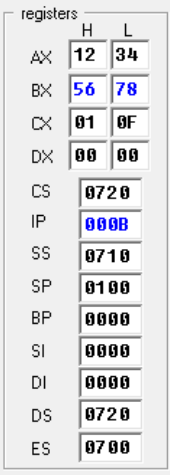
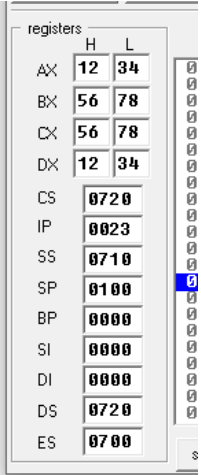
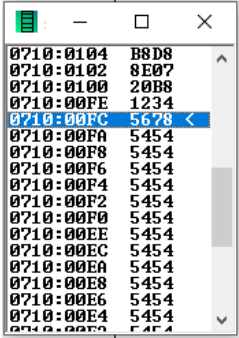
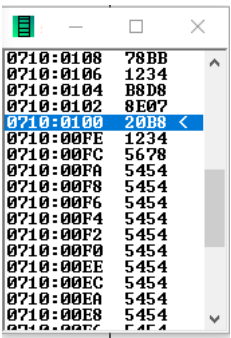
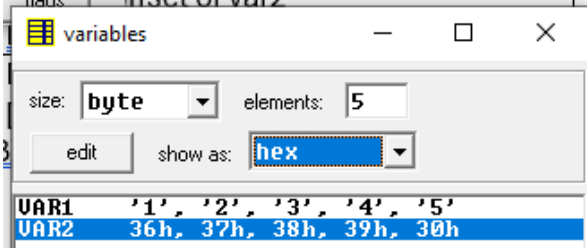
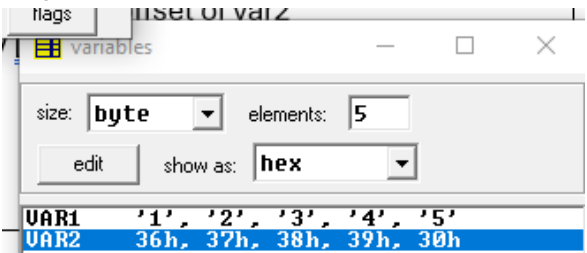
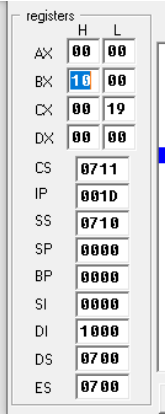
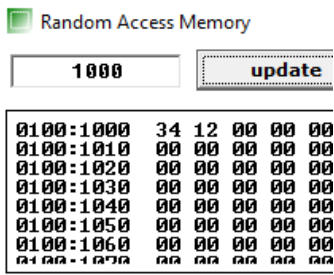
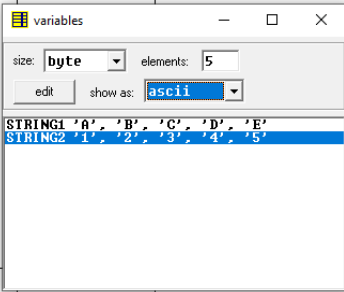
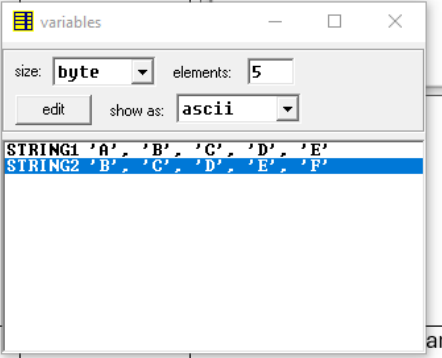
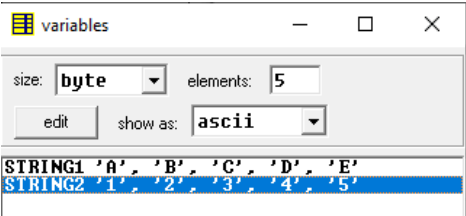
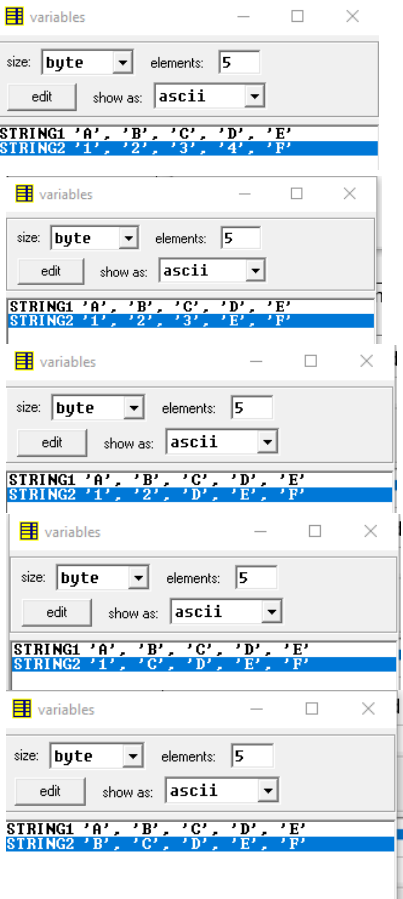
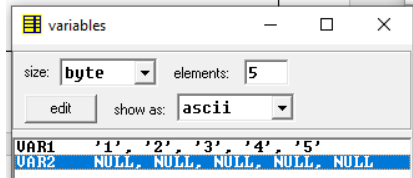
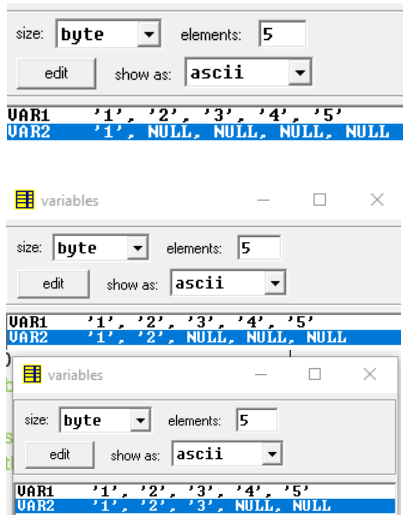
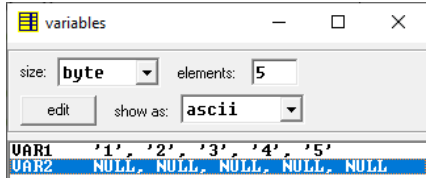
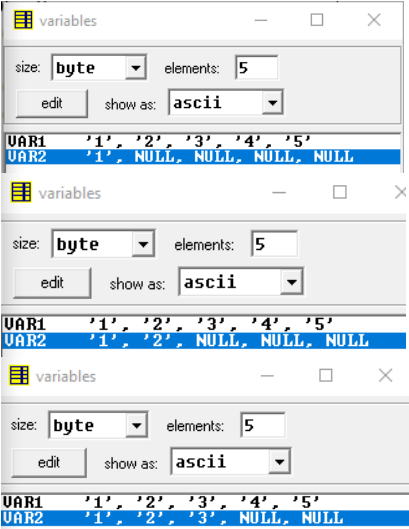
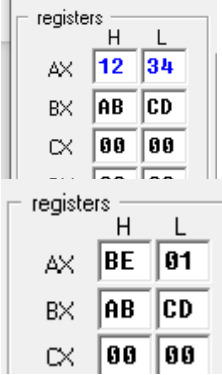
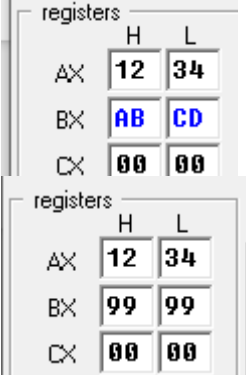
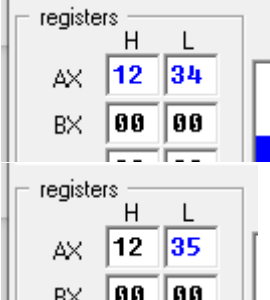

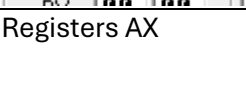


Data Movement Instructions		
Code	Instruction	Screenshot
<pre>.MODEL SMALL .STACK 100H .DATA .CODE MOV AX, @DATA MOV DS, AX mov ax,1234h mov bx,5678h push ax push bx pop cx pop dx</pre>	Push and Pop	<div>Register(Before and After)</div> <div>Stack(After PUSH and After POP)</div> <div><div>Register Before:</div><div>Register After:</div></div> <div><div>After Push:</div><div>After Pop:</div></div>
<pre>.model small .data var1 db '12345',13,10,'\$' var2 db '67890',13,10,'\$' .code MOV AX, @DATA MOV DS, AX ; init DS LEA SI, var1 ; offset of var1 LEA DI, var2; offset of var2 MOV DI,OFFSET VAR2</pre>	LEA	<div>Register SI and DI</div> <div>Variable var1 and var2 (Before and After)</div> <div>Before:</div>  <div>After:</div> 

<pre>MOV DI,1000H MOV [DI],1234H LEA BX,[DI]</pre>		<div> <div>Register BX</div> <div>Memory pointed by DI</div> <div>  </div> <div>  </div> </div>
<pre>.model small .data string1 db 'ABCDE',13,10,'\$' string2 db '12345',13,10,'\$' .code MOV AX, @DATA MOV ES, AX ; initialize ES MOV DS, AX LEA SI, string1 LEA DI, string2 CLD ; read from location 0 mov ax,0000h mov cx,5 ;counter ulit: lodsb ;AL = DS:[SI] inc al stosb ;ES:[DI] = AL loop ulit</pre>	<div> <div>LODSB and STOSB</div> <div>W/ CLD</div> </div>	<div> <div>Variables before</div> <div>  </div> <div>Variables after</div> <div>  </div> </div>
<pre>.model small .data string1 db 'ABCDE' string2 db '12345' .code MOV AX, @DATA MOV ES, AX ; initialize ES MOV DS, AX LEA SI, string1+04h LEA DI, string2+04h STD ; Read string from end to start mov ax,0000h mov cx,5 ulit: lodsb ;AL = DS:[SI] inc al Increase ASCII value (e.g., 'E' -> 'F') stosb ;ES:[DI] = AL loop ulit</pre>	<div> <div>LODSB and STOSB</div> <div>W/ STD</div> </div>	<div> <div>Variables (before lods and stos)</div> <div>  </div> </div>

		<p>Variables (after lods and stos)</p> 
<pre>.model small .data var1 db '12345' var2 db 3 dup(0) .code MOV AX, @DATA MOV DS, AX ; init DS MOV ES, AX ; init ES LEA SI, var1 ; source LEA DI, var2 ; destination CLD ; DF = 0 MOVSB ; Move byte from DS:[SI] to ES:[DI], SI++, DI++ MOVSB ; Move second byte MOVSB ; Move third byte</pre>	MOVSB	<p>Variables before movs</p>  <p>Variables after movs</p> 
<pre>.model small .data var1 db '12345' var2 db 3 dup(0) .code MOV AX, @DATA MOV DS, AX ; init DS MOV ES, AX ; init ES LEA SI, var1 ; source LEA DI, var2 ; destination</pre>	MOVSB /W REP	<p>Variables before movs</p> 

CLD ; DF = 0 mov cx,3 ; SET COUNTER FOR REP rep MOVSB		Variables after movs 
---	--	--

Code	Instruction	Screenshot
<pre> ;addition mov ax,1234h mov bx,0ABCDh add ax,bx </pre>		Registers AX and BX 
<pre> ;subtraction mov ax,1234h mov bx,0ABCDH sub bx,ax </pre>		Registers AX and BX 
<pre> ;increment mov ax,1234h inc ax </pre>		Registers AX 
<pre> ;decrement mov ax,1234h dec ax </pre>		Registers AX 
<pre> ;multiply 8-bit mov al,20h mov ah,10h mul ah </pre>		Registers AX 

		<div> <div> <div>registers</div> <div> <div>H</div> <div>L</div> </div> <div> <div>AX</div> <div>10</div> <div>20</div> </div> </div> <div> <div>registers</div> <div> <div>H</div> <div>L</div> </div> <div> <div>AX</div> <div>02</div> <div>00</div> </div> <div> <div>DX</div> <div>00</div> <div>00</div> </div> </div> </div>
<pre> ;multiply 16-bit mov ax,0ABCDh mov bx,100h mul bx </pre>		<div>Registers AX and DX</div> <div> <div> <div>registers</div> <div> <div>H</div> <div>L</div> </div> <div> <div>AX</div> <div>AB</div> <div>CD</div> </div> <div> <div>BX</div> <div>01</div> <div>00</div> </div> </div> <div> <div>registers</div> <div> <div>H</div> <div>L</div> </div> <div> <div>AX</div> <div>CD</div> <div>00</div> </div> <div> <div>BX</div> <div>01</div> <div>00</div> </div> </div> </div>
<pre> ;division 8-bit mov al,12h mov bl,10h div bl </pre>		<div>Registers AX</div> <div> <div> <div>registers</div> <div> <div>H</div> <div>L</div> </div> <div> <div>AX</div> <div>00</div> <div>12</div> </div> <div> <div>BX</div> <div>00</div> <div>10</div> </div> <div> <div>CX</div> <div>00</div> <div>00</div> </div> <div> <div>DX</div> <div>00</div> <div>00</div> </div> </div> <div> <div>registers</div> <div> <div>H</div> <div>L</div> </div> <div> <div>AX</div> <div>02</div> <div>01</div> </div> <div> <div>BX</div> <div>00</div> <div>10</div> </div> <div> <div>CX</div> <div>00</div> <div>00</div> </div> <div> <div>DX</div> <div>00</div> <div>00</div> </div> </div> </div>
<pre> ;division 16-bit mov ax,1234h mov bx,100h div bx </pre>	<div>1234/100 = 12.34</div>	<div>Registers AX and DX</div> <div> <div> <div>registers</div> <div> <div>H</div> <div>L</div> </div> <div> <div>AX</div> <div>12</div> <div>34</div> </div> <div> <div>BX</div> <div>01</div> <div>00</div> </div> <div> <div>CX</div> <div>00</div> <div>00</div> </div> <div> <div>DX</div> <div>00</div> <div>00</div> </div> </div> <div> <div>registers</div> <div> <div>H</div> <div>L</div> </div> <div> <div>AX</div> <div>00</div> <div>12</div> </div> <div> <div>BX</div> <div>01</div> <div>00</div> </div> <div> <div>CX</div> <div>00</div> <div>00</div> </div> <div> <div>DX</div> <div>00</div> <div>34</div> </div> </div> </div>
<pre> ;add w/ carry stc mov ax,1234h mov bx,0ABCDH adc ax,bx </pre>		<div>Register and Flag</div> <div> <div> <div>registers</div> <div> <div>H</div> <div>L</div> </div> <div> <div>AX</div> <div>1C</div> <div>F2</div> </div> <div> <div>BX</div> <div>0A</div> <div>BD</div> </div> <div> <div>CX</div> <div>00</div> <div>00</div> </div> <div> <div>DX</div> <div>00</div> <div>00</div> </div> </div> <div> <div>flags</div> <div> <div>CF</div> <div>0</div> </div> <div> <div>ZF</div> <div>0</div> </div> <div> <div>SF</div> <div>0</div> </div> <div> <div>OF</div> <div>0</div> </div> <div> <div>PF</div> <div>0</div> </div> <div> <div>AF</div> <div>1</div> </div> <div> <div>IF</div> <div>1</div> </div> <div> <div>DF</div> <div>0</div> </div> <div>analyse</div> </div> </div>
<pre> ;sub w/ borrow stc mov ax,1234h mov bx,0ABCDH sbb bx,ax </pre>		<div>Register and Flag</div> <div> <div> <div>flags</div> <div> <div>CF</div> <div>0</div> </div> <div> <div>ZF</div> <div>0</div> </div> <div> <div>SF</div> <div>1</div> </div> <div> <div>OF</div> <div>0</div> </div> <div> <div>PF</div> <div>0</div> </div> <div> <div>AF</div> <div>0</div> </div> <div> <div>IF</div> <div>1</div> </div> <div> <div>DF</div> <div>0</div> </div> <div>analyse</div> </div> </div>

		<div>registers</div> <table><tr><td></td><td>H</td><td>L</td></tr><tr><td>AX</td><td>12</td><td>34</td></tr><tr><td>BX</td><td>99</td><td>98</td></tr><tr><td>CX</td><td>00</td><td>00</td></tr><tr><td>DX</td><td>00</td><td>00</td></tr></table>		H	L	AX	12	34	BX	99	98	CX	00	00	DX	00	00															
	H	L																														
AX	12	34																														
BX	99	98																														
CX	00	00																														
DX	00	00																														
<pre>;and mov ax,1234h mov bx,0ABCDH and ax,bx</pre>		Register AX and BX <div><div>registers</div><table><tr><td></td><td>H</td><td>L</td></tr><tr><td>AX</td><td>12</td><td>34</td></tr><tr><td>BX</td><td>AB</td><td>CD</td></tr><tr><td>CX</td><td>00</td><td>00</td></tr><tr><td>DX</td><td>00</td><td>00</td></tr></table><div>registers</div><table><tr><td></td><td>H</td><td>L</td></tr><tr><td>AX</td><td>02</td><td>04</td></tr><tr><td>BX</td><td>AB</td><td>CD</td></tr><tr><td>CX</td><td>00</td><td>00</td></tr><tr><td>DX</td><td>00</td><td>00</td></tr></table></div>		H	L	AX	12	34	BX	AB	CD	CX	00	00	DX	00	00		H	L	AX	02	04	BX	AB	CD	CX	00	00	DX	00	00
	H	L																														
AX	12	34																														
BX	AB	CD																														
CX	00	00																														
DX	00	00																														
	H	L																														
AX	02	04																														
BX	AB	CD																														
CX	00	00																														
DX	00	00																														
<pre>;or mov ax,1234h mov bx,0ABCDH or ax,bx</pre>		Register AX and BX <div><div>registers</div><table><tr><td></td><td>H</td><td>L</td></tr><tr><td>AX</td><td>12</td><td>34</td></tr><tr><td>BX</td><td>AB</td><td>CD</td></tr><tr><td>CX</td><td>00</td><td>00</td></tr><tr><td>DX</td><td>00</td><td>00</td></tr></table><div>registers</div><table><tr><td></td><td>H</td><td>L</td></tr><tr><td>AX</td><td>BB</td><td>FD</td></tr><tr><td>BX</td><td>AB</td><td>CD</td></tr><tr><td>CX</td><td>00</td><td>00</td></tr><tr><td>DX</td><td>00</td><td>00</td></tr></table></div>		H	L	AX	12	34	BX	AB	CD	CX	00	00	DX	00	00		H	L	AX	BB	FD	BX	AB	CD	CX	00	00	DX	00	00
	H	L																														
AX	12	34																														
BX	AB	CD																														
CX	00	00																														
DX	00	00																														
	H	L																														
AX	BB	FD																														
BX	AB	CD																														
CX	00	00																														
DX	00	00																														
<pre>;xor mov ax,1234h mov bx,0ABCDH xor ax,bx</pre>		Register AX and BX <div><div>registers</div><table><tr><td></td><td>H</td><td>L</td></tr><tr><td>AX</td><td>12</td><td>34</td></tr><tr><td>BX</td><td>AB</td><td>CD</td></tr><tr><td>CX</td><td>00</td><td>00</td></tr><tr><td>DX</td><td>00</td><td>00</td></tr></table><div>registers</div><table><tr><td></td><td>H</td><td>L</td></tr><tr><td>AX</td><td>B9</td><td>F9</td></tr><tr><td>BX</td><td>AB</td><td>CD</td></tr><tr><td>CX</td><td>00</td><td>00</td></tr><tr><td>DX</td><td>00</td><td>00</td></tr></table></div>		H	L	AX	12	34	BX	AB	CD	CX	00	00	DX	00	00		H	L	AX	B9	F9	BX	AB	CD	CX	00	00	DX	00	00
	H	L																														
AX	12	34																														
BX	AB	CD																														
CX	00	00																														
DX	00	00																														
	H	L																														
AX	B9	F9																														
BX	AB	CD																														
CX	00	00																														
DX	00	00																														
<pre>;not mov ax,1234h not ax</pre>		Register AX <div><div>registers</div><table><tr><td></td><td>H</td><td>L</td></tr><tr><td>AX</td><td>12</td><td>34</td></tr><tr><td>BX</td><td>00</td><td>00</td></tr><tr><td>CX</td><td>00</td><td>00</td></tr><tr><td>DX</td><td>00</td><td>00</td></tr></table><div>registers</div><table><tr><td></td><td>H</td><td>L</td></tr><tr><td>AX</td><td>ED</td><td>CB</td></tr><tr><td>BX</td><td>00</td><td>00</td></tr><tr><td>CX</td><td>00</td><td>00</td></tr><tr><td>DX</td><td>00</td><td>00</td></tr></table></div>		H	L	AX	12	34	BX	00	00	CX	00	00	DX	00	00		H	L	AX	ED	CB	BX	00	00	CX	00	00	DX	00	00
	H	L																														
AX	12	34																														
BX	00	00																														
CX	00	00																														
DX	00	00																														
	H	L																														
AX	ED	CB																														
BX	00	00																														
CX	00	00																														
DX	00	00																														
<pre>;neg mov ax,1234h neg ax</pre>		Register AX <div><div>registers</div><table><tr><td></td><td>H</td><td>L</td></tr><tr><td>AX</td><td>12</td><td>34</td></tr><tr><td>BX</td><td>00</td><td>00</td></tr><tr><td>CX</td><td>00</td><td>00</td></tr><tr><td>DX</td><td>00</td><td>00</td></tr></table><div>registers</div><table><tr><td></td><td>H</td><td>L</td></tr><tr><td>AX</td><td>ED</td><td>CC</td></tr><tr><td>BX</td><td>00</td><td>00</td></tr><tr><td>CX</td><td>00</td><td>00</td></tr><tr><td>DX</td><td>00</td><td>00</td></tr></table></div>		H	L	AX	12	34	BX	00	00	CX	00	00	DX	00	00		H	L	AX	ED	CC	BX	00	00	CX	00	00	DX	00	00
	H	L																														
AX	12	34																														
BX	00	00																														
CX	00	00																														
DX	00	00																														
	H	L																														
AX	ED	CC																														
BX	00	00																														
CX	00	00																														
DX	00	00																														

```
.MODEL SMALL
.DATA
hello db 'The sum of the numbers'
num1 db '2345',10,13,'$'
num2 db '5678',10,13,'$'
sum db ' ',10,13,'$'

.CODE
mov ax, @DATA
mov ds, ax

mov si, 03h
mov cx, 05h
clc

add_loop:
    mov al, [num1+si]
    adc al, [num2+si]
    aaa
    pushf
    or al, 30h
    popf
    mov [sum+si], al
    dec si
    loop add_loop

mov ah, 09
mov dx, offset hello
int 21h

mov ah, 09
mov dx, offset sum
int 21h
```

Variables after each loop
CX 4

variables

size: **byte** elements: **1**

edit show as: **hex**

HELLO	'T'	'h'	'e'	's'	
NUM1	'2'	'3'	'4'	'5'	'0'
NUM2	35h				
SUM	20h				

CX 3

variables

size: **byte** elements: **1**

edit show as: **hex**

HELLO	'T'	'h'	'e'	's'	
NUM1	'2'	'3'	'4'	'5'	'0'
NUM2	35h				
SUM	20h				

CX 2

variables

size: **byte** elements: **1**

edit show as: **hex**

HELLO	'T'	'h'	'e'	's'	
NUM1	'2'	'3'	'4'	'5'	'0'
NUM2	35h				
SUM	20h				

CX 1

variables

size: **byte** elements: **1**

edit show as: **hex**

HELLO	'T'	'h'	'e'	's'	
NUM1	'2'	'3'	'4'	'5'	'0'
NUM2	35h				
SUM	20h				

CX 0

variables

size: **byte** elements: **1**

edit show as: **hex**

HELLO	'T'	'h'	'e'	's'	
NUM1	'2'	'3'	'4'	'5'	'0'
NUM2	35h				
SUM	38h				

emulator screen (80x25 chars)

```
The sum of the numbers
8023
```