


# Intel 8080/Коды команд

Материал из Emuverse  
< Intel 8080

Этот документ создан для Emuverse и распространяется на условиях лицензии **CC-BY-SA-3.0** (<http://creativecommons.org/licenses/by-sa/3.0/>).

## Содержание

- 1 Таблица кодов
- 2 Расшифровка сокращений
- 3 Размещение двухбайтовых адресов в памяти
- 4 Операции со стеком
- 5 Начальное состояние процессора
- 6 Особенности

## Таблица кодов

| Код <sub>2</sub> | Код <sub>16</sub> | Данные | Команда       | Действие            | Флаги | Такты |
|------------------|-------------------|--------|---------------|---------------------|-------|-------|
| Группа 00        |                   |        |               |                     |       |       |
| 00 000 000       | 00                |        | NOP           |                     |       | 4     |
| 00 001 000       | 08                |        | ?             |                     |       |       |
| 00 010 000       | 10                |        | ?             |                     |       |       |
| 00 011 000       | 18                |        | ?             |                     |       |       |
| 00 100 000       | 20                |        | ? (8085: RIM) | 8085: READ INT MASK |       |       |
| 00 101 000       | 28                |        | ?             |                     |       |       |
| 00 110 000       | 30                |        | ? (8085: SIM) | 8085: SET INT MASK  |       |       |
| 00 111 000       | 38                |        | ?             |                     |       |       |
| 00 RP1 001       |                   |        | DAD RP        | HL ← HL + RP        | C     | 4     |
|                  |                   |        |               |                     |       |       |

|                 |    |        |                |                  |             |       |
|-----------------|----|--------|----------------|------------------|-------------|-------|
| 00 RP0 001      |    | DATA16 | LXI RP, DATA16 | RP ← DATA16      |             | 10    |
| 00 OR0 010      |    |        | STAX [R], A    | [R] ← A          |             | 7     |
| 00 OR1 010      |    |        | LDAX A, [R]    | A ← [R]          |             | 7     |
| 00 100 010      | 22 | ADDR16 | SHLD ADDR16    | [ADDR16] ← HL    |             | 16    |
| 00 101 010      | 2A | ADDR16 | LHLD ADDR16    | HL ← [ADDR16]    |             | 16    |
| 00 110 010      | 32 | ADDR16 | STA ADDR16     | [ADDR16] ← A     |             | 13    |
| 00 111 010      | 3A | ADDR16 | LDA ADDR16     | A ← [ADDR16]     |             | 13    |
| 00 RP1 011      |    |        | DCX RP         | RP ← RP-1        |             | 5     |
| 00 RP0 011      |    |        | INX RP         | RP ← RP+1        |             | 5     |
| 00 SSS 100      |    |        | INR SSS        | SSS ← SSS+1      | все кромє C | 5/10  |
| 00 SSS 101      |    |        | DCR SSS        | SSS ← SSS-1      | все кромє C | 5/10  |
| 00 DDD 110      |    | DATA8  | MVI DDD, DATA8 | DDD ← DATA8      |             | 7/11  |
| 00 000 111      | 07 |        | RLC            | CY+A LEFT        | C           | 4     |
| 00 001 111      | 0F |        | RRC            | CY+A RIGHT       | C           | 4     |
| 00 010 111      | 17 |        | RAL            | CY+A CYCLE RIGHT | C           | 4     |
| 00 011 111      | 1F |        | RAR            | CY+A CYCLE LEFT  | C           | 4     |
| 00 100 111      | 27 |        | DAA            |                  |             | 5     |
| 00 101 111      | 2F |        | CMA            | A ← NOT A        |             | 4     |
| 00 110 111      | 37 |        | STC            | CY = 1           | C           | 4     |
| 00 111 111      | 3F |        | CMC            | CY ← NOT CY      | C           | 4     |
| <b>Група 01</b> |    |        |                |                  |             |       |
| 01 110 110      | 76 |        | HLT            |                  |             | 7 (?) |
| 01 DDD SSS      |    |        | MOV DDD, SSS   | DDD ← SSS        |             | 5/7   |
| <b>Група 10</b> |    |        |                |                  |             |       |
| 10 000 SSS      |    |        | ADD SSS        | A ← A+SSS        | все         | 4/7   |
| 10 001 SSS      |    |        | ADC SSS        | A ← A+SSS+CY     | все         | 4/7   |
| 10 010 SSS      |    |        | SUB SSS        | A ← A-SSS        | все         | 4/7   |
| 10 011 SSS      |    |        | SBB SSS        | A ← A-SSS-CY     | все         | 4/7   |
| 10 100 SSS      |    |        | ANA SSS        | A ← A AND SSS    | все         | 4/7   |
| 10 101 SSS      |    |        | XRA SSS        | A ← A XOR SSS    | все         | 4/7   |
| 10 110 SSS      |    |        | ORA SSS        | A ← A OR SSS     | все         | 4/7   |

|                  |     |     |    |        |               |                       |                    |         |
|------------------|-----|-----|----|--------|---------------|-----------------------|--------------------|---------|
| 10               | 111 | SSS |    |        | CMP SSS       | COMPARE A, SSS        | все                | 4/7     |
| <b>Группа 11</b> |     |     |    |        |               |                       |                    |         |
| 11               | XXX | 000 |    |        | RETIF         | RETURN IF XXX IS TRUE |                    | 5 (10)  |
| 11               | RP0 | 001 |    |        | POP RP        |                       | С (только POP PSW) | 10      |
| 11               | 001 | 001 | C9 |        | RET           |                       |                    | 10      |
| 11               | 011 | 001 | D9 |        | ?             |                       |                    |         |
| 11               | 101 | 001 | E9 |        | PCHL          | JMP [HL]              |                    | 5       |
| 11               | 111 | 001 | F9 |        | SPHL          | SP <- HL              |                    | 5       |
| 11               | XXX | 010 |    | ADDR16 | JMP IF ADDR16 | JUMP IF XXX IS TRUE   |                    | 10      |
| 11               | 000 | 011 | C3 | ADDR16 | JMP ADDR16    |                       |                    | 10      |
| 11               | 010 | 011 | D3 | PORT8  | OUT PORT8     | [PORT8] <- A          |                    | 10      |
| 11               | 011 | 011 | DB | PORT8  | IN PORT8      | A <- [PORT8]          |                    | 10      |
| 11               | 001 | 011 | CB |        | ?             |                       |                    |         |
| 11               | 100 | 011 | E3 |        | XTHL          | [SP] <-> HL           |                    | 18      |
| 11               | 101 | 011 | EB |        | XCHG          | DE <-> HL             |                    | 4       |
| 11               | 110 | 011 | F3 |        | DI            | INT DISABLE           |                    | 4       |
| 11               | 111 | 011 | FB |        | EI            | INT ENABLE            |                    | 4       |
| 11               | XXX | 100 |    | ADDR16 | CALLIF ADDR16 | CALL IF XXX IS TRUE   |                    | 11 (17) |
| 11               | RP0 | 101 |    |        | PUSH RP       |                       |                    | 11      |
| 11               | 001 | 101 | CD | ADDR16 | CALL ADDR16   |                       |                    | 17      |
| 11               | 011 | 101 | DD |        | ?             |                       |                    |         |
| 11               | 101 | 101 | ED |        | ?             |                       |                    |         |
| 11               | 111 | 101 | FD |        | ?             |                       |                    |         |
| 11               | 000 | 110 | C6 | DATA8  | ADI DATA8     | A <- A+DATA8          | все                | 7       |
| 11               | 001 | 110 | CE | DATA8  | ACI DATA8     | A <- A+DATA8+CY       | все                | 7       |
| 11               | 010 | 110 | D6 | DATA8  | SUI DATA8     | A <- A-DATA8          | все                | 7       |
| 11               | 011 | 110 | DE | DATA8  | SBI DATA8     | A <- A-DATA8-CY       | все                | 7       |
| 11               | 100 | 110 | E6 | DATA8  | ANI DATA8     | A <- A AND DATA8      | все                | 7       |
| 11               | 101 | 110 | EE | DATA8  | XRI DATA8     | A <- A XOR DATA8      | все                | 7       |
| 11               | 110 | 110 | F6 | DATA8  | ORI DATA8     | A <- A OR DATA8       | все                | 7       |
| 11               | 111 | 110 | FE | DATA8  | CPI DATA8     | COMPARE A, DATA8      | все                | 7       |

|            |  |         |         |  |    |
|------------|--|---------|---------|--|----|
| 11 NNN 111 |  | RST NNN | INT NNN |  | 11 |
|------------|--|---------|---------|--|----|

## Расшифровка сокращений

### DDD, SSS

- 000=B
- 001=C
- 010=D
- 011=E
- 100=H
- 101=L
- 110=M (т.е. [HL])
- 111=A

### R

- 0=BC
- 1=DE

### RP

- 0=BC
- 1=DE
- 2=HL
- 3=SP или PSW (Status word) для PUSH/POP

### XXX

- 000: NOT ZERO
- 001: ZERO
- 010: NOT CARRY
- 011: CARRY
- 100: NOT PARITY
- 101: PARITY
- 110: POSITIVE
- 111: NEGATIVE

## Размещение двухбайтовых адресов в памяти

Для всех трёхбайтовых команд (CALL, JMP, STA, ...) второй байт команды содержит младший байт, третий байт команды - старший байт. Т.е. команда LXI BC, 0x00ff будет состоять из трёх байт в такой последовательности 0x01 0xff 0x00.

## Операции со стеком

16 байтный регистр SP служит указателем на вершину стека. Помещение 16 байтного значения в стек происходит следующим образом: 1. SP декрементируется; 2. старший байт заносится в ячейку памяти по адресу на который указывает SP; 3. SP декрементируется; 4. младший байт заносится в ячейку памяти по адресу SP.

Извлечение 16 байтного значения в стек происходит следующим образом: 1. из ячейки по адресу SP извлекается младший байт; 2. SP инкрементируется; 3. из ячейки по адресу SP извлекается старший байт; 4. SP инкрементируется.

## Начальное состояние процессора

После подачи сигнала RESET процессор обнуляет регистр PC. Все остальные регистры остаются неизменными. Если RESET происходит в начальный момент работы процессора, все регистры содержат случайные величины.

## Особенности

- Вычисление флага AC: Обсуждение на форуме zx.pk.ru (<http://www.zx.pk.ru/showthread.php?t=9826>)

Источник — «[http://www.emuverse.ru/w/index.php?title=Intel\\_8080/Коды\\_команд&oldid=1414](http://www.emuverse.ru/w/index.php?title=Intel_8080/Коды_команд&oldid=1414)»

Категория: Intel 8080

- 
- Последнее изменение этой страницы: 02:31, 31 марта 2009.