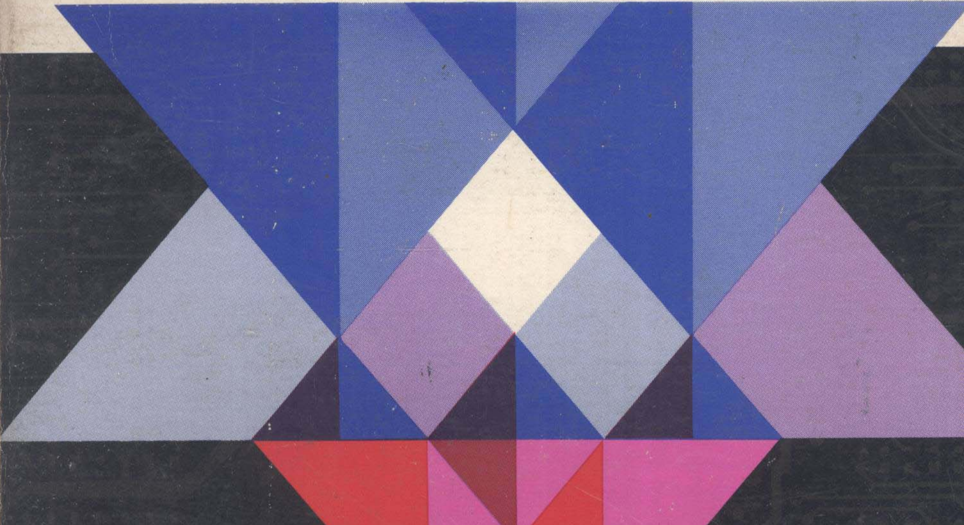
6809 MICROCOMPUTER PROGRAMMING & INTERFACING

WITH EXPERIMENTS

BY ANDREW C. STAUGAARD, JR.



BLACKSBURG CONTINUING EDUCATION SERIES™
edited by Titus, Titus & Larsen

6809 Microcomputer Programming & Interfacing, With Experiments

by

Andrew C. Staugaard, Jr.

Howard W. Sams & Co., Inc.
4300 WEST 62ND ST. INDIANAPOLIS, INDIANA 46268 USA

557

FIRST EDITION
SECOND PRINTING—1981

International Standard Book Number: 0-672-21798-6
Library of Congress Catalog Card Number: 81-50567

Printed in the United States of America.

Preface

Welcome to the world of *advanced* microprocessors! In the early seventies we witnessed the dawn of a second industrial revolution with the introduction of first-generation programmable logic devices. These “smart” devices on a single piece of sand (silicon chip) were appropriately called *microprocessors*. They revolutionized the engineering of many everyday products, from toys and appliances to the automobile and large computer systems.

In the past few years the microprocessor chip industry has *exploded* into a multibillion dollar business. As stated in the June 30, 1980, issue of *Newsweek* magazine, “The explosion is just beginning. In 1979, the world market for microelectronics topped $11 billion. Over the next five years, chip sales are expected to grow by at least 20 percent annually, and the market for microprocessors ‘computers on a chip’ will expand by *50* percent each year—even though the chips themselves and the computing power they represent are diving in price.”

Two large microprocessor application markets have emerged—the *dedicated* market and the *systems* market. The resources of the first- and second-generation devices, such as the 6800, 8080, and Z-80, were made to satisfy both market applications. Each market, however, requires separate microprocessor features for efficient utilization of the device. For example, the dedicated market requires a device which incorporates many functions such as CPU, R/W memory, ROM/EPROM, timer, serial i/o, etc., onto *one* chip to minimize the chip count for such applications as the automobile, appliances, machine tool control, toys, etc. The systems market, on the other hand, requires a very powerful software device such that high-level language programming can be efficiently implemented. Most microprocessor chip manufacturers have taken these two directions in their newer-generation chip designs. In the Motorola family the 6801 and 6805 satisfy the dedicated market applications, with the 6809 and 68000 having been designed particularly for the systems market.

The 6809 is a *high-performance* 8-bit microprocessor. It has many very powerful software features which are particularly useful for high-level language (Pascal, FORTRAN, BASIC, COBOL, etc.) implementation. In fact, as you are about to discover, the 6809 approaches the performance of many 16-bit devices, such as the 8086, Z-8000, and 68000, without the inherent overhead costs required to engineer such a 16-bit system. Flexible 8-bit devices, like the 6809, will be around for many years to come despite the onslaught of the 16-bitters, since many applications do not require such high performance. In addition, when 16-bit systems do become common, they will rely on 8-bit devices to perform many dedicated tasks such as peripheral control, data acquisition, etc.

This book is meant to be a tutorial type of text for a first exposure to the 6809 or to high-performance microprocessors in general. I am confident that you will also find it extremely valuable as a "cookbook" type aid when you are working with the 6809. Since the 6809 is a "souped up" 6800, a basic understanding of the 6800 will be assumed throughout the text. If a review is needed, you may wish to consult my *How to Program and Interface the 6800*, published by Howard W. Sams & Co., Inc.

A set of objectives is provided in the first part of each chapter, with review questions and answers provided at the end of each chapter. There are also numerous examples that illustrate the text. I encourage you to study the examples in detail, since many of the important software concepts are demonstrated within the example programs. I also encourage you to pay particular attention to Chapter 2, "6809 Addressing Modes." You will find that the *secret* to understanding the 6809 software concepts is understanding its 19 addressing modes.

Finally, I would like to express my appreciation to Motorola Semiconductor Products at Austin, Texas, and Phoenix, Arizona, for their technical assistance and permission to use their 6809 documentation in this text.

ANDREW C. STAUGAARD, JR.

To my mother and father, who provided me the good home life and education required to be successful in today's world. And to one of my best friends, my father in law, Zane Benefiel, whose encouragement in my early professional days led to the completion of this and two previous manuscripts.

Contents

CHAPTER 6

CHAPTER 7

APPENDIX A

APPENDIX B

APPENDIX C

APPENDIX D

CHAPTER 1

Fundamental 6809 Concepts and Chip Structure

INTRODUCTION

As you will discover in this and subsequent chapters, the 6809 is one of the most powerful 8-bit processors to come on the market to date. In conceiving the 6809 the Motorola design engineers wanted to maintain compatibility with the popular 6800 at some level, yet "soup up" the 6800 architecture and instruction set to approach the performance of a 16-bit processor, such as the 68000 or Z-8000. From user surveys Motorola concluded that many customers desired such performance from an 8-bit, 40-pin device. It was found that many users did not want to pay the price for conversion to a 16-bit device with up to 64 pins if a high-performance 8-bit device were available. Therefore, the 6809 was designed to approach 16-bit performance at minimum cost to the user.

In this chapter we begin by discussing the 6809 evolution and design philosophy. You will see that the 6809 has been designed primarily for the systems market, where high-level language implementation is common. Throughout the discussion, comparisons will be made between the 6809, 6800, and other competitive processors. In addition, as an introduction to subsequent chapters, this chapter will summarize the 6809's architecture, software, and hardware. It is important that you understand what improvements have been made in the 6809 over the 6800 since this will enable you to better understand the material to follow.

OBJECTIVES

At the end of this chapter you will be able to do the following:

- Explain the evolution of the 6800 family.
- Understand the design philosophy that created the 6809.
- State the differences between the 6809, 68A09, 68B09, and 6809E.
- List the additional registers present in the 6809 that do not exist in the 6800.
- Describe the two additional condition code flags available in the 6809.
- Define indirect addressing.
- Compare 6809 branching to 6800 branching.
- Explain what is meant by "memory paging" and how this is accomplished with the 6809.
- Describe the fundamental hardware differences between the 6809 and 6800.
- Explain the difference between a standard interrupt request and a fast interrupt request.

6809 EVOLUTION AND DESIGN PHILOSOPHY

As stated in the introduction to this chapter the 6809 was designed to upgrade or "soup up" the 6800 to be superior to any 8-bit microprocessor. Also, Motorola wanted to capitalize on their customers' familiarity with the 6800 so that exposure to the 6809 would not create severe learning problems for those 6800 users. Therefore this design philosophy dictates that the fundamental 6800 architecture be used as a basis for the 6809 architecture and that software compatibility be available at some level. As you will discover in Chapter 3 this compatibility exists at the source code (mnemonic) level and *not* the object code (op-code) level. You will also discover that the 6809 *does not* contain dozens of new instructions. However, it uses over three times as many addressing modes as the 6800 to provide more efficient utilization of the existing instructions. The power of a processor is not a function of the number of unique instructions available in its instruction set. The *real* power of a processor lies in how many different ways a given instruction can operate on the same data and also how the given instruction set can operate on different data in the same manner. This flexible instruction power is provided by the different addressing modes available to the fundamental instruction set. The 6809 uses its 19 addressing modes in conjunction with 59 fundamental instructions to provide a total of *1464* unique operations. Motorola believes that the 6809 contains the most powerful addressing modes available in any microprocessor to date.

Since the 6809 is primarily designed for the systems market, program position independence, program re-entrancy, and easy implementation of block-structured high-level languages (such as Pascal) were also prime design considerations. These terms have the following meanings:

- *program position independence*—that quality of a program to execute properly when placed anywhere within the memory address map. Thus the program is *independent* of its position within the memory map.
- *program re-entrancy*—that quality of a program which allows a subroutine to be shared by several tasks concurrently, without destroying the return addresses by nesting routines.

With position independence, programs can be loaded from a mass storage disc and located anywhere within R/W memory without requiring the use of a relocating loader routine. In addition, position independence will allow ROM programs to be written for general distribution. The user can assign any arbitrary set of addresses to the ROM since the program execution is "independent" of its position within the memory map. This will eliminate the necessity for full ROM address decoding and will also allow the user to locate the ROM such that it will not interfere with other software. As you will see shortly, the advanced 6809 architecture also facilitates the use of modular programming. Such programming will allow the system software designer to divide a project up into modular programs which can be designed and tested independently before being incorporated into the final system design. This same architecture allows programs to be structured and interrupted in any part of the address map and still execute properly on return, thus satisfying the program re-entrancy design goal. In addition, high-level block-structured languages, such as Pascal, BASIC, FORTRAN, and COBOL, can be compiled into more efficient and faster-running machine code than was possible with earlier processors.

The 6800 family evolution scheme is shown in Fig. 1-1. Note the position of the 6809. As far as central processing unit (CPU) performance is concerned the 6809 is the most advanced *processor* in the Motorola 8-bit family. However, the 6809, by itself, is a microprocessor and *not* a microcomputer. It requires external R/W memory and ROM to function as a microcomputer. Therefore a minimum 6809 system would consist of three chips. Recall from our discussion in the Preface that microprocessor/microcomputer applications take two directions: small, dedicated applications and systems applications. In the 8-bit Motorola family the 6809 satisfies the needs of the latter, while the 6801, 6802, 6803, and 6805 were designed for the dedicated applications. These four devices all contain various amounts of R/W memory, ROM, and parallel/serial i/o capabilities. They are therefore more advanced as far as chip integration is concerned since some or all of these capabilities are integrated into the

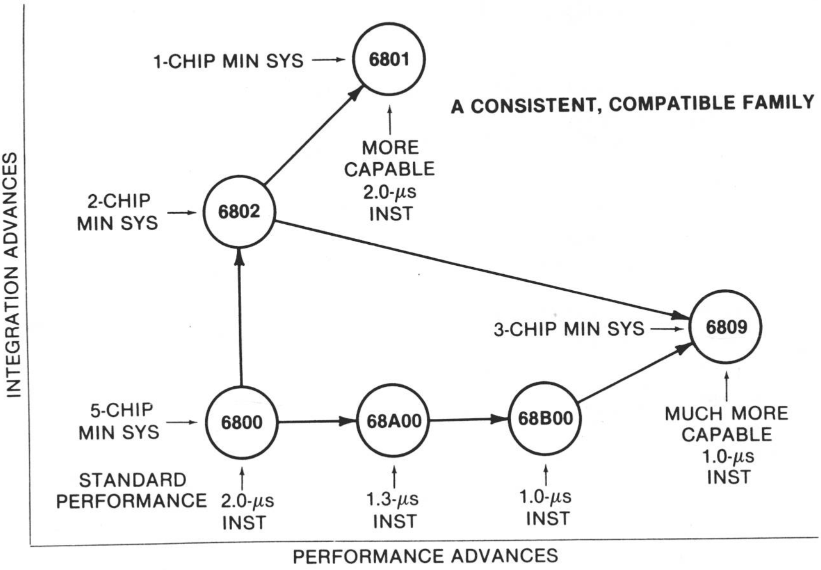


Fig. 1-1. M6800 family evolution.

same chip, along with the CPU. For example, the 6801 contains an enhanced 6800 CPU, R/W memory, ROM, timer, parallel interfacing ports, and a serial port, all on one 40-pin chip. Hence, we say that the 6801 is a "single-chip" microcomputer.* The 6809, however, is a far more advanced processor, approaching 16-bit performance.

As you will see in Chapter 6, the standard 6809 does contain an on-chip clock/oscillator and therefore only requires an external crystal to provide the clock signal. The standard 6809 operates at 1 MHz. However, it is also available in 1.5-MHz and 2.0-MHz versions: the 68A09 and 68B09, respectively. In addition, there is an off-chip clock version of the 6809 available, the 6809E.

Now let's compare the performance of the 6809 to the 6800 and some other well-known processors. Compared to the 6800, the 6809 boasts the following performance:

- 72 percent decrease in the number of instructions required for a program compared to similar 6800 programs.
- 58 percent decrease in the required program memory compared to 6800 programs.
- 167 percent increase in 6809 processor throughput compared to 6800 throughput.

* For a detailed discussion of the 6801/68701 and 6803 consult Staugaard, A.C. *6801/68701 and 6803 Microcomputer Programming and Interfacing,* published by Howard W. Sams & Co., Inc., Indianapolis.

Motorola claims that these statistics allow the user to achieve 2.5 to 5 times the performance from a 6809 system compared with a similar 6800-based system. Of course, the exact amount of increased performance depends on how efficiently the increased capabilities of the 6809 are utilized and on the specific application.

Comparisons of the 6809 to the 6800 and other processors are summarized in Fig. 1-2 and in Tables 1-1 through 1-3. Note, especially from Fig. 1-2, that the 6809 approaches 16-bit performance. The comparison values in these tables and Fig. 1-2 were supplied by Motorola and thus tend to illustrate the better aspects of the 6809 over the other processors.

6809 IMPROVEMENTS

The increased performance of the 6809 over other 8-bit processors is made possible by specific improvements in architecture, software, and hardware. Each of the following improvements over the 6800 is discussed in detail in subsequent chapters; however, we will summarize them here.

Architectural Improvements

Compared with the 6800 architecture the 6809 adds an 8-bit register and three 16-bit registers as shown in Fig. 1-3. The additional 8-bit register is the direct page register, which will allow you to use the direct addressing mode anywhere within the 6809 memory map. (Recall that direct addressing is limited to the first 256 bytes of memory with the 6800.) The three additional 16-bit registers include a 16-bit accumulator, index register, and stack pointer. The additional accumulator is referred to as accumulator D and is simply the concatenation of the two 8-bit accumulators, A and B. The additional index register is called the Y index register (Y) and the additional stack pointer is referred to as the user stack pointer (U). The two index registers, X and Y, will also function as pointers and the two stack pointers (S and U) can be used for indexing.

In addition, you will note from Fig. 1-3 that all 8 bits of the condition code register (CC) are being used in the 6809. Recall that only the first 6 bits are utilized in the 6800. The functions of the first 6 CC bits (C, V, Z, N, I, H) in the 6809 are identical with those of the 6800. The two additional bits (F and E) are used in conjunction with the 6809's interrupts. The use of these two additional bits will be discussed in detail in subsequent chapters.

These architectural improvements along with the 6809's powerful addressing modes speed processor throughput, since less data movement is required between the internal registers and memory, and

Table 1-1. Relative Processor Execution-Time Comparisons for Eight Software Operations

		I/O Handler	Character Search	Computed Go To	Double Shift Right 5 Bits	Vector Addition 16-Bit Elements	Vector Addition 8-Bit Elements	16 × 16-Bit Multiplication	Move Block (64 Bytes)	Average Execution Time
6809	2.0 MHz	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
	1.5 MHz	1.3	1.3	1.3	1.3	1.3	1.3	1.3	1.3	1.3
	1.0 MHz	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0
Z-80	4.0 MHz	1.4	0.8	2.1	2.7	1.6	1.8	3.3	1.0	1.8
	2.5 MHz	2.2	1.2	3.4	4.4	2.6	2.9	5.2	1.6	2.9
9900	3.0 MHz	2.6	2.3	2.8	1.5	1.7	3.0	0.5	1.6	2.0
6800	2.0 MHz	0.9	1.4	1.9	1.3	3.1	2.8	5.0	3.3	2.4
	1.5 MHz	1.2	1.9	2.5	1.7	4.1	3.7	6.7	4.3	3.3
	1.0 MHz	1.8	2.8	3.7	2.5	6.1	5.5	10	6.5	4.9
8080	3.0 MHz	1.9	1.8	2.8	6.1	2.3	2.7	9.6	2.4	3.7
8085	2.0 MHz	2.8	2.6	4.2	9.1	3.4	4.1	14.3	3.7	5.5

Table 1-2. Actual Processor Execution-Time Comparisons for Eight Software Operations

		I/O Handler	Character Search	Computed Go To	Double Shift Right 5 Bits	Vector Addition 16-Bit Elements	Vector Addition 8-Bit Elements	16 × 16-Bit Multiplication	Move Block (64 Bytes)
6809	2.0 MHz	28	287.5	34.5	15	325	180	82	344.5
	1.5 MHz	37.3	383	46	20	433	240	109.3	459.3
	1.0 MHz	56	575	69	30	650	360	164	689
Z-80	4.0 MHz	38.3	220.5	73.3	41	518	323	267	342
	2.5 MHz	61.3	352.8	117.2	65.6	828.8	516.8	427.2	547.6
9900	3.0 MHz	72	661	98	22	537	537	42	537
6800	2.0 MHz	24.5	404	64.5	19	993.5	498.5	409.5	1123.5
	1.5 MHz	32.7	539	86	25.3	1325	665	546	1498
	1.0 MHz	49	808	129	38	1987	997	819	2247
8080	3.0 MHz	52.7	506.7	96.7	91.3	732	492	784	841
8085	2.0 MHz	79	760	145	137	1098	738	1176	1262

Table 1-3. Summarized Processor Performance Comparisons

Performance Criteria	MC6809	Z-80A	MC6800	8085
Number of Instructions	1.0*	1.56	1.72	2.30
Number of Bytes	1.0	1.31	1.58	1.80
Number of Microseconds	1.0	1.80	2.40	2.20
	(2 MHz)	(4 MHz)	(2 MHz)	(5 MHz)

*Normalized to 1.00 for the MC6809—poorer performance has higher numbers.

they also aid software development since many of the internal registers can be made to perform different functions at different times.

Software Improvements

Improvements in the 6809's architecture have allowed the Motorola designers to make many significant software improvements over

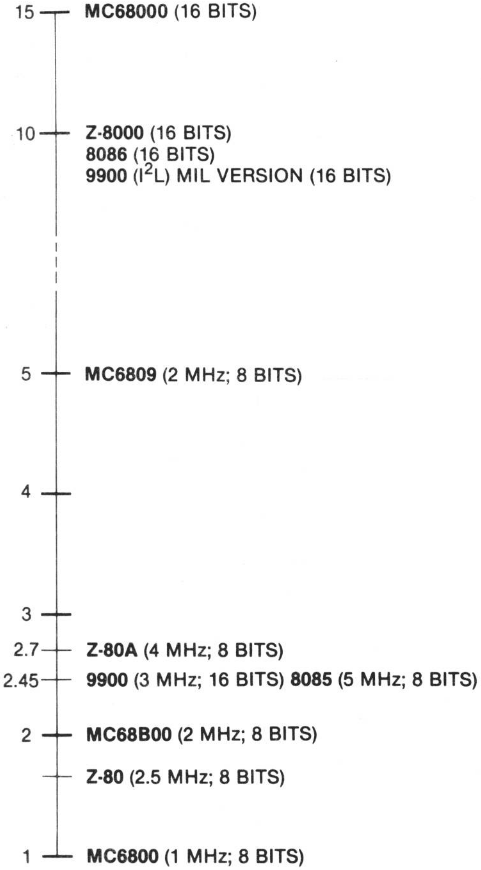


Fig. 1-2. Relative processor execution-time comparisons.

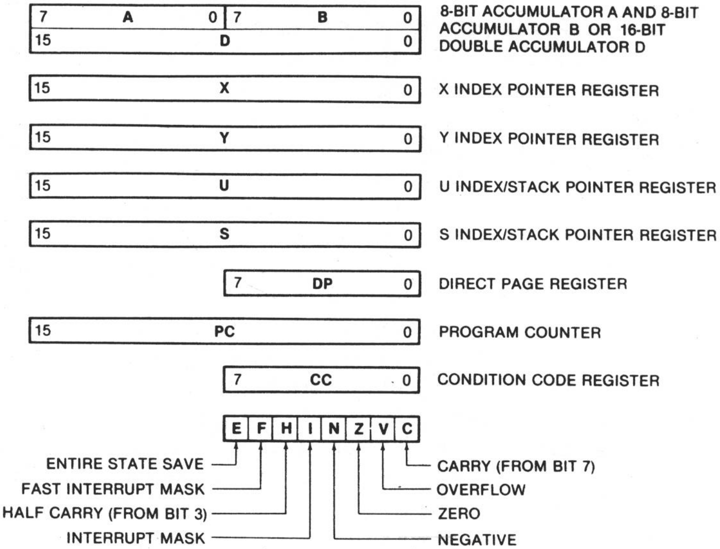


Fig. 1-3. 6809 internal registers.

the 6800. In fact, the 6809 has been termed "the programmer's dream machine." Prior to firm definition of the 6809 instruction set Motorola conducted a survey of 6800 users to determine: (1) if a 16-bit architecture was more desirable than an 8-bit architecture and (2) should 6809 compatibility with the 6800 occur at the object code level or source code level? The survey results indicated that most of the 6800 users felt that an 8-bit architecture was adequate for their future applications. They did not want to pay the price for a new 16-bit device if an 8-bit device could be designed to perform common 16-bit operations, such as load, store, add, subtract, compare, and multiply. As you will discover in Chapter 3, the inclusion of the 16-bit accumulator and associated instructions will allow such 16-bit operations to be performed internally. In answer to the second survey question, almost all of the responses indicated that source code compatibility would be adequate, mainly because they did not foresee using 6800 ROMs in future 6809 systems. Source code compatibility meant that users could take full advantage of their familiarity with the 6800 mnemonic code for assembly language programming and it also allowed the 6809 designers to completely remap or reassign the 6800 op codes to produce more efficient and faster running 6809 programs. Also, source code compatibility would allow any 6800 programs to be processed through a 6809 assembler to produce 6809 code. Therefore, you will see a familiar mnemonic code in Chapters 3, 4, and 5; however, most of the corresponding op codes are different from those of the 6800.

As mentioned earlier, the 6809 adds many new and powerful addressing modes. For example, the indexed mode of addressing can use the four 16-bit indexible registers (X, Y, U, and S) to point to the address of an operand or to the address of the address of an operand. The latter is referred to as *indirect addressing*. These registers can also be incremented and decremented automatically or under program control. The 6809 can even use the program counter to access operands or operand addresses. Furthermore, with the use of the direct page register, you can access any part of the 6809 memory map using direct addressing. You can also *branch* to any part of the program using relative addressing. Recall that with the 6800, you were limited to a plus 127_{10} and minus 128_{10} branch range within the program, since the relative address offset is only 1 byte. However, the 6809 allows for a 2-byte relative address offset and thus permits branching anywhere within the 64K memory map (long branch). This allows position-independent programs to be written for the 6809. We will discuss the idea of position independence in more detail when discussing relative addressing.

Other software features of the 6809 include:

- An 8 × 8 unsigned multiply instruction which generates a 16-bit result.
- 2-byte instructions which will push or pull any or all registers onto or from either stack (U or S).
- 16-bit add, subtract, load, store, and compare instructions which utilize the 16-bit accumulator (D).
- Instructions which permit you to add any of the accumulators (A, B, or D) to any of the index registers and stack pointers (X, Y, S, or U).
- Instructions which permit you to perform exchanges and transfers between any two like-size CPU registers.

Hardware Improvements

Besides the architectural and software improvements of the 6809 over the 6800, many hardware improvements were also made. Most of these improvements involve the 6809's interrupts, control signals, and associated control lines. We will summarize these new hardware features here; however, a complete discussion of the 6809 pin-outs and associated chip operation is provided in Chapter 6.

Like the 6800, the 6809 is a 40-pin device available in both a plastic (P-suffix) and ceramic (L-suffix) package. A bus and control signal diagram for the 6809 is shown in Fig. 1-4. The most obvious change to the chip hardware is the inclusion of an on-chip clock/oscillator for the standard 6809 package. (Recall that the 6809E is an off-chip clock version of the 6809.) The 6809 on-chip clock re-