Data transfer instructions of 8086 microprocessor

General purpose byte or word transfer instructions:

- **MOV:** copy byte or word from specified source to specified destination
- **PUSH:** copy specified word to top of stack.
- **POP:** copy word from top of stack to specified location
- **PUSHA:** copy all registers to stack
- **POPA:** copy words from stack to all registers.
- **XCHG:** Exchange bytes or exchange words

These are I/O port transfer instructions:

- **IN:** copy a byte or word from specific port to accumulator
- **OUT:** copy a byte or word from accumulator to specific port

Special address transfer Instructions:

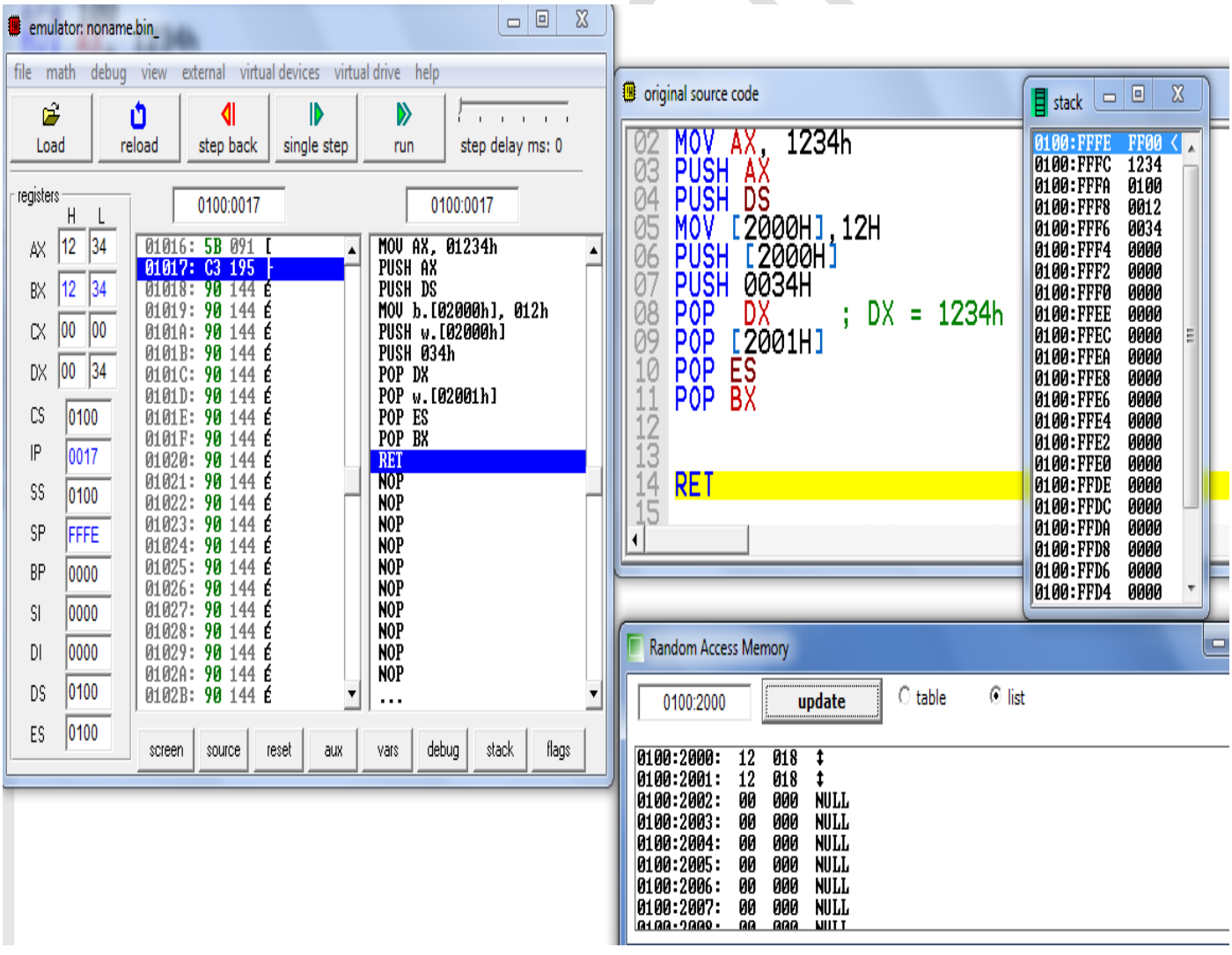
- **LEA:** load effective address of operand into specified register
- **LDS:** load DS register and other specified register from memory
- **LES:** load ES register and other specified register from memory

Flag transfer instructions:

- **LAHF:** load AH with the low byte of flag register
- **SAHF:** Stores AH register to low byte of flag register
- **PUSHF:** copy flag register to top of stack
- **POPF:** copy top of stack word to flag register

PUSH	REG SREG memory immediate	Store 16 bit value in the stack. Note: **PUSH immediate** works only on 80186 CPU and later! Algorithm: • SP = SP - 2 • SS:[SP] (top of the stack) = operand Example: MOV AX, 1234h PUSH AX POP DX ; DX = 1234h RET C Z S O P A unchanged

POP	REG SREG memory	Get 16 bit value from the stack. Algorithm: • operand = SS:[SP] (top of the stack) • SP = SP + 2 Example: MOV AX, 1234h PUSH AX POP DX ; DX = 1234h RET C Z S O P A unchanged



<table>
<tr>
<td>PUSHA</td>
<td>No operands</td>
<td>Push all general purpose registers AX, CX, DX, BX, SP, BP, SI, DI in the stack.
Original value of SP register (before PUSHA) is used.

Note: this instruction works only on 80186 CPU and later!

Algorithm:

• PUSH AX
• PUSH CX
• PUSH DX
• PUSH BX
• PUSH SP
• PUSH BP
• PUSH SI
• PUSH DI

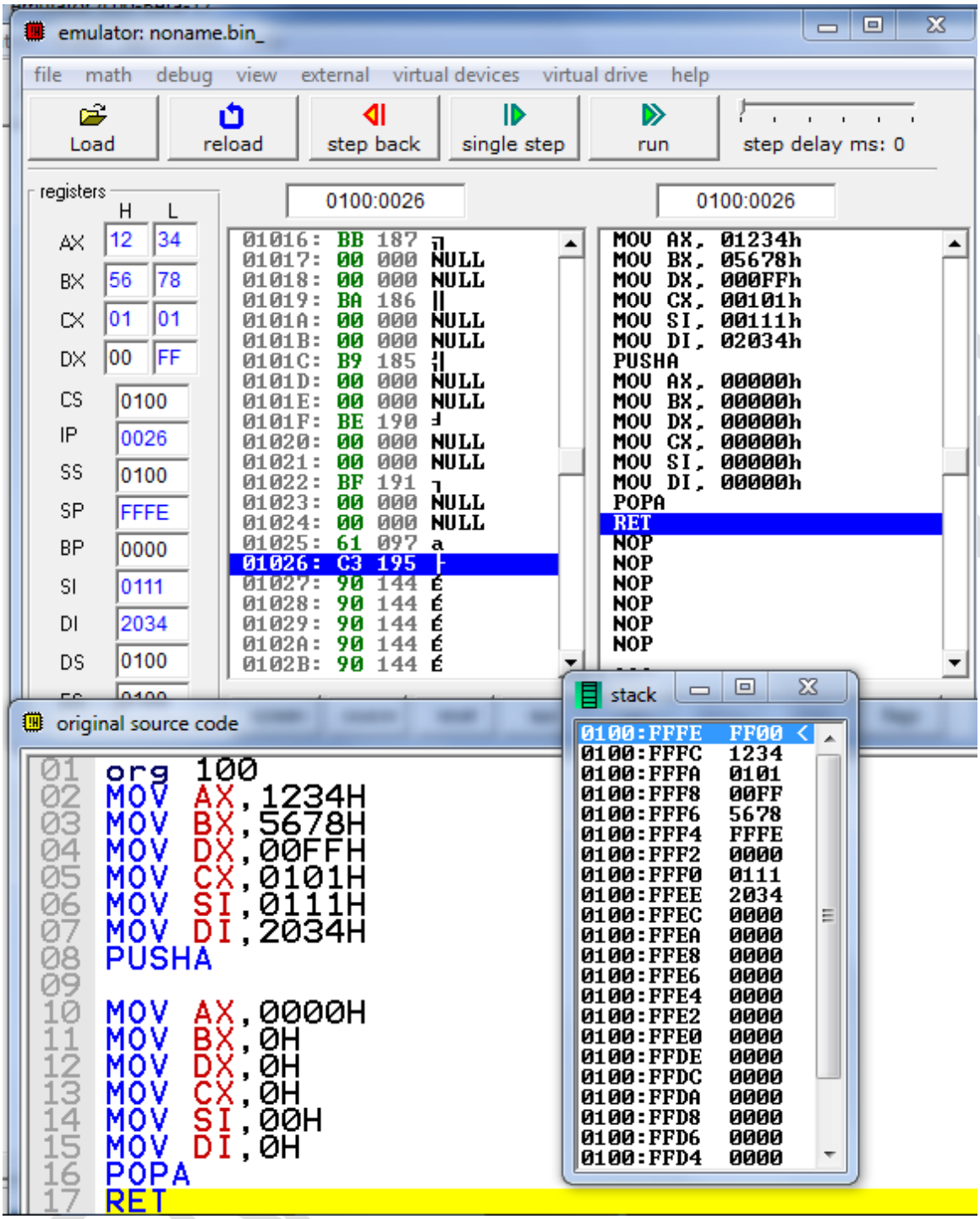
<table><tr><td>C</td><td>Z</td><td>S</td><td>O</td><td>P</td><td>A</td></tr><tr><td colspan="6">unchanged</td></tr></table></td>
</tr>
<tr>
<td>POPA</td>
<td>No operands</td>
<td>Pop all general purpose registers DI, SI, BP, SP, BX, DX, CX, AX from the stack.
SP value is ignored, it is Popped but not set to SP register).

Note: this instruction works only on 80186 CPU and later!

Algorithm:

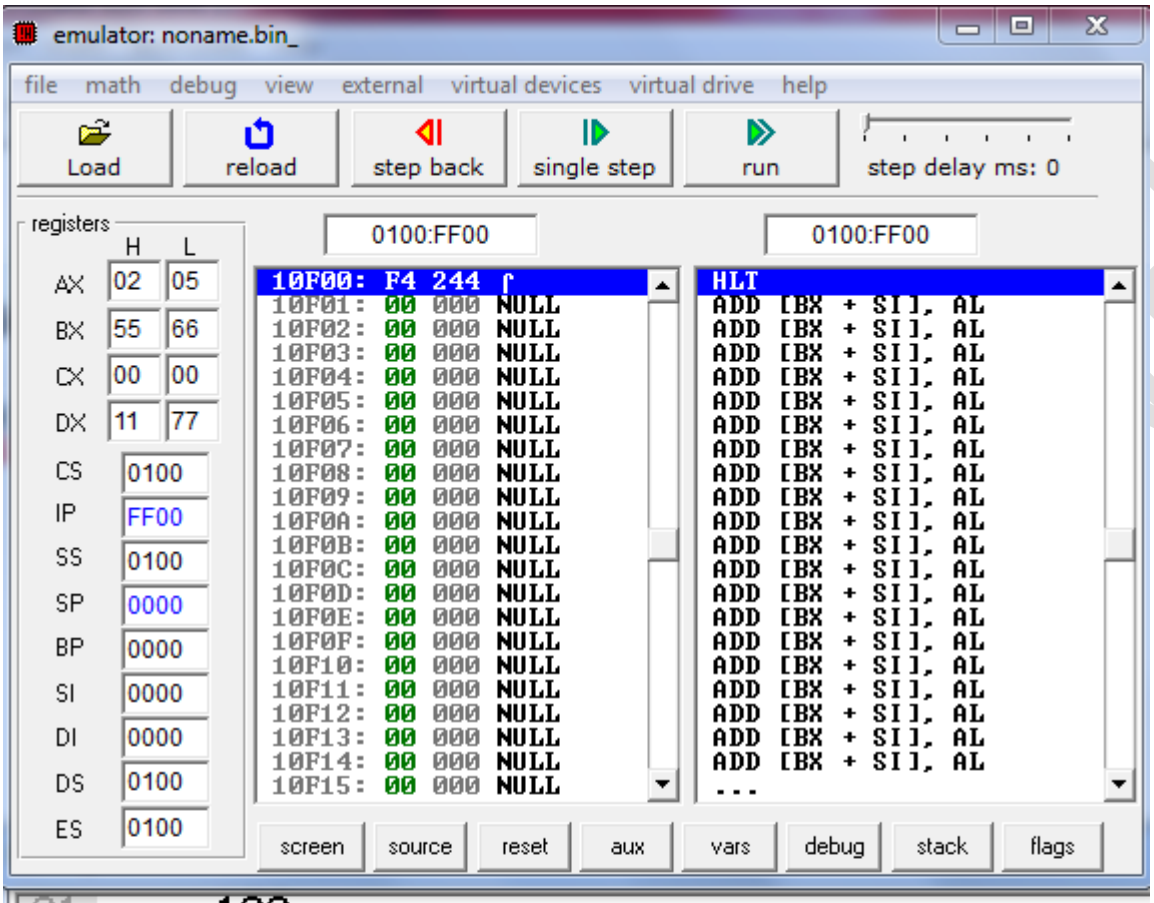
• POP DI
• POP SI
• POP BP
• POP xx (SP value ignored)
• POP BX
• POP DX
• POP CX
• POP AX

<table><tr><td>C</td><td>Z</td><td>S</td><td>O</td><td>P</td><td>A</td></tr><tr><td colspan="6">unchanged</td></tr></table></td>
</tr>
</table>

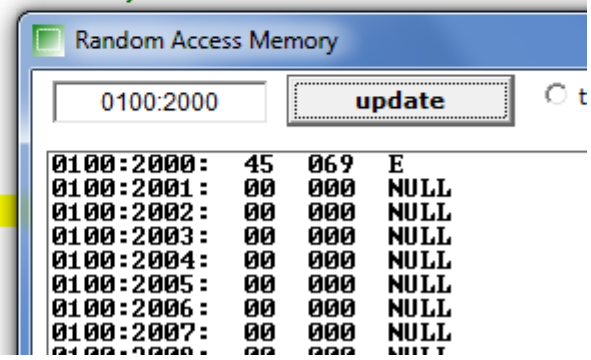


XCHG	REG, memory
	memory, REG
	REG, REG
		Exchange values of two operands.
		
		Algorithm:
		
		operand1 < - > operand2
		
		Example:
		MOV AL, 5
		MOV AH, 2
		XCHG AL, AH ; AL = 2, AH = 5
		XCHG AL, AH ; AL = 5, AH = 2
		RET

		C Z S O P A unchanged

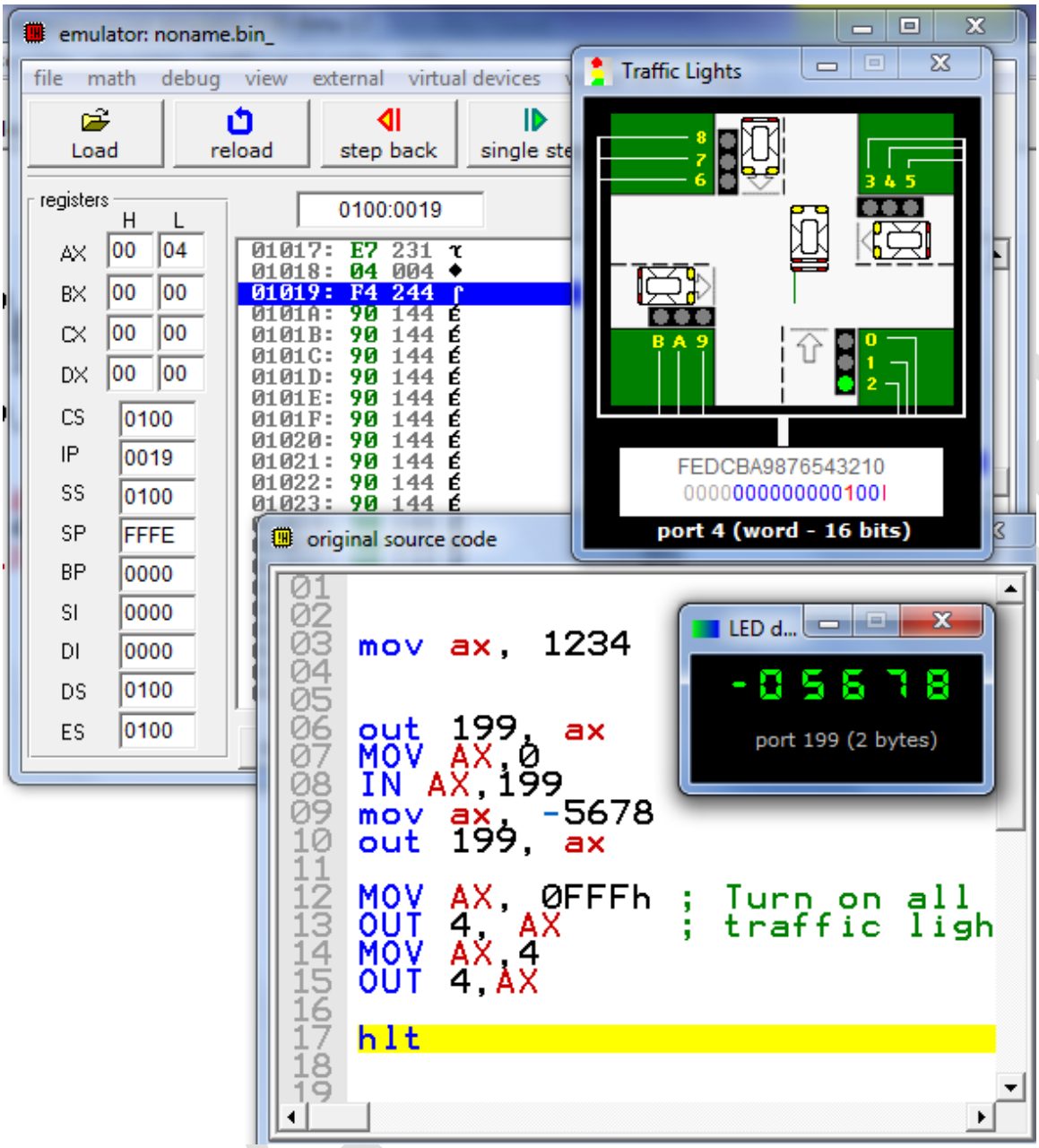



```
01 org 100
02 MOV AL, 5
03 MOV AH, 2
04 XCHG AL, AH    ; AL = 2, AH = 5
05 XCHG AL, AH    ; AL = 5, AH = 2
06 MOV [2000H],45H
07 MOV BX,1177H
08 MOV DX,5566H
09 XCHG BX,[2000H]
10 XCHG [2000H],BX
11 XCHG DX,BX
12 RET
13
14
15
16
```



IN	AL, im.byte AL, DX AX, im.byte AX, DX	Input from port into **AL** or **AX**. Second operand is a port number. If required to access port number over 255 - **DX** register should be used. Example: IN AX, 4 C Z S O P A unchanged

OUT	im.byte, AL
	im.byte, AX
	DX, AL
	DX, AX
		Output from AL or AX to port.
		First operand is a port number. If required to access port number over 255 - DX register should be used.
		
		Example:
		MOV AX, 0FFFh ; Turn on all
		OUT 4, AX ; traffic lights.
		
		MOV AL, 100b ; Turn on the third
		OUT 7, AL ; magnet of the stepper-motor.
		C Z S O P A
		unchanged


emulator: noname.bin_
file
math
debug
view
external
virtual devices
Load
reload
step back
registers
H
L
AX 00 04
BX 00 00
CX 00 00
DX 00 00
CS 0100
IP 0019
SS 0100
SP FFFE
BP 0000
SI 0000
DI 0000
DS 0100
ES 0100
0100:0019
01017: E7 231 τ
01018: 04 004 ♦
01019: F4 244 ⌠
0101A: 90 144 É
0101B: 90 144 É
0101C: 90 144 É
0101D: 90 144 É
0101E: 90 144 É
0101F: 90 144 É
01020: 90 144 É
01021: 90 144 É
01022: 90 144 É
01023: 90 144 É
Traffic Lights
8 7 6
3 4 5
B A 9
0 1 2
FEDCBA9876543210
0000000000001001
port 4 (word - 16 bits)
original source code
01
02
03 mov ax, 1234
04
05
06 out 199, ax
07 MOV AX,0
08 IN AX,199
09 mov ax, -5678
10 out 199, ax
11
12 MOV AX, 0FFFh ; Turn on all
13 OUT 4, AX ; traffic ligh
14 MOV AX,4
15 OUT 4,AX
16
17 hlt
18
19
LED d...
-05678
port 199 (2 bytes)