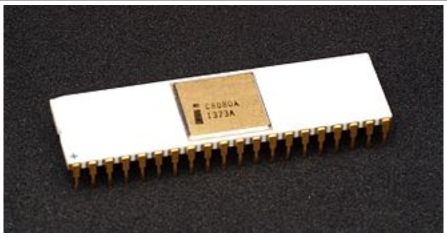
WIKIPEDIA
The Free Encyclopedia

Intel 8080

The **Intel 8080** (*"eighty-eighty"*) is the second 8-bit microprocessor designed and manufactured by Intel. It first appeared in April 1974 and is an extended and enhanced variant of the earlier 8008 design, although without binary compatibility.[2] The initial specified clock rate or frequency limit was 2 MHz, with common instructions using 4, 5, 7, 10, or 11 cycles. As a result, the processor is able to execute several hundred thousand instructions per second. Two faster variants, the 8080A-1 (sometimes referred to as the 8080B) and 8080A-2, became available later with clock frequency limits of 3.125 MHz and 2.63 MHz respectively.[3] The 8080 needs two support chips to function in most applications: the i8224 clock generator/driver and the i8228 bus controller. It is implemented in N-type metal–oxide–semiconductor logic (NMOS) using non-saturated enhancement mode transistors as loads[4][5] thus demanding a +12 V and a −5 V voltage in addition to the main transistor–transistor logic (TTL) compatible +5 V.

Intel 8080


An Intel C8080A processor variant with white ceramic package, solder seal metal lid, and gold pins.

General information	
Launched	April 1974
Discontinued	1990[1]
Marketed by	Intel
Designed by	Intel
Common manufacturer(s)	Intel
Performance	
Max. CPU clock rate	2 MHz to 3.125 MHz
Data width	8 bits
Address width	16 bits
Architecture and classification	
Technology node	6 µm
Instruction set	8080
Physical specifications	
Transistors	4,500
Cores	1
Package(s)	40-pin DIP
Socket(s)	DIP40
History	
Predecessor	Intel 8008
Successor	Intel 8085
Support status	
Unsupported	

Although earlier microprocessors were commonly used in mass-produced devices such as calculators, cash registers, computer terminals, industrial robots,[6] and other applications, the 8080 saw greater success in a wider set of applications, and is largely credited with starting the microcomputer industry.[7] Several factors contributed to its popularity: its 40-pin package made it easier to interface than the 18-pin 8008, and also made its data bus more efficient; its NMOS implementation gave it faster transistors than those of the P-type metal–oxide–semiconductor logic (PMOS) 8008, while also simplifying interfacing by making it TTL-compatible; a wider variety of support chips were available; its instruction set was enhanced over the 8008;[8] and its full 16-bit address bus (versus the 14-bit one of the 8008) enabled it to access 64 KB of memory, four times more than the 8008's range of 16 KB. It was used in the Altair 8800 and subsequent S-100 bus personal computers until it was replaced by the Z80 in this role, and was the original target CPU for CP/M operating systems developed by Gary Kildall.

The 8080 directly influenced the later x86 architecture. Intel designed the 8086 to have its assembly language be similar enough to the 8080, with most instructions mapping directly onto each other, that transpiled 8080 assembly code could be executed on the 8086.[9]

History

Microprocessor customers were reluctant to adopt the 8008 because of limitations such as the single addressing mode, low clock speed, low pin count, and small on-chip stack, which restricted the scale and complexity of software. There were several proposed designs for the 8080, ranging from simply adding stack instructions to the 8008 to a complete departure from all previous Intel architectures.[10] The final design was a compromise between the proposals.

Federico Faggin, the originator of the 8080 architecture in early 1972, proposed the chip to Intel's management and pushed for its implementation. He finally got the permission to develop it six months later. Faggin hired Masatoshi Shima, who helped design the 4004 with him, from Japan in November 1972. Shima did the detailed design under Faggin's direction,[11] using the design methodology for random logic with silicon gate that Faggin had created for the 4000 family.

The 8080 was explicitly designed to be a general-purpose microprocessor for a larger number of customers. Much of the development effort was spent trying to integrate the functionalities of the 8008's supplemental chips into one package. It was decided early in development that the 8080 was not to be binary-compatible with the 8008, instead opting for source compatibility once run through a transpiler, to allow new software to not be subject to the same restrictions as the 8008. For the same reason, as well as the expand the capabilities of stack-based routines and interrupts, the stack was moved to external memory.

Noting the specialized use of general-purpose registers by programmers in mainframe systems, Stanley Mazor, the chip architect, decided the 8080's registers would be specialized, with register pairs having a different set of uses.[12] This also allowed the engineers to more effectively use transistors for other purposes.

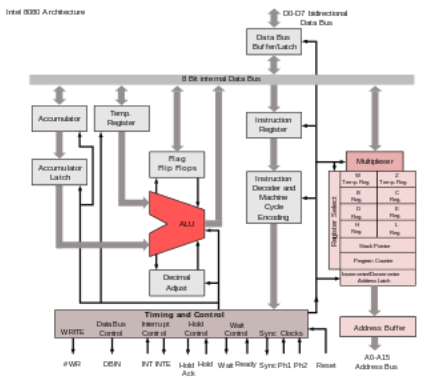
Shima finished the layout in August 1973. After the regulation of NMOS fabrication, a prototype of the 8080 was completed in January 1974. It had a flaw, in that driving with standard TTL devices increased the ground voltage because high current flowed into the narrow line. Intel had already produced 40,000 units of the 8080 at the direction of the sales section before Shima characterized the prototype. It was released as requiring Low-power Schottky TTL (LS TTL) devices. The 8080A fixed this flaw.[13]

Intel offered an instruction set simulator for the 8080 named INTERP/80 to run compiled PL/M programs. It was written by Gary Kildall while he worked as a consultant for Intel.[14]

Description

Programming model

The Intel 8080 is the successor to the 8008. It uses the same basic instruction set and register model as the 8008, although it is neither source code compatible nor binary code compatible with its predecessor. Every instruction in the 8008 has an equivalent instruction in the 8080. The 8080 also adds 16-bit operations in its instruction set. Whereas the 8008 required the use of the HL register pair to indirectly access its 14-bit memory space, the 8080 added addressing modes to allow direct access to its full 16-bit memory space. The internal 7-level push-down call stack of the 8008 was replaced by a dedicated 16-bit stack-pointer (SP) register. The 8080's 40-pin DIP packaging permits it to provide a 16-bit address bus and an 8-bit data bus, enabling access to 64 KiB (2^{16} bytes) of memory.



i8080 microarchitecture

Registers

Intel 8080 registers

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	*(bit position)*
Main registers																

The processor has seven 8-bit registers (A, B, C, D, E, H, and L), where A is the primary 8-bit accumulator. The other six registers can be used as either individual 8-bit registers or in three 16-bit register pairs (BC, DE, and HL, referred to as B, D and H in Intel documents) depending on the particular instruction. Some instructions also enable the HL register pair to be used as a (limited) 16-bit accumulator. A pseudo-register M, which refers to the dereferenced memory location pointed to by HL, can be used almost anywhere other registers can be used. The 8080 has a 16-bit stack pointer to memory, replacing the 8008's internal stack, and a 16-bit program counter.

A	Flags	**P**rogram **S**tatus **W**ord
B	C	**B**
D	E	**D**
H	L	**H** (indirect address)
Index registers		
SP		**S**tack **P**ointer
Program counter		
PC		**P**rogram **C**ounter
Status register		
	S Z - AC - P - C	Flags

Flags

The processor maintains internal flag bits (a status register), which indicate the results of arithmetic and logical instructions. Only certain instructions affect the flags. The flags are:

- Sign (S), set if the result is negative.
- Zero (Z), set if the result is zero.
- Parity (P), set if the number of 1 bits in the result is even.
- Carry (C), set if the last addition operation resulted in a carry or if the last subtraction operation required a borrow
- Auxiliary carry (AC or H), used for binary-coded decimal arithmetic (BCD).

The carry bit can be set or complemented by specific instructions. Conditional-branch instructions test the various flag status bits. The flags can be copied as a group to the accumulator. The A accumulator and the flags together are called the PSW register, or program status word.

Commands, instructions

As with many other 8-bit processors, all instructions are encoded in one byte (including register numbers, but excluding immediate data), for simplicity. Some can be followed by one or two bytes of data, which can be an immediate operand, a memory address, or a port number. Like more advanced processors, it has automatic CALL and RET instructions for multi-level procedure calls and returns (which can even be conditionally executed, like jumps) and instructions to save and restore any 16-bit register pair on the machine stack. Eight one-byte call instructions (`RST`) for subroutines exist at the fixed addresses 00h, 08h, 10h, ..., 38h. These are intended to be supplied by external hardware in order to invoke a corresponding interrupt service routine, but are also often employed as fast system calls. The instruction that executes slowest is `XTHL`, which is used for exchanging the register pair HL with the value stored at the address indicated by the stack pointer.

8-bit instructions

All 8-bit operations with two operands can only be performed on the 8-bit accumulator (the A register). The other operand can be either an immediate value, another 8-bit register, or a memory byte addressed by the 16-bit register pair HL. Increments and decrements can be performed on any 8 bit register or an HL-addressed memory byte. Direct copying is supported between any two 8-bit registers and between any 8-bit register and an HL-addressed memory byte. Due to the regular encoding of the `MOV` instruction (using a quarter of available opcode space), there are redundant codes to copy a register into itself (`MOV B,B`, for instance), which are of little use, except for delays. However, the systematic opcode for `MOV M,M` is instead used to encode the halt (`HLT`) instruction, halting execution until an external reset or interrupt occurs.

16-bit operations

Although the 8080 is generally an 8-bit processor, it has limited abilities to perform 16-bit operations. Any of the three 16-bit register pairs (BC, DE, or HL, referred to as B, D, H in Intel documents) or SP can be loaded with an immediate 16-bit value (using `LXI`), incremented or decremented (using `INX` and `DCX`), or added to HL (using `DAD`). By adding HL to itself, it is possible to achieve the same result as a 16-bit arithmetical left shift with one instruction. The only 16-bit instructions that affect any flag is `DAD`, which sets the CY (carry) flag in order to allow for programmed 24-bit or 32-bit arithmetic (or larger), needed to implement floating-point arithmetic. A stack frame can be allocated using `DAD SP` and `SPHL`. A branch to a computed pointer can be executed with `PCHL`. `LHLD` loads HL from directly addressed memory and `SHLD` stores HL likewise. The `XCHG`[15] instruction exchanges the values of the HL and DE register pairs. `XTHL`exchanges last item pushed on stack with HL.

Input/output scheme

Input output port space

The 8080 supports up to 256[16] input/output (I/O) ports, accessed via dedicated I/O instructions taking port addresses as operands. This I/O mapping scheme is regarded as an advantage, as it frees up the processor's limited address space. Many CPU architectures instead use so-called memory-mapped I/O (MMIO), in which a common address space is used for both RAM and peripheral chips. This removes the need for dedicated I/O instructions, although a drawback in such designs may be that special hardware must be used to insert wait states, as peripherals are often slower than memory. However, in some simple 8080 computers, I/O is indeed addressed as if they were memory cells, "memory-mapped", leaving the I/O commands unused. I/O addressing can also sometimes employ the fact that the processor outputs the same 8-bit port address to both the lower and the higher address byte (i.e., `IN 05h` would put the address 0505h on the 16-bit address bus). Similar I/O-port schemes are used in the backward-compatible Zilog Z80 and Intel 8085, and the closely related x86 microprocessor families.

Separate stack space

One of the bits in the processor state word (see below) indicates that the processor is accessing data from the stack. Using this signal, it is possible to implement a separate stack memory space. This feature is seldom used.

The internal state word

For more advanced systems, during one phase of its working loop, the processor set its "internal state byte" on the data bus. This byte contains flags that determine whether the memory or I/O port is accessed and whether it is necessary to handle an interrupt.

The interrupt system state (enabled or disabled) is also output on a separate pin. For simple systems, where the interrupts are not used, it is possible to find cases where this pin is used as an additional single-bit output port (the popular Radio-86RK computer made in the Soviet Union, for instance).

Example code

The following 8080/8085 assembler source code is for a subroutine named `memcpy` that copies a block of data bytes of a given size from one location to another. The data block is copied one byte at a time, and the data movement and looping logic utilizes 16-bit operations.

```
                 ; memcpy --
                 ; Copy a block of memory from one location to another.
                 ;
                 ; Entry registers
                 ;       BC - Number of bytes to copy
                 ;       DE - Address of source data block
                 ;       HL - Address of target data block
                 ;
                 ; Return registers
                 ;       BC - Zero

1000                         org     1000h       ;Origin at 1000h
1000             memcpy      public
1000   78                    mov     a,b         ;Copy register B to register A
1001   B1                    ora     c           ;Bitwise OR of A and C into register A
1002   C8                    rz                  ;Return if the zero-flag is set high.
1003   1A        loop:       ldax    d           ;Load A from the address pointed by DE
1004   77                    mov     m,a         ;Store A into the address pointed by HL
1005   13                    inx     d           ;Increment DE
1006   23                    inx     h           ;Increment HL
1007   0B                    dcx     b           ;Decrement BC   (does not affect Flags)
1008   78                    mov     a,b         ;Copy B to A    (so as to compare BC with zero)
1009   B1                    ora     c           ;A = A | C      (are both B and C zero?)
100A   C2 03 10              jnz     loop        ;Jump to 'loop:' if the zero-flag is not set.
100D   C9                    ret                 ;Return
```

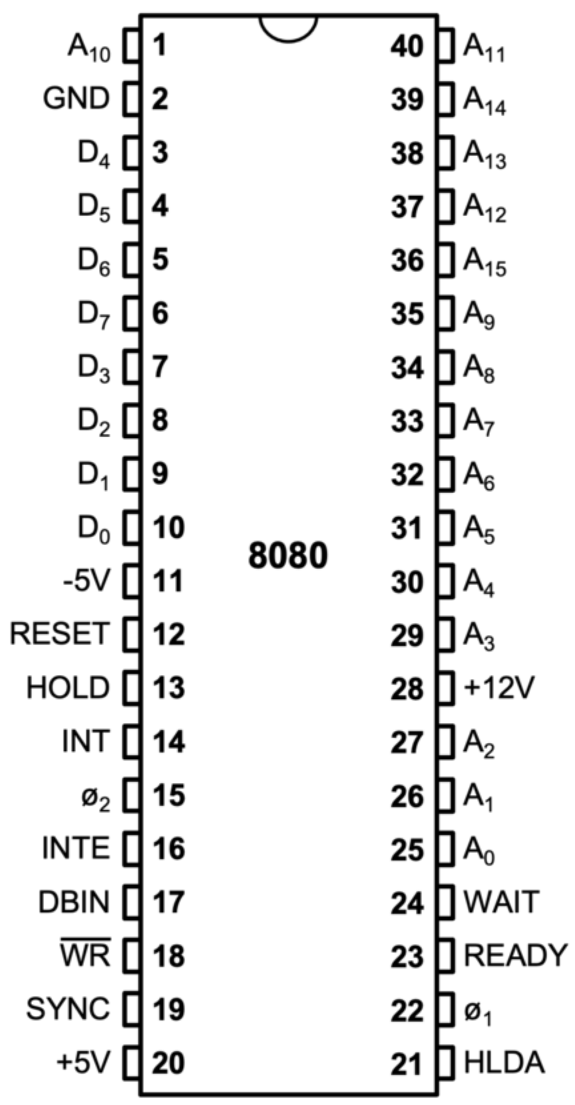
Pin use

The address bus has its own 16 pins, and the data bus has 8 pins that are usable without any multiplexing. Using the two additional pins (read and write signals), it is possible to assemble simple microprocessor devices very easily. Only the separate IO space, interrupts, and DMA need added chips to decode the processor pin signals. However, the pin load capacity is limited; even simple computers often require bus amplifiers.

The processor needs three power sources (−5, +5, and +12 V) and two non-overlapping high-amplitude synchronizing signals. However, at least the late Soviet version KP580BM80A was able to work with a single +5 V power source, the +12 V pin being connected to +5 V and the −5 V pin to ground.

The pin-out table, from the chip's accompanying documentation, describes the pins as follows:

Pin number	Signal	Type	Comment
1	A10	Output	Address bus 10
2	GND	—	Ground
3	D4	Bidirectional	Bidirectional data bus. The processor also transiently sets here the "processor state", providing information about what the processor is currently doing: ▪ D0 reading interrupt command. In response to the interrupt signal, the processor is reading and executing a single arbitrary command with this flag raised. Normally the supporting chips provide the subroutine call command (CALL or RST), transferring control to the interrupt handling code. ▪ D1 reading (low level means writing) ▪ D2 accessing stack (probably a separate stack memory space was initially planned) ▪ D3 doing nothing, has been halted by the HLT instruction ▪ D4 writing data to an output port ▪ D5 reading the first byte of an executable instruction ▪ D6 reading data from an input port ▪ D7 reading data from memory
4	D5		
5	D6		
6	D7		
7	D3		
8	D2		
9	D1		
10	D0		
11	−5 V	—	The −5 V power supply. This must be the first power source connected and the last disconnected, otherwise the



8080 Pinout

			processor will be damaged.
12	RESET	Input	Reset. The signal forces execution of commands located at address 0000. The content of other processor registers is not modified. This is an inverting input (the active level being logical 0)
13	HOLD	Input	Direct memory access request. The processor is requested to switch the data and address bus to the high impedance ("disconnected") state.
14	INT	Input	Interrupt request
15	φ2	Input	The second phase of the clock generator signal
16	INTE	Output	The processor has two commands for setting 0 or 1 level on this pin. The pin normally is supposed to be used for interrupt control. However, in simple computers it was sometimes used as a single bit output port for various purposes.
17	DBIN	Output	Read (the processor reads from memory or input port)
18	WR	Output	Write (the processor writes to memory or output port). This is an inverted output, the active level being logical zero.
19	SYNC	Output	Active level indicates that the processor has put the "state word" on the data bus. The various bits of this state word provide added information to support the separate address and memory spaces, interrupts, and direct memory access. This signal is required to pass through additional logic before it can be used to write the processor state word from the data bus into some external register, e.g., 8238 (http://www.datasheets360.com/pdf/-4828066515233335508)-System Controller and Bus Driver.
20	+5 V	—	The + 5 V power supply
21	HLDA	Output	Direct memory access confirmation. The processor switches data and address pins into the high impedance state, allowing another device to manipulate the bus
22	φ1	Input	The first phase of the clock generator signal
23	READY	Input	Wait. With this signal it is possible to suspend the processor's work. It is also used to support the hardware-based step-by step debugging mode.
24	WAIT	Output	Wait (indicates that the processor is in the waiting state)
25	A0	Output	Address bus
26	A1		
27	A2		
28	12 V	—	The +12 V power supply. This must be the *last* connected and first disconnected power source.

29	A3	Output	The address bus; can switch into high impedance state on demand
30	A4		
31	A5		
32	A6		
33	A7		
34	A8		
35	A9		
36	A15		
37	A12		
38	A13		
39	A14		
40	A11		

Support chips

A key factor in the success of the 8080 was the broad range of support chips available, providing serial communications, counter/timing, input/output, direct memory access, and programmable interrupt control amongst other functions:

- 8238 (http://www.datasheets360.com/pdf/-4828066515233335508) – System controller and bus driver
- 8251 – Communication controller
- 8253 – Programmable interval timer
- 8255 – Programmable peripheral interface
- 8257 – DMA controller
- 8259 – Programmable interrupt controller

Physical implementation

The 8080 integrated circuit uses non-saturated enhancement-load nMOS gates, demanding extra voltages (for the load-gate bias). It was manufactured in a silicon gate process using a minimal feature size of 6 µm. A single layer of metal is used to interconnect the approximately 4,500 transistors[17] in the design, but the higher resistance polysilicon layer, which required higher voltage for some interconnects, is implemented with transistor gates. The die size is approximately 20 mm^2.

Industrial impact

Applications and successors

The 8080 is used in many early microcomputers, such as the MITS Altair 8800 Computer, Processor Technology SOL-20 Terminal Computer and IMSAI 8080 Microcomputer, forming the basis for machines running the CP/M operating system (the later, almost fully compatible and more able, Zilog Z80 processor would capitalize on this, with Z80 and CP/M becoming the dominant CPU and OS combination of the period circa 1976 to 1983 much as did the x86 and DOS for the PC a decade later).

In 1979, even after the introduction of the Z80 and 8085 processors, five manufacturers of the 8080 were selling an estimated 500,000 units per month at a price around $3 to $4 each.[18]

The first single-board microcomputers, such as MYCRO-1 and the *dyna-micro* / MMD-1 (see: Single-board computer) were based on the Intel 8080. One of the early uses of the 8080 was made in the late 1970s by Cubic-Western Data of San Diego, California, in its Automated Fare Collection Systems custom designed for mass transit systems around the world. An early industrial use of the 8080 is as the "brain" of the DatagraphiX Auto-COM (Computer Output Microfiche) line of products which takes large amounts of user data from reel-to-reel tape and images it onto microfiche. The Auto-COM instruments also include an entire automated film cutting, processing, washing, and drying sub-system. Several early video arcade games were built around the 8080 microprocessor, including *Space Invaders*, one of the most popular arcade games ever made.

Zilog introduced the Z80, which has a compatible machine language instruction set and initially used the same assembly language as the 8080, but for legal reasons, Zilog developed a syntactically-different (but code compatible) alternative assembly language for the Z80. At Intel, the 8080 was followed by the compatible and electrically more elegant 8085.

Later, Intel issued the assembly-language compatible (but not binary-compatible) 16-bit 8086 and then the 8/16-bit 8088, which was selected by IBM for its new PC to be launched in 1981. Later NEC made the NEC V20 (an 8088 clone with Intel 80186 instruction set compatibility) which also supports an 8080 emulation mode. This is also supported by NEC's V30 (a similarly enhanced 8086 clone). Thus, the 8080, via its instruction set architecture (ISA), made a lasting impact on computer history.

A number of processors compatible with the Intel 8080A were manufactured in the Eastern Bloc: the KR580VM80A (initially marked as КР580ИК80) in the Soviet Union, the MCY7880[19] made by Unitra CEMI in Poland, the MHB8080A[20] made by TESLA in Czechoslovakia, the 8080APC[20] made by Tungsram / MEV in Hungary, and the MMN8080[20] made by Microelectronica Bucharest in Romania.

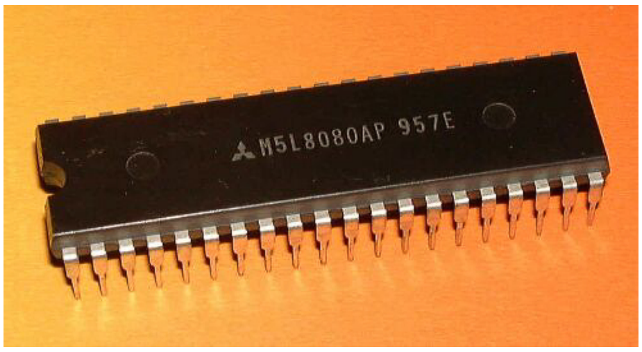
As of 2017, the 8080 is still in production at Lansdale Semiconductors.[21]

Intel 8080 second sources

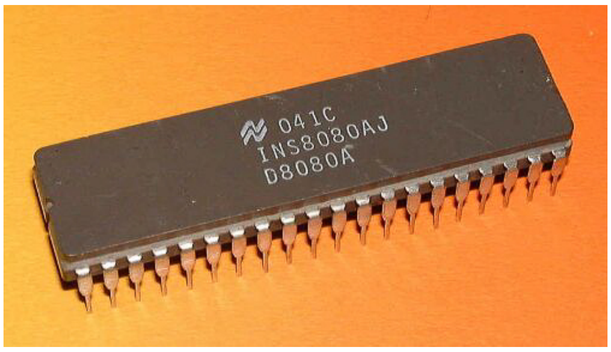
AMD Am9080

CEMI MCY7880 (Poland)

Kvazar Kiev K580IK80 (Soviet Union)

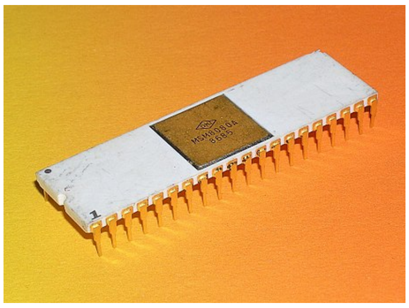


Mitsubishi Electric M5L8080

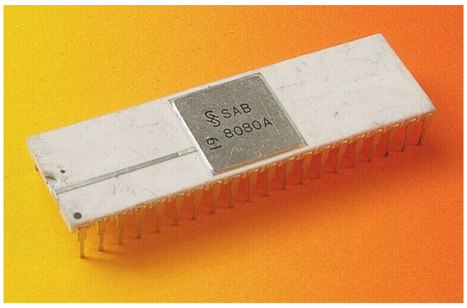


National Semiconductor INS8080

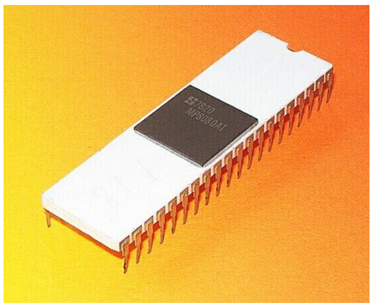
NEC μPD8080AF



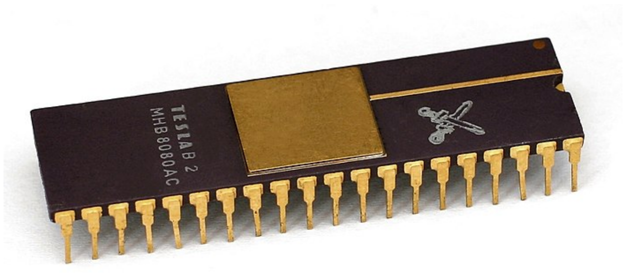
OKI MSM8080



Siemens SAB8080



Signetics MP8080



Tesla MHB8080

Texas Instruments TMS8080

5G8080 (PR China)

Industry change

The 8080 also changed how computers were created. When the 8080 was introduced, computer systems were usually created by computer manufacturers such as Digital Equipment Corporation, Hewlett Packard, or IBM. A manufacturer would produce the whole computer, including processor, terminals, and system software such as compilers and operating system. The 8080 was designed for almost any application *except* a complete computer system. Hewlett Packard developed the HP 2640 series of smart terminals around the 8080. The HP 2647 is a terminal which runs the programming language BASIC on the 8080. Microsoft's founding product, Microsoft BASIC, was originally programmed for the 8080.

The 8080 and 8085 gave rise to the 8086, which was designed as a source code compatible, albeit not binary compatible, extension of the 8080.[22] This design, in turn, later spawned the x86 family of chips, which continue to be Intel's primary line of processors. Many of the 8080's core machine instructions and concepts survive in the widespread x86 platform. Examples include the registers named *A*, *B*, *C*, and *D* and many of the flags used to control conditional jumps. 8080 assembly code can still be directly translated into x86 instructions, since all of its core elements are still present.

Cultural impact

- Asteroid 8080 Intel is named as a pun and praise on the name of Intel 8080.[23]
- Microsoft's published phone number, 425-882-8080, was chosen because much early work was on this chip.
- Many of Intel's main phone numbers also take a similar form: xxx-xxx-8080

See also

- CP/M – operating system
- S-100 bus
- MPT8080

References

1. "CPU History – The CPU Museum – Life Cycle of the CPU" (https://web.archive.org/web/20100116143420/http://www.cpushack.net/life-cycle-of-cpu.html). Archived from the original (http://www.cpushack.com/life-cycle-of-cpu.html) on January 16, 2010.
2. "From CPU to software, the 8080 Microcomputer is here". *Electronic News*. New York: Fairchild Publications. April 15, 1974. pp. 44–45. *Electronic News* was a weekly trade newspaper. The same advertisement appeared in the May 2, 1974, issue of *Electronics* magazine.
3. "8080A/8080A-1/8080A-2 8-Bit N Channel Microprocessor" (http://www.elektronikjk.com/technika_komputerowa/CPU/Intel_8080A.pdf) (PDF). Intel. Archived (https://web.archive.org/web/20211115165927/http://www.elektronikjk.com/technika_komputerowa/CPU/Intel_8080A.pdf) (PDF) from the original on November 15, 2021. Retrieved November 16, 2021.
4. similar to *pull-up resistors*
5. Tohya, Hirokazu (2013). *Switching Mode Circuit Analysis and Design: Innovative Methodology by Novel Solitary Electromagnetic Wave Theory* (https://books.google.com/books?id=ILceDgAAQBAJ&q=8080+non-saturated+enhancement&pg=PA4). Bentham Science Publishers. p. 4. ISBN 9781608054497. Archived (https://web.archive.org/web/20211115163542/https://books.google.com/books?id=ILceDgAAQBAJ&q=8080+non-saturated+enhancement&pg=PA4) from the original on November 15, 2021. Retrieved November 28, 2020.
6. The 8008 (1972) was used for interpolation and control in ASEA's (now ABB) first line of general industrial robots, introduced October 1973.
7. Mueller, Scott (2006). *Upgrading and Reparing PCs* (https://www.informit.com/articles/article.aspx?p=482324&seqNum=2) (17th ed.). Pearson Education. p. 37. ISBN 978-0-7897-3404-4. Archived (https://web.archive.org/web/20211116001341/https://www.informit.com/articles/article.aspx?p=482324&seqNum=2) from the original on November 16, 2021. Retrieved November 16, 2021.
8. The enhancements were largely based on customer feedback and Federico Faggin and others listening to minicomputer-oriented professionals about certain problems and lack of features in the 8008 architecture. (Source: 8008 and 8080 oral histories.)

9. Mazor, Stanley (June 1978). "The Intel 8086 Microprocessor: a 16-bit Evolution of the 8080" (https://ieeexplore.ieee.org/document/5430762). *IEEE Computer*. **11** (6): 18–27. doi:10.1109/C-M.1978.218219 (https://doi.org/10.1109%2FC-M.1978.218219). S2CID 16962774 (https://api.semanticscholar.org/CorpusID:16962774). Archived (https://web.archive.org/web/20210919153639/https://ieeexplore.ieee.org/document/5430762) from the original on September 19, 2021. Retrieved November 18, 2021.
10. Miller, Michael. "Creating the 8080: The Processor That Started the PC Revolution" (https://www.pcmag.com/news/creating-the-8080-the-processor-that-started-the-pc-revolution). *PCMag*. Zaff Davis. Archived (https://web.archive.org/web/20211114174610/https://www.pcmag.com/news/creating-the-8080-the-processor-that-started-the-pc-revolution) from the original on November 14, 2021. Retrieved November 14, 2021.
11. Faggin, Federico. "8008 and 8080 Q&A" (https://sites.google.com/site/microprocessorintel4004/8008-8080-q-a). *Microprocessor Intel 4004*. Archived (https://web.archive.org/web/20211115233233/https://sites.google.com/site/microprocessorintel4004/8008-8080-q-a) from the original on November 15, 2021. Retrieved November 15, 2021.
12. Mazor, Stanley (April–June 2007). "Intel 8080 CPU Chip Development". *IEEE Annals of the History of Computing*. **29** (2): 70–73. doi:10.1109/MAHC.2007.25 (https://doi.org/10.1109%2FMAHC.2007.25). S2CID 14755544 (https://api.semanticscholar.org/CorpusID:14755544).
13. Shima, Masatoshi; Nishimura, Hirohiko; Ishida, Haruhisa (1979). "座談会 マイクロコンピュータの誕生 開発者 嶋 正利氏に聞く". *Bit* (in Japanese). 共立出版. **11** (11): 4–12. ISSN 0385-6984 (https://www.worldcat.org/issn/0385-6984).
14. Kildall, Gary. "High-level language simplifies microcomputer programming" (https://www.retrotechnology.com/dri/kildall_highlevel_1974.pdf) (PDF). *Electronics*. McGraw-Hill Education. p. 108. Archived (https://web.archive.org/web/20211114174610/https://www.retrotechnology.com/dri/kildall_highlevel_1974.pdf) (PDF) from the original on November 14, 2021. Retrieved November 14, 2021.
15. 8080 instruction encoding (http://www.classiccmp.org/dunfield/r/8080.txt) Archived (https://web.archive.org/web/20180305170555/http://www.classiccmp.org/dunfield/r/8080.txt) March 5, 2018, at the Wayback Machine. ClassicCMP.org. Retrieved on October 23, 2011.
16. Note: Some Intel datasheets from the 1970s advertise 512 I/O ports, because they count input and output ports separately.
17. "Intel Chips timeline" (https://www.intel.com/content/dam/www/public/us/en/documents/corporate-information/history-intel-chips-timeline-poster.pdf) (PDF). *Intel*. Intel Corporation. Archived (https://web.archive.org/web/20211114170816/https://www.intel.com/content/dam/www/public/us/en/documents/corporate-information/history-intel-chips-timeline-poster.pdf) (PDF) from the original on November 14, 2021. Retrieved November 14, 2021.
18. Libes, Sol (November 1979). "Byte News". *Byte*. 11. Vol. 4. p. 82. ISSN 0360-5280 (https://www.worldcat.org/issn/0360-5280).
19. MCY7880—a Polish-made clone of 8080 (http://www.cpu-world.com/CPUs/8080/Poland-MCY7880.html) Archived (https://web.archive.org/web/20160817083511/http://www.cpu-world.com/CPUs/8080/Poland-MCY7880.html) August 17, 2016, at the Wayback Machine. CPU World. Retrieved on October 23, 2011.
20. Soviet chips and their western analogs (http://www.cpu-world.com/info/exUSSR-chips.html) Archived (https://web.archive.org/web/20170209223725/http://www.cpu-world.com/info/exUSSR-chips.html) February 9, 2017, at the Wayback Machine. CPU-world. Retrieved on October 23, 2011.
21. "Intel – Microprocessor 8080A Family & 828X Series" (http://lansdale.com/parts_reference.php?manufacturer=Intel&series=Microprocessor+8080A+Family+%26+828X+Series). Lansdale Semiconductor Inc. Archived (https://web.archive.org/web/20151014132249/http://www.lansdale.com/parts_reference.php?manufacturer=Intel&series=Microprocessor+8080A+Family+%26+828X+Series) from the original on October 14, 2015. Retrieved June 20, 2017.

22. Morse, Stephen; Ravenel, Bruce; Mazor, Stanley; Pohlman, William (October 1980). "Intel Microprocessors: 8008 to 8086" (https://stevemorse.org/8086history/8086history.pdf) (PDF). *IEEE Computer*. **13** (10): 42–60. doi:10.1109/MC.1980.1653375 (https://doi.org/10.1109%2FMC.1980.1653375). S2CID 206445851 (https://api.semanticscholar.org/CorpusID:206445851). Archived (https://web.archive.org/web/20210914205011/https://stevemorse.org/8086history/8086history.pdf) (PDF) from the original on September 14, 2021. Retrieved November 5, 2021.
23. "(8080) Intel = 1958 QC = 1987 WU2 = 1989 AS5" (https://minorplanetcenter.net/db_search/show_object?object_id=8080). *Minor Planet Center*. International Astronomical Union. Archived (https://web.archive.org/web/20190925141642/https://minorplanetcenter.net/db_search/show_object?object_id=8080) from the original on September 25, 2019. Retrieved November 14, 2021.

Further reading

- Leventhal, Lance (1978). *8080A/8085 Assembly Language Programming* (https://archive.org/details/8080a8085AssemblyLanguageProgramming/) (1st ed.). Adam Osborne & Associates.; 495 pages
- Miller, Alan (1981). *8080/Z80 Assembly Language - Techniques for Improved Programming* (https://archive.org/details/8080_and_Z-80_Assembly_Language_Techniques_1981_John_Wiley_and_Sons/) (1st ed.). John Wiley & Sons. ISBN 978-0471081241.; 332 pages
- Zaks, Rodnay; Lesea, Austin (1979). *Microprocessor Interfacing Techniques* (https://archive.org/details/MicroprocessorInterfacingTechniques_3rd_ed/) (3rd ed.). Sybex. ISBN 978-0-89588-029-1.; 466 pages
- Spracklen, Kathe (1979). *Z80 and 8080 Assembly Language Programming* (https://archive.org/details/z808080assemblyl00kath/) (1st ed.). Hayden. ISBN 978-0810451674.; 180 pages

External links

- Intel and other manufacturers' 8080 CPU images and descriptions at cpu-collection.de (http://www.cpu-collection.de/?tn=0&l0=cl&l1=8080)
- Scan of the Intel 8080 data book at DataSheetArchive.com (https://web.archive.org/web/20070928060215/http://www.datasheetarchive.com/search.php?q=8080)
- Microcomputer Design, Second Edition, 1976 (http://donbot.com/MicrocomputerDesign/SE/M001.html)
- 8080 Emulator written in JavaScript (http://www.bluishcoder.co.nz/js8080/)
- Intel 8080/KR580VM80A emulator in JavaScript (https://github.com/begoon/i8080-js/)
- Intel 8080 Microcomputer Systems User's Manual (September 1975, 262 pages) (http://www.nj7p.info/Manuals/PDFs/Intel/9800153B.pdf)
- Intel 8080 Microcomputer Systems User's Manual (September 1975, 234 pages) (http://www.elenota.pl/datasheet-pdf/133557/Intel/8080)
- Intel 8080/8085 Instruction Reference Card (https://sbc-85.com/download/8085-reference-card-hex/)
- US patent 4010449 (https://worldwide.espacenet.com/textdoc?DB=EPODOC&IDX=US4010449), Federico Faggin, Masatoshi Shima, Stanley Mazor, "MOS computer employing a plurality of separate chips", issued March 1, 1977