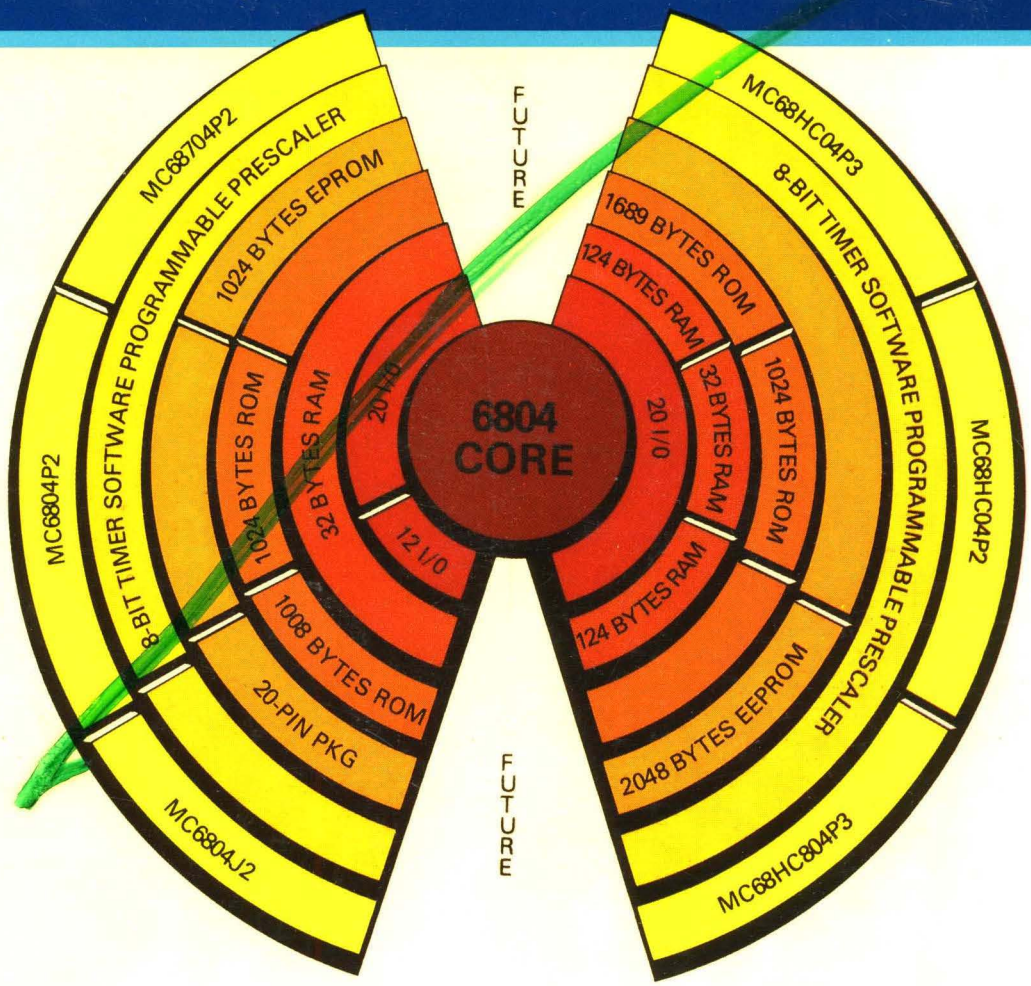
DLE404/D

M6804 MCU MANUAL



MOTOROLA

M6804 MCU MANUAL

First Edition 1984 – F110
Reprinted 1985 – DLE404/D
Printed in Great Britain by Eyre & Spottiswoode Ltd, 7500, 9/85

Table of Contents

M6804 MCU Family Introduction and Features

1

1.1 INTRODUCTION

MOTOROLA has evolved a large and comprehensive family of microprocessors from the original M6800 family through to the M68000 16-bit family. (see figure 1-1). The introduction of the M6804 family marks another major milestone in the growth of our range.

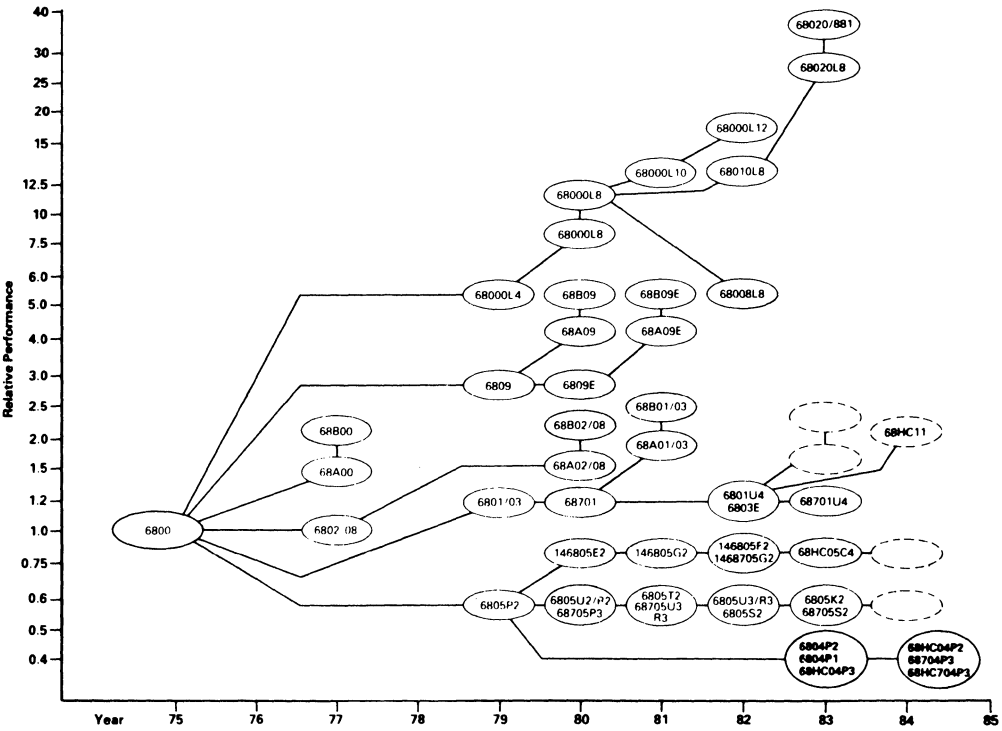


Figure 1.1. Performance Summary Geneology of a Cohesive Microprocessor Family

Advanced Design

Prior to the introduction of the computer-based architecture of single chip microcomputers all 4-bit MCU's and most 8-bit MCU's were evolved from a calculator base. These calculator-based, control-orientated microprocessors have the disadvantage of using a split memory architecture containing separate data paths between the CPU and peripherals (memory, or I/O registers), which forces the inclusion of many special purpose instructions and results in an irregular architecture.

The advanced computer-based design of the M6804 family means that the devices contain a single data bus so that all I/O, program and data may be accessed with the same instruction, therefore, there are fewer instructions to remember. The actual number of unique instructions is increased by a variety of addressing modes which define how an instruction can access any data required for the operation.

Cost Versus Performance

Whilst the M6805 family provides the most cost effective solution for the mid-range control orientated microprocessor market, there are applications where the lower cost of a 4 bit MCU is required. By using ingenuity, MOTOROLA have managed to lower the cost without sacrificing performance. Some of the methods used in achieving this goal are:

- SMALL DIE SIZE – By processing the 8 bit variables serially we have reduced the number of connections and hence the die size below that of most 4 bit machines.
- SELF TEST – The extensive on-chip self test routine as pioneered on the 6805 family reduces test time and also helps customers by allowing easy incoming inspection without the large capital outlay of a LSI tester. A simple and inexpensive go/no go field test can also be performed.
- SOFTWARE COMPATIBILITY – Close similarity with 6800/6805 software ensures that engineers familiar with programming on these machines can quickly adapt to programming 6804. It also means that only one development system is required for both low-end (6804) and midrange (6805) projects, with the ability to change from one processor to the other half way through a project without excessive delay.

These features allow us to offer you, for the first time, 8 bit processing at a 4 bit price.

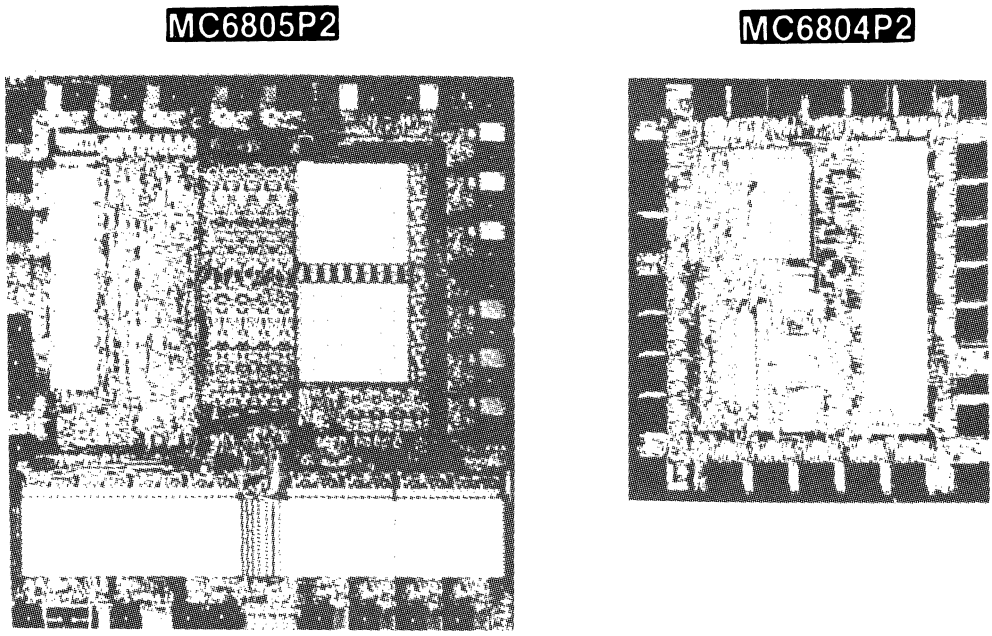


Figure 1.2. Die Comparison of MC 6805P2 and MC 6804P2 (Same Scale)

1.2 FAMILY FEATURES

1.2.1 M6804 Architecture

One of the principle design objectives of the M6804 family was to reduce die size due to its effect on product cost. Figure 1-2 illustrates the difference in size between the MC6804P2 and MC6805P2. Whilst the latter device requires considerable die area for the ALU, I/O ports, and interconnection buses, the most significant contribution on the MC6804P2 comes from ROM and RAM.
This is a direct result of the novel architectural approach used in which the M6804 processes 8-bit variables serially, one bit at a time. This inherently provides several major advantages in the quest to reduce die size:

1. Instruction data buses to RAM and I/O are 1-bit rather than 8-bit wide.

2. The ALU reduces to a one bit adder with the register storage relegated to RAM locations in place of dedicated latches.

3. The program counter is 12 bits long and incremented by another one bit adder.

4. RAM is implemented as pseudo-static, i.e. it uses compact dynamic RAM cells refreshed during the serial processing cycle. In the CMOS version however static cells are used to allow a power-down mode to be implemented.

5. Since the programmer's register set (accumulator, X and Y registers) is implemented as RAM locations, many instructions need not be implemented as opcodes directly in the ALU but as implied instructions. For example, the assembler, on recognizing BMI (Branch if Minus), inserts the code for BRSET 7, $FF (Branch if bit 7 set location $FF) where location $FF corresponds to the accumulator.

All members of the M6805 HMOS and CMOS family are designed around a common core which consists of CPU, Timer, Oscillator, ROM (EPROM), Control section (for interrupt and reset), and a variable amount of I/O lines. This versatile common core design philosophy has already provided many different M6805 family devices in a very short time. The same successful approach has been taken in the design of the M6804 family.

1.2.2 Instruction set

The instruction set used with M6804 family is specifically designed for byte-efficient program storage. Byte efficiency enables a maximum amount of program function to be implemented within a finite amount of on-chip ROM. Improved ROM efficiency allows the M6804 family to be used in applications where other processors might not perform the task in the available ROM space, or more features may be included in applications where ROM space is more than adequate. In some cases the user might wish to include programs for more than one application. In such cases the appropriate program could be selected by the power-up initialization program. The ability to nest subroutines, the addition of true bit test and bit manipulation instructions, the multi-function instructions, and the versatile addressing modes, all contribute to byte efficiency.
Superficial comparisons of the number of bytes per instruction for the M6804 family compared to other machines in this class may be very misleading. A simple M6804 instruction occupying 2 or 3 bytes accomplishes as much real programming work as several single byte instructions, or a subroutine, would accomplish in many other processors.
The bit test and bit manipulation instructions permit the programmer to:

- branch on bit set
- branch on bit clear
- set bit
- clear bit

These instructions operate on any individual bit in the first 256 address spaces (page zero). As such, the bit manipulations access I/O pins, RAM bits, and ROM bits.

1.2.3 Adressing Modes

One of the chief measures of the effectiveness of a computer architecture is its ability to access data. The M6804 has several major memory addressing modes. They include immediate, direct, short direct, register indirect, bit-test-direct, and bit-direct.

The register indirect addressing mode replaces the indexed addressing mode as it is known on the M6805 family. It permits access to conversion tables and data tables located in the Data Space. The use of tables is an important tool in controller type applications. In the Register Indirect addressing mode, the operand is at the address (in data space) pointed to by the contents of one of the indirect registers (X or Y).

Efficient addressing methods are coupled with instructions which manipulate memory without disturbing the program registers. Thus, RAM may be used for the same functions that other processors use general purpose registers (increment, decrement, complement etc.). The M6804 family members have a very versatile, efficient, and easy-to-use I/O structure. All microcomputer I/O function registers are memory mapped into the first 10 processor addresses. Advantage is thus taken of the efficient addressing modes, the many memory reference instructions, and the use of RAM (or I/O registers) as general purpose registers.

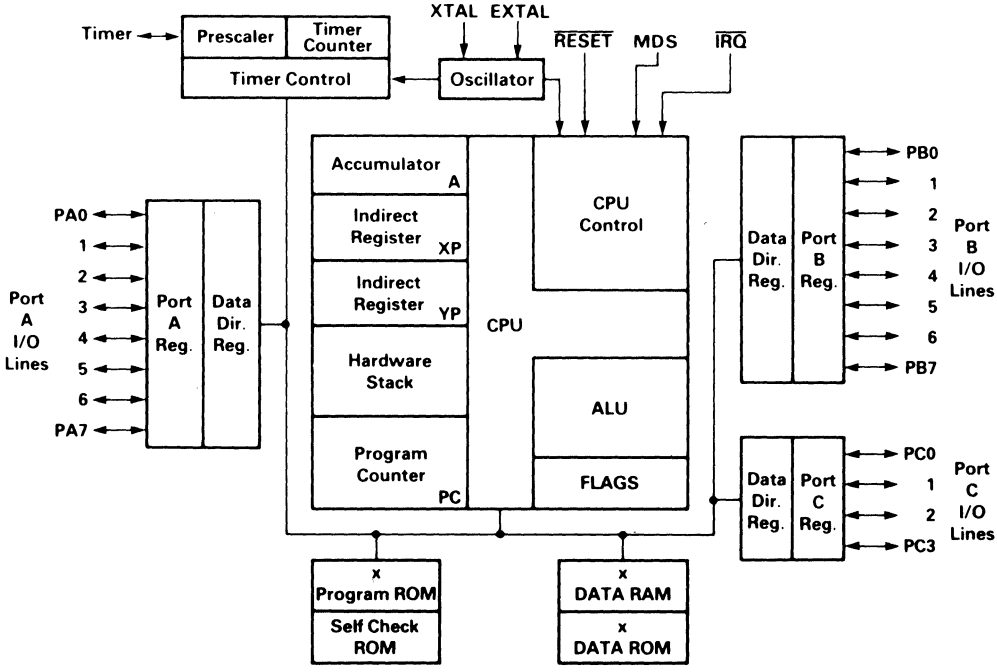


Figure 1.3. Hardware Functional Blocks Common to All M6804 Family Members

1.2.4 Common Core of M6804 Family

Every M6804 family microcomputer contains hardware common to all versions, plus a combination of options unique to a particular version. Fig. 1-3 depicts the hardware functional blocks common to all M6804 family devices.

The Central Processor Unit (CPU) contains a 1-bit arithmetic logic unit, Program Counter, Stack, Accumulator, Indirect Registers, Instruction Decoders, and Control Logic. These elements resemble the M6800 family of microprocessors which constitute the M6805 and M6804 family heritage.

The M6804 family has on-chip RAM and ROM with varying sizes to suit different applications. The addressing modes and register-like memory operations use the RAM to the fullest extent possible.

Parallel I/O capability, with pins individually programmable as input or output are available on every unit. Also included are an external interrupt input and the capability for multiple nesting of subroutines, features usually found only in much more powerful architectures.

A feature which generally simplifies software development and extends the capability of a microcomputer is an on-chip timer/ counter. The 6804's 8-bit counter and 7-bit prescaler can be programmed for a variety of functions. It can generate an interrupt at software selected intervals. It can be used as an event counter to generate an interrupt after some software selected number of external events. The timer/counter can also be used for time keeping, generating pulses, and counting external events.

1.2.5 Enhanced Microcomputer Test Capability

As the complexity of VLSI (Very Large Scale Integration) rises, increasingly complex and costly test hardware is required. This is especially true of ROM-based microcomputers, which are supposed to be used in the low-end market. Here the cost of testing starts to considerably affect the end price of the device, as well as the cost of incoming inspection.

The M6804 family has a very sophisticated self test capability built into the chip. Placing the MCU in SELF-CHECK mode will cause execution of a functional test program stored in the program ROM which verifies correct operation of most of the hardware included on the chip. Verification is achieved by means of signature analysis using an on-chip Cyclic Redundancy Check (CRC) circuit. This circuit contains a 16-bit shift register configured to perform the check using the CCITT polynominal. Similarly, program ROM contents can be checked by placing the MCU in ROM VERIFY mode, which again uses the CRC circuitry.

Use of this self test hardware during manufacture allows the M6804 family to be tested quickly and inexpensively, with a high degree of confidence.

1.3 TECHNOLOGIES FOR THE M6804 FAMILY

One of the first options to be selected by the system designer is the choice between HMOS and HCMOS processor technologies.

1.3.1 HMOS Features

The NMOS (N-channel metal oxide on silicon) technology has been the mainstay of the M6800 family. The current state of the continual shrinking of NMOS is referred to as HMOS (high density NMOS). The prime consideration when choosing a M6804 family member in HMOS technology is its lower price compared with HCMOS.

The HMOS inverter circuit, shown in Fig. 1-4, illustrates the operating principles of HMOS logic.Two transistors are series connected between ground V_{SS} and V_{DD}; one is an active N-channel transistor and the

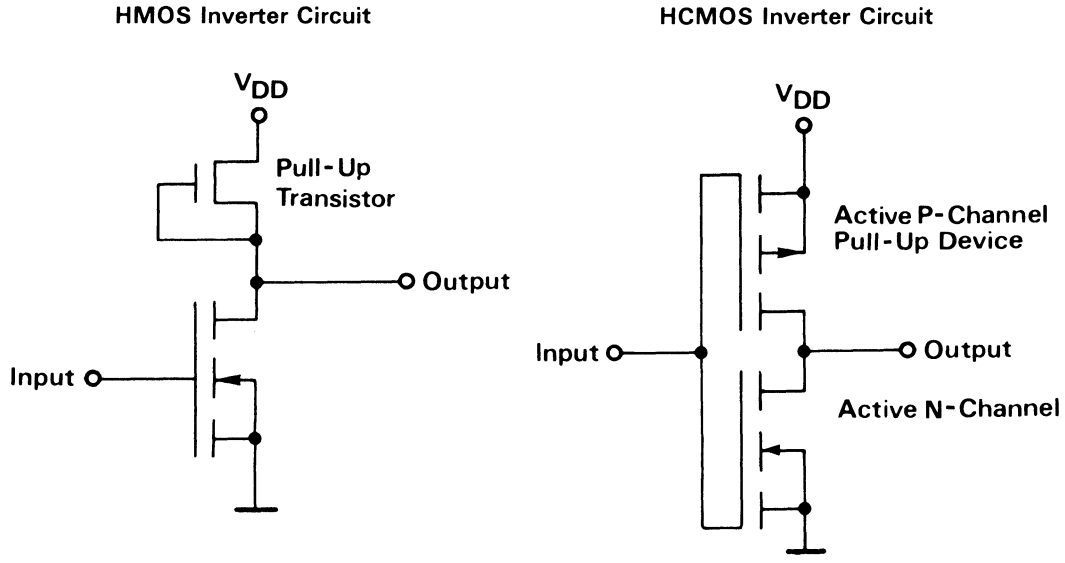


Figure 1.4. HMOS and HCMOS Inverters

other is a turned-on pull-up transistor. When a logic low is applied to the circuit input, the N-channel transistor is reverse biased and represents a high impedance, compared to the pull-up transistor (which provides the same function as a resistor). A load connected to the circuit output can be driven to a logic high through the pull-up transistor.
When a logic high is applied to the circuit input, the N-channel transistor is turned on and becomes a very low resistance to V_{SS} causing the output to go low. In this situation, current will flow through the N-channel transistor from both the pull-up transistor and any load on the output.
Other logic circuits constructed in HMOS technology use series and parallel combinations of the N-channel transistors. However, they all rely on the same operating principle, that is, the active N-channel transistor is used to sink current from the output, and a passive load transistor, which behaves similarly to a resistor, is used to source current to the output.
It is the current flowing through the pull-up load transistor, when the N-channel transistor is turned on, that accounts for most of the power consumed in an HMOS integrated circuit.

1.3.2 HCMOS Features

The HCMOS inverter circuit, shown in Fig. 1-4, illustrates the operating principles of the HCMOS logic. In HCMOS the pull-up transistor is replaced with an active P-channel transistor. In this type of circuit, one transistor complements the other, i.e. when one is turned on the other is turned off. The characteristics of the P-channel transistor are such that a high signal input turns it off, conversely, a low signal input turns it on.
The active P-channel transistor sources current when the output is high (input low), and presents a high impedance when the output is low (input high). Thus, there is essentially no current flow within the inverter whenever the output is low. The overall result is extremely low power consumption because there is no power loss through the active pull-up transistor.
The switch point of the HCMOS inverter is approximately 50% of the supply voltage (V_{DD}) rather than being determined by the threshold of the N-channel transistor. Because of this, the operating voltage range of a HCMOS device is much wider than that of a HMOS device. This permits a greater choice of supply voltages or allows the use of a less regulated power supply.
The properties of HCMOS (complementary metal oxide) technology are inceasingly attractive, in spite of a higher price. Some applications are simply not feasible with PMOS, NMOS or HMOS microcomputers, e.g. telephone sets powered from the telephone line where the power consumption is strictly limited by PTT regulations. In others HCMOS offers significant performance advantages, e.g.:

- power consumptions ranging from 1/15 to 1/200 of the equivalent HMOS part;
- fully static operation;
- higher noise immunity;
- greater tolerance of ambient temperature variations;
- wider operating voltage range;
- higher frequency of operation.

Another important feature of HCMOS is that the internal circuitry only dissipates power during switching transitions, which allows the addition of WAIT and STOP instructions. In WAIT mode, the only circuitry left running is the oscillator and timer, reducing power consumption by a significant factor. In STOP mode all circuitry is closed down and the device requires only a very small current in order to retain the RAM contents.
Fig. 1.5 illustrates how the CMOS process has evolved in recent years to produce faster and smaller devices, so that today's HCMOS-1 process can support switching speeds in excess of HMOS. Plans are already in hand to shrink both the HMOS and HCMOS process still further in the near future. The M6804 family will continue to benefit from these improvements in processing technology in order to provide a high performance, cost effective family of MCUs.

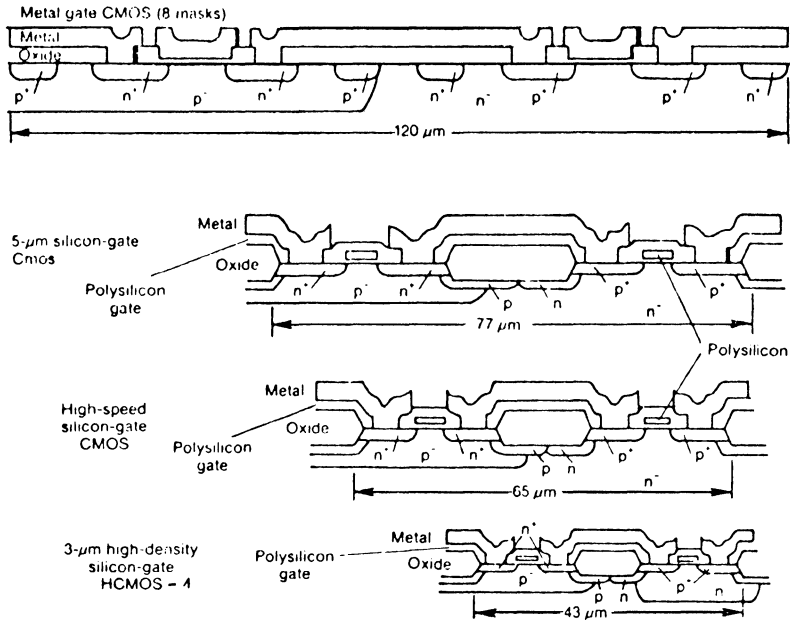


Figure 1.5. Development of CMOS Technology in the Last Decade

1.4 M6805 AND M6804 COMPATIBILITY

Where consistent with the design goal of small silicon size, compatibility with the M6805 has been maintained.

- The M6805 type index register has been replaced with 2 indirect registers (X and Y) which are used to address data space memory. These registers are actually data space RAM and may be manipulated in a manner similar to any memory location in data space. An indirect register is equivalent to an index register where the offset is always zero.
- The M6805 has a stack pointer, which can be read, and an accessible stack register. The M6804 uses a true LIFO stack to store the subroutine addresses. Thus no stack pointer is needed.
- The M6804 does not have a Condition Code register of the M6805 kind. It has only two Condition Code Flags — "Zero" and "Carry". There are two sets of these flags — one for normal operation and the second for interrupt processing. When an interrupt occurs a context switch is made from the program flags to the interrupt flags (interrupt mode). A RTI forces the context switch back to the program flags (program mode). While in either mode, only the flags for that mode are available. Further, the interrupt flags will not be cleared upon entering interrupt mode. Instead, the flags will be as they were at the exit of the last interrupt.
- The M6805 family allows multiple levels of interrupts which are limited only by the available stack depth. The M6804 allows only one interrupt to be queued.
- While different vectors are provided for the timer interrupt, external interrupt, software interrupt in the M6805 family, only one interrupt vector is provided in the M6804 family.
- Bit manipulation instructions, so successful on M6805 machines, are maintained on M6804 as well: BSET, BRSET, BCLR, BRCLR.
- From the Read-Modify-Write instructions the M6804 has INC, DEC, BSET and BCLR.

- The M6804 family introduces Short-Direct type of instructions taking only 1 byte of Program ROM, which are not available on the M6805. Using these instructions one byte can efficiently access frequently used RAM registers ($80, $81, $82, $83). The locations $80, $81, are the X and Y indirect registers respectively.
- Another new type of instruction has been introduced on M6804 family – Move Immediate to Memory – «MVI» instruction. This instruction is not available on the M6805 and allows immediate data to be moved from program ROM to any data space register, without destroying the value in the accumulator.
- There is an incompatibility in the instruction time execution between the M6805 and M6804. Fig. 1.6 shows the operation timing comparison between the M6805 and M6804 families.

OPERATION	MC 6804	MC 6805	MC 68HC04*
BRANCH	8.8 µSec.	4 µSec.	4.4 µSec.
BIT TEST & BRANCH	22 µSec.	10 µSec.	11 µSec.
ADD (Direct)	17,5 µSec.	4 µSec.	8.8 µSec.
MACHINE CYCLE TIME	4.4 µSec.	1 µSec.	2,2 µSec.
XTAL FREQUENCY	11 MHz	4 MHz	11 MHz*

*Assuming ÷ 2 option.

1.5 M6804 AND M68HC04 COMPATIBILITY

Although all members of the M6804 family are basically the same, there are some differences between HMOS and HCMOS versions due to different logic implementations and different technologies. These are summarised below and discussed further in later chapters where appropriate.

- The MC68HC04 features power saving STOP and WAIT instructions, which the MC6804 does not have.
- The MC6804 uses dynamic RAM cells while the MC68HC04 uses fully static low power RAM cells.
- The MC6804 maximum bus frequency is 2.75 MHz, compared with 5.5 MHz for the MC68HC04, i.e. HCMOS is twice as fast.
- The MC6804 oscillator frequency is limited to 4-11 MHz while the MC68HC04 operates from 0 to 11 MHz.
- The MC68HC04 oscillator frequency is divisible by 1,2 or 4 as a photomask option, while the MC6804 only offers divide by 4.
- The MC6804 can only generate a timer interrupt by hardwiring the timer pin to $\overline{IRQ}$ and operating the timer in output mode. The MC68HC04 is capable of generating a (maskable) timer interrupt internally, in either input or output mode. See section 3.9 for further details.
- The MC68HC04 can perform pulse width measurement using the timer pin as an input while the MC6804 cannot. See section 2.14 for further details.
- The MC6804 offers different mask options from the MC68HC04 for the port output drive characteristics. See section 2.10.
- The MC6804 treats latched interrupts in a different way from the MC68HC04. See section 2.15.
- The MC68HC04 features a power-up detect circuit on-chip, while the MC6804 needs an external delay on the $\overline{RESET}$ pin. See section 2.16.
- In single chip and non-user modes the MC68HC04 allows both CRC registers to be used as normal RAM locations while the MC6804 does not. See section 2.18.

1.6 SUMMARY LIST OF FEATURES

The list below summarises the features of M6804 family members available, or in design, at the time of publication. Further members will be added in due course.

FEATURES	MC6804P2	MC6804J2	MC68704P2	MC68HC04P2	MC68HC04P3
TECHNOLOGY	HMOS	HMOS	HMOS	HCMOS	HCMOS
NUMBER OF PINS	28	20	28	28	28
RAM(bytes)	32	32	32	32	124
USER PROGRAM ROM(bytes)	1024	1008	1024 EPROM	1024	1689
USER DATA ROM(bytes)	64	64	64	64	64
BIDIRECTIONAL I/O LINES	20	12	20	20	20
HIGH CURRENT SINK LINES	8	8	8	8	8
OTHER I/O FEATURES	Timer	Timer	Timer	Timer	Timer
EXTERNAL INTERRUPT INPUTS	1	1	1	1	1
STOP and WAIT	No	No	No	Yes	Yes

Hardware Description

2

2.1 ARCHITECTURE

To save silicon area, the M6804 family is implemented as a serial machine. To the user, however, it appears to be an 8-bit parallel processor. Despite serial architecture, the execution speed compares quite favorably with parallel implementation. This is mainly due to the use of the most advanced HMOS and HCMOS technologies which enable high speed operation at low voltages and low power consumption.

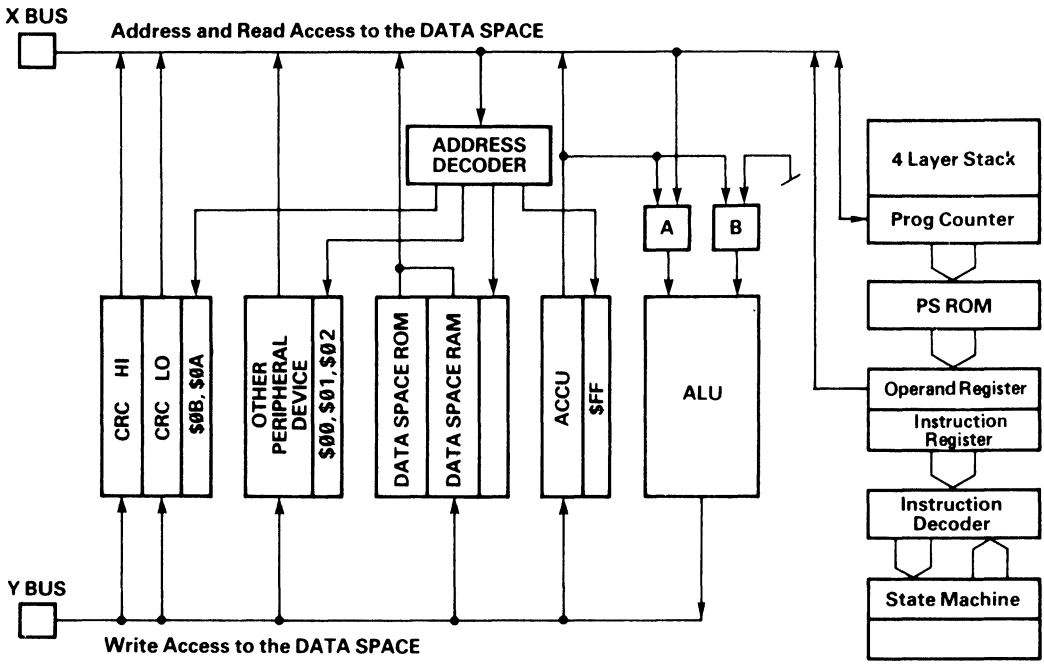


Figure 2.1. Architecture of the M6804 Family of MCUs

Fig. 2.1 depicts the architecture of the M6804 family. It is not of the Von-Neumann type in that it has separately addressed program memory, data memory and stack memory. The data space contains RAM used for program variables, ROM used for constant values or tables, I/O ports, and the timer registers. (Timer function will be discussed in a separate chapter). The program space contains only the executable code and immediate data used by instructions in the immediate addressing mode.

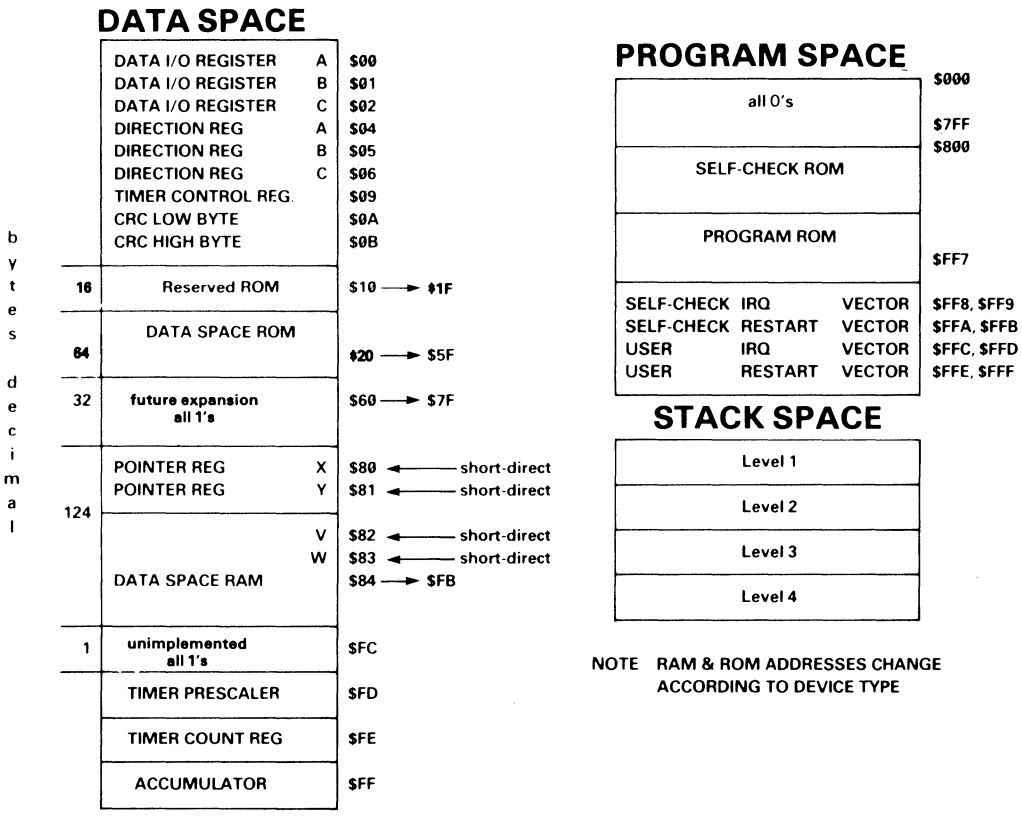


Figure 2.2. MC68HC04P3 Address Map – Other family members are similar except for ROM and RAM values.

The M6804 address map organization is depicted in Fig. 2.2 where the division between the data space, program space and stack space is distinguished.
The serial architecture of the 8-bit serial processor requires only a 1 bit arithmetic logic unit (ALU), with address and data buses reduced to one line connections.
The X index register in the M6800 or M6805 families has been replaced in the M6804 by two registers X and Y which are placed into the data space RAM, with no offset addressing possibility – Indirect Registers. Fig. 2.3 shows the registers available to the programmer.
The M6804 instructions execute in 2,4 or 5 machine cycles. Each machine cycle requires 48 master clock cycles in the HMOS part which is designed to operate with clock frequencies in the 4 to 11 MHz range. The HCMOS part is designed to operate from 0 to 11 MHz clock frequencies, i.e. it is fully static and a machine cycle may take 48, 24 or 12 clocks (see data sheet for confirmation) depending on a mask option. The HMOS design incorporates dynamic RAM as opposed to fully static very low power consumption RAM in the HCMOS design.

2.2 MEMORY

The M6804 family MCUs operate in three different memory spaces: program space, data space, and stack space.

2.2.1 Data Space RAM

The data space RAM consists of 8-bit wide registers whose number depends on the device type, eg. for

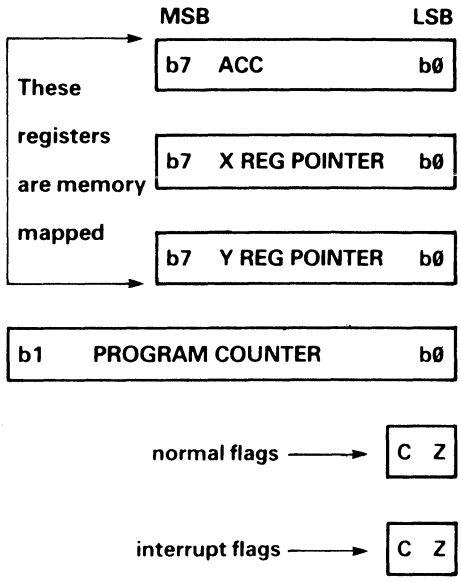


Figure 2.3. Programming Model of the M6804 Family

the MC68HC04P3 there are 124 available at address locations $80 - $FB. For the HMOS versions the RAM is dynamic and for the HCMOS versions the RAM is fully static.

2.2.2 Data Space ROM

The data space ROM can be of variable length, it depends on the device type. This ROM contains constants, addresses, and tables that would be manipulated in data space. A BCD to seven segment decode table would, for example, be located in this ROM.

2.2.3 Program ROM

The program ROM contains user ROM of variable length, e.g. the MC6804P2 has 1024 words of 8 bits each. The ROM address inputs come from the PC. The ROM provides operation codes as well as addresses and operands which may be fixed at assembly time.
The ROM also includes some additional bytes of self-check ROM reserved for Motorola usage.
The user is requested to consult the appropriate data sheets for every M6804 family member in order to know the program ROM size and the self-check ROM size.
At the top of the program ROM space are placed all the vectors necessary for Restart and Interrupt operation. See Fig. 2.2.

2.3 STACK

The M6804 contains a 4-level 12-bit wide stack used to store return addresses during subroutine calls and interrupts. It is implemented in RAM separate from the data RAM, and is not addressable, not readable and not writeable. See Fig. 2.2.
The stack RAM is in stack space which means that the address is inherent; that is, it is implied by calls and returns.
Whenever a subroutine call or interrupt occurs, the contents of the PC are shifted into the top register of the stack. At the same time (same cycle) the top register is shifted to the next level down. This happens to all registers with the bottom register disappearing from the stack. Whenever a subroutine or interrupt

return occurs, the top register is shifted into the PC and all lower registers are shifted up one level. Thus it operates as a true LIFO stack.
The values of the accumulator and X,Y registers are not stored on the stack, and must be treated by software, in the subroutine or interrupt routine.

2.4 CENTRAL PROCESSING UNIT

The CPU of the M6804 family is implemented independently from the I/O or memory configuration. Consequently, it can be treated as an independent central processor communicating with I/O and memory via internal addresses, data, and control buses. See Fig. 1.3.

2.5 ARITHMETIC LOGIC UNIT

The Arithmetic Logic Unit (ALU) is a one bit logic unit allowing two inputs to be ADDED, SUBTRACTED, or ANDED. The inputs (A,B) have connections to data space locations, immediate operands, and the accumulator. Outputs from the ALU may be routed to the data space locations or accumulator.

2.6 ACCUMULATOR

The accumulator is an 8-bit general purpose register used in all arithmetic calculations, logical operations, and data manipulations. The accumulator is implemented as the highest RAM location ($FF) in data space and this implies that several instructions exist which are not explicitly implemented. The accumulator can be treated in the same way as any data space register. This is a novelty not available on the M6805 family.

2.7 X AND Y INDIRECT REGISTERS

The X and Y Indirect Registers are in data space RAM locations with addresses $80 and $81, and may be manipulated in a manner similar to any memory location in data space. An indirect register is equivalent to an index register whose offset is always zero. They are used in the Register Indirect addressing mode, and can be accessed with the Direct, Indirect, Short Direct, or Bit-Set/Clear addressing modes.
X and Y Indirect Registers are used as pointers to other memory locations in data space (e.g. for data conversion tables).

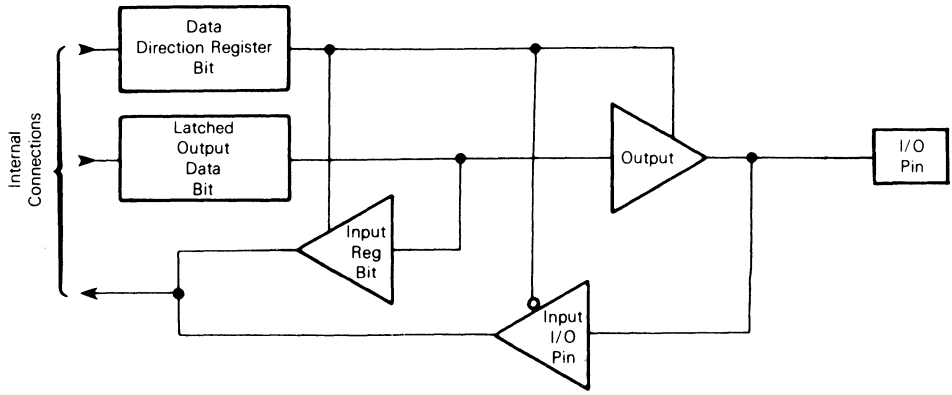
2.8 CONDITION CODES, FLAGS

Condition code indicators are provided to indicate zero and carry results of an operation.
The Carry (C) bit is set on a carry or a borrow out of the ALU. It is cleared if the result of an arithmetic operation does not result in a carry or a borrow. The (C) bit is also set to the value of the bit tested in a bit test instruction, and participates in the rotate left instruction.
The Zero (Z) bit is set if the result of the last arithmetic or logical operation was equal to zero, otherwise it is cleared.
There are two sets of these condition codes, one for interrupt mode and the other for normal operation. See Fig. 2.3. When an interrupt occurs while the machine is in normal program execution mode, a context switch of the condition code flags is made to the interrupt set of condition codes which will be used by any instructions which test or affect condition codes, until a RTI (Return from Interrupt) is executed. The normal mode condition code flags will be left as they were before the interrupt occured. Every time a new interrupt is taken, the interrupt mode condition codes will be as they were at the end of the last interrupt.

2.9 PROGRAM COUNTER

The Program Counter is a 12-bit register that contains the address of the next ROM word to be fetched (may be opcode, operand, or address of operand).
The contents of the PC are gated out in parallel to the ROM address inputs. The PC may be changed in the following ways:

a. To increment the PC its output is shifted through a 1-bit adder and back to the PC input.
b. For an RTS or RTI operation, the contents of the top level of the stack RAM are shifted into the PC.
c. For a JUMP or JUMP to SUBROUTINE operation, the jump address is shifted into the PC.
d. For a BRANCH operation, the Branch Offset is shifted into the PC via a 1-bit adder. This adds the offset to the PC. The sign extension of the offset is likewise added to the PC.
e. Upon $\overline{\text{RESET}}$ or $\overline{\text{INTERRUPT}}$ the address of the corresponding vector is loaded into the PC. Fig.2.2 shows the address locations. This vector must contain a JMP instruction so that the PC is next loaded with the starting address of an appropriate service routine.



Data Direction Register Bit	Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	Hi-Z	Pin

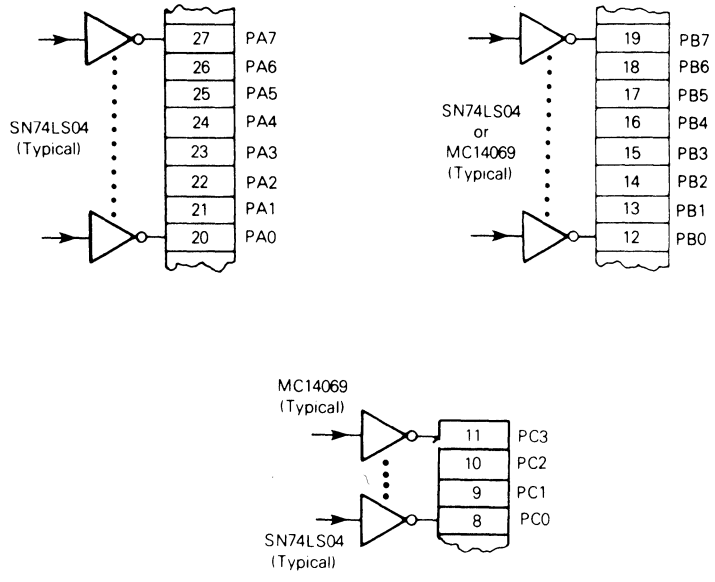
Figure 2.4. Typical I/O Port Circuitry. The Addresses of Data Direction Registers and I/O Ports are given in the relevant Address Map

2.10 I/O PORTS

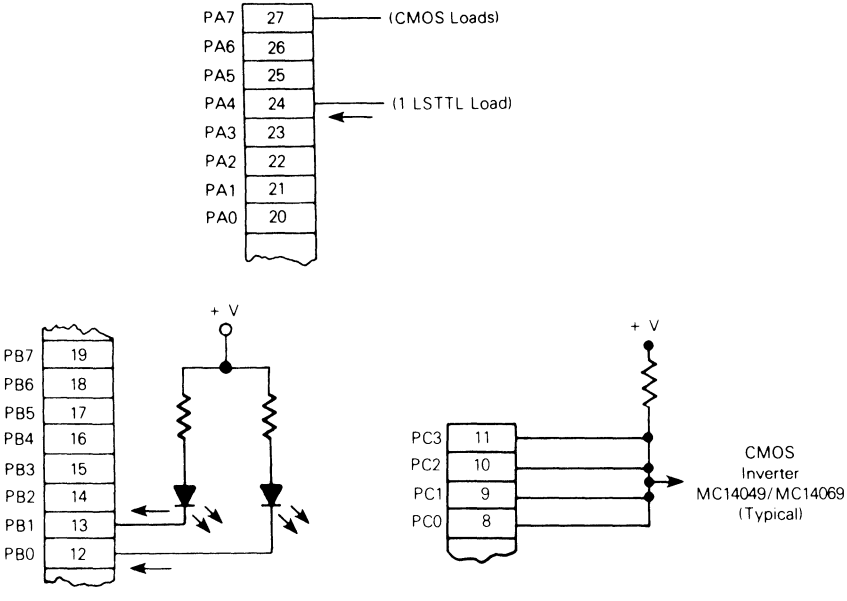
The M6804 MCU contains memory mapped I/O ports. They consist of separately addressable Data Registers and Data Direction Registers (DDR). All pins are programmable as either inputs or outputs under software control of the corresponding Data Direction Register. Typical I/O port circuitry is shown in Fig. 2.4. The port I/O programming is accomplished by writing the corresponding bit in the DDR as a logic one for output or a logic zero state for input. On reset, all the DDRs are initialized to a logic zero state to put the ports in the input mode. The port output registers are not initialized on reset but should be initialized before changing the DDR bits to avoid undefined levels.
In HMOS devices all I/O pins are LS TTL compatible as both inputs and outputs. In addition, one of two mask options may be selected for each port:

1. Internal pull up resistor for CMOS output compatibility.
2. Open Drain Output.

HCMOS devices are LS TTL compatible as inputs, provided V_{cc} = 5V ± 10%. Under this condition they will also drive two LS TTL loads in output mode. This is in addition to being fully CMOS compatible. The only mask option available is a pull down device, which can be selected on any number of the I/O pins, and is particularly useful for keyboard encoding applications.
Fig. 2.5 shows typical port connections for the MC6804P2 MCU in the Input Mode or Output Mode.



a) Input Mode



b) Output

Figure 2.5. Typical Port Connections for the MC6804 P2 MCU

2.11 BUSES ON M6804 MCU

From Fig. 2.1, it is apparent that there are two buses on the M6804 MCU, the X-Bus and Y-Bus. These two serial buses fulfill the following functions.
The X-Bus serves for transfer of address and for the read access to the data space; while the Y-Bus serves only for the write access to the data space.

2.12 INSTRUCTION REGISTER

The Instruction Register is a set of latches connected between the ROM output and the Instruction Decoder input.
These latches hold the operation code for the current instruction. Loading the Instruction Register is thus an instruction «fetch».

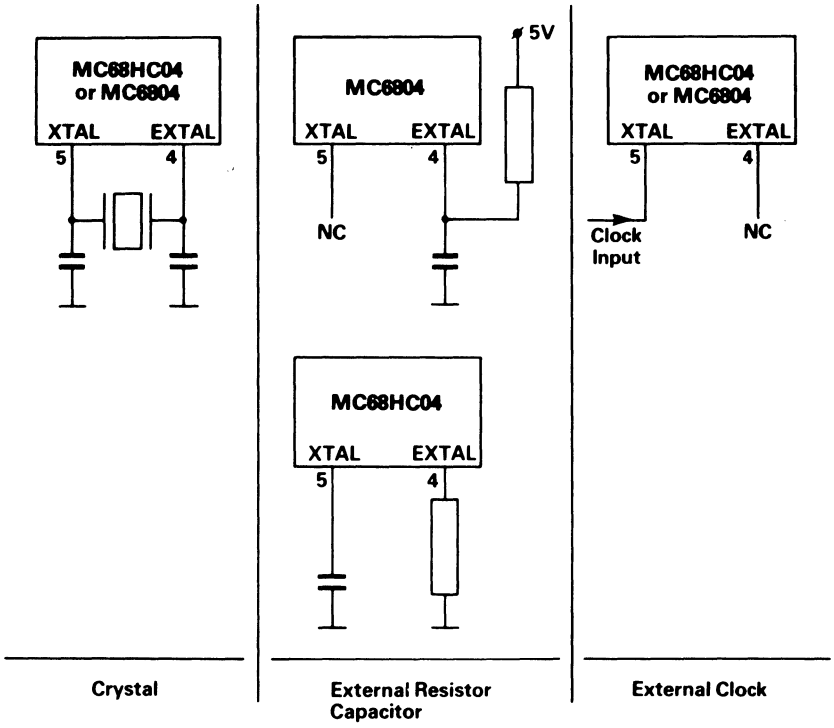


Figure 2.6. Clock Generator Options of the M6804 Family of MCU's. All options are mask selectable. Note the difference between the External Resistor - Capacitor options for the HMOS and HCMOS versions.

2.13 CLOCK GENERATOR OPTIONS, TIMING

All microprocessors require a generator which generates the internal clock frequency. The M6804 family has been designed to use either a crystal oscillator; an external resistor and capacitor; or an external clock. All these possibilities are depicted in Fig. 2.6. Clock generator options are mask selectable. There is a difference in use of the external resistor capacitor option between the HMOS and HCMOS versions.
Whichever clock generator is used the oscillator frequency is internally divided to produce the internal bus cycle clocks, Ø1 and Ø2. See Fig. 2.7. In the case of HMOS devices, the oscillator is divided by 4, whilst for HCMOS it can be divided by 4,2 or 1 depending on a photomask option. The divisions by 2 and 1 are available on HCMOS in order to save power when higher bus frequencies are required. This is an

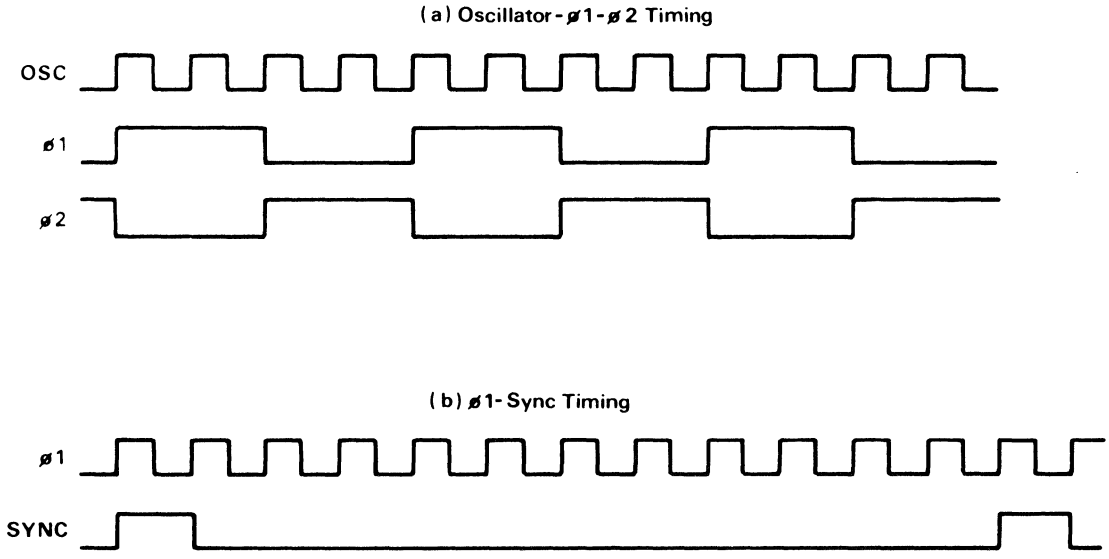


Figure 2.7. Clock Generator Timing Diagram

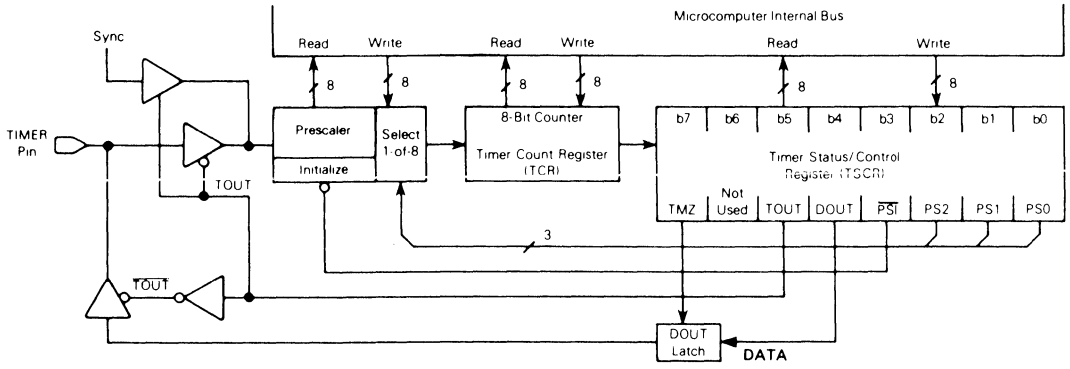
important benefit in low power applications. Note that it is bus frequency rather than oscillator frequency which limits the performance of HCMOS devices — it must not exceed 5.5 MHz, whatever the oscillator frequency.

The Ø1 clock is further divided by 12 to produce the machine cycle clock in both HMOS and HCMOS versions. A machine cycle is the minimum time needed to execute any operation, i.e. increment the 12-bit program counter. Instructions require either two, four or five machine cycles to execute depending on their type. This is shown in Fig. 2.8.

INSTRUCTION TIMING

Branch if bit set or clear	5 cycles
Short-direct instructions Direct, Indirect, Immediate instructions Move immediate to memory Jump and Jump to subroutine Roll accumulator Complement accumulator, Bit set, Bit clear	4 cycles
Short branch Return from interrupt Stop Wait	2 cycles

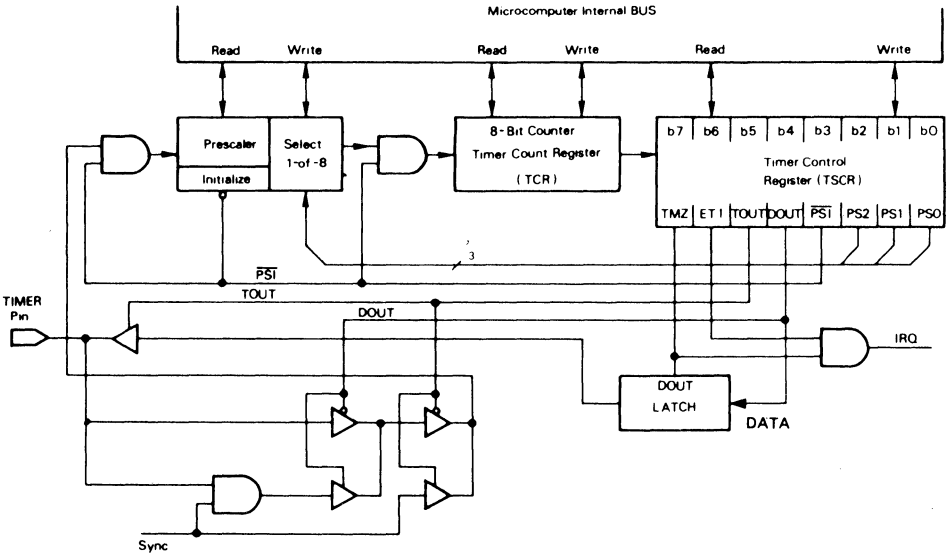
Figure 2.8. M6804 Instructions and their Execution Times measured in Machine Cycles



TIMER Pin Status

TOUT	Prescaler Clock	TIMER Pin
0	TIMER Pin	Input Mode
1	Sync	Output Mode

(a) HMOS Timer



TIMER STATUS			
TIMER PIN	TOUT	DOUT	TIMER MODE
INPUT	0	0	EVENT COUNTER
	0	1	INPUT GATED MODE
OUTPUT	1	0	OUTPUT
	1	1	

(b) HCMOS Timer

Figure 2.9. M6804 Timer Block Diagrams. Note that HMOS Devices differ from HCMOS

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TMZ	ETI*	Tout	Dout	$\overline{PSI}$	PS2	PS1	PS0

*HCMOS only

Figure 2.10. Timer Status/ Control Register (TSCR)

2.14 TIMER

The timer section contains an 8-bit count register; 7-bit software programmable prescaler register; and a status/control register. See Fig. 2.9 and 2.10. Note that the timer logic is slightly different for the HMOS and HCMOS family members.

All the registers are placed in Data Space RAM under the following locations:

Timer Count Register (TCR) – $FE
Timer Prescaler – $FD
Timer status/ control register (TSCR) – $09

The TCR, which may be loaded under program control, is decremented towards zero by a clock input (prescaler output). The prescaler is used to extend the maximum interval of the overall timer. The prescaler tap is selected by bits PS0-PS2 of the TSCR which control the actual division of the prescaler within the range given by a 2^n scale factor, where «n» is a value from 0 through 7, see Fig. 2.11.

PS2	PS1	PS0	
0	0	0	Divide by 1
0	0	1	Divide by 2
0	1	0	Divide by 4
0	1	1	Divide by 8
1	0	0	Divide by 16
1	0	1	Divide by 32
1	1	0	Divide by 64
1	1	1	Divide by 128

Figure 2.11. Timer Prescaler coding of the M6804 Timer.

The TCR and prescaler are decremented on rising clock edges.

The timer pin can be selected as either an input or an output by clearing or setting the Tout bit of the TSCR. In the output mode, the content of the timer data output control bit (Dout) is copied to the timer pin each time the count register is decremented to zero. The internal clock frequency is used to decrement the prescaler. In the input mode, the timer pin is used as a clock input to decrement the prescaler. The frequency of the external clock applied to the timer pin must be less than the machine cycle time (f_{osc}/ 48 for HMOS).

In HCMOS devices a second input mode exists in which the internal clock frequency is used to decrement the prescaler, but is gated by the timer pin so that counting is enabled only when the timer pin is high. This "input gated" mode is selected by setting Dout (bit 4) of the TSCR to one, and can be useful for pulse width measurement. It is not available on HMOS devices.

In either the input or output mode, a status bit (TMZ) will be set in the TSCR indicating that the count register has decremented to zero. This bit will remain set until the TSCR is read under program control. The prescaler/timer can be inhibited from counting by clearing the Prescaler Initialize bit (PSI) in the TSCR. This also sets the prescaler register to all 1's.

TMZ is normally set to logic one when the timer times out (TCR decrements to $00). However, it may be set at any time, by writing $00 to the TCR or by setting bit 7 of the TSCR. During Reset, the TCR and prescaler are set to $FF, while the TSCR is cleared to $00 and the Dout latch is forced to a logic high.

The prescaler and TCR are implemented in data space RAM locations ($FD, $FE); therefore, they are both readable, and writable. A write to either will dominate over the TCR decrement-to-$00 function; i.e. if a write and TCR decrement-to-$00 occur simultaneously, the write will take precedence, and the TMZ bit is not set until the next timer time out.

Care must also be used in reading or writing the TSCR to assure that a time out does not occur as the instruction executes, thus risking the loss of the TMZ state. If TMZ attempts to go high (timer times out) as the TSCR is read or written, the previous low state of TMZ is restored at the completion of the instruction.

2.15 INTERRUPT

The M6804 family of MCUs can be interrupted by applying a logic low signal to the $\overline{IRQ}$ pin; a mask option selected at the time of manufacture determines whether the negative-going edge or the actual low level is sensed to indicate an interrupt. See Fig. 2.12.

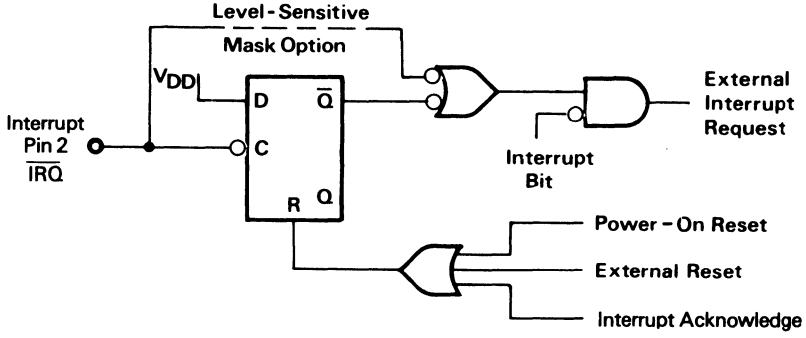


Figure 2.12. Interrupt Configuration

Please note the following difference between HMOS and HCMOS family members:

- HMOS devices are interrupted only by applying a logic low signal to the $\overline{IRQ}$ pin.
- HCMOS devices are interrupted either by applying a logic low signal to the $\overline{IRQ}$ pin, or by a timer underflow if the ETI bit in the timer status/control register is set (Fig. 2.9).

2.15.1 Edge-Sensitive Option

When $\overline{IRQ}$ is pulled low, the internal interrupt request flip-flop is set. Prior to each instruction fetch, the interrupt request flip-flop is tested and, if its output is low, an interrupt sequence is initiated at the end of the current instruction (provided the interrupt mask is cleared). Fig. 2.13 contains a flow chart which illustrates both the Reset and Interrupt sequences.

The interrupt sequence consists of one cycle during which the interrupt mode flags are selected, the PC is saved on the stack, the interrupt mask is set, the $\overline{IRQ}$ vector (single chip or non-user mode = $FFC/$FFD, self-check mode = $FF8/$FF9) is loaded into the PC, and finally the interrupt request latch is cleared.

Unlike the M6805 the vector contents of the M6804 are not inherently decoded as an address to which program control (the PC) should jump, but instead are decoded like any other ROM word. It is therefore

essential that the M6804 vector specifies a JMP instruction to the starting address of the interrupt service routine. This is illustrated in the programming examples in section 6.3.
Note that if it is required to save the values of the accumulator and X,Y registers this should be done by the interrupt service routine, since they will not be stored on the stack.
Internal processing of the interrupt continues until the RTI instruction is processed. During the RTI instruction, the interrupt mask is cleared and the program mode flags are selected. The next instruction of the program is then fetched and executed.
Once an interrupt has been detected and the interrupt sequence started, the interrupt request latch is cleared so that a second interrupt may be detected even while the previous (first) one is being serviced. However, even though the second interrupt sets the interrupt request latch, it will not be serviced until completion of the first interrupt service routine.
At this point there is another difference between HMOS and HCMOS family members:

- HMOS devices will return control to the main program and execute one instruction in normal mode before servicing the latched interrupt.
- HCMOS devices will service the latched interrupt immediately on completion of the first interrupt service routine.

Completion of an interrupt service routine is always accomplished using an RTI instruction to return to the main program. The interrupt mask, which is not directly available to the programmer is cleared during the last cycle of the RTI instruction.
Maximum interrupt response time is six machine (t_{byte}) cycles. This includes five machine cycles for the longest instruction to execute, plus one machine cycle for stacking the PC and switching flags from the Normal Operation flags to Interrupt flags. Minimum response time is one machine cycle for stacking the PC and switching flags.

2.15.2 Level-Sensitive Option

The actual operation of the level-sensitive and edge-sensitive options are similar except that the level-sensitive option does not have an interrupt request latch. With no interrupt request latch, the logic level of the $\overline{IRQ}$ pin is checked for detection of the interrupt. Unlike the edge-sensitive option there is no way of latching an additional interrupt while a current interrupt is being serviced. Also in the interrupt sequence, there is no need to clear the interrupt request latch. These differences are illustrated in Fig. 2.12 and Fig. 2.13.

2.16 RESETS

The M6804 family has two reset modes:

- Power-up reset function.
- Active low external reset pin ($\overline{RESET}$).

At power-up, a delay is needed to allow the clock generator to stabilise before starting normal operation. On HCMOS devices a power-up detect circuit is provided on-chip, and the timer is used to control the delay required for the oscillator to stabilise. On HMOS devices this delay must be provided externally. Connecting a capacitor and resistor to the $\overline{RESET}$ input typically provides sufficient delay. See Fig. 2.14. There is a Schmitt trigger at the pin to improve its noise immunity.
During a reset cycle the interrupt mask is set to prevent any false or ghost interrupts occuring, and the PC is loaded with the appropriate restart vector (single chip or non-user mode = $FFE/$FFF, self check mode = $FFA/$FFB).
As with the $\overline{IRQ}$ vector, it must contain a JMP instruction to a ROM address, which in this case should specify the start of the initialization routine. It is essential that the interrupt mask is cleared at the end of the routine, which requires an RTI (not RTS) as the last instruction. This means the routine must start with a JSR in order to push a suitable PC value onto the stack for subsequent use by the RTI instruction when returning to the main program. This is illustrated in Fig. 2.15 and later programming examples (section 6.3).
The $\overline{RESET}$ input pin is used to reset the MCU to provide an orderly software start-up procedure. When using the external reset mode, the pin must stay low for a minimum of 2 machines cycles after the oscil-

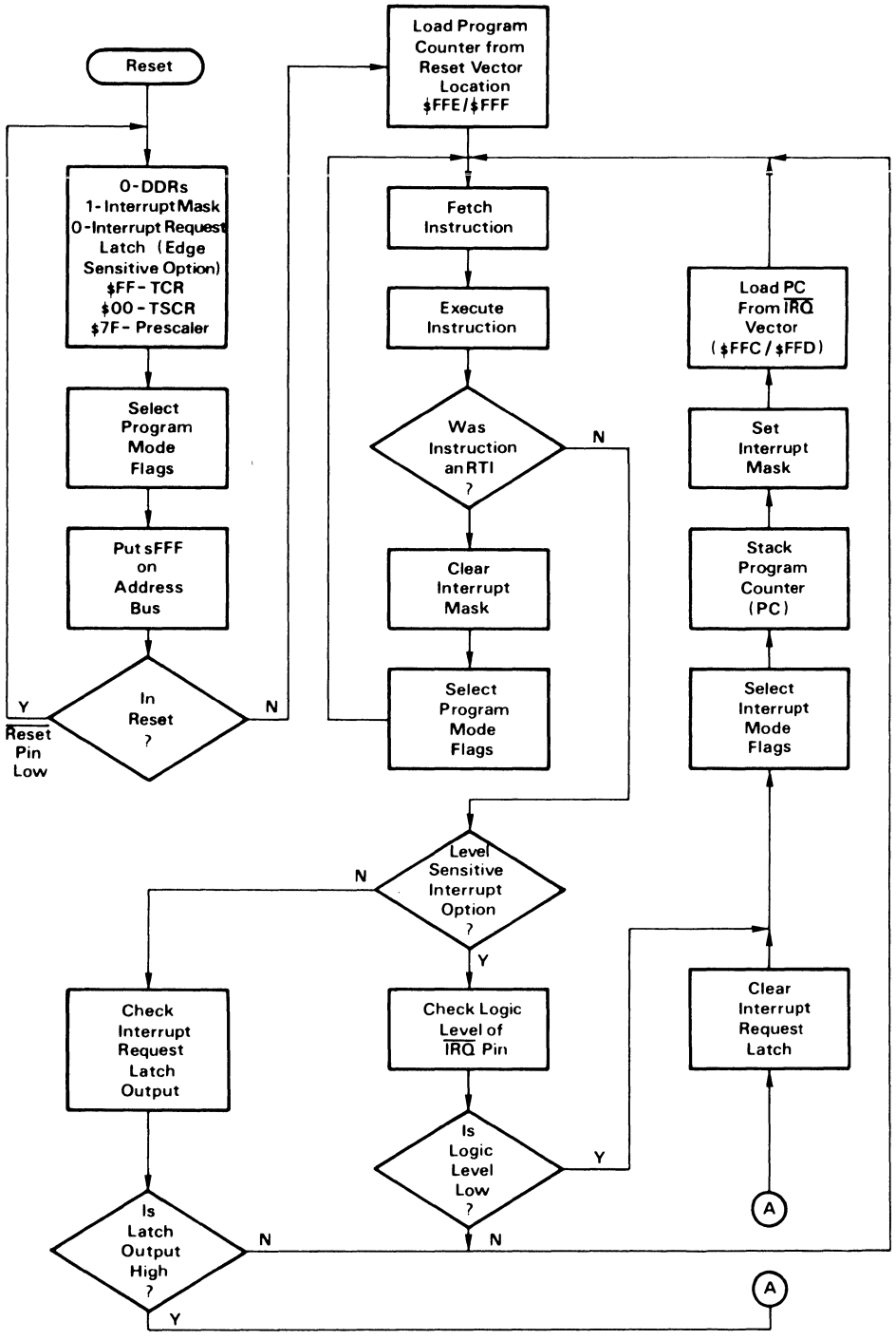


Figure 2.13. Reset and Interrupt Processing Flowchart for the MC6804 (Refer to data sheet for the MC68HC04)

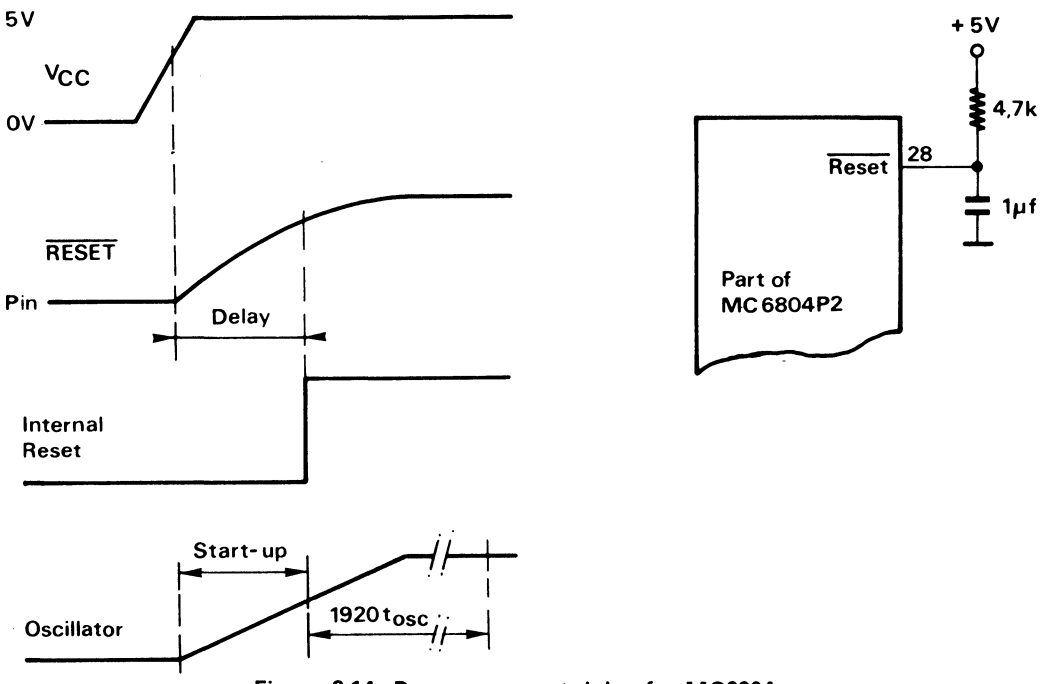


Figure 2.14. Power-on reset delay for MC6804
(MC68HC04 has on-chip power-up detect circuit)

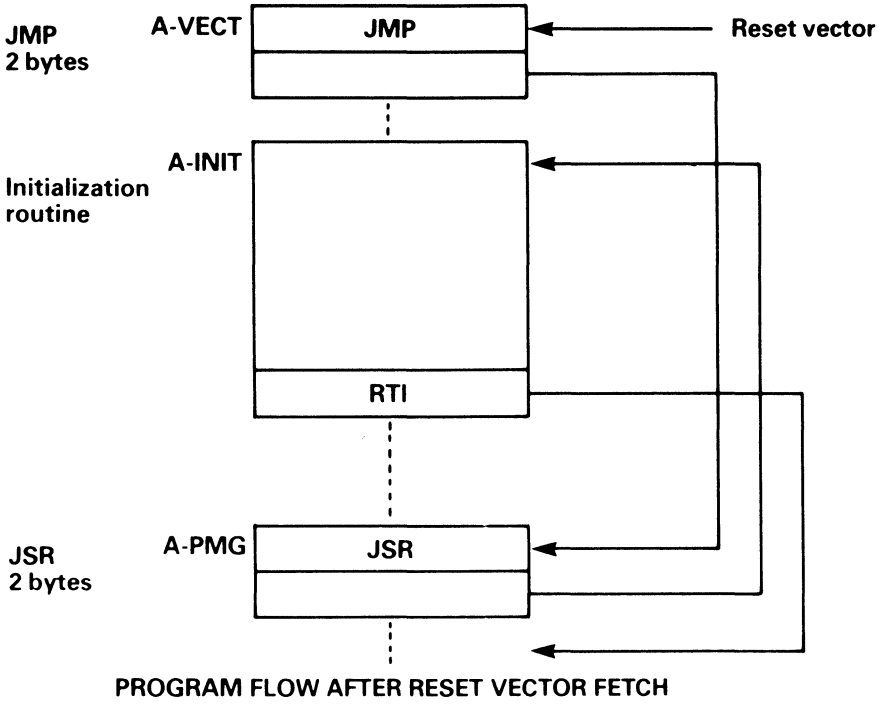


Figure 2.15. Flow Chart of the Initialization of Program after Reset or Power – up

lator has stabilized. In this mode, the MCU executes the software in the same manner as in the power-up mode.

2.17 MODES OF OPERATION

Four modes of operation are provided in the M6804 family: Single Chip, Self Check, Non User, and ROM Verify. The appropriate mode of operation is selected according to the voltages applied to the pins MDS and pins PA6, PA7 at the moment of exit from the reset state.
If at the exit from reset, MDS pin is held high (V_{DD},) the states of pins PA6 and PA7 are latched and decoded to determine the chip operating mode.

The coding is:

PA6	PA7	MODE
0	0	SINGLE CHIP
0	1	SELF CHECK
1	0	NON USER
1	1	ROM VERIFY

The normal operating mode is the Single Chip mode in which only the resources on the chip are utilized. This is the mode to be selected for end user applications. If MDS is grounded at the exit from reset (this is the default mode), the Single Chip mode is selected.
When the Non User mode is selected, Ports A, B and C are used to interface with an external instruction source (ROM, EPROM, RAM, etc.). See the appropriate data sheets for further details.
The Self-Check mode and ROM Verify mode were incorporated into the M6804 operating modes in order to facilitate test procedures. The testing of MCU devices is a very complex and expensive process. Saving money on testing can reduce the final price of the device considerably. This was very carefully examined during the design of the M6804 family and the following test capabilities were included on the chip:

Self-Check Mode A powerful self test routine is used to test all of the MCU except the program ROM.

ROM Verify ModeThis mode checks, by means of signature analysis, the contents of the program ROM.

These test modes are treated extensively in a separate chapter.

2.18 CRC-CYCLIC REDUNDANCY CHECK CIRCUIT

To facilitate testing, a signature analysis circuit has been included on the chip. This circuit consists of two 8-bit shift registers configured to perform a Cyclic Redundancy Check using the CCITT polynominal. (see chapter 5).
These two registers are memory mapped in the data space at addresses $0A and $0B.
In HCMOS devices only they may be used as normal RAM locations by performing a write operation to the above addresses when in Single Chip or Non-User mode. Their subsequent use for CRC operation will only be possible after a RESET operation.

Software Description

3.1 INTRODUCTION

Microprocessors accomplish a task by using software to define the operations of the system at the programming stage, rather than by using digital logic to construct a system which has its operations defined at the design stage.

During the early 1970's microprocessors and microcomputers helped ease the shortage of hardware designers by providing the hardware with more intelligence, reducing hardware development costs. However, rising software development costs in recent years make it important for the system designer of today to carefully weigh the software and support cost of his system.

Processors of the M6804 family, which are designed to include the programming features inherited from microcomputers like the M6805, M6800, require less effort from the programmer and make system design much more efficient.

3.2 M6804 PROGRAMMING MODEL

Programming registers available to the user are shown in Fig. 2.3. They comprise an 8-bit accumulator (ACC), two 8-bit pointer registers (X and Y), a 12-bit program counter (PC), and two sets of flags. The ACC and X,Y registers are memory mapped in data space and allow all instructions which manipulate memory to access them in any of the memory addressing modes. Unlike the M6805 family there is no stack pointer required in programming the M6804. Further details of each register are given below.

3.2.1 Accumulator

The accumulator is an 8-bit general purpose register that is used by the programmer for arithmetic calculation and data manipulation. A special feature of the M6804 architecture is that it is implemented in data space at location $FF and may be addressed as such. This gives rise to several assembler-recognised instructions which are not explicitly implemented, e.g. ASLA (arithmetic shift left of accumulator) is converted by the assembler to ADD $FF (add contents of accumulator to itself).

Listed below are some examples of instructions which operate on the accumulator.

```
0D00 F8 84      A       LDA     $84     LOAD ACCUMULATOR WITH CONTENTS
                *                       MEMORY LOCATION  $84
0D02 FA 87      A       ADD     $87     ADD THE CONTENTS OF MEMORY LOCATION
                *                       $87 TO THE ACCUMULATOR
0D04 F9 84      A       STA     $84     STORE THE ACCUMULATOR CONTANTS IN
                *                       MEMORY LOCATION $84
0D06 FA FF      A       ASLA            ARITHMETIC SHIFT LEF OF ACCUMULATOR
                *                       INSTRUCTION WITHOUT OPCODE
0D08 FA FF      A       ADD     $FF     THIS IS MAXHINE CODE FOR THE PREVIUOS
                *                       INSTROCTION
```

3.2.2 Indirect Registers X and Y

The two Indirect Registers (X and Y) are used to maintain pointers to other locations in data space. They are used in the register-indirect addressing mode, and can be accessed with the direct, indirect, short direct, or bit set/clear addressing modes. They are implemented in data space RAM at locations $80, $81, giving rise to further assembler-recognised instructions.

The following example shows a typical use of the index registers.
The example performs a block move of length NLENGH. X is pointing to the origin, Y to the destination. NLENCO is a counter for the length of block in the RAM data space.

```
0CA0 B0 0A 85     A         MVI    #NLENGH,NLENCO STORE THE LENGTH OF THE BLOCK
                    *                          IN THE COUNTER

0CA3 F8 87        A         LDA    #BLOCPO  LOAD POINTER TO THE ORIGIN
                    *                         BLOCK

0CA5 BC                     STA    RX       STORE IT IN THE X REGISTER

0CA6 F8 86        A         LDA    #DESTPO  LOAD POINTER TO THE DESTINATION
                    *                         BLOCK

0CA8 BD                     STA    RY       STORE IT IN THE Y REGISTER

0CA9 E0             CONTIN  LDA    [X]      LOAD VALUE POINTED TO BY THE
                    *                         X REGISTER
0CAA A8                     INCX             INCREMENT X REGISTER

0CAB F1                     STA    [Y]      STORE VALUE IN THE DESTINATION
                    *                         BLOCK

0CAC A9                     INCY             INCREMENT Y REGISTER

0CAD FF 85        A         DEC    NLENCO   DECREMENT COUNTER FOR THE BLOCK
                    *                         LENGTH

0CAF 19         0CA9        BNE    CONTIN   BRANCH IF BLOCK UNFINISHED

0CB0 B3                     RTS              OTHERWISE EXECUTION
                  **                      FINISHED
                  *
```

3.2.3 Program Counter

The Program Counter is a 12 bit wide register that contains the address of the next ROM word to be used (may be opcode, operand, or address of operand).

3.2.4 Flags

The Carry (C) bit is set on a carry or borrow out of the ALU. It is cleared if the result of an arithmetic operation does not result in a carry or borrow. The C bit is also set to the value of the bit tested in a bit test instruction, and participates in the rotate left instruction (ROL).
The Zero (Z) bit is set if the result of the last arithmetic or logical operation was equal to zero, otherwise it is cleared.
The effects of different instructions on the flags are shown in Table 3.1.

3.2.5 Stack

There is a true LIFO stack incorporated in the M6804 which eliminates the need for a stack pointer. Stack space is implemented in separate RAM (12-bits wide). Whenever a subroutine call or interrupt occurs, the contents of the PC are shifted into the top register of the stack. At the same time (same cycle) the top register is shifted to the next level deeper. Whenever a subroutine or interrupt return occurs, the top register is shifted into the PC and all lower registers are shifted up, one level higher. The stack is 4 layers deep.There is no stack pointer available and the stack is not readable or writeable. If the stack is pushed more than 4 times consecutively, the information in the bottom register will be lost.

Immediate	**2 Bytes**	**LDA #$AA**
Short Direct	**1 Byte**	**LDAX**
Direct	**2 Bytes**	**LDA $90**
Bit-Direct	**2 Bytes**	**BSET 0, PORT A**
Bit-Test-Direct	**3 Bytes**	**BRCLR 7, PORT B, 0FSX**
Register Indirect X & Y Registers	**1 Byte**	**AND X**
Absolute Jumps (extended)	**2 Bytes**	**JMP $EFF**
Relative Short		
Branch (±16)	**1 Byte**	**BNE NEXT**
Branch (±128)	**3 Bytes**	**BRSET 0, $SAD, 0FSY**
Accumulator	**1 Byte**	**COM**
Inherent	**1 Byte**	**RTS, NOP**

Figure 3.1. M6804 Addressing Modes, summary and examples

3.3 ADDRESSING MODES

The addressing modes provide different ways for instructions to access memory. The M6804 family has a set of powerful flexible addressing modes, especially when compared with other processors in its price range.

The M6804 addresses memory in three different address spaces:

Program Space
Data Space
Stack Space

Program space contains the instructions which are to be executed, plus the data for immediate mode instructions.

Data space contains all of the RAM locations, X and Y registers, accumulator, timer, I/O locations, CRC registers, and ROM (for storage of tables and constants).

Stack space contains RAM for use in stacking the return addresses for subroutines and interrupts.

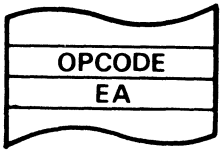
Instructions take between 1 and 3 bytes of storage, depending on the addressing mode. This is shown in Fig. 3.1 which summarises all the addressing modes available on the M6804. Each of these modes is now discussed in detail.

In the following descriptions, the term "effective address" (EA) is used. The EA is the address in memory from which the argument for an instruction is fetched or stored.

3.3.1 Immediate Addressing Mode – 2 Bytes

In the immediate addressing mode, the operand is located in program ROM and is contained in a byte following the opcode. The immediate addressing mode is used to access constants which do not change during program execution – constants which were known at assembly time. e.g. constants used to initialize a loop counter.

An immediate instruction is two bytes long.



PC + 1→ PC
EA = PC
PC + 1→ PC

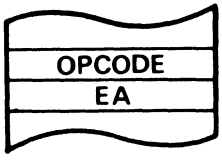
Assembly examples.

```
00043A 0C00 E8 03      A      LDA    #$03
00045A 0C02 EA 08      A      ADD    #$08
00047A 0C04 EB 05      A      SUB    #$05
00049A 0C06 EC F1      A      CMP    #$F1
00051A 0C08 ED FF      A      AND    #$FF
```

3

3.3.2 Direct Addressing Mode – 2 Bytes

In the direct addressing mode, the effective address of the operand is contained in a single byte following the opcode byte. The byte addressed is in data space. Direct addressing allows the user to directly address the 256 bytes in data space memory with a single two byte instruction.



PC + 1→ PC
EA = (PC) + $000
PC + 1→ PC

Assembly examples.

```
0CF2 F8 50   A   LDA   $50    LOAD VALUE FROM THE MEMORY LOCATION $50
0CF4 F9 84   A   STA   RAM1   STORE IT INTO THE LOCATION RAM1
0CF6 FA 85   A   ADD   RAM2   ADD TO ACCUMULATOR THE VALUE FROM THE LOCATION RAM2
0CF8 FB 70   A   SUB   $70    SUBSTRACT $70 FROM ACCUMULATOR
0CFA FC 86   A   CMP   RAM3   COMPARE ACUUMULATOR WITH THE VALUE IN THE RAM3
0CFC FE 84   A   INC   RAM1   INCREMENT LOCATION RAM1
```

3.3.3 Short Direct Addressing Mode – 1 Byte

The M6804 has four locations in data space RAM ($80, $81, $82, $83) which may be used for short direct addressing.
Instructions in this mode of addressing are 1 byte long since the opcode itself determines which of the four locations contains the operand. Short direct addressing is a subset of the direct addressing mode. Note that it can be used to alter the contents of the X and Y registers, which are at locations $80 and $81 respectively.

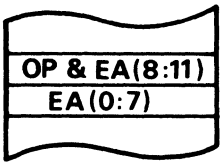
EA CONTAINED IN OPCODE ($080, $081, $082, $083)
PC + 1→ PC

Assembly examples.

```
0CD2 AC        LDA    RX      LOAD ACCUMULATOR WITH THE VALUE IN THE X REGISTER
0CD3 BD        STA    RY      STORE IT INTO THE Y REGISTER
0CD4 A8        INCX           INREMENT X REGISTER
0CD5 A9        INCY           INCREMENT Y REGISTER
0CD6 AA        INC    RV      INCREMENT SHORT DIRECT ADDRESABLE REGISTER - $82
0CD7 AB        INC    RW      INCREMENT SHORT DIRECT ADDRESABLE REGISTER - $83
```


3.3.4 Extended Addressing Mode – 2 Bytes

In the extended addressing mode, the effective address is obtained by concatenating the four least significant bits of the opcode with the byte following the opcode (twelve bit address). Instructions using the extended addressing mode (JMP, JSR) are capable of branching anywhere in program space. An extended addressing mode instruction is two bytes long.



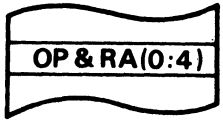
EA (8: 11) = (PC(0:3))
PC + 1 → PC
EA (0:7) = (PC)
PC + 1 → PC

Assembly examples.

```
0CF0 EC F0       A          CMP    #$F0     COMPARE ACCUMULATOR WITH THE VALUE $F0
0CF2 02       0CF5          BNE    SUBR1
0CF3 9C F9       A          JMP    SUBR2
0CF5 8C F7       A SUBR1    JSR    ROUT1
```

3.3.5 Relative Addressing Mode – 1 Byte

The relative addressing mode is only used in conditional branch instructions. In relative addressing the EA for a branch is formed by adding the sign extended lower five bits of the opcode (the offset) to the program counter if and only if the condition is true. Otherwise, control proceeds to the next instruction. The space of relative address is from – 15 to + 16 from the opcode address.



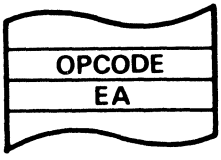
(PC(0:4)) → TEMP
PC + 1 → PC
EA = PC + TEMP IF BRANCH IS TAKEN

Assembly examples.

```
0CA0 B0 B1 E0     A         LDYI    #$E0

0CA3 A8             L10     INX

0CA4 1E        0CA3         BNE     L10       VALUE $1E CORRESPONDS TO - 2 ,AND WILL
                  *                          BE SUBSTRACTED FROM THE ACTUAL PC VALUE
0CA5 A9                     INY

0CA6 1C        0CA3         BNE     L10

0CA7 B3                     RTS


0CA8 F0                     LDA     [Y]

0CA9 E4                     CMP     [X]       VALUE $09 WILL BE ADDED TO THE ACTUAL
                  *                          PC VALUE
0CAA 00        0CAB         BNE     SNDERR


0CAB 8C F7        A SNDERR  JSR     ROUT1
```

3.3.6 Bit Set/Clear Addressing Mode – 2 Bytes

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct address of the byte in which the specified bit is to be set or cleared. Thus, any bit in the 256 locations of data space memory, which can be written to, can be set or cleared.



PC + 1→ PC
EA = (PC) + $000
PC + 1→ PC

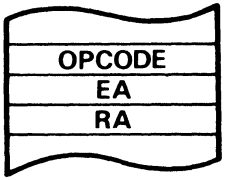
Assembly examples.

```
0DEA 9D EE        A         JMP     STRB

0DEC D8 01        A ONE     BSET    0,PORTB   SET FIRST BIT OF PORTB

0DEE D9 01        A STRB    BSET    1,PORTB   SEND DATA STROBE, RISING EDGE

0DF0 D1 01        A         BCLR    1,PORTB   RETURN STROBE TO ZERO
```

3.3.7 Bit Test and Branch Addressing Mode – 3 Bytes

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit and condition (set or clear) which is to be tested is included in the opcode, and the data space address of the byte to be tested is in the single byte immediately following the opcode byte. The third byte is sign extended to twelve bits and becomes the offset added to the program counter if the condition is true.

The single three-byte instruction allows the program to branch based on the condition of any bit in data space memory. The span of branching is from – 126 to + 129 from the opcode address. The state of the tested bit is also transferred to the carry flag.



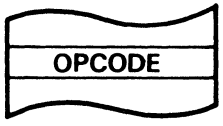
PC + 1 → PC
EA = (PC) + $000
PC + 1 → PC
(PC) → TEMP
PC + 1 → PC
EA2 = PC + TEMP IF BRANCH IS TAKEN

Assembly examples.

```
0CAB B0 00 EF     A START3 MVI    PORTA,#%11101111

0CAE CB 00 FD 0CAE START4 BRSET  3,PORTA,*

0CB1 B0 81 1E     A        LDYI   #30

0CB4 B0 81 1E     A        LDYI   #30

0CB7 8C BC        A        JSR    DEBNCE

0CB9 CB 00 F2 0CAE         BRSET  3,PORTA,START4
```

3.3.8 Register - Indirect Addressing Mode — 1 Byte

In the register-indirect addressing mode, the operand is at the address in data space pointed to by the contents of one of the indirect registers (X and Y). The particular X or Y register is selected by bit 4 of the opcode. (X register = 0, Y register = 1). Bit 4 of the opcode is decoded into an address which selects the desired X or Y register ($80 or $81). A register-indirect instruction is one byte long.



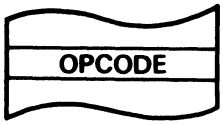
→ X or Y [EA] EA = ($80 + PC(4))
PC + 1 → PC

Assembly examples.

```
0CCB 8C D6        A        JSR    RANDOM

0CCD E1                    STA    [X]

0CCE EA 08        A        ADDA   #SEVSEG

0CD0 BD                    TAY

0CD1 F0                    LDA    [Y]

0CD2 8C D4        A        JSR    DSPLY
```

3.3.9 Inherent Addressing Mode — 1 Byte

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. These instructions are one byte long.



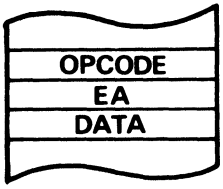
NO OPERAND NEEDED

Assembly examples.

```
00219A 0C72 B5          ROLA

00221A 0C73 B4          COMA

00223A 0C74 B7          WAIT

00225A 0C75 B6          STOP
```

3.3.10 Immediate Direct Addressing Mode

The immediate direct addressing mode is used only with the MVI instruction, where a constant is transferred into the data space memory without destroying the accumulator value. This applies to both register and RAM locations within data space.



PC + 1 → PC
EA = PC **RAM ADDRESS**
PC → PC
PC ← **DATA ADDRESS**
PC + 1 → PC **DATA TRANSFERRED TO RAM**

Assembly examples.

```
0C85 B0 04 F0    A        MVI    PORTA+DDR,#$F0 BITS 0 - 3  INPUTS
                   *                              BITS 4 - 7  OUTPUTS
0C88 B0 05 FF    A        MVI    PORTB+DDR,#$FF PORTB ALL I/O'S OUTPUTS

0C8B B0 01 00    A        MVI    PORTB,#$00 CLEAR PORTB

0C8E B0 06 01    A        MVI    PORTC+DDR,#$01 BIT 0 PORTC OUTPUT
                   *                              OTHERWISE REST INPUTS
0C91 B0 02 01    A        MVI    PORTC,#$01 SET BIT 0 PORTC
```

3.4 INSTRUCTION SET

Instructions are provided to manipulate data via an 8-bit accumulator, perform true bit manipulation, integer arithmetic, bit test and branch operations, conditional branching, subroutine call and return, and interrupt return.

Arithmetic operations include add memory to accumulator (ADD), subtract memory from accumulator (SUB), compare memory to accumulator (CMP), complement accumulator (COMA), increment and decrement memory (INC and DEC).

The instruction set of the M6804 family is symmetrical. This means that for most instructions, there is a complement instruction. Some of these instructions (plus complements) are listed below:

LDA – STA	Load and Store
BEQ – BNE	Branch if Equal and Branch if Not Equal
ADD – SUB	Add and Subtract
AND – ORA	Logic AND and Logic OR
BCLR – BSET	Bit Clear and Bit Set
JSR – RTS	Jump to Subroutine and Return from Subroutine

The symmetry provided by the M6804 family instruction set means that the programmer need only remember a few separate instructions to know the entire instruction set.
The M6804 MCU family has a set of 42 basic instructions, combined with nine addressing modes. The opcode map for these instructions is shown in Table 3.7.
They can be divided into five different types:

Register/Memory
Read-Modify-Write
Branch
Bit Manipulation
Control Instructions.

3.4.1 Register/ Memory Instructions

Most of these instructions use two operands. One operand is the accumulator and the other operand is obtained from memory using one of the addressing modes.
The Jump Unconditional (JMP) and Jump to Subroutine (JSR) instructions have no register operands. Refer to Table 3.2.

3.4.2 Read-Modify-Write Instructions

These instructions read a memory location or a register, modify or test the contents, and then write the modified value back to memory or the register. Refer to Table 3.3.

3.4.3 Branch Instructions

In this set of instructions the program branches to a different routine when a particular condition is met. When the specified condition is not met, execution continues with the next instruction. Most of the branch instructions test the state of one or both of the condition code bits. Refer to Table 3.4.
Each mnenomic of the branch instructions covers a range of 32 opcodes: e.g. BCC ranges from $40 through $5F. The actual memory location (target address) to which the branch is made is formed by adding the sign extended lower five bits of the opcode to the contents of the program counter.

3.4.4 Bit Manipulation Instructions

These instructions are used on any bit in data space memory. There are four types of bit manipulation instructions.
One group either sets or clears any single bit in a memory byte. This instruction group uses the bit set/clear addressing mode which is similar to direct addressing. The bit number (0-7) is part of the opcode.
The other group tests the state of any single bit in a memory location and branches if the bit is set or clear. These instructions have test and branch addressing. Refer to Table 3.5.

3.4.5 Control Instructions

These instructions manipulate condition code bits, control stack and interrupt operations, transfer data between the accumulator and X or Y registers, and do nothing (NOP). In the HCMOS versions there are WAIT and STOP instructions added. Refer to Table 3.6.

3.5 IMPLIED INSTRUCTIONS

Due to the fact that the accumulator and all other registers are located in RAM on the M6804, it is possible to implement some additional instructions at the assembler level which do not have specific opcodes. For example an ASLA (arithmetic shift left of accumulator) can be implemented by adding the accumulator to itself, i.e. ADD $FF; similarly, SUB $FF will effectively clear the accumulator written CLRA. These

are known as implied or assembler-recognised instructions, since the assembler will convert the mnemonic to the appropriate instruction.
Fig. 3.2 shows the implied instructions recognised by the M6804 assembler.

ASSEMBLER	INSTRUCTION	
ASLA	ADD $FF	Adds contents of the ACC to itself.
CLRA	SUB $FF	Substr. contents of the ACC from itself.
DECA	DEC $FF	Subtract *one* from ACC.
INCA	INC $FF	Add *one* to the ACC.
CLRX	MVI #$0, XREG	Move *0* to X.
CLRY	MVI #$0, YREG	Move *0* to Y.
DEC X	DEC $80	
DEC Y	DEC $81	
INCX	INC $80	
INCY	INC $81	
LDXI	MVI DATA, $80	Move constant into X
LDYI	MVI DATA, $81	Move constant into Y
TAX	STA X	
TAY	STA Y	
TXA	LDA X	
TYA	LDA Y	

Figure 3.2. Assembler Recognized Instructions (derived) with no Opcode

3.6 MOVE IMMEDIATE (MVI) INSTRUCTION

The M6804 features a powerful new instruction that allows immediate data to be transferred to data RAM with only one instruction, MVI.
When invoking MVI the accumulator is not used and the condition codes are not destroyed. Two examples for use of the MVI instruction are shown in Fig. 3.3.

3.7 STOP AND WAIT INSTRUCTIONS

In many low end applications the MCU spends much of its time waiting for external events to occur. In such cases, it is very useful to reduce the power consumption when the MCU is not actually at work, especially when low power consumption is critical.
For this reason there are STOP and WAIT instructions on the HCMOS devices of the M6804 family.
The STOP instruction places the HCMOS MCU in its lowest power consumption mode. In STOP mode, the internal oscillator is turned off causing all internal processing and the timer to be halted. The timer status/control register bits are not changed. The external interrupt is enabled. All I/O lines remain unchanged. The processor can only be brought out of the STOP mode by pulling low either the $\overline{IRQ}$ or $\overline{RESET}$ input pins. During the exit from the STOP mode, the timer is used to provide a delay for the oscillator to stabilize. The TSCR will be as it was before; the counter register will be in an all zero state.
The WAIT instruction places the MCU in a low-power consumption mode, but not as low as in the STOP mode. In WAIT mode, the clock is disabled from all internal circuitry except the timer circuit. Thus all internal processing is halted. The timer may, if desired, continue to count down (by setting the PSI, bit in the TSCR). All other registers, memory and I/O lines remain in their last state. A timer interrupt (ETI bit) may be enabled by software prior to entering the WAIT mode to allow an exit via a timer interrupt. Alternatively an exit may be made by pulling low either the $\overline{IRQ}$ or $\overline{RESET}$ pins.

Example 1 Port initialization

M6805 code

```
LDA #$F0
STA PORTA+DDR
```

M6804 code

```
MVI #$F0,PORTA+DDR
```

Example 2 Read Port A immmediately after load Port B, and test on ZERO flag

M6805 code

```
LDA PORTA   read Port A
STA ATEMP
LDA #$AA
STA PORTB   load Port B
LDA ATEMP   work on previous
            reading
SUB #$02
BEQ NEXT    test on ZERO flag
```

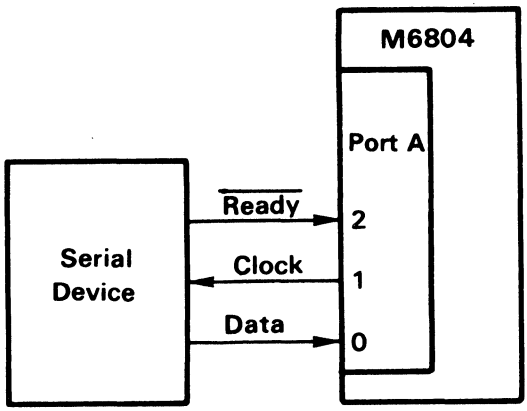
M6804 code

```
LDA PORTA       read Port A
MVI #$AA,PORTB  load Port B
SUB #$02
BEQ NEXT        test on ZERO bit flag
```

Figure 3.3. Examples of MVI Instruction

3.8 BIT MANIPULATION EXAMPLE

The M6804 in common with the M6805, has the ability to set or clear any single random access memory (RAM) writable bit with a single instruction (BSET, BCLR). Any bit in data space, including ROM, can be tested, using the BRSET or BRCLR instructions, and the program branches as a result of its state. The carry bit equals the value of the bit referenced by BRSET or BRCLR. A rotate instruction may then be used to accumulate serial input data in a RAM location or register. The capability to work with any bit in RAM, ROM or I/O allows the user to have individual flags in RAM or to handle I/O bits as control lines.
The example in Fig. 3.4 illustrates the usefulness of the bit manipulation and test instructions.
Assume that the MCU is to communicate with an external serial device. The external device has a data ready signal, a data output line, and a clock line (to clock data one bit at a time, MSB first, out of the device). The MCU waits until data is ready, clocks the external device, picks up the data in the carry flag (C bit), clears the clock line, and finally accumulates the data bit in the accumulator.




```
0D50 CA 00 FD 0D50 SELF   BRSET  2,PORTA,SELF WAIT FOR READY SIGNAL



0D53 D9 00       A        BSET   1,PORTA  DATA READY CLOCK IT IN
0D55 C0 00 00 0D58        BRCLR  0,PORTA,CONT TEST  0 - BIT LINE OF PORTA
                     *                            SO THAT DATA IS STORED IN
                     *                            CARRY BIT
0D58 B5              CONT ROLA             THE VALUE OF CARRY COMES INTO
                     *                            THE ACCUMULATOR
                     *
```

Figure 3.4. Bit Manipulation Example

3.9 PROGRAMMING INTERRUPTS

The MCUs of the M6804 family provide only one user interrupt vector. Nevertheless a timer interrupt is possible although the technique is different on HMOS and HCMOS devices. In both cases, interrupt polling is necessary in the interrupt routine.

In general the HMOS and HCMOS devices are interrupted by applying a logic low signal to the $\overline{IRQ}$ pin. The HCMOS devices can also be interrupted by a timer counter underflow if the ETI bit (Enable Timer Interrupt) is set in the TSCR.

For HMOS devices it is necessary to generate a quasi timer interrupt in one of the following ways.

1. The timer is set to work in the output mode and the timer pin is hardwired to the $\overline{IRQ}$ pin. Bit 4 (D_{out}) of the TSCR should be "0" which will cause an interrupt every time the timer counter decrements to zero. At the same time bit 7 (TMZ) of the TSCR gets set to "1". Thus within the interrupt routine it is necessary to read the TMZ bit to determine the source of the interrupt. A "1" indicates a timer interrupt, while "0" indicates an external interrupt. Program control should then be diverted to the appropriate part of the interrupt routine, see Fig. 3.11.
2. During normal program execution the state of the TMZ bit is periodically examined. If high, it indicates that the timer counter has decremented to zero, and program flow can then be directed to the interrupt routine.

Note that in HCMOS devices a timer interrupt can be enabled in both input or output modes. In either mode, the interrupt routine should contain the decision process described in point 1) to determine the source of the interrupt.

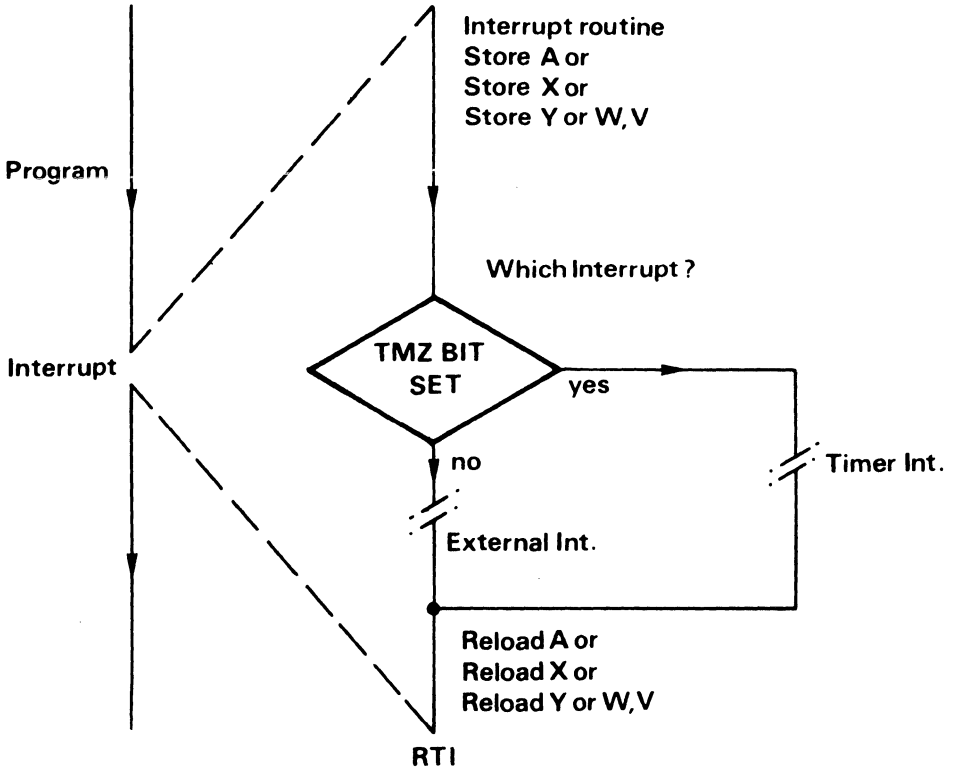


Figure 3.11. Interrupt Flow Chart.
Decision about Timer Interrupt or Hardware Interrupt is done by examining the situation of bit TMZ of the Timer/Status Control Register.

Mnemonic	Addressing Modes									Flags	
	Inherent	Immediate	Direct	Short Direct	Bit Set Clear	Bit Test Branch	Register Indirect	Extended	Relative	Z	C
ADD		X	X				X			Λ	Λ
AND		X	X				X			Λ	•
ASLA	Assembler converts this to "ADD $FF"									•	•
BCC									X	•	•
BCLR					X					•	•
BCS									X	•	•
BEQ									X	•	•
BHS	Assembler converts this to "BCC"									•	•
BLO	Assembler converts this to "BCS"									•	•
BNE									X	•	•
BRCLR						X				•	Λ
BRSET						X				•	Λ
BSET					X					•	•
CLRA	Assembler converts this to "SUB $FF"									Λ	Λ
CLRX	Assembler converts this to "MVI $80,#0"									•	•
CLRY	Assembler converts this to "MVI $81,#0"									•	•
CMP		X	X				X			Λ	Λ
COMA	X									Λ	Λ
DEC			X	X			X			Λ	•
DECA	Assembler converts this to "DEC $FF"									Λ	•
DECX	Assembler converts this to "DEC $80"									Λ	•
DECY	Assembler converts this to "DEC $81"									Λ	•
INC			X	X			X			Λ	•
INCA	Assembler converts this to "INC $FF"									Λ	•
INCX	Assembler converts this to "INC $80"									Λ	•
INCY	Assembler converts this to "INC $81"									Λ	•
JMP								X		•	•
JSR								X		•	•
LDA		X	X	X			X			Λ	•
LDXI	Assembler converts this to "MVI $80,DATA"									•	•
LDYI	Assembler converts this to "MVI $81,DATA"									•	•
MVI		X	X							•	•
NOP	Assembler converts this to "BEQ (PC) + 1"									•	•
ROLA	X									Λ	Λ
RTI	X									Λ	Λ
RTS	X									•	•
STA			X	X			X			Λ	•
STOP*	X									•	•
SUB		X	X				X			Λ	Λ
TAX	Assembler converts this to "STA $80"									•	•
TAY	Assembler converts this to "STA $81"									•	•
TXA	Assembler converts this to "LDA $80"									•	•
TYA	Assembler converts this to "LDA $81"									•	•
WAIT*	X									•	•

Flag Symbols: Z = Zero, C = Carry/Borrow, Λ = Test and Set if True, Cleared Otherwise, • = Not Affected ***HCMOS ONLY**

Table 3.1. M6804 Instruction Set

Table 3.2 Register/Memory Instructions

		Addressing Modes																			
		Indirect				Immediate			Direct			Inherent			Extended			Short-Direct			
Function	Mnem	Opcode XP	Opcode YP	# Bytes	# Cycles	Opcode	# Bytes	# Cycles	Opcode	# Bytes	# Cycles	Opcode	# Bytes	# Cycles	Opcode	# Bytes	# Cycles	Opcode	# Bytes	# Cycles	Special Notes
Load A from Memory	LDA	E0	F0	1	4	E8	2	4	F8	2	4	–	–	–	–	–	–	AC-AF	1	4	1
Load XP from Memory	LDXI	–	–	–	–	B0	3	4	–	–	–	–	–	–	–	–	–	–	–	–	4
Load YP from Memory	LDYI	–	–	–	–	B0	3	4	–	–	–	–	–	–	–	–	–	–	–	–	4
Store A in Memory	STA	E1	F1	1	4	–	–	–	F9	2	4	–	–	–	–	–	–	BC-BF	1	4	2
Add to A	ADD	E2	F2	1	4	EA	2	4	FA	2	4	–	–	–	–	–	–	–	–	–	–
Subtract from A	SUB	E3	F3	1	4	EB	2	4	FB	2	4	–	–	–	–	–	–	–	–	–	–
Arithmetic Compare with Memory	CMP	E4	F4	1	4	EC	2	4	FC	2	4	–	–	–	–	–	–	–	–	–	–
AND Memory to A	AND	E5	F5	1	4	ED	2	4	FD	2	4	–	–	–	–	–	–	–	–	–	–
Jump to Subroutine	JSR	–	–	–	–	–	–	–	–	–	–	–	–	–	8 (TAR)	2	4	–	–	–	3
Jump Unconditional	JMP	–	–	–	–	–	–	–	–	–	–	–	–	–	9 (TAR)	2	4	–	–	–	3
Clear A	CLRA	–	–	–	–	–	–	–	FB	2	4	–	–	–	–	–	–	–	–	–	–
Clear XP	CLRX	–	–	–	–	–	–	–	FB	2	4	–	–	–	–	–	–	–	–	–	–
Clear YP	CLRY	–	–	–	–	–	–	–	FB	2	4	–			–	–	–	–	–	–	–
Complement A	COMA	–	–	–	–	–	–	–	–	–	–	B4	1	4	–	–	–	–	–	–	–
Move Immediate Value to Memory	MVI	–	–	–	–	B0	3	4	B0	3	4	–	–	–	–	–	–	–	–	–	5
Rotate A Left and Carry	ROLA	–	–	–	–	–	–	–	–	–	–	B5	1	4	–	–	–	–	–	–	–
Arithmetic Left Shift of A	ASLA	–	–	–	–	–	–	–	FA	2	4	–	–	–	–	–	–	–	–	–	–

SPECIAL NOTES

1. In Short-Direct addressing, the LDA mnemonic represents opcode AC, AD, AE, and AF. This is equivalent to RAM locations $80 (AC), $81 (AD), $82 (AE), and $83 (AF).
2. In Short-Direct addressing, the STA mnemonic represents opcode BC, BD, BE, and BF. This is equivalent to RAM locations $80 (BC), $81 (BD), $82 (BE), and $83 (BF).
3. In Extended addressing, the four LSBs of the opcode (Mnemonic JSR and JMP) are formed by the four MSBs of the target address.
4. In Immediate addressing, the LDXI and LDYI are mnemonics which are recognized as follows:
   LDXI = MVI $80,data
   LDYI = MVI $81,data Where data is a one-byte hexadecimal number.
5. In both Immediate and Direct addressing, the MVI instruction has the same opcode (80).

Branch Instructions

Hi / Low	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111
0 0000	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL
1 0001	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL
2 0010	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL
3 0011	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL
4 0100	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL
5 0101	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL
6 0110	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL
7 0111	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL
8 1000	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL
9 1001	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL
A 1010	2 BNE 1 REL	2 BNE 2 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL
B 1011	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL
C 1100	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL
D 1101	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL
E 1110	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL
F 1111	2 BNE 1 REL	2 BNE 1 REL	2 BEQ 1 REL	2 BEQ 1 REL	2 BCC 1 REL	2 BCC 1 REL	2 BCS 1 REL	2 BCS 1 REL

Abbreviations for Address Modes

INH	Inherent
S-D	Short Direct
B-T-B	Bit Test and Branch
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
R-IND	Register Indirect

* Indicates Instruction Reserved for Future Use

\# Indicates Illegal Instruction

Table 3.7. M6804 Opcode Map

	Register/Memory, Control, and Read/Modify/Write Instructions				Bit Manipulation Instructions		Register/Memory and Read/Modify/Write		
	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	Hi / Low
	4 JSRn 2 EXT	4 JMPn 2 EXT	*	4 MVI 3 IMM	5 BRCLR0 3 B-T-B	4 BCLR0 2 BSC	4 LDA 1 R-IND	4 LDA 1 R-IND	0 0000
	4 JSRn 2 EXT	4 JMPn 2 EXT	*	*	5 BRCLR1 3 B-T-B	4 BCLR1 2 BSC	4 STA 1 R-IND	4 STA 1 R-IND	1 0001
	4 JSRn 2 EXT	4 JMPn 2 EXT	*	2 RTI 1 INH	5 BRCLR2 3 BT-B	4 BCLR2 2 BSC	4 ADD 1 R-IND	4 ADD 1 R-IND	2 0010
	4 JSRn 2 EXT	4 JMPn 2 EXT	*	2 RTS 1 INH	5 BRCLR3 3 B-T-B	4 BCLR3 2 BSC	4 SUB 1 R-IND	4 SUB 1 R-IND	3 0011
	4 JSRn 2 EXT	4 JMPn 2 EXT	*	4 COMA 1 INH	5 BRCLR4 3 B-T-B	4 BCLR4 2 BSC	4 CMP 1 R-IND	4 CMP 1 R-IND	4 0100
	4 JSRn 2 EXT	4 JMPn 2 EXT	*	4 ROLA 1 INH	5 BRCLR5 3 B-T-B	4 BCLR5 2 BSC	4 AND 1 R-IND	4 AND 1 R-IND	5 0101
	4 JSRn 2 EXT	4 JMPn 2 EXT	*	2 STOP 1 INH	5 BRCLR6 3 B-T-B	4 BCLR6 2 BSC	4 INC 1 R-IND	4 INC 1 R-IND	6 0110
	4 JSRn 2 EXT	4 JMPn 2 EXT	*	2 WAIT 1 INH	5 BRCLR7 3 B-T-B	4 BCLR7 2 BSC	4 DEC 1 R-IND	4 DEC 1 R-IND	7 0111
	4 JSRn 2 EXT	4 JMPn 2 EXT	4 INC 1 S-D	4 DEC 1 S-D	5 BRSET0 3 B-T-B	4 BSET0 2 BSC	4 LDA 2 IMM	4 LDA 2 DIR	8 1000
	4 JSRn 2 EXT	4 JMPn 2 EXT	4 INC 1 S-D	4 DEC 1 S-D	5 BRSET1 3 B-T-B	4 BSET1 2 BSC	#	4 STA 2 DIR	9 1001
	4 JSRn 2 EXT	4 JMPn 2 EXT	4 INC 1 S-D	4 DEC 1 S-D	5 BRSET2 3 B-T-B	4 BSET2 2 BSC	4 ADD 2 IMM	4 ADD 2 DIR	A 1010
	4 JSRn 2 EXT	4 JMPn 2 EXT	4 INC 1 S-D	4 DEC 1 S-D	5 BRSET3 3 B-T-B	4 BSET3 2 BSC	SUB IMM	4 SUB 2 DIR	B 1011
	4 JSRn 2 EXT	4 JMPn 2 EXT	4 LDA 1 S-D	4 STA 1 S-D	5 BRSET4 3 B-T-B	4 BSET4 2 BSC	4 CMP 2 IMM	CMP DIR	C 1100
	4 JSRn 2 EXT	4 JMPn 2 EXT	4 LDA 1 S-D	4 STA 1 S-D	5 BRSET5 3 B-T-B	4 BSET5 2 BSC	4 AND 2 IMM	4 AND 2 DIR	D 1101
	4 JRSn 2 EXT	4 JMPn 2 EXT	4 LDA 1 S-D	4 STA 1 S-D	5 BRSET6 3 B-T-B	4 BSET6 2 BSC	#	4 INC 2 DIR	E 1110
	4 JSRn 2 EXT	4 JMPn 2 EXT	4 LDA 1 S-D	4 STA 1 S-D	5 BRSET7 3 B-T-B	4 BSET7 2 BSC	#	4 DEC 2 DIR	F 1111

LEGEND

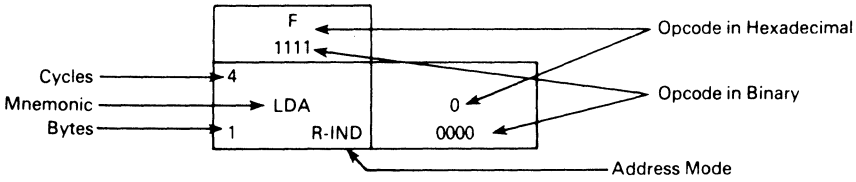


Table 3.7. M6804 Opcode Map (Continued)

Function	Mnem	Addressing Modes: Indirect Opcode XP	Indirect Opcode YP	Indirect # Bytes	Indirect # Cycles	Direct Opcode	Direct # Bytes	Direct # Cycles	Short-Direct Opcode	Short-Direct # Bytes	Short-Direct # Cycles	Special Notes
Increment Memory Location	INC	E6	F6	1	4	FE	2	4	A8-AB	1	4	1, 3
Increment A	INCA	–	–	–	–	FE	2	4	–	–	–	–
Increment XP	INCX	–	–	–	–	–	–	–	A8	1	4	–
Increment YP	INCY	–	–	–	–	–	–	–	A9	1	4	–
Decrement Memory Location	DEC	E7	F7	1	4	FF	2	4	B8-BB	1	4	2, 4
Decrement A	DECA	–	–	–	–	FF	2	4	–	–	–	–
Decrement XP	DECX	–	–	–	–	–	–	–	B8	1	4	–
Decrement YP	DECY	–	–	–	–	–	–	–	B9	1	4	–

SPECIAL NOTES

1. In Short-Direct addressing, the INC mnemonic represents opcode A8, A9, AA, and AB. These are equivalent to RAM locations $80 (A8), $81 (A9), $82 (AA), and $83 (AB).
2. In Short-Direct addressing, the DEC mnemonic represents opcode B8, B9, BA, and BB. These are equivalent to RAM locations $80 (B8), $81 (B9), $82 (BA), and $83 (BB).
3. In Indirect addressing, the INC mnemonic represents opcode E6 or F6, and causes the location pointed to by XP (E6 opcode) or YP (F6 opcode) to be incremented.
4. In Indirect addressing, the INC mnemonic represents opcode E7 or F7, and causes the location pointed to by XP (E7 opcode) or YP (F7 opcode) to be incremented.

Table 3.3. Read-Modify-Write Instructions

Function	Mnem	Relative Addressing Mode: Opcode	# Bytes	# Cycles	Special Notes
Branch if Carry Clear	BCC	40-5F	1	2	1
Branch if Higher or Same	(BHS)	40-5F	1	2	1, 2
Branch if Carry Set	BCS	60-7F	1	2	1
Branch if Lower	(BLO)	60-7F	1	2	1, 3
Branch if Not Equal	BNE	00-1F	1	2	1
Branch if Equal	BEQ	20-3F	1	2	1

SPECIAL NOTES

1. Each mnemonic of the Branch Instructions covers a range of 32 opcodes; e.g., BCC ranges from 40 through 5F. The actual memory location (target address) to which the branch is made is formed by adding the sign extended lower five bits of the opcode to the contents of the program counter.
2. The BHS instruction (shown in parentheses) is identical to the BCC instruction. The C bit is clear if the register was higher or the same as the location in the memory to which it was compared.
3. The BLO instruction (shown in parentheses) is identical to the BCS instruction. The C bit is set if the register was lower than the location in memory to which it was compared.

Table 3.4. Branch Instructions

Function	Mnem	Addressing Modes: Bit Set/Clear Opcode	Bit Set/Clear # Bytes	Bit Set/Clear # Cycles	Bit Test and Branch Opcode	Bit Test and Branch # Bytes	Bit Test and Branch # Cycles	Special Note
Branch IFF Bit n is set	BRSET n (n=0 . . . . 7)	–	–	–	C8+n	3	5	1
Branch IFF Bit n is clear	BRCLR n (n=0 . . . . 7)	–	–	–	C0+n	3	5	1
Set Bit n	BSET n (n=0 . . . . 7)	D8+n	2	4	–	–	–	1
Clear Bit n	BCLR n (n=0 . . . . 7)	D0+n	2	4	–	–	–	1

SPECIAL NOTE

1. The opcode is formed by adding the bit number (0-7) to the basic opcode. For example: to clear bit six using the BSET6 instruction the opcode becomes DE (D8+6); BCLR5 becomes (C0+5); etc.

Table 3.5. Bit Manipulation Instructions

		Addressing Modes									
		Short-Direct			Inherent			Relative			
Function	Mnem	Opcode	# Bytes	# Cycles	Opcode	# Bytes	# Cycles	Opcode	# Bytes	# Cycles	Special Notes
Transfer A to XP	TAX	BC	1	4	–	–	–	–	–	–	–
Transfer A to YP	TAY	BD	1	4	–	–	–	–	–	–	–
Transfer XP to A	TXA	AC	1	4	–	–	–	–	–	–	–
Transfer YP to A	TYA	AD	1	4	–	–	–	–	–	–	–
Return from Subroutine	RTS	–	–	–	B3	1	2	–	–	–	–
Return from Interrupt	RTI	–	–	–	B2	1	2	–	–	–	–
No-Operation	NOP	–	–	–	–	–	–	–	–	–	1
Stop	STOP	–	–	–	B6	1	2	–	–	–	–
Wait	WAIT	–	–	–	B7	1	2	–	–	–	–

SPECIAL NOTE

1 . The NOP instruction is equivalent to a branch if equal (BEQ) to the location designated by PC + 1.

Table 3.6. Control Instructions

Development Tools

4.1 INTRODUCTION

Motorola Microsystems assists you with advanced Development Systems during the entire design cycle of your 6804 design.

The M6804 processors are amongst the most complex integrated circuits. It is mandatory for a designer to understand and analyze all processes (externally and internally) to achieve a proper and reliable application of the M6804. A voltmeter and oscilloscope are necessary, but not sufficient by themselves. You will also need A REAL TIME EMULATOR. This enables the substitution of the M6804 in your application, as well as providing the interface to a high performance computer. This computer prepares the information coming from the emulator so that programs can be written and debugged easily.

The basic work station for your 6804 design is comprised of two important elements:

- A central development computer
- A hardware development station

A block diagram is shown in Fig. 4-1.

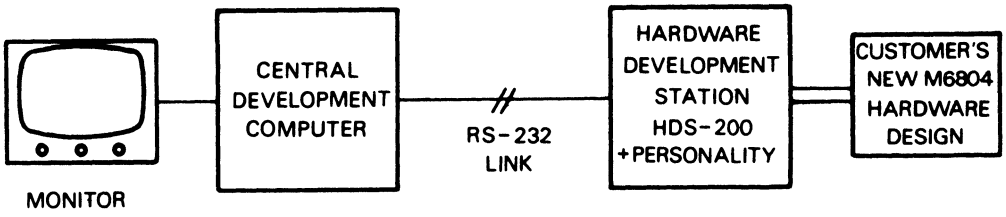


Figure 4.1. General Block Diagram of an M6804 Development Station

4.2 CENTRAL DEVELOPMENT COMPUTERS

For the **Central Development Computer** Motorola can offer the following options:

- Single or multi-user station.
- 6804 development support only; upgradable support for all Motorola 8-bit processors; **or** upgradable support for all Motorola 8/16/32-bit processors.
- Mass-storage between 300 Kbyte and 100 Mbyte.

Common to all development computers are the following:

- Cross-macroassembler for M6804
- Screen oriented CRT-editor
- Convenient utilities provided by operating system
- Communication support for remote development station

Thus, Motorola's product range can cover all the different needs of an M6804 designer.

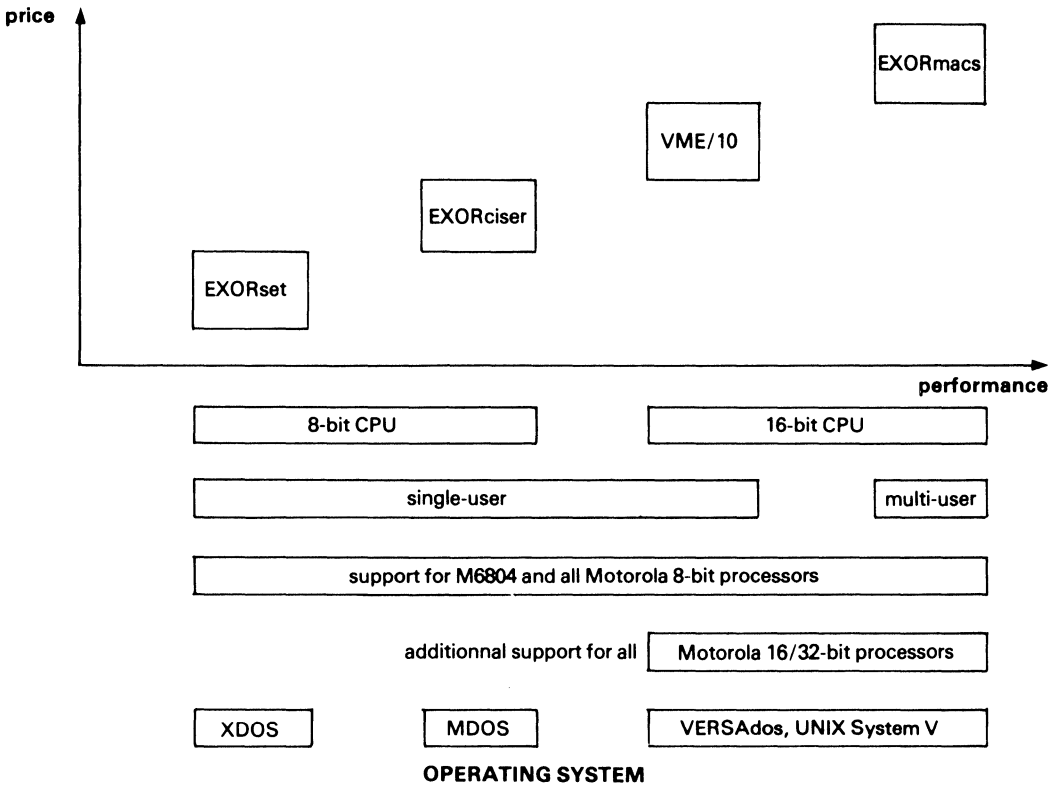


Figure 4.2. Product Overview Central Development Computer

The criteria to select among the central development computers are as follows:

- How fast do I want data processed (i.e. assembly speed)? Select between an 8- and 16-bit CPU.
- Is there only one engineer or are there several engineers working in parallel? Select between a single or multi-user system.
- Do I plan to use this computer for other processors as well? Select between 8-bit only and 8/16/32-bit computers.
- How much data am I going to create? Select between:

EXORset : 2 x 160 Kbyte floppy (XDOS)
expandable to 1.3 Mbyte

EXORciser : 2 x 0.5 Mbyte floppy (MDOS)
expandable to 2 Mbyte

VME/10 : 15 Mbyte Winchester (VERSAdos/UNIX)
+ 0.5 Mbyte floppy

EXORmacs : 15 Mbyte Lark Winchester (VERSAdos/UNIX)
or 32 Mbyte Hard disk (VERSAdos/UNIX)
or 50 Mbyte Lark Winchester (VERSAdos/UNIX)
or 96 Mbyte Hard disk (VERSAdos/UNIX)

4.3 THE HDS-200 HARDWARE DEVELOPMENT STATION

Apart from the central development computer, which is independent of the target processor, the second important element of any development work station is the **Hardware Development Station.**
This station is dependent on the target processor. For the M6804 family, Motorola Microsystems offers the popular HDS-200 station, which can also support any member of the M6805 processor family. The features of the HDS-200 station are:

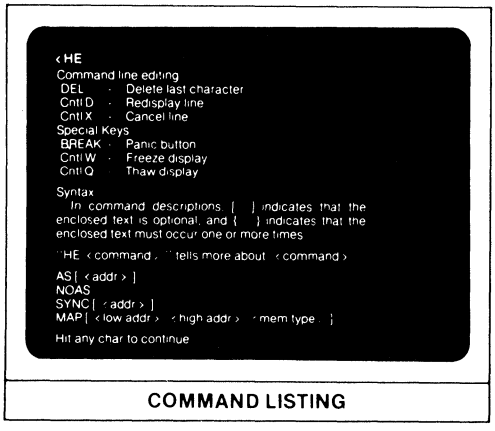
- RS-232 link to central development computer, allowing long distance operation independent of the type of central development computer.
- Supports all M6804/M6805 processors by exchanging different personality emulators (see ordering information in section 4.6).
- Real-time emulation of all processor features.
- Assembly/disassembly of object code
- Emulation RAM to load/display/modify object code
- Macro commands to simplify debug session
- Chained breakpoints
- Breakpoints applicable to program flow and/or data access
- HELP function
- Printer port
- Cost saving structure

The cost-saving structure of the HDS-200 system becomes obvious when you consider that it provides all the basic features of a realtime emulator common to all M6804/M6805 processors. Only the personality is totally processor dependent since it acts as an adaptor between your hardware and the generic HDS-200. If you want to work with different processors in the M6804/M6805 family, just exchange the personality and continue to use your development computer and the HDS-200.

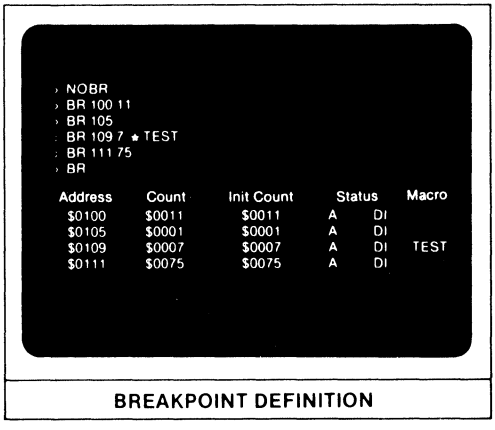
4.4 DEBUGGING WITH THE HDS-200

The use of the HDS-200's features can be best explained by looking at the typical design cycle of a microcomputer application:
The first contact between the design engineers and the development system is established when the details of software and hardware are finalized. Let us focus on the software side.
Any new software will have to be keypunched the first time. A screen oriented editor aids you in typing your M6804 assembler source code into one of Motorola's central development computers. This editor will also store it on the appropriate mass-storage device or will divert it to a printer which can be connected to any Motorola development system. As long as you want to make changes to your source code, the editor will help you.
Once the editing work is finalized, you can call the cross-macroassembler for M6804. This will translate your source code into executable hexadecimal object code. These cross-assemblers on VME/10 and EXORmacs provide you with important features such as structure commands. With extra commands like FOR...TO, WHILE...DO, IF... THEN...ELSE, the M6804 reaches a level of convenience which only high-level languages can give. The object code is transposed into a special Motorola format, the S-record, which can then be downloaded into the HDS-200 at a speed of 9600 Baud, and stored in the emulation RAM. Since the emulator is basically an M6804 processor in discrete logic, the software functionality can be checked under real-time conditions running in the emulator. The object code is stored in the emulation RAM. Thus you can display and modify it at any time, eg. when a bug must be located and fixed.
To keep you constantly informed about the status of your program, a disassembler translates your object code back into source code mnemonics. If you want to insert new instructions you do not have to start editor/assembler and downloader again. Instead, the line-assembler of the HDS-200 assembles your new statements into hexadecimal object code. If you forget how to do this or how to use other features of the HDS-200, just call for HELP and you will be informed of the options. There are many options which are very flexible in that they allow the insertion of parameters. To make use of these options as well as to simplify your debug work, you can define macrocommands which are a user-defined sequence of possible HDS-200 commands. These macros can be stored to have them available at all times.

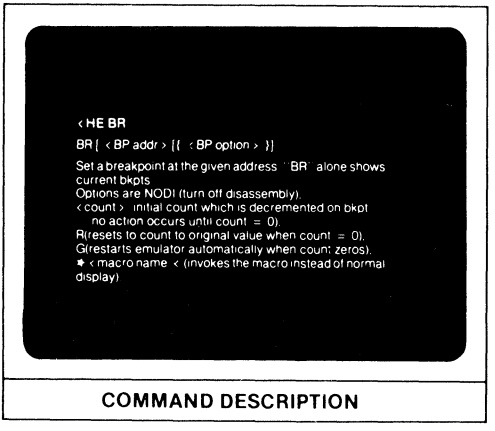
Meanwhile, the hardware engineer has put together all elements of the design and wants to test its functionality. Together with the HDS-200, he can simulate the presence of the M6804 and can thus try to access all of the hardware elements under the control of the central development computer.
The final step is the system integration where software and hardware are brought together and tested. The HDS-200 assists you in this. Since the whole system has become more complex by now, a feature like chained breakpoints (one breakpoint is armed only if another one was reached before) helps a lot. External signals can be controlled by software and visualized on a logic analyzer or oscilloscope which can be synchronized by the HDS-200. Fig. 4-3 gives four examples of typical HDS-200 screens during a debug session.
Finally, when everything is running as desired, just remove the emulation plug and insert the 'real' M6804.



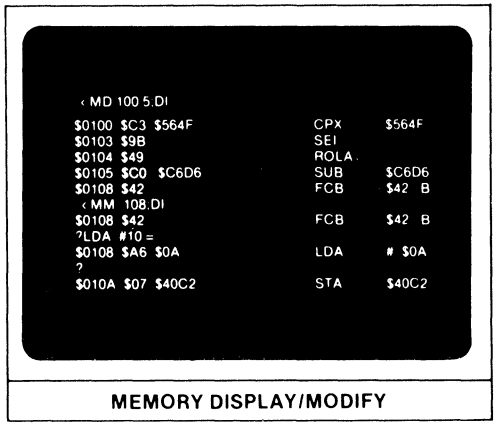
COMMAND LISTING



BREAKPOINT DEFINITION



COMMAND DESCRIPTION



MEMORY DISPLAY/MODIFY

Figure 4.3. Emulation Features of the HDS-200

4.5 SINGLE USER VS. MULTI-USER

If both hardware and software designers can share the same work station, a single-user station like the EXORset or VME/10 would be the best choice for you. If you want to cut down your design time and require several engineers to work in parallel, or if you have several projects with M6804 and other Motorola microprocessors, or if you have a multi-processor application, a multi-user station is more advisable.

Possible configurations of development systems are shown in Figs. 4-4 to 4-7.

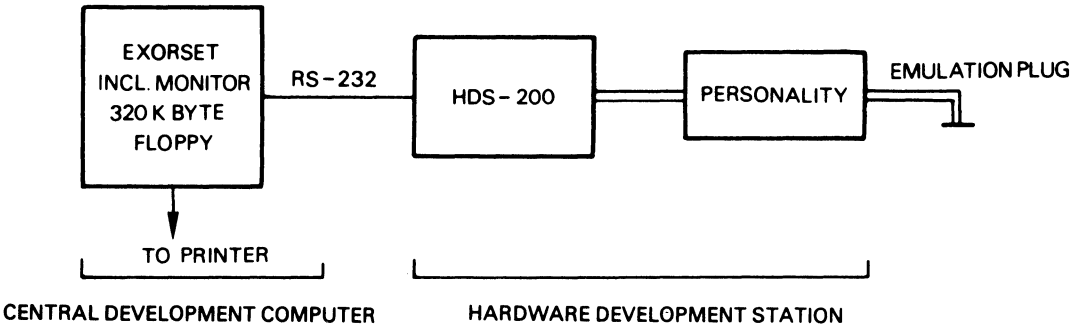


Figure 4.4. Single-User M6804 Development Workstation Minimum Configuration with EXORset

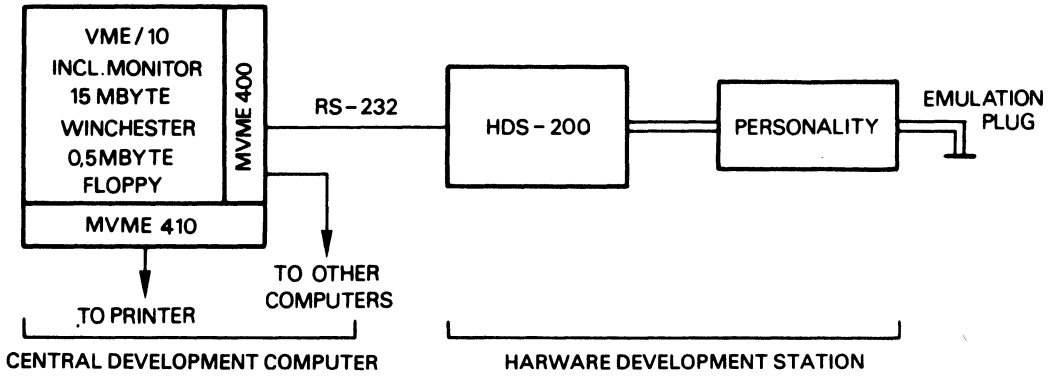


Figure 4.5. Single-User M6804 Development Workstation Mid-Range Configuration with VME/10

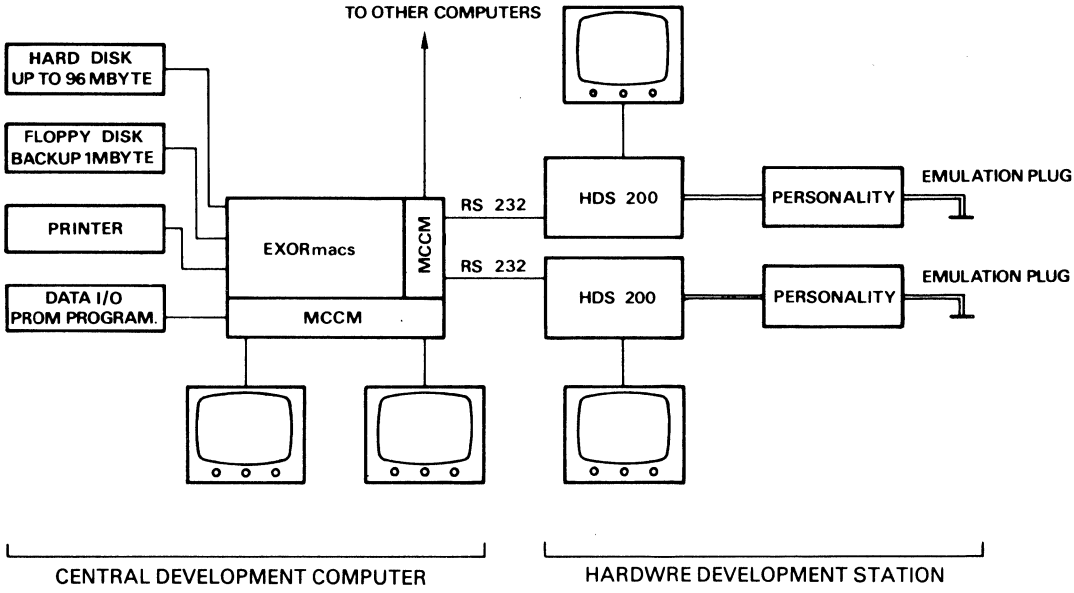


Figure 4.6. Multi-user M6804 development system here: four user EXORmacs configuration

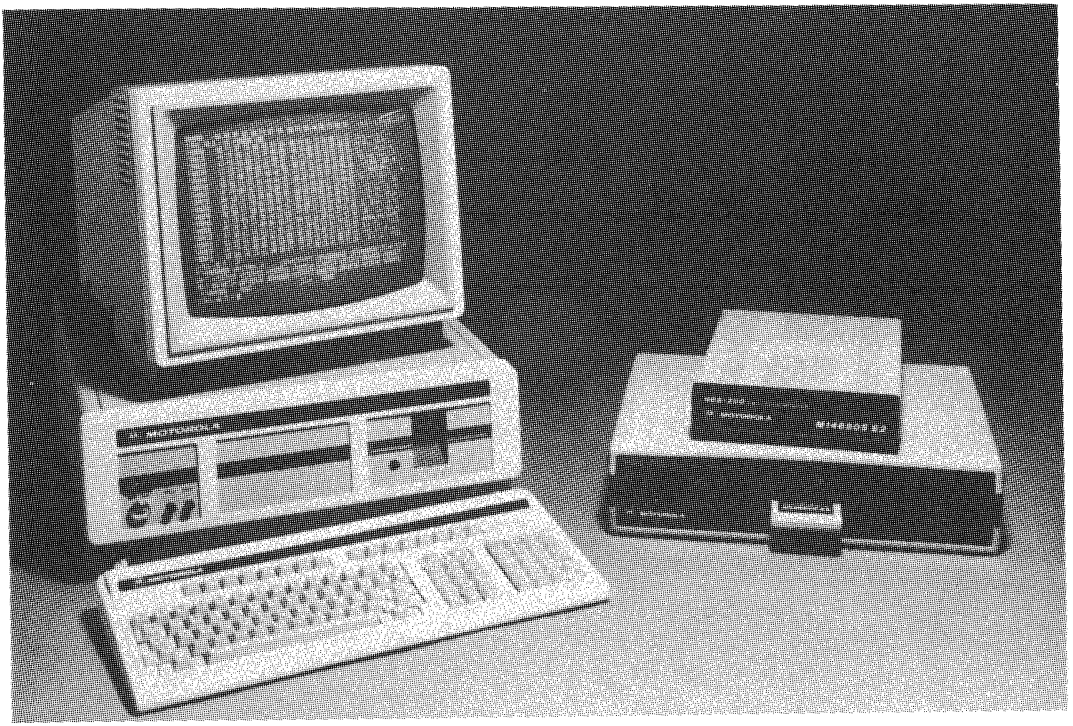


Figure 4.7. VME/10 Central development computer with HDS-200 hardware development station

4.6 ORDERING INFORMATION

FUNCTION	PART NUMBER
HARDWARE	
● Development station HDS-200	M68HDS202
6804 Personality emulators working with HDS-200	
● MC6804P2	M6804P2HM
● MC68HC04P2/P3	M68HC04P23HM
6805 Personality Emulators working with HDS-200	
● MC146805E2	M146805E2HM
● MC146805F2/MC1468705F2	M146805F2HM
● MC146805G2/MC1468705G2	M146805G2HM
● MC68HC05C4	M68HC05C4HM
● MC6805P2/P3/P4/MC68705P3/P5	M6805P234HM
● MC6805R2/R3/U2/U3/K2/MC68705R3/U3	M6805RU23HM
● MC6805S2/MC68705S3	M6805S2HM
● MC6805T2	M6805T2HM
● Exorset Development Computer	M6809SET100
● VME/10 Development Computer	M68K102B2
● RS-232 Interface for VME/10	MVME400
● Printer interface for VME/10	MVME410
● EXORmacs Multi-user Development Computer with 16 Mbyte Lark Winchester	M68KMACSL2-512
● 4 X RS-232 expansion	M68KMCCM
● Terminal for EXORmacs	M68SXD10255A
● Floppy disk backup drive	M68DSK4-2E
● Printer for EXORset / VME/10 / EXORmacs 250 characters/sec dot matrix	M68PRT400N2
SOFTWARE	
● CRT Editor, Operating system and download software	Comes with central development computer
● 6804 Assembler for EXORset / EXORciser	Comes with personality
● 6804 Assembler on VME/10	M68V4XBASM
● 6804 Assembler on EXORmacs	Contact sales office

Self-Check and Testing

5.1 INTRODUCTION

The purpose of this chapter is to describe the test philosophy of the M6804 using the SELF-CHECK and the ROM VERIFY modes. These two test mechanisms are used during the production phase of the device and provide the user with test procedures which, when passed, assure the total functionality of the chip. To implement these test concepts only simple test circuits are required externally, or alternatively they can be run under control of an LSI tester.

Four modes of operation are provided in the MC6804/MC68HC04 MCU's: Single Chip (default), Non-User, Self-Check and Rom Verify. If at the exit of Reset (external reset or power-on reset), the mode select pin (MDS) is held high, the states of pins PA6 and PA7 are latched and decoded to determine the chip operating mode.

The M6804 contains two separate ROM spaces the contents of which will change based on customer pattern requirements, the Data ROM and the Program ROM (see Figure 2-2). The contents of each ROM are verified to be correct at test time. The Data ROM is verified during Self-Check mode, the Program ROM is verified in ROM Verify mode. Note that it is not necessary to know the ROM contents to perform these tests.

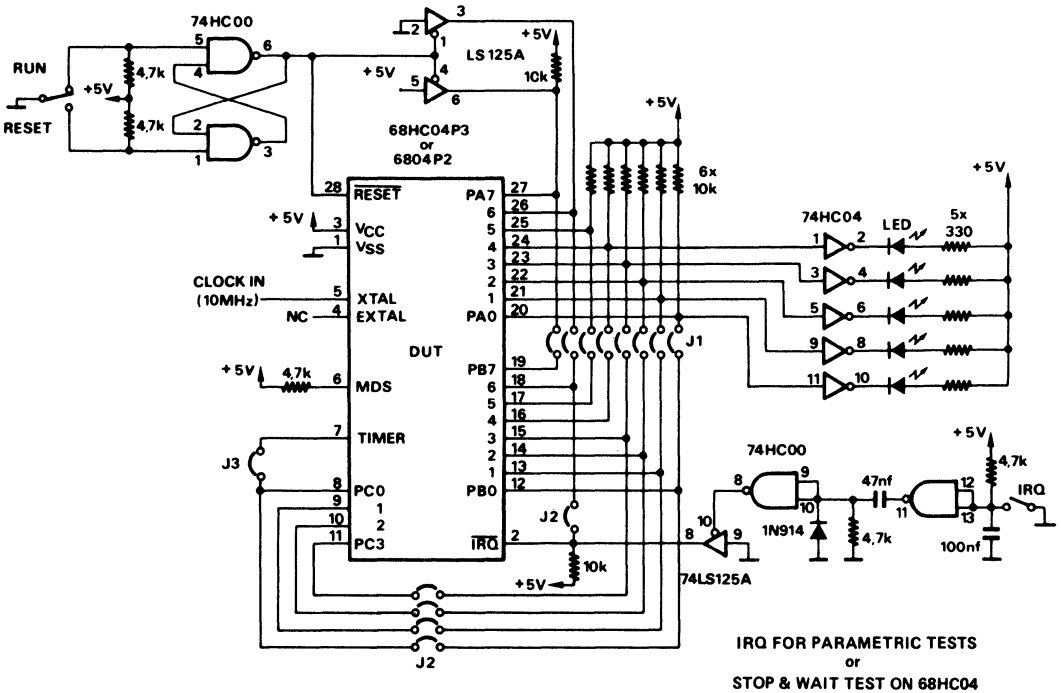


Figure 5.1. Self-Check Circuit

5.2 SELF-CHECK MODE

The self-check capability of the M6804 MCU provides an internal check to determine if the part is functional without the need of an external tester. To facilitate testing, a signature analysis circuit has been included on the chip. This circuit consists of a 16-bit shift register configured to perform a Cyclic Redundancy Check (CRC) using the CCITT polynomial. It is memory mapped in two bytes of data space at addresses $0A and $0B. (All addresses will refer to the MC68HC04P3 address map.)

To perform a functional check of the MCU, a simple external circuit needs to be connected as shown in Figure 5-1 and its operation is visualized on the LEDS on Port A. The MDS, PA7 pins are held high while PA6 pin is held low during a $\overline{\text{RESET}}$ low to high transition, which forces the Self-Check mode to be selected. Then the Self-Test program located in program space (address $800 - $95F and $FF8 - $FFB) is executed, causing all data on the bus to be monitored by the CRC.

The self-test program was designed to exercise as many portions of the processor as possible in less than 350 bytes. It is divided into six sections which are described below. Also reference to Figure 5-2 should be made:

- **Stack Test**

  This test verifies the stack by filling it with four successive subroutine calls and emptying it with five successive returns. The entire stack is filled and emptied twice.

- **I/O Ports Test**

  All ports are tested as inputs and outputs beginning with port C and ending with port A. The value $FF followed by $00 is successively written out of each port in output mode and then read back on each port in input mode. Data read back is checked by the CRC.

- **RAM Test**

  This routine tests RAM with several bit patterns. The first pattern is a walking '1' to find address decode problems. This is followed by two complementary checkerboard patterns ($55/$AA and $AA/$55). Each pattern is read back and checked by the CRC. The RAM is cleared after the third pattern test.

- **Timer Test**

  This routine tests the timer functions in both the input and output mode of operation. All bits of the Timer Status Control Register (TSCR) are exercised.

  In the output mode the timer counts down from $FF to $00. The TIMER pin is monitored to make sure it does not change from logic "1" to logic "0" until a timer underflow occurs.

  In the input mode a low to high transition is forced on the TIMER pin to clock the prescaler. The same operation is repeated to check each prescaler tap. On the MC68HC04 chip, additional tests are performed for the "STOP" and "WAIT" instructions, and also the timer input gated modes.

- **Data Space ROM**

  Each byte of the data space ($00 — $FF) is circulated through the CRC registers by adding it to the accumulator. If any byte is improperly fetched, the final CRC result will be incorrect. This includes unimplemented data space bytes which are equal to $FF.

- **Parametric Test (Electrical)**

  After the self-test program has verified the functionality of the MCU, it enables parametric testing of the port pins and TIMER pin. By causing an IRQ all ports and the TIMER pin are switched to outputs and forced high. Subsequent IRQs are then used to toggle these pins alternately low and high, allowing the source and sink capabilities of the output buffers to be measured.

The interrupt service routine is entered several times during the self-test program. The ALU and Accumulator as well as the index registers X and Y are used extensively throughout the program. All instructions and addressing modes are exercised.

As self-test is run, the result of each test is displayed on the port A LEDs. If the part operates properly, the LEDs will flicker briefly then settle to the final state which indicates the signature analysis verified state. Any other stable LED state indicates a failure in the next section of self-test. Figure 5-2 shows the overall test algorithm as well as the LED results after each step.

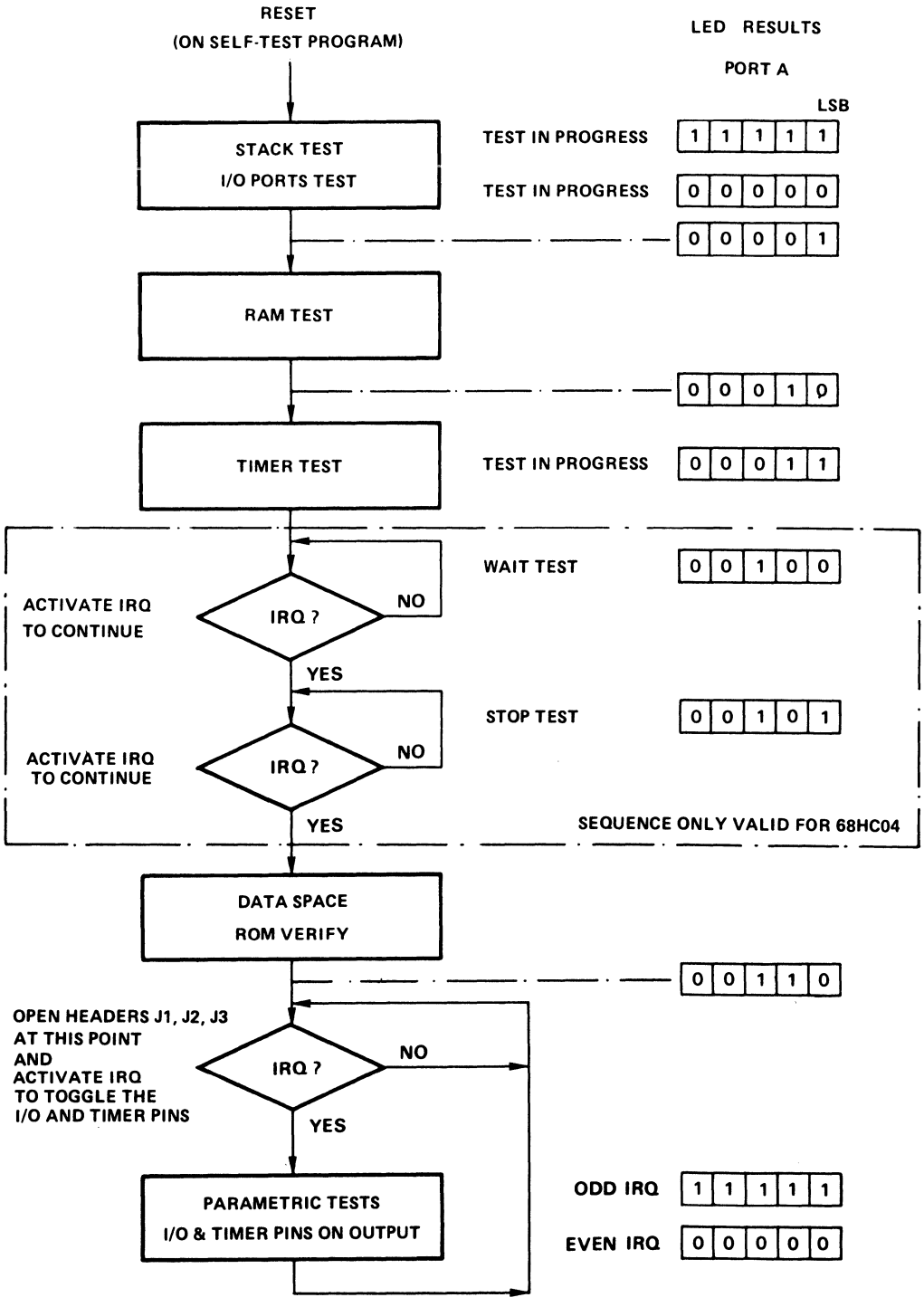


Figure 5.2. Self-Test Flowchart

NOTE:
Four CRC values are checked during execution of the self-test program. These values are determined by running the self-test on the SIM6804/SIMHC04 software simulators and stored in program space. The first three depend on the self-test program only, while the fourth depends on the data space ROM content (user dependent). The CRC value that the self-test program uses to validate the data space must be recalculated for each new data space ROM pattern.

5.3 ROM VERIFY MODE

The ROM verify mode is used to check, by means of signature analysis, the contents of the program space ROM of the M6804. In this mode, the processor (under hardware control only) executes a PC incrementation sequence from the start address $000 to end address $FFF and the ROM contents are routed to the CRC check circuitry via the X-BUS.
To initiate the ROM verify mode the MDS, PA7, PA6 pins are held high during a $\overline{\text{RESET}}$ low to high transition. A reset action in ROM verify mode presets the CRC circuit to a known precalculated value (named CRC seed value) and forces $000 into the PC. Upon exit from reset, the PC increments and performs a CRC check on each byte addressed. After the last byte ($FFF) is checked, the contents of the CRC registers are moved out of the chip and should have the value of $AA and $55 - if the program space ROM is properly programmed. The circuit shown in Figure 5.3 will turn on the lower LED ("ROM good") if the CRC high and low bytes have complementary values.

NOTE:
— Unimplemented program space ROM locations are also tested and should read as $00.
— The initial CRC value (seed) must be recalculated for each new program or data space ROM contents.

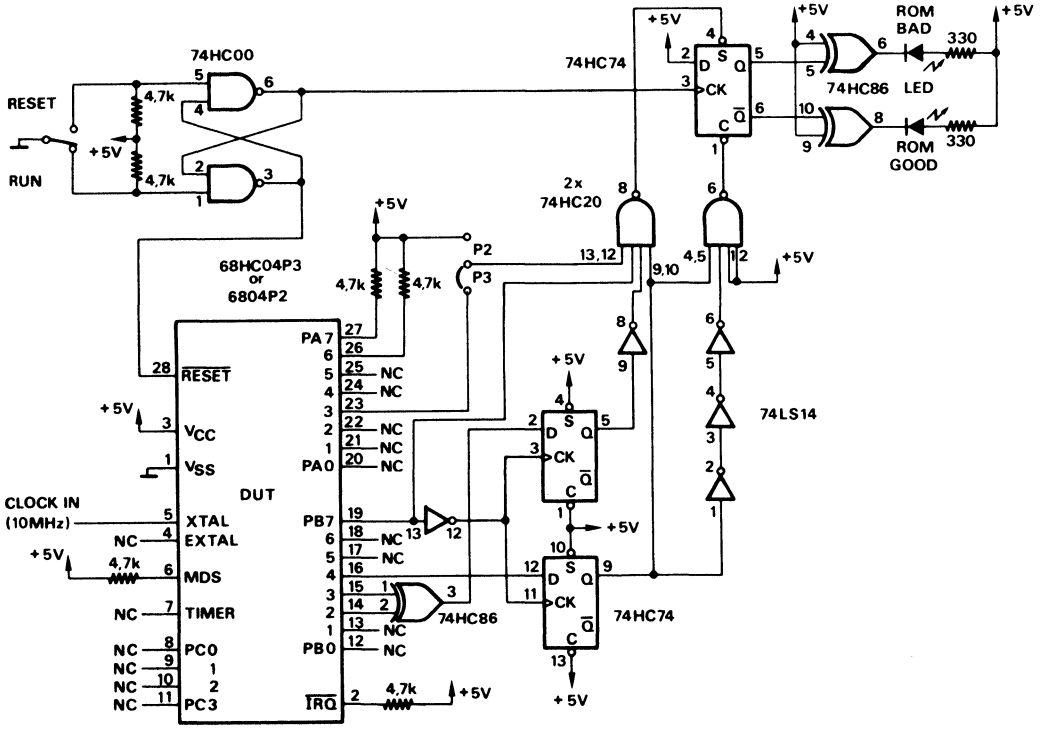


Figure 5.3. ROM Verify Circuit

Application Ideas and Hints

6.1 INTRODUCTION

The low cost of the M6804, coupled with the inherent flexibility offered by an MCU approach, makes it suitable for almost any type of application where some form of logical control is required.
In many cases it offers a cost effective alternative to a dedicated hardware circuit based around CMOS or TTL logic. The exceptionally low power requirements of the CMOS 68HC04 further extend the family versatility, finding application in automotive, telecommunication and battery powered equipments.
This versatility is illustrated in the application examples which follow.

6.2 HARDWARE EXAMPLES

6.2.1 TV Synthesiser (Fig 6.1)

In this example the MC6804P2 is used to form the heart of a low cost tuning and control system for colour or monochrome TV.
The UAA 4800 (available in 1985) is a frequency synthesiser designed to cover the VHF and UHF bands up to 1 GHz. It contains a ÷ 8 prescaler, reference divider, 15 bit programmable divider, phase comparator, filter amplifier, and 4 open collector band switches. The MC 6804P2 supplies control information to the IC in serial form via a 2 wire bus consisting of data and clock signals. To synthesise a particular frequency it is necessary to shift in a 15 bit word to the programmable divider corresponding to the divide ratio for that frequency. It is thus possible to synthesise a large range of nominal channel frequencies by

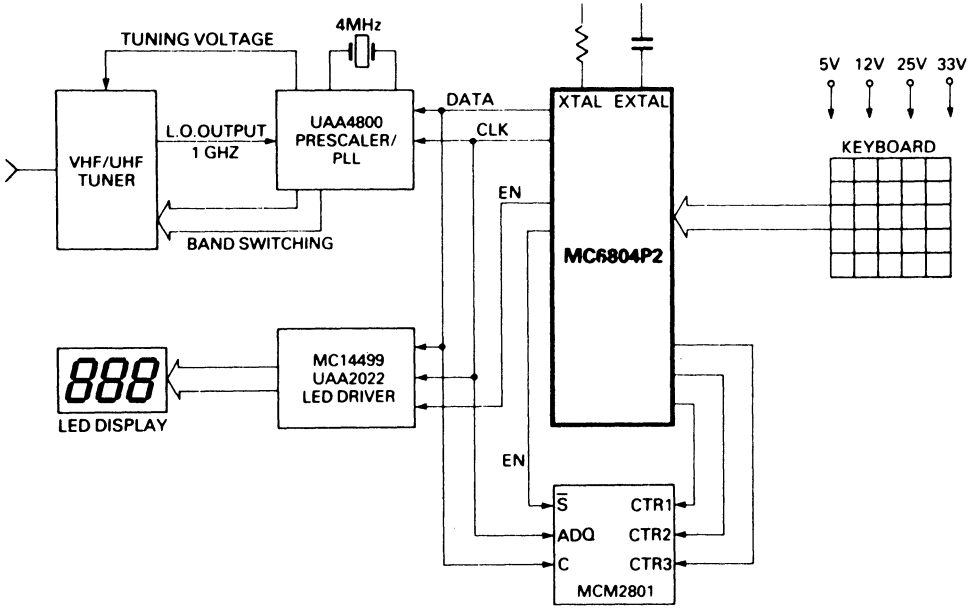


Figure 6.1. Manual TV Synthesiser

altering this 15 bit word according to an algorithm or look-up table stored in the MCU ROM.
User interface is through a local keyboard, which is scanned and read directly by the MCU ports. In this example 12 I/0 lines are available for this purpose, allowing up to 36 commands arranged as a 6 × 6 matrix.
Once the user has selected a particular channel, he has the possibility to store it under a program number for future recall in the MCM 2801. This is a non-volatile memory, arranged as 16 × 16 bits, which retains data even when the TV is powered down. The MC 6804P2 is responsible for reading and writing of the necessary data, i.e. the programmable divider ratio and band information, and handles all interface requirements of the memory. Up to 16 programs can be stored in one MCM 2801.
Another function handled by the MCU is to send control data to an LED driver circuit for display of channel or program number. The UAA 2022 is a 16 segment driver for up to 2 digits, while the MC 14499 will handle up to 4 digits. In both cases data is clocked serially into the chip via a 3 wire bus (data, clock, enable), although the data and clock lines can be shared between all the peripherals to maximise I/0 useage.
Since many of the system features are software dependent, each manufacturer can customise it according to his wishes — e.g. in the method of program storage, channel search, display, etc. In this way he gets the advantages of low cost and low component count but without sacrificing the individuality of design which may be essential to the success of the final product.

6.2.2 AM/FM Radio Synthesiser (Fig. 6.2)

The MC 145157 is a digitally controlled frequency synthesiser using PLL techniques. It contains a 14 bit reference divider, a 14 bit frequency divider, two phase detectors and a lock detector. Silicon-gate CMOS technology allows low power and high frequency operation, with a guaranteed f_{IN} of at least 22 MHz @ 5V, 25°C.
In this application it is used in conjunction with the MC 68HC04P3 to make a low power digital tuning and display system for use in radio, especially portables and car radios where power consumption is important. Information for either the reference divider or frequency divider ratio is transferred in serial form to shift registers on the MC 145157 via the Data and Clock inputs, with an extra bit added to determine which of the dividers is selected. Each low-to-high transition of the clock shifts one bit of data. Once the MCU has transferred 15 bits, it must pull the LE (Latch Enable) pin high to latch the data into the appropriate divider.
For AM coverage, the output from the tuner local oscillator is coupled directly into the frequency divider input of the MC 145157.
For FM a prescaler (÷ P) is required, in this case a ÷ 20, the MC 3396. The phase comparator outputs ØR and ØV provide loop error signals which are integrated and amplified in the MC 1458, and the resultant voltage is applied to the local oscillator varicap to control the tuning point. Alternatively a second phase detector output is available on PDout, in the form of a tristate loop error signal. The LD (Lock detect) pin goes high when the loop is in lock, and could be used to drive a visual indicator.
The reference divider ratio (÷ R) is selected according to the band coverage and step size required. For example, with a 4 MHz reference, a ratio of 4000 will produce a step size of 1 KHz and maximum frequency of 16 MHz, suitable for AM. For FM, a ratio of 3200 will produce a step size of 25 KHz and maximum frequency of 400 MHz. Other values are possible by altering either R,P or the reference oscillator.
Specific stations can be memorised in the MC 68HC04P3 RAM by storing the frequency divider ratio (15 bits) corresponding to that station. With 124 bytes available it is possible to store 30 stations or more. During power down this information is preserved by putting the MCU into "STOP" mode, which requires very little supply current, small enough to give long life with even small capacity batteries.
Frequency display is made with an MC 144117 driving a 4 digit LCD display. Data for display is shifted serially into this circuit from the MC 68HC04P3 via Data, Clock and Enable lines.
User commands are given through a local keyboard which can be scanned and read directly from the MCU ports. Since all control functions such as search, station memorise, station recall, etc. are a function of software routines, there is considerable scope to give individuality to the final product. The 1.7K ROM of the MC 68HC04P3 is large enough to allow a high degree of sophistication in the product, but the overall system cost is low enough for mass market use.

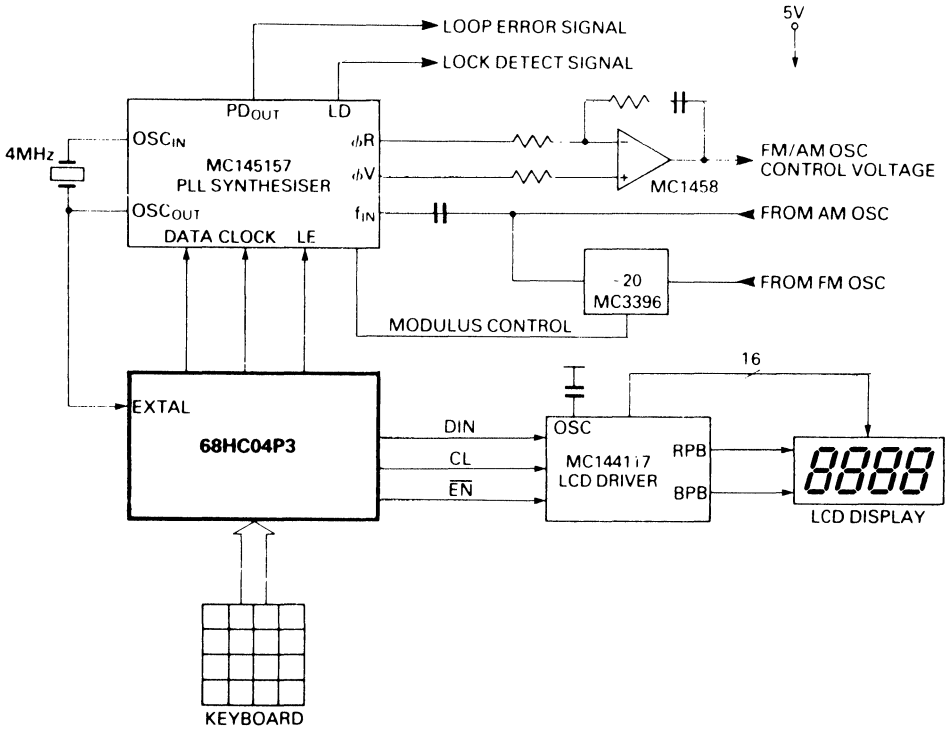


Figure 6.2. AM/FM Radio Synthesiser

6

6.2.3 Multiplex Wiring Corner Outstation (Fig. 6.3)

Multiplex wiring in a car is a system which uses serial communications with local decoding and load switching to replace the conventional wiring system of individual wires from each switch to each load. Using a multiplex system results in a less complicated wiring harness with less connectors (hence improved reliability), reduced weight and the ability to feed back more diagnostic information to the driver and garage mechanic. The reason that such systems are not yet a production reality is simply a question of economics, in that the electronics involved have been more expensive than the savings resulting from the reduction in complexity of the wiring harness.

The MC 68HC04P2 is ideal for this application, combining, as it does, low cost and low power dissipation with a powerful and efficient instruction set.

In the implementation shown in the diagram the TCF5600 universal microprocessor power supply part is used to supply the MC 68HC04P2 with 5V and a reset signal and also includes a watchdog and a low voltage inhibit facility. Additionally the TCF5600 provides a 30V supply to drive the gates of the n-channel TMOS load switches so that they can be used as high side (i.e. between battery and load) switches which is more efficient than using p-channel TMOS parts.

The 20 I/O pins of the MC 68HC04P2 make it possible to drive up to seven loads, assuming that full load status sensing is implemented and that address selection is by wire links on the board (so that the same program can be used for all locations), as shown in the diagram. Variation of the load status sensing strategy, or putting the outstation address in the data space ROM will make more I/O lines available for additional load control or for other purposes such as control of a simple A/D converter to provide an interface with a local sensor.

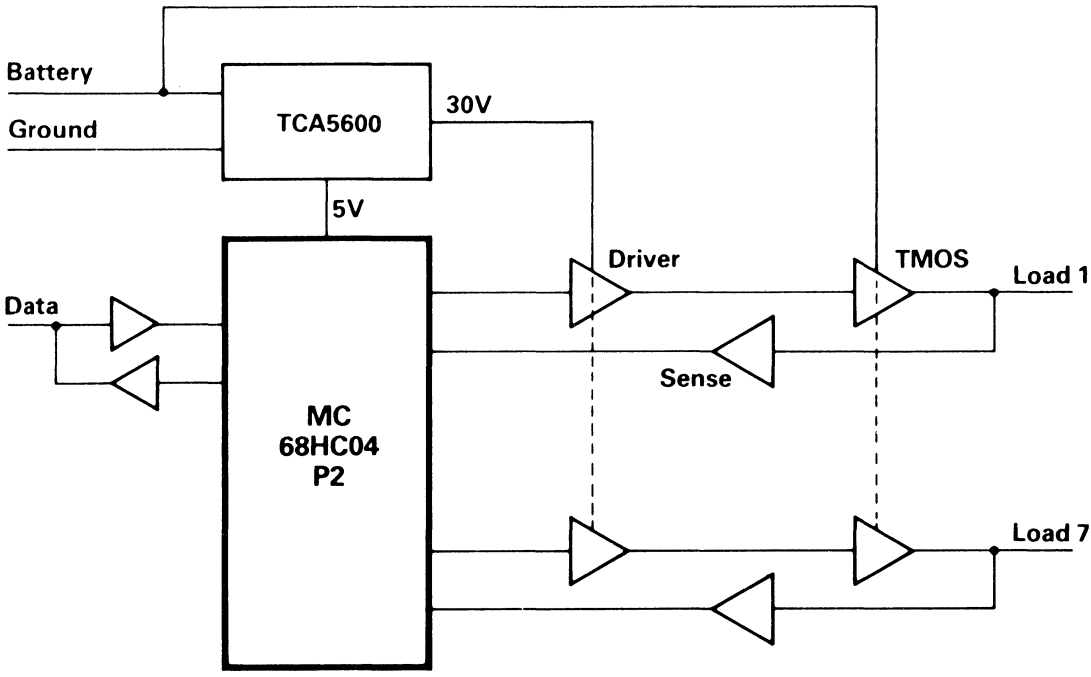


Figure 6.3. Multiplex Wiring Corner Outstation

6

Use of the bit manipulation instructions makes the task of decoding and encoding the serial information on the data bus simple to do under software control without the requirement for any special hardware, other than buffering. This means that the communications protocol is totally flexible and under the system designers control, with no concessions having to be made due to the inflexibility of a hardware based serial communication scheme.

The availability of an HCMOS version of the microcomputer is particularly beneficial in this application as it is necessary to cater for the conditions that occur with the car switched off. In this situation it is necessary to minimise the drain on the battery to ensure that the car will still start after being left for a period of some weeks (whilst the owner is on holiday, for instance). It is also necessary, however, that the system should remain active during this period so that those loads that can be switched on, such as hazard flashers, parking lights and courtesy lights, can be controlled by the multiplex system.

6.2.4 Cruise Control (Fig. 6.4)

Cruise control is a system whereby the speed of a car is maintained at a fixed value, determined by the driver, without the driver having to touch the accelerator. The driver has three or four direct controls and one or two indirect controls.

The direct controls are:

1. Enable. This is simply an on/off control.
2. Accel/set. This control has two functions. If actuated momentarily the control unit memorises current road speed as the controlled speed. If actuated continuously the car is accelerated until the control is released, when the speed at that point is memorised.
3. Resume. This control causes the system to resume control at the last memorised speed after it has been de-activated by one of the indirect control switches.
4. Decel/set. (Optional) This control functions in the same way as the Accel/set, but decelerates the car when continuously operated.

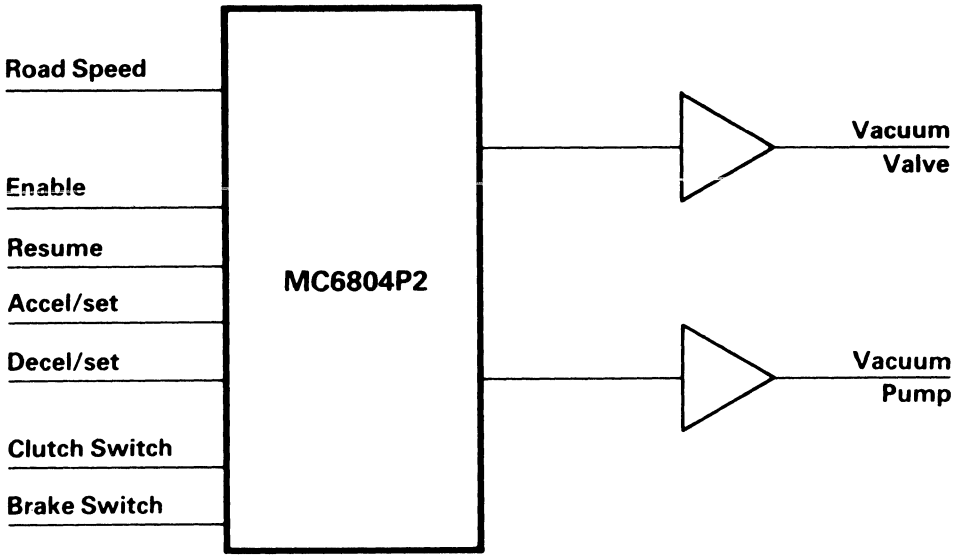


Figure 6.4. Cruise Control

The indirect controls consist of switches on the brake and clutch (if there is one) pedals that cause the system to de-activate immediately either of the pedals is touched.
The system controls speed by controlling the position of an actuator that operates on the accelerator linkage, usually at the engine end, which is arranged in such a way as to not interfere with the normal operation of the linkage.
The MC 6804J2 is ideal for this application as it is small and cheap, but has a powerful and efficient instruction set.
The limited I/O available on the 20 pin MC 6804J2 is more than adequate for this task, as can be seen from the diagram. The scheme chosen for this implementation is to use a vacuum actuator to position the throttle with a vacuum pump to control it in one direction (acceleration) and a vacuum dump valve to control it in the other. The only addition that could be required is a second, larger vacuum dump valve if the requirements of immediate deactivation on operation of the brake or clutch switches cannot be met by the valve used for control. There are a variety of other actuators available for controlling the throttle position, but three output lines are adequate for all of them. Some control schemes call for the throttle position to be measured, but the MC 6804J2 has adequate spare I/O lines to control a simple off board A/D converter.
The particular advantage of using the MC 6804J2 in this application is that it is possible to use a more sophisticated control algorithm than it is with the custom circuit or standard gate approach that is normally used, while still remaining cost competitive. The bit manipulation instructions make the device especially easy to use in this application where individual port lines are assigned specific tasks.

6.2.5 Electronic Telephones (Figs 6.5, 6.6)

The HCMOS MC 68HC04P3 in these applications controls the extra features and facilities of an all electronic telephone. HCMOS allows all of the extra features to be powered from the telephone line, and the 124 bytes of static RAM provide sufficient storage for about 10 telephone numbers either with battery back-up or by drawing a few microamps from the line in the on-hook condition.
In the example shown in Figure 6-5, the MC 68HC04P3 interfaces with the MC 34010 Electronic Telephone Chip (ETC) via six pins. Keyboard data is input to the MC 34010 DTMF Dialler circuit and output to the MCU serially on I/O, clocked by CL. DD determines the data direction. Hence numbers can be stored in the MCU memory to be subsequently used for automatic dialling, whereby the numbers are trans-

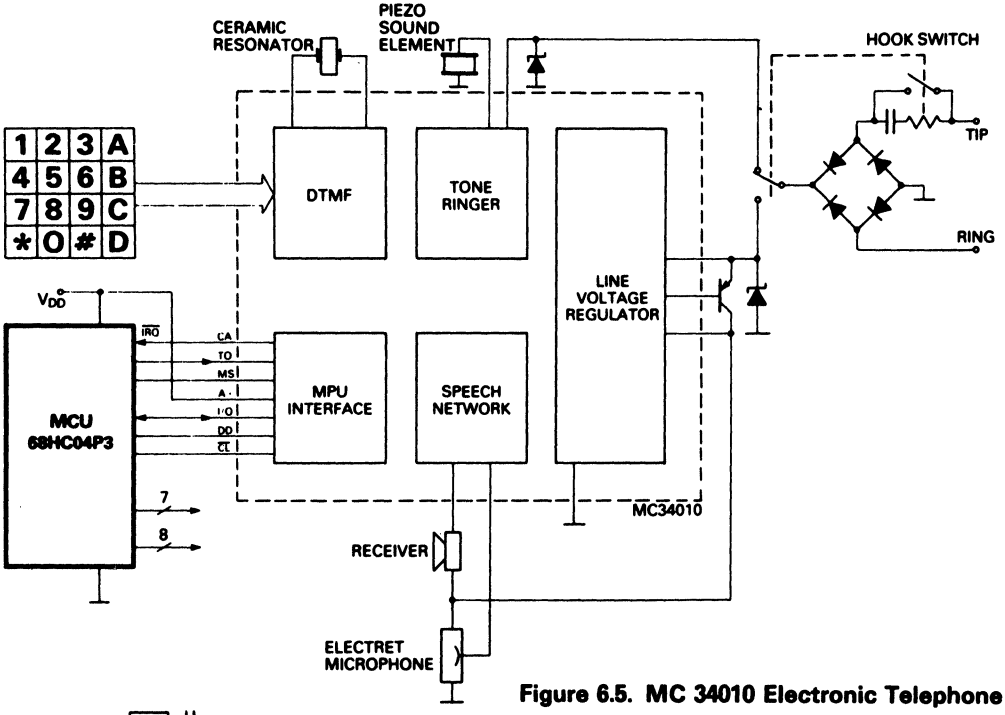


Figure 6.5. MC 34010 Electronic Telephone

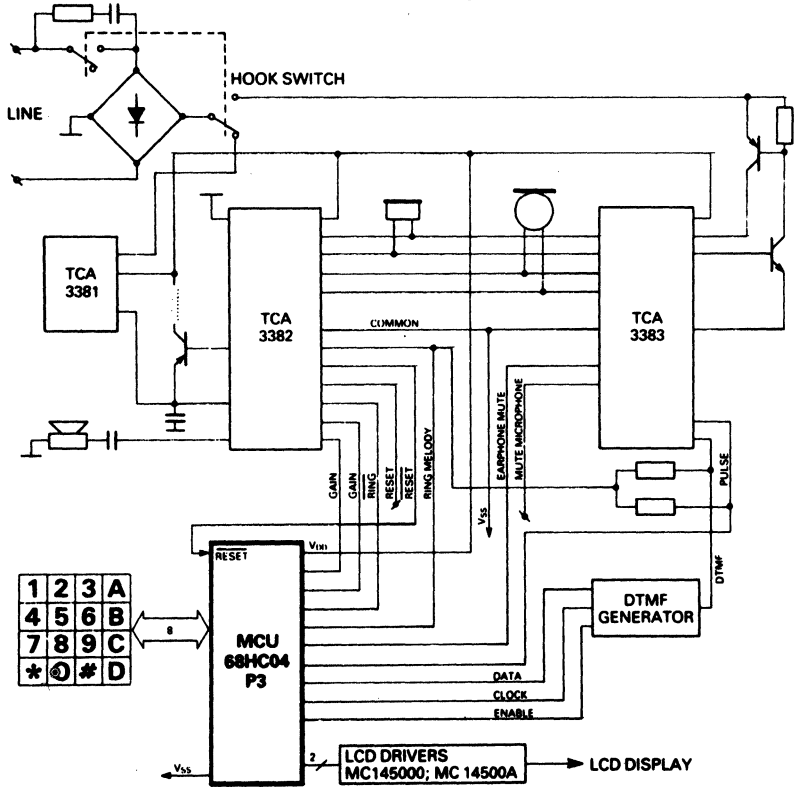


Figure 6.6. Line Powered Feature Telephone

ferred in the opposite direction. When valid data has been transferred to the ETC, TO enables the DTMF generator. The DP output indicates (logic 1) that one and only one key is depressed thereby signalling the MPU that data is available, for instance to initiate a data transfer to the MCU. Sufficient I/O is vacant on the 68HC04P3 to provide other facilities such as display interface, extra memory interface for a large repertory, or for interface to a small office workstation.

The second application shown in Figure 6-6 is a Feature Telephone using TCA 3381/2/3, which meets the transmission requirements for the French PTT network. The TCA 3383 provides the basic transmission 2/4 wire conversion. TCA 3382 provides loudspeaker drive for a monitor, and also for the Ring tone. In the on-hook condition power is derived from a received ring signal via TCA 3381, and powers up TCA 3382 and the MC 68HC04P3. The MCU generates a Ring tone which is amplified by TCA 3382. A variety of tones, or even melodies can thus be programmed to replace the old telephone bell. In the off-hook condition, pulse dialling is timed by the MC 68HC04, and the received line signal can be monitored on the loudspeaker. The gain at the loudspeaker is controlled by the two logic "gain" inputs to the TCA 3382 so that the user can set the volume he requires through the MCU keyboard.

6.2.6 Professional Drill Control (Fig. 6.7)

Electronic motor control offers many advantages over traditional electro-mechanical methods — it gives greater stability of motor speed, independence of load and line variations, overload protection, energy efficiency, etc. The principle consists of varying the firing angle of a triac in series with a Universal motor driven from the AC mains, thereby varying the amount of energy per cycle delivered to the motor.

In this application the HCMOS MC 68HC04P2 is used as a low cost controller for a drilling machine. Its low current consumption, and wide operating voltage range, allow a considerable reduction in both cost and size of the power supply (no transformer is needed) compared with HMOS.

The MCU's primary function is to fire the triac during each half cycle of the mains through one of its I/O lines.

A phase reference is provided by the zero crossing detector (ZCD) circuit which gives a short pulse to the $\overline{IRQ}$ pin at the beginning of each half cycle, i.e. every 10 ms with 50 Hz mains. The internal timer is then used to determine how soon after each pulse the triac is triggered, and hence the power delivered to the motor. The flexibility of the M6804 timer is particularly useful here.

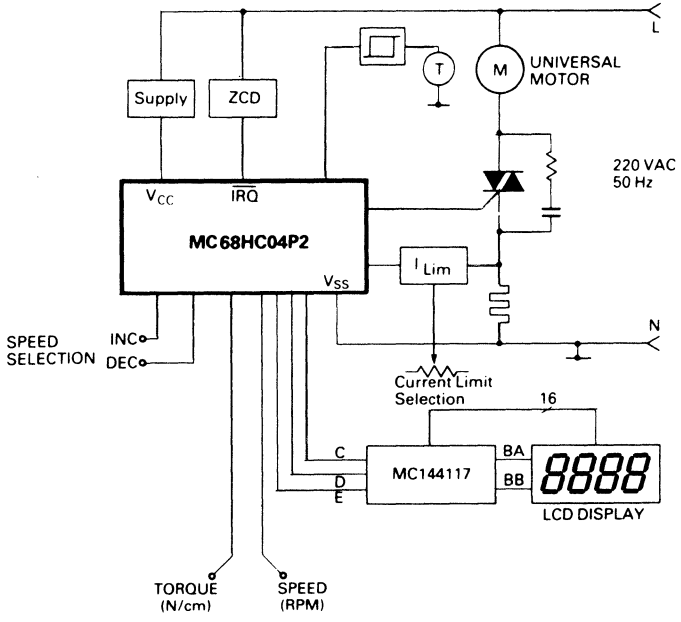


Figure 6.7. Professional Drill Control

In order to control the motor speed, it is of course necessary to measure it, and this is achieved by a tachogenerator on the motor, which outputs a sine wave whose frequency and amplitude increases with speed. This signal must be amplified, clamped, and fed through a Schmitt trigger to produce a digital square wave. It is then input to the MC 68HC04P2, where the motor speed is calculated by counting the frequency of incoming pulses, again using the timer function. Software algorithms in the MCU use this information to maintain a constant speed, irrespective of any variations in load torque or mains supply, by suitably altering the triac firing angle.
Speed selection could be achieved using a potentiometer, but this would require an external A/D circuit. Since visual feedback to the user in this application is provided by an LCD display, it is much simpler and cheaper to use two switches connected to I/O lines, as shown, one for increasing motor speed (INC) and one for decreasing (DEC). The rate of change is a function of the software, and could be made non-linear if required. Information for the LCD is transferred from the MCU to the display driver MC 144117 in serial form by three lines-Data, Clock, Enable. This information can be either motor speed or torque, selectable through an I/O option. The torque figure is calculated by an algorithm from the motor speed and triac current.
A further refinement of this application is triac protection. A low value series resistor detects the triac current, and the current limit circuit compares it with an adjustable reference corresponding to the maximum current allowable. If this reference is exceeded the MCU will reduce the firing angle accordingly.

6.3 SOFTWARE EXAMPLES

This section contains some example programs illustrating techniques which can be useful in developing M6804 software generally. The reader is advised to consult the Motorola Macro Assembler reference manual for more details concerning assembly language syntax and assembler directives.

6.3.1 Initialisation Routine

The first step in any software program is the initialisation of the MCU. Fig. 6.8 shows a possible routine for the M6804 in which the use of the RESET and INTERRUPT vector is emphasised. Note also the use of the MVI instruction, rather than LDA and STA, to load data directly into the appropriate data direction registers in order to define the I/O states.

```
00001                   *
00002                   *
00003                   ********************************************************************************
00004                   *
00005                   *       INITIALISATION ROUTINE
00006                   *
00007                   ********************************************************************************
00008                   *
00009                   *
00010                   *               This routine illustrates an initialisation procedure for the
00011                   * MC68HC04. The reset vector JMPs to a JSR instruction which carries out
00012                   * initialisation of the ports, timer, DDRs and clears all the RAM
00013                   * locations.This routine is terminated by an RTI instruction and not
00014                   * by RTS to ensure the interrupt mask is cleared.
00015                   *               After the initialisation routine has been executed the main
00016                   * program is executed.
00017                   *               A detailed description of the actual initialisation is given
00018                   * as follows.
00019                   *       1 - The ports are initialised as required by the hardware
00020                   *           configuration.
00021                   *       2 - All RAM is cleared.
00022                   *       3 - The starting values are loaded into the timer registers.
```

Figure 6.8. Example of an Initialisation Routine

```
00024                    ********************************************************************************
00025                    *
00026                    *       EQUATES
00027                    *
00028                    ********************************************************************************
00029                    *
00030                    *
00031                0000 PORTA   EQU     $00       address of PORTA
00032                0001 PORTB   EQU     $01       address of PORTB
00033                0002 PORTC   EQU     $02       address of PORTC
00034                0004 DDR     EQU     $04       data direction register offset
00035                0080 XREG    EQU     $80       X - register address
00036                0081 YREG    EQU     $81       Y - register address
00037                0009 TSCR    EQU     $09       address of TIMER CONTROL/STATUS REGISTER
00038                00FD PRESCL  EQU     $FD       address of TIMER PRESCALER
00039                00FE TDR     EQU     $FE       address of TIMER COUNT (DATA) REGISTER
00040                00FF ACC     EQU     $FF       address of ACCUMULATOR
00041
00042
00043                    ********************************************************************************
00044                    *
00045                    * MAIN PROGRAM : Device is initialised and then the main program
00046                    *                is executed
00047                    *
00048                    ********************************************************************************
00049                    *
00050                    *
00052A 0C00                       ORG     $C00

00054A 0C00 8C 02       A INIT1  JSR     INIT2     Reset vector points here.

00056                    *       MAIN PROGRAM
00057                    *       MAIN PROGRAM
00058                    *       MAIN PROGRAM
00059                    *       MAIN PROGRAM
00060                    *       MAIN PROGRAM
00061                    *
00062                    *
00063                    ********************************************************************************
00064                    *
00065                    * INIT2 : Initialisation routine for the MC68HC04
00066                    *
00067                    ********************************************************************************

00069                    * INITIALISE PORTS, DDRs, TIMER ------------------------------------

00071A 0C02 B0 00 00    A INIT2  MVI     PORTA,$00 clear PORTA
00072A 0C05 B0 04 F0    A        MVI     PORTA+DDR,$F0 Hi nibble=output,Lo nibble=input
00073A 0C08 B0 01 01    A        MVI     PORTB,$01 PORTB bit 0=1
00074A 0C0B B0 05 FF    A        MVI     PORTB+DDR,$FF PORTB all outputs
00075A 0C0E B0 02 0F    A        MVI     PORTC,$0F set allbits high
00076A 0C11 B0 06 0F    A        MVI     PORTC+DDR,$0F all bits as output

00078                    * CLEAR ALL RAM LOCATIONS ----------------------------------------

00080A 0C14 B0 81 82    A        LDXI    #$82      Start address of RAM

00082A 0C17 FB FF       A CLRAM  CLRA              Clear A
00083A 0C19 E1                   STA     [X]       Clear RAM location
00084A 0C1A A8                   INX                Increment RAM location pointer
00085A 0C1B AC                   TXA                Let A = RAM pointer
00086A 0C1C EB FC       A        SUB     #$FC      Subtract $FC (last RAM address)
00087A 0C1E 18       0C17        BNE     CLRAM     Branch until all Ram is cleared

00089                    * INITIALISE TIMER REGISTERS -------------------------------------

00091A 0C1F B0 09 2F    A        MVI     TSCR,$2F $2F=x00101111 - /128, start timer
00092A 0C22 B0 FE FF    A        MVI     TDR,$FF   load timer count register
00093A 0C25 B2                   RTI                return to main program
```

Figure 6.8. (cont.)

```
00095                    ********************************************************************************
00096                    *
00097                    * INTER - Interrupt routine
00098                    *
00099                    ********************************************************************************
00100                    *
00101                    *
00102A 0C26 20            INTER  NOP
00103A 0C27 20                   NOP
00104A 0C28 20                   NOP
00105A 0C29 B2                   RTI
00106A 0FFC                      ORG    $FFC

00108                    ********************************************************************************

00110A 0FFC 9C 26       A INTR   JMP    INTER    Interrupt vector

00112                    ********************************************************************************

00114                    ********************************************************************************

00116A 0FFE 9C 00       A RESET0 JMP    INIT1    Reset vector

00118                    ********************************************************************************
```

Figure 6.8. (cont.)

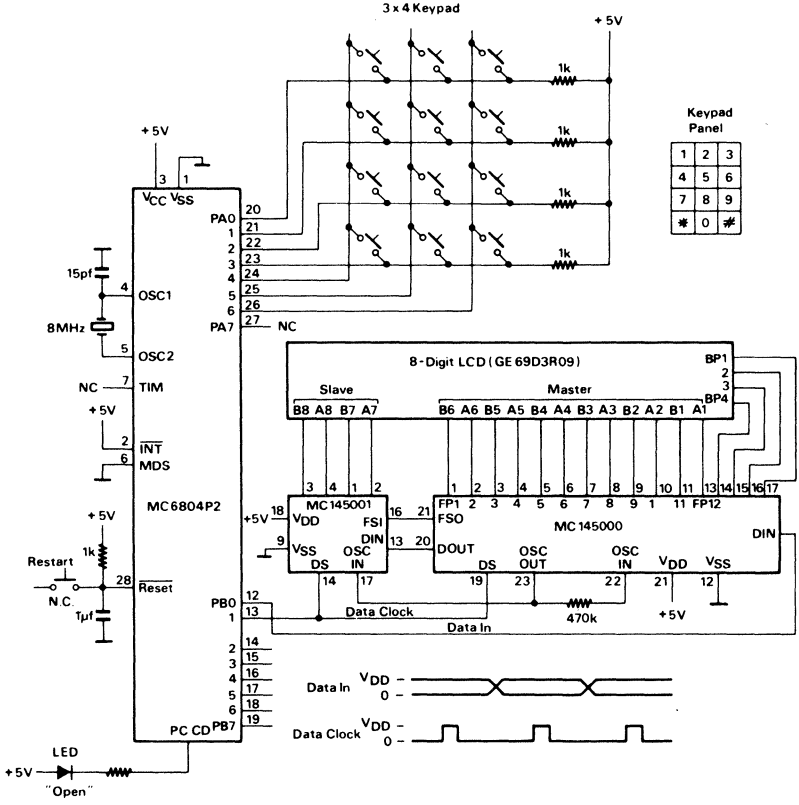


Figure 6.9. Keyboard Encoder/LCD Driver Schematic

6.3.2. Keyboard Encode Routine

A common task for control orientated MCUs is to scan and encode a keyboard matrix like the 4 x 3 matrix shown in Fig. 6.9. The routine shown in Fig. 6.10 will scan the 12 keys and detect a switch closure. It then debounces the switch closure by rechecking it after a small delay, and also checks that multiple keys were not depressed simultaneously. If all is well with the switch closure, its position in the keyboard matrix is decoded and the result made available in the accumulator for further processing.

For low power applications this routine could be used with one of the HCMOS members of the M6804 family. In this case the keyboard matrix hardware can be arranged so that any switch closure pulls the $\overline{IRQ}$ pin low, causing selection of the interrupt vector, which should point to the keyboard encode routine. When processing of this, and any consequent routines, is finished, the STOP instruction can then be used to power down the MC 68HC04 until the next switch closure occurs, causing it to wake up.

```
00120                    *
00121                    *
00122                    ********************************************************************************
00123                    *
00124                    * KEYPAD SCAN ROUTINE
00125                    *
00126                    ********************************************************************************
00127                    *
00128                    *
00129                    *       This routine scans the entire keypad once and returns ( in ACC )
00130                    *       the number of the key that was hit. If an invalid/no key was hit,
00131                    *       it returns with ACC=$FF. This routine uses a negative keyboard
00132                    *       scan, with a column being selected by a 0 in porta.
00133                    *       In porta, bits 0-3 are row inputs, bits 4-6 are column outputs,
00134                    *       and bit 7 is not connected.
00135                    *
00136                    ********************************************************************************
00137                    *
00138                    *
00139                 008F COL    EQU     $8F       Keypad column counter
00140                 0090 TEMPA  EQU     $90       temporary storage for accumulator
00141                    *
00142                    *
00143A 0C50                         ORG     $C50
00144                    *
00145                    *
00146A 0C50 8C 57       A LP7       JSR     SCAN      try to scan keypad
00147A 0C52 EC FF       A           CMP     #$FF      invalid/no key ?
00148A 0C54 3B       0C50           BEQ     LP7       try again

00150A 0C55 9C AE       A           JMP     NEXT      execute according to the key hit

00152A 0C57 B0 00 EF    A SCAN      MVI     PORTA,#%11101111 activate column 1
00153A 0C5A B0 8F 01    A           MVI     COL,#1    initialize COL COUNTER to 1
00154A 0C5D F8 00       A LP2       LDA     PORTA     check keys in this column
00155A 0C5F F9 90       A           STA     TEMPA     save initial reading
00156A 0C61 ED 0F       A           ANDA    #$0F      check input pins
00157A 0C63 EC 0F       A           CMP     #$0F      if no key pressed..
00158A 0C65 2C       0C72           BEQ     NOKEY     ...continue scan..
00159A 0C66 B0 81 1E    A           LDYI    #30       otherwise debounce key..
00160A 0C69 8C AF       A           JSR     DEBNCE    ...for about 50 ms
00161A 0C6B F8 90       A           LDA     TEMPA     get initial reading back
00162A 0C6D FC 00       A           CMP     PORTA     compare with current reading
00163A 0C6F 02       0C72           BNE     NOKEY     if not the same, ignore and
00164                    *                             continue scan
00165A 0C70 9C 86       A           JMP     KEYHIT    otherwise compute key number
00166                    *
00167A 0C72 FE 8F       A NOKEY     INC     COL       inc COL count
00168A 0C74 E8 04       A           LDA     #4
00169A 0C76 FC 8F       A           CMP     COL       finished scan?
00170A 0C78 02       0C7B           BNE     C3        if not, activate next column and scan
```

Figure 6.10. Keyboard Encoder Listing

```
00171                          *                       again
00172A 0C79 9C AB        A     JMP     EXIT1           if finished go back to main program
00173                          *
00174A 0C7B F8 90        A C3  LDA     TEMPA           upper nibble has previous COL activation
00175A 0C7D ED F0        A     ANDA    #$F0            clear out lower nibble before rotate
00176A 0C7F EA 0F        A     ADDA    #$0F            set to rotate a 1 into upper nibble
00177A 0C81 B5                 ROLA                    activate next column
00178A 0C82 F9 00        A     STA     PORTA
00179A 0C84 9C 5D        A     JMP     LP2             continue scan
00180                          *
00181                          * Compute key
00182                          *
00183A 0C86 F8 90        A KEYHIT LDA    TEMPA           get back initial reading
00184A 0C88 B5                 ROLA                    move up the...
00185A 0C89 B5                 ROLA                    lower nibble..
00186A 0C8A B5                 ROLA                    to check...
00187A 0C8B B5                 ROLA                    which key was pushed.
00188A 0C8C ED F0        A     ANDA    #$F0            clear lower nibble
00189A 0C8E EA 0F        A     ADDA    #$0F            set lower nibble to all 1's
00190                          *
00191A 0C90 B0 81 09     A     LDXI    #9              see explanation below
00192                          *
00193                          * Because we can only rotate to the left, we have to check the
00194                          * bottom row, row 4, first. Reg X will hold a number which ,
00195                          * when added to COL ( the column count ), will give us the key's
00196                          * number. Reg X initially holds 9 because it is the last number in
00197                          * row 3. If a key on row 4 was hit, then adding COL to reg X will
00198                          * give us the correct key number. If the key is not on this row we
00199                          * decrement X by 3 and check the 3rd row. This procedure continues
00200                          * until we determine on which row the key-stroke was made. Then
00201                          * COL is added to X and we have our number.
00202                          *
00203A 0C93 B5            LP5   ROLA                    rotate bit into carry bit
00204A 0C94 45       0C9A       BCC     CHECK2          if row is found, check for one key pushed
00205A 0C95 B8                  DECX                    subtract to get to next higher row on keypad
00206A 0C96 B8                  DECX
00207A 0C97 B8                  DECX
00208A 0C98 9C 93        A      JMP     LP5             continue row check
00209                          *
00210                          * Check that only one key was initially pushed
00211                          *
00212A 0C9A ED F0        A CHECK2 ANDA   #$F0            clear out lower nibble
00213A 0C9C EC F0        A      CMP     #$F0            upper nibble should be all 1's
00214A 0C9E 0C       0CAB       BNE     EXIT1           if not then bad keystroke
00215A 0C9F AC                  TXA
00216A 0CA0 FA 8F        A      ADDA    COL             compute key number
00217A 0CA2 EC 0A        A      CMP     #10             is it >=10?
00218A 0CA4 65       0CAA       BLO     EXIT2           no, return with a=key number
00219A 0CA5 EC 0B        A      CMP     #11             yes, but is it the "0" key?
00220A 0CA7 03       0CAB       BNE     EXIT1           if not then invalid key
00221A 0CA8 FB FF        A      CLRA                    adjust for "0" key
00222                          *
00223A 0CAA B3           EXIT2  RTS
00224                          *
00225                          * Exit point for invalid/no keypad entry
00226                          *
00227A 0CAB E8 FF        A EXIT1  LDA    #$FF
00228A 0CAD B3                  RTS
00229                          *
00230A 0CAE B3           NEXT   RTS
00231                          *
00232                          ********************************************************************************
00233                          *
00234                          * DEBNCE - Delay for a period of time ( in the millisecond range)
00235                          *          as specified by Y. Use the prescaler register as a
00236                          *          timer
00237                          *
00238                          ********************************************************************************
```

Figure 6.10. (cont.)

```
00239                       *
00240                       *
00241A 0CAF C7 FD FD 0CAF DEBNCE BRCLR  7,PRESCL,* wait for a high in bit 7 of prescaler
00242A 0CB2 B9                   DECY           decrement on positive edge of bit 7
00243A 0CB3 25       0CB9        BEQ    LP9     if finished return
00244A 0CB4 CF FD FD 0CB4        BRSET  7,PRESCL,* wait for a low in bit 7
00245A 0CB7 9C AF       A        JMP    DEBNCE
00246A 0CB9 B3             LP9   RTS
00247                       *
00248                       *
```

Figure 6.10. (cont.)

6.3.3 LCD Driver (Serial I/O) Routine

In this example the M6804 is used to drive an 8 digit LCD via the MC 145000 and MC 145001, see Fig. 6.9. These are CMOS devices intended for driving liquid crystal displays in a multiplexed-by-four arrangement.

The MC 145001 master driver generates frontplane and backplane waveforms, and is capable of independent operation. The MC 145001 slave generates only frontplane waveforms, and is synchronised with the backplane drive from the master. Data for display is clocked serially into the driver ICs by the MCU, a task made simpler by the bit manipulation instructions of the M6804. Fig. 6.11 shows the routine listing.

```
00250                       *
00251                       *
00252                       ************************************************************************
00253                       *
00254                       * GETLCD - get LCD code for a character in the ACC.
00255                       *
00256                       ************************************************************************
00257                       *
00258                       *
00259                       *        ---a---
00260                       *       |       |
00261                       *       f       b
00262                       *       |       |
00263                       *        ---g---
00264                       *       |       |
00265                       *       e       c   --
00266                       *       |       |  |dp|
00267                       *        ---d---    --
00268                       *
00269                       *
00270A 0020                       ORG    $20      start of data ROM
00271                       *
00272                       *                dp
00273                       *              degficba
00274A 0020    D7       A SEVSEG FCB    %11010111 0
00275A 0021    06       A        FCB    %00000110 1
00276A 0022    E3       A        FCB    %11100011 2
00277A 0023    A7       A        FCB    %10100111 3  0 = segment off
00278A 0024    36       A        FCB    %00110110 4  1 = segment on
00279A 0025    B5       A        FCB    %10110101 5
00280A 0026    F5       A        FCB    %11110101 6
00281A 0027    07       A        FCB    %00000111 7
00282A 0028    F7       A        FCB    %11110111 8
00283A 0029    37       A        FCB    %00110111 9
00284                       *
00285                       *
00286A 002A E1              GETLCD STA    [X]      save value in RAM, BCD value is in ACC
00287A 002B EA 20       A        ADDA   #SEVSEG  offset for LCD numeral pattern
00288A 002D BD                   TAY
```

Figure 6.11. LCD Driver Listing

```
00289A 002E F0                    LDA    [Y]      get LCD pattern for generated number
00290A 002F 80 31       A         JSR    DSPLY    display it
00291                     *
00292                     *
00293                     ************************************************************************
00294                     *
00295                     * DISPLAY ROUTINE - Display one character ( 8 bits ) on the
00296                     *                   8 segment LCD connected to the port B.
00297                     *                   The character is initially stored in ACC.
00298                     *
00299                     ************************************************************************
00300                     *
00301                     *
00302                 008D DLPCTR EQU    $8D      multi-purpose loop counter
00303                     *
00304A 0031 B0 8D 08    A DSPLY  MVI    DLPCTR,#8 set to send 8 bits
00305A 0034 B5            ROTATE ROLA             rotate bits into carry bit
00306A 0035 64        003A        BCS    ONE      bit = 1 or 0?
00307A 0036 D0 01       A         BCLR   0,PORTB  send 0 to bit 0 portb
00308A 0038 90 3C       A         JMP    STRB
00309A 003A D8 01       A ONE     BSET   0,PORTB  send 1nto bit 0 portb
00310                     *
00311A 003C D9 01       A STRB    BSET   1,PORTB  send data strobe, rising edge
00312A 003E D1 01       A         BCLR   1,PORTB  return strobe to logic zero
00313A 0040 FF 8D       A         DEC    DLPCTR   finished?
00314A 0042 11        0034        BNE    ROTATE   if not continue sending bits to LCD
00315A 0043 B3                    RTS
00316                     *
00317                     ************************************************************************
00318                     *
00319                     * CLEAR - Clear entire LCD
00320                     *
00321                     ************************************************************************
00322                     *
00323A 0044 B0 81 08    A CLEAR  LDYI   #8       send 8 blank characters to the LCD
00324A 0047 FB FF       A        CLRA
00325A 0049 B5                   ROLA            clear the carry bit
00326A 004A FB FF       A        CLRA            ACC must be clear
00327A 004C 80 31       A LP1    JSR    DSPLY
00328A 004E B9                   DECY            sent 8 blanks?
00329A 004F 1C        004C       BNE    LP1      if not continue sending
00330A 0050 B3                   RTS
00331                     *
00332                     ************************************************************************
00333                     *
00334                     * DSPLY8 - Display a string of 8 characters. Y points to the beginning
00335                     *          of the string
00336                     *
00337                     ************************************************************************
00338                     *
00339A 0051 B0 81 08    A DSPLY8 LDXI   #8       display 8 characters
00340A 0054 F0            LP8    LDA    [Y]      get start of string
00341A 0055 80 31       A        JSR    DSPLY    display a character
00342A 0057 A9                   INCY            move to the next character
00343A 0058 B8                   DECX            finished?
00344A 0059 1A        0054       BNE    LP8      if not, continue sending characters
00345A 005A B3                   RTS
00346                     *
00347                     *
00348                     *
00349                     *
00350                            END
TOTAL ERRORS 00000--00000
```

Figure 6.11. (cont.)

6.4 MC68HC04 DESIGN CONSIDERATIONS

Designing with HCMOS devices is, in many respects, very straightforward. There are, however, a few fundamental points that should be noted to help produce reliable applications based around the MC68HC04.

6.4.1 Correct Termination

It is good design practice to terminate all unused pins to Vdd or ground through a resistor, say 100K OHM. A floating pin can self-bias around the CMOS switching point, resulting in the N and P Channel devices being on together. This will substantially increase Idd, and when reading a port configured as an input, will produce unpredictable results from the floating lines. The possibility of exceeding the capabilities of the static protection circuitry is also inevitabily increased for an unterminated pin.

6.4.2 Latch-Up

This phenomenom is found in all CMOS devices to a lesser or greater extent. A device can «latch-up» when the voltage on any pin becomes greater than Vdd (or less than Vss) by more than 0.5V. Under these biasing conditions, a parasitic thyristor is formed between Vdd and Vss from the existing structures on the silicon, the offending pin acting as its gate. Provided sufficient gate current is also available, then the thyristor will latch on, virtually short circuiting Vdd to Vss. Today'a devices are, of course, being designed to increase their immunity to this inherent effect. However, the systems designer should still be aware of the potential problem and how best to avoid it. The following guidelines will help.

A) Adequately decouple the supply lines. An HCMOS Idd waveform consists of very narrow pulses of tens of milliamps (corresponding to the clock frequency) and port plus leakage current the rest of the time. The power supply should be of low enough impedance to cope with these spikes without Vdd dropping by more than 0.5V. A 10 microfarad tantalum capacitor (or other high frequency arrangement) will often suffice.

B) Safeguard against high energy transients which may appear on I/O lines, especially if terminated some distance away from the MCU in a noisy enviroment.

C) When powering up and down, ensure that no voltages greater than Vdd are applied to the ports as Vdd rises, and that there is no line with so much capacitance on it that it will not track Vdd as it falls. This latter point could, for example, be a problem when moving from a Vdd of 5V to a back-up Vdd of 2V. Any large capacitor should be clamped to Vdd via a diode.

D) For inputs connected remotely or to edge connectors, it is recommended that a high value resistor be connected in series with the I/0 line. Input leakage is very low, so the maximum value will be limited by the acceptable port rise time. Edge connector lines should also be treated as floating inputs by connecting them to Vdd or ground through a resistor.

6.4.3 Power Dissipation

As with all CMOS devices, Idd is proportional to operating frequency and supply voltage. However, the 68HC04 (and CMOS 6805 devices) employs power saving instructions STOP and WAIT. STOP reduces Idd to leakage levels, however, it should be remembered that this does not take into account any port current that may also exist. Since a port will remain in the state it was placed in prior to executing STOP, the total current sourced from the ports can easily swamp the STOP Idd, effectively negating any power saving advantage.

6.4.4 I/0 drive

As Vdd drops, Idd also drops — but so does the I/0 drive capability as well. For a MOS transistor, Iout/Vin (transconductance) is proportional to Vg-Vt. Since Vg is close to Vdd (or GND for the P-Channel device) when the transistor is on, and Vt is relatively constant, then Iout will vary with Vdd. This characteristic also tends to increase propagation delays (as internal capacitances take longer to charge and dis-

charge) and consequently reduces the maximum operating frequency. Designers should take these inherent characteristics into consideration when designing around a 5 volt system which must also operate successfully at 2 volts.

Complementary Devices

7.1 INTRODUCTION

Motorola's product range includes many integrated circuits designed to perform specific functions under MCU control. In many cases, a member of the M6804 family will make an excellent partner to these circuits for a flexible yet cost effective system.

7.2 MEMORIES

These memories are designed principally for consumer and automotive applications requiring non-volatile storage of relatively small amounts of system information.

Device Type	Description	Features	Technology Package
MCM 144102	256 bit static RAM designed for 1.2V battery back up	* 16 x 16 bit organisation * Serial interface to MCU * Standby mode takes less than 10 μA * Directly expandable to 32 words	CMOS 8 pin DIL
MCM 2801	256 bit EEPROM	* 16 x 16 bit organisation * Serial interface compatible with 144102 * 10 year minimum data retention * +25V required for Erase and Program * +5V main supply	NMOS 14 pin DIL
MCM 2802	1024 bit EEPROM	* 32 x 32 bit organisation * 3 line interface to MCU * +25V required for Erase and Program * + 5V main supply * Multi-chip systems possible up to 16K bits	NMOS 14 pin DIL

7.3 DISPLAY DRIVERS

Motorola offers a range of MCU driven LED and LCD drivers for applications in TV, Hi-Fi, Car Radio, Appliance, Instrumentation, etc.

Device Type	Description	Features	Technology Package
UAA 2022	16 segment LED driver for use with common anode LEDs	* 3 line serial data bus input * Direct drive, non multiplexed * DC input for brightness control * No current limiting resistors required * Cascadable	LINEAR I^2L 24 pin DIL
MC 14499	4 digit LED decoder/driver for use with common cathode 7-segment LEDs	* 3 line serial data bus input * 4 way multiplexed by internal scanner * RC oscillator * Cascadable	CMOS 18 pin DIL
MC 144100	32 segment LED driver for use with common cathode LEDs	* 3 line serial data bus input * Duplex drive from 50/60 Hz mains * Minimal RFI * No flicker * Cascadable	CMOS 24 pin DIL
MC144115	16 segment LCD driver	* 3 line serial data bus input * Direct drive, non multiplexed * On chip oscillator provides AC drive * Wide operating voltage 3V-18V * Cascadable	CMOS 24 pin DIL
MC 144117	4 digit LCD decoder/driver compatible with MC 14499	* 3 line serial data bus input * Duplex drive * On chip oscillator	CMOS 24 pin DIL
MC 145000 **MC 145001**	Master/slave LCD drivers for use with 7-segment or dot matrix displays	* Serial data input, externally clocked * Master provides front and back-plane drive for 48 segments * Slave provides frontplane drive for 44 segments * 4 way multiplexed * Cascadable slaves	CMOS 24/18 pin DIL

7.4 D/A CONVERTERS

Motorola offers two D/A converters which will convert serial data from an MCU into an analog voltage.

Device Type	Description	Features	Technology Package
MC 144110	Six channel D/A Each channel consists of a shift register, latch and D/A converter	* 3 line serial data bus input * 6 bit resolution * Static R-2R conversion * 6 emitter follower outputs * Wide voltage range 4.5V-15V * Level translators on inputs * Data cascade output	CMOS 18 pin DIL
MC 144111	Four channel D/A Each channel consists of a shift register, latch and D/A converter	* 3 line serial data bus input * 6 bit resolution * Static R-2R conversion * 4 emitter follower outputs * Wide voltage range 4.5V-15V * Level translators on inputs * Data cascade output	CMOS 14 pin DIL

7.5 REMOTE CONTROL

Commands from Motorola's infra red transmitters can, after suitable detection, be easily decoded by an MCU for applications in consumer and industrial control.

Device Type	Description	Features	Technology Package
MC 14497	PCM infra red transmitter for up to 62 commands	* FSK or biphase AM coding * 6 bit data word * Low current in standby mode * Low duty cycle during transmission * 5V-10V supply * Uses inexpensive LC or ceramic oscillator	CMOS 18 pin DIL
MC 144105	PCM infra red transmitter for up to 512 commands	* Biphase AM coding similar to 14497 * 9 bit data word * 8 pages of 64 commands * Low current in standby mode * Low duty cycle * 4V-10V supply * Uses inexpensive LC or ceramic oscillator	CMOS 20 pin DIL
MC 3373	Infra red amplifier-detector	* High gain preamp. * Envelope detector for PCM demodulation * Simple interface to MCU * 5V-15V supply	BIPOLAR 8 pin DIL

7.6 PLL FREQUENCY SYNTHESISERS

Motorola's extensive range of MCU controlled frequency synthesisers will find applications in all types of electronic tuning, e.g. TV, Radio, CB, Avionics...

Device Type	Description	Features	Technology Package
MC 145144	General purpose PLL containing reference oscillator, reference divider, frequency divider, phase comparators 25 MHz input capability	* Divider programming by 4 bit data bus * 12 bit reference divider * 12 bit single modulus divider	CMOS 16 pin DIL
MC 145155	As above plus lock detector	* Divider programming by 4 bit data bus * 12 bit reference divider * 14 bit single modulus divider	CMOS 18 pin DIL
MC 145156	As above	* Divider programming by 4 bit data bus * 12 bit reference divider * 10/7 bit dual modulus divider	CMOS 20 pin DIL
MC 145151	As above	* Divider programming by parallel data bus * 14 bit reference divider * 14 bit single modulus divider	CMOS 28 pin DIL
MC 145152	As above	* Divider programming by parallel data bus * 14 bit reference divider * 10/6 bit dual modulus divider	CMOS 28 pin DIL
MC 145155	As above	* Divider programming by serial data bus * 16 bit reference divider * 14 bit single modulus divider * Band switch outputs	CMOS 18 pin DIL
MC 145156	As above	* Divider programming by serial data bus * 14 bit reference divider * 10/7 bit dual modulus divider * Band switch outputs	CMOS 20 pin DIL
MC 145157	As above	* Divider programming by serial data bus * 14 bit reference divider * 14 bit single modulus divider	CMOS 16 pin DIL

MC 145158	As above	* Divider programming by serial data bus * 14 bit reference divider * 10/7 bit dual modulus divider	CMOS 16 pin DIL
MC 145159	As above	* Divider programming by serial data bus * 14 bit reference divider * 10/7 bit dual modulus divider * Sample and hold phase detector	CMOS 20 pin DIL
UAA 4800	UHF PLL-Prescaler for TV and radio	* 1 GHz input capability * Bypassable ÷ 8 prescaler * Programmable reference divider * 15 bit frequency divider * Phase comparator * On chip op-amp (30V) * Programming via 2 line M-BUS	MOSAIC 18 pin DIL

7.7 TELECOM CIRCUITS

Those telephone circuits are designed for MCU control to give maximum performance and flexibility.

Device Type	Description	Features	Technology Package
MC 34010	Telephone chip (DTMF)	* All basic Telephone Station apparatus in a single I.C. * Low cost transducers * Low voltage operation : 1.4V * MCU interface for add-on features	BIPOLAR/ LINEAR I^2L 40 pin DIL
TCA 3381/2/3	Analogue telephone chip set	* Transmission circuit to French PTT specifications * DTMF or Pulse dial inputs * Optional speaker-phone function (TCA 3382) * Programmable ring signature impedance (TCA 3381)	BIPOLAR/ LINEAR 28, 22, 8 pin DIL
MC 145422/3/6	Universal Digital Loop Transceivers (UDLT)	* Synchronous Duplex 64 Kbit/s Voice/Data channel, plus two 8 Kb/s Signalling Data channels over telephone wire pair * Protocol independent	CMOS 20/22 pin DIL
MC 145429	Digital Telephone Set Audio Interface	* Interface between Codec/filter and telephone handset in MCU controlled Digital Telephone using UDLT MC 145422/3/6	CMOS 18 pin DIL

Quality

8.1 INTRODUCTION

MOTOROLA has a long standing reputation for manufacturing products of excellent QUALITY AND RELIABILITY since the introduction of the first car radio in 1928.
This has helped MOTOROLA to become one of the largest corporations exclusively devoted to electronics. Today, the company with sales revenues in 1982 of $ 3.8 billion and over 78,000 employees worldwide, consists of five separate and independent operations:

- Communications Sector
- Semiconductor Products Sector
- Information Systems Group
- Automotive & Industrial Electronics Group
- Government Electronics Group

The Semiconductor Products Sector is headquartered in Phoenix Arizona (USA) and it has manufacturing facilities throughout the world with a strong presence in Europe.
This section is intended to demonstrate the QUALITY AND RELIABILITY aspects of the semiconductor products supplied by MOTOROLA's European Facilities in:

- Toulouse, France
- East Kilbride, Scotland
- Munich, West Germany

8.2 MOTOROLA'S QUALITY PHILOSOPHY

MOTOROLA has always been dedicated to manufacture products which consistently meet and exceed our customers requirements.

We define quality as customer satisfaction

MOTOROLA's objective is now to achieve a zero defect level which is intended to minimise our customers quality costs.
During the last few years, MOTOROLA has continued to provide the resources necessary to support and encourage a wide range of Quality Improvement programs which consist of:

1. Develop quality consciousness
2. Revise and tighten design rules
3. Install vendor improvement programs
4. Improve process technology and equipment
5. Customer feedback programs
6. Reinforcement of inspection and analysis resources
7. Correlation programs

MOTOROLA's philosophy and culture regarding Quality is designed to ensure that every single person involved in the manufacture of semiconductors is completely aware of his or her responsibility for the Quality of the products they are producing. Every MOTOROLA employee is in reality a member of the Reliability and Quality Assurance department. This attitude is impressed upon each employee through training when they initially take up employment with MOTOROLA.
The Reliability and Quality Assurance departments are integral parts of the manufacturing activities. Their charter is to continuously measure and report on the status of the Quality and Reliability performance of all products whilst actively supporting and coordinating Quality improvement activities.

8.3 QUALITY IN MANUFACTURING

8.3.1 Quality in Design

MOTOROLA's quality activity starts at the product design stage. It is our philosophy to "design in" reliability. At all development points of any new design reliability orientated guidelines are continuously used to ensure that a thoroughly reliable part is ultimately produced. This is demonstrated by the excellent in-house reliability testing results obtained for all MOTOROLA's semiconductor products and, more importantly, by our numerous customers.

8.3.2 Material Incoming Controls

Each vendor is supplied with a copy of the MOTOROLA Procurement Specification which must be agreed in detail between both parties before any purchasing agreement is made. This is followed by a vendor appraisal report whereby each vendor's manufacturing facility is visited by MOTOROLA Quality Engineers responsible for ensuring that the vendor has a well organized and adequately controlled manufacturing process capable of supplying the high quality material required to meet the MOTOROLA Incoming Inspection Specification. Large investments have and are continuously being made and Quality Improvement programs developed with our main suppliers concerning:

Masks — Silicon — Piece-parts — Chemical products — Industrial gas, etc.

Each batch of material delivered to MOTOROLA is quarantined at Goods-in until the Incoming Quality Organization has subjected adequate samples to the incoming detailed inspection specification. In the case of masks this will include mask inspection for:

1. Defect Density
2. Intermask Alignment
3. Mask Revision
4. Device to Device Alignment
5. Mask Type

Silicon will undergo the following inspections:

1. Type "N" or "P"
2. Resistivity
3. Resistivity Gradient
4. Defects
5. Physical Dimensions
6. Dislocation Density

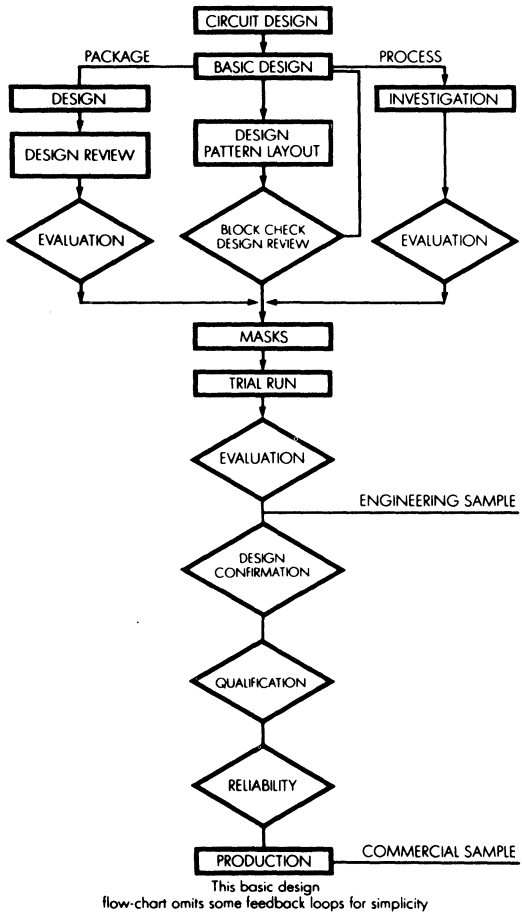


Figure 8.1. New Product Design Flow

8.3.3 Wafer Fabrication

All processing stages of MOTOROLA products are subjected to demanding manufacturing and quality control standards.
The MOS Wafer Fabrication flow chart in Figure 8.2 highlights the various in-process control points audited by both Manufacturing and Quality people. The majority of these inspections are control audit points with inspection gates at critical points of the process : this is in line with MOTOROLA's policy of all personnel being responsible for quality at each maunfacturing stage.
Diffusion and ion implantation processing is subject to oxide thickness controls and penetration evaluations. Controls are also performed on resistivity and defect density. Diffusion furnaces, metallization and passivation equipment are subjected to daily qualification requirements by using C-V plotting techniques. C-V techniques are used also to ensure ongoing stability as they do provide a very sensitive measurement of ionic species concentration.
In addition many other specific controls are used as a means to ensure built-in reliability and provide statistical trend data, which include :

- Environment monitoring for humidity, temperature and particles.
- Deionozed water resistivity, particles and bacteria checks in water.
- Epitaxial material : resistivity - thickness - crystal defects.
- Oxide : thickness - charges - pinhole density.
- Metallization : thickness adherence - metal composition - ohmic contacts.
- Doping profiles.
- Pre and post etch inspections.
- In process SEM analysis for step coverage : metallization - grain size - phosphorous concentration.
- Passivation integrity checks.
- Calibration.
- Final visual inspection gate.

After all processing stages are completed, every wafer lot is subjected to a detailed electrical parameter check. Parameters such as threshold voltage, junction breakdown voltages, resistivity, field inversion voltages, etc. are measured and each batch is sentenced accordingly. The data generated at this point is treated statistically as a control on the distribution of each key electrical parameter thus allowing corrective action adjustments to be implemented in a timely manner.
Every wafer lot is submitted to an electrical probe test during which every individual die is tested to its electrical specification. Chips which fail are individually inked.

8.3.4 Assembly

MOTOROLA continuously makes major investments in specialized assembly areas located in Malaysia, the Philippines and Korea. These assembly plants employ the latest technologies available to ensure that all MOTOROLA semiconductors are produced to the highest standards of Quality and Reliability. In addition, each European wafer fabrication facility has in-house assembly capability which allows some production, specific engineering activity and qualification of European piece-parts suppliers.
Identical Quality and Reliability philosophies are practised in the assembly areas as within the wafer fabrication facilities. Quality Assurance Audits for immediate corrective actions are performed after major process steps as demonstrated in the flow-chart. In addition, screening options are available. The statistical data obtained from quality audits are reported to the appropriate European business centers either daily, weekly or monthly for review.
MOTOROLA is particularly aware of the major impact moisture can have on the reliability performance of either plastic or ceramic parts. With this in mind several major new innovations have been introduced to safeguard MOTOROLA products and thus enhance their overall reliability performance, these include :

- Faraday shield vacuum packed wafer shipping system
- Temperature and humidity controlled wafer inventory stores
- Inert atmosphere for metal can packages encapsulation
- New design lead frames (plastic assembly)
- New molding compounds
- Low moisture content glass

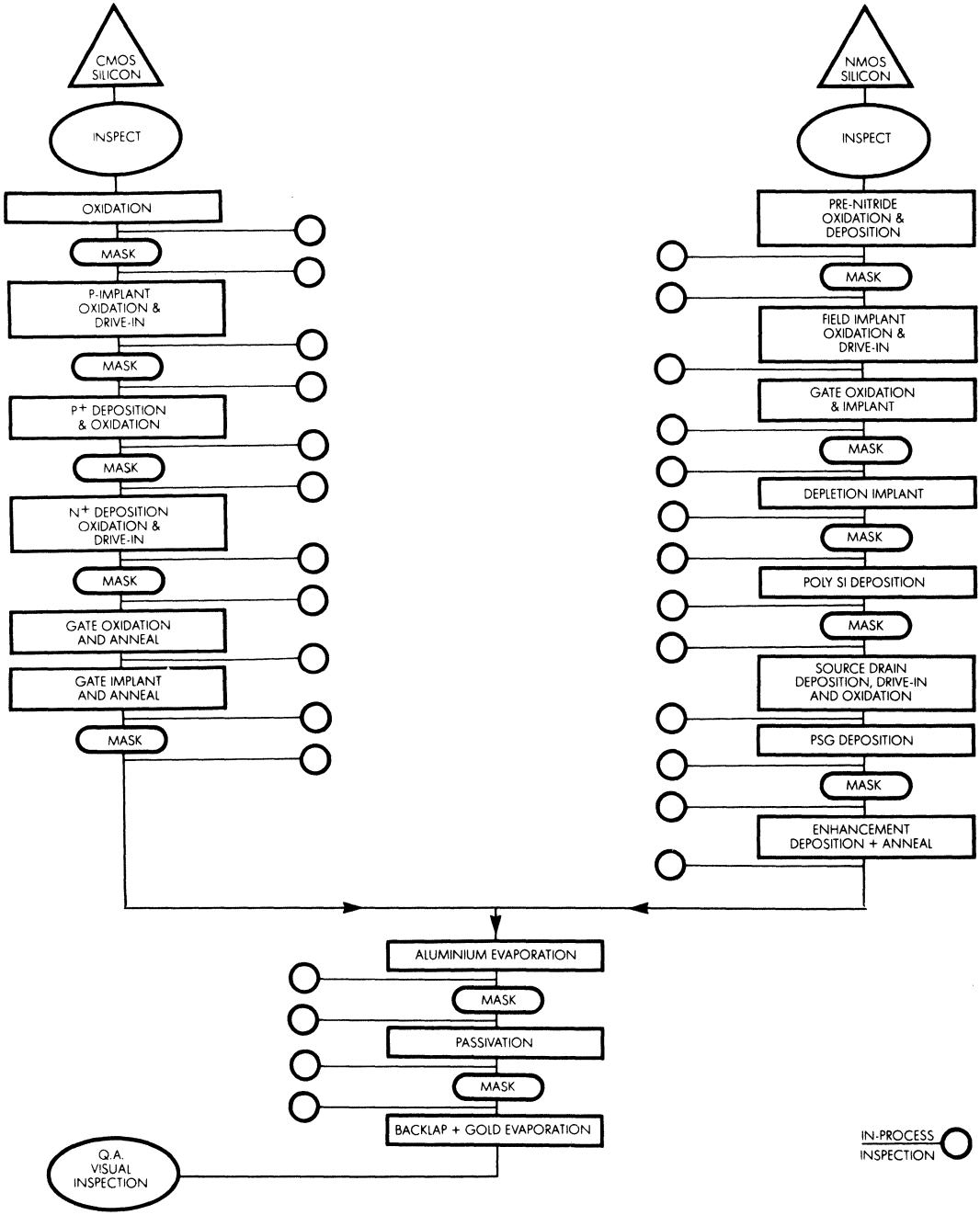
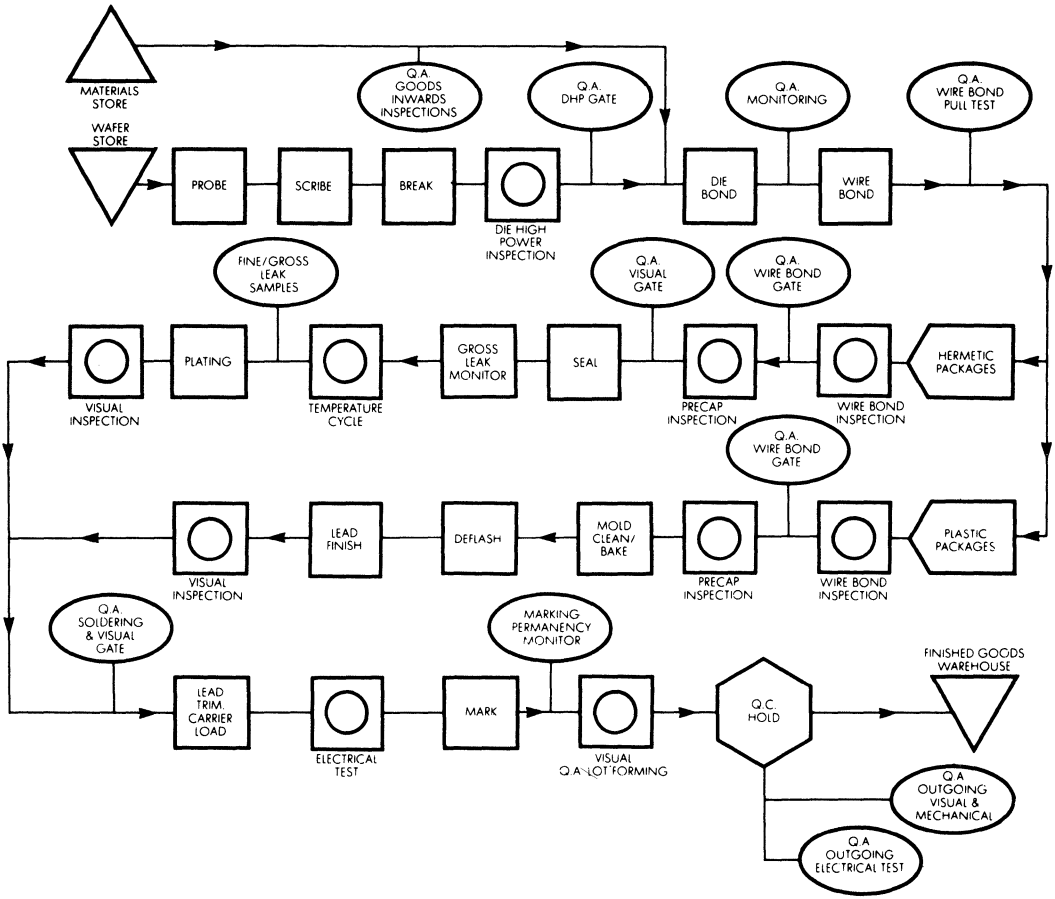


Figure 8.2. MOS Wafer Fabrication Flowchart

- Moisture content audit procedures
- Super dry piece-part controls.

8.3.5 Final Testing

Each of MOTOROLA's European facilities has a complete Final Test capability for all of the products fabricated and assembled. The majority of products after assembly are tested and Q.A. released at the European facility responsible for that product. Some product is tested in the offshore assembly site ; however, this is always returned to the European facility for Q.A. release prior to final shipment to customer.
Final Test is a comprehensive series of D.C., functional and speed orientated electrical tests as well as adapted forced tests. These tests are normally more stringent than data sheet requirements and are finally sampled by Outgoing Quality Assurance.
In practise, the test flow philosophies vary according to product. For instance, in Toulouse most of the Discrete devices are double tested as part of a zero defect quality improvement program. As well, many Integrated Circuits are tested at various temperatures. There are also many burn-in options available.



8.3.6 Outgoing Quality

Although test procedures may vary from product to product within MOTOROLA, the same philosophy applies when considering quality objectives. MOTOROLA's mission is to be a Quality and Reliability leader in Europe.

Our customers measure us by the level of defects in the products we supply at incoming inspection, during assembly and, most important, field reliability.

During the past years, MOTOROLA in Europe has achieved impressive reductions in defect rates known as A.O.Q. or Average Outgoing Quality. Instrumental in this success has been the planned continuous reduction in outgoing A.Q.L. to a point where MOTOROLA in Europe believes that over all products it can demonstrate the most aggressive A.Q.L.'s in the industry.

This aggressive program has been designed to help eliminate expensive incoming inspection at our customers.

All of the European facilities also practise an extremely demanding parts per million program (PPM).

The PPM performance of all MOTOROLA products is calculated in each location using the same method; they are, therefore, directly comparable. MOTOROLA is well aware that when discussing PPM with existing or potential customers, it is of paramount importance to explain exactly which failure categories are included in the stated PPM figure. MOTOROLA's PPM figures will include:

- Electrical Inoperative failures
- Electrical Parametric failures (D.C. and A.C.)
- Visual and Mechanical criteria.

In many published cases, stated PPM values refer to Electrical Inoperative failures only.

At MOTOROLA, the Electrical Inoperative, the Electrical Parametric and the Visual Mechanical failure rates are calculated separately and then combined to reach an overall total. In this way MOTOROLA believes that it is giving its customers a true and accurate assessment of the quality of the product. Unqualified PPM statements can be misleading and cause the customer to expect quality levels which cannot be achieved. For example, MOTOROLA CMOS A.O.Q. is quoted at 1,250 PPM overall Electrical parameters and including Visual/ Mechanical categories. However, the function failure level is less than 200 PPM. Other product families such as Small Signal Plastic Transistors are already reaching 50 PPM in Electrical Inoperative failure rate.

Throughout the semiconductor industry there have been, and there still are, examples of manufacturers offering higher quality standards at a premium. This is not a MOTOROLA strategy, we believe that our customers should expect high quality products at no extra cost. This is MOTOROLA's aim and we will continue to aggressively pursue Quality and Reliability improvements which will be passed on to our customers as an obligation on our part.

Also, we actively encourage our customers to provide their quality results at their Incoming Inspection, during their manufacturing process and from the field in order to better correlate and further improve our quality performance.

8

8.4 RELIABILITY TESTS: DEFINITION, PURPOSE AND PROCEDURES

These definitions are intended to give the reader a brief understanding of the tests currently used at MOTOROLA for reliability checking. They also state which main failure mechanisms are accelerated by the test.

High Temperature Storage Life

An environmental test where only temperature is the stress. Temperature and test duration must be specified, usually temperature is the maximum storage temperature of the devices under test. Main failure mechanisms are metallization, bulk silicon, corrosion.

High Temperature Reverse Bias (HTRB)

An environmental stress combined with an electrical stress whereby devices are subjected to an elevated temperature and simultaneously reverse biased. To be effective voltage must be applied to the devices until they reach room temperature at the completion of the test. Temperature, time and voltage levels

must be specified. Accelerated failure mechanisms are inversion, channeling, surface contamination.

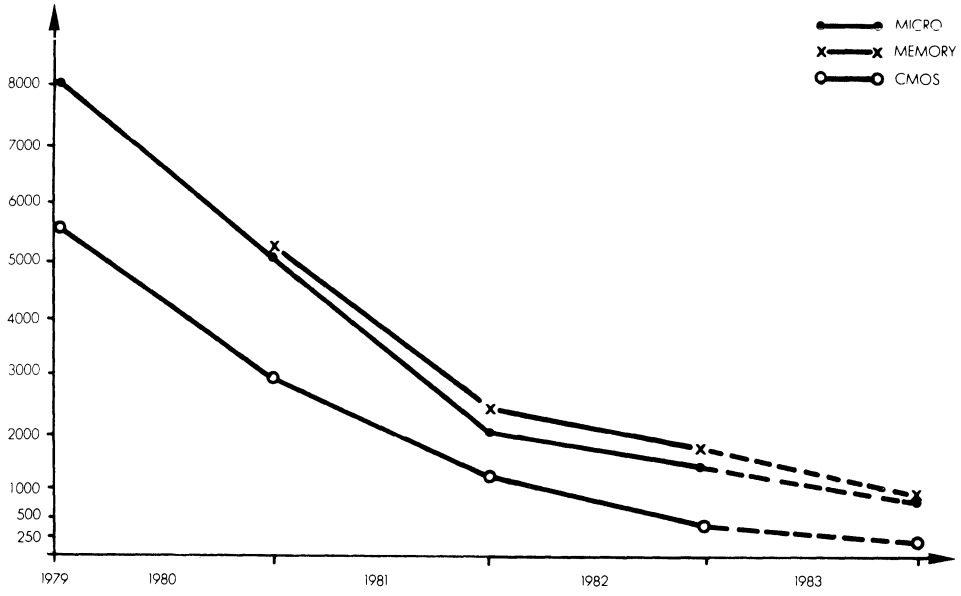


Figure 8.4. Average Outgoing Quality in PPM for MOS Products Functional/Parametric Failures

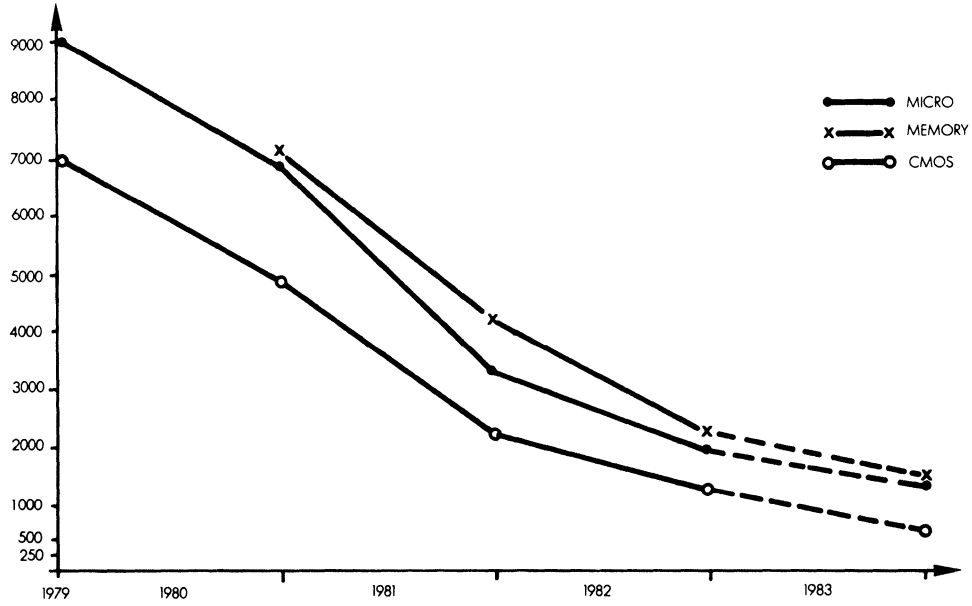


Figure 8.5. Average Outgoing Quality in PPM for MOS Products Functional/Parametric/Visual/Mechanical Failures

High Humidity, High Temperature Reverse Bias (H^3 TRB)

A combined environmental/electrical stress whereby devices are subjected to an elevated ambient temperature and high humidity, simultaneously reverse biased for a period of time. Normally performed on a sample basis (qualification) on non-hermetic devices. The most common conditions are 85°C and 85% relative humidity. More extreme conditions generally are very destructive to the chambers used. Time, temperature, humidity and voltage must be specified. This accelerated test mainly detects corrosion risks.

Steady State Operating Life

An electrical stress whereby devices are forward (reverse for zeners) biased at full rated power for prolonged duration. Test is normally 25°C ambient and power is 100% of full rated. (For power devices and I/C's maximum operating T_J is used.) Duration, power and ambient, if other than 25°C, must be specified. Accelerated failure mechanisms mainly are metallization, bulk silicon, oxide, inversion and channeling.

Dynamic Operating Life

An electrical stress whereby devices are alternately subjected to forward bias at full rated power or current and reverse bias.
Duration, power, duty cycle, reverse voltage ambient and frequency must be specified. Used normally for rectifiers and silicon controlled rectifiers. Failure mechanisms are essentially the same as steady state operating life.

Intermittent Operating Life (Power Cycling)

An electrical stress whereby devices are turned on and off for a period of time. During the "on" time the devices are turned on at a power such that the junction temperature reaches its maximum rating. During "off" cycle the devices return to 25°C ambient. Duration, power or duty cycle must be individually specified. Accelerated failures mechanisms mainly are die bonds, wire bond, metallization, bulk silicon, oxide.

Thermal Shock (Temperature Cycling)

An environmental stress whereby devices are alternately subjected to a low and high temperature with or without a dwell time in between to stabilize the devices to 25°C ambient — the medium is usually air. Temperatures, dwell times and cycles must be specified. Failure mechanisms are essentially die bonds, wire bonds, package.

Thermal Shock (Glass Strain)

An environmental stress whereby the devices are subjected to a low temperature, stabilized and immediately transferred to a high temperature. The medium is usually liquid. Failures mechanisms essentially are the same as temperature cycling.

Mechanical Shock

A mechanical stress whereby the devices are subjected to high impact forces normally in two or more of the six orientations X1, Y1, Z1, X2, Y2, Z2. Tests are to verify the physical integrity of the devices. G forces, pulse duration and number of shocks and axes must be specified.

Vibration Variable Frequency

Same as Vibration Fatigue except that frequency is logarithmically varied from 100 Hz to 1 kHz and back. Number of cycles is normally four. Cycle time, amplitude and total duration must be specified. Failure mechanisms are mainly package, wire bond — this test is not applicable to molded devices.

The reliability approach at MOTOROLA Semiconductors is based on designing-in reliability rather than testing for reliability only. This concept is reflected by MOTOROLA's mandatory procedures which require product, process and packaging qualification on three independently produced lots before any product is released to volume production. Reliability engineering approval supported by an officially documented report is required before any product is released to manufacturing. Tests at both maximum rated and accelerated stress levels are performed. Acceleration is important to determine how and at what stress level a new design, product process or package would fail. This information provides an indication of what design changes can be implemented to ensure a wider and safer margin between the maximum rated stress condition and the devices stress limitation.

As well as qualifying all new products, processes and piece-parts, each MOTOROLA manufacturing facility operates an ongoing reliability monitor which covers all process and packaging options. This program provides a continuous up-to-date data base which is summarized in periodical reports.

Reliability statistics supporting all MOTOROLA Semiconductor devices can be obtained from any of the MOTOROLA Sales Offices upon request. The present operating life test results demonstrates MOTOROLA's reputation for producing semiconductors with reliability second to none.

Handling Precautions

The following precautions should be taken with all HCMOS devices:

1. All IC's should be stored or transported in materials that are antistatic. Devices must not be inserted into conventional plastic "snow" styrofoam or plastic trays, but should be left in their original container until ready for use.
2. IC's should be placed on a grounded bench surface and operators should ground themselves prior to handling devices. Since a worker can be statically charged with respect to the bench surface, wrist straps in contact with skin are strongly recommended. See Figure 9-1.
3. Nylon or other static generating materials should not come in contact with IC's.
4. If automatic handling is being used, high levels of static electricity may be generated by the movement of devices or boards. Avoid this by grounding suspect area and/or by the use of ionized air blowers.
5. Cold chambers using CO_2 for cooling should be equipped with baffles, and devices must be contained on or in conductive material.
6. When lead-straightening or hand-soldering is necessary, provide ground straps for the apparatus used and be sure that soldering ties are grounded.
7. The following steps should be observed during wave solder operations.
   a) The solder pot and conductive conveyor system of the wave soldering machine must be grounded to an earth ground.
   b) The loading and unloading work benches should have conductive tops which are grounded to an earth ground.
   c) Operators must comply with precautions previously explained.
   d) Completed assemblies should be placed in antistatic containers prior to being moved to subsequent stations.
8. The following should be observed during board cleaning operation.
   a) Vapor degreasers and baskets must be grounded to an earth ground. Operators must likewise be grounded.
   b) Brush or spray cleaning should not be used.
   c) Assemblies should be placed into the vapour degreaser immediately upon removal from the antistatic container.
   d) Cleaned assemblies should be placed in antistatic container immediately after removal from the cleaning basket.
   e) High velocity air movement or application of solvents and coatings should be employed only when module circuits are grounded and a static eliminator is directed at the module.
9. The use of static direction meters for line surveillance is highly recommended.
10. All low impedence equipment (pulse generators, etc.) should be connected to inputs only after the device is powered up. Similarly, this type of equipment should be disconnected before power is turned off.
11. A circuit board containing devices is merely an extension of the device and the same handling precautions apply. Contacting edge connectors wired directly to inputs can cause damage. Plastic wrapping should be avoided. When external connections to a PC board address only an input of an integrated circuit, it is recommended that a resistance of 10K or greater be used in series with the input.

This resistor will limit accidental damage if the PC board is removed and brought into contact with static generating materials.

12. Do not insert or remove devices from test sockets with power applied. Check all power supplies to be used for testing devices to be certain there are no voltage transients present.
13. Double check test equipment setup for proper polarity of voltage before conducting parametric or functional testing.
14. Do not exceed the maximum electrical voltage ratings specified by the data sheet.
15. All unused device inputs should be connected to V_{CC} or V_{EE}.

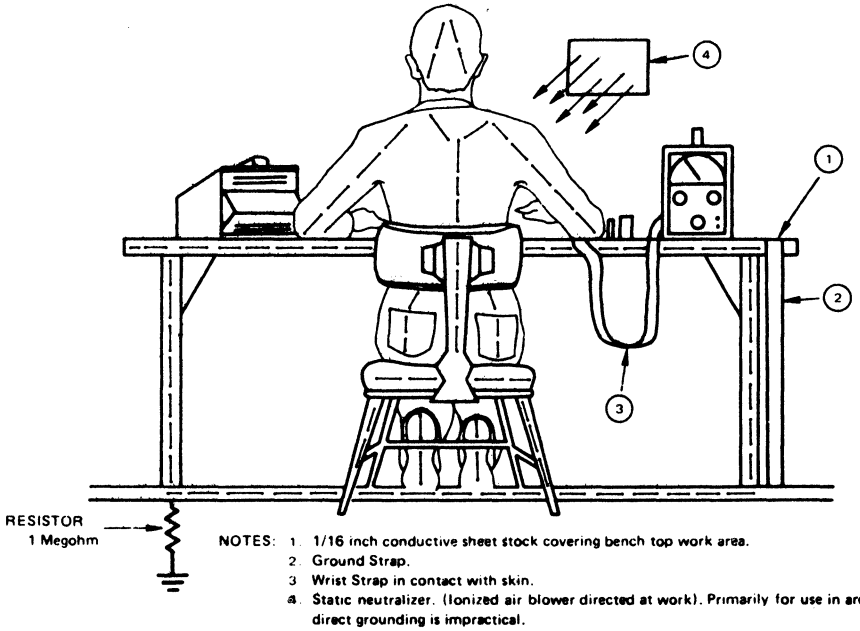


Figure 9.1. Typical Manufacturing Work Station

MOTOROLA SEMICONDUCTOR SALES OFFICES
EUROPE and AFRICA

AUSTRIA
Motorola Ges.m.b.H.
Alserbachstrasse 30 — A-1090 Wien
Tel. (0222) 31 65 45

DENMARK
Motorola Semiconductors A.S.
Tornerosevej 127
2730 Herlev
Tel. 452 92 00 99

FRANCE
Motorola Semiconducteurs Commerciale S.A.
Main Sales Office
2, rue Auguste-Comte — Bp. 39
92173 Vanves-Cedex
Tel. 736 01 99

Sales Office
Chemin de Malacher-Zirst
38240 Meylan (Grenoble)
Tel. (76) 90 22 81

Sales Office
Zone artisanale — 35740 Pace
Tel. (99) 60 65 48

Sales Office
Avenue General-Eisenhower
31023 Toulouse Cedex
Tel. (61) 41 90 00

WEST GERMANY
Motorola GmbH. Geschäftsbereich Halbleiter
Main Sales Office
Arabellastrasse 17
8000 München 81
Tel. (089) 92 720

Sales Office
Hans-Bockler-Strasse 30
3012 Langenhagen
Hanover
Tel. (0511) 78 20 37 38

Sales Office
Vimsbergerstrasse 43
8500 Nürnberg
Tel. (0911) 6 57 61

Sales Office
Stralsunder-Strasse 1
7032 Sindelfingen
Tel. (0703) 18 30 74 75

Sales Office
Abraham-Lincoln-Strasse 28
6200 Wiesbaden
Tel. (06121) 76 19 21

HOLLAND
Motorola B.V. Semiconductor Division
Maarssenbroeksidijk 37
3606 AG Maarssen
Tel. (030) 43 96 53

ITALY
Motorola S.p.A. Divisione Semiconduttori
Main Sales Office
Centro Milanofiori-Strade 2-C2
20090 Assago (Milano)
Tel. 928 22 01

Sales Office
Via Costantino Maes 68
00162 Roma
Tel. 831 47 46

NORWAY
Motorola A.S.
Plogveien 1A, N-9679 Oslo 6
Tel. (02) 198070

SOUTH AFRICA
Motorola South Africa (Pty) Ltd.
5th Street, Wynberg, Transvaal
P.O. Box 39586, Bramley, 2018
Tel. 786-6165

SPAIN
Motorola Espana S.A.
Albert Alcocer, 46 Dpdo
28016 Madrid
Tel. 457.82.04

SWEDEN
Motorola AB.
Dalvagen 2
S-17136 Solna
Tel. (08) 83 02 00

SWITZERLAND
Motorola (Schweiz) AG
Main Sales Office
Uitikonerstr. 9
8952 Schlieren
Tel. (01) 730 40 74

Sales Office
16, chemin de la Voie-Creuse
P.O. Box 8
1211 Geneve 20
Tel. (022) 99 11 11

UNITED KINGDOM
Motorola Ltd.
Main Sales Office
Motorola House
69 Buckingham Street
Aylesbury, Bucks, HP20 2NF
Tel. 0296 35252

Sales Office
Colvilles Road, Kelvin Estate
East Kilbride, Scotland G75 0TG
Tel. (03552) 39101

European Literature Centre
88 Tanners Drive, Blakelands
Milton Keynes MK14 5BP
Tel. (0908) 614614

HEADQUARTERS EUROPEAN OPERATIONS SWITZERLAND
Motorola Inc. European Semiconductor Division
16, chemin de la Voie-Creuse
P.O. Box 8
12111 Geneve 20
Tel. (022) 99 11 11

EUROPEAN FACTORIES

FRANCE
Motorola Semiconducteurs S.A.
Avenue General-Eisenhower
31023 Toulouse CEDEX
Tel. (61) 41 90 00

GERMANY
Motorola GmbH
Münchner Strasse 18
8043 Unterföhring
Tel. (089) 92 481

UNITED KINGDOM
Motorola Limited
Colvilles Road, Kelvin Estate
East Kilbride, Scotland G75 0TG
Tel. (03552) 39101

YOUR LOCAL DISTRIBUTOR IS:

MOTOROLA