Introducing Microprocessors Part 2

The basic terms of the microprocessor system.

By Mike Tooley

Learning Objectives

The general learning objectives for part two of *Introducing Microprocessors* is that readers should be able to:
(a) draw a block diagram showing the internal architecture of a representative 8-bit microprocessor and state the function of each of the principal internal elements (2.1)
(b) state and explain the function of each of the principal external connections of a representative 8-bit microprocessor (2.1)
(c) explain the need for a clock and state typical frequencies and periodic times for microprocessor clocks (2.1)
(d) make appropriate use of manufacturers' data sheets (2.1)
The specific objectives for the part are as follows:
2.1 Internal Architecture of a Microprocessor
2.1.1 Draw and interpret a block diagram showing the internal architecture of a representative 8-bit microprocessor.
2.1.2 State and explain the function of each of the principal internal registers of a representative 8-bit microprocessor.
2.1.3 State and explain the function of each of the principal external connections of a representative 8-bit microprocessor.
2.1.4 Explain the need for a clock and distinguish between external and internal microprocessor clocks.
2.1.5 State the range of typical clock frequencies and periodic times for common 8-bit microprocessors.
2.1.6 Use manufacturers' literature to determine the supply voltage, pin-out, and internal features of any common 8-bit microprocessor.

Microprocessors

In Part 1 we briefly mentioned that a microprocessor performs the functions of a central processing unit (CPU) within a microcomputer. We also stated that the microprocessor provides control and synchronization signals for the rest of the system. From this, it should be obvious that the microprocessor is the single most important component within any microcomputer system.
The basic internal elements of a microprocessor are as follows:
(a) registers for temporary storage of instructions, data, and addresses
(b) an arithmetic logic unit (ALU) able to perform a variety of arithmetic and logic functions
(c) control logic which accepts and generates external control and supervisory signals (such as RESET and READ/WRITE) and synchronizes data transfers within the system.

Registers

Internal registers can be thought of as arrangements of pigeon holes into which data (in binary form) can be placed during processing. Some registers are directly accessible to the programmer (i.e. he can set to read their contents at will) whilst others are reserved for the machine's own use. Registers may also be classified as "dedicated" (i.e. they have a specific purpose such as pointing to a memory location or holding the results of an ALU operation) whilst others are described as "general purpose".

In the case of an 8-bit microprocessor, most of the general purpose registers will be capable of storing eight bits. Furthermore, since each of the bits may be either 0 or 1, there will be a total of 256 possibilities for the contents of such a register, ranging from 00000000 to 11111111. Registers used for "pointing" to memory locations, on the other hand, will generally be capable of holding sixteen bits and consequently their contents may range from 0000000000000000 to 1111111111111111 (ie, 0 to 65535 decimal).

The data bus lines in an 8-bit microcomputer are labelled D0 to D7. The most significant data bit (i.e. that with the greatest binary weight) appears on D7 whilst the least significant bit (i.e. that with the least binary weight) appears on D0. In the case of a 16-bit address bus, the lines are labelled A0 to A15 and the most and least significant address bits are respectively those which appear on address lines A15 and A0. The most and least significant bits are often referred to as the MSB and LSB last (see Part 1).

Unfortunately, there is some considerable variation in both the internal architecture and terminology used by different microprocessors manufacturers. Despite this, there are a number of common themes. The major microprocessors families, for example, tend to retain a high degree of upward compatibility both in terms of internal architecture and the software "instruction set" and that is clearly an important consideration in making a new product attractive to the equipment manufacturer.

Introducing IMP

IMP stands for introductory microprocessor, a hypothetical device

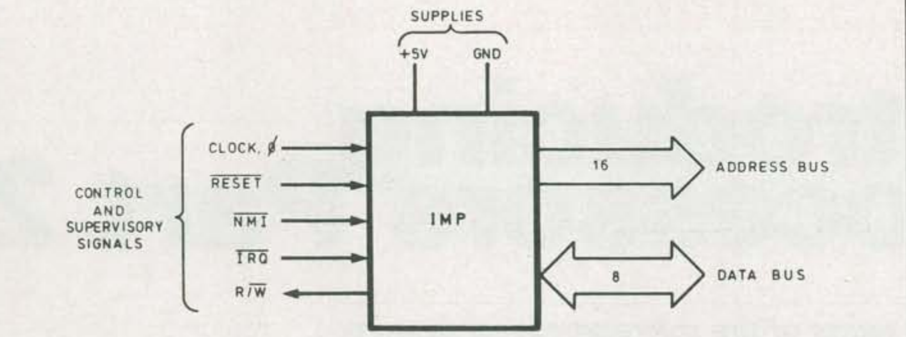


Fig. 2.1. IMP's external connections. Note that a bar over a particular signal indicates that it is active-low (logic 0 when activated).

which we shall be using to explain some of the fundamental concepts of microprocessors. We have chosen to follow this route, rather than tailor our description to real microprocessor, in order to keep the explanation as simple as possible. IMP contains many of the features found in a real 8-bit microprocessor without favouring the architecture of any particular processor family. By this means, we hope to provide readers with a gentle introduction to microprocessor avoiding superfluous or processor specific information which may otherwise serve only to confuse the newcomer.

Important differences between IMP and real microprocessors will be discussed as we progress but readers who require detailed information on particular microprocessors need not worry as we shall be presenting this information in the current series of Data Cards. These cards will feature all the most popular 8-bit microprocessors (6502, 6800, 8085 and Z80) and will build to provide a useful library of microprocessor related data.

IMP has an 8-bit data bus, 16-bit address bus and five control and supervisory signal lines. Like most 8-bit microprocessors, IMP has a 40-pin d.i.l. package and operates from a +5V supply. IMP's connections with the outside world are shown in simplified form in Fig. 2.1

Internal architecture

The internal arrangement (architecture) of the IMP is shown in Fig. 2.2. At first sight this diagram may look rather complex so we will spend some time explaining each individual feature and how it relates to the working of the unit as a whole.

The majority of IMP's internal registers are linked together by means of an internal data bus. This bus can be thought of as a highway along which bytes are transferred from one register to another. Since we are dealing with an 8-bit microprocessor, the internal data bus is naturally eight bits wide.

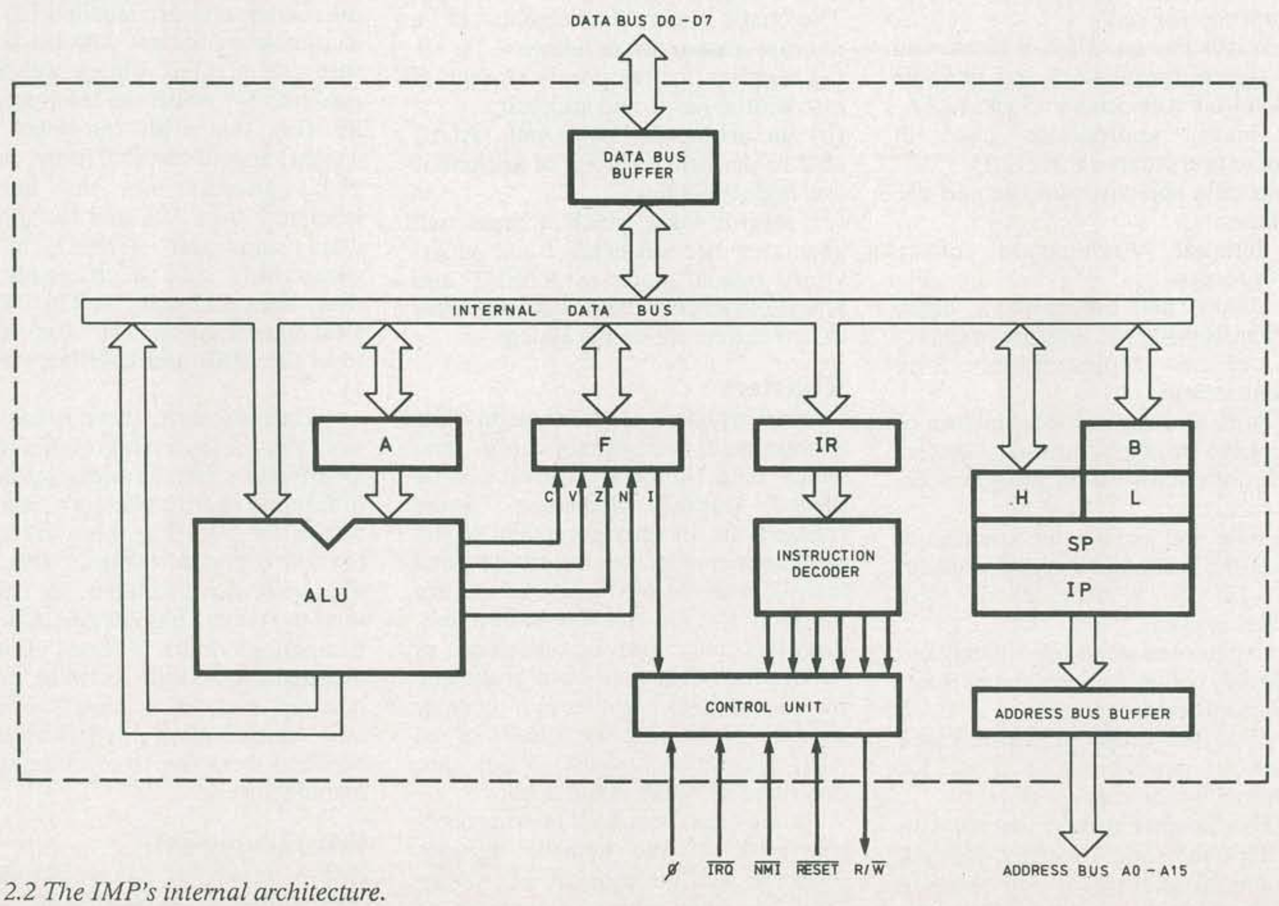


Fig. 2.2 The IMP's internal architecture.

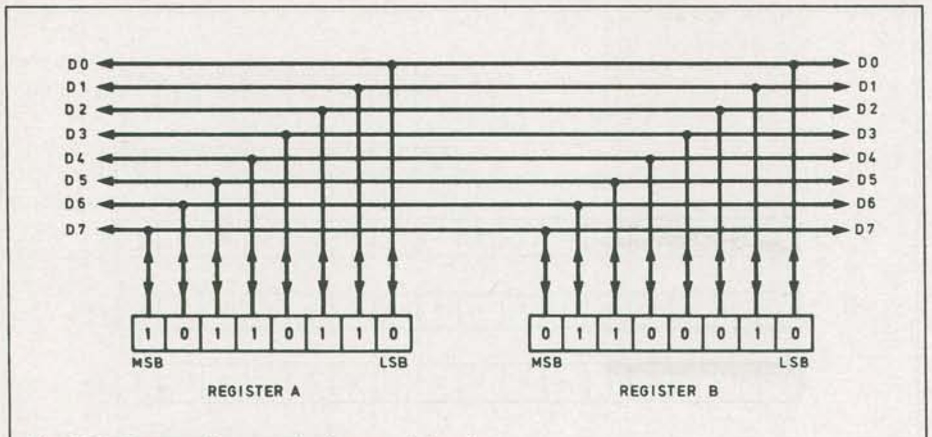


Fig. 2.3. Connections to the internal data bus.

Fig. 2.3 shows how two 8-bit registers (A and B) are coupled to the internal bus. Separate lines from the control unit (not shown on either Fig. 2.1. or Fig. 2.3) are used to determine whether:

(a) the contents of the register are to be made available on the bus (so that it may be copied elsewhere),

or (b) the data currently present on the bus is to be latched into the register (replacing whatever was there before),

or (c) the register is to be isolated from the bus (preserving its contents for future use).

Now, consider the process of copying data from register A to register B. We would need to make the data in register A available on the bus (case (a) above), latch the data into register B (case (b) above), and ensure that every other register connected to the bus was currently isolated (case (c) above). If this is beginning to sound rather complex, there is no need to worry as the generation of the necessary internal control signals is both implicit in a particular instruction and entirely automatic.

Data Bus Buffer

The data bus buffer separates the internal data bus from the external data bus and it incorporates eight individual bidirectional current amplifiers. The buffers may be made to receive data from the external bus or transmit data to the external bus in response to control signals (not shown in Fig. 2.2). The buffer helps regularize the logic levels received by the microprocessor and provides a reasonable amount of current gain for "driving" the external bus. The data bus buffer thus provides a means of isolating the microprocessor from the harsh world outside!

[Some microprocessors allow the microprocessor to isolate itself from the data bus by placing the data bus buffer in an open- circuit (i.e. disconnected) or "tri-state" condition. This allows other "intelligent" devices to place information on the data bus.]

Address bus buffer

The address bus buffer behaves in a similar fashion to that of the data bus buffer. It is, however, important to note that the individual address bus buffers are unidirectional since address information is only generated by the microprocessor and not received by it.

[Some microprocessors allow the microprocessor to isolate itself from the address bus by placing the address bus buffer in an open-circuit (i.e. disconnected) or "tri-state" condition. This allows other "intelligent" devices to place information on the address bus.]

Instruction Pointer (IP)

The instruction pointer is a 16-bit register which contains the address of the next instruction byte to be executed. The contents of the register is thus said to "point" to the next instruction byte. The contents of the instruction pointer is automatically incremented each time an instruction byte is fetched.

[Note: Many microprocessors refer to this register as a Program Counter (PC)]

Accumulator (A)

The accumulator is an 8-bit register which functions both as a source and destination register; not only is it the source of one of the data bytes required for an ALU operation but it is also the location in which the result of an ALU operation is placed.

Flag Register (F)

The flag register contains information on the internal status of the microprocessor and, in particular, signals the result of the last ALU operation. It is important to note that the flag register is not a register in the conventional sense; it is simply a collection of bistable latches which can be "set" or "reset" depending upon the result of an ALU operation. The output of each bistable can be considered to act as a "flag". IMP has the following flags:

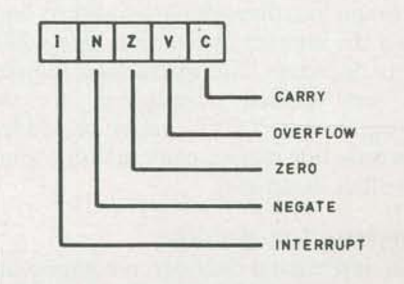


Fig. 2.4. IMP's flag register.

CARRY (C) – set to 1 when the last ALU operation has produced a carry

OVERFLOW (V) – set to 1 if the last ALU operation resulted in an overflow

ZERO (Z) – set to 1 if the result of the last ALU operation was zero

NEGATE (N) – set to .1 when subtraction has taken place, otherwise reset to 0

INTERRUPT (I) – set to 1 when interrupts are disabled (in this state the microprocessor is unable to accept and "interrupt request" generated by an external device).

The composition of IMP's flag register is shown in Fig. 2.4. It is important to note that once changed, the various flag bits remain either set or reset until a further change occurs. The programmer is only able to directly affect the state of the CARRY and INTERRUPT flags. The others change state indirectly as a result of program execution.

Stack Pointer (SP)

IMP needs to have access to an external area of read/write memory (RAM) which permits temporary storage of data. This area of memory is known as a "stack" and it may typically occupy between 16 and 256 bytes of memory. (Note, however, that the stack is a dynamic structure and its size varies continuously during processing.)

The stack operates on a "last-in first-out" (LIFO) basis; data is "pushed" onto the stack and later "pulled" off it. The "stack pointer" keeps track of the extent of the stack by holding the address of the last used stack location.

[Note: Some popular microprocessors (e.g. 6809) have two independent stack pointers; a "system stack pointer" (SSP) and a "user stack pointer", (USP).)

Instruction register

The instruction register is a temporary storage location which is used to contain the current instruction byte whilst it is decoded. The instruction register is not directly accessible to the programmer (i.e. it cannot be loaded directly nor can its contents be copied to other locations).

Instruction decoder

The instruction decoder, a complex arrangement of logic gates with outputs which are fed to the control unit, operates on the instruction currently held in the instruction register. The instruction decoder informing the control unit of the actions demanded by the current instruction (e.g. the need to take the R/W line low and latch the contents of the HL register pair into the address bus buffer).

Control Unit

The control unit generates internal control signals (not shown in Fig. 2.2) which determine the direction, source and destination of internal data transfers, activates external control lines when required, and responds to external signals which arrive on the control bus. The control unit is also responsible for internal synchronization.

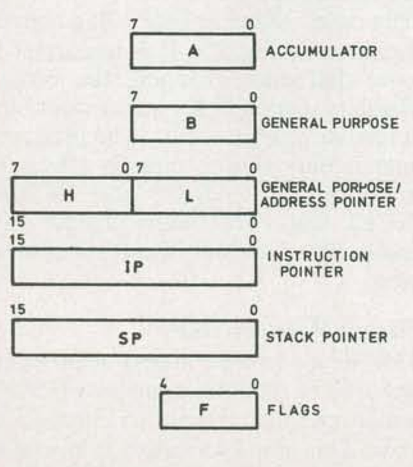


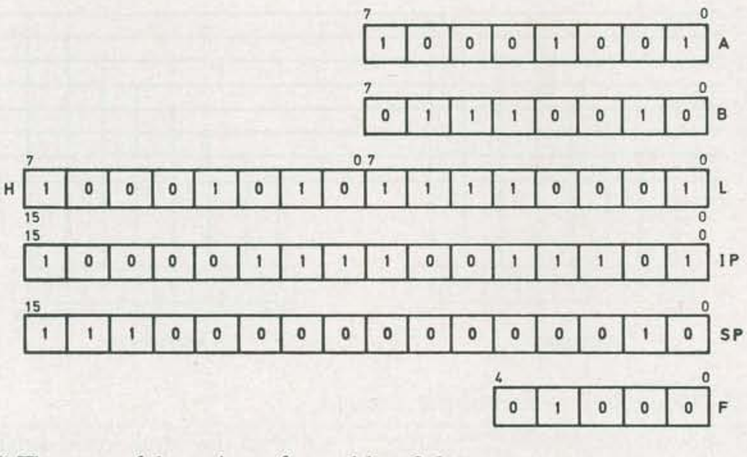
Fig. 2.5. IMP's register model.

General purpose registers

Apart from the accumulator (A), IMP has three 8-bit registers which may be classed as "general purpose". These are B, H and L. Register B is often used as an "alternative accumulator", results which appear in the accumulator being regularly transferred to and from register B. The H and L registers can be used as individual 8-bit registers and may also be used "end- on" to provide a 16-bit general purpose register (referred to as the "HL register pair"). In this mode, HL can be used as a pointer to data stored in memory (i.e. the 16-bits in HL from an address at which data is to be stored or from which data is to be fetched). The HL register pair can thus be used as an "address pointer" and, in this context, the H register contains the "high" (most significant) byte of the address whilst the L register contains the "low" (least significant) byte of the address.

Register model

Whilst Fig. 2.2 provides us with some idea of IMP's internal arrangement, it is unnecessarily complex from the point of view of the programmer. The programmer is neither concerned with the links between registers nor need he/she be aware of internal features over which he/she has no direct control. In this context, the "register model" depicted in Fig. 2.5 provides a more useful representation of IMP and this merely shows the registers which are directly accessible to the programmer and over which the programmer has control.



7 0
1 0 0 0 1 0 0 1 A
7 0
0 1 1 1 0 0 1 0 B
7 0 7 0
H 1 0 0 0 1 0 1 0 1 1 1 1 0 0 0 1 L
15 0
15 0
1 0 0 0 0 1 1 1 1 0 0 1 1 1 0 1 IP
15 0
1 1 1 0 0 0 0 0 0 0 0 0 0 0 1 0 SP
4 0
0 1 0 0 0 F

Fig. 2.6. The state of the registers for problem 2.1.

Problem 2.1

Fig. 2.6 shows the state of IMP's internal registers at a particular point in the execution of a program. The MSB of each register (with the exception of the flag register) appears on the left and the layout follows that shown in the register model of Fig. 2.5.

(a) What is decimal value of the data in the accumulator?

(b) What is the hexadecimal value of the data in the accumulator?

(c) Which one of the three 8-bit general purpose registers has the GREATEST value?

(d) What hexadecimal address is pointed to by the Instruction Pointer?

(e) What decimal address is pointed to by the Stack Pointer?

(f) In which of the 8-bit registers is the MSB set?

(g) In which of the 8-bit registers is the LSB set?

(h) Which of the flags is set?

(i) Which of the flags is reset?

(j) If the HL register pair is currently being used as an address pointer, what hexadecimal address is it pointing to?

(k) Are-interrupts currently enabled or disabled?

Control Signals

IMP has five control bus signals. Four of these are inputs and one is an output. We shall briefly discuss the function of each:

Read/Write (R/W)

(output)

This line is taken low (i.e. to logic 0)

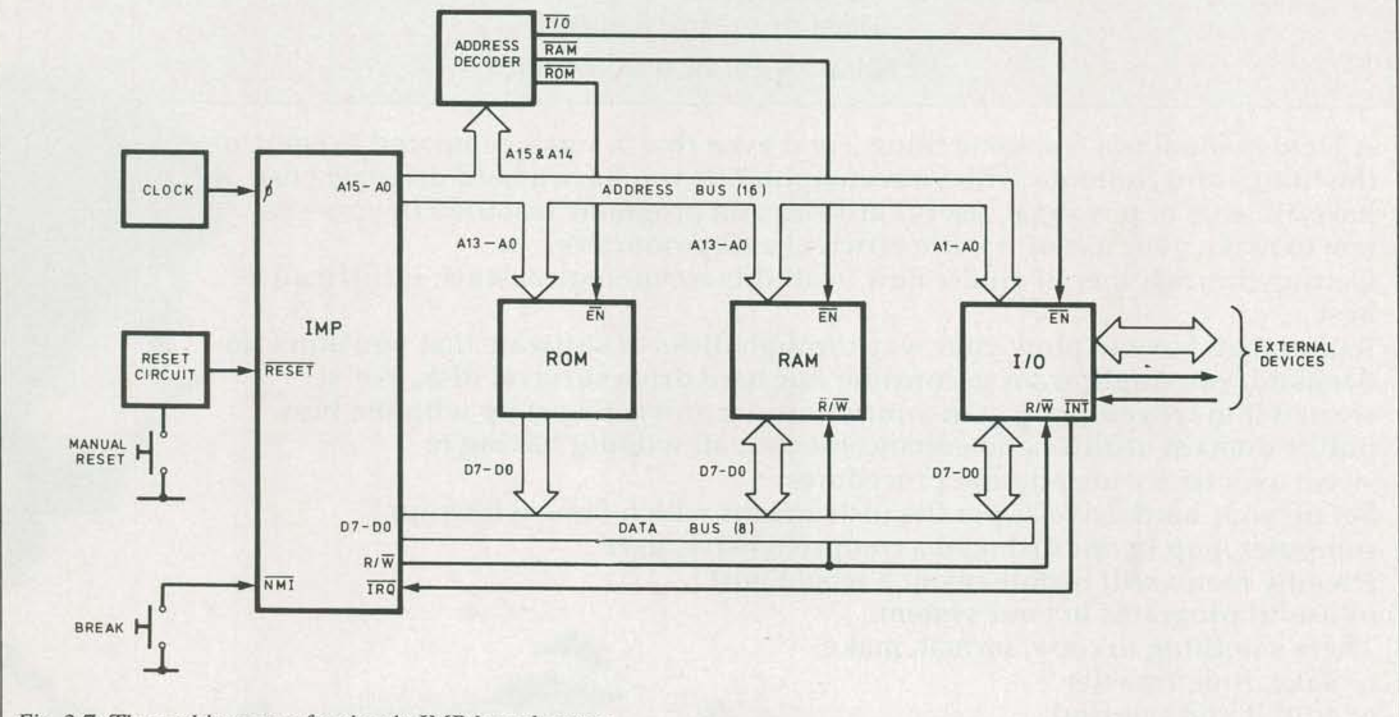


Fig. 2.7. The architecture of a simple IMP-based system.

when IMP is performing a "write" operation (e.g. when data is to be transferred from one of IMP's internal registers to an address in RAM). IMP takes the line high (i.e. to logic 1) when a "read" operation is being carried out.

[NB: Some microprocessors (e.g. Z80) have separate READ and WRITE lines.]

Interrupt request (IRQ)

(input)

This line serves as an input to the microprocessor and is taken low by an external device wishing to signal the fact that it requires attention. Provided the "interrupt flag" is reset (i.e. logic 0) this request will be honoured and the microprocessor will cease normal processing and execute the required "interrupt service routine". The interrupt line is said to be "active-low" (i.e. it is taken to logic 0 when asserted).

Non-maskable interrupt (NMI)

(input)

As we have seen, the response to an ordinary interrupt (IRQ) is determined by the interrupt status flag and thus the interrupt may be "masked". Instructions may be placed within the program which "set" or "reset" the interrupt flag hence disabling or enabling interrupts. This technique provides us with a flexible method of responding to interrupts; we can accept them or reject them at will. There are, however, some situations in which it is desirable that an interrupt should be serviced regardless of what else is going on. Hence a separate "non-maskable interrupt" line is provided. When this line is taken low, normal program execution is interrupted regardless of the state of the interrupt flag (i.e. regardless of whether interrupts are currently enabled or disabled).

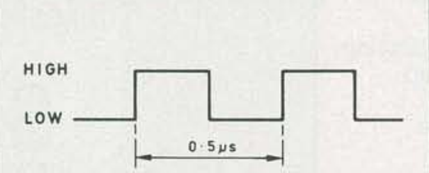


Fig. 2.8. Idealized 2MHz clock signal.

Reset (RESET)

(input)

This active low input to the microprocessor is used to initialize the system into a known state prior to normal execution of the program. When the RESET line is taken low, the program counter (PC) is placed in a defined state (by loading it with zero) and interrupts are disabled.

Clock (O)

(input)

IMP requires an accurate and stable square wave clock having a frequency of typically 2MHz. The clock is used to provide an accurate time reference for the control unit (see below).

[Many microprocessors have internal clock oscillators and merely require that a quartz crystal of appropriate resonant frequency be connected to two of the microprocessor's pins. The vast majority of 8-bit microprocessors operate with clock frequencies between 1MHz and 8MHz.]

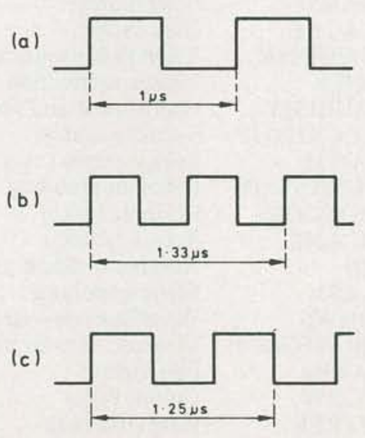


Fig. 2.9. Clock waveforms for problem 2.2.

Problem 2.3
The data sheet in Fig. 2.10 relates to the Intel 8080A microprocessor. Read the data sheet carefully (don't worry if you don't understand all of it) and answer the following questions:

(a) How many address lines are provided?

(b) How many addresses are available for input/output?

(c) What pin number is used for the RESET input?

(d) What pin number is used for the INTERRUPT input?

(e)How many clock inputs are required?

(f) Are the clock inputs TTL compatible?

(g) What supply voltages are required?

(h) Is the WRITE line "active-high", or "active-low"?

(i) What is the logical state of the WRITE line when a write operation is being carried out?

(j) From what address does execution commence when the RESET input is activated?

Fig. 2.10 Data sheet for Problem 2.3.

intel®

8080A/808A-1/8080A-2
8-BIT N-CHANNEL MICROPROCESSOR

- **TTL Drive Capability**
- **2 μs (—1:1.3 μs, —2:1.5 μs) Instruction Cycle**
- **Powerful Problem Solving Instruction Set**
- **6 General Purpose Registers and an Accumulator**
- **16-Bit Program Counter for Directly Addressing up to 64K Bytes of Memory**
- **16-Bit Stack Pointer and Stack Manipulation Instructions for Rapid Switching of the Program Environment**
- **Decimal, Binary, and Double Precision Arithmetic**
- **Ability to Provide Priority Vectored Interrupts**
- **512 Directly Addressed I/O Ports**
- **Available in EXPRESS — Standard Temperature Range**
- **Available in 40-Lead Cerdip and Plastic Packages**

The Intel® 8080A is a complete 8-bit parallel central processing unit (CPU). It is fabricated on a single LSI chip using Intel's n-channel silicon gate MOS process. This offers the user a high performance solution to control and processing applications.

The 8080A contains 6 8-bit general purpose working registers and an accumulator. The 6 general purpose registers may be addressed individually or in pairs providing both single and double precision operators. Arithmetic and logical instructions set or reset 4 testable flags. A fifth flag provides decimal arithmetic operation.

The 8080A has an external stack feature wherein any portion of memory may be used as a last in/first out stack to store/retrieve the contents of the accumulator, flags, program counter, and all of the 6 general purpose registers. The 16-bit stack pointer controls the addressing of this external stack. This stack gives the 8080A the ability to easily handle multiple level priority interrupts by rapidly storing and restoring processor status. It also provides almost unlimited subroutine nesting.

This microprocessor has been designed to simplify systems design. Separate 16-line address and 8-line bidirectional data busses are used to facilitate easy interface to memory and I/O. Signals to control the interface to memory and I/O are provided directly by the 8080A. Ultimate control of the address and data busses resides with the HOLD signal. It provides the ability to suspend processor operation and force the address and data busses into a high impedance state. This permits OR-tying these busses with other controlling devices for (DMA) direct memory access or multi-processor operation.

NOTE:
The 8080A is functionally and electrically compatible with the Intel® 8080.

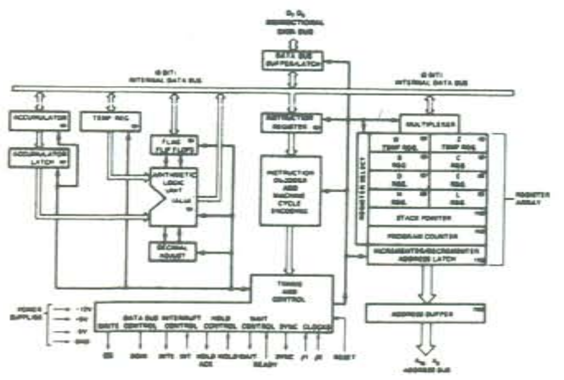


Figure 1. Block Diagram

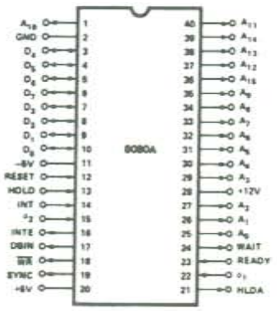


Figure 2. Pin Configuration

Table 1. Pin Description

Symbol	Type	Name and Function
A_{15}-A_0	O	**Address Bus:** The address bus provides the address to memory (up to 64K 8-bit words) or denotes the I/O device number for up to 256 input and 256 output devices. A_0 is the least significant address bit.
D_7-D_0	I/O	**Data Bus:** The data bus provides bi-directional communication betweeen the CPU, memory, and I/O devices for instructions and data transfers. Also, during the first clock cycle of each machine cycle, the 8080A outputs a status word on the data bus that describes the current machine cycle. D_0 is the least significant bit.
SYNC	O	**Synchronizing Signal:** The SYNC pin provides a signal to indicate the beginning of each machine cycle.
DBIN	O	**Data Bus In:** The DBIN signal indicates to external circuits that the data bus is in the input mode. This signal should be used to enable the gating of data onto the 8080A data bus from memory or I/O.
READY	I	**Ready:** The READY signal indicates to the 8080A that valid memory or input data is available on the 8080A data bus. This signal is used to synchronize the CPU with slower memory or I/O devices. If after sending an address out the 8080A does not receive a READY input, the 8080A will enter a WAIT state for as long as the READY line is low. READY can also be used to single step the CPU.
WAIT	O	**Wait:** The WAIT signal acknowledges that the CPU is in a WAIT state.
$\overline{WR}$	O	**Write:** The $\overline{WR}$ signal is used for memory WRITE or I/O output control. The data on the data bus is stable while the $\overline{WR}$ signal is active low ($\overline{WR}$ = 0).
HOLD	I	**Hold:** The HOLD signal requests the CPU to enter the HOLD state. The HOLD state allows an external device to gain control of the 8080A address and data bus as soon as the 8080A has completed its use of these busses for the current machine cycle. It is recognized under the following conditions: • the CPU is in the HALT state. • the CPU is in the T2 or TW state and the READY signal is active. As a result of entering the HOLD state the CPU ADDRESS BUS (A_{15}-A_0) and DATA BUS (D_7-D_0) will be in their high impedance state. The CPU acknowledges its state with the HOLD ACKNOWLEDGE (HLDA) pin.
HLDA	O	**Hold Acknowledge:** The HLDA signal appears in response to the HOLD signal and indicates that the data and address bus will go to the high impedance state. The HLDA signal begins at: • T3 for READ memory or input. • The Clock Period following T3 for WRITE memory or OUTPUT operation. In either case, the HLDA signal appears after the rising edge of ϕ_2.
INTE	O	**Interrupt Enable:** Indicates the content of the internal interrupt enable flip/flop. This flip/flop may be set or reset by the Enable and Disable Interrupt instructions and inhibits interrupts from being accepted by the CPU when it is reset. It is automatically reset (disabling further interrupts) at time T1 of the instruction fetch cycle (M1) when an interrupt is accepted and is also reset by the RESET signal.
INT	I	**Interrupt Request:** The CPU recognizes an interrupt request on this line at the end of the current instruction or while halted. If the CPU is in the HOLD state or if the Interrupt Enable flip/flop is reset it will not honor the request.
RESET[1]	I	**Reset:** While the RESET signal is activated, the content of the program counter is cleared. After RESET, the program will start at location 0 in memory. The INTE and HLDA flip/flops are also reset. Note that the flags, accumulator, stack pointer, and registers are not cleared.
V_{SS}		**Ground:** Reference.
V_{DD}		**Power:** +12 ±5% Volts.
V_{CC}		**Power:** +5 ±5% Volts.
V_{BB}		**Power:** −5 ±5% Volts.
ϕ_1, ϕ_2		**Clock Phases:** 2 externally supplied clock phases. (non TTL compatible)

A complete IMP Microcomputer System

Fig. 2.7 shows the internal architecture of a complete microcomputer based on the IMP. Since this diagram is somewhat more complicated than that in Fig. 1 of Part 1, we shall attempt to justify the additional features which have appeared.

The reset circuitry is designed to take IMP's RESET input low for a short time (typically 20ms) when the power is first applied to the system or when the manual reset button is pressed. This ensures that the system initializes itself in an orderly fashion as IMP always commences program execution from address 0000H when its RESET input is taken low.

In order that data flow within the system is orderly and that there is no uncertainty as to whether the data present is valid or not, it is necessary to synchronize all data transfers using a reference clock signal. This signal, a symmetrical square wave, is generated by an external oscillator. For accuracy and stability, the clock is crystal controlled the functions at a fixed frequency of 2MHz.

The relationship between the frequency and periodic time (period) of a microprocessor clock is given by:

$$f = \frac{1}{t} \quad \text{or} \quad t = \frac{1}{f}$$

Where f is the frequency (in Hz) and t is the periodic time (in seconds). In practice, it is often more convenient to work in terms of MHz and us, and th same formula will apply. As an example, suppose the clock in Fig. 2 operates at 2MHz. Its periodic time (period) will be 1/2 us (i.e. 0.5us or 500ns) and its idealized waveform is shown in Fig. 2.8.

The clock cycle (often known as a T-state) is the fundamental timing interval used by the microprocessor. A "machine cycle" (M-cycle) is the smallest indivisible unit of microprocessor activity and usually comprises between three and five T-states. An instruction cycle (i.e. that associated with fetching an instruction, decoding and executing it) normally requires between one and five M-cycles.

To put this into context, suppose that IMP is operating at its maximum clock frequency of 4MHz. The periodic time of the clock (T-state) will be 250ns. A machine cycle (M-cycle) will then occupy from 0.7us to 1.25us whereas an instruction cycle will require some 1.25us to 6.25us depending upon its complexity. To put this another way, IMP is capable of executing between 160,000 and 800,000 instructions every second!

The two most significant address lines are fed to an address decoder which generates active low signals to enable the ROM, RAM and I/O devices (more of this in Part 5). At this stage it is merely necessary for readers to understand that ONLY one of these devices is enabled (i.e. linked to the data bus) at any particular time.

The all-important "break" key is connected to IMP's non- maskable interrupt (NMI) input. This allows the user to regain control WITHOUT having to reset the system (and erase the data and/or program currently present in RAM). The general topic of interrupts is outside the scope of *Introductory Microprocessors* and therefore readers need not at this stage concern themselves with the action which takes place when an interrupt is received.

Problem 2.2

What is the frequency of each of the microprocessor clock signals depicted in Fig. 2.9?

Glossary for Part Two

Accumulator
One or more registers associated with the ALU which temporarily store the results of ALU operations.

Arithmetic Logic Unit (ALU)
One of the essential elements of a CPU. The ALU performs various forms of addition, subtraction and logical operations.

Buffer
A hardware device which provides isolation between two parts of a circuit and which usually increases the drive capability of a signal. In the context of software, the word refers to a contiguous area of memory used for temporary storage of data when performing I/O.

Clock
The clock provides a reference timing source within a microcomputer system. The output of a clock comprises regular pulses of accurately defined frequency and period.

Flag
A bit contained within a flag (or status) register which indicates the internal status of the microprocessor or which signals the outcome of an ALU operation.

Instruction
A single command within a program. A complete sequence of instructions constitutes a program.

Interrupt
A signal generated by an external device which requires the services of the microprocessor or which needs to alert the microprocessor to a particular condition (such as imminent power failure).

Stack
The stack is a contiguous area of read/write memory that is accessed on a last-in first-out (LIFO) basis by the microprocessor and used for temporary storage of data and addresses.

Answers to Problems

2.1 (a) 137
(b) 89
(c) L
(d) 879D
(e) 57346
(f) A, H and L
(g) A and L
(h) N
(i) C, Z, V and I
(j) 8AF1
(k) enabled

2.2 (a) 1MHz
(b) 1.5MHz
(c) 1.2MHz

2.3 (a) 16
(b) 256 for input and 256 for output
(c) 12
(d) 14
(e) 2
(f) no
(g) +5V, +12V and -5V
(h) active-low
(i) 0
(j) 0(0000H)