Wikipedia
The Free Encyclopedia

RCA 1802

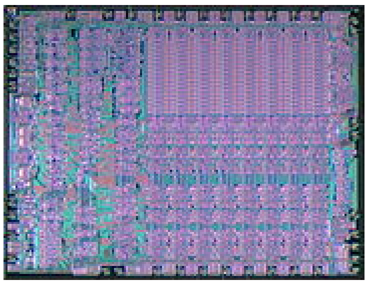
The **COSMAC** (Complementary Symmetry Monolithic Array Computer) is an 8-bit microprocessor family introduced by RCA. It is historically notable as the first CMOS microprocessor.[1] The first production model was the two-chip **CDP1801R** and **CDP1801U**, which were later combined into the single-chip **CDP1802**.[2] The 1802 represented the majority of COSMAC production, and today the entire line is known simply as the **RCA 1802**.

COSMAC

RCA CDP 1802

General information	
Launched	1974
Physical specifications	
Package(s)	40 pin DIP ,44 pin PLCC
History	
Successor(s)	CDP1804,CDP1805,CDP1806

The processor design traces its history to an experimental home computer designed by Joseph Weisbecker in the early 1970s, built at his home using TTL components. RCA began development of the CMOS version of the processor design in 1973, sampling it in 1974 with plans to move to a single-chip implementation immediately. Jerry Herzog led the design of the single-chip version, which sampled in 1975 and entered production in 1976.[3][4]



RCA CDP1802 die shot

In contrast to most designs of the era, which were fabricated using the NMOS process, the COSMAC was implemented in CMOS form and used static logic. This allowed it to run at lower power settings and even be stopped completely; in addition it would run cooler and not generate as much heat as NMOS chips. RCA referred to its CMOS process as "complementary silicon/metal-oxide semiconductor", giving rise to the acronym COS/MAC,[5] which was then backronymed to "complementary-symmetry monolithic-array computer" when referring to the processor. RCA also produced radiation hardened versions using a silicon on sapphire process, which found use in the aerospace field.[6] These remain in production as of 2022,[6] and as of 2008 continued to be produced by Renesas (formerly Intersil).[7][8]

Successors to the 1802 are the CDP1804, CDP1805, and CDP1806, which have an extended instruction set, other enhanced features (like on-chip RAM and ROM, and built-in timer), with some versions running at faster clock speeds, though not a significant speed difference. Some features are also lost, like the DMA auto-boot loader functionality. There are also some minor pin function changes, but the line continues to be produced in its original 40-pin dual in-line package (DIP) format.

History

FRED

Joseph Weisbecker had long been fascinated with the potential for computers in the home, having stated as early as 1955 that he expected they would one day be built into practically every device. The technology of the era made small embedded computers impossible, but the introduction of the integrated circuit (IC) in the

1960s changed things dramatically. In 1974 he described the possibilities in an IEEE Computer article:

> For 20 years computer hardware has become increasingly complex, languages more devious, and operating systems less efficient. Now, microcomputers afford some of us the opportunity to return to simpler systems. Inexpensive…microcomputers could open up vast new markets.[9]

Beginning in 1970,[a] Weisbecker began the design of a small machine using RCA transistor-transistor logic (TTL) ICs to construct the processor. Other parts, switches and lamps and such, he had to buy from Radio Shack, deliberately spreading his purchases around four stores so no one would ask him why he was buying so many parts.[10] The design was running in October 1971, containing 100 chips[1] spread over multiple circuit boards.[1]

The result, which he called FRED, ostensibly for Flexible Recreational Educational Device, was packaged into a box that was not unlike the Altair 8800 of a few years later, with toggle switches on the front panel for input, lamps for output, and later adding a hex pad keyboard.[10] Weisbecker added new features continually and by 1972 it had gained a character generator and the ability to load and save programs on cassette tapes.[1]

Weisbecker's daughter, Joyce Weisbecker, was immediately drawn to the system and began writing programs for it. This included several games, which were ported to later machines based on the COSMAC. When RCA entered the game console business in the later 1970s, these games were burned to ROM cartridge form, and Joyce became the first known female commercial videogame developer.[10]

Release

Weisbecker demonstrated the machine to RCA management throughout this period, but there was little interest at first. This was shortly after David Sarnoff had retired and handed the CEO role to his son, Robert Sarnoff. Robert was more interested in building the media side of the company while dating recording stars, ignoring RCA Laboratories in spite of a number of industry-leading developments taking place there. Some of the skepticism displayed by management may have had to do with the company's recent sale of their mainframe computer business to Sperry Rand with a huge writedown.[10]

Eventually, the company became interested in the system and began adapting it to their newly introduced COS/MOS fabrication system. A 1973 lab report[b] refers to a "prototype" being delivered in 1972, but this is likely referring to the original TTL implementation. It goes on to note that an effort to reduce the processor to a two-chip implementation with deliveries in COS/MOS in 1974. It is here that the processor is first referred to as COSMAC, for COmplementary-Symmetry-Monolithic-Array Computer. It goes on to state that another lab will be producing the system in an 8-chip silicon-on-sapphire format, although the date is simply "soon after" the CMOS versions, and that plans for a single-chip version were already being planned.[11][c]

COSMAC devices

Although RCA began the development of the COSMAC in the early 1970s, it was some time before they introduced their own products based on it. In 1975, a prototype of an arcade game machine with swappable ROMs was experimented with for the coin-op business, but was ultimately abandoned.[10]

Meanwhile, Weisbecker had adapted the original FRED, known within RCA as System 00 by this time, using the new chipset to produce a greatly simplified single-board system known as then COSMAC ELF. Building instructions were described in an article in Popular Electronics magazine in 1976, and an expanded version with various upgrades in a second article in 1977. A unique feature of the ELF is that it did not require any read only memory (ROM) for startup, instead, the processor's direct memory access (DMA) system was used to read front-panel switches directly into memory.[10]

RCA debated whether to introduce pre-packaged versions of the ELF to the market. While they debated, further development led to a simplified machine combining the ELF with a new display driver chip, the CDP1861, to produce a game console. During this time, Joyce was hired by RCA to write several videogames for the platform, including a quiz-style educational product in partnership with Random House, one of the many companies that had been picked up by RCA's buying sprees.[10]

After a year of discussion, the company eventually decided to release two mass-market products based on the platform, a kit computer known as the COSMAC VIP, and a game console known as the RCA Studio II. The machines had been available since 1975, but the Studio II was announced only in January 1977, a couple of months after the Fairchild Channel F became the first cartridge-based machine on the market. Both would soon be eclipsed and largely forgotten due to the release of the Atari 2600 later that year. RCA canceled the Studio II in February 1978.[10]

RCA also released a series of modular computer systems, based on the RCA Microboard form factor from the 1802's initial release, up until the collapse of RCA itself. These were mainly aimed at industrial applications and systems development, and were highly configurable.[13]

Embedded use

The COSMAC was unique among early 8-bit processors in that it had been explicitly designed for microcomputer use; other designs of the era were invariably aimed at the embedded processor space, and those that had been designed for computer use were generally more complex systems, and often 16-bit. Although the COSMAC had been designed for computer use, RCA's slow market entry and undersupported attempts in this market ultimately failed and other processors like the MOS 6502 and Zilog Z80 would come to dominate this market. Ironically, COSMAC would ultimately find great success in the embedded market, because its CMOS design allowed it to work at lower power. By the late 1970s it was widely used in many industrial settings, and especially aerospace. The 1802 ran the Galileo probe to Jupiter in 1989, and it remains in use in similar roles to this day.[10]

Applications

Microcomputer systems

A number of early microcomputers were based on the 1802, including the COSMAC ELF (1976), Netronics ELF II, Quest SuperELF, COSMAC VIP, Comx-35, Finnish Telmac 1800, Telmac TMC-600 and Oscom Nano, Yugoslav Pecom 32 and 64, and the Cybervision systems sold through Montgomery Ward in the late 1970s,[14] as well as the RCA Studio II video game console (one of the first consoles to use bitmapped graphics). The Edukit single-board computer trainer system, similar to an expanded COSMAC Elf, was offered by Modus Systems Ltd. in Britain in the early 1980s.[15] Infinite Incorporated produced an 1802-based, S-100 bus expandable console computer trainer in the late 1970s called the UC1800, available assembled or in kit form.[16][17]

As part of 1802 retrocomputing hobbyist work, other computers have been built more recently (post-2000), including the Membership Card microcomputer kit that fits in an Altoids tin[18] and the Spare Time Gizmos Elf 2000 (Elf 2K),[19] among others. See § Emulators and simulators for other systems.

Product integration

The 1802 was also used in scientific instruments and commercial products.[20] [21]

Post-1980 Chrysler and associated model vehicles use the 1802 in their second-generation Electronic Lean-Burn System, with electronic spark control, one of the first on-board auto computer-based control systems.[22][23]

The 1802 was used in the manufacture of many pinball machines and video arcade games in Spain.[24]

Radiation hardening

In addition to "bulk silicon" C2L CMOS technology, the 1802 was also available fabricated in Silicon on Sapphire (SOS) semiconductor process technology, which gives it a degree of resistance to radiation and electrostatic discharge (ESD). Along with its extreme low-power abilities, this makes the chip well-suited in space and military applications (also, at the time the 1802 was introduced, very few, if any, other radiation-hardened microprocessors were available in the market).[25][26] The radiation hardened 1802 version was manufactured at Sandia National Laboratories in agreement with RCA.[27]

Space technology and science

The 1802 was used in many spacecraft and space science programs, experiments, projects and modules such as the Galileo spacecraft,[28] Magellan,[29] the Plasma Wave Analyzer instrument on ESA's Ulysses spacecraft, various Earth-orbiting satellites[30] and satellites carrying amateur radio.[31]

The 1802 has also been verified from NASA source documentation to have been used in the Hubble Space Telescope.[32]

Military uses

A number of British military items from the 1980s and 1990s used the 1802, amongst them:

- L1A1 Fuze Setter[33]
- SAWES training system (Small Arms Weapons Effects Simulator) fitted to SLR / SA80 rifles
- Ptarmigan battlefield communications system

Programming languages

The first high-level language available for the 1802 was Forth, provided by Forth, Inc. and it was known as MicroFORTH, in 1976 (see Forth Inc's archive). Other available programming languages, both interpreters and compilers, are CHIP-8 (It was also invented by Joseph Weisbecker) (and variants), 8th (a version of Forth created by Lee Hart),[34] Tom Pittman's Tiny BASIC,[35] C, various Assemblers and cross-assemblers, and others. Other specialty languages were used by federal agencies such as NASA and its installations, including Johnson Space Center, AMES, Goddard, Langley, Marshall, and Jet Propulsion Laboratory (JPL), which included the HAL/S cross-compiler,[36] STOIC, a Forth-like language,[37] and others.

Interpreter for Process Structures (IPS), a programming language and development environment, was specifically written and used for real-time control of AMSAT satellites.

Emulators and simulators

The 1802 chip and computers using the microprocessor have been emulated and simulated in hardware and/or software by hobbyists. There are three designs in VHDL for an FPGA.[38][39][40] A bus-accurate, full speed COSMAC Elf clone was created without a CDP1802 microprocessor chip or CDP1861 video chip using PIC microcontrollers.[41] An online simulator of the COSMAC Elf (enhanced) written in JavaScript runs in the user's browser with no need to download.[42]

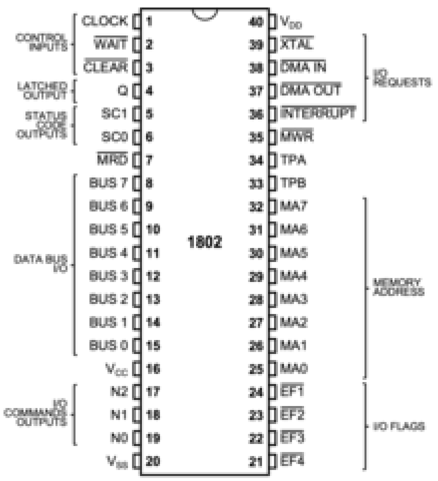
Description

Introduction

The RCA 1802 has a static core CMOS design with no minimum clock frequency, so that it can be run at very low speeds and low power, including a clock frequency of zero to suspend the microprocessor without affecting its operation.

It has two separate 8-pin buses: an 8-bit bidirectional data bus and a time-multiplexed address bus, with the high-order and low-order 8-bits of the 16-bit address being accessed on alternate clock cycles. This contrasts with most designs of the era, like the MOS 6502 and Intel 8080, which used a 16-bit address bus.

The 1802 has a single bit, programmable and testable output port (Q), and four input pins that are directly tested by branch instructions (EF1-EF4). These pins allow simple input/output (I/O) tasks to be handled directly and easily programmed.



RCA CDP1802 COSMAC processor DIP chip pinout

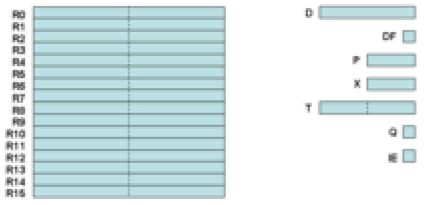
Because the instructions took 16 or 24 clock cycles to complete, the 1802 was not particularly fast. For comparison, the 6502 completes most instruction in 2 to 4 clock cycles, with the longest taking 7 cycles.[43]

Part number suffix designations

Various suffixes to the CDP1802 part number denote technical specifications, including (A, B, & C) *operating speed* (3.2 MHz to 6.4 MHz), *temperature* (-40 °C to +85 °C, -55 °C to +125 °C), and *voltage ranges* (4V to 10.5V), *package type* (D, E, Q), and *burn-in* (X). These were somewhat standardized between the various source suppliers, including RCA, Intersil, Harris, Hughes Aircraft, and Solid State Scientific (SSS). Hughes used the HCMP prefix, and SSS used the SCP (and possibly BCP) prefix, instead of CDP, and had additional suffixes that have not been documented as of yet. (examples: CDP1802A, CDP1802ACE, CDP1802BCD, HCMP1802AP, SCP1802D)[44]

Suffix	Package Type
E	PDIP = **Plastic** Dual In-Line Package
D	SBDIP = Side-Brazed **Ceramic** Dual In-Line Package
Q	**PLCC** = Plastic Leaded Chip Carrier
X	Burn-in

Registers and I/O



Register model

The 1802 is an 8-bit byte machine, with minimal support for 16-bit operations, except via 2-byte manipulation. The primary accumulator is the 8-bit 'D' register (Data register). The single bit carry flag is DF (Data Flag). Most operations use the D register, including arithmetic and logic functions, and memory referencing load and store instructions. Most 16-bit operations have to work on the lower byte and then the upper byte, via D, using the DF as carry and borrow as needed.

An important feature of the 1802 is a set of sixteen registers of 16 bits each, used primarily for addressing. Using the SEP instruction, you can select any of the 16 registers to be the program counter; using the SEX instruction, you can select any of the 16-bit registers to be the index register.[45] Register R0 has the special use of holding the memory address for the built-in DMA controller. Register R1 has the special use of being the program counter for the interrupt handler.[46]

There are instructions that allow the values in these registers to be set and read via D, separately working the upper and lower 8-bits at a time. There are also instructions to perform increment and decrement of the entire 16-bit value, and a few instructions perform automatic increment and decrement, like LDA (load advance) and STXD (store via X and decrement). 16-bit register and value comparisons would also need to use the D register as a go-between, using multiple instructions to perform the operations.

The processor has five special I/O lines. There is a single Q output that can be set with the SEQ instruction and reset with the REQ instruction. There are four external, single-bit flag inputs, EF1, EF2, EF3, and EF4, and there are eight dedicated branch instructions to conditionally branch based on the state of those input lines. There are seven Input and seven Output port instructions that utilize the RX register and D accumulator.

The EF and Q lines were typically used for multiple interfaces on 1802-based hobbyist computers because of the lines' favorable and easy handling. It was typical for the Q line to drive a status LED, a cassette interface, an RS-232 interface, and the speaker. This meant that the user could actually hear RS-232 and cassette data being transmitted (unless a volume control was implemented). Traditionally, the EF4 line is attached to the INPUT momentary pushbutton on the COSMAC Elf. Other systems might use one of the other lines.

There are some other special use registers and flags, some internal, and some usable programmatically: 4-bit N, P, X, and I; 8-bit T; and 1-bit IE. On the Internet, there is many versions of a Table of the 1802 Instructions, here is one link : https://www.atarimagazines.com/computeii/issue3/page52.php There is a number of Tables in the File area for logged in members of the 1802 online club for forum at https://groups.io/g/cosmacelf/files. Each Instruction is a single Byte, of 8 Bits. The 4 Bits to the Left, sometimes called High Order Hex Digit, are to do with the actual nature of the Instruction, and those 4 Bits are involved with the I Register. The 4 Bits to the Right, sometimes called Low Order Hex Digit, are to do with the working Register, where Data is taken from, or put into it, and those 4 Bits are involved with the N

Register. When a Program is being done, the various stages of the work and the Processes, are stored temporarily in the Register pointed to by the N Register, like what happens, when long Multiplication, or long Division, is done on a piece of paper. Of course sometimes the Data is moved to and from the Ram Memory section, and different Branches of the Program are done, as needed, in the flow of time.

Branching

There are three types of unconditional and conditional branching in the 1802, Short and Long, and Skips.

Short branches are 2-byte instructions, and use 256-byte range, single byte address, page absolute addressing in the range 0 to 255 (hex FF). There is no relative branching. The short branch always jumps within the page that contains the address byte.[47]

Long branches use full 16-bit addressing to support the 64K memory address space, and are the only 3-byte instructions.

Skip instructions increment the PC by one for the unconditional Short Skip, or two for the Long Skips. Only the Long Skip has conditional branching.

Subroutine calls

The processor does not have standard subroutine CALL address and RET instructions, though they can be simulated. The 16-register design makes possible some interesting subroutine call and return mechanisms, though they are better suited to small programs than general purpose coding.

A few commonly used subroutines can be called quickly by keeping their address in one of the 16 registers; however, the called subroutine must know (hard coded) what the calling PC register is to perform the "return" instruction. The SEP instruction is used to call a subroutine pointed to by one of the 16-bit registers and another SEP to return to the caller (SEP stands for *Set Program Counter*, and selects which one of the 16 registers is to be used as the program counter from that point onward). Before a subroutine returns, it jumps to the location immediately preceding its entry point so that after the SEP "return" instruction returns control to the caller, the register will be pointing to the right value for next usage. (the processor always increments the PC after reference and usage (retrieving the next instruction to execute), so this technique works as noted)

An interesting variation of this scheme is to have two or more subroutines in a ring so that they are called in round robin order. On early hobbyist computers, tricks and techniques like this were commonly used in the horizontal refresh interrupt to reprogram the scan line address to repeat each scan line four times for the video controller.

One well-known and often-used routine is known as SCRT (Standard CALL and RETURN Technique), which allows general purpose subroutine Call and Return, including passing of parameters "in line", and nested subroutines using a stack. Although any of the available registers can be used for this technique, per programmer's preference, many use the routine supplied by RCA in the CDP1802 User Manual, where the suggested register usage is R2 = Stack Pointer, R3 = General Program Counter (PC), R4 = Call, R5 = Return, R6 = Passed Arguments Pointer (non-destructive). Even though these supportive routines are small, there is an execution speed overhead using them. (as opposed to what would be incurred if actual CALL and RET instructions were part of the microprocessor's design) This setup allows R0 to be used for DMA and R1 to be used for Interrupts, if desired, allowing R7 through RF (hex) for general program usage.

Addressing modes

Because of the 16-bit address bus, and the 8-bit data bus, the sixteen general purpose registers are 16 bits wide, but the accumulator D-register is only 8 bits wide. The accumulator, therefore, tends to be a bottleneck. Transferring the contents of one register to another involves four instructions (one Get and one Put on the HI byte of the register, and a similar pair for the LO byte: GHI R1; PHI R2; GLO R1; PLO R2). Similarly, loading a new constant into a register (such as a new address for a subroutine jump, or the address of a data variable) also involves four instructions (two load immediate, LDI, instructions, one for each half of the constant, each one followed by a Put instruction to the register, PHI and PLO).

The two addressing modes *Indirect register*, and *Indirect register with auto-increment* are then fairly efficient, to perform 8-bit operations on the data in the accumulator. There are no other addressing modes, though. Thus, the *direct addressing* mode needs to be emulated using the four instructions mentioned earlier to load the address into a spare register; followed by an instruction to select that register as the index register; followed, finally, by the intended operation on the data variable that is pointed to by that address.

DMA and load mode

The CDP1802 has a simple built-in DMA controller, having two DMA request lines for DMA input and output. The CPU only accesses memory during certain cycles of the multi-step machine cycle, which required between 8 and 16 clock cycles. External hardware could read or write data during these periods without interrupting the processor, a general concept known as cycle stealing.

R0 is used as the DMA address pointer. The starting address of the DMA data would be put in R0 and then pulling the appropriate read or write pin on the CPU low. The CPU responded to the DMA request by incrementing the value in R0, so that the next request automatically stored in the next location in memory. Thus by simply repeatedly triggering the DMA pins, the system would walk through the entire memory.

The DMA controller also provides a special "load mode", which allows loading of memory while the CLEAR and WAIT inputs of the processor are active. This allows a program to be loaded without the need for a ROM-based bootstrap loader. This was used by the COSMAC Elf microcomputer and its successors to load a program from toggle switches or a hexadecimal keypad with no required software and minimal hardware. The user could simply set the switches to the next value, toggle the read, and then move on. There was no need to change the addresses, that was done automatically by the DMA stepping.

Instruction timing

Clock cycle efficiency is poor in comparison to most 8-bit microprocessors. Eight clock cycles makes up one machine cycle. Most instructions take two machine cycles (16 clock cycles) to execute; the remaining instructions take three machine cycles (24 clock cycles). By comparison, the MOS Technology 6502 takes two to seven clock cycles to execute an instruction, and the Intel 8080 takes four to 18 clock cycles.

RCA CDP 1802 instructions in Hexadecimal order

Opcode	Instruction	Mnemonic	Machine Cycles
00	It is idle, doing fetch cycles (I think)	IDL	2
0N	Load D from address in RN	LDN N	2
1N	Increment the 16 bit number in RN	INC N	2
2N	Decrement the 16 bit number in RN	DEC N	2
30 MM	Branch R(P) unconditionally to MM	BR	2
31 MM	Branch R(P) to MM if Q is 1	BQ	2
32 MM	Branch R(P) to MM if D is 00	BZ	2
33 MM	Branch R(P) to MM if DF is 1	BDF	2
34 MM	Branch R(P) if the External Flag EF1 is 1	B1	2
35 MM	Branch R(P) if the External Flag EF2 is 1	B2	2
36 MM	Branch R(P) if the External Flag EF3 is 1	B3	2
37 MM	Branch R(P) if the External Flag EF4 is 1	B4	2
38	Skip R(P) by 1 addressed memory byte	SKP	2
39 MM	Branch R(P) to MM if Q is 0	BNQ	2
3A MM	Branch R(P) to MM if D is NOT 00	BNZ	2
3B MM	Branch R(P) to MM if DF is 0	BNF	2
3C MM	Branch R(P) if the External Flag EF1 is 0	BN1	2
3D MM	Branch R(P) if the External Flag EF2 is 0	BN2	2
3E MM	Branch R(P) if the External Flag EF3 is 0	BN3	2
3F MM	Branch R(P) if the External Flag EF4 is 0	BN4	2
4N	Load D from address in register RN and advance RN by 1	LDA N	2
5N	Store D into memory pointed to by register RN	STR N	2
60	Increment the 16 bit number in R(X)	IRX	2
61	Output from memory byte at R(X) to the BUS condition	OUT 1	2
62	Output from memory byte at R(X) to the BUS condition	OUT 2	2
63	Output from memory byte at R(X) to the BUS condition	OUT 3	2
64	Output from memory byte at R(X) to the BUS condition	OUT 4	2
65	Output from memory byte at R(X) to the BUS condition	OUT 5	2
66	Output from memory byte at R(X) to the BUS condition	OUT 6	2
67	Output from memory byte at R(X) to the BUS condition	OUT 7	2
68	(Undefined for the RCA 1802)		
69	Input from the BUS condition to the memory byte at R(X) and the D register as well	INP 1	2
6A	Input from the BUS condition to the memory byte at R(X) and the D register as well	INP 2	2

6B	Input from the BUS condition to the memory byte at R(X) and the D register as well	INP 3	2
6C	Input from the BUS condition to the memory byte at R(X) and the D register as well	INP 4	2
6D	Input from the BUS condition to the memory byte at R(X) and the D register as well	INP 5	2
6E	Input from the BUS condition to the memory byte at R(X) and the D register as well	INP 6	2
6F	Input from the BUS condition to the memory byte at R(X) and the D register as well	INP 7	2
70	The memory byte at the address in R(x) is Returned to the memory byte at the address in R(P) after an Interrupt, and the Interrupt Enable register IE is enabled by setting to 1, R(X) is incremented	RET	2
71	The memory byte at the address in R(P) is moved into the memory byte at the address in R(X), Interrupt Enable register IE is Disabled by setting it to 0, R(X) is incremented	DIS	2
72	Load the memory byte at R(X) into the D register, increment R(X)	LDXA	2
73	Store the D register into the memory byte at R(X) and decrement R(X)	STXD	2
74	Add the memory byte at R(X) to the D register and the carry register DF, and the answer is moved to the D register and the carry register DF	ADC	2
75	Subtract the D register and the carry register DF from the memory byte at R(X), and move the answer to the D register and the carry register DF	SDB	2
76	Shift the D register right and the carry register DF	SHRC	2
77	Subtract the memory byte at R(X) from the D register and the carry register, and move the answer to the D register and the carry register DF	SMB	2
78	Save the byte in register T into the memory byte at R(x)	SAV	2
79	Save the 4 bit nibble in register X and the 4 bit nibble in register P into the byte in register T, the byte in register T is moved into the byte in register R(2), the nibble in X is moved into the nibble P, R(2) is decremented	MARK	2
7A	Reset Q output to 0	REQ	2
7B	Set Q output to 1	SEQ	2
7C KK	Add the byte KK to the D register with the carry register DF, move the answer to the D register and the carry register DF immediate	ADCI	2
7D KK	Subtract the byte in the D register and the carry register DF from the byte KK, and move the answer to the D register and the carry register DF, immediate 2	SBDI	2
7E	Shift the D register left and the carry register DF	SHLC	2
7F KK	Subtract the byte KK from the D register and the carry register DF, move the answer to the D register and the carry register DF immediate	SMBI	2
8N	Move the low byte of the RN register into the D register	GLO N	2
9N	Move high byte of the RN register into the D register	GHI N	2
AN	Put the byte in the D register into the low byte of the RN register	PLO N	2
BN	Put the byte in the D register into the high byte of the RN register	PHI N	2
C0 MMMM	Long branch R(P) to MMMM unconditionally	LBR	3

C1 MMMM	Long branch R(P) to MMMM if Q is 1	LBQ	3
C2 MMMM	Long branch R(P) to MMMM if D is 00	LBZ	3
C3 MMMM	Long branch R(P) to MMMM if DF is 1	LBDF	3
C4	No operation	NOP	3
C5	Long skip R(P) by 2 addressed memory bytes if Q is 0	LSNQ	3
C6	Long skip R(P) by 2 addressed memory bytes if D is NOT 00	LSNZ	3
C7	Long skip R(P) by 2 addressed memory bytes if DF is 0	LSNF	3
C8	Long skip R(P) by 2 addressed memory bytes	LSKP	3
C9 MMMM	Long branch R(P) to MMMM if Q is 0	LBNQ	3
CA MMMM	Long branch R(P) to MMMM if D is NOT 00	LBNZ	3
CB MMMM	Long branch R(P) to MMMM if DF is 0	LBNF	3
CC	Long skip R(P) by 2 addressed memory bytes if the Interrupts Enabled IE is 1	LSIE	3
CD	Long skip R(P) by 2 addressed memory bytes if Q is 1	LSQ	3
CE	Long skip R(P) by 2 addressed memory bytes if D is 00	LSZ	3
CF	Long skip R(P) by 2 addressed memory bytes if DF is 1	LSDF	3
DN	Set the nibble in the P register to point to the RN register	SEP N	2
EN	Set nibble in the X register to point to the RN register	SEX 0	2
F0	Load the byte in the D register with the memory byte at R(X)	LDX	2
F1	Logical OR the byte in the D register with the memory byte at R(X), move the result into the D register	OR	2
F2	Logical AND the byte in the D register with the memory byte at R(X), move the result into the D register	AND	2
F3	Exclusive OR the byte in the D register with the memory byte at R(X), move the result into the D register	XOR	2
F4	Add the byte in the D register to the memory byte at R(X), move the result into the D register		2
F5	Subtract the byte in the D register from the memory byte at R(X), move the result into the D register	SD	2
F6	Shift the D register right 2	SHR	2
F7	Subtract memory byte at R(X) from the D register, move the result into the D register	SM	2
F8 KK	Load KK into the D register	LDI	2
F9 KK	OR the byte in the D register with KK, move the answer into the D register	ORI	2
FA KK	AND the byte in the D register with KK, move the answer into the D register	ANI	2

FB KK	Exclusive OR the byte in the D register with KK, move the answer into the D register	XRI	2
FC KK	Add the byte in the D register with KK, move the answer into the D register and the carry register DF	ADI	2
FD KK	Subtract the byte in the D register from KK, move the answer into the D register	SDI	2
FE	Shift the D register left	SHL	2
FF KK	Subtract the byte KK from the D register, move the answer to the D register and the carry register DF	SMI byte	2

Key to symbols in instruction table

RN/RX	16 BIT GENERAL PURPOSE REGISTER (16 OF THESE LABLED R0 TO RF)
RN.0	LOW-ORDER REGISTER NUMBER
RN.1	HIGH-ORDER REGISTER NUMBER
MN/MX	MEMORY BYTE LOCATION ADDRESSED BY CONTENTS OF RN/RX
P	4-BIT PROGRAM COUNTER REGISTER
X	4-BIT X REGISTER NUMBER
D	8-BIT HOLDING REGISTER
DF	1-BIT OVERFLOW REGISTER
IE	1-BIT INTERUPT ENABLE REGISTER
Q	1-BIT OUTPUT REGISTER
T	8-BIT SPECIAL REGISTER

Support chips

Graphics

In early 1802-based microcomputers, the companion graphics Video Display Controller chip, RCA CDP1861 (for the NTSC video format, CDP1864 variant for PAL), used the built-in DMA controller to display black and white bitmapped graphics on standard TV screens at up to 64 pixels horizontally by 128 pixels vertically. The 1861 was also known as the Pixie graphics system.

Although the faster versions of 1802 could operate at 4–5 MHz (at 5 V; it was faster (6.4 MHz) at 10 V), it was usually operated at 3.58 MHz, divided by 2 (1.79 MHz) to suit the requirements of the 1861 chip, which gave a speed of a little over 100,000 instructions per second, though some ran at other speeds such as the ~2.8 MHz of the Comx or 5 MHz of the Pecom. The COSMAC VIP, which integrated the video chip with the processor as a single purpose-built computer (rather than as an add-on to a hobbyist kit), notably ran the 1802 much slower, synchronising it exactly with the 1861 - at a non-standard 1.76064 MHz, as recommended in the Pixie's spec sheet reference design.[d]

The CDP1862 Color Generator Circuit IC, an 1861 companion chip, could be used to generate color graphics. Some computer systems, like the Pecom 64or the Telmac TMC-600, used the VIS (Video Interface System), consisting of the CDP1869 and CDP1870 companion ICs, for distinctly higher resolution color graphics, comparable to other 8-bit systems of the 1980s.

Code samples

This code exemple is a minimal code that allows for an quick test on the Event Flag (EF) pins, you can test the pins one thru four by changing B4 to B2 wich tests EF2 pin

```
LOOP

B4 EFbranch ;because EF pins pins are active low this makes it behave as an not gate

REQ

BR LOOP

EFbranch
SEQ
BR LOOP
```

This code snippet example is a diagnostic routine that tests ALU (Arithmetic and Logic Unit) Operations.[48]

```
..  TEST ALU OPS
0000 90         GHI 0      .. SET UP R6
0001 B6         PHI 6
0002 F829       LDI DOIT   .. FOR INPUT OF OPCODE
0004 A6         PLO 6
0005 E0         SEX 0      .. (X=0 ALREADY)
0006 6400       OUT 4,00   .. ANNOUNCE US READY
0008 E6         SEX 6      .. NOW X=6
0009 3F09       BN4 *      .. WAIT FOR IT
000B 6C         INP 4      .. OK, GET IT
000C 64         OUT 4      .. AND ECHO TO DISPLAY
000D 370D       B4 *       .. WAIT FOR RELEASE
000F F860       LDI #60    .. NOW GET READY FOR
0011 A6         PLO 6      .. FIRST OPERAND
0012 E0         SEX 0      .. SAY SO
0013 6401       OUT 4,01
0015 3F15       BN4 *
0017 E6         SEX 6      .. TAKE IT IN AND ECHO
0018 6C         INP 4      .. (TO 0060)
0019 64         OUT 4      .. (ALSO INCREMENT R6)
001A 371A       B4 *
001C E0         SEX 0      .. DITTO SECOND OPERAND
001D 6402       OUT 4,02
001F E6         SEX 6
0020 3F20 LOOP: BN4 *      .. WAIT FOR IT
0022 6C         INP 4      .. GET IT (NOTE: X=6)
0023 64         OUT 4      .. ECHO IT
0024 3724       B4 *       .. WAIT FOR RELEASE
0026 26         DEC 6      .. BACK UP R6 TO 0060
0027 26         DEC 6
0028 46         LDA 6      .. GET 1ST OPERAND TO D
0029 C4   DOIT: NOP        .. DO OPERATION
002A C4         NOP        .. (SPARE)
002B 26         DEC 6      .. BACK TO 0060
002C 56         STR 6      .. OUTPUT RESULT
002D 64         OUT 4      .. (X=6 STILL)
002E 7A         REQ        .. TURN OFF Q
002F CA0020     LBNZ LOOP .. THEN IF ZERO,
0032 7B         SEQ       .. TURN IT ON AGAIN
0033 3020       BR LOOP   .. REPEAT IN ANY CASE
```

Note: The above routine presumes that the CDP1802 microprocessor is in an initial reset state (or that it has been set as such prior to executing this code). Therefore, the program counter (PC) and the X indirect register 'pointer' are both set to 16-bit register R0. That is why you can output an immediate value, as in the example 'OUT 4,00', because PC and X are both pointing to R0. The PC is incremented after the opcode instruction byte is retrieved from memory, so it points to the next address when the OUT 4 is executed. Therefore, it outputs the value in memory pointed to by RX = R0, which is the next immediate byte. The OUT instruction also increments the X register, which is R0, which is also the PC, so it outputs the immediate value after the OUT and continues program execution at the next instruction address after the immediate value. This is why you see the routine set X (SEX) to register R6 and R0 as needed. Also note that, although the OUT opcode increments the RX register, to easily output a section of memory ('buffer'), INP does not. It stores the value at the address pointed to by RX and into the D 8-bit data byte accumulator, but RX is not modified.

The routine also presumes that OUT 4 will display the value in the CPU system's 8-bit LED or 2-digit hex display, and IN 4 gets the value from the eight toggle switches (or possibly the hex keypad). The BN4 opcode (loop; * = 'this address'), "branch if the single-bit input EF4 line is lo", is used to test if the momentary 'Input' pushbutton is pressed. The B4 opcode ('if hi') loop waits for the button to be released. SEQ and REQ turn the single Q line, which is usually attached to an LED, on and off.

The 1802 is a "byte machine", but has 16 16-bit registers, R0-RF (sometimes referred to as 0-F without the 'R' prefix). To deal with 16-bit register data, the programmer must Get and Put the Hi or Lo values of the registers using the D accumulator as the go-between. These high and low bytes of the registers are sometimes referred to as Rn.0 (lo) and Rn.1 (hi). Short Branches are 2-byte opcodes with page-absolute addressing, and a 256-byte address boundary. Long Branches are 3-byte opcodes with full 16-bit address branching.

This information should make the routine more understandable to any computer programmer who is knowledgeable enough to read "pseudo-code" and is minimally familiar with assembly and machine language programming.

Notes

a. The exact date varies between references, as is the case for most dates related to the COSMAC. Edwards puts it in 1969,[10] while most others say 1970, the date used here.

b. This was the annual review of 1973's operations, published some time in 1974.[11]

c. The exact dates of the sampling and general release of the various COSMAC devices remain imprecise. Herb Johnson has produced an extensive list of RCA reports that form the basis of the dates in this article.[12]

d. However, given the age of the machine, this may be due to the higher speed grades not having yet been developed, meaning the processor was only rated for a maximum of 3.2, or possibly even just 2.5 MHz. Although it is an extreme case, the machine wouldn't be alone in running a CPU well below its rated speed in order to save cost and complexity in the timing system, and simply running at 3.52 MHz would have represented a risky 10%, or even an unsustainably extreme 41% overclock.

References

Citations

1. Cass 2018.
2. "RCA COSMAC 1802" (https://web.archive.org/web/20130102213115/http://www.antiquetech.com/chips/RCA1802.htm). *The Antique Chip Collector's Page*. AntiqueTech.com. 21 April 2009. Archived from the original (http://www.antiquetech.com/chips/RCA1802.htm) on 2 January 2013. Retrieved 27 December 2010.
3. "Joseph Weisbecker" (http://www.wiki.vintage-computer.com/index.php/Joseph_Weisbecker). Vintage-Computer.com. 2010-02-08. Retrieved 2010-12-27.
4. "Joseph A. Weisbecker (1932 - 1990)" (http://www.cosmacelf.com/weisbecker.html). CosmacElf.com. Retrieved 2010-12-27.
5. "RCA Laboratories Research Report 1973" (http://www.retrotechnology.com/memship/RCA_Res_1973_COSMAC.pdf) (PDF). *RetroTechnology*. RCA. Retrieved 24 May 2016.
6. Cass, Stephen. "Chip Hall of Fame: RCA CDP 1802 - IEEE Spectrum" (https://spectrum.ieee.org/chip-hall-of-fame-rca-cdp-1802#toggle-gdpr). *spectrum.ieee.org*. IEEE. Retrieved 19 July 2023.
7. "CDP1802A" (https://web.archive.org/web/20120309014541/http://www.intersil.com/products/deviceinfo.asp?pn=CDP1802A). Archived from the original (http://www.intersil.com/products/deviceinfo.asp?pn=CDP1802A) on 2012-03-09. Retrieved 2010-08-23.
8. "CDP1802AC/3 High-Reliability CMOS 8-Bit Microprocessor" (https://www.renesas.com/jp/ja/document/dst/cdp1802ac3-datasheet?r=481786). Intersil Americas LLC. 17 October 2008.
9. Weisbecker 1974, p. 41.
10. Edwards 2017.
11. Lab 1973, p. 152.
12. Johnson, Herb (11 December 2018). "COSMAC 1801, 1802 "dates" " (http://www.retrotechnology.com/memship/1802_dates.html).
13. "RCA MCDS, Microboard Computer Development System" (https://www.retrotechnology.com/memship/cosmac_dev_sys.html). RetroTechnology.com. 2020-10-28. Retrieved 2022-05-08.
14. Ruske, Dave. "Cybervision 2001, 3001, and 4001" (http://www.cosmacelf.com/gallery/cybervision/). *COSMAC Elf*. COSMACELF.COM. Retrieved 30 June 2016.
15. "Wireless World magazine ad on page 22" (http://www.americanradiohistory.com/Archive-Wireless-World/80s/Wireless-World-1981-05.pdf) (PDF). *American Radio History*. May 1981. Retrieved 21 Jan 2017.
16. "Equipment Report - Infinite UC1800 Microcomputer" (http://www.classiccmp.org/cini/pdf/re/1977/RE1977-Aug-Pg27.pdf) (PDF). *ClassicCmp.org - Classic Computing*. Radio Electronics Magazine. Aug 1977. Retrieved 22 Jan 2017.
17. Haberhern, William (Feb 1977). "Kilobaud Magazine Article Pg 90" (https://archive.org/details/Kilobaud197702). *Archive.org*. Wayne Green. Retrieved 22 Jan 2017.
18. Hart, Lee. "The 1802 Membership Card Computer" (http://www.sunrise-ev.com/membershipcard.htm). *Lee Hart's Homepage*. Lee Hart. Retrieved 22 May 2016.
19. "Spare Time Gizmos Elf 2000 (Elf 2K)" (http://www.sparetimegizmos.com/Hardware/Elf2K.htm).

20. "Five generations of Sinar Moisture Meters" (http://www.inventio.co.uk/album_large13.html). Retrieved 22 May 2016.
21. "Inforcel" (http://www.decodesystems.com/cosmac/infocel.html). Retrieved 22 May 2016.
22. Johnson, Herbert R. (22 June 2016). "RCA/Weisbecker "System 00" aka "FRED" " (http://www.retrotechnology.com/vcfe91/herb_vcf_apr14.html). *Retrotechnology*. Retrieved 23 June 2016.
23. "The Chrysler Lean Burn engine control system" (http://www.allpar.com/mopar/lean-burn.html). *All Mopar Chrysler Info*. AllPar.com. Retrieved 23 June 2016.
24. Donnelly, William (7 July 2016). "MDCCCII (1802) | Product Integration: Commercial & Scientific Applications" (http://www.mdccii.com/products.php). *A COSMAC 1802 CPU RetroComputing Extravaganza*. William Donnelly. Retrieved 7 July 2016.
25. Dingwall, A.; Stricker, R.; Sinniger, J. (October 1977). "A high speed bulk CMOS C2L microprocessor". *IEEE Journal of Solid-State Circuits*. IEEE. **12** (5): 457–462. doi:10.1109/ISSCC.1977.1155726 (https://doi.org/10.1109%2FISSCC.1977.1155726).
26. "A Radiation-Hardened Bulk Si-Gate CMOS Microprocessor Family" (http://www.iaea.org/inis/collection/NCLCollectionStore/_Public/11/506/11506044.pdf?r=1) (PDF). *IAEA.org*. Retrieved 4 June 2016.
27. Gülzow, Peter. "No RISC, No Fun!" (http://www.amsat-dl.org/yahue.html). *AmSat Germany*.
28. Tomayko, James (April 1987). "Computers in Spaceflight: The NASA Experience" (https://history.nasa.gov/computers/Ch6-3.html). NASA. Retrieved February 6, 2010.
29. http://www2.jpl.nasa.gov/magellan/guide4.html#4.11 The Magellan Venus Explorer's Guide, Chapter 4 - The Magellan Spacecraft - Computing and Software
30. "RCA COSMAC VIP" (http://oldcomputers.net/rca-cosmac-vip.html). *Obsolete Technology Website*. Retrieved January 31, 2010.
31. http://www.amsat.org/amsat-new/AboutAmsat/amsat_history.php AMSAT History
32. Afshari, A. (January 1993). "Hubble Space Telescope's Wide Field/Planetary Camera" (https://web.archive.org/web/20161006205644/http://trs-new.jpl.nasa.gov/dspace/bitstream/2014/35164/1/93-0731.pdf) (PDF). *Shutterbug*. Archived from the original (http://trs-new.jpl.nasa.gov/dspace/bitstream/2014/35164/1/93-0731.pdf) (PDF) on October 6, 2016.
33. *Royal Ordnance L1A1 fuze setter teardown* (https://www.youtube.com/watch?v=RpbWH_fQab4), retrieved 2022-10-16
34. "Membership Card Software" (http://www.retrotechnology.com/memship/mship_soft.html).
35. "Itty Bitty Computers & TinyBasic" (http://www.ittybittycomputers.com/IttyBitty/TinyBasic/).
36. "Current Status of the HAL/S Compiler on the Modcomp Classic 7870 Computer" (http://ipnpr.jpl.nasa.gov/progress_report/42-64/64AA.PDF) (PDF).
37. *MASCOT (MIT Astronomical Spectrometer / Camera for Optical Telescopes)* (https://ntrs.nasa.gov/search.jsp?R=19810041565). January 1980.
38. Baker, Scott (2016). "1802 CPU coded in VHDL" (https://github.com/scottlbaker/1802-SOC). *Scott L Baker Github*. Scott Baker. Retrieved 24 July 2016.
39. Smith, Eric (2009). "cosmac - RCA COSMAC CDP1802 functional equivalent CPU core in VHDL" (https://github.com/brouhaha/cosmac). *Eric Smith Github*. Eric Smith. Retrieved 9 July 2019.
40. Teal, Steve (2016). "VHDL 1802 Core with TinyBASIC for the Lattice MachXO2 Pico board" (https://github.com/Steve-Teal/1802-pico-basic). *Steve Teal Github*. Steve Teal. Retrieved 9 July 2019.
41. Rossin, Ted (2011). "Elf Clone" (https://www.sites.google.com/site/tedrossin/home/electronics/rca). *Ted Rossin Homepage*. Ted Rossin. Retrieved 24 July 2016.

42. Donnelly, William (2011). "COSMAC Elf-ish CDP1802 Simulator in JavaScript" (http://www.donnelly-house.net/programming/cdp1802/simelf/). *Donnelly-House Homepage*. William Donnelly. Retrieved 24 July 2016.
43. "6502 Instruction Set" (https://www.masswerk.at/6502/6502_instruction_set.html). *mass:werk*.
44. Shvets, Gennadiy (2 Oct 2016). "RCA 1802 (CDP1802) microprocessor family" (http://www.cpu-world.com/CPUs/1802/). *CPU-World*. Gennadiy Shvets. Retrieved 17 October 2016.
45. "What does SEX mean?" (http://fullforms.com/SEX). Retrieved December 26, 2013.
46. User Manual for the CDP1802 COSMAC Microprocessor
47. User Manual for the CDP1802 COSMAC Microprocessor
48. Pittman, Tom (1980). "A Short Course In Programming" (http://www.cosmacelf.com/publications/books/short-course-in-programming.html#chapter5). *cosmacelf.com*. Retrieved 20 May 2017.

Bibliography

- *Research Report 1973* (http://www.retrotechnology.com/memship/RCA_Res_1973_COSMAC.pdf) (PDF) (Technical report). RCA Laboratories. 1973.
- Cass, Stephen (2 July 2018). "Chip Hall of Fame: RCA CDP 1802" (https://spectrum.ieee.org/semiconductors/processors/chip-hall-of-fame-rca-cdp-1802). *IEEE Spectrum*.
- Edwards, Benj (2017-10-27). "Rediscovering History's Lost First Female Video Game Designer" (https://www.fastcodesign.com/90147592/rediscovering-historys-lost-first-female-video-game-designer). *Fast Company*. Retrieved 2017-10-27.
- Weisbecker, Joe (March 1974). "A simplified microcomputer architecture". *IEEE Computer*. **7** (3): 41–47. doi:10.1109/MC.1974.6323475 (https://doi.org/10.1109%2FMC.1974.6323475). S2CID 30965828 (https://api.semanticscholar.org/CorpusID:30965828).

External links

- CDP1802A/AC/BC datasheet, 1997 (http://www.cosmacelf.com/publications/data-sheets/cdp1802.pdf) (PDF)
- CDP1802AC/3 datasheet, 2008 (https://www.renesas.com/eu/en/www/doc/datasheet/cdp1802ac-3.pdf) (PDF)
- COSMAC ELF website (http://www.cosmacelf.com/)
- A Short Course in Programming (http://www.ittybittycomputers.com/IttyBitty/ShortCor.htm) (1980 text on RCA 1802 assembler)
- High resolution die shot (http://visual6502.org/images/pages/RCA_1802.html)

-