applications for the system include data acquisition and digital control systems formerly realized with dedicated hardware or larger and more expensive minicomputer systems.

Processor, memory and I/O subsystems are first described from an architectural viewpoint with the aid of block diagram. The instruction set for the system is introduced in an informal manner within the main text while Appendix B formalizes the instruction definitions. Finally, each circuit card used in the prototype system is documentated with schematics and operational descriptions of the logic used on each circuit card.

CHAPTER 1

INTRODUCTION

Since the perfection of Large Scale Integration (LSI) technology to the point where economics allow marketing of a single chip microprocessor (uP) with a versatile instruction set and a sophisticated architecture, the digital designer has new options in design. It has become economically possible to incorporate programmable hardware into smaller systems, resulting in a host of new applications in the areas of microcomputers and new realizations for digital control systems formerly constructed as dedicated hardwired logic.

Presented is a modular 8-bit microcomputer system based on a microprocessor, the INTEL 8008. By selecting the desired amount and type of memory, the necessary Input/Output (I/O) channels and developing suitable software, the system may be configured to the complexity of the task at hand. Typical uses for this system could include data acquisition and preprocessing or digital control.

Features of the system include low power consumption to allow battery operation, and interrupt capability for real time applications. Modular design with separate circuit cards for the processor, memory and I/O allow the system to be expanded easily to the maximum capability of 16k of memory and eight I/O channels.

DESIGN GOALS

The goals and design constraints set for the system in the original specification stages were very general in nature. Rather than define the system with a hard and fast set of specifications, only the desired features and a few of the more general hardware specifications were detailed.

First, the system was to exhibit low power consumption. This was accomplished by designing the system with low power logic families. Low Power Transistor to Transistor Logic (LPTTL) and Complimentary Metal Oxide Semiconductor (CMOS) logic was used wherever possible. In keeping with the desire to conserve power the voltage regulators should run in the switched mode.

Second, the system was to accommodate both Random Access Memory (RAM) and programmable Read Only Memory (PROM). PROM allows non-volatile storage of code necessary to start the system and eliminates the need to enter program loaders by hand. The PROMs used in this system may be electrically reprogrammed and are more correctly referred to as Electrically Alterable Read Only Memories (EAROM). The RAM is static, requiring no refresh clocks but code stored in RAM will be lost if power is removed from the memory.

Third, the system was to have a variety of data I/O options. At this time two are available including an 8-bit Parallel I/O

channel (PIO) and a Serial I/O channel (SIO). Data may be input and output eight bits at a time in parallel through a PIO channel, while the SIO allows a similar transfer of data in serial form at a variety of baud rates and word formats. The SIO interface is compatible with most commonly available serial devices using asynchronous start/stop bit data formats.

Fourth, the system was to have real time capabilities in the form of interrupts. This capability allows servicing of infrequent or short, high priority data transfers while allowing the processor to spend most of the time on longer but less critical routines.

Included in the I/O section, but not handling data is the Real Time Clock (RTC) which allows the processor to be interrupted after a programmable amount of time has passed since the last clock interrupt. The RTC allows software to keep "real time" by interrupting the processor at known intervals. This feature makes programming of systems that must react in real time a much simpler task.

LOGIC CONVENTIONS

Positive logic is used throughout this discussion. Thus a logical 1 (also referred to as a high signal) is defined to be the more positive output voltage (generally 3.5 to 5.0 volts) and a logical 0 (or low signal) to be the more negative voltage (generally

0.0 to 1.5 volts). The actual 0 and 1 voltages developed by a given output will vary as a function of the type of output (LPTTL or CMOS) supplying the drive and the number and type of inputs that comprise the load.

Positive logic convention also applies to the names (mnemonics) used in the discussion of the logic timing and control signals. Each signal may be either true or false. Signals will be called true when they are performing their function and false when they are not. True and false are thus logical designations as opposed to the electrical designations of 0 and 1. A signal may be logically true and electrically a 0 or 1 depending on its sense. Signals that are true when a 1 will be referred to as upright (positive true) and will have names without bars over the top (example: XYZ) while signals that are true when a 0 will be referred to as inverted (ground true) and will have names with a bar over the top (example: $\overline{XYZ}$). Conversion between the two senses is accomplished by the logic inverter. Thus $\overline{XYZ}$ becomes XYZ or vica versa after being inverted.

In the naming of data carrying signals similar conventions apply. Data is transferred eight bits at a time. Thus an eight bit binary integer having a value of 0 to 255 is the basic word (byte) used in the system. When a byte is transferred along a bus

each bit is given its own signal path. Each signal (or bit) is given a numbered name in the order of increasing significance. Thus DATA0 is the Least Significant Bit (LSB) and DATA7 the Most Significant Bit (MSB) of the eight bit byte (or bus) called DATA.

Since each bit in a binary word may assume either a 1 or 0 state depending on the value of the word, naming the upright and inverted senses requires a slightly different convention than that used with the timing signals. When a bit signal mnemonic appears without a bar on top (e.g. DATA) then the sense is upright and the electrical 1 corresponds to a true bit.

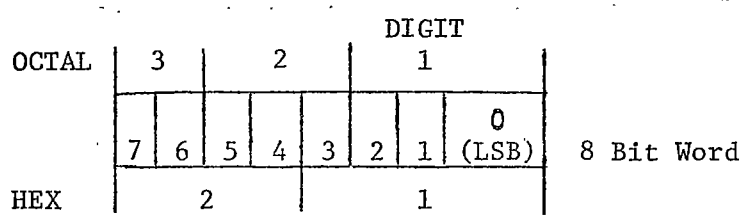
DATA FORMATS

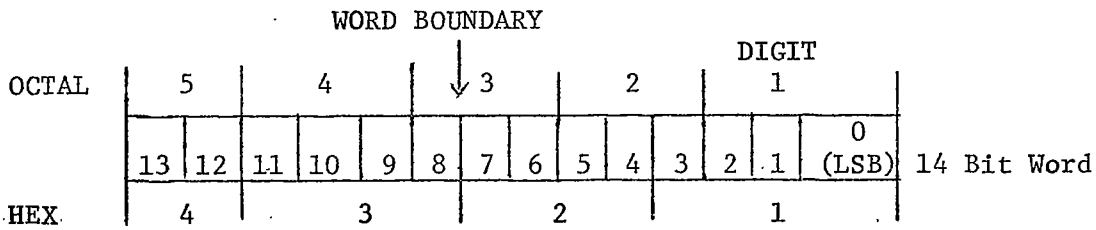
An 8-bit byte can be represented in several ways. It may be written as a string of ones and zeros in binary form. Alternately, a byte may be expressed as a three digit octal or two digit hexadecimal number with each digit representing a three or four bit field respectively. Octal numbers will be written as 123B (or 123_8) while hex numbers will be written as ABC' (or ABC_{16}). A number written in decimal notation will have its digits written down as in 356. Thus each form of notation is unique to eliminate confusion.

When expressing hex numbers, the standard notation of using letters A through F for binary values of 9 through 15, respectively, is observed. The octal notation uses only the digits of 0 through

7. Figure 1 defines the bit fields for the hex and octal representations of both eight and fourteen bit words.

While all of the data transfers in the system take place eight bits at a time in a bit parallel-byte serial form, when the processor is addressing memory it uses a 14 bit address word (H and L bus). This larger word is formed by concatinating two bytes output at different times. This system ignores the top two bits of the most significant word resulting in a 14 bit address.





EXAMPLE:

10110101110111 = 26567B = 2D77'

Or Split Into 6 and 8 Bit Fields:

(101101), (01110111) = 55B, 167B = 2D', 77'

FIGURE 1: HEX AND OCTAL BIT FIELDS

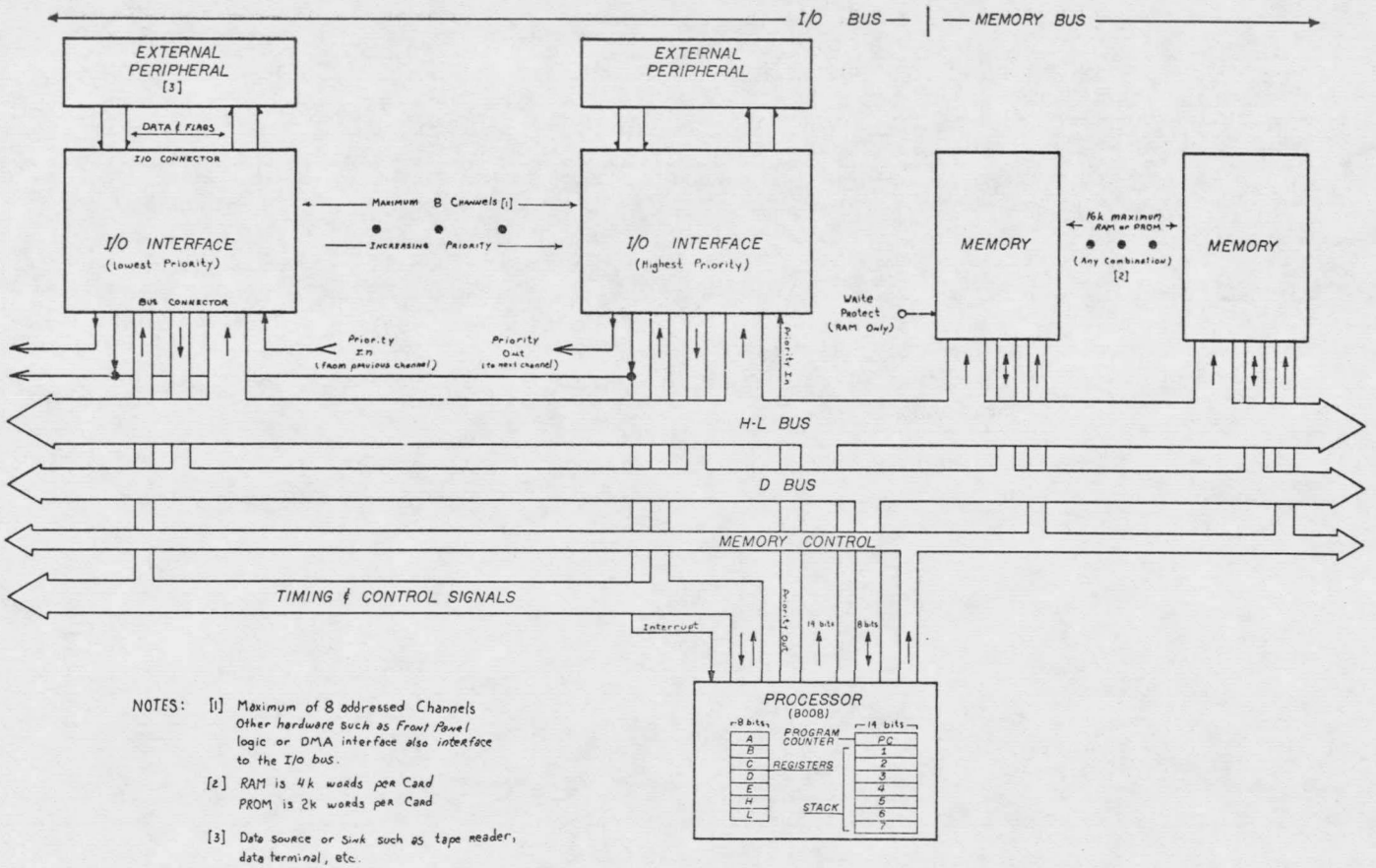
CHAPTER 2

MACHINE ARCHITECTURE

The machine block diagram is presented in Figure 2. Architecturally, the system is composed of a series of modules (circuit cards) interconnected by a bus structure divided into two sections; Memory and I/O. Appendix A defines the bus signal set. Under control of a set of instructions (program) stored in memory, the processor performs a series of data transfers between the I/O interfaces and memory. As the data passes through the processor, a combination of things may happen: data may be modified arithmetically or logically, stored temporarily in a processor register, or used to make a decision about future program flow (conditional branching).

Memory is used for both program and scratch or buffer storage as determined by the programming. The configuration of the memory is expandable to a maximum of 16k words with many different combinations of RAM and PROM possible.

Eight bit data words are passed in and out of the system through the I/O interfaces, each of which provides an input port and an output port. The processor may control up to eight I/O interface channels for a total of 16 ports. Each port present in the system may generate interrupts if desired, forcing the program to execute in real time. Interrupt priority is controlled by position relative to the processor in the priority chain.



NOTES: [1] Maximum of 8 addressed Channels
Other hardware such as Front Panel logic or DMA interface also interface to the I/O bus.

[2] RAM is 4k words per Card
PROM is 2k words per Card

[3] Data source or sink such as tape reader, data terminal, etc.

FIGURE 2: SYSTEM BLOCK DIAGRAM

PROCESSOR

The system processor block diagram is presented in Figure 3. An Intel 8008 microprocessor (enclosed within dotted lines) serves to provide the intelligent functions of the processor. The remaining portions of the processer serve as an interface between the 8008 and the bus. Functions performed by this logic include generation of the clock signals, decoding of the cycle and state signals and control of the three data paths within the system (D, H and L buses). Memory control and system timing signals are also generated by the processor.

Internally the 8008 contains seven general purpose registers each of which is eight bits wide. These are named A, B, C, D, E, H and L. Any of these registers may be used as general registers as determined by the program. Additionally register A functions as the accumulator and is always one argument of an arithmetic operation involving two words. Registers H and L also have an additional function. When addressing memory to read or write data (not instructions which are addressed by the Program Counter) they contain the memory address. Register L provides the eight LSB while H supplies the six MSB. The top two bits of H are ignored when using H and L as address registers.

The 8008 also contains a 14 bit wide address stack 8 levels