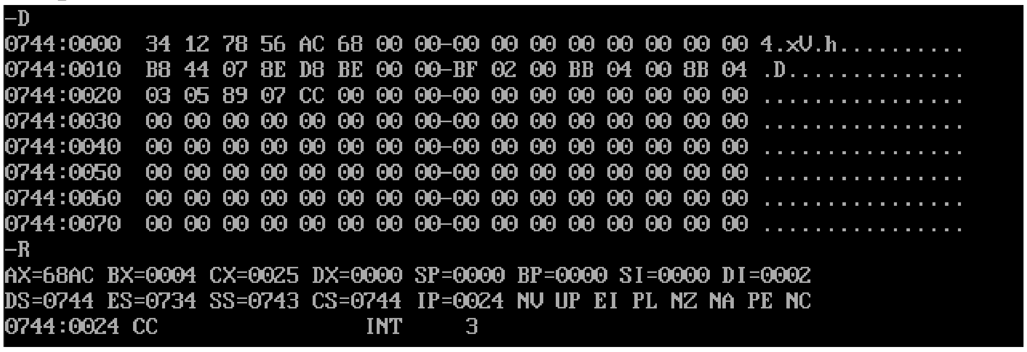
Microprocessor Fundamentals and Programming

Assignment 2 **CE092 : Nevil Parmar**

1) **Add 2 16-bit numbers. The 16-bit numbers are stored into the data segment.**
- **Code:**

```
- data segment
-     n1 dw 1234h
-     n2 dw 5678h
-     result dw ?
- data ends
- code segment
- assume cs:code,ds:data
-     mov ax,data
-     mov ds,ax
-     lea si,n1
-     lea di,n2
-     lea bx,result
-     mov ax,[si]
-     add ax,[di]
-     mov [bx],ax
-     int 03
- code ends
- end
-
```

- **Output:**



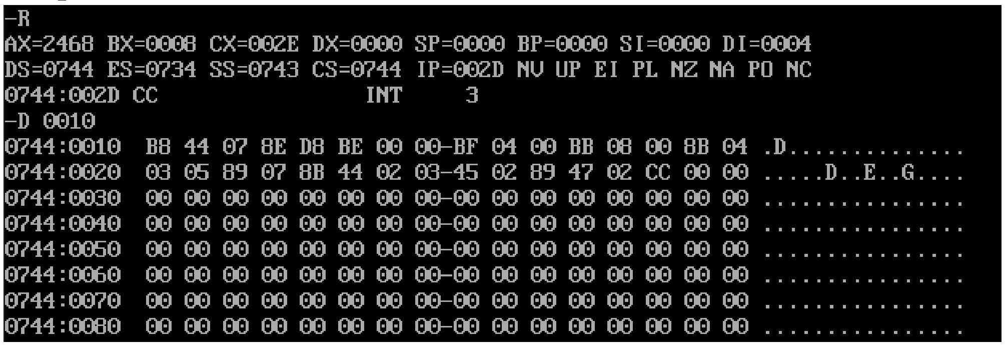
```
-D
0744:0000  34 12 78 56 AC 68 00 00-00 00 00 00 00 00 00 00 4.xV.h..........
0744:0010  B8 44 07 8E D8 BE 00 00-BF 02 00 BB 04 00 8B 04 .D..............
0744:0020  03 05 89 07 CC 00 00 00-00 00 00 00 00 00 00 00 ................
0744:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0744:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0744:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0744:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0744:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
-R
AX=68AC BX=0004 CX=0025 DX=0000 SP=0000 BP=0000 SI=0000 DI=0002
DS=0744 ES=0734 SS=0743 CS=0744 IP=0024 NV UP EI PL NZ NA PE NC
0744:0024 CC            INT     3
```

-

2) **Add 2 32-bit numbers stored in the data segment.**
- **Code:**

```
- data segment
-     n1 dd 12345678h
-       n2 dd 12342345h
-       result dd ?
-   data ends
-   code segment
-   assume cs:code,ds:data
-       mov ax,data
-       mov ds,ax
-       lea si,n1
-       lea di,n2
-       lea bx,result
-       mov ax,[si]
-       add ax,[di]
-       mov [bx],ax
-       mov ax,[si+2]
-       add ax,[di+2]
-       mov [bx+2],ax
-       int 03
-   code ends
-   end
-
```

- **Output:**



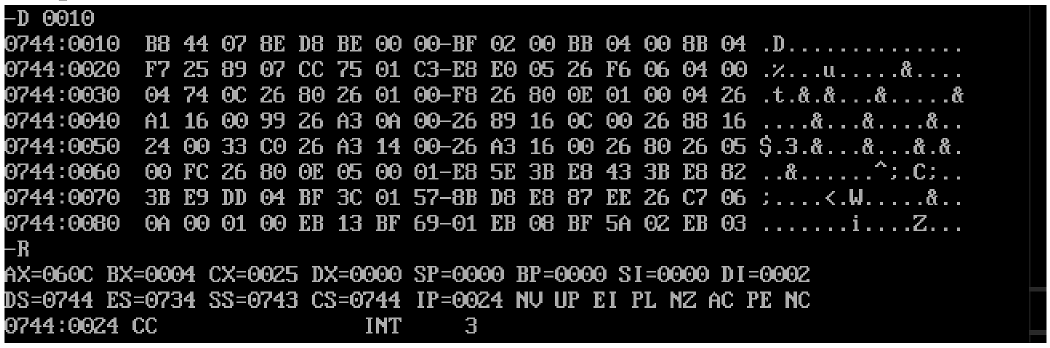
```
-R
AX=2468 BX=0008 CX=002E DX=0000 SP=0000 BP=0000 SI=0000 DI=0004
DS=0744 ES=0734 SS=0743 CS=0744 IP=002D NV UP EI PL NZ NA PO NC
0744:002D CC                   INT     3
-D 0010
0744:0010  B8 44 07 8E D8 BE 00 00-BF 04 00 BB 08 00 8B 04 .D..............
0744:0020  03 05 89 07 8B 44 02 03-45 02 89 47 02 CC 00 00 .....D..E..G....
0744:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0744:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0744:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0744:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0744:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0744:0080  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
```

-

3) **Program to multiply two unsigned 16-bit numbers.**
- **Code:**

```
-   data segment
-       n1 dw 12h
-       n2 dw 56h
-       result dd ?
-   data ends
-   code segment
-       assume cs:code,ds:data
-       mov ax,data
-       mov ds,ax
-       lea si,n1
-       lea di,n2
-       lea bx,result
-       mov ax,[si]
-       mul [di]
-       mov [bx],ax
-       int 03
-   code ends
-   end
-
```

- **Output:**



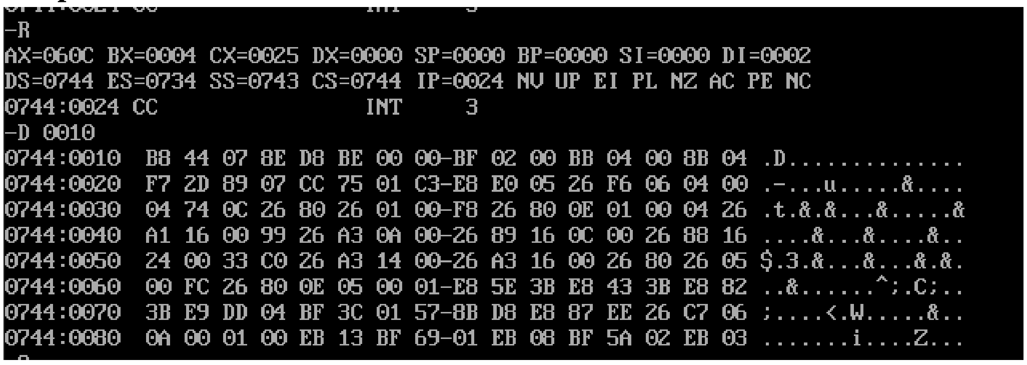
```
-D 0010
0744:0010  B8 44 07 8E D8 BE 00 00-BF 02 00 BB 04 00 8B 04  .D..............
0744:0020  F7 25 89 07 CC 75 01 C3-E8 E0 05 26 F6 06 04 00  .%...u.....&....
0744:0030  04 74 0C 26 80 26 01 00-F8 26 80 0E 01 00 04 26  .t.&.&...&.....&
0744:0040  A1 16 00 99 26 A3 0A 00-26 89 16 0C 00 26 88 16  ....&...&....&..
0744:0050  24 00 33 C0 26 A3 14 00-26 A3 16 00 26 80 26 05  $.3.&...&...&.&.
0744:0060  00 FC 26 80 0E 05 00 01-E8 5E 3B E8 43 3B E8 82  ..&......^;.C;..
0744:0070  3B E9 DD 04 BF 3C 01 57-8B D8 E8 87 EE 26 C7 06  ;....<.W.....&..
0744:0080  0A 00 01 00 EB 13 BF 69-01 EB 08 BF 5A 02 EB 03  .......i....Z...
-R
AX=060C BX=0004 CX=0025 DX=0000 SP=0000 BP=0000 SI=0000 DI=0002
DS=0744 ES=0734 SS=0743 CS=0744 IP=0024 NV UP EI PL NZ AC PE NC
0744:0024 CC                    INT     3
```

4) Program to multiply signed 16-bit numbers.

- **Code:**

```
-   data segment
-       n1 dw -12h
-       n2 dw -56h
-       result dd ?
-   data ends
-   code segment
-       assume cs:code,ds:data
-       mov ax,data
-       mov ds,ax
-       lea si,n1
-       lea di,n2
-       lea bx,result
-       mov ax,[si]
-       imul [di]
-       mov [bx],ax
-       int 03
-   code ends
-   end
-
```

- **Output:**



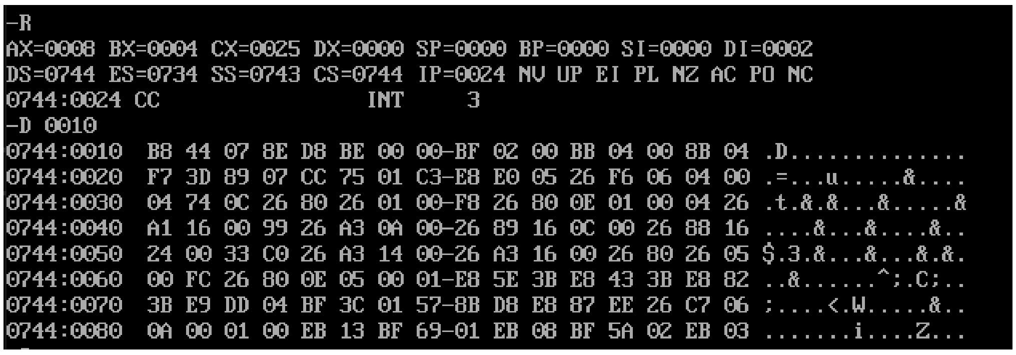
```
-R
AX=060C BX=0004 CX=0025 DX=0000 SP=0000 BP=0000 SI=0000 DI=0002
DS=0744 ES=0734 SS=0743 CS=0744 IP=0024 NV UP EI PL NZ AC PE NC
0744:0024 CC            INT     3
-D 0010
0744:0010  B8 44 07 8E D8 BE 00 00-BF 02 00 BB 04 00 8B 04  .D..............
0744:0020  F7 2D 89 07 CC 75 01 C3-E8 E0 05 26 F6 06 04 00  .-...u.....&....
0744:0030  04 74 0C 26 80 26 01 00-F8 26 80 0E 01 00 04 26  .t.&.&...&.....&
0744:0040  A1 16 00 99 26 A3 0A 00-26 89 16 0C 00 26 88 16  ....&...&....&..
0744:0050  24 00 33 C0 26 A3 14 00-26 A3 16 00 26 80 26 05  $.3.&...&...&.&.
0744:0060  00 FC 26 80 0E 05 00 01-E8 5E 3B E8 43 3B E8 82  ..&......^;.C;..
0744:0070  3B E9 DD 04 BF 3C 01 57-8B D8 E8 87 EE 26 C7 06  ;....<.W.....&..
0744:0080  0A 00 01 00 EB 13 BF 69-01 EB 08 BF 5A 02 EB 03  .......i....Z...
```

5) **Program to divide a 16-bit unsigned/signed number by 16-bit number. The numbers are stored into the data segment.**

- **Code:**

```
-   data segment
-       n1 dw 20h
-       n2 dw 4h
-       result dw ?
-   data ends
-   code segment
-       assume cs:code,ds:data
-       mov ax,data
-       mov ds,ax
-       lea si,n1
-       lea di,n2
-       lea bx,result
-       mov ax,[si]
-       idiv [di]
-       mov [bx],ax
-       int 03
-   code ends
-   end
-
```

- **Output:**



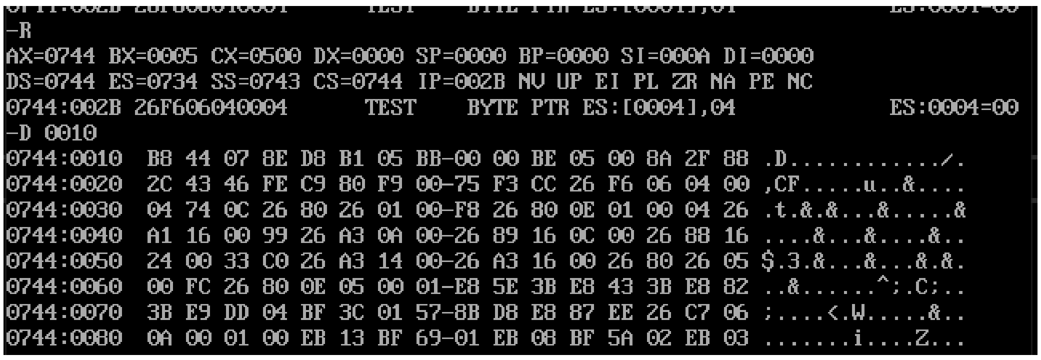
```
-R
AX=0008 BX=0004 CX=0025 DX=0000 SP=0000 BP=0000 SI=0000 DI=0002
DS=0744 ES=0734 SS=0743 CS=0744 IP=0024 NV UP EI PL NZ AC PO NC
0744:0024 CC                INT     3
-D 0010
0744:0010  B8 44 07 8E D8 BE 00 00-BF 02 00 BB 04 00 8B 04 .D..............
0744:0020  F7 3D 89 07 CC 75 01 C3-E8 E0 05 26 F6 06 04 00 .=...u.....&....
0744:0030  04 74 0C 26 80 26 01 00-F8 26 80 0E 01 00 04 26 .t.&.&...&.....&
0744:0040  A1 16 00 99 26 A3 0A 00-26 89 16 0C 00 26 88 16 ....&...&....&..
0744:0050  24 00 33 C0 26 A3 14 00-26 A3 16 00 26 80 26 05 $.3.&...&...&.&.
0744:0060  00 FC 26 80 0E 05 00 01-E8 5E 3B E8 43 3B E8 82 ..&......^;.C;..
0744:0070  3B E9 DD 04 BF 3C 01 57-8B D8 E8 87 EE 26 C7 06 ;....<.W.....&..
0744:0080  0A 00 01 00 EB 13 BF 69-01 EB 08 BF 5A 02 EB 03 .......i....Z...
```

6) **Program to copy an array of bytes/words from the variable "SOURCE" to variable "DEST" which are defined in data segment.**

- **Code:**

```
-  data segment
-      arr1 db 1,2,3,4,5
-      arr2 db 0,0,0,0
-  data ends
-  code segment
-      assume cs:code,ds:data
-      MOV AX,DATA
-      MOV DS,AX
-      MOV CL,5
-      LEA BX,arr1
-      LEA SI,arr2
-  L:  MOV CH,[BX]
-      MOV [SI],CH
-      INC BX
-      INC SI
-      DEC CL
-      CMP CL,00
-      JNZ L
-      INT 3
-  code ends
-  end
-
```

- **Output:**



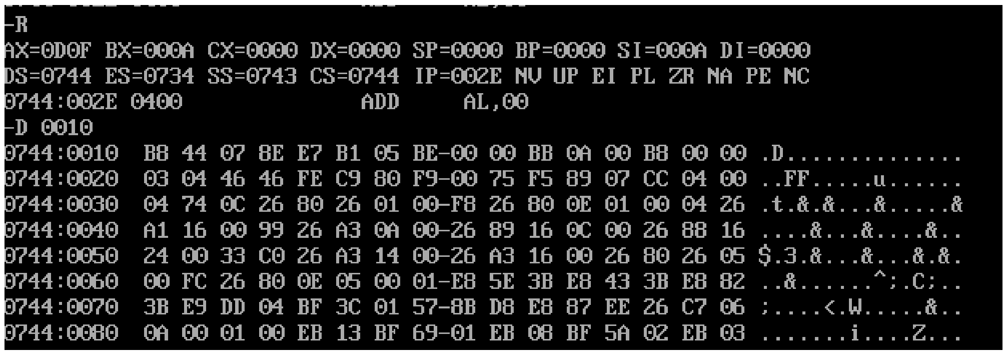
```
-R
AX=0744 BX=0005 CX=0500 DX=0000 SP=0000 BP=0000 SI=000A DI=0000
DS=0744 ES=0734 SS=0743 CS=0744 IP=002B NV UP EI PL ZR NA PE NC
0744:002B 26F606040004      TEST    BYTE PTR ES:[0004],04          ES:0004=00
-D 0010
0744:0010  B8 44 07 8E D8 B1 05 BB-00 00 BE 05 00 8A 2F 88 .D............/.
0744:0020  2C 43 46 FE C9 80 F9 00-75 F3 CC 26 F6 06 04 00 ,CF.....u..&....
0744:0030  04 74 0C 26 80 26 01 00-F8 26 80 0E 01 00 04 26 .t.&.&...&.....&
0744:0040  A1 16 00 99 26 A3 0A 00-26 89 16 0C 00 26 88 16 ....&...&....&..
0744:0050  24 00 33 C0 26 A3 14 00-26 A3 16 00 26 80 26 05 $.3.&...&...&.&.
0744:0060  00 FC 26 80 0E 05 00 01-E8 5E 3B E8 43 3B E8 82 ..&......^;.C;..
0744:0070  3B E9 DD 04 BF 3C 01 57-8B D8 E8 87 EE 26 C7 06 ;....<.W.....&..
0744:0080  0A 00 01 00 EB 13 BF 69-01 EB 08 BF 5A 02 EB 03 .......i....Z...
```

7) **To sum an array of numbers stored in the data segment.**

- **Code:**

```
- data segment
-     arr1 dw 0201h,0202h,0203h,0404h,0305h
-     result dw ?
- data ends
- code segment
-     assume cs:code,ds:data
-     MOV AX,DATA
-     MOV DS,AX
-     MOV CL,5
-     LEA SI,arr1
-     LEA BX,result
-     MOV AX,0000h
- L:  ADD AX,[SI]
-     INC SI
-     INC SI
-     DEC CL
-     CMP CL,00
-     JNZ L
-     MOV [BX],AX
-     INT 3
- code ends
- end
-
```

- **Output:**



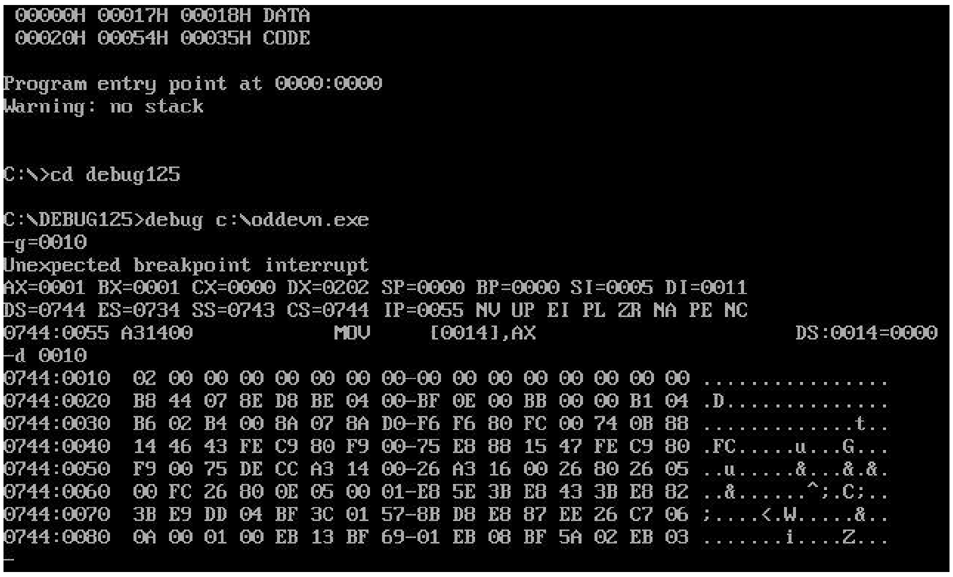
```
-R
AX=0D0F  BX=000A  CX=0000  DX=0000  SP=0000  BP=0000  SI=000A  DI=0000
DS=0744  ES=0734  SS=0743  CS=0744  IP=002E   NV UP EI PL ZR NA PE NC
0744:002E 0400          ADD     AL,00
-D 0010
0744:0010  B8 44 07 8E E7 B1 05 BE-00 00 BB 0A 00 B8 00 00   .D..............
0744:0020  03 04 46 46 FE C9 80 F9-00 75 F5 89 07 CC 04 00   ..FF.....u......
0744:0030  04 74 0C 26 80 26 01 00-F8 26 80 0E 01 00 04 26   .t.&.&...&.....&
0744:0040  A1 16 00 99 26 A3 0A 00-26 89 16 0C 00 26 88 16   ....&...&....&..
0744:0050  24 00 33 C0 26 A3 14 00-26 A3 16 00 26 80 26 05   $.3.&...&...&.&.
0744:0060  00 FC 26 80 0E 05 00 01-E8 5E 3B E8 43 3B E8 82   ..&......^;.C;..
0744:0070  3B E9 DD 04 BF 3C 01 57-8B D8 E8 87 EE 26 C7 06   ;....<.W.....&..
0744:0080  0A 00 01 00 EB 13 BF 69-01 EB 08 BF 5A 02 EB 03   .......i....Z...
```

8) Program to separate even and odd numbers from an array of words.

- **Code:**

```
-  data segment
-      arr db 1,2,3,4
-      oddarr db 10 dup(?)
-      evenarr db 10 dup(?)
-  data ends
-  code segment assume cs:code,ds:data
-      mov ax,data
-      mov ds,ax
-      lea si,oddarr
-      lea di,evenarr
-      lea bx,arr
-      mov cl,4
-      mov dh,2
-  find:
-      mov ah,0000h
-      mov al,[bx]
-      mov dl,al
-      div dh
-      cmp ah,00
-      je even
-      mov [si],dl
-      inc si
-      inc bx
-      dec cl
-      cmp cl,00
-      jnz find
-  even:
-      mov [di],dl
-      inc di
-       dec cl
-       cmp cl,00
-       jnz find
-       int 03
-   code ends
-   end
```

- **Output:**



```
 00000H 00017H 00018H DATA
 00020H 00054H 00035H CODE

Program entry point at 0000:0000
Warning: no stack


C:\>cd debug125

C:\DEBUG125>debug c:\oddevn.exe
-g=0010
Unexpected breakpoint interrupt
AX=0001 BX=0001 CX=0000 DX=0202 SP=0000 BP=0000 SI=0005 DI=0011
DS=0744 ES=0734 SS=0743 CS=0744 IP=0055 NV UP EI PL ZR NA PE NC
0744:0055 A31400             MOV     [0014],AX                       DS:0014=0000
-d 0010
0744:0010  02 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0744:0020  B8 44 07 8E D8 BE 04 00-BF 0E 00 BB 00 00 B1 04 .D..............
0744:0030  B6 02 B4 00 8A 07 8A D0-F6 F6 80 FC 00 74 0B 88 .............t..
0744:0040  14 46 43 FE C9 80 F9 00-75 E8 88 15 47 FE C9 80 .FC.....u...G...
0744:0050  F9 00 75 DE CC A3 14 00-26 A3 16 00 26 80 26 05 ..u.....&...&.&.
0744:0060  00 FC 26 80 0E 05 00 01-E8 5E 3B E8 43 3B E8 82 ..&......^;.C;..
0744:0070  3B E9 DD 04 BF 3C 01 57-8B D8 E8 87 EE 26 C7 06 ;....<.W.....&..
0744:0080  0A 00 01 00 EB 13 BF 69-01 EB 08 BF 5A 02 EB 03 .......i....Z...
-
```