《微机原理实验》实验报告

实验名称： 多位 16 进制加法运算实验　　指导教师：肖山林

姓名：徐睿琳　学号：23342107　专业/班级：微电子/三班　分组序号：A412-A05

实验日期：2025.11.13　实验地点：教学楼 A412　是否调课/补课：否　成绩：

目录

1 实验内容与设计

1.1 多位十六进制加法运算实验

1.1.1 实验电路图

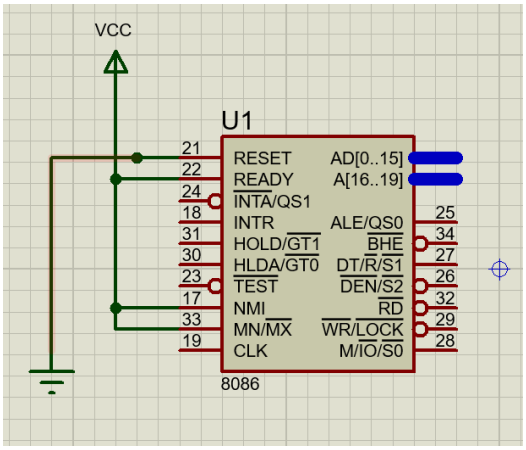


Figure 1: 实验电路图

1.1.2 实验设计

实验	实验目的	DATA 段数据	CODE 段指令	预期标志位结果

基础实验	观察“无特殊情况”的加法	NUM1 DW 1111H
		NUM2 DW 2222H
		NUM3 DW 3333H
			ADD AX,[SI+0]
			ADD AX,[SI+2]
			ADD AX,[SI+4]
				AX = 6666H
				CF=0, ZF=0
				OF=0, SF=0

扩展一	观察进位 (CF) 和零 (ZF)	NUM1 DW 0F000H NUM2 DW 00FFFH NUM3 DW 00001H	ADD AX,[SI+4]	AX = 0000H CF = 1 (进位) ZF = 1 (结果为零)
扩展二	观察溢出 (OF) 和符号 (SF)	NUM1 DW 6000H NUM2 DW 0000H NUM3 DW 6000H	ADD AX,[SI+4]	AX = C000H OF = 1 (溢出) SF = 1 (结果为负)
扩展三	观察 OF 和 CF 同时为 1	NUM1 DW 8000H NUM2 DW 0000H NUM3 DW 8000H	ADD AX,[SI+4]	AX = 0000H CF = 1 (无符号进位) OF = 1 (有符号溢出) ZF = 1
扩展四	实现 ADC (带进位加法)	NUM1 DW 0FFFFH NUM2 DW 00001H NUM3 DW 00001H	ADD AX,[SI+2] ADC AX,[SI+4]	ADD 后 CF=1 ADC 后 AX = 0002H (因为 0+1+CF)
扩展五	实现 INC (不影响 CF)	NUM1 DW 0FFFFH	MOV AX, [SI+0] INC AX	AX = 0000H ZF = 1 CF = 0 (或保持不变)

1.2 多位十六进制减法运算实验

1.2.1 实验设计

实验	实验目的	DATA 段数据	CODE 段指令	预期标志位结果
基础实验	实现基础 SUB 指令，无特殊情况	N1 DW 3333H N2 DW 1111H	MOV AX, [N1] SUB AX, [N2]	AX = 2222H CF=0 (无借位) ZF=0, SF=0, OF=0

扩展一	SUB 指令，相减为零、借位情况	N1 DW 5555H
		N2 DW 5555H
		N3 DW 0001H
			MOV AX, [N1]
			SUB AX, [N2]
			SUB AX, [N3]
				第一次 SUB 后:
				AX = 0000H
				ZF = 1 (零标志)
				第二次 SUB 后:
				AX = FFFFH
				CF = 1 (借位标志)

扩展二	SUB 溢出情况	N1 DW 8000H N2 DW 0001H	MOV AX, [N1] SUB AX, [N2]	(负 - 正 = 正) AX = 7FFFH OF = 1 (溢出标志) SF = 0 (结果为正) CF = 0 (无借位)

扩展三	掌握 SBB 指令	N1 DW 1000H
		N2 DW 2000H
		N3 DW 0001H
			MOV AX, [N1]
			SUB AX, [N2]
			SBB AX, [N3]
				SUB 后: CF = 1
				SBB 后:
				AX = AX-N3-CF
				AX = F000-1-1
				AX = EFFEH

扩展四	掌握 DEC 指令	N1 DW 0000H	MOV AX, [N1] DEC AX	AX = FFFFH CF = 0 (关键!) (DEC 不影响 CF) ZF=0, SF=1

扩展五	掌握 CMP 指令	N1 DW 5000H
		N2 DW 6000H
			MOV AX, [N1]
			CMP AX, [N1]
			CMP AX, [N2]
				第一次 CMP (相等):
				AX 不变 (5000H)
				ZF = 1, CF = 0
				第二次 CMP (小于):
				AX 不变 (5000H)
				ZF = 0, CF = 1

扩展六	掌握 NEG 指令	N1 DW 8000H	MOV AX, [N1] NEG AX	(特殊: 最小负数) AX = 8000H CF = 1 (有借位) OF = 1 (溢出)

1.3 多位十六进制乘法运算实验

1.3.1 实验设计

实验名称	实验目的	DATA 段数据	CODE 段指令	预期标志位/寄存器结果

基础实验	MUL (8 位)	N1 DB 10H (16)
		N2 DB 05H (5)
			MOV AL, [N1]
			MOV BL, [N2]
			MUL BL
				(AX = AL * BL)
				AX = 0050H (80)
				AH = 00H, 因此:
				CF = 0, OF = 0

扩展一	MUL (16 位)	N1 DW 8000H
		N2 DW 0010H
			MOV AX, [N1]
			MOV BX, [N2]
			MUL BX
				(DX:AX = AX * BX)
				AX = 0000H
				DX = 0008H
				DX != 0, 因此:
				CF = 1, OF = 1

扩展二	IMUL (8 位)	N1 DB 0FEH (-2)
		N2 DB 04H (+4)
			MOV AL, [N1]
			MOV BL, [N2]
			IMUL BL
				(AX = AL * BL)
				AX = FFF8H (-8)
				(AH 是 AL 的符号扩展)
				CF = 0, OF = 0

扩展三	IMUL (16 位)	N1 DW 4000H (+16384)
		N2 DW 0003H (+3)
			MOV AX, [N1]
			MOV BX, [N2]
			IMUL BX
				(DX:AX = AX * BX)
				AX = C000H
				DX = 0000H
				(结果 +49152 无法存入 AX)
				CF = 1, OF = 1

扩展四	MUL vs IMUL	N1 DW 0FFFFH N2 DW 0002H	MOV AX, [N1] MOV BX, [N2]	MUL (无符号): 65535 * 2 = 131070 DX:AX = 0001:FFFEH CF=1, OF=1 IMUL (有符号): -1 * 2 = -2 DX:AX = FFFF:FFFEH CF=0, OF=0

2 实验结果

2.1 多位十六进制加法运算实验

2.1.1 基础实验

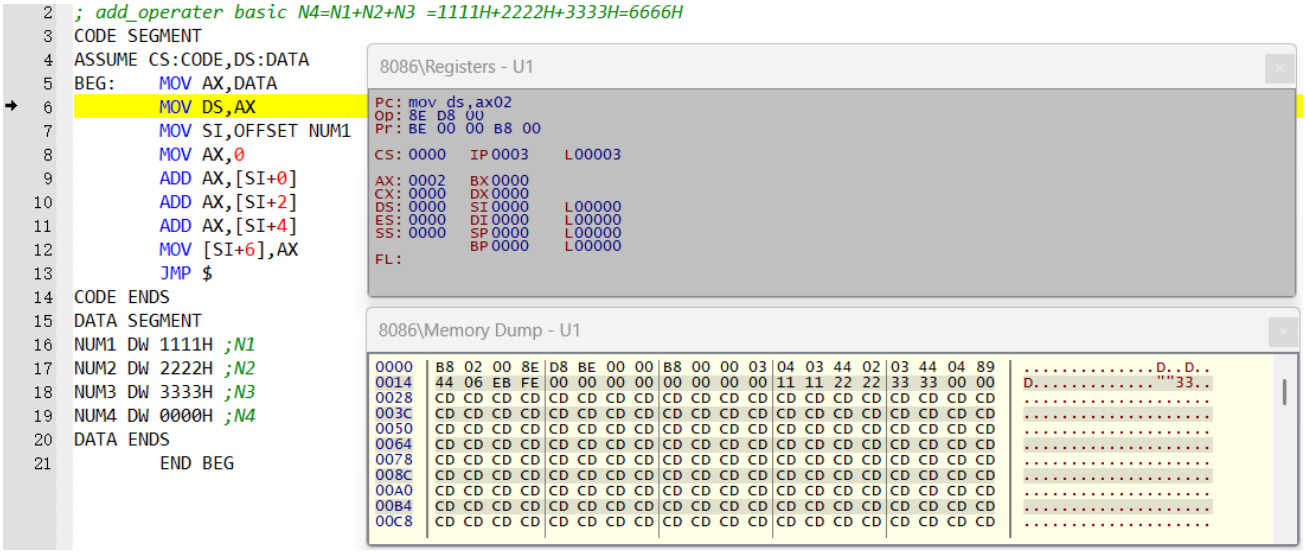


Figure 2: MOV AX, DATA 后断点

程序刚刚执行完指令 MOV AX, DATA，DATA 是数据段的段地址，汇编器分配给它的值是 0002H。AX 已经准备好了数据段的起始地址 0002H，但地址尚未交付给 DS 寄存器。需要下一步指令：MOV DS,AX 将 AX 中的 0002H 移入 DS，从而建立起正确的数据段寻址环境。

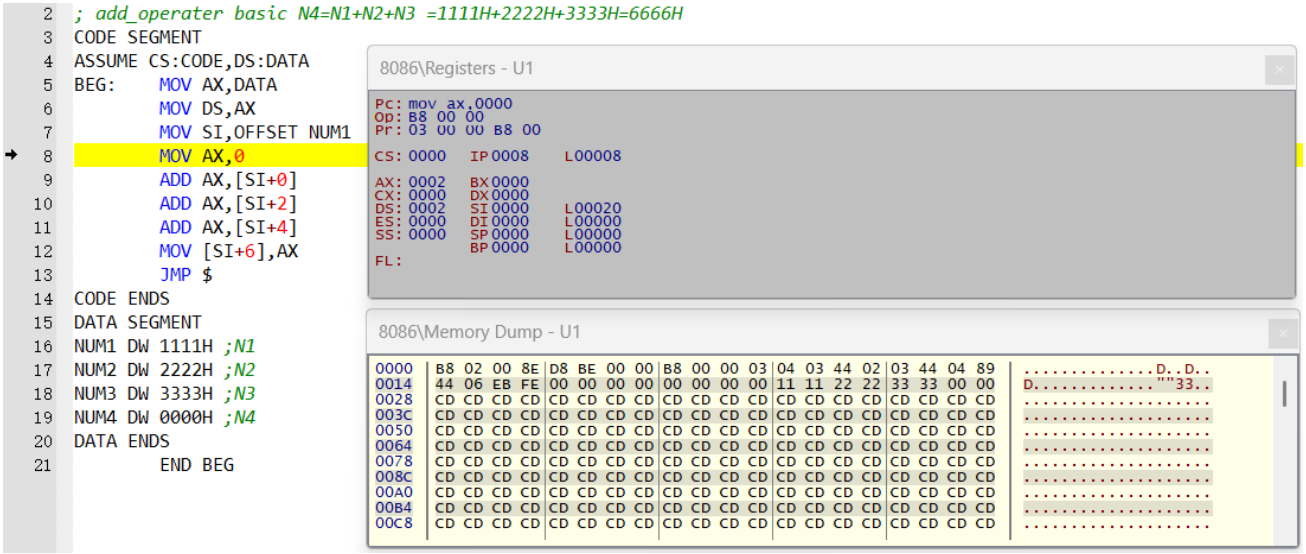
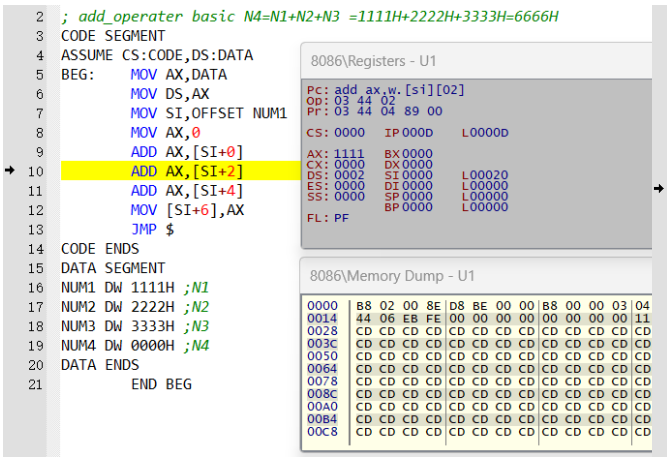
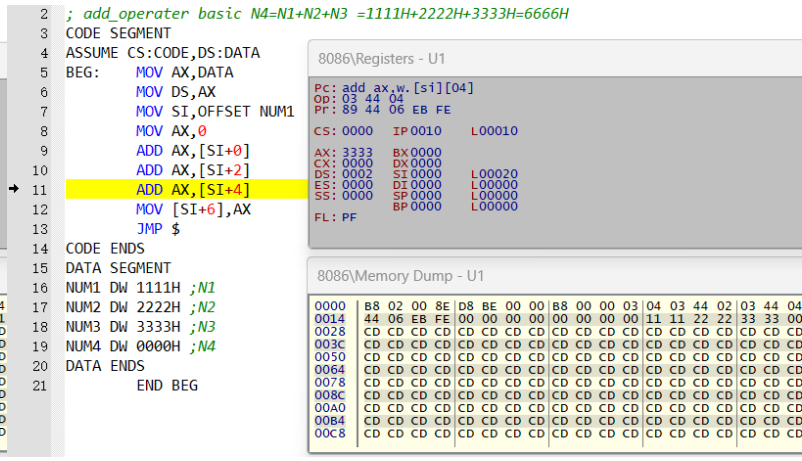


Figure 3: MOV SI,OFFSET NUM1 后断点

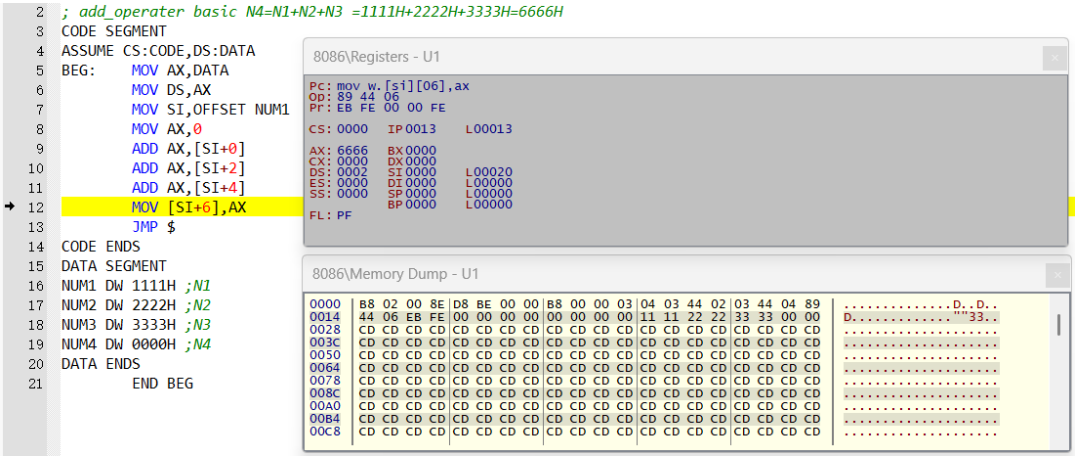
SI 寄存器现在存储了变量 NUM1 在数据段内的起始偏移地址 0002H，这为后续通过 DS:SI 访问数据做好了源指针准备。



(a) ADD AX,[SI+0]



(b) ADD AX,[SI+1]



(c) ADD AX,[SI+2]

Figure 4: 加法运算过程

如图可以看到 AX 寄存器的数值逐步累加，最终结果为 6666H，符合预期。同时，标志寄存器中的 PF 被置位，表示结果为偶数个 1，这与 6666H 的二进制表示相符。

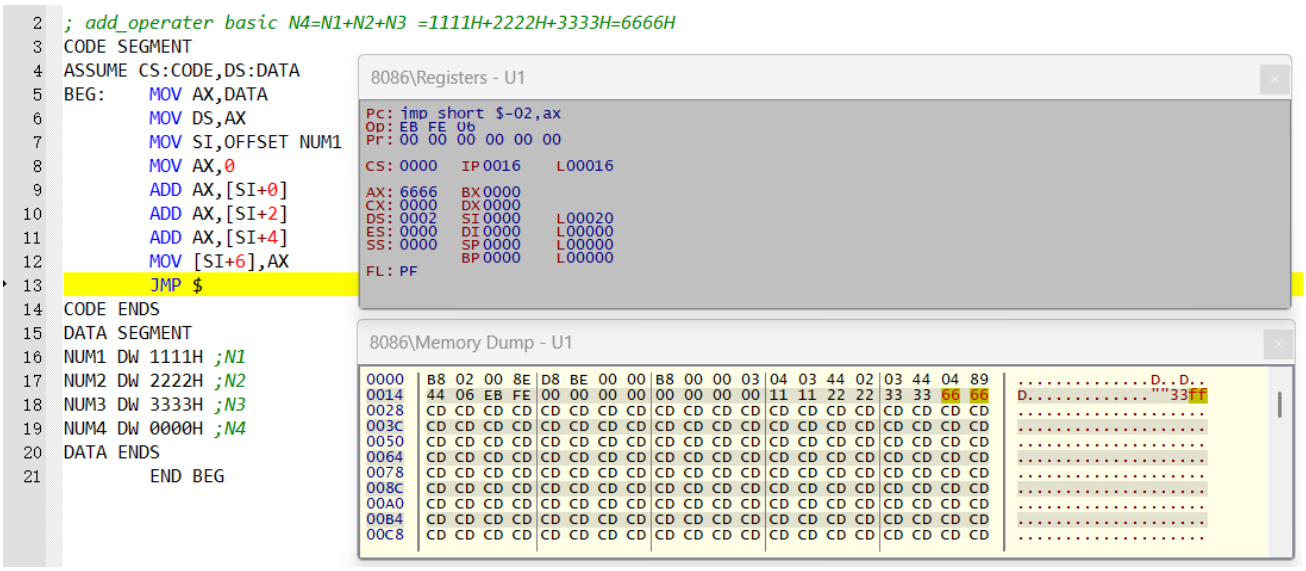


Figure 5: MOV [SI+6],AX 后断点

如图可以看到最终结果 `6666H` 已经成功存储在变量 `NUM4` 对应的内存地址中，验证了加法运算的正确性。

2.1.2 扩展一

注意：由于部分扩展实验为课后本地实验，Debug.exe 无法直接运行，因此以下扩展实验的结果均通过8086 Emulator 仿真软件进行验证，截图均来自该软件。且由于运行环境不同，部分代码与预期代码略有差异，但均实现了相同的功能。

Reg	H	L
A	ff	ff
B	0f	ff
C	00	00
D	00	00

Segments	
SS	0000
DS	0000
ES	0000

Pointers	
SP	0000
BP	0000
SI	0000
DI	0000

Flags:								
OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	0	0	0	1	0	0	1	0

Figure 6: `F000H + 0FFFH` 后断点

如图可以看到 SF 和 PF 被置位，表示结果为负数且二进制表示中有偶数个 1。

Reg	H	L
A	00	00
B	00	01
C	00	00
D	00	00

Segments	
SS	0000
DS	0000
ES	0000

Pointers	
SP	0000
BP	0000
SI	0000
DI	0000

Flags:								
OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	0	0	0	0	1	1	1	1

Figure 7: `FFFFH + 0001H` 后断点

如图可以看到 ZF,PF,CF 被置位，表示结果为零，且发生了进位, 符合预期。而且 AF（辅助进位标志）也被置位，表示低四位发生了进位。

2.1.3 扩展二

Reg	H	L
A	c0	00
B	00	00
C	60	00
D	00	00

Segments	
SS	0000
DS	0000
ES	0000

Pointers	
SP	0000
BP	0000
SI	0000
DI	0000

Flags:								
OF	DF	IF	TF	SF	ZF	AF	PF	CF
1	0	0	0	1	0	0	1	0

Figure 8: `6000H + 6000H` 后断点

如图可以看到 OF,SF,PF 被置位，表示结果为负数且二进制表示中有偶数个 1。并且发生了溢出，符合预期。

2.1.4 扩展三

Reg	H	L
A	00	00
B	00	00
C	80	00
D	00	00

Segments	
SS	0000
DS	0000
ES	0000

Pointers	
SP	0000
BP	0000
SI	0000
DI	0000

Flags:								
OF	DF	IF	TF	SF	ZF	AF	PF	CF
1	0	0	0	0	1	0	1	1

Figure 9: `8000H + 8000H` 后断点

如图可以看到 OF，ZF,PF,CF 被置位，表示结果为零，且同时发生了进位和溢出，符合预期。

2.1.5 扩展四

Reg	H	L
A	10	01
B	00	01
C	00	01
D	00	00

Segments	
SS	0000
DS	0000
ES	0000

Pointers	
SP	0000
BP	0000
SI	0000
DI	0000

Flags:								
OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	0	0	0	0	0	0	0	0

Figure 10: ADC 后断点

2.1.6 扩展五

Reg	H	L
A	00	00
B	00	01
C	00	01
D	00	00

Segments	
SS	0000
DS	0000
ES	0000

Pointers	
SP	0000
BP	0000
SI	0000
DI	0000

Flags:								
OF	DF	IF	TF	SF	ZF	AF	PF	CF
0	0	0	0	0	1	1	1	1

Figure 11: ADC 后断点

> 如图，和之前 FFFFH + 0001H 的结果一致。

2.2 多位十六进制减法运算实验

2.2.1 基础实验

3 数据分析与思考总结

附录　实验汇编代码

```
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
BEG:  MOV AX,DATA
  MOV DS,AX
  MOV SI,OFFSET NUM1
        MOV AX,0
        ADD AX,[SI+0]
        ADD AX,[SI+2]
        ADD AX,[SI+4]
        MOV [SI+6],AX
  JMP $
CODE ENDS
DATA SEGMENT
NUM1 DW 1111H ;N1
NUM2 DW 2222H ;N2
NUM3 DW 3333H ;N3
NUM4 DW 0000H ;N4
DATA ENDS
  END BEG
```

Listing 1: 加法运算-基础实验

```
OPR1: DW 0x0000
OPR2: DW 0xF000
OPR3: DW 0x0FFF
OPR4: DW 0x0001
RESULT: DW 0

start:
MOV AX, word OPR1
MOV BX, word OPR2
CLC
ADD AX, BX

MOV BX, word OPR3
ADD AX, BX

MOV BX, word OPR4
ADD AX, BX

MOV DI, OFFSET RESULT
MOV word [DI], AX
print reg
```

Listing 2: 加法运算-扩展一

```
OPR1: DW 0x6000
OPR2: DW 0x0000
OPR3: DW 0x6000
RESULT: DW 0

start:
MOV AX, word OPR1
MOV BX, word OPR2
MOV CX, word OPR3

CLC
ADD AX, BX
ADD AX, CX
```

```
14
15  MOV DI, OFFSET RESULT
16  MOV word [DI], AX
17
18  print reg
```

Listing 3: 加法运算-扩展二

```
1  CODE SEGMENT
2  ASSUME CS:CODE
3  BEG:
4    MOV AX, 0
5    ADD AX, 8000H
6    ADD AX, 8000H
7    JMP $
8  CODE ENDS
9    END BEG
```

Listing 4: 加法运算-扩展三

```
1  CODE SEGMENT
2  ASSUME CS:CODE
3  BEG:
4    MOV AX, FFFFH
5    ADD AX, 0001H
6    ADC AX, 0001H
7    JMP $
8  CODE ENDS
9    END BEG
```

Listing 5: 加法运算-扩展四

```
1  CODE SEGMENT
2  ASSUME CS:CODE
3  BEG:
4    MOV AX, FFFFH
5    INC AX
6    JMP $
7  CODE ENDS
8    END BEG
```

Listing 6: 加法运算-扩展五