BBM436: Microprocessors Lab.
2020-2021 Autumn
Homework 2

İbrahim Burak Tanrıkulu, 21827852

November 4, 2020

1 Phase: Creating workspace

- Firstly i used minecraft to create an 74181 ALU and i did half of it. But doing it in minecraft was time consuming. So i decided to use Proteus.
- I downloaded and installed Proteus and will use it.

2 Phase: Running Examples

- I ran examples and did my first 74181 ALU in Proteus 1 . I checked this ALU with random inputs.

3 Phase: Designing ALU and Register

- I decided to do 16 bit ALU and Registers for to use in next projects.
- I designed 16-bit ALU with 74181 ALU 2 and 74182 Lookahead Carry Generator 3 .
  There are 4 piece of 74181 to make 16 bit ALU. Lookahead Carry Generator makes this ALUs faster.
- I designed 16-bit register file with open-collector D Flip-Flops 4 .
  Every register is 16 bit length and there are 16 registers in this register file.
  I used 16 piece of OR gate to multiplex these register BUSes.
- I checked these circuits with pattern generator 5 and logic analyzer 6 .

4 Phase: Sample 8086 compiler

- I used emu8086 application to simulate 8086 microprocessor. Also i installed MASM32 .
- There is a sample addition/substraction code and simulation of this code. 7
  You can run it step by step or completely. While you stepping this commands, You can see assembly codes.

5 Phase: Machine Code Extraction

- I used another addition code to extract machine code. I will use WinHex to view these machine codes.
  sample.asm 8
  sample.obj 9
  sample.exe 10

I can share my Proteus project and Minecraft map.

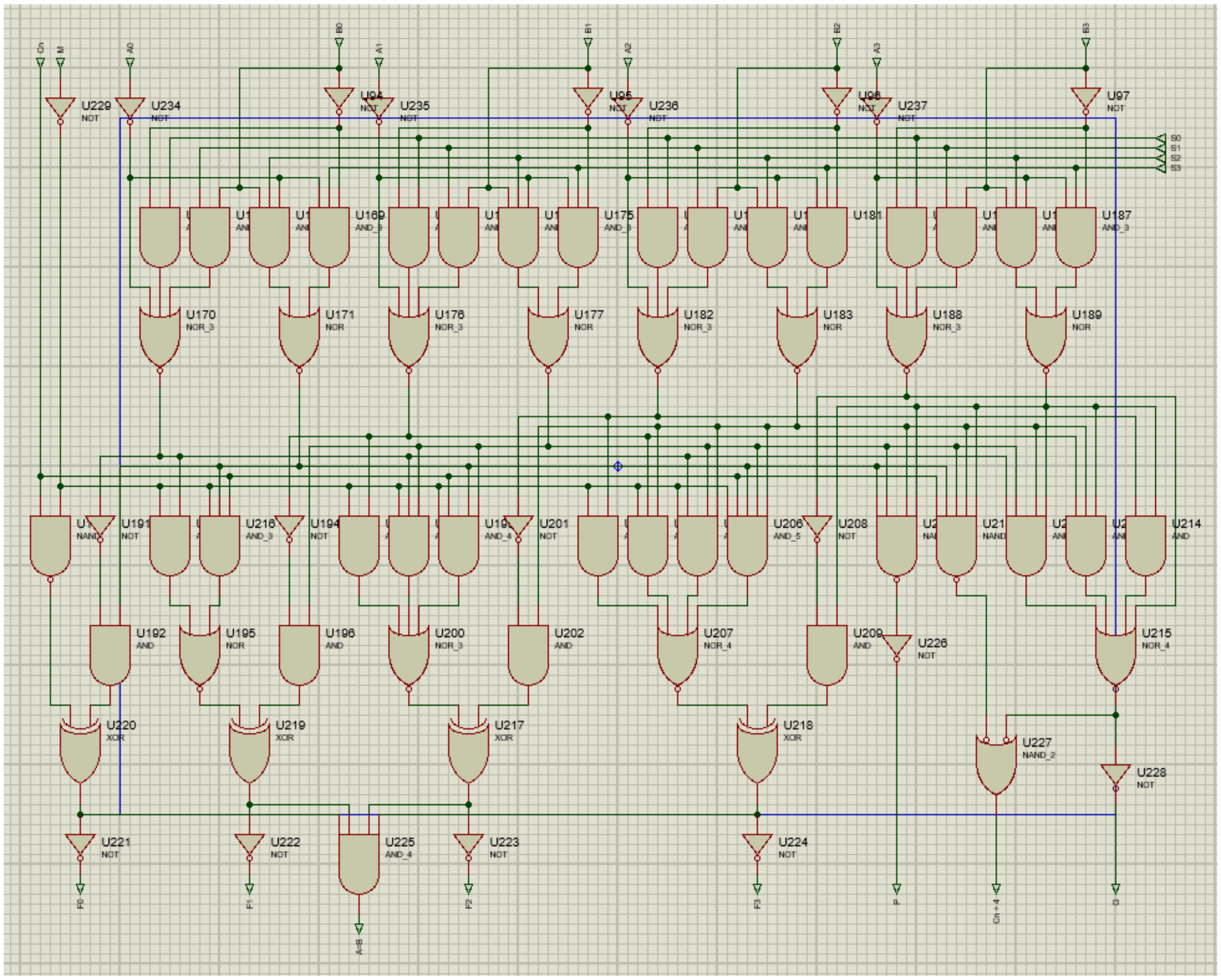


Figure 1: 74181 ALU

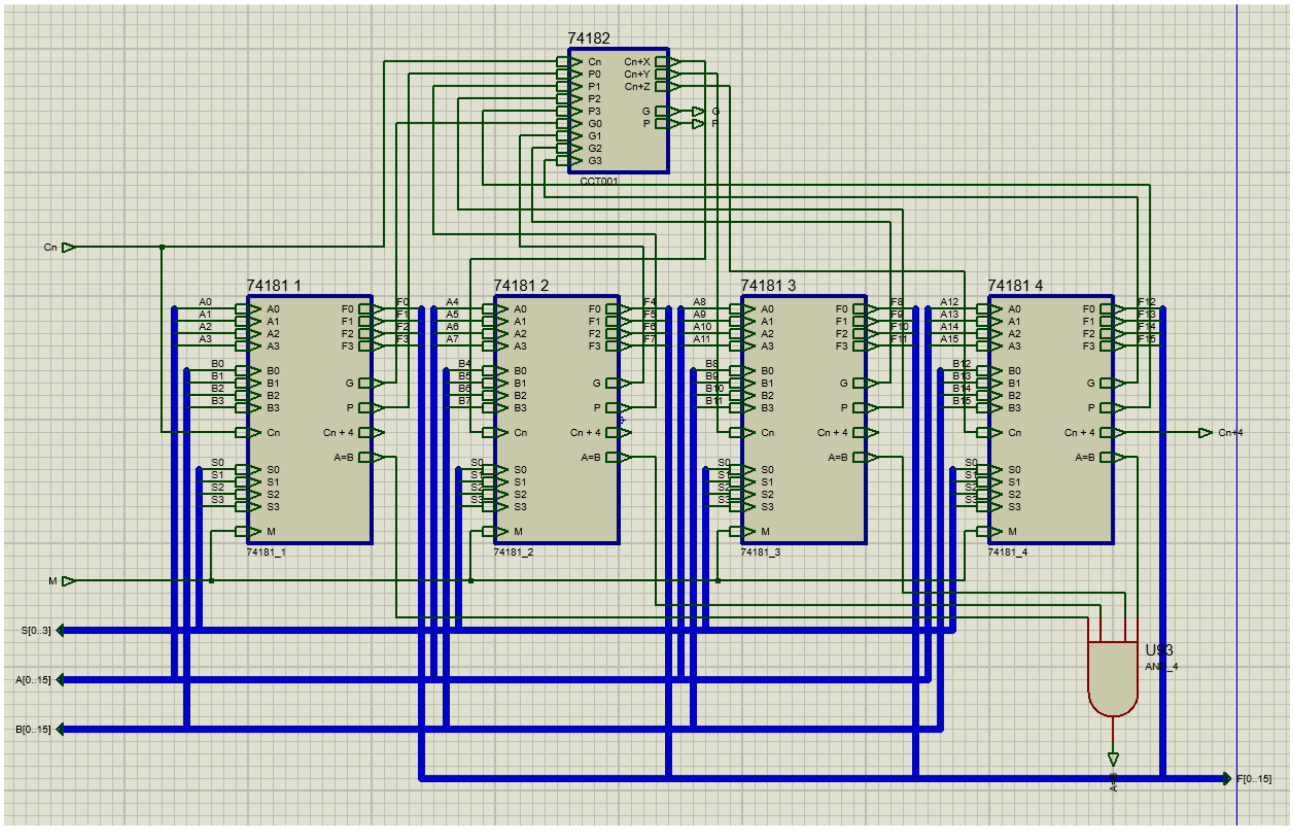


Figure 2: 16-bit ALU

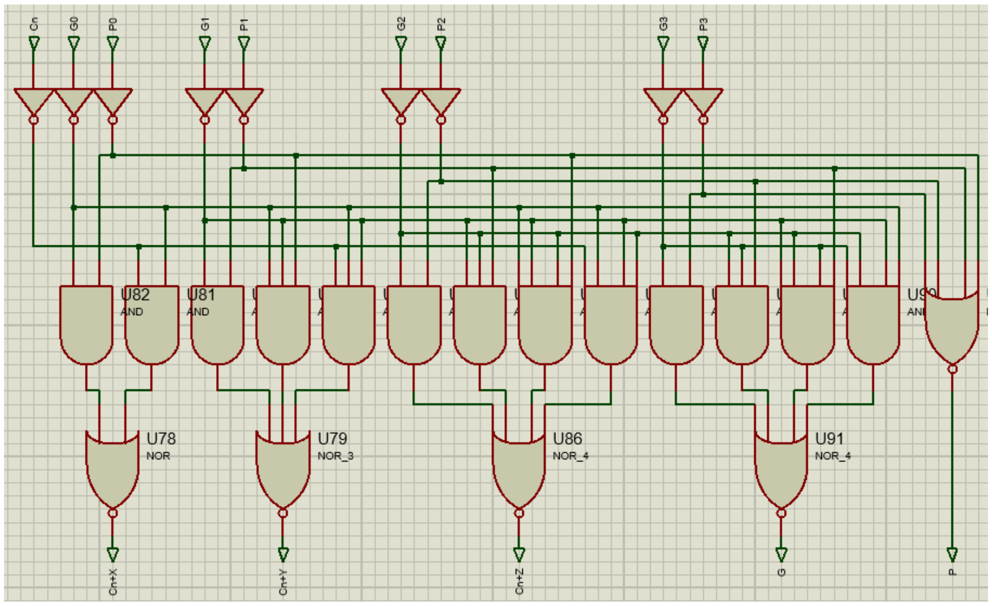


Figure 3: 74182 Lookahead Carry Generator

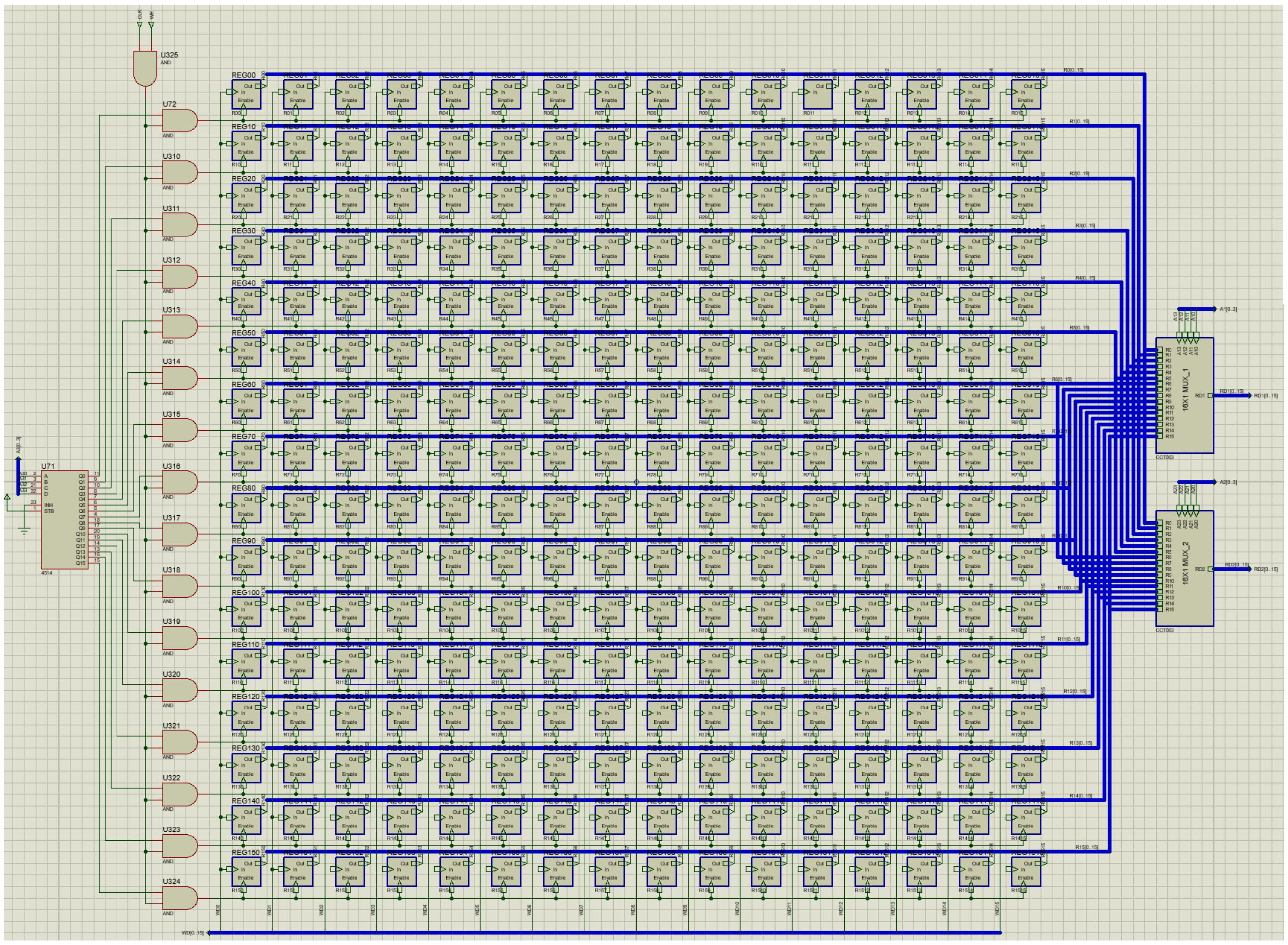


Figure 4: 16-bit Register File

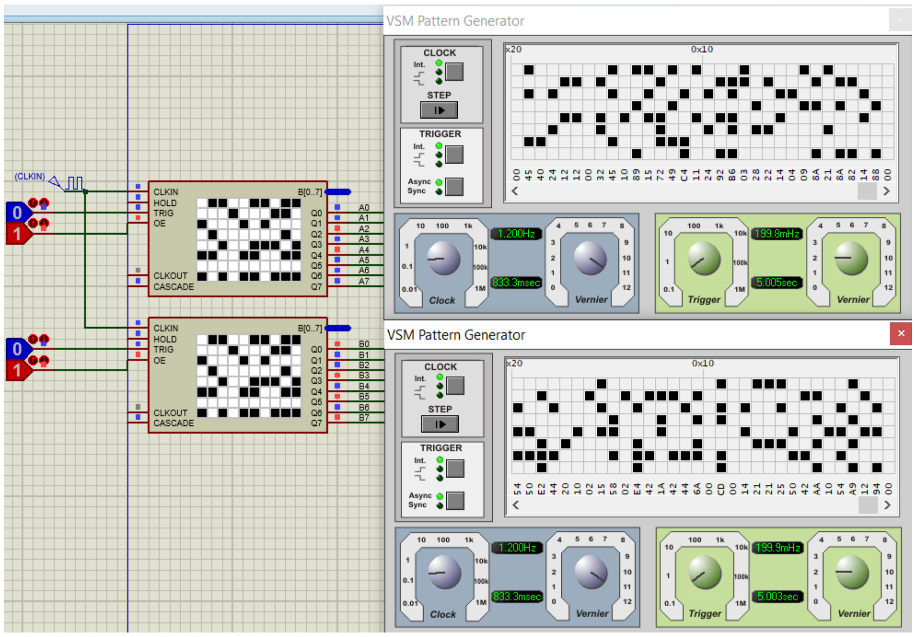


Figure 5: Pattern Generator

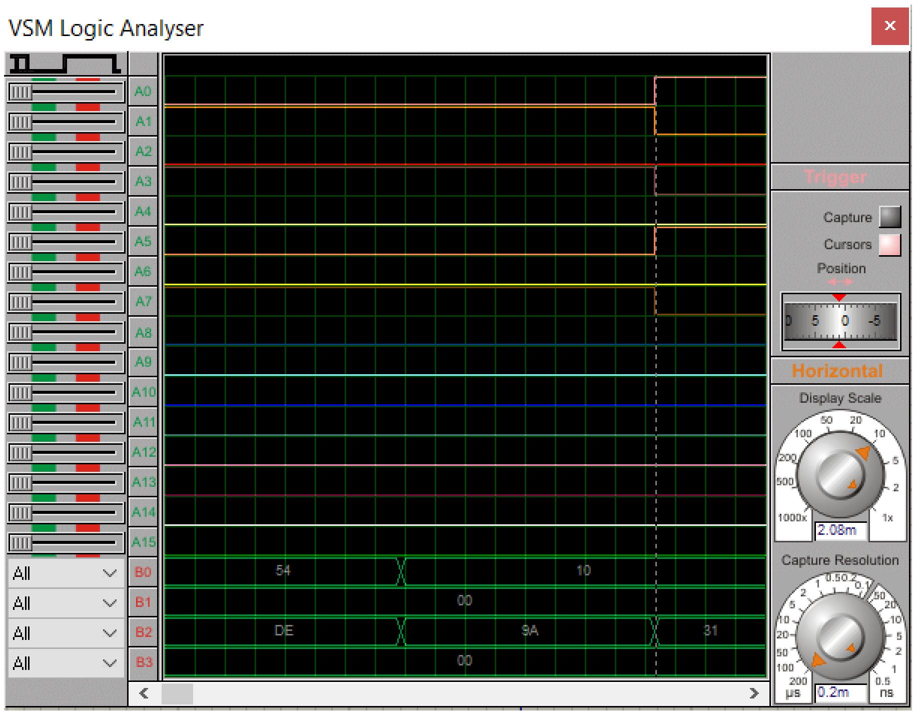
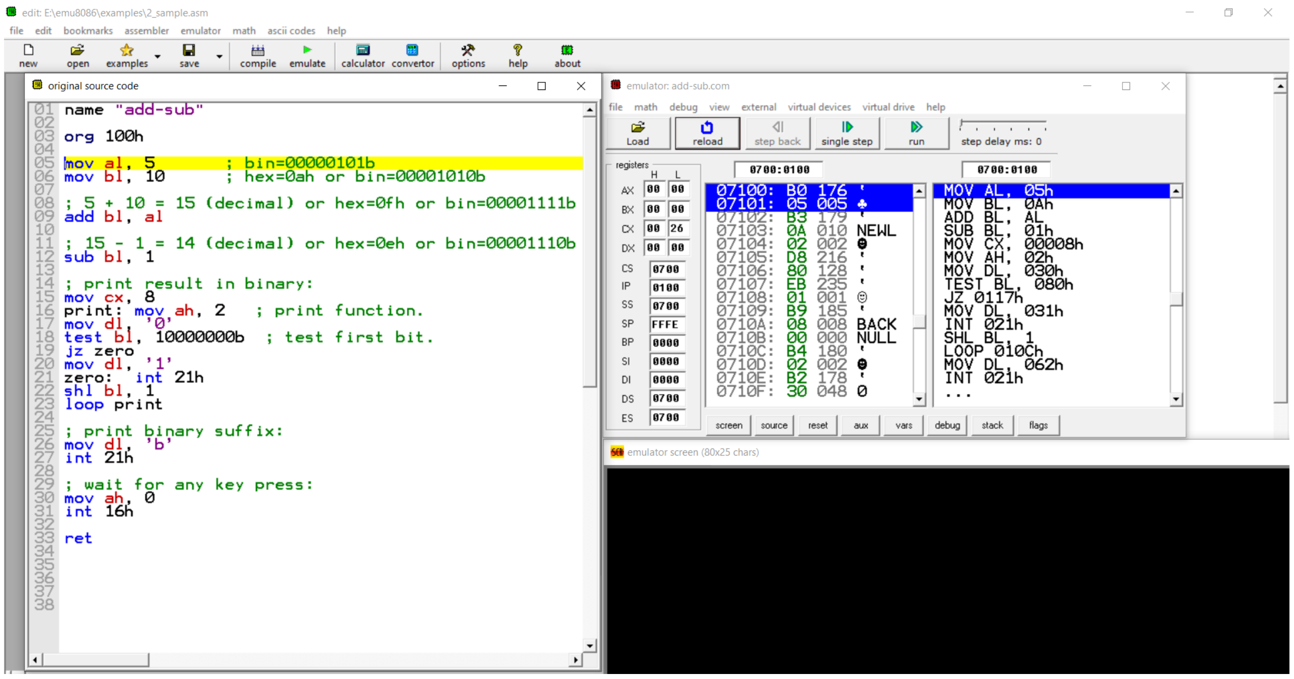


Figure 6: Logic Analyzer

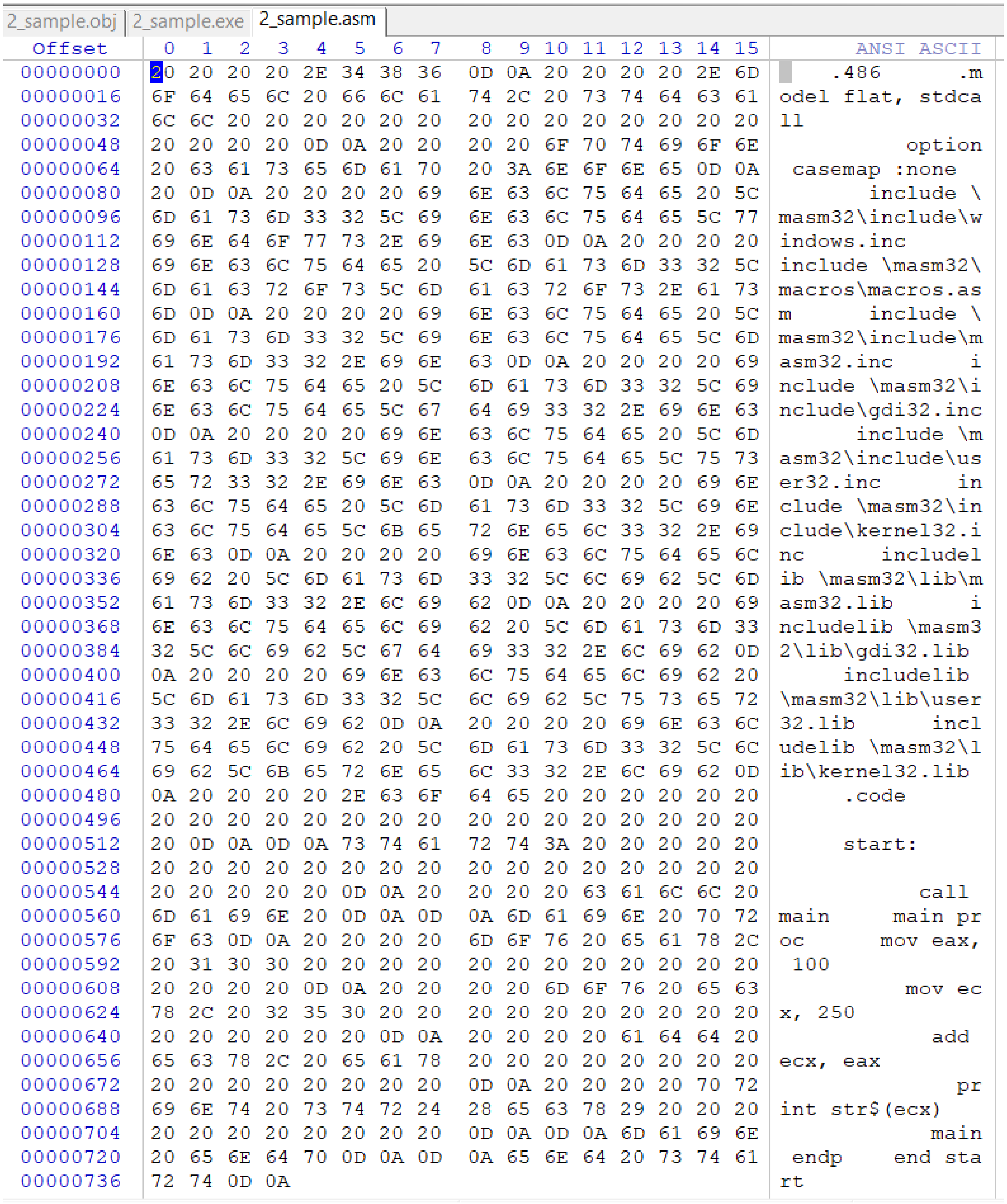

edit: E:\emu8086\examples\2_sample.asm
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
original source code
emulator: add-sub.com
file math debug view external virtual devices virtual drive help
Load reload step back single step run step delay ms: 0
registers
screen source reset aux vars debug stack flags
emulator screen (80x25 chars)

```
name "add-sub"

org 100h

mov al, 5          ; bin=00000101b
mov bl, 10         ; hex=0ah or bin=00001010b

; 5 + 10 = 15 (decimal) or hex=0fh or bin=00001111b
add bl, al

; 15 - 1 = 14 (decimal) or hex=0eh or bin=00001110b
sub bl, 1

; print result in binary:
mov cx, 8
print: mov ah, 2   ; print function.
mov dl, '0'
test bl, 10000000b  ; test first bit.
jz zero
mov dl, '1'
zero:  int 21h
shl bl, 1
loop print

; print binary suffix:
mov dl, 'b'
int 21h

; wait for any key press:
mov ah, 0
int 16h

ret
```

```
     H  L
AX  00 00
BX  00 00
CX  00 26
DX  00 00
CS  0700
IP  0100
SS  0700
SP  FFFE
BP  0000
SI  0000
DI  0000
DS  0700
ES  0700
```

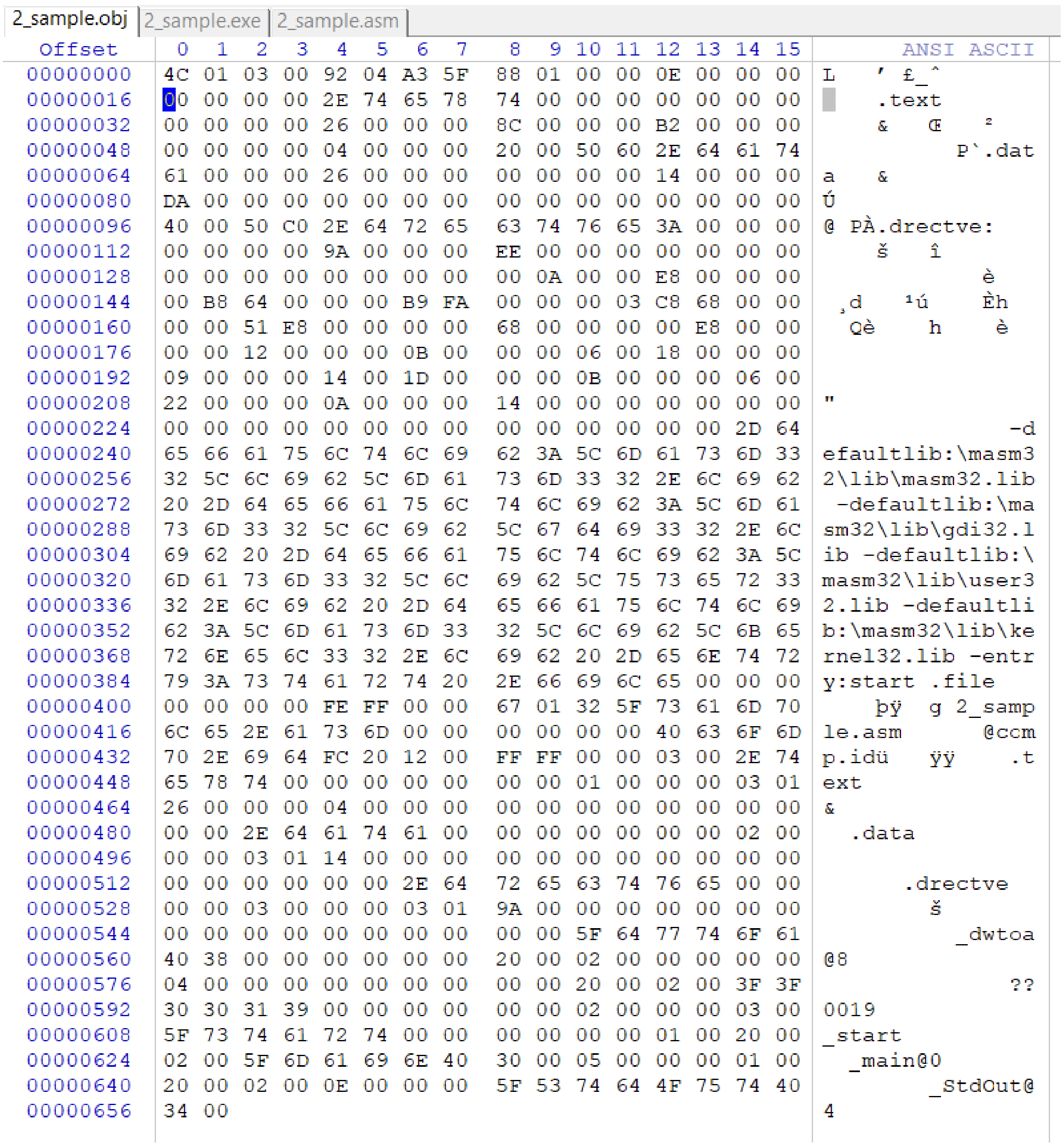
```
0700:0100                      0700:0100
07100: B0 176 '                MOV AL, 05h
07101: 05 005 ♣                MOV BL, 0Ah
07102: B3 179 '                ADD BL, AL
07103: 0A 010 NEWL             SUB BL, 01h
07104: 02 002 ☻                MOV CX, 00008h
07105: D8 216 '                MOV AH, 02h
07106: 80 128 '                MOV DL, 030h
07107: EB 235 '                TEST BL, 080h
07108: 01 001 ☺                JZ 0117h
07109: B9 185 '                MOV DL, 031h
0710A: 08 008 BACK             INT 021h
0710B: 00 000 NULL             SHL BL, 1
0710C: B4 180 '                LOOP 010Ch
0710D: 02 002 ☻                MOV DL, 062h
0710E: B2 178 '                INT 021h
0710F: 30 048 0                ...
```

Figure 7: Sample Program in emu8086



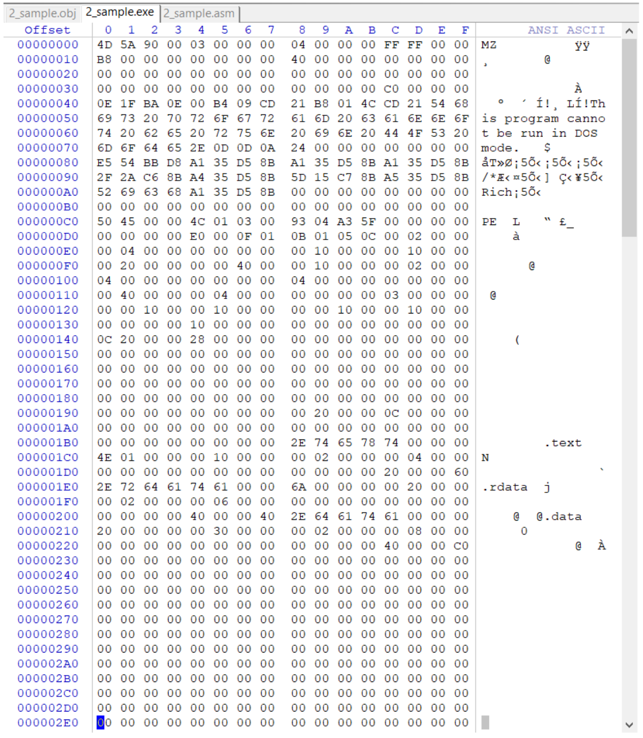
```
  Offset    0  1  2  3  4  5  6  7   8  9 10 11 12 13 14 15        ANSI ASCII
00000000   20 20 20 20 2E 34 38 36  0D 0A 20 20 20 20 2E 6D       .486      .m
00000016   6F 64 65 6C 20 66 6C 61  74 2C 20 73 74 64 63 61   odel flat, stdca
00000032   6C 6C 20 20 20 20 20 20  20 20 20 20 20 20 20 20   ll
00000048   20 20 20 20 0D 0A 20 20  20 20 6F 70 74 69 6F 6E             option
00000064   20 63 61 73 65 6D 61 70  20 3A 6E 6F 6E 65 0D 0A    casemap :none
00000080   20 0D 0A 20 20 20 20 69  6E 63 6C 75 64 65 20 5C          include \
00000096   6D 61 73 6D 33 32 5C 69  6E 63 6C 75 64 65 5C 77   masm32\include\w
00000112   69 6E 64 6F 77 73 2E 69  6E 63 0D 0A 20 20 20 20   indows.inc
00000128   69 6E 63 6C 75 64 65 20  5C 6D 61 73 6D 33 32 5C   include \masm32\
00000144   6D 61 63 72 6F 73 5C 6D  61 63 72 6F 73 2E 61 73   macros\macros.as
00000160   6D 0D 0A 20 20 20 20 69  6E 63 6C 75 64 65 20 5C   m      include \
00000176   6D 61 73 6D 33 32 5C 69  6E 63 6C 75 64 65 5C 6D   masm32\include\m
00000192   61 73 6D 33 32 2E 69 6E  63 0D 0A 20 20 20 20 69   asm32.inc      i
00000208   6E 63 6C 75 64 65 20 5C  6D 61 73 6D 33 32 5C 69   nclude \masm32\i
00000224   6E 63 6C 75 64 65 5C 67  64 69 33 32 2E 69 6E 63   nclude\gdi32.inc
00000240   0D 0A 20 20 20 20 69 6E  63 6C 75 64 65 20 5C 6D         include \m
00000256   61 73 6D 33 32 5C 69 6E  63 6C 75 64 65 5C 75 73   asm32\include\us
00000272   65 72 33 32 2E 69 6E 63  0D 0A 20 20 20 20 69 6E   er32.inc      in
00000288   63 6C 75 64 65 20 5C 6D  61 73 6D 33 32 5C 69 6E   clude \masm32\in
00000304   63 6C 75 64 65 5C 6B 65  72 6E 65 6C 33 32 2E 69   clude\kernel32.i
00000320   6E 63 0D 0A 20 20 20 20  69 6E 63 6C 75 64 65 6C   nc      includel
00000336   69 62 20 5C 6D 61 73 6D  33 32 5C 6C 69 62 5C 6D   ib \masm32\lib\m
00000352   61 73 6D 33 32 2E 6C 69  62 0D 0A 20 20 20 20 69   asm32.lib      i
00000368   6E 63 6C 75 64 65 6C 69  62 20 5C 6D 61 73 6D 33   ncludelib \masm3
00000384   32 5C 6C 69 62 5C 67 64  69 33 32 2E 6C 69 62 0D   2\lib\gdi32.lib
00000400   0A 20 20 20 20 69 6E 63  6C 75 64 65 6C 69 62 20        includelib
00000416   5C 6D 61 73 6D 33 32 5C  6C 69 62 5C 75 73 65 72   \masm32\lib\user
00000432   33 32 2E 6C 69 62 0D 0A  20 20 20 20 69 6E 63 6C   32.lib      incl
00000448   75 64 65 6C 69 62 20 5C  6D 61 73 6D 33 32 5C 6C   udelib \masm32\l
00000464   69 62 5C 6B 65 72 6E 65  6C 33 32 2E 6C 69 62 0D   ib\kernel32.lib
00000480   0A 20 20 20 20 2E 63 6F  64 65 20 20 20 20 20 20        .code
00000496   20 20 20 20 20 20 20 20  20 20 20 20 20 20 20 20
00000512   20 0D 0A 0D 0A 73 74 61  72 74 3A 20 20 20 20 20        start:
00000528   20 20 20 20 20 20 20 20  20 20 20 20 20 20 20 20
00000544   20 20 20 20 20 0D 0A 20  20 20 20 63 61 6C 6C 20              call
00000560   6D 61 69 6E 20 0D 0A 0D  0A 6D 61 69 6E 20 70 72   main     main pr
00000576   6F 63 0D 0A 20 20 20 20  6D 6F 76 20 65 61 78 2C   oc      mov eax,
00000592   20 31 30 30 20 20 20 20  20 20 20 20 20 20 20 20    100
00000608   20 20 20 20 0D 0A 20 20  20 20 6D 6F 76 20 65 63             mov ec
00000624   78 2C 20 32 35 30 20 20  20 20 20 20 20 20 20 20   x, 250
00000640   20 20 20 20 20 20 0D 0A  20 20 20 20 61 64 64 20               add
00000656   65 63 78 2C 20 65 61 78  20 20 20 20 20 20 20 20   ecx, eax
00000672   20 20 20 20 20 20 20 20  0D 0A 20 20 20 20 70 72                 pr
00000688   69 6E 74 20 73 74 72 24  28 65 63 78 29 20 20 20   int str$(ecx)
00000704   20 20 20 20 20 20 20 20  0D 0A 0D 0A 6D 61 69 6E               main
00000720   20 65 6E 64 70 0D 0A 0D  0A 65 6E 64 20 73 74 61    endp    end sta
00000736   72 74 0D 0A                                        rt
```

Figure 8: Assembly Code with WinHex

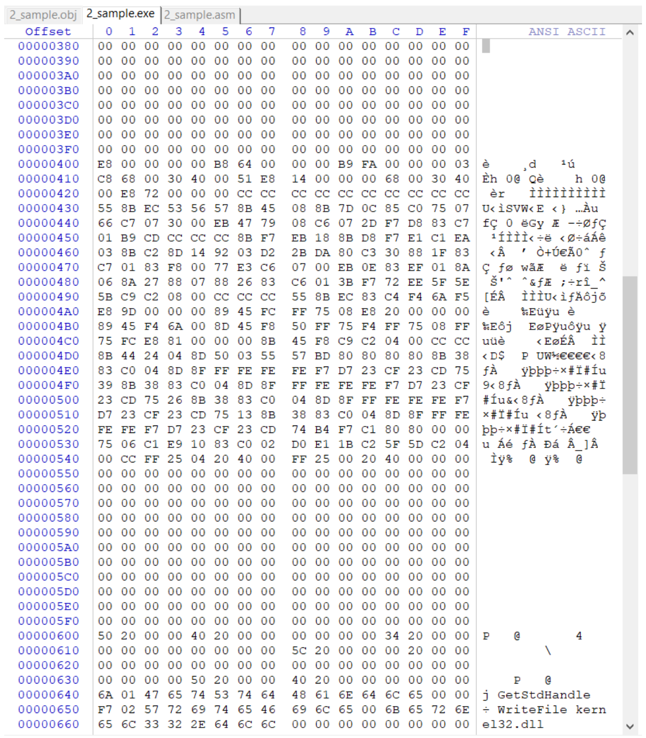


```
2_sample.obj | 2_sample.exe | 2_sample.asm

  Offset    0  1  2  3  4  5  6  7   8  9 10 11 12 13 14 15   ANSI ASCII
00000000   4C 01 03 00 92 04 A3 5F  88 01 00 00 0E 00 00 00   L   ’ £_ˆ
00000016   00 00 00 00 2E 74 65 78  74 00 00 00 00 00 00 00       .text
00000032   00 00 00 00 26 00 00 00  8C 00 00 00 B2 00 00 00       &   Œ   ²
00000048   00 00 00 00 04 00 00 00  20 00 50 60 2E 64 61 74             P`.dat
00000064   61 00 00 00 26 00 00 00  00 00 00 00 14 00 00 00   a   &
00000080   DA 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   Ú
00000096   40 00 50 C0 2E 64 72 65  63 74 76 65 3A 00 00 00   @ PÀ.drectve:
00000112   00 00 00 00 9A 00 00 00  EE 00 00 00 00 00 00 00       š   î
00000128   00 00 00 00 00 00 00 00  00 0A 00 00 E8 00 00 00               è
00000144   00 B8 64 00 00 00 B9 FA  00 00 00 03 C8 68 00 00    ¸d   ¹ú    Èh
00000160   00 00 51 E8 00 00 00 00  68 00 00 00 00 E8 00 00     Qè    h    è
00000176   00 00 12 00 00 00 0B 00  00 00 06 00 18 00 00 00
00000192   09 00 00 00 14 00 1D 00  00 00 0B 00 00 00 06 00
00000208   22 00 00 00 0A 00 00 00  14 00 00 00 00 00 00 00   "
00000224   00 00 00 00 00 00 00 00  00 00 00 00 00 00 2D 64                 -d
00000240   65 66 61 75 6C 74 6C 69  62 3A 5C 6D 61 73 6D 33   efaultlib:\masm3
00000256   32 5C 6C 69 62 5C 6D 61  73 6D 33 32 2E 6C 69 62   2\lib\masm32.lib
00000272   20 2D 64 65 66 61 75 6C  74 6C 69 62 3A 5C 6D 61    -defaultlib:\ma
00000288   73 6D 33 32 5C 6C 69 62  5C 67 64 69 33 32 2E 6C   sm32\lib\gdi32.l
00000304   69 62 20 2D 64 65 66 61  75 6C 74 6C 69 62 3A 5C   ib -defaultlib:\
00000320   6D 61 73 6D 33 32 5C 6C  69 62 5C 75 73 65 72 33   masm32\lib\user3
00000336   32 2E 6C 69 62 20 2D 64  65 66 61 75 6C 74 6C 69   2.lib -defaultli
00000352   62 3A 5C 6D 61 73 6D 33  32 5C 6C 69 62 5C 6B 65   b:\masm32\lib\ke
00000368   72 6E 65 6C 33 32 2E 6C  69 62 20 2D 65 6E 74 72   rnel32.lib -entr
00000384   79 3A 73 74 61 72 74 20  2E 66 69 6C 65 00 00 00   y:start .file
00000400   00 00 00 00 FE FF 00 00  67 01 32 5F 73 61 6D 70       þÿ  g 2_samp
00000416   6C 65 2E 61 73 6D 00 00  00 00 00 00 40 63 6F 6D   le.asm      @com
00000432   70 2E 69 64 FC 20 12 00  FF FF 00 00 03 00 2E 74   p.idü   ÿÿ    .t
00000448   65 78 74 00 00 00 00 00  00 00 01 00 00 00 03 01   ext
00000464   26 00 00 00 04 00 00 00  00 00 00 00 00 00 00 00   &
00000480   00 00 2E 64 61 74 61 00  00 00 00 00 00 00 02 00     .data
00000496   00 00 03 01 14 00 00 00  00 00 00 00 00 00 00 00
00000512   00 00 00 00 00 00 2E 64  72 65 63 74 76 65 00 00         .drectve
00000528   00 00 03 00 00 00 03 01  9A 00 00 00 00 00 00 00           š
00000544   00 00 00 00 00 00 00 00  00 00 5F 64 77 74 6F 61             _dwtoa
00000560   40 38 00 00 00 00 00 00  20 00 02 00 00 00 00 00   @8
00000576   04 00 00 00 00 00 00 00  00 00 20 00 02 00 3F 3F                 ??
00000592   30 30 31 39 00 00 00 00  00 00 02 00 00 00 03 00   0019
00000608   5F 73 74 61 72 74 00 00  00 00 00 00 01 00 20 00   _start
00000624   02 00 5F 6D 61 69 6E 40  30 00 05 00 00 00 01 00     _main@0
00000640   20 00 02 00 0E 00 00 00  5F 53 74 64 4F 75 74 40           _StdOut@
00000656   34 00                                              4
```

Figure 9: Object Code with WinHex



```
 Offset    0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F        ANSI ASCII
00000000  4D 5A 90 00 03 00 00 00  04 00 00 00 FF FF 00 00  MZ          ÿÿ
00000010  B8 00 00 00 00 00 00 00  40 00 00 00 00 00 00 00  ¸       @
00000020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000030  00 00 00 00 00 00 00 00  00 00 00 00 C0 00 00 00              À
00000040  0E 1F BA 0E 00 B4 09 CD  21 B8 01 4C CD 21 54 68    º  ´ Í!¸ LÍ!Th
00000050  69 73 20 70 72 6F 67 72  61 6D 20 63 61 6E 6E 6F  is program canno
00000060  74 20 62 65 20 72 75 6E  20 69 6E 20 44 4F 53 20  t be run in DOS
00000070  6D 6F 64 65 2E 0D 0D 0A  24 00 00 00 00 00 00 00  mode.   $
00000080  E5 54 BB D8 A1 35 D5 8B  A1 35 D5 8B A1 35 D5 8B  åT»Ø¡5Õ‹¡5Õ‹¡5Õ‹
00000090  2F 2A C6 8B A4 35 D5 8B  5D 15 C7 8B A5 35 D5 8B  /*Æ‹¤5Õ‹] Ç‹¥5Õ‹
000000A0  52 69 63 68 A1 35 D5 8B  00 00 00 00 00 00 00 00  Rich¡5Õ‹
000000B0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000000C0  50 45 00 00 4C 01 03 00  93 04 A3 5F 00 00 00 00  PE  L   “ £_
000000D0  00 00 00 00 E0 00 0F 01  0B 01 05 0C 00 02 00 00      à
000000E0  00 04 00 00 00 00 00 00  00 10 00 00 00 10 00 00
000000F0  00 20 00 00 00 00 40 00  00 10 00 00 00 02 00 00        @
00000100  04 00 00 00 00 00 00 00  04 00 00 00 00 00 00 00
00000110  00 40 00 00 00 04 00 00  00 00 00 00 03 00 00 00   @
00000120  00 00 10 00 00 10 00 00  00 00 10 00 00 10 00 00
00000130  00 00 00 00 10 00 00 00  00 00 00 00 00 00 00 00
00000140  0C 20 00 00 28 00 00 00  00 00 00 00 00 00 00 00      (
00000150  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000160  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000170  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000180  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000190  00 00 00 00 00 00 00 00  00 20 00 00 0C 00 00 00
000001A0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000001B0  00 00 00 00 00 00 00 00  2E 74 65 78 74 00 00 00          .text
000001C0  4E 01 00 00 00 10 00 00  00 02 00 00 00 04 00 00  N
000001D0  00 00 00 00 00 00 00 00  00 00 00 00 20 00 00 60                 `
000001E0  2E 72 64 61 74 61 00 00  6A 00 00 00 00 20 00 00  .rdata  j
000001F0  00 02 00 00 00 06 00 00  00 00 00 00 00 00 00 00
00000200  00 00 00 00 40 00 00 40  2E 64 61 74 61 00 00 00      @  @.data
00000210  20 00 00 00 00 30 00 00  00 02 00 00 00 08 00 00       0
00000220  00 00 00 00 00 00 00 00  00 00 00 00 40 00 00 C0              @  À
00000230  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000240  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000250  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000260  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000270  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000280  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000290  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000002A0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000002B0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000002C0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000002D0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000002E0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
```



```
 Offset    0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F        ANSI ASCII
00000380  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000390  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000003A0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000003B0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000003C0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000003D0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000003E0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000003F0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000400  E8 00 00 00 00 B8 64 00  00 00 B9 FA 00 00 00 03  è    ¸d   ¹ú
00000410  C8 68 00 30 40 00 51 E8  14 00 00 00 68 00 30 40  Èh 0@ Qè    h 0@
00000420  00 E8 72 00 00 00 CC CC  CC CC CC CC CC CC CC CC   èr   ÌÌÌÌÌÌÌÌÌÌ
00000430  55 8B EC 53 56 57 8B 45  08 8B 7D 0C 85 C0 75 07  U‹ìSVW‹E ‹} …Àu
00000440  66 C7 07 30 00 EB 47 79  08 C6 07 2D F7 D8 83 C7  fÇ 0 ëGy Æ -÷ØƒÇ
00000450  01 B9 CD CC CC CC 8B F7  EB 18 8B D8 F7 E1 C1 EA   ¹ÍÌÌÌ‹÷ë ‹Ø÷áÁê
00000460  03 8B C2 8D 14 92 03 D2  2B DA 80 C3 30 88 1F 83   ‹Â  ’ Ò+Ú€Ã0ˆ ƒ
00000470  C7 01 83 F8 00 77 E3 C6  07 00 EB 0E 83 EF 01 8A  Ç ƒø wãÆ  ë ƒï Š
00000480  06 8A 27 88 07 88 26 83  C6 01 3B F7 72 EE 5F 5E   Š'ˆ ˆ&ƒÆ ;÷rî_^
00000490  5B C9 C2 08 00 CC CC CC  55 8B EC 83 C4 F4 6A F5  [ÉÂ  ÌÌÌU‹ìƒÄôjõ
000004A0  E8 9D 00 00 00 89 45 FC  FF 75 08 E8 20 00 00 00  è    ‰Eüÿu è
000004B0  89 45 F4 6A 00 8D 45 F8  50 FF 75 F4 FF 75 08 FF  ‰Eôj  EøPÿuôÿu ÿ
000004C0  75 FC E8 81 00 00 00 8B  45 F8 C9 C2 04 00 CC CC  uüè    ‹EøÉÂ  ÌÌ
000004D0  8B 44 24 04 8D 50 03 55  57 BD 80 80 80 80 8B 38  ‹D$  P UW½€€€€‹8
000004E0  83 C0 04 8D 8F FF FE FE  FE F7 D7 23 CF 23 CD 75  ƒÀ   ÿþþþ÷×#Ï#Íu
000004F0  39 8B 38 83 C0 04 8D 8F  FF FE FE FE F7 D7 23 CF  9‹8ƒÀ   ÿþþþ÷×#Ï
00000500  23 CD 75 26 8B 38 83 C0  04 8D 8F FF FE FE FE F7  #Íu&‹8ƒÀ   ÿþþþ÷
00000510  D7 23 CF 23 CD 75 13 8B  38 83 C0 04 8D 8F FF FE  ×#Ï#Íu ‹8ƒÀ   ÿþ
00000520  FE FE F7 D7 23 CF 23 CD  74 B4 F7 C1 80 80 00 00  þþ÷×#Ï#Ít´÷Á€€
00000530  75 06 C1 E9 10 83 C0 02  D0 E1 1B C2 5F 5D C2 04  u Áé ƒÀ Ðá Â_]Â
00000540  00 CC FF 25 04 20 40 00  FF 25 00 20 40 00 00 00   Ìÿ%  @ ÿ%  @
00000550  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000560  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000570  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000580  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000590  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000005A0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000005B0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000005C0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000005D0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000005E0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
000005F0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000600  50 20 00 00 40 20 00 00  00 00 00 00 34 20 00 00  P   @       4
00000610  00 00 00 00 00 00 00 00  5C 20 00 00 00 20 00 00          \
00000620  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00000630  00 00 00 00 50 20 00 00  40 20 00 00 00 00 00 00      P   @
00000640  6A 01 47 65 74 53 74 64  48 61 6E 64 6C 65 00 00  j GetStdHandle
00000650  F7 02 57 72 69 74 65 46  69 6C 65 00 6B 65 72 6E  ÷ WriteFile kern
00000660  65 6C 33 32 2E 64 6C 6C  00 00 00 00 00 00 00 00  el32.dll
```

Figure 10: Executable Code with WinHex