Listing 2.

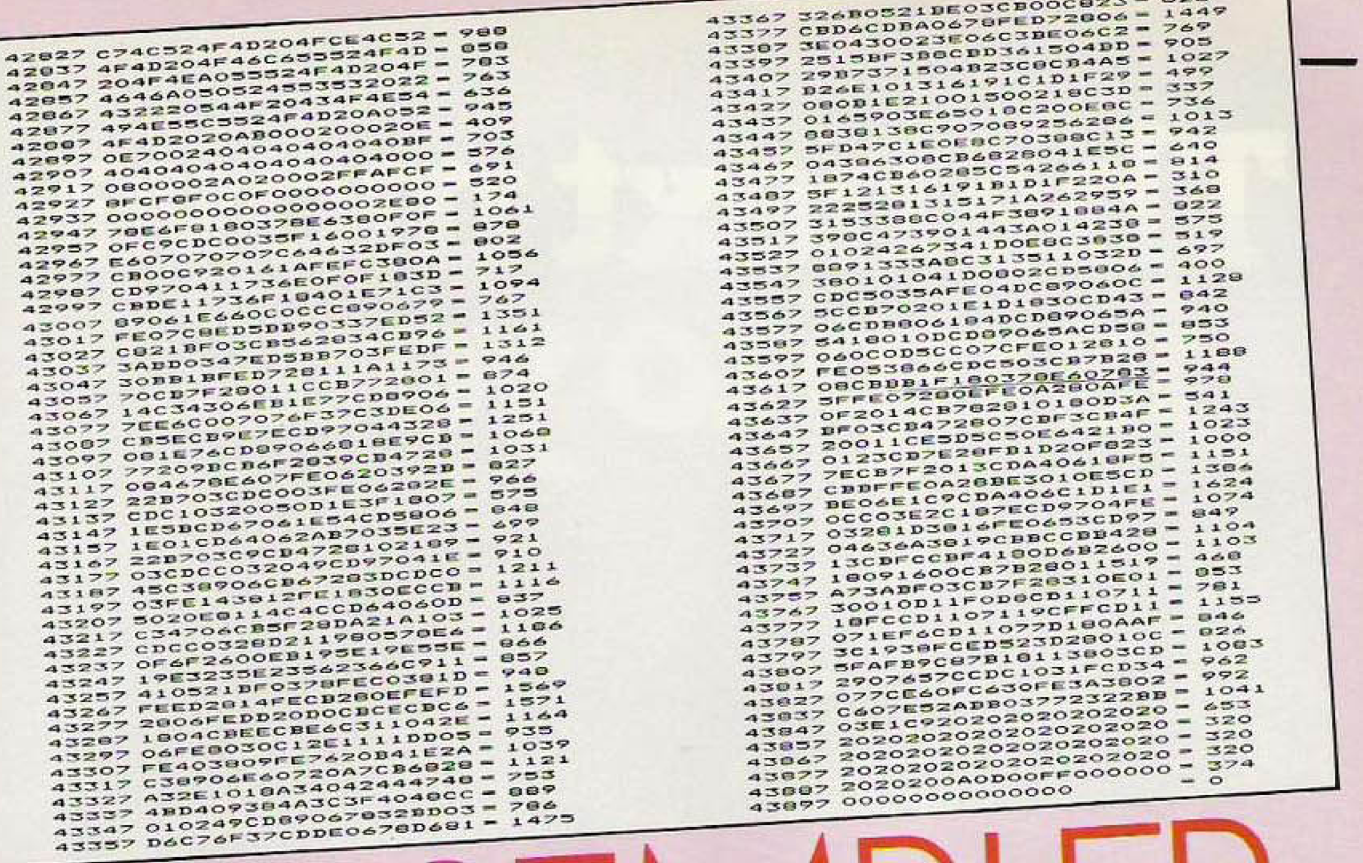
```
41767 43073E072F072B072607 = 292
41777 0C0707070107FB06EC06 = 540
41787 CB06BF06B306AE06A006 = 937
41797 8F067C06590649064506 = 528
41807 3D063806350629062306 = 276
41817 20067F05710568056505 = 503
41827 5F055805540533052605 = 381
41837 1B05F904F604DE04D904 = 982
41847 D604CF04CB04B904B204 = 1007
41857 AD04A804A5049D049804 = 835
41867 950490048B047F047804 = 699
41877 75045A0452044A044004 = 447
41887 3A0420041B0412040B04 = 166
41897 0304FC03E903DC03CD03 = 929
41907 8D017C0170016B016701 = 592
41917 62015F01520143013C01 = 407
41927 33012901250120011B01 = 193
41937 1801150112010B010801 = 87
41947 05010201FD00FA00F700 = 759
41957 F400ED00D800D400BC00 = 1097
41967 8A006C0064005B003100 = 486
41977 2C002900240021000000 = 154
41987 2124FF194E23462378B1 = 864
41997 2814E56B62094E2346E5 = 915
42007 6B62094D44E1702B71E1 = 1077
42017 18E401290021B103C3D1 = 911
42027 BCA401C34500060E1196 = 804
42037 01CD78BB252004050513 = 615
42047 131ACD5ABB1310F9C9DD = 1233
42057 E5E1FE0238E20E002807 = 1053
42067 FE0320DA4E23233EC0A1 = 1070
42077 32BF03CBB95E2356ED53 = 1167
42087 B903235E2356ED53B503 = 942
42097 CB412017CD11BCD83E07 = 1018
42107 CDB4BB47C52100001118 = 914
42117 27CD66BBC11813217007 = 921
42127 237EFEFF280A064BCD2B = 1049
42137 BD38F310F9C9C5CD06B9 = 1547
42147 57CB49CC09B9CD12B95F = 1264
42157 CB51280ACB590E012801 = 682
42167 0DCD0FB9C1D5C5CDDE00 = 1448
42177 C1CB41200CCD78BB78E5 = 1366
42187 CDB4BBE1CD75BBC1C318 = 1718
42197 B92AB503ED4BB703A7ED = 1313
42207 42C90640CB4120070618 = 674
42217 C5CD6CBBC1C521BF037E = 1440
42227 E6CC772AB50322B70321 = 1032
42237 5E0722BB030EFECD1104 = 819
42247 21470722BB03CDD30028 = 791
42257 2309ED43B50322B703CD = 957
42267 EA062ABB03232322BB03 = 766
42277 E5AFCDBE06D1CDD30028 = 1470
42287 05EB232318EB2AB9030B = 810
42297 ED42C130053ABF03E40C = 1043
42307 282DCD6F010428A30510 = 630
42317 A0CB41280F3E07CD9001 = 902
42327 CD18BBCBAFFE432881C9 = 1485
42337 21500722BB031E75CD89 = 833
42347 060DCD6F0118E5214707 = 700
42357 16287E3620CB412805CD = 792
42367 90011803CD5ABB231520 = 742
42377 EDCB41C87E23CD90017E = 1342
42387 CD2BBD30FBC90D0A5379 = 1164
42397 6E746178206572726F72 = 1029
42407 444953415353454D424C = 743
42417 45D20080C2C3C4C5C8CC = 1593
42427 28484CA9C1C148CCC142 = 1278
42437 C344C58A53D041C62842 = 1258
42447 43A9284445A92804A928 = 835
42457 04A94EDADA4EC3C350CF = 1442
42467 50C5D0CD41444420A041 = 1148
42477 444320A053554220A053 = 836
42487 424320A0414E4420A058 = 816
42497 4F5220A04F522020A043 = 805
42507 502020A052455420A045 = 800
42517 58D84A50202020280AA9 = 773
42527 4C4420202053502C8A4A = 659
42537 502020208448414CD44F = 812
42547 555420202802292CC149 = 626
42557 4E202020412C2802A945 = 563
42567 58202020285350292C8A = 610
42577 455820202044452C48CC = 710
42587 44C945C94E4FD0455820 = 1093
42597 202041462C4146A2444A = 682
42607 4E5A20834A5220202083 = 714
42617 4A522020A04A502020A0 = 758
42627 43414C4CA04C442020A0 = 812
42637 494E4320A044454320A0 = 806
42647 504F5020A050555348A0 = 911
42657 494E2020A04F555420A0 = 815
42667 524C4320A052524320A0 = 840
42677 524C2020A052522020A0 = 770
42687 534C4120A053524120A0 = 838
42697 494E56414C494420434F = 697
42707 44C553524C20A0284958 = 899
42717 2B02A92849592B02A952 = 712
42727 535420A049D849D94CC4 = 1210
42737 43D049CE4FD4C9C449D2 = 1525
42747 44D2B0B1B2B3B4B5B6B7 = 1714
42757 42495420A052455320A0 = 841
42767 53455420A0494D2020A0 = 802
42777 5252C4524CC4CEC9D228 = 1371
42787 43A9444154C1524C43C1 = 1064
42797 525243C1524CC15252C1 = 1132
42807 4441C14350CC5343C643 = 1092
42817 43C643414C4C20844E45 = 860
```

AMSTRAD D

Listing 1.

```
10  MEMORY 41766:LOAD "!CODE",41767:GOTO 110
20  SAVE "HEX LOADER":SAVE "CODE",b,41767,2136:END
100 MEMORY 41766
110 MODE 1
120 DIM check(2)
130 LOCATE 1,24:INPUT;"START ADDRESS ";address:CLS
140 LOCATE 1,24:PRINT address
150 INPUT ;"CODE,CHECKSUM ",a$,check(2)
160 check(1)=0
170 FOR n=1 TO LEN(a$) STEP 2
180 a=VAL("&"+MID$(a$,n,2))
190 check(1)=check(1)+a
200 POKE address+n-1,a
210 NEXT
220 IF check(1)=check(2) THEN 240
230 LOCATE 1,24:PRINT "CHECKSUM ERROR.PLEASE INPUT AGAIN":GOTO 260
240 address=address+10
250 LOCATE 10,24:PRINT UPPER$(a$);TAB(32)"= ";check(2)
260 LOCATE 1,25:PRINT SPACE$(41)
270 GOTO 140
```

Kevin Probert with a more subtle way of disassembling than the picture shows.



```
42827 C74C524F4D204FCE4C52 = 988
42837 4F4D204F46C655524F4D = 858
42847 204F4EA055524F4D204F = 783
42857 4646A050524553532022 = 763
42867 432220544F20434F4E54 = 636
42877 494E55C5524F4D20A052 = 945
42887 4F4D2020AB000200020E = 409
42897 0E7002404040404040BF = 703
42907 40404040404040404000 = 576
42917 0800002A020002FFAFCF = 691
42927 8FCF8F0C0F0000000000 = 520
42937 00000000000000002E80 = 174
42947 78E6F8180378E6380F0F = 1061
42957 0FC9CDC0035F16001978 = 878
42967 E607070707C64632DF03 = 802
42977 CB00C920161AFEFC380A = 1056
42987 CD970411736E0F0F183D = 717
42997 CBDE11736F18401E71C3 = 1094
43007 89061E660C0CCC890679 = 767
43017 FE07C8ED5DB80337ED52 = 1351
43027 C821BF03CB562834CB96 = 1161
43037 3ABD0347ED5BB703FEDF = 1312
43047 30BB1BFED728111A1173 = 946
43057 70CB7F28011CCB772801 = 874
43067 14C34306EB1E77CD8906 = 1020
43077 7EE6C007076F37C3DE06 = 1151
43087 CB5ECB9E7ECD97044328 = 1251
43097 081E76CD89066818E9CB = 1068
43107 77209BCB6F2839CB4728 = 1031
43117 084678E607FE0620392B = 827
43127 22B703CDC003FE06282E = 966
43137 CDC10320050D1E3F1807 = 575
43147 1E5BCD67061E54CD5806 = 848
43157 1E01CD64062AB7035E23 = 699
43167 22B703C9CB4728102189 = 921
43177 03CDCC032049CD97041E = 910
43187 45C38906CB67283DCDC0 = 1211
43197 03FE143812FE1830ECCB = 1116
43207 5020E8114C4CCD64060D = 837
43217 C34706CB5F28DA21A103 = 1025
43227 CDCC0328D211980570E4 = 1186
43237 0F6F2600EB195E19E55E = 866
43247 19E3235E23562366C911 = 857
43257 410521BF0378FEC0381D = 948
43267 FEED2814FECB280EFEFD = 1569
43277 2806FEDD20D0CBCECBC6 = 1571
43287 1804CBEECBE6C311042E = 1164
43297 06FE8030C12E1111DD05 = 935
43307 FE403809FE7620B41E2A = 1039
43317 C38906E60720A7CB6828 = 1121
43327 A32E1018A34042444748 = 753
43337 4BD409384A3C3F4048CC = 889
43347 010249CD89067832BD03 = 786
43357 D6C76F37CDDE0678D681 = 1475

43367 326B0521BE03CB00C823 = 826
43377 CBD6CDBA0678FED72806 = 1449
43387 3E0430023E06C3BE06C2 = 769
43397 2515BF3B8CBD361504BD = 905
43407 29B7371504B23C8CB4A5 = 1027
43417 B26E101316191C1D1F29 = 499
43427 080B1E21001500218C3D = 337
43437 0165903E65018C200E8C = 736
43447 8838138C907089256286 = 1013
43457 5FD47C1E0E8C70388C13 = 942
43467 04386308CB6828041E5C = 640
43477 1874CB60285C54266118 = 814
43487 5F121316191B1D1F220A = 310
43497 2225281315171A262959 = 368
43507 3153388C044F3891884A = 822
43517 398C473901443A014238 = 575
43527 01024267341D0E8C3838 = 519
43537 8891333A8C313511032D = 697
43547 380101041D0802CD5806 = 400
43557 CDC5035AFE04DC89040C = 1128
43567 5CCB70201E1D1830CD43 = 842
43577 06CDB806184DCD89065A = 940
43587 5418010DCD89065ACD58 = 853
43597 060C0D5CC07CFE012810 = 750
43607 FE053866CDC503CB7B28 = 1188
43617 08CBBB1F180378E60783 = 944
43627 5FFE07280EFE0A280AFE = 978
43637 0F2014CB782810180D3A = 541
43647 BF03CB472807CBF3CB4F = 1243
43657 20011CE5D5C50E6421B0 = 1023
43667 0123CB7E28FB1D20F823 = 1000
43677 7ECB7F2013CDA40618F5 = 1151
43687 CBBFFE0A28BE3010E5CD = 1384
43697 BE06E1C9CDA406C1D1E1 = 1624
43707 0CC03E2C187ECD9704FE = 1074
43717 03281D3816FE0653CD97 = 849
43727 04636A3819CBBCCBB428 = 1104
43737 13CBFCCBF4180D6B2600 = 1103
43747 18091600CB7B28011519 = 468
43757 A73ABF03CB7F28310E01 = 853
43767 30010D11F0D8CB110711 = 781
43777 18FCCD1107119CFFCD11 = 1155
43787 071EF6CD11077D180AAF = 846
43797 3C1938FCED523D28010C = 826
43807 5FAFB9C87B18113803CD = 1083
43817 2907657CCDC1031FCD34 = 962
43827 077CE60FC630FE3A3802 = 992
43837 C607E52ABB03772322BB = 1041
43847 03E1C920202020202020 = 653
43857 20202020202020202020 = 320
43867 20202020202020202020 = 320
43877 20202020202020202020 = 320
43887 2020200A0D00FF000000 = 374
43897 0000000000000000 = 0
```

ISASSEMBLER

THIS DISASSEMBLER occupies less than 2K once loaded and can be located anywhere in memory. It can be called from within Basic as an external command and (a) correctly disassembles all Z80 op codes using standard mnemonics including the RST instructions as implemented in the 464 Operating System (OS); (b) disassembles routines in Rom or Ram; (c) sends output in hex or decimal to either screen or printer.

Those who have tried to Peek the Rom to get a glimpse of the O.S. will appreciate option (b) since Peek always returns the contents of Ram. The program can be altered to a certain degree to suit user requirements.

The program has been implemented as an RSX — Resident System Extension — and once loaded sits above Himem. However, since Himem can vary dependent on space reserved for expansion Roms, for example, disc Rom or user machine-code routines then an RSX needs to be relocatable. This is achieved by an additional routine which is called before the program is "logged on" with the OS and makes use of the fact that a Call from within Basic enters the routine with the DE register pair containing the argument of the Call instruction. For those interested in using the RSX facility, the following illustrates how "logging on" is achieved:

LD BC,NNNN	Address of Command Table Pointer
LD HL,NNNN	Address of 4 bytes for OS to use as workspace
JP BCD1	Log on external command(s) with OS
COM TAB POINTER	2 bytes holding start address of keyword table
JP NNNN	FLASH routine
JP NNNN	INVERSE routine
etc	
FLAS H + 80h	Last letter in each keyword has bit 7 set
INVERS E + 80h	
etc	
0	End of keywords marker

Once logged on an external command server is recognised by preceding the keyword with shifted @

:FLASH

Parameters can also be passed to the external command server routine in the following fashion:

:FLASH,400,23

On entry to the routine the A register holds the number of parameters being passed, the index register IX points to an area in memory where the parameters are stored in two-byte integer form in the reverse order to that in which they were entered, that is, from the above example:

A = 2
IX + 0 = 23
IX + 1 = 0
IX + 2 = 144
IX + 3 = 1 (1*256 + 144 = 400)

Strings can also be passed to the external command server routine: in this case IX

(continued on page 123)

Table 1.

	SET (1)	RESET (0)
Bit 0 (1)	Output to printer	Output to screen
Bit 1 (2)	LROM On	LROM Off
Bit 2 (4)	UROM On	UROM Off
Bit 3 (8)	Disc ROM	BASIC ROM
Bit 4 (16)	Not used	
Bit 5 (32)	Not used	
Bit 6 (64)	Output as DATA	Output as code
Bit 7 (128)	Output in decimal	Output in hex

Bit 0. Self-explanatory.

Bit 1. Selects what will be disassembled (LRom or Ram) when addressing memory between 0000h and 3FFFh.

Bit 2. Selects what will be disassembled (URom or Ram) when addressing memory between C000h and FFFFh.

Bit 3. When Bit 2 is Reset, this Bit is ignored. When Bit 2 is Set, this Bit selects the URom which will be addressed.

Bit 6. When Reset memory between the start and finish addresses is disassembled; when Set this memory area is output as Data.

Bit 7. Self-explanatory.

Listing 3.

```
300 MODE 1
310 LOCATE 1,25:INPUT;"START ADDRESS ";address:CLS
320 LOCATE 1,25
330 c$="":check=0
340 FOR n=0 TO 9
350 a=PEEK(address+n)
360 check=check+a:c$=c$+HEX$(a,2)
370 NEXT
380 PRINT address;TAB(10);c$;TAB(32);"= ";check
390 a$=INKEY$:IF a$="" THEN 390
400 address=address+10:GOTO 330
```

(continued from page 121)

points to an address which in turn points to a three-byte string describer where byte 0 holds the length of the string, bytes 1 and 2 hold the address where the string is stored.

I have selected a method of entering the program often seen in this magazine, because I have found this method to produce the least number of errors. The program should be entered with all expansion Roms disconnected. This can be checked by:

PRINT HIMEM

which should return the value 43903. Enter the hex loader program in listing 1 and when complete type Run 100. Input as prompted the code in listing 2, code and Checksum being separated by a comma. The code can be Saved at any stage of its development by Escaping from the loader program and typing Run 20. It can then be reLoaded at a later date by

RUN"

Once all code has been entered, delete the loader program and enter the Basic Load/Save program in listing 4. Save the completed program to tape by typing Run 4. Enter as a direct command

CALL 41987

and if the ready prompt appears go on to use the program as described later in this article. If the system crashes, reset the system then enter:

MEMORY 41766: LOAD "DISASSEMBLER":LOAD "!CODE", 41767

Once Loaded, check the code by entering listing 3 and Run 300. Compare the results to listing 2 and, after all errors have been corrected, reSave as above.

Once the program has been Saved in its final form it can be recalled by

RUN"

if no other program is in memory or by

CHAIN MERGE "DISASSEMBLER"

if a program in memory is to be retained. To use the disassembler the following syntax is required:

!disassembler, <start address>, <finish address>, <output state>

The start and finish addresses are mandatory, the output state optional defaulting to zero if omitted. All parameters can be either numeric or variable expressions.

For example, if it is required to send output to the printer, LRom on, URom off, as code in decimal, then the output state value is as follows:

$(1^*1)+(1^*2)+(0^*4)+(0^*64)+(1^*128)=131$

When output to the screen is selected, the disassembler uses stream 7 and sets a window size on this stream 40 characters wide and 25 high. The disassembler must be used in Mode 1 or 2 or it will not operate.

As it stands, the disassembler produces mnemonics using upper case. For those who prefer mnemonics in lower case the program in listing 5 should be entered and run before saving the disassembler in its final form.

For those who intend to use a printer with the disassembler, various options are available. If using cut sheet paper then the number of lines per sheet can be set by Poking the value to 42210 — A4E2h. The disassembler will then wait at the end of the sheet for "C" to be pressed giving time to change the sheet.

Listing 4.

```
1  a=HIMEM-2137:MEMORY a:LOAD "!CODE",a+1:CLOSEIN
2  CALL a+221:REM RELOCATE AND LOG ON DISASSEMBLER
3  MEMORY a+261:DELETE 1-4:END
4  SAVE "DISASSEMBLER":SAVE "!CODE",b,41767,2136:END
```

If using continuous paper this value should be set to 255 — FFh. The area from 43893 (AB75h) to 43903 (AB7Fh) has been set aside for up to 10 control codes which the user can enter and are sent before printing starts. The control codes must be terminated by a marker byte 255 — FFh.

The only non-standard results produced by the disassembler are those of the RST 08h, 10h, 18h and 28h instructions. The 464 OS implements these op codes as "extensions" to the instruction set. When executed, the following two bytes are taken as an inline address; the RST 08h uses bits 14 and 15 to select the Rom state, the RST 10h uses these bits for Rom select.

The RST 18h instruction takes a third byte for Rom state/select but is thus able to jump to any location in memory in any Rom. Typical examples of the output produced by the disassembler for each of these RST instructions are given below plus their interpretation.

Mnemonic	Remark
RST 08,3B0F	JP 3B0F
UROM OFF LROM ON	Upper ROM disabled, Lower ROM enabled
RST 10,DFCC	CALL DFCC
ROM +2	Side CALL to an expansion ROM
RST 18,OD17	CALL OD17
UROM ON LROM OFF	Upper ROM enabled, Lower ROM disabled
ROM 199	Select ROM 199
RST 28,224C	JP 224C
LROM ON	Lower ROM enabled ■

Listing 5.

```
10 FOR address=42421 TO 42828
20 a=PEEK(address)
30 IF (a>64 AND a<91) OR (a>192
AND a<219) THEN 40 ELSE 50
40 a=a+32:POKE address,a
50 NEXT
```

Memory map.

Address	Routine
41767 (A327h)	Data for relocating routine.
41987 (A403h)	Relocating routine.
42019 (A423h)	Log on disassembler with OS.
42028 (A42Ch)	Data for log-on.
42033 (A431h)	Print syntax error.
42056 (A448h)	Entry to disassembler: check syntax.
42074 (A45Ah)	Set up output state, finish and start address.
42101 (A475h)	Set up stream and window size.
42124 (A48Ch)	Check if printer connected: send control codes.
42143 (A49Fh)	Set up required rom state.
42174 (A4BEh)	Call disassember start.
42177 (A4C1h)	Restore previous Rom state and stream: return.
42198 (A4D6h)	Check if start = current address.
42209 (A4E1h)	Disassembler start: set up No. of lines for screen/printer.
42236 (A4FCh)	Set up print position in print buffer.
42244 (A504h)	Call build up mnemonic in print buffer.
42253 (A50Dh)	Print address, op code and operand to print buffer.
42327 (A557h)	Wait for key press.
42354 (A572h)	Copy print buffer to current screen line/printer: return.
42393 (A599h)	Data for syntax error.
42407 (A5A7h)	Data for log-on assembler.
42420 (A5B4h)	Mnemonic keyword list/control codes.
42892 (A78Ch)	Data for valid IX/IY op codes.
42924 (A7ACh)	Data for valid ed op codes.
42932 (A7B4h)	Workspace for OS.
42936 (A7B8h)	Start, current and finish addresses.
42942 (A7BEh)	Print position in print buffer.
42944 (A7COh)	RST flags.
42946 (A7C2h)	Disassembler flag.
42947 (A7C3h)	Conversion routines.
42958 (A7CEh)	Check for valid IX/IY and ed op codes routine.
42980 (A7E4h)	RST 18/28 handler.
43009 (A801h)	Data handler.
43028 (A814h)	Entry to build up mnemonic in print buffer.
43035 (A81Bh)	RST 08/10 handler.
43114 (A86Ah)	CB op code handler.
43175 (A8A7h)	IX/IY op code handler.
43195 (A8BBh)	Ed op code handler.
43253 (A8F5h)	Check for CB,ED and IX/IY op codes.
43295 (A91Fh)	Op codes <64.
43332 (944h)	Data for op codes >= 192.
43419 (A99Bh)	Data for ed op codes.
43488 (A9EOh)	Data for op codes <64.
43575 (AA37h)	Various print routines.
43660 (AA8Ch)	Find mnemonic keyword and print to print buffer.
43713 (AAC1h)	Print decimal/hex number to print buffer.
43850 (AB4Ah)	Print buffer.
43890 (AB72h)	Carriage return: line feed.
43892 (AB74h)	Control codes.
43903 (AB7Fh)	End.