

Disassembled Handbook for TRS-80

Richcraft Engineering Ltd.
Drawer 1065
Chautauqua, New York 14722

Copyright © 1980

"TRS-80 is a Registered Trademark of TANDY CORP."

VOLUME I

| CHAPTER | SUBJECT | PAGE |
|---------|---|------|
| i | - Foreword | 3 |
| ii | - Introduction | 4 |
| 1. | - Decoding Level II ROM Function CALL Locations | 6 |
| 2. | - Integer, Single & Double Precision Arithmetic | 13 |
| 3. | - Using ROM Trig, Exponent, Log, et al Subroutines | 23 |
| 4. | - Ancillary Level II ROM Subroutines | 29 |
| 5. | - Level II BASIC ROM CALL Addresses-Alphabetical | 35 |
| 6. | - Level II ROM Hex Code | 37 |
| 7. | - Multi-Base Number Conversion Program | 53 |
| 8. | - Print All Zeros With A Slash (Source/Object Code) | 56 |
| 9. | - Self-Test Questions | 58 |
| 10. | - Bibliography and Self-Test Answers | 65 |

- NOTICE -

TRS-80 is a trademark of the Tandy Corporation. The users of the CALLED Level II ROM subroutines presented herein, must of necessity have purchased Level II BASIC and thereby paid the royalties due the copyright owners. All data in this handbook is purely instructional in nature and serves as a supplement to the original documentation provided by Radio Shack/Micro-soft and Zilog. Richcraft's absolute refusal to provide copies or extracts of the original documentation assures the copyright owners that all users must purchase Level II ROM from Radio Shack, thereby paying royalties due. Furthermore, a considerable number of Level II ROM memory locations have been blanked out to render the program useless to those who have not purchased the original copyrighted program.

Copyright (c) 1980 by Richcraft Engineering Ltd.

Third Printing: March 1, 1980

All rights reserved. Printed in the United States of America. No part of this publication may be reproduced, stored in any retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior written permission of the publisher.

Published by Richcraft Engineering Ltd.
Drawer 1065
Chautauqua, New York 14722

(716) 753 2654

- FOREWORD -

This handbook started out as a collection of lectures prepared as "fill-routines" by the author while on the tour circuit promoting his new book, "The Gunnplexer Cookbook - A 10 GHz Microwave Primer." It is a rather long jump (at least in frequency and wave-length) from the 1.77 MHz clock of the TRS-80 to the 10250 MHz band where the Gunnplexer operates, but surprisingly the microwave buff and computer buff have more in common than meets the eye; i.e., a need to communicate whether it be by simple fm voice modulation or ASCII digital data link.

It was both gratifying and surprising to find that microwave radio amateurs had as much interest in TRS-80 assembly language programming as did computer hobbyists in setting up low-cost microwave digital data links. Ours is indeed a very small world with most all things interrelated.

This handbook is NOT for the beginning assembly language programmer who should certainly learn the fundamentals of the art before attempting to use the shortcuts and "tight code" programming made possible by using the myriad excellent Level II ROM subroutines presented. The moderately experienced assembly language programmer who understands the difference between his/her JPs and JRs, SETs and REsets, and BITs and bytes will find that utilizing Level II ROM subroutines opens up an entirely new vista to the wonderful world of assembly language programming. Many dull but demanding rote subroutines may now be accomplished often with a single CALL compared with previous multi-line/multi-page programming that had to be entered a line at a time. Programmers arise. Cast off the yoke of ignorance and START HAVING FUN writing assembly language programs that run 300+ times faster than BASIC in less than 1/10th memory. With practice, you (and the author) will soon be writing assembly language programs as quickly and with as little effort as those written in BASIC, Fortran, Cobol, or Pascal.

ACKNOWLEDGMENTS:

Without the generous aid and assistance of both Nancy A. Courtney and Margaret C. Merz this handbook would have never come to pass. Their encouragement, hard work, and persistent insistence that we keep at it are gratefully appreciated.

ADDITIONAL THANK YOUS: to the late Charles Tandy for his courage and investment in making the TRS-80 a happening; to Mumford Micro for a 3 speed clock; to Western I/O for an IBM Selectric printer that never quits; to Apparat for NEWDOS+, to Shroyer for Electric Pencil; to Microsoft for the world's most extensively used BASIC; and lastly to Radio Shack, who inspite of other shortcomings build the WORLD'S MOST COST EFFECTIVE MICROCOMPUTER. Thank you each and every one.

- INTRODUCTION -

Disassembled machine code of any variety WITHOUT comments is worth about as much as a TRS-80 without electric power. Constants, address table entry points, and data lists are translated into utter meaningless/misleading garbage by the disassembler program. Disassembled Level II ROM prints out EX AF,AF' from memory locations 005AH, 1479H, 1619H, 18BAH, 18BCH, etc., etc., when in actuality none of the alternate register pairs are ever used in this ROM's BASIC program.

Decoding and making any sense out of any object code program is impossible if one does not at least have a clue to the program's intended function. Fortunately we know exactly what Level II ROM's functions and capabilities are, so what appeared to be an impossible decoding problem is reduced to only an extremely difficult one. So difficult in fact, that even Radio Shack's computer division in Fort Worth, Texas does not fully understand this excellent "Tight Code" written by Microsoft's Paul Allen and Bill Gates. An example that proves this point is the Radio Shack book, "TRS-80 Assembly Language Programming," #62-2006, that was introduced mid-year 1979 has virtually no references to Level II ROM subroutines of any variety. This is incredible and truly theater of the absurd. It is either criminal negligence on Radio Shack's part, or just plain stupidity. Always hoping for the best, we shall presume the latter.

The level of difficulty in decoding the TRS-80 ROM may be measured by the fact that after 2 years of worldwide use by over 200,000 computer buffs ranging from beginners to advanced programmers with years of experience, the Level II ROM entry points for virtually ALL the BASIC functions and related subroutines have never been published with comments, or disclosed by anyone, anywhere, EXCEPT to a very limited audience by a genius named Andrew Hildebrand, located in the southwestern wilderness of the US.

This handbook is dedicated to Mr. Hildebrand's genius, to his persistence in unraveling a very tangled web, and to him personally for this very considerable accomplishment. The author's only contribution is this handbook which will hopefully remove the shroud of mystery from a previous "black hole" by attempting to make the subject understandable to hobbyists, high school and college students, and even computer science professors, many of whom do not have the slightest idea of how to efficiently utilize the myriad Level II ROM subroutines in TRS-80 assembly language programming. This handbook will also assist users in decoding the majority of ROM subroutines in ALL Microsoft BASIC's including those used by Apple, Pet, KIM, Heathkit, et al microcomputer systems.

Each section of this handbook is hopefully self-explanatory. The Level II ROM hex code presented in Chapter 6, though very

lengthy, is included as a necessary reference. The self-test questions & answers in Chapters 9 & 10 allow you to check your progress after each Chapter. The limited first edition of this handbook was bound and glued with a cloth backing, but even so would not stay together under repeated usage. As such, this second edition is being delivered in loose leaf binder format.

SUMMARY:

Chapter 1 illustrates how the majority of the ROM functions' CALL locations are decoded.

Chapter 2 includes three source/object code programs and text illustrating the use of simple ROM " + - * / " integer, single precision, and double precision arithmetic subroutines.

Chapter 3 describes how to use the trigonometric, exponent, log, CINT, CSNG, CDBL, et al, functions in concert with the RAM accumulator and CDBL store, plus a demonstration program.

Chapter 4 covers a number of the more useful and important Level II ROM ancillary subroutines; Chapter 5 is a summary of virtually all BASIC function CALL addresses.

Chapter 6 is the PARTIAL Level II ROM code in hexadecimal.

Chapter 7 presents and explains an extremely useful number conversion program that is virtually a must for the serious assembly language programmer working with the TRS-80. It is written in BASIC for comprehension/modification and covers:

- DECIMAL TO BINARY ENTER D
- BINARY TO DECIMAL ENTER B
- HEXADECIMAL TO BINARY ENTER HB
- DECIMAL TO HAXADECIMAL ENTER DH
- HEXADECIMAL TO DECIMAL ENTER HD
- SPLIT TRS-80 TO DECIMAL ENTER SP
- DECIMAL TO SPLIT HEXADECIMAL ENTER DS
- SPLIT HEXADECIMAL TO DECIMAL ENTER SD

The first 5 conversions are obvious ones. The 6th, SPLIT TRS-80 TO DECIMAL, is unique in that it takes decimal values displayed via the TRS-80 PEEK command from 2 adjacent memory locations (two INPUTS required), then converts them to hex, reverses the 2 hex numbers as the Z-80 stores LSB first and MSB second, then converts the 4 digit hex number to decimal, and displays it on video. This conversion is a real time saver when extracting addresses (0 to 65535) from ROM or RAM.

Chapter 8 is a useful print all zeros with a slash program.

Chapter 9 includes self-test questions for Chapters 1 - 8.

Chapter 10 is a bibliography and answers to the self-test.

- CHAPTER 1 -

DECODING LEVEL II ROM FUNCTION CALL LOCATIONS

INTRODUCTION:

During the past two years, the Level II BASIC written by Microsoft originally for the TRS-80 has become the world standard de facto BASIC used by virtually every significant microcomputer manufacturer. As of January 1980, the number of microcomputers delivered to end users with Level II or modest variations of Level II BASIC is estimated to exceed 300,000 plus. With the number of BASICs in the marketplace counted in the dozens including: Hewlett-Packard BASIC, General Electric BASIC, and even monolithic IBM's "VS BASIC," there must be a number of good reasons for the near universal adoption of Microsoft's BASIC.

1. It is cheap? Answer: no, in actuality the license to use this BASIC is incredibly expensive. Heathkit does not charge hobbyists \$100. a copy just for the program for fun and games.

2. Is it efficient? Answer: you bet it is. Previous BASICs the author has studied required 22K to 32K memory to minimally perform the same functions, if indeed as many.

3. It is cost-effective? Answer: even at the high licensing price it is VERY cost-effective when one considers the trade-offs between available program memory remaining with the inherent 64K maximum imposed by most all 8 bit microprocessors of the current generation. Some of the other BASICs mentioned above only leave the user about 18K of useful RAM when a disk operating system and extended disk BASIC program is added. This is ridiculous in a 64K MEM system.

Let's now take a look at the prodigious functions and their CALL locations in this marvel of "tight code" programming written by Paul Allen and Bill Gates. We doubt if there was any intentional encrypting involved when the program was written as encryption takes memory and memory costs money either directly or indirectly as pointed out above. The main reason it has been difficult to decode the Level II ROM was brought about primarily by its compactness; i.e., nothing wasted, nothing unused, and no easily deciphered pointers telling the code breaker, "here I am, use me."

LEVEL II ROM FUNCTION NAME LOCATIONS:

Are the easiest to find of all. Only a retarded 3rd grader would overlook them at memory locations 5712 through 6172. Figure 1 is an excruciatingly simple BASIC program that will display these names and their location on video for you. The first letter of each name's MSB is masked by subtracting 128 to obtain its ASCII equivalent. Figure 2 is a print out of this program. Remember, the numbers are the name's location, NOT the CALL location.

```

10 ' MICROSOFT BASIC FUNCTIONS' NAME LIST IN LEVEL II ROM
15 '
20 CLS:FORN=5712TO6175:Y=PEEK(N):IFY>127THENY=Y-128:M=N
25 Z=N+1:IFPEEK(Z)>127THENPRINTCHR$(Y);"=";ELSEGOTO35
30 PRINTM,:GOTO40
35 PRINTCHR$(Y);
40 NEXT
    
```

Figure 1

| | | | |
|-----------------|----------------|---------------|---------------|
| END = 5712 | FOR = 5715 | RESET = 5718 | SET = 5723 |
| CLS = 5726 | CMD = 5729 | RANDOM = 5732 | NEXT = 5738 |
| DATA = 5742 | INPUT = 5746 | DIM = 5751 | READ = 5754 |
| LET = 5758 | GOTO = 5761 | RUN = 5765 | IF = 5768 |
| RESTORE = 5770 | GOSUB = 5777 | RETURN = 5782 | REM = 5788 |
| STOP = 5791 | ELSE = 5795 | TRON = 5799 | TROFF = 5803 |
| DEFSTR = 5808 | DEFINT = 5814 | DEFSNG = 5820 | DEFDBL = 5826 |
| LINE = 5832 | EDIT = 5836 | ERROR = 5840 | RESUME = 5845 |
| OUT = 5851 | ON = 5854 | OPEN = 5856 | FIELD = 5860 |
| GET = 5865 | PUT = 5868 | CLOSE = 5871 | LOAD = 5876 |
| MERGE = 5880 | NAME = 5885 | KILL = 5889 | LSET = 5893 |
| RSET = 5897 | SAVE = 5901 | SYSTEM = 5905 | LPRINT = 5911 |
| DEF = 5917 | POKE = 5920 | PRINT = 5924 | CONT = 5929 |
| LIST = 5933 | LLIST = 5937 | DELETE = 5942 | AUTO = 5948 |
| CLEAR = 5952 | CLOAD = 5957 | CSAVE = 5962 | NEW = 5967 |
| TAB(= 5970 | TO = 5974 | FN = 5976 | USING = 5978 |
| VARPTR = 5983 | USR = 5989 | ERL = 5992 | ERR = 5995 |
| STRING\$ = 5998 | INSTR = 6005 | POINT = 6010 | TIME\$ = 6015 |
| MEM = 6020 | INKEY\$ = 6023 | THEN = 6029 | NOT = 6033 |
| STEP = 6036 | + = 6040 | - = 6041 | * = 6042 |
| / = 6043 | ↑ = 6044 | AND = 6045 | OR = 6048 |
| > = 6050 | = = 6051 | < = 6052 | SGN = 6053 |
| INT = 6056 | ABS = 6059 | FRE = 6062 | INP = 6065 |
| POS = 6068 | SQR = 6071 | RND = 6074 | LOG = 6077 |
| EXP = 6080 | COS = 6083 | SIN = 6086 | TAN = 6089 |
| ATN = 6092 | PEEK = 6095 | CVI = 6099 | CVS = 6102 |
| CVD = 6105 | EOF = 6108 | LOC = 6111 | LOF = 6114 |
| MKI\$ = 6117 | MKS\$ = 6121 | MKD\$ = 6125 | CINT = 6129 |
| CSNG = 6133 | CDBL = 6137 | FIX = 6141 | LEN = 6144 |
| STR\$ = 6147 | VAL = 6151 | ASC = 6154 | CHR\$ = 6157 |
| LEFT\$ = 6161 | RIGHT\$ = 6166 | MID\$ = 6172 | |

Figure 2

MATCHING BASIC FUNCTIONS WITH ROM CALL ADDRESSES:

Now the decoding game becomes somewhat more interesting. Not difficult yet, because no tricky encipherment was used. We should remember that encoding costs memory and memory = money. One does not have to search very far in memory for the location of each BASIC function's CALL address

These addresses are split into two groups. The first group begins a few bytes after the end of the function name list at MEM location 6178 and runs through 6351. This group covers all functions from END through 'less than.' The second group begins at MEM location 5640 and runs through 5711. This group includes all BASIC functions from SGN through the end of the function list, MID\$.

With the exception of those BASIC functions from TAB to 'less than,' all CALL locations are stored in MEM using the standard Zilog Z-80 format (the genius of Federico Faggin, Z-80 creator appears again), with the LSB (least significant byte) first, and the MSB (most significant byte) second, in the following memory location. Figure 3 illustrates a little BASIC program that will display the BASIC function, an = sign, then the MEM location of the stored CALL address for this specific function and lastly the CALL address in standard TRS-80 "PEEK" format.

```

10 ' PROGRAM TO MATCH BASIC FUNCTIONS WITH CALL ADDRESSES
15 '
20 CLS:PRINT"FUNCT=ADDRESS      LSB-MSB          FUNCT=ADDRESS      LSB-MSB"
25 A=6176:FORX=5712TO6175:Y=PEEK(X):IFY>127THENY=Y-128
30 Z=X+1:IFPEEK(Z)>127THENPRINTCHR$(Y);"=";ELSEGOTO45
35 A=A+2:IFA=6352THENA=5640
40 PRINTA,PEEK(A);"-";PEEK(A+1),:GOTO50
45 PRINTCHR$(Y);
50 NEXT
55 END

```

- Figure 3 -

Figure 4 is a printout of Figure 3's program output using those addresses whose LSB/MSB are directly translatable to CALL addresses. The other functions' CALL addresses are fully covered in Chapter 5. Figure 5 is a printout of Level II ROM MEM locations 1600H through 18FFH to illustrate how Level II ROM BASIC function CALL locations are determined.

| FUNCT=ADDRESS | LSB-MSB | FUNCT=ADDRESS | LSB-MSB |
|----------------|----------|----------------|----------|
| END = 6178 | 174 - 29 | FOR = 6180 | 161 - 28 |
| RESET = 6182 | 56 - 1 | SET = 6184 | 53 - 1 |
| CLS = 6186 | 201 - 1 | CMD = 6188 | 115 - 65 |
| RANDOM = 6190 | 211 - 1 | NEXT = 6192 | 182 - 34 |
| DATA = 6194 | 5 - 31 | INPUT = 6196 | 154 - 33 |
| DIM = 6198 | 8 - 38 | READ = 6200 | 239 - 33 |
| LET = 6202 | 33 - 31 | GOTO = 6204 | 194 - 30 |
| RUN = 6206 | 163 - 30 | IF = 6208 | 57 - 32 |
| RESTORE = 6210 | 145 - 29 | GOSUB = 6212 | 177 - 30 |
| RETURN = 6214 | 222 - 30 | REM = 6216 | 7 - 31 |
| STOP = 6218 | 169 - 29 | ELSE = 6220 | 7 - 31 |
| TRON = 6222 | 247 - 29 | TROFF = 6224 | 248 - 29 |
| DEFSTR = 6226 | 0 - 30 | DEFINT = 6228 | 3 - 30 |
| DEFSNG = 6230 | 6 - 30 | DEFDBL = 6232 | 9 - 30 |
| LINE = 6234 | 163 - 65 | EDIT = 6236 | 96 - 46 |
| ERROR = 6238 | 244 - 31 | RESUME = 6240 | 175 - 31 |
| OUT = 6242 | 251 - 42 | ON = 6244 | 108 - 31 |
| OPEN = 6246 | 121 - 65 | FIELD = 6248 | 124 - 65 |
| GET = 6250 | 127 - 65 | PUT = 6252 | 130 - 65 |
| CLOSE = 6254 | 133 - 65 | LOAD = 6256 | 136 - 65 |
| MERGE = 6258 | 139 - 65 | NAME = 6260 | 142 - 65 |
| KILL = 6262 | 145 - 65 | LSET = 6264 | 151 - 65 |
| RSET = 6266 | 154 - 65 | SAVE = 6268 | 160 - 65 |
| SYSTEM = 6270 | 178 - 2 | LPRINT = 6272 | 103 - 32 |
| DEF = 6274 | 91 - 65 | POKE = 6276 | 177 - 44 |
| PRINT = 6278 | 111 - 32 | CONT = 6280 | 228 - 29 |
| LIST = 6282 | 46 - 43 | LLIST = 6284 | 41 - 43 |
| DELETE = 6286 | 198 - 43 | AUTO = 6288 | 8 - 32 |
| CLEAR = 6290 | 122 - 30 | CLOAD = 6292 | 31 - 44 |
| CSAVE = 6294 | 245 - 43 | NEW = 6296 | 73 - 27 |
| INT = 5642 | 55 - 11 | ABS = 5644 | 119 - 9 |
| FRE = 5646 | 212 - 39 | INP = 5648 | 239 - 42 |
| POS = 5650 | 245 - 39 | SQR = 5652 | 231 - 19 |
| RND = 5654 | 201 - 20 | LOG = 5656 | 9 - 8 |
| EXP = 5658 | 57 - 20 | COS = 5660 | 65 - 21 |
| SIN = 5662 | 71 - 21 | TAN = 5664 | 168 - 21 |
| ATN = 5666 | 189 - 21 | PEEK = 5668 | 170 - 44 |
| CVI = 5670 | 82 - 65 | CVS = 5672 | 88 - 65 |
| CVD = 5674 | 94 - 65 | EOF = 5676 | 97 - 65 |
| LOC = 5678 | 100 - 65 | LOF = 5680 | 103 - 65 |
| MKI\$ = 5682 | 106 - 65 | MKS\$ = 5684 | 109 - 65 |
| MKD\$ = 5686 | 112 - 65 | CINT = 5688 | 127 - 10 |
| CSNG = 5690 | 177 - 10 | CDBL = 5692 | 219 - 10 |
| FIX = 5694 | 38 - 11 | LEN = 5696 | 3 - 42 |
| STR\$ = 5698 | 54 - 40 | VAL = 5700 | 197 - 42 |
| ASC = 5702 | 15 - 42 | CHR\$ = 5704 | 31 - 42 |
| LEFT\$ = 5706 | 97 - 42 | RIGHT\$ = 5708 | 145 - 42 |
| MID\$ = 5710 | 154 - 42 | | |

Figure 4

Function=Call Address MEM Location Call-Address

| | | | | | | | | | |
|------|------|------|------|------|------|------|------|------|----------------------|
| 1600 | 6CAA | AA7F | 0000 | 0081 | 8A09 | 370B | 7709 | D427 |7..... |
| 1610 | EF2A | F527 | E713 | C914 | 0908 | 3914 | 4115 | 4715 | .*'.....9.A.G. |
| 1620 | A815 | BD15 | AA2C | 5241 | 5841 | 5E41 | 6141 | 6441 |,RAXA.A.A.A |
| 1630 | 6741 | 6A41 | 6D41 | 7041 | 7F0A | B10A | DB0A | 260B | .A.A.A.A.....& |
| 1640 | 032A | 3628 | C52A | 0F2A | 1F2A | 612A | 912A | 9A2A | .*6(*.*.*.*.*.* |
| 1650 | C54E | 44C6 | 4F52 | D245 | 5345 | 54D3 | 4554 | C34C | .ND.OR.ESET.ET.L |
| 1660 | 53C3 | 4D44 | D241 | 4E44 | 4F4D | CE45 | 5854 | C441 | S.MD.ANDOM.EXT.A |
| 1670 | 5441 | C94E | 5055 | 54C4 | 494D | D245 | 4144 | CC45 | TA.NPUT.IM.EAD.E |
| 1680 | 54C7 | 4F54 | 4FD2 | 554E | C946 | D245 | 5354 | 4F52 | T.OTO.UN.F.ESTOR |
| 1690 | 45C7 | 4F53 | 5542 | D245 | 5455 | 524E | D245 | 4DD3 | E.OSUB.ETURN.EM. |
| 16A0 | 544F | 50C5 | 4C53 | 45D4 | 524F | 4ED4 | 524F | 4646 | TOP.LSE.RON.ROFF |
| 16B0 | C445 | 4653 | 5452 | C445 | 4649 | 4E54 | C445 | 4653 | .EFSTR.EFINT.EFS |
| 16C0 | 4E47 | C445 | 4644 | 424C | CC49 | 4E45 | C544 | 4954 | NG.EFDBL.INE.DIT |
| 16D0 | C552 | 524F | 52D2 | 4553 | 554D | 45CF | 5554 | CF4E | .RROR.ESUME.UT.N |
| 16E0 | CF50 | 454E | C649 | 454C | 44C7 | 4554 | D055 | 54C3 | .PEN.IELD.ET.UT. |
| 16F0 | 4C4F | 5345 | CC4F | 4144 | CD45 | 5247 | 45CE | 414D | LOSE.OAD.ERGE.AM |
| 1700 | 45CB | 494C | 4CCC | 5345 | 54D2 | 5345 | 54D3 | 4156 | E.ILL.SET.SET.AV |
| 1710 | 45D3 | 5953 | 5445 | 4DCC | 5052 | 494E | 54C4 | 4546 | E.YSTEM.PRINT.EF |
| 1720 | D04F | 4B45 | D052 | 494E | 54C3 | 4F4E | 54CC | 4953 | .OKE.RINT.ONT.IS |
| 1730 | 54CC | 4C49 | 5354 | C445 | 4C45 | 5445 | C155 | 544F | T.LIST.ELETE.UTO |
| 1740 | C34C | 4541 | 52C3 | 4C4F | 4144 | C353 | 4156 | 45CE | .LEAR.LOAD.SAVE. |
| 1750 | 4557 | D441 | 4228 | D44F | C64E | D553 | 494E | 47D6 | EW.AB(.O.N.SING. |
| 1760 | 4152 | 5054 | 52D5 | 5352 | C552 | 4CC5 | 5252 | D354 | ARPTR.SR.RL.RR.T |
| 1770 | 5249 | 4E47 | 24C9 | 4E53 | 5452 | D04F | 494E | 54D4 | RING\$.NSTR.OINT. |
| 1780 | 494D | 4524 | CD45 | 4DC9 | 4E4B | 4559 | 24D4 | 4845 | IME\$.EM.NKEY\$.HE |
| 1790 | 4ECE | 4F54 | D354 | 4550 | ABAD | AAAF | DBC1 | 4E44 | N.OT.TEP.....ND |
| 17A0 | CF52 | BEBD | BCD3 | 474E | C94E | 54C1 | 4253 | C652 | .R.....GN.NT.BS.R |
| 17B0 | 45C9 | 4E50 | D04F | 53D3 | 5152 | D24E | 44CC | 4F47 | E.NP.OS.QR.ND.OG |
| 17C0 | C558 | 50C3 | 4F53 | D349 | 4ED4 | 414E | C154 | 4ED0 | .XP.OS.IN.AN.TN. |
| 17D0 | 4545 | 4BC3 | 5649 | C356 | 53C3 | 5644 | C54F | 46CC | EEK.VI.VS.VD.OF. |
| 17E0 | 4F43 | CC4F | 46CD | 4B49 | 24CD | 4B53 | 24CD | 4B44 | OC.OF.KI\$.KS\$.KD |
| 17F0 | 24C3 | 494E | 54C3 | 534E | 47C3 | 4442 | 4CC6 | 4958 | \$.INT.SNG.DBL.IX |
| 1800 | CC45 | 4ED3 | 5452 | 24D6 | 414C | C153 | 43C3 | 4852 | .EN.TR\$.AL.SC.HR |
| 1810 | 24CC | 4546 | 5424 | D249 | 4748 | 5424 | CD49 | 4424 | \$.EFT\$.IGHT\$.ID\$ |
| 1820 | A780 | AE1D | A11C | 3801 | 3501 | C901 | 7341 | D301 |8.5.....A.. |
| 1830 | B622 | 051F | 9A21 | 0826 | EF21 | 211F | C21E | A31E | ."...!.&!!!..... |
| 1840 | 3920 | 911D | B11E | DE1E | 071F | A91D | 071F | F71D | 9..... |
| 1850 | F81D | 001E | 031E | 061E | 091E | A341 | 602E | F41F |A..... |
| 1860 | AF1F | FB2A | 6C1F | 7941 | 7C41 | 7F41 | 8241 | 8541 | ...*...A.A.A.A.A |
| 1870 | 8841 | 8B41 | 8E41 | 9141 | 9741 | 9A41 | A041 | B202 | .A.A.A.A.A.A.A.. |
| 1880 | 6720 | 5B41 | B12C | 6F20 | E41D | 2E2B | 292B | C62B | ...A.,.....+)+.+ |
| 1890 | 0820 | 7A1E | 1F2C | F52B | 491B | 7979 | 7C7C | 7F50 |,+I.....P |
| 18A0 | 46DB | 0A00 | 007F | 0AF4 | 0AB1 | 0A77 | 0C70 | 0CA1 | F..... |
| 18B0 | 0DE5 | 0D78 | 0A16 | 0713 | 0747 | 08A2 | 080C | 0AD2 |G..... |
| 18C0 | 0BC7 | 0BF2 | 0B90 | 2439 | 0A4E | 4653 | 4E52 | 474F |\$9.NFSNRGO |
| 18D0 | 4446 | 434F | 564F | 4D55 | 4C42 | 5344 | 442F | 3049 | DFCOVOMULBSDD/OI |
| 18E0 | 4454 | 4D4F | 534C | 5353 | 5443 | 4E4E | 5252 | 5755 | DTMOSLSSTCNNRRWU |
| 18F0 | 454D | 4F46 | 444C | 33D6 | 006F | 7CDE | 0067 | 78DE | EMOFDL3..... |

Figure 5

CONVERTING BASIC FUNCTION CALL ADDRESSES TO HEX AND DECIMAL:

Converting the BASIC function CALL addresses printed out in Figure 4 to hexadecimal and decimal is certainly simple, but nevertheless a tedious job whether done with a calculator or using Disk BASIC's &H function. There is a considerably easier way to do it.

The Mult-Base Number Conversion Program presented in Chapter 6 makes the conversion of the CALL locations to hexadecimal and decimal a real pleasure instead of a rote chore. This program may be easily modified to take the LSB and MSB from Figure 3's program output and automatically generate the CALL's decimal and hexadecimal location, but for simplicity's sake let's run through Chapter 6's standard number conversion routine for generating the CALL address for the first function shown in Figure 4, which is END.

Load the conversion program. We will use the program's SPLIT DECIMAL TO DECIMAL routine so press 'SP' ENTER. Figure 4 shows that END's CALL location is LSB = 174 and MSB = 29. The program will display "DECIMAL ?" Press 174 then ENTER. After a moment, the 174 will disappear and "DECIMAL ?" will again be displayed on video. Now, press 29 then ENTER. Faster than a speeding bullet, HEXADECIMAL 1DAE will be displayed on video followed by DECIMAL 7598 a second or two later. Aha, it really does work.

The conversion program's logic and flow is as follows:

1. Convert '174' to hexadecimal and store it.
2. Convert '29' to hexadecimal and store it.
3. Reverse the 2 hexadecimal numbers so it will read MSB/LSB.
4. Display the hexadecimal number on video.
5. Convert the hexadecimal number to decimal.
6. Display the decimal number on video.

Figure 6 is an exact printout of Figure 4's BASIC function CALL addresses in both hexadecimal and decimal. Those CALL addresses above 12288 are of course for coupling to DOS/Disk. So far, so good. Now let's JP to Chapter 2 and get some experience using Level II ROM's subroutines with integer, single precision, and double precision arithmetic. We will initially only be doing 3rd grade add, subtract, multiply, and divide, but we will be doing it with only a few CALLs instead of lines/pages of assembly language programming.

Please disregard the "curse you Red Baron," comments in Chapters 2 and 3. They were put in for reviewers/publishers who were given only a single chapter of this handbook.

MATCHING BASIC FUNCTIONS WITH MEM CALL LOCATIONS

| FUNCTION | HEX | DECIMAL | FUNCTION | HEX | DECIMAL |
|----------|------|---------|----------|------|---------|
| END | 1DAE | 7598 | FOR | 1CA1 | 7329 |
| RESET | 0138 | 312 | SET | 0135 | 309 |
| CLS | 01C9 | 457 | CMD | 4173 | 16755 |
| RANDOM | 01D3 | 467 | NEXT | 22B6 | 8886 |
| DATA | 1F05 | 7941 | INPUT | 219A | 8602 |
| DIM | 2608 | 9736 | READ | 21EF | 8687 |
| LET | 1F21 | 7969 | GOTO | 1EC2 | 7874 |
| RUN | 1EA3 | 7843 | IF | 2039 | 8249 |
| RESTORE | 1D91 | 7569 | GOSUB | 1EB1 | 7857 |
| RETURN | 1EDE | 7902 | REM | 1F07 | 7943 |
| STOP | 1DA9 | 7593 | ELSE | 1F07 | 7943 |
| TRON | 1DF7 | 7671 | TROFF | 1DF8 | 7672 |
| DEFSTR | 1E00 | 7680 | DEFINT | 1E03 | 7683 |
| DEFSNG | 1E06 | 7686 | DEFDBL | 1E09 | 7689 |
| LINE | 41A3 | 16803 | EDIT | 2E60 | 11872 |
| ERROR | 1FF4 | 8180 | RESUME | 1FAF | 8111 |
| OUT | 2AFB | 11003 | ON | 1F6C | 8044 |
| OPEN | 4179 | 16761 | FIELD | 417C | 16764 |
| GET | 417F | 16767 | PUT | 4182 | 16770 |
| CLOSE | 4185 | 16773 | LOAD | 4188 | 16776 |
| MERGE | 418B | 16779 | NAME | 418E | 16782 |
| KILL | 4191 | 16785 | LSET | 4197 | 16791 |
| RSET | 419A | 16794 | SAVE | 41A0 | 16800 |
| SYSTEM | 02B2 | 690 | LPRINT | 2067 | 8295 |
| DEF | 415B | 16731 | POKE | 2CB1 | 11441 |
| PRINT | 206F | 8303 | CONT | 1DE4 | 7652 |
| LIST | 2B2E | 11054 | LLIST | 2B29 | 11049 |
| DELETE | 2BC6 | 11206 | AUTO | 2008 | 8200 |
| CLEAR | 1E7A | 7802 | CLOAD | 2C1F | 11295 |
| CSAVE | 2BF5 | 11253 | NEW | 1B49 | 6985 |
| INT | 0B37 | 2871 | ABS | 0977 | 2423 |
| FRE | 27D4 | 10196 | INP | 2AEF | 10991 |
| POS | 27F5 | 10229 | SQR | 13E7 | 5095 |
| RND | 14C9 | 5321 | LOG | 0809 | 2057 |
| EXP | 1439 | 5177 | COS | 1541 | 5441 |
| SIN | 1547 | 5447 | TAN | 15A8 | 5544 |
| ATN | 15BD | 5565 | PEEK | 2CAA | 11434 |
| CVI | 4152 | 16722 | CVS | 4158 | 16728 |
| CVD | 415E | 16734 | EOF | 4161 | 16737 |
| LOC | 4164 | 16740 | LOF | 4167 | 16743 |
| MKI\$ | 416A | 16746 | MKS\$ | 416D | 16749 |
| MKD\$ | 4170 | 16752 | CINT | 0A7F | 2687 |
| CSNG | 0AB1 | 2737 | CDBL | 0ADB | 2779 |
| FIX | 0B26 | 2854 | LEN | 2A03 | 10755 |
| STR\$ | 2836 | 10294 | VAL | 2AC5 | 10949 |
| ASC | 2A0F | 10767 | CHR\$ | 2A1F | 10783 |
| LEFT\$ | 2A61 | 10849 | RIGHT\$ | 2A91 | 10897 |
| MID\$ | 2A9A | 10906 | | | |

- CHAPTER 2 -

INTEGER, SINGLE, AND DOUBLE PRECISION ARITHMETIC

INTRODUCTION:

After one has recovered from the shock of learning the fundamentals of assembly language programming it is ridiculous to "re-invent the wheel" by writing dozens of lines or pages of source code to perform simple single and double precision arithmetic calculations when these routines already exist in Level II ROM and may be accessed with a single call.

Assembly language programming with its resulting source code programs running 300+ times faster than BASIC and requiring on the average only 1/10th as much memory to perform the same functions as BASIC is really the ne plus ultra for the serious amateur programmer who wishes to advance beyond the inherent limitations of BASIC, Fortran, Cobol, Pascal or any of the high level computer languages. Prior to the "TRS-80 Disassembled Handbook," would be assembly language programmers were forced to learn by rote those assembly language subroutines for ALL the functions that were already extant in the Level II ROM because no one had ever figured out exactly how to access all these subroutines; i.e., break the beautifully "TIGHT CODE" code written by Microsoft's Paul Allen and Bill Gates.

Would be assembly programmers arise. The Level II ROM code has now been broken. As every cryptographer knows, every lock has a key. It is just that some locks take a bit longer to pick than others, (ask N.S.A. or MI.5 about this). For some perverse reason, (probably money and the Chinese secrecy syndrome), neither Radio Shack or Microsoft have been willing to come forward and tell the 200,000+ TRS-80 users how to use the myriad Level II ROM subroutines in assembly language programs. This point is best illustrated by Radio Shack's book, "TRS-80 Assembly Language Programming," introduced in mid-1979 which either for stupidity or duplicity or both leads the would be assembly language programmer into T-Bug (surely the height of backwardness/retrogression), and then goes on with multi-line demonstration programs covering keyboard scan, video display, fill, move, muladd, mulsub, compare, mul16, div16, etc. that could be accomplished with only a few lines of assembly language programming IF the extant Level II ROM subroutines had been used. Let us be kind though, and presume that Radio Shack had not the slightest idea what Level II ROM contained, and if they did, had not the slightest idea on how to find it and use it. If such is not the case, they surely stand guilty of gross negligence and malice aforethought before the 200,000+ TRS-80 user community. If you have mastered Level II BASIC, and can at least read and write the English language at the 5th grade level, you should have great fun with this totally NEW approach to assembly language programming. By mastering Level II BASIC you have demonstrated that you have the skills and persistence to become an advanced

assembly language programmer with only a few weeks study rather than what heretofore took many months or years. The supposed "experts" in the field of assembly language programming have created an aura and mystique about the subject which is totally undeserved and seeks only to promote their own self esteem. "Bull roar," as the philosopher said. Let us take a brief look at how very simple assembly language programming can be by illustrating our point with a few simple arithmetic programs that almost exclusively use Level II ROM subroutines.

FUNDAMENTALS OF LEVEL II ROM ARITHMETIC:

ROM arithmetic subroutines are virtually identical to those you would HAVE TO write were they not NOW available to the assembly language programmer. This is true whether we are discussing integer, single precision, double precision, addition, subtraction, multiplication, division, as well as ALL the trigonometric, exponential, and log functions too. In fact, it is true for all Level II ROM functions which are nothing more than binary bytes we may manipulate as long as we know where they are located. Let's get on with the primer for + - * and /.

ROM (read-only-memory) means just what it says, READ-ONLY. Since it may be only read from, Level II ROM uses the RAM memory from 14302 to 17129 for all its housekeeping chores. The keyboard, from 14336 to 15360 is not really RAM at all, but a simple key/switch matrix which the rest of the system thinks is RAM. Video memory occupies memory locations 15360 to 16383. Except for memory locations 14302 to 14336, all the non-disk Level II RAM housekeeping chores are done between 16384 and 17129. Three RAM memory locations are of particular interest while discussing arithmetic + - * / subroutines. They are the ACCUMulator, CDBL store (or "CS" abbreviation), and NT (number type). Arithmetic numbers stashed in RAM use the following conventions: integer = LSB first and MSB second using two's complement format, and single and double precision numbers = normalized exponential format with 129 added to the exponent and the high bit of the MSB reflecting the + or - sign of the number. Do not concern yourself with these number formats as our Level II ROM will handle all the conversions necessary if we use them properly. The ACCUM occupies memory locations 411DH through 4124H (8 bytes) and CDBL store occupies 4127H through 412EH, also 8 bytes. We must concern ourselves with NT (number type) as it will "blow" our whole subroutine if we try to perform arithmetic operations with dissimilar number types; i.e., add an integer to a double precision number, etc. Do not fret though, ROM lets us use its CINT, CSGN, CDBL functions with only a single CALL to make the numbers we are using compatible.

CINT = CALL 0A7FH CSGN = CALL 0AB1H CDBL = CALL 0ADBH

The 3 programs in this chapter provide these functions in each routine, so it takes real effort to foul them up if you abide by each number types minimal rules.

The NT (number type) single byte storage in RAM is located at 40AFH. NT (40AFH) = 2 for an integer number, = 3 for a string, = 4 for a single precision number, and = 8 for a double precision number. To change these numbers to ASCII and display them on video, simply ADD 30H to the contents of MEM location 40AFH and output to the video display as follows:

```
LD      A, (40AFH)      ;NT location
ADD     A,30H           ;convert to ASCII
CALL    032AH           ;display on video
```

INTEGER ARITHMETIC + - * / :

Fig. 7 is the source code and Fig. 8 is the object code of the demonstration program that will allow you to add, subtract, multiply, or divide integers strictly using the ROM subroutines. Fast? You bet it is. As soon as you press ENTER you'll have the answer. Remember, your Model I TRS-80 with its clock running at approximately 1,774,000 cycles per second is no slow poke. Instead of BASIC's "slowsville," you are now conversing with your Z-80 microprocessor directly, IN ITS OWN LANGUAGE. With no interpreter (BASIC) required, it will zap along at what appears to be the speed of light. All this integer program does is to place the first number you input into the DE register, the second number you input into the HL register, and then CALL whatever + - * / operation you requested. This simple program is completely straightforward except for line 330's PUSH HL and line 400's POP DE. The stack begins at RAM memory location 4288H when operating in the SYSTEM mode. What we are doing here is "saving" the first integer number in the stack by PUSHing HL in line 330. The program then uses the HL register in obtaining the second number you input in line 340. The POP DE in line 400 merely takes the previous HL value from the stack and places it into the DE register. The stack could care less where its contents go as it is just a sophisticated FILO (first-in-last-out) memory created and controlled by your Z-80/ROM (unless you choose to modify its location with the LD SP (stack pointer) opcode and operand instruction. Remember, integer arithmetic is nothing more than placing the 'F'irst number in the DE register, the 'S'econd number in the HL register, and specifying which + - * / operation you desire with the following CALLS:

```
ADD      = CALL 0BD2H      SUBTRACT = CALL 0BD7H
MULTIPLY = CALL 0BF2H      DIVIDE   = CALL 2490H
```

The result of any of these operations is always placed in the ACCUM. To display the result on video, merely:

```
CALL     0FBDH           ;convert ACCUM to ASCII string
CALL     28A7H           ;display ASCII string on video
```

Do not forget the Opcode, 'ADD HL,(register pair)' for integer adds up to 65535. Integers are simplesville, indeed.

SINGLE PRECISION ARITHMETIC + - * / :

Is very similar to integer arithmetic, except ROM now wants the 'F'irst number in registers BC and DE, and the 'S'econd number in the ACCUM. The desired operation is performed by:

| | | | |
|----------|--------------|----------|--------------|
| ADD | = CALL 0716H | SUBTRACT | = CALL 0713H |
| MULTIPLY | = CALL 0847H | DIVIDE | = CALL 08A2H |

For memory storage, we again use the stack as shown in lines 340 & 350 PUSH instructions, and lines 420 & 430 POP same. Figs. 9 & 10 are the source code and object code for the single precision arithmetic demonstration program.

DOUBLE PRECISION ARITHMETIC + - * / :

Is not significantly different from either integer or single precision arithmetic subroutines, except now ROM wants the 'F'irst number in the ACCUM and the 'S'econd number in the CDBL store RAM location. Desired operation is performed by:

| | | | |
|----------|--------------|----------|--------------|
| ADD | = CALL 077CH | SUBTRACT | = CALL 0C70H |
| MULTIPLY | = CALL 0DA1H | DIVIDE | = CALL 0DE5H |

Figs. 11 & 12 are the source and object code for the double precision arithmetic demonstration program.

SUMMARY:

Each of these source code programs may be input by the user in about 5 minutes time using the EXCELLENT Radio Shack Editor/Assembler that was written by Zilog's President and in-house-genius, Federico Faggin, and his staff. Previous assembly language teaching programs required months of study and page upon page of source code to accomplish all the double precision arithmetic routines. Now the average TRS-80 buff can master the subject in a matter of hours.


```

00100 ;
00110
00120 ;
00130
00140 W4UCH EQU 7D00H
00150 ORG W4UCH
00160 BEGIN LD A,4FH
00170 CALL 032AH
00180 LD A,3FH
00190 CALL 032AH
00200 LD A,20H
00210 CALL 032AH
00220 CALL 049H
00230 CALL 032AH
00240 LD (FUNCT),A
00250 LD A,0DH
00260 CALL 032AH
00270 LD A,46H
00280 CALL 032AH
00290 CALL 1BB3H
00300 RST 10H
00310 CALL 0E6CH
00320 CALL 0A7FH
00330 PUSH HL
00340 LD A,53H
00350 CALL 032AH
00360 CALL 1BB3H
00370 RST 10H
00380 CALL 0E6CH
00390 CALL 0A7FH
00400 POP DE
00410 LD A,(FUNCT)
00420 CP 2BH
00430 JR Z,ADD
00440 CP 2DH
00450 JR Z,SUB
00460 CP 2AH
00470 JR Z,MULT
00480 CP 2FH
00490 JR Z,DIVIDE
00500 VIDEO LD A,3DH
00510 CALL 032AH
00520 LD A,20H
00530 CALL 032AH
00540 CALL 0FBDH
00550 CALL 28A7H
00560 LD A,0DH
00570 CALL 032AH
00580 JR BEGIN
00590 ADD CALL 0BD2H
00600 JR VIDEO
00610 SUB CALL 0BC7H
00620 JR VIDEO
00630 MULT CALL 0BF2H
00640 JR VIDEO
00650 DIVIDE CALL 2490H
00660 JR VIDEO
00670 FUNCT DEFB
00680 END W4UCH

```

```

;= 32000 DECIMAL
;PROGRAM WILL START HERE
;"O" OPERATION DESIRED
;DISPLAY "O" ON VIDEO
;= ASCII ?
;DO IT - ON VIDEO
;= ASCII SPACE
;DO IT - ON VIDEO
;KYBD INPUT + - * /
;DISPLAY FUNCTION
;STASH DESIRED OPERATION
;0DH = SKIP A LINE
;DO IT - ON VIDEO
;"F" = FIRST NUMBER
;DO IT - ON VIDEO
;KYBD/VIDEO INPUT ROUTINE
;SCAN $ SET "C" FLAG
;ASCII-ACCUM RET MIN
;CONVERT TO INTEGER
;SAVE INTEGER IN STACK
;"S" = 2ND NUMBER
;DISPLAY "S" ON VIDEO
;KYBD/VIDEO INPUT ROUTINE
;SCAN $ SET "C" FLAG
;ASCII$ TO ACCUM RET MIN
;CONVERT TO INTEGER
;PREVIOUS HL TO DE REG
;RECALL + - * / FROM MEM
;IS IT + ?
;IF SO GOTO ADD
;IS IT - ?
;IF SO GOTO SUBTRACT
;IS IT * ?
;IF SO GOTO MULTIPLY
;IS IT / ?
;IF SO GOTO DIVIDE
;3DH IS ASCII = SIGN
;DO IT - ON VIDEO
;= ASCII SPACE
;DO IT - ON VIDEO
;CONV ACCUM TO STRING
;DISPLAY STRING ON VIDEO
;= SKIP A LINE
;DO IT - ON VIDEO
;REPEAT ROUTINE
;ADD DE + HL
;OUTPUT RESULT
;SUBTRACT DE - HL
;OUTPUT RESULT
;MULTIPLY DE * HL
;OUTPUT RESULT
;DIVIDE DE / HL
;OUTPUT RESULT
;SAVE BYTE-STASH FUNCTION
;AMATEUR RADIO CALL LTRS

```

| | | | | |
|------|-------|--------|------|-----------|
| | 00130 | | | |
| 7D00 | 00140 | W4UCH | EQU | 7D00H |
| 7D00 | 00150 | | ORG | W4UCH |
| 7D00 | 00160 | BEGIN | LD | A,4FH |
| 7D02 | 00170 | | CALL | 032AH |
| 7D05 | 00180 | | LD | A,3FH |
| 7D07 | 00190 | | CALL | 032AH |
| 7D0A | 00200 | | LD | A,20H |
| 7D0C | 00210 | | CALL | 032AH |
| 7D0F | 00220 | | CALL | 049H |
| 7D12 | 00230 | | CALL | 032AH |
| 7D15 | 00240 | | LD | (FUNCT),A |
| 7D18 | 00250 | | LD | A,0DH |
| 7D1A | 00260 | | CALL | 032AH |
| 7D1D | 00270 | | LD | A,46H |
| 7D1F | 00280 | | CALL | 032AH |
| 7D22 | 00290 | | CALL | 1BB3H |
| 7D25 | 00300 | | RST | 10H |
| 7D26 | 00310 | | CALL | 0E6CH |
| 7D29 | 00320 | | CALL | 0A7FH |
| 7D2C | 00330 | | PUSH | HL |
| 7D2D | 00340 | | LD | A,53H |
| 7D2F | 00350 | | CALL | 032AH |
| 7D32 | 00360 | | CALL | 1BB3H |
| 7D35 | 00370 | | RST | 10H |
| 7D36 | 00380 | | CALL | 0E6CH |
| 7D39 | 00390 | | CALL | 0A7FH |
| 7D3C | 00400 | | POP | DE |
| 7D3D | 00410 | | LD | A,(FUNCT) |
| 7D40 | 00420 | | CP | 2BH |
| 7D42 | 00430 | | JR | Z,ADD |
| 7D44 | 00440 | | CP | 2DH |
| 7D46 | 00450 | | JR | Z,SUB |
| 7D48 | 00460 | | CP | 2AH |
| 7D4A | 00470 | | JR | Z,MULT |
| 7D4C | 00480 | | CP | 2FH |
| 7D4E | 00490 | | JR | Z,DIVIDE |
| 7D50 | 00500 | VIDEO | LD | A,3DH |
| 7D52 | 00510 | | CALL | 032AH |
| 7D55 | 00520 | | LD | A,20H |
| 7D57 | 00530 | | CALL | 032AH |
| 7D5A | 00540 | | CALL | 0FBDH |
| 7D5D | 00550 | | CALL | 28A7H |
| 7D60 | 00560 | | LD | A,0DH |
| 7D62 | 00570 | | CALL | 032AH |
| 7D65 | 00580 | | JR | BEGIN |
| 7D67 | 00590 | ADD | CALL | 0BD2H |
| 7D6A | 00600 | | JR | VIDEO |
| 7D6C | 00610 | SUB | CALL | 0BC7H |
| 7D6F | 00620 | | JR | VIDEO |
| 7D71 | 00630 | MULT | CALL | 0BF2H |
| 7D74 | 00640 | | JR | VIDEO |
| 7D76 | 00650 | DIVIDE | CALL | 2490H |
| 7D79 | 00660 | | JR | VIDEO |
| 7D7B | 00670 | FUNCT | DEFB | 0 |
| 7D00 | 00680 | | END | W4UCH |

00110

00120 ; USING LEVEL II ROM SUBROUTINES + - * / BY W4UCH

00130

```

00140 W4UCH EQU 7D00H ;= 32000 DECIMAL
00150 ORG W4UCH ;PROGRAM WILL START HERE
00160 BEGIN LD A,4FH ;"O" OPERATION DESIRED
00170 CALL 032AH ;DISPLAY "O" ON VIDEO
00180 LD A,3FH ;= ASCII ?
00190 CALL 032AH ;DO IT - ON VIDEO
00200 LD A,20H ;= ASCII SPACE
00210 CALL 032AH ;DO IT - ON VIDEO
00220 CALL 049H ;KYBD INPUT + - * /
00230 CALL 032AH ;DISPLAY FUNCTION
00240 LD (FUNCT),A ;STASH DESIRED OPERATION
00250 LD A,0DH ;0DH = SKIP A LINE
00260 CALL 032AH ;DO IT - ON VIDEO
00270 LD A,46H ;"F" = FIRST NUMBER
00280 CALL 032AH ;DO IT - ON VIDEO
00290 CALL 1BB3H ;KYBD/VIDEO INPUT ROUTINE
00300 RST 10H ;SCAN $ SET "C" FLAG
00310 CALL 0E6CH ;ASCII-ACCUM RET MIN
00320 CALL 0AB1H ;CONV SINGLE PRECISION
00330 CALL 09BFH ;LOAD BCDE FROM ACCUM
00340 PUSH BC ;STORE IN STACK
00350 PUSH DE ;STORE IN STACK
00360 LD A,53H ;"S" = 2ND NUMBER
00370 CALL 032AH ;DISPLAY "S" ON VIDEO
00380 CALL 1BB3H ;KYBD/VIDEO INPUT ROUTINE
00390 RST 10H ;SCAN $ SET "C" FLAG
00400 CALL 0E6CH ;ASCII$ TO ACCUM RET MIN
00410 CALL 0AB1H ;CONV TO SINGLE PRECISION
00420 POP DE ;RESTORE DE REGISTER
00430 POP BC ;RESTORE BC REGISTER
00440 LD A,(FUNCT) ;RECALL + - * / FROM MEM
00450 CP 2BH ;IS IT + ?
00460 JR Z,ADD ;IF SO GOTO ADD
00470 CP 2DH ;IS IT - ?
00480 JR Z,SUB ;IF SO GOTO SUBTRACT
00490 CP 2AH ;IS IT * ?
00500 JR Z,MULT ;IF SO GOTO MULTIPLY
00510 CP 2FH ;IS IT / ?
00520 JR Z,DIVIDE ;IF SO GOTO DIVIDE
00530 VIDEO LD A,3DH ;3DH IS ASCII = SIGN
00540 CALL 032AH ;DO IT - ON VIDEO
00550 LD A,20H ;= ASCII SPACE
00560 CALL 032AH ;DO IT - ON VIDEO
00570 CALL 0FBDH ;CONV ACCUM TO STRING
00580 CALL 28A7H ;DISPLAY STRING ON VIDEO
00590 LD A,0DH ;= SKIP A LINE
00600 CALL 032AH ;DO IT - ON VIDEO
00610 JR BEGIN ;REPEAT ROUTINE
00620 ADD CALL 0716H ;ADD BCDE REGS TO ACCUM
00630 JR VIDEO ;OUTPUT RESULT
00640 SUB CALL 0713H ;SUB ACCUM FM BCDE REGS
00650 JR VIDEO ;OUTPUT RESULT
00660 MULT CALL 0847H ;MULT ACCUM * BCDE REGS
00670 JR VIDEO ;OUTPUT RESULT
00680 DIVIDE CALL 08A2H ;DIV ACCUM INTO BCDE REGS
00690 JR VIDEO ;OUTPUT RESULT
00700 FUNCT DEFB ;SAVE BYTE-STASH FUNCTION
00710 END W4UCH ;EL FIN = EL PRIMERO

```

| | | | | |
|------|--------|---------|-------------|-----------|
| | | 00130 ; | | |
| 7D00 | | 00140 | W4UCH EQU | 7D00H |
| 7D00 | | 00150 | ORG | W4UCH |
| 7D00 | 3E4F | 00160 | BEGIN LD | A,4FH |
| 7D02 | CD2A03 | 00170 | CALL | 032AH |
| 7D05 | 3E3F | 00180 | LD | A,3FH |
| 7D07 | CD2A03 | 00190 | CALL | 032AH |
| 7D0A | 3E20 | 00200 | LD | A,20H |
| 7D0C | CD2A03 | 00210 | CALL | 032AH |
| 7D0F | CD4900 | 00220 | CALL | 049H |
| 7D12 | CD2A03 | 00230 | CALL | 032AH |
| 7D15 | 32807D | 00240 | LD | (FUNCT),A |
| 7D18 | 3E0D | 00250 | LD | A,0DH |
| 7D1A | CD2A03 | 00260 | CALL | 032AH |
| 7D1D | 3E46 | 00270 | LD | A,46H |
| 7D1F | CD2A03 | 00280 | CALL | 032AH |
| 7D22 | CDB31B | 00290 | CALL | 1B73H |
| 7D25 | D7 | 00300 | RST | 10H |
| 7D26 | CD6C0E | 00310 | CALL | 0E6CH |
| 7D29 | CDB10A | 00320 | CALL | 0AB1H |
| 7D2C | CDBF09 | 00330 | CALL | 09BFH |
| 7D2F | C5 | 00340 | PUSH | BC |
| 7D30 | D5 | 00350 | PUSH | DE |
| 7D31 | 3E53 | 00360 | LD | A,53H |
| 7D33 | CD2A03 | 00370 | CALL | 032AH |
| 7D36 | CDB31B | 00380 | CALL | 1BB3H |
| 7D39 | D7 | 00390 | RST | 10H |
| 7D3A | CD6C0E | 00400 | CALL | 0E6CH |
| 7D3D | CDB10A | 00410 | CALL | 0AB1H |
| 7D40 | D1 | 00420 | POP | DE |
| 7D41 | C1 | 00430 | POP | BC |
| 7D42 | 3A807D | 00440 | LD | A,(FUNCT) |
| 7D45 | FE2B | 00450 | CP | 2BH |
| 7D47 | 2823 | 00460 | JR | Z,ADD |
| 7D49 | FE2D | 00470 | CP | 2DH |
| 7D4B | 2824 | 00480 | JR | Z,SUB |
| 7D4D | FE2A | 00490 | CP | 2AH |
| 7D4F | 2825 | 00500 | JR | Z,MULT |
| 7D51 | FE2F | 00510 | CP | 2FH |
| 7D53 | 2826 | 00520 | JR | Z,DIVIDE |
| 7D55 | 3E3D | 00530 | VIDEO LD | A,3DH |
| 7D57 | CD2A03 | 00540 | CALL | 032AH |
| 7D5A | 3E20 | 00550 | LD | A,20H |
| 7D5C | CD2A03 | 00560 | CALL | 032AH |
| 7D5F | CDBD0F | 00570 | CALL | 0FBDH |
| 7D62 | CDA728 | 00580 | CALL | 28A7H |
| 7D65 | 3E0D | 00590 | LD | A,0DH |
| 7D67 | CD2A03 | 00600 | CALL | 032AH |
| 7D6A | 1894 | 00610 | JR | BEGIN |
| 7D6C | CD1607 | 00620 | ADD CALL | 0716H |
| 7D6F | 18E4 | 00630 | JR | VIDEO |
| 7D71 | CD1307 | 00640 | SUB CALL | 0713H |
| 7D74 | 18DF | 00650 | JR | VIDEO |
| 7D76 | CD4708 | 00660 | MULT CALL | 0847H |
| 7D79 | 18DA | 00670 | JR | VIDEO |
| 7D7B | CDA208 | 00680 | DIVIDE CALL | 08A2H |
| 7D7E | 18D5 | 00690 | JR | VIDEO |
| 7D80 | 00 | 00700 | FUNCT DEFB | 0 |
| 7D00 | | 00710 | END | W4UCH |

```

00100 ;      DOUBLE PRECISION DEMONSTRATION PROGRAM
00110
00120 ;      USING LEVEL II ROM SUBROUTINES + - * /
00130
00140 W4UCH EQU      7D00H      ;= 32000 DECIMAL
00150      ORG      W4UCH      ;PROGRAM WILL START HERE
00160 BEGIN LD      A,4FH      ;"O" OPERATION DESIRED
00170      CALL     032AH      ;DISPLAY "O" ON VIDEO
00180      LD      A,3FH      ;= ASCII ?
00190      CALL     032AH      ;DO IT - ON VIDEO
00200      LD      A,20H      ;= ASCII SPACE
00210      CALL     032AH      ;DO IT - ON VIDEO
00220      CALL     049H      ;KYBD INPUT + - * /
00230      CALL     032AH      ;DISPLAY FUNCTION
00240      LD      (FUNCT),A    ;STASH DESIRED OPERATION
00250      LD      A,0DH      ;0DH = SKIP A LINE
00260      CALL     032AH      ;DO IT - ON VIDEO
00270      LD      A,46H      ;"F" = FIRST NUMBER
00280      CALL     032AH      ;DO IT - ON VIDEO
00290      CALL     1BB3H      ;KYBD/VIDEO INPUT ROUTINE
00300      RST      10H      ;SCAN $ SET "C" FLAG
00310      CALL     0E65H      ;ASCII$ TO ACCUM RET CDBL
00320      LD      DE,411DH    ;MOVE FROM ACCUM RAM MEM
00330      LD      HL,TACCUM   ;TO TEMPORARY ACCUM STASH
00340      LD      B,8        ;NUMBER OF BYTES TO MOVE
00350      CALL     09D7H      ;MOVE IT - SUBROUTINE
00360      LD      A,53H      ;"S" = 2ND NUMBER
00370      CALL     032AH      ;DISPLAY "S" ON VIDEO
00380      CALL     1BB3H      ;KYBD/VIDEO INPUT ROUTINE
00390      RST      10H      ;SCAN $ SET "C" FLAG
00400      CALL     0E65H      ;ASCII$ TO ACCUM RET CDBL
00410      CALL     09FCH      ;TRANSFER ACCUM TO CDBL
00420      LD      DE,TACCUM   ;MOVE ACCUM FROM STASH TO
00430      LD      HL,411DH    ;PERMANENT RAM LOCATION
00440      LD      B,8        ;NUMBER OF BYTES TO MOVE
00450      CALL     09D7H      ;MOVE IT - RIGHT NOW
00460      LD      A,(FUNCT)   ;RECALL + - * / FROM MEM
00470      CP      2BH      ;IS IT + ?
00480      JR      Z,ADD      ;IF SO GOTO ADD
00490      CP      2DH      ;IS IT - ?
00500      JR      Z,SUB      ;IF SO GOTO SUBTRACT
00510      CP      2AH      ;IS IT * ?
00520      JR      Z,MULT     ;IF SO GOTO MULTIPLY
00530      CP      2FH      ;IS IT / ?
00540      JR      Z,DIVIDE   ;IF SO GOTO DIVIDE
00550 VIDEO LD      A,3DH      ;3DH IS ASCII = SIGN
00560      CALL     032AH      ;DO IT - ON VIDEO
00570      LD      A,20H      ;= ASCII SPACE
00580      CALL     032AH      ;DO IT - ON VIDEO
00590      CALL     0FBDH      ;CONV ACCUM TO STRING
00600      CALL     28A7H      ;DISPLAY STRING ON VIDEO
00610      LD      A,0DH      ;= SKIP A LINE
00620      CALL     032AH      ;DO IT - ON VIDEO
00630      JR      BEGIN     ;REPEAT ROUTINE
00640 ADD   CALL     0C77H      ;ADD ACCUM TO CDBL
00650      JR      VIDEO     ;OUTPUT RESULT
00660 SUB   CALL     0C70H      ;SUBTRACT CDBL FROM ACCUM
00670      JR      VIDEO     ;OUTPUT RESULT
00680 MULT CALL     0DA1H      ;MULTIPLY ACCUM * CDBL
00690      JR      VIDEO     ;OUTPUT RESULT
00700 DIVIDE CALL     0DE5H     ;DIVIDE ACCUM BY CDBL
00710      JR      VIDEO     ;OUTPUT RESULT
00720 FUNCT DEFB      ;SAVE BYTE-STASH FUNCTION
00730 TACCUM DEFS      8      ;TEMPORARY ACCUM STASH
00740      END      W4UCH     ;AMATEUR RADIO CALL LTRS

```

| | | | |
|-------------|-------|-------------------------------|-----------|
| | 00120 | ;DOUBLE PRECISION - PAGE 22 - | |
| | 00130 | ; | |
| 7D00 | 00140 | W4UCH EQU | 7D00H |
| 7D00 | 00150 | ORG | W4UCH |
| 7D00 3E4F | 00160 | BEGIN LD | A,4FH |
| 7D02 CD2A03 | 00170 | CALL | 032AH |
| 7D05 3E3F | 00180 | LD | A,3FH |
| 7D07 CD2A03 | 00190 | CALL | 032AH |
| 7D0A 3E20 | 00200 | LD | A,20H |
| 7D0C CD2A03 | 00210 | CALL | 032AH |
| 7D0F CD4900 | 00220 | CALL | 049H |
| 7D12 CD2A03 | 00230 | CALL | 032AH |
| 7D15 328C7D | 00240 | LD | (FUNCT),A |
| 7D18 3E0D | 00250 | LD | A,0DH |
| 7D1A CD2A03 | 00260 | CALL | 032AH |
| 7D1D 3E46 | 00270 | LD | A,46H |
| 7D1F CD2A03 | 00280 | CALL | 032AH |
| 7D22 CDB31B | 00290 | CALL | 1BB3H |
| 7D25 D7 | 00300 | RST | 10H |
| 7D26 CD650E | 00310 | CALL | 0E65H |
| 7D29 111D41 | 00320 | LD | DE,411DH |
| 7D2C 218D7D | 00330 | LD | HL,TACCUM |
| 7D2F 0608 | 00340 | LD | B,8 |
| 7D31 CDD709 | 00350 | CALL | 09D7H |
| 7D34 3E53 | 00360 | LD | A,53H |
| 7D36 CD2A03 | 00370 | CALL | 032AH |
| 7D39 CDB31B | 00380 | CALL | 1BB3H |
| 7D3C D7 | 00390 | RST | 10H |
| 7D3D CD650E | 00400 | CALL | 0E65H |
| 7D40 CDFC09 | 00410 | CALL | 09FCH |
| 7D43 118D7D | 00420 | LD | DE,TACCUM |
| 7D46 211D41 | 00430 | LD | HL,411DH |
| 7D49 0608 | 00440 | LD | B,8 |
| 7D4B CDD709 | 00450 | CALL | 09D7H |
| 7D4E 3A8C7D | 00460 | LD | A,(FUNCT) |
| 7D51 FE2B | 00470 | CP | 2BH |
| 7D53 2823 | 00480 | JR | Z,ADD |
| 7D55 FE2D | 00490 | CP | 2DH |
| 7D57 2824 | 00500 | JR | Z,SUB |
| 7D59 FE2A | 00510 | CP | 2AH |
| 7D5B 2825 | 00520 | JR | Z,MULT |
| 7D5D FE2F | 00530 | CP | 2FH |
| 7D5F 2826 | 00540 | JR | Z,DIVIDE |
| 7D61 3E3D | 00550 | LD | A,3DH |
| 7D63 CD2A03 | 00560 | CALL | 032AH |
| 7D66 3E20 | 00570 | LD | A,20H |
| 7D68 CD2A03 | 00580 | CALL | 032AH |
| 7D6B CDBD0F | 00590 | CALL | 0FBDH |
| 7D6E CDA728 | 00600 | CALL | 28A7H |
| 7D71 3E0D | 00610 | LD | A,0DH |
| 7D73 CD2A03 | 00620 | CALL | 032AH |
| 7D76 1888 | 00630 | JR | BEGIN |
| 7D78 CD770C | 00640 | CALL | 0C77H |
| 7D7B 18E4 | 00650 | JR | VIDEO |
| 7D7D CD700C | 00660 | CALL | 0C70H |
| 7D80 18DF | 00670 | JR | VIDEO |
| 7D82 CDA10D | 00680 | CALL | 0DA1H |
| 7D85 18DA | 00690 | JR | VIDEO |
| 7D87 CDE50D | 00700 | CALL | 0DE5H |
| 7D8A 18D5 | 00710 | JR | VIDEO |
| 7D8C 00 | 00720 | FUNCT | 0 |
| 0008 | 00730 | TACCUM | 8 |
| 7D00 | 00740 | END | W4UCH |

- Fig. 12 -

- CHAPTER 3 -

USING LEVEL II ROM SUBROUTINES
IN ADVANCED ASSEMBLY LANGUAGE PROGRAMMING

- TRIGONOMETRIC, LOG, EXPONENT, ET AL FUNCTIONS -

(notes from a lecture)

Here is an interesting test program for the advanced assembly language programmer. It allows the user to access and test many of the myriad arithmetic/trigonometric subroutines that are extant in the excellent TRS-80 Level II ROM that was written by Microsoft's Bill Gates and Paul Allen.

The beginning assembly language programmer should certainly be taught and learn the how, why, and wherefores of writing fundamental arithmetic/trig functions by him/her self, but once these techniques have been mastered as part of the learning process, it is certainly inefficient, time wasting, and rather ridiculous to re-invent the wheel by duplicating in assembly language those subroutines already extant in the Level II ROM.

Table 1 lists those functions and their addresses that may be accessed and tested by this mini-program that only occupies 144 bytes of high memory and may be entered using the TRS-80 Editor/Assembler in about 5 minutes.

Figure 13 is a print-out of the test program's source code and Figure 14 a listing of the program's object code. As may be easily seen, the majority of this program is written using Level II ROM subroutines. Were these subroutines not used in this particular assembly language test program, it would require approximately 10 times as much program memory and occupy 550 rather than 55 assembly language program lines.

PROGRAM FLOW:

The comments included with the source code program delineate each line's function. There is no need to duplicate or expand upon the comments here as they are largely self-explanatory. This program operates equally well with non-disk Level II, DOS 2.1, DOS 2.2 and NEWDOS+. Program operation is as follows:

1. Load the program under the SYSTEM or DOS command. Give it any name you wish. We like the program name DISCOV, for discovery, since that is what the program is all about. After loading is complete, type in /32000 to activate the program (with disk you must first load BASIC, then type SYSTEM, ENTER, and then type in /32000 ENTER, if you loaded the program in DOS).

2. The letter 'N' ? will appear on video. The program is asking you for a number to work on. Any number up to 16 digits is alright depending on the function you wish to test. Let us start out with a simple example by entering the number 10000, a nice round easily visualized number, ENTER.
3. The numbers '2' '10000' will appear on the next line of the video display. The '2' is the number 'type' brilliantly calculated by the Level II ROM. Since we are dealing only with numbers in this Chapter we will blithely skip over strings et al for the time being. The number types are as follows: 2 = integer, 4 = single precision, and 8 = double precision. Table 1 lists those operations that can be performed on a number for a given number type; i.e., it is against the rules to take the square root SQR of an integer. We must first change to single precision.
4. On the following line of the video display you will see 'C ? '. The program is asking you what type of CONVERSION you wish. Let's enter 2737 which = CSNG, change our number from an integer to single precision, ENTER. The next line will show, '4' '1000'. We now have a single precision number to work with, so let's now try taking its square root by typing in 5095 = SQR, then ENTER. ZAP... ..the next line shows '100'. Ah, the miracle of modern computer science at work. It sure was easier than writing a complete stand alone assembly language square root subroutine. Let's try it again. Type in 5095 ENTER. Again the line below displays the square root. This time the numeral 10.
5. Stick around as this is only the beginning. To insert a new number to try your program on merely type in 32000 ENTER. This brilliantly brings us back to where we started by displaying 'N ?'. Is 32000 a subroutine? Sure it is. You wrote it. Our assembly language program does not discriminate between ROM or RAM. It could care less.
6. We could go on and on converting numbers like deriving the natural LOG of any number and then restoring it with the EXP command, and/or deriving the TANGent of a number, then its arc tangent ATN, and then the TANgent againad infinitum. You may escape this conversion routine any time your wish by typing 6681 ENTER which will take you back to BASIC with a READY displayed. All you need do to return to your conversion routine is type SYSTEM then ENTER and type /32000 then ENTER.

This lecture covers only a few of the subroutines extant in Level II BASIC ROM that are illustrated in Table 1.

Assembly language programming is the ne plus ultra of serious computing and really the Mt. Everest of our hobby. You master it and you climb it because it is there. The fallout of having any assembly language program run 300 times faster than the same program in BASIC, and the extra plus of only using 1/10th as much memory as the same program in BASIC are really the topping/icing of the cake.

Learning to talk to your computer in its own language rather than through an intrepter (BASIC, Fortran, Pascal, or what-have-you), is probably one of the most satisfying and rewarding experiences you will ever have if you have the patience and fortitude to master it.

ADDENDUM:

The demonstration program illustrated in Figure 13 will easily perform many more functions than the short list covered in Table 1. Make a note to come back to this program after you have finished Chapter 4 and have become familiar with data movement and data conversion subroutines.

Most all of the arithmetic + - * / subroutines using integer, single precision, and double precision numbers may be used by judiciously storing one number in "CS", the CDBL Store. -CALL 2556 decimal = 09FCH moves data from the ACCUM to "CS". The next number is then input by loading "32000" into CONV ? and then entering this number in the ACCUM.

By keeping close track of the NT (number type) so you call the appropriate arithmetic/conversion subroutine, and using the data movement subroutines covered in the next chapter, it is quite easy for this demo program to calculate LOG to the base 10, manipulate trig functions as desired, etc. Though you will never write a real-time program such as that given in Figure 13, it nevertheless offers you an excellent opportunity to practice and become familiar with Chapter 4's data conversion and data movement subroutines. Besides, it is an intellectual challenge....and challenges are fun when you WIN.

note: number types 2 = integer 4 = single precision
8 = double precision

| FUNCTION | NUMBER TYPE | DECIMAL | HEXADECIMAL |
|------------------|-----------------|---------|-------------|
| ABS | 2-4-8 | 2423 | 0977 |
| ATN | 4-8 | 5565 | 15BD |
| BASIC | (RETURN L II) | 6681 | 1A19 |
| BASIC | (RETURN DISK) | 112 | 0075 |
| BREAK | (RST ADDRESS) | 16396 | 400C |
| CDBL | 2-4 | 2779 | 0ADB |
| CINT | 4-8 | 2687 | 0A7F |
| CLS | 2-4-8 | 457 | 01C9 |
| COS | 4-8 | 5441 | 1541 |
| CSNG | 2-8 | 2737 | 0AB1 |
| EXP | 4-8 | 5177 | 1439 |
| FIX | 2-4 | 2854 | 0B26 |
| INT | 2 | 2871 | 0B37 |
| INVERT SIGN | 2 | 3153 | 0C51 |
| INVERT SIGN | 4-8 | 2434 | 0982 |
| LOG | 4-8 | 2057 | 0809 |
| MEMORY | (DEFINE SIZE) | 181 | 00B5 |
| RANDOM | 2-4-8 | 467 | 01D3 |
| RETURN | (TO SUBROUTINE) | 32000 | 7D00 |
| RND (see limits) | 2-4-8 | 5321 | 14C9 |
| SGN | 2 | 2442 | 098A |
| SIN | 4-8 | 5447 | 1547 |
| SQR | 4-8 | 5095 | 13E7 |
| TAN | 4-8 | 5544 | 15A8 |

| | | | | |
|-------|--------|------|-----------|---------------------------------|
| 00100 | W4UCH | EQU | 7D00H | ;7D00H = 32000 DECIMAL -PAGE 27 |
| 00110 | | ORG | W4UCH | ;PROGRAM WILL START HERE |
| 00120 | | LD | A,4EH | ;4EH="N"=NUMBER DESIRED ? |
| 00130 | | CALL | 032AH | ;DISPLAY "N" ON VIDEO |
| 00140 | | CALL | 1BB3H | ;KYBD/VIDEO INPUT ROUTINE |
| 00150 | | RST | 10H | ;SCAN STRING - SET C FLAG |
| 00160 | | CALL | 0E6CH | ;ASCII-ACCUM RET MINIMUM |
| 00170 | RETURN | EX | AF,AF' | ;EXCHANGE REGISTERS- |
| 00180 | | EXX | | ;TO PRESERVE VALUES. |
| 00190 | | LD | DE,411DH | ;MOVE MEM ACCUM DATA FROM |
| 00200 | | LD | HL,STORE | ;TO TEMPORARY STASH. |
| 00210 | | LD | B,8 | ;NUMBER OF BYTES TO MOVE |
| 00220 | | CALL | 09D7H | ;MOVE IT - SUBROUTINE |
| 00230 | | LD | DE,4127H | ;MOVE CDBL DATA FROM- |
| 00240 | | LD | HL,CDBL | ;TO TEMPORARY STASH. |
| 00250 | | LD | B,8 | ;NUMBER OF BYTES TO MOVE |
| 00260 | | CALL | 09D7H | ;MOVE IT - SUBROUTINE |
| 00270 | | LD | A,(40AFH) | ;NUMBER TYPE MEM LOCATION |
| 00280 | | LD | (FLAG),A | ;MOVE TO TEMPORARY STASH |
| 00290 | | ADD | A,48 | ;CONVERT TO ASCII NUMBER |
| 00300 | | CALL | 032AH | ;DISPLAY NUMBER TYPE |
| 00310 | | LD | A,20H | ;20H = ASCII SPACE |
| 00320 | | CALL | 032AH | ;DISPLAY SPACE ON VIDEO |
| 00330 | | CALL | 0FBDH | ;CONV MEM ACCUM TO ASCII\$ |
| 00340 | | CALL | 28A7H | ;DISPLAY CONVERTED NUMBER |
| 00350 | | LD | A,0DH | ;0DH=SKIP A LINE/CARR RTN |
| 00360 | | CALL | 032H | ;DO IT - ON VIDEO DISPLAY |
| 00370 | | LD | A,43H | ;"C" = CONVERSION NUMBER? |
| 00380 | | CALL | 32AH | ;DISPLAY "C" ON VIDEO |
| 00390 | | CALL | 1BB3H | ;KYBD/VIDEO INPUT ROUTINE |
| 00400 | | RST | 10H | ;SCAN STRING - SET C FLAG |
| 00410 | | CALL | 0E6CH | ;ASCII-ACCUM RET MINIMUM |
| 00420 | | CALL | 0A7FH | ;CONVERT TO INTEGER |
| 00430 | | LD | (CONV),HL | ;STORE CONVERSION ADDRESS |
| 00440 | | LD | DE,CDBL | ;MOVE CDBL DATA FM STASH- |
| 00450 | | LD | HL,4127H | ;TO PERMANENT ADDRESS. |
| 00460 | | LD | B,8 | ;NUMBER OF BYTES TO MOVE |
| 00470 | | CALL | 09D7H | ;MOVE IT - SUBROUTINE |
| 00480 | | LD | DE,STORE | ;MOVE MEM ACCUM FM STASH- |
| 00490 | | LD | HL,411DH | ;TO PERMANENT ADDRESS. |
| 00500 | | LD | B,8 | ;NUMBER OF BYTES TO MOVE |
| 00510 | | CALL | 09D7H | ;MOVE IT - SUBROUTINE |
| 00520 | | LD | A,(FLAG) | ;NUMBER TYPE FROM STASH- |
| 00530 | | LD | (40AFH),A | ;TO PERMANENT ADDRESS. |
| 00540 | | LD | HL,RETURN | ;RETURN MEM LOCATION- |
| 00550 | | PUSH | HL | ;LOADED INTO STACK. |
| 00560 | | LD | HL,(CONV) | ;CONVERSION MEM LOCATION- |
| 00570 | | PUSH | HL | ;LOAD ON TOP OF STACK. |
| 00580 | | EX | AF,AF' | ;RESTORE REGISTERS- |
| 00590 | | EXX | | ;TO ORIGINAL VALUES. |
| 00600 | | RET | | ;SNEAKY CALL-TOP OF STACK |
| 00610 | FLAG | DEFS | 1 | ;NUMBER TYPE STASH |
| 00620 | CONV | DEFS | 2 | ;CONVERSION ADDRESS STASH |
| 00630 | CDBL | DEFS | 8 | ;CDBL DATA STASH |
| 00640 | STORE | DEFS | 8 | ;ACCUMULATOR STASH |
| 00650 | | END | W4UCH | ;AMATEUR RADIO CALL LTRS |

| | | | | |
|------|--------|-------|--------|-----------|
| 7D00 | 00100 | W4UCH | EQU | 7D00H |
| 7D00 | 00110 | | ORG | W4UCH |
| 7D00 | 3E4E | 00120 | LD | A,4EH |
| 7D02 | CD2A03 | 00130 | CALL | 032AH |
| 7D05 | CDB31B | 00140 | CALL | 1BB3H |
| 7D08 | D7 | 00150 | RST | 10H |
| 7D09 | CD6C0E | 00160 | CALL | 0E6CH |
| 7D0C | 08 | 00170 | RETURN | EX AF,AF' |
| 7D0D | D9 | 00180 | EXX | |
| 7D0E | 111D41 | 00190 | LD | DE,411DH |
| 7D11 | 21837D | 00200 | LD | HL,STORE |
| 7D14 | 0608 | 00210 | LD | B,8 |
| 7D16 | CDD709 | 00220 | CALL | 09D7H |
| 7D19 | 112741 | 00230 | LD | DE,4127H |
| 7D1C | 217B7D | 00240 | LD | HL,CDBL |
| 7D1F | 0608 | 00250 | LD | B,8 |
| 7D21 | CDD709 | 00260 | CALL | 09D7H |
| 7D24 | 3AAF40 | 00270 | LD | A,(40AFH) |
| 7D27 | 32787D | 00280 | LD | (FLAG),A |
| 7D2A | C630 | 00290 | ADD | A,48 |
| 7D2C | CD2A03 | 00300 | CALL | 032AH |
| 7D2F | 3E20 | 00310 | LD | A,20H |
| 7D31 | CD2A03 | 00320 | CALL | 032AH |
| 7D34 | CDBD0F | 00330 | CALL | 0FBDH |
| 7D37 | CDA728 | 00340 | CALL | 28A7H |
| 7D3A | 3E0D | 00350 | LD | A,0DH |
| 7D3C | CD3200 | 00360 | CALL | 032H |
| 7D3F | 3E43 | 00370 | LD | A,43H |
| 7D41 | CD2A03 | 00380 | CALL | 32AH |
| 7D44 | CDB31B | 00390 | CALL | 1BB3H |
| 7D47 | D7 | 00400 | RST | 10H |
| 7D48 | CD6C0E | 00410 | CALL | 0E6CH |
| 7D4B | CD7F0A | 00420 | CALL | 0A7FH |
| 7D4E | 22797D | 00430 | LD | (CONV),HL |
| 7D51 | 117B7D | 00440 | LD | DE,CDBL |
| 7D54 | 212741 | 00450 | LD | HL,4127H |
| 7D57 | 0608 | 00460 | LD | B,8 |
| 7D59 | CDD709 | 00470 | CALL | 09D7H |
| 7D5C | 11837D | 00480 | LD | DE,STORE |
| 7D5F | 211D41 | 00490 | LD | HL,411DH |
| 7D62 | 0608 | 00500 | LD | B,8 |
| 7D64 | CDD709 | 00510 | CALL | 09D7H |
| 7D67 | 3A787D | 00520 | LD | A,(FLAG) |
| 7D6A | 32AF40 | 00530 | LD | (40AFH),A |
| 7D6D | 210C7D | 00540 | LD | HL,RETURN |
| 7D70 | E5 | 00550 | PUSH | HL |
| 7D71 | 2A797D | 00560 | LD | HL,(CONV) |
| 7D74 | E5 | 00570 | PUSH | HL |
| 7D75 | 08 | 00580 | EX | AF,AF' |
| 7D76 | D9 | 00590 | EXX | |
| 7D77 | C9 | 00600 | RET | |
| 0001 | 00610 | FLAG | DEFS | 1 |
| 0002 | 00620 | CONV | DEFS | 2 |
| 0008 | 00630 | CDBL | DEFS | 8 |
| 0008 | 00640 | STORE | DEFS | 8 |
| 7D00 | 00650 | | END | W4UCH |

- CHAPTER 4 -

ANCILLARY LEVEL II ROM SUBROUTINES

INTRODUCTION:

Chapters 2 and 3 used a number of Level II ROM ancillary subroutines that were not fully explained except for a few words in the source code comment column. Using Level II ROM BASIC functions' CALL subroutines efficiently requires a modest understanding of how to use other ancillary ROM subroutines that are there just waiting to be used. They include: KEYBOARD input, MOVE data, COMPARE data, CONVERT data, VIDEO output, and LINE PRINTER output amongst others. This chapter will present the CALL addresses and briefly explain the most useful ROM ancillary subroutines that will truly make shorthand assembly language programming a reality for you.

KEYBOARD REVIEW:

Every advanced assembly language programmer knows that the keyboard is simply nothing more than a key/switch matrix that "looks like" RAM memory to Level II ROM. The keyboard's eight MEM locations are shown below in decimal. UA = up arrow, DA = down arrow, LA = left arrow and RA = right arrow.

| | | | | | | | | | |
|--------------|---|-------|-----|-----|----|----|----|----|-----|
| PEEK (14337) | = | @ | A | B | C | D | E | F | G |
| PEEK (14338) | = | H | I | J | K | L | M | N | O |
| PEEK (14340) | = | P | Q | R | S | T | U | V | W |
| PEEK (14344) | = | X | Y | Z | | | | | |
| PEEK (14352) | = | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| PEEK (14368) | = | 8 | 9 | : | ; | , | - | . | / |
| PEEK (14400) | = | ENT | CLR | BRK | UA | DA | LA | RA | SPA |
| PEEK (14464) | = | SHIFT | | | | | | | |

| | | | | | | | | | |
|-------|---|---|---|---|---|----|----|----|-----|
| VALUE | = | 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
|-------|---|---|---|---|---|----|----|----|-----|

The keyboard/switch matrix will output the VALUES shown above when a single key is pressed at the corresponding MEM location. For multiple keys pressed simultaneously, add up the values for each key; i.e., "JKL" = 4 + 8 + 16 = 28 total at MEM location 14338 decimal. With these facts in hand it is easy to see how very simply NEWDOS+ includes the feature of line printing out the video display contents whenever "JKL" are pressed simultaneously. Now, go write a brief assembly language program that will do so with non-disk Level II. For the inveterate experimenter, try this little 1 line program and press any combination of keys in the PEEK(14338) row:

```
10 X=PEEK(14338):PRINT@478,X:GOTO10
```

The 3 most useful Level II ROM ancillary subroutines for the keyboard follow. CALL locations are in hex.

CALL 002B: This is the most fundamental keyboard subroutine that scans the entire key board and returns the ASCII character in the "A" register. 'A JR Z loop must be created to repeat the scan as shown below:

```

      KYBD          CALL          002BH
                        CP          00
                        JR          Z,KYBD

```

Whenever a key is pressed, the CP (compare) "A" register with 00H is NOT zero so the program falls through to the line following JR Z,KYBD. This was the most commonly used keyboard subroutine by early assembly language programmers who did not know any better. It is seldom used any longer.

CALL 0049: This is the ROM subroutine most similar to BASIC's INKEY\$ function. It automatically scans the keyboard UNTIL a key is pressed and then places the value in "A" register. No assembly language loop subroutine is required. A giant step forward from the 002BH fundamental keyboard CALL.

CALL 1BB3: This is the big bazoo keyboard subroutine you will be using most frequently. It first displays the "?" prompt . Then input via the keyboard is converted to string format and terminated with a zero (up to 240 bytes). This string is stored at 40A7H + string length. It is usually followed by an RST 10 which sets the "C" flag. See the programs in Chapters 2 and 3 which use this call. This call automatically outputs keyboard input to the device specified by the contents of MEM location (409CH): -1 = cassette, 0 = video display and +1 = line printer. ROM initializes (409CH)=0. This subroutine is surely one of the most time saving, valuable, and frequently used ROM subroutines you will be using henceforth. A single CALL 1BB3H will replace dozens of lines, if not pages, of assembly language programming for you.

DATA MOVEMENT:

Level II ROM ancillary subroutines exist for moving data in assembly language programs FROM - TO virtually any and all locations conceivable, often with only a single CALL. They are tremendous time and line savers and well worth becoming acquainted with on a first-hand basis.

One of the most useful data movement subroutines is that given in the data movement table's first line, CALL 09A4H. This CALL automatically transfers either an integer or single precision number from the ACCUM at MEM locations 411DH through 4124DH to the stack. POP BC and then POP DE will retrieve the number. NOTE: Add, CALL 1C90H to compare table next page (HL-DE NT=2).

If the number is single precision, registers BCDE will contain it. If an integer, register DE will hold the number.

Though ALL the data movement subroutines are very useful, those in lines "J" and "K" deserve special note. Each will MOVE up to 255 bytes from (DE) to (HL). The subroutine at "J" requires that the number of bytes to be moved be in the "A" register, and the subroutine at "K" requires that the number of bytes to be moved be in the "B" register.

NT = number type which is stored in MEM at 40AFH: 2 = integer, 4 = double precision, and 8 = double precision. CS=CDBL store.

- DATA MOVEMENT TABLE -

| NO. | FROM | TO | CALL | NT(40AFH) |
|-----|-------|-------|------------|-----------|
| A. | ACCUM | STACK | 09A4H/2468 | 2,4 |
| B. | (HL)+ | ACCUM | 09B1H/2481 | 4 |
| C. | BCDE | ACCUM | 09B4H/2484 | 4 |
| D. | ACCUM | BCDE | 09BFH/2495 | 4 |
| E. | (HL)+ | BCDE | 09C2H/2498 | 4 |
| F. | ACCUM | (HL)+ | 09CBH/2507 | 4 |
| G. | (DE)+ | (HL)+ | 09CEH/2510 | 4 |
| H. | (HL)+ | (DE)+ | 09D2H/2514 | 2,4,8 |
| I. | (DE)+ | (HL)+ | 09D3H/2515 | 2,4,8 |
| J. | (DE)+ | (HL)+ | 09D6H/2518 | A REG |
| K. | (DE)+ | (HL)+ | 09D7H/2519 | B REG |
| L. | "CS" | ACCUM | 09F4H/2548 | 2,4,8 |
| M. | ACCUM | "CS" | 09FCH/2556 | 2,4,8 |
| N. | HL | ACCUM | 0A9AH/2714 | 2 |
| O. | DE | HL | EX DE,HL | 2 |
| P. | HL | DE | EX DE,HL | 2 |
| Q. | BC | STACK | PUSH BC | 2,4 |
| R. | DE | STACK | PUSH DE | 2,4 |
| S. | HL | STACK | PUSH HL | 2 |
| T. | STACK | HL | POP HL | 2 |
| U. | STACK | DE | POP DE | 2,4 |
| V. | STACK | BC | POP BC | 2,4 |

NOTE: Lines "O" through "V" are just plain old Z-80 OPCODES, but are included to remind the programmer that when dealing with integers or single precision numbers they often are the simplest means of moving or temporarily storing data. See Chapter 2 where PUSH and POP are used to store both integer and single precision numbers while these registers are being used for other purposes.

DATA COMPARISONS:

Of equal to, less than, and greater than are some of the most frequently used functions in computer programming. Level II ROM very thoughtfully includes these functions that may be performed with a single CALL. The result is returned in the "A" register and = zero if the compare is equal, = +1 if the compare is >, and = 255 (OFFH) if the compare is <.

- COMPARE TABLE -

| NO. | ITEM #1 | SUBTRACT | ITEM #2 | CALL | NT (40AFH) |
|-----|---------|----------|----------------|-------|------------|
| A. | ACCUM | - | BCDE | 0A0CH | 4 |
| B. | HL | - | DE | 0A39H | 2 |
| C. | ACCUM | - | "CS" | 0A4FH | 8 |
| D. | "CS" | - | ACCUM | 0A78H | 8 |
| E. | ACCUM | | DETERMINE SIGN | 0994H | 2,4,8 |

NOTE: No. E above is same as the BASIC SGN function, but returns to register "A": zero if ACCUM = 0, +1 if ACCUM greater than zero, and 255 (0FFH) if ACCUM is less than zero.

- DATA CONVERSIONS -

Are straightforward and very necessary in most all arithmetic operations as the NT (number type) must match-up with the CALL subroutine's function; i.e., integer, single precision or double precision + - * /. The most useful conversions are:

- CALL 0A7FH: any ACCUM to integer ACCUM (CINT).
- CALL 0AB1H: any ACCUM to single precision ACCUM (CSNG).
- CALL 0ACCH: integer ACCUM to single precision ACCUM.
- CALL 0ACFH: integer HL to single precision ACCUM.
- CALL 0ADBH: any ACCUM to double precision ACCUM.
- CALL 0E65H: ASCII string to ACCUM in double precision format.
- CALL 0E6CH: ASCII string to ACCUM; NT will = minimum required.

- ARITHMETIC CALL SUMMARY -

| | INTEGER NO. | SINGLE PRECISION | DOUBLE PRECISION |
|----------|----------------------|--------------------------|--------------------------|
| ADDITION | 0BD2H/3026 DE+HL | 0716H/1814 BCDE+ACCUM | 0C77H/3191 ACCUM+"CS" |
| SUBTRACT | 0BC7H/3015 DE-HL | 0713H/1811 BCDE-ACCUM | 0C70H/3184 ACCUM-"CS" |
| MULITPLY | 0BF2H/3058 DE*HL | 0847H/2119 BCDE*ACCUM | 08A2H/2210 ACCUM*"CS" |
| DIVIDE | 2490H/10560 DE/HL | 08A2H/2210 BCDE/ACCUM | 0DE5H/3557 ACCUM/"CS" |

NOTE: NT (number type) at (40AFH) must agree with operation CALLED. NT: 2 = integer, 3 = string, 4 = single precision, and 8 = double precision.

* ACCUM at MEM locations 411DH through 4124H.

* "CS" = CDBL Store at MEM 4127H through 41.EH.

- VIDEO DISPLAY -

Most all TRS-80 video display subroutines have been well known to computer buffs the last 2 years, including the fundamental ROM video subroutine, CALL 033H which display the "A" register on video. CALL 032AH also displays the "A" register on video if MEM location 409CH contains a zero which is the value stored upon initialization. Most IMPORTANTLY, CALL 032AH does indeed store the video display LINE cursor position at MEM location 40A6H which is very useful and eliminates redundant programming on your part.

One of the most important subroutines for reflecting keyboard input on video is CALL 1BB3H which was covered earlier in this Chapter. This CALL displays the string beginning at (HL) and terminated with a zero on video if MEM location 409CH contains a zero. The video display control block, page D/1, Level II Manual in conjunction with the line cursor position at 40A6H allows you to modify and/or use the video display as you wish.

One of the more fascinating assembly language exercises using the video display, is to write a "tight" source code program that creates SPLIT-SCREEN video operation. The upper half of the video display serves as the RECEIVE sector for Morse code, radio teletype (ASCII now allowed), or even simple phone line MODEM communications, while the lower half of the video display would serve as the TRANSMIT segment. This segment allows the user either "look-ahead" or "type-ahead" FIFO (first-in-first-out) operation from RAM at whatever output baud rate desired. If you have the upper-case/lower-case modification recommended by Electric Pencil, it is a simple matter to have those characters that have already been transmitted in upper-case, and those characters yet to be transmitted in lower-case. Alternatively, a moving cursor or a moving CHR\$(170) figure may be used to indicate what data has been transmitted versus data yet to be transmitted in the TRANSMIT sector of the video display. Both halves of the video display operate entirely independently; i.e., from their own separate video MEMs in RAM with their own scrolling, etc. The transmit sector utilizes the Z-80's interrupt mode for "type-ahead" simplex operation. Remember, video memory is just plain old RAM and may be used for storage (as in FIFO) just like any other RAM memory segment.

- LINE PRINTER -

Much like the video display, there are few significant new surprises about line printer ROM subroutines. Again, the value stored in MEM location 409CH determines where the output from CALL 032AH & 1BB3H keyboard subroutines goes; i.e., if (409CH) contains +1, the output will go to the line printer. As shown on page D/1 of the Level II Manual, the line printer address is 37E8H and line printer control block from MEM locations 4025H to 402CH. MEM location 37E8H will contain the

value 63 decimal = ASCII ? when your line printer is ready to accept another character (handshake). A few cheap surplus printers do not have this handshake feature and should be avoided like the plague; caveat emptor, especially with old DATEL printers of ALL types and ALL varieties. Old DATEL printers, even those with the handshake feature do not even make good boat anchors for small dinghys. As soon as MEM location 37E8H receives the 63 handshake from your line printer it is ok (in most cases) to load the next character to be printed into (37E8H) via "A" register and the LD opcode. An exception to this rule is illustrated in Chapter 7's "Print All Zeros With A Slash Program," where an extra 20 millisecond delay was required to allow the vibration from a BACKSPACE to dampen/die out.

This slash/zero program has not been previously published and illustrates a few interesting points about line printer programming. It intercepts the NEXT character to be printed by modifying the line printer driver address at 4026H and 4027H to allow a moment's branching to this brief routine. It just so happens that the "C" register contains the NEXT character to be printed when using the LLIST command with non-disk Level II, DOS 2.1, DOS 2.2, and NEWDOS+, as well as the NEWDOS+ "JKL" feature that LPRINTS the contents of the video display. By simply testing the next character to be printed, a variety of options are made available to the programmer.

To illustrate a few points, let us assume that your line printer utilizes IBM's highest-quality heavy-duty Selectric mechanism like the Western I/O (IBM #2970) Printer Terminal for the TRS-80 that sells for \$1100. Only 8 years ago, these IBM #2970's sold new for over \$7000. each, so here is the industry's most cost-effective printer that is completely refurbished, on the market today. It is the ne plus ultra for those who demand IBM quality print out in both lower and upper case, in addition to the decided advantage of being able to use any or all of the myriad type faces offered in inexpensive IBM Selectric snap-in type spheres.

Even the excellent ASCII IBM type spheres do not include the zero with a slash across it as it looks mighty STRANGE indeed to non-computer types reading a business letter. In some program listings though, it is of considerable assistance to the reader to have the slash/zero printed as such, to avoid confusing zeros with capital "O". Chapter 7's short program prints all zeros with a slash and all lines with 64 characters. It may be modified to print all > = GT and all < = LT, etc., as desired. As it is a teaching program, it may be compacted considerably. Try cutting it down by 1/3rd.

* NOTE: the author, editor, and publisher have done their utmost to find and eliminate typographical errors, but would very much appreciate hearing from readers who find errors we have overlooked. With well over 1,000,000 bytes of data in this handbook, some have surely slipped by us.

CHAPTER 5

-SUMMARY OF LEVEL II ROM CALL ADDRESSES IN ALPHABETICAL ORDER-

This summary includes the coupling CALL addresses for DOS and Disk BASIC because they are called from Level II ROM. In addition, the link address for BREAK is included as it may be intercepted at 400CH before calling an RST and either rendered inoperative or used for whatever purpose the programmer wishes. One extra bonus for disk users under the heading "MASTER" is the master password "F3GUM," which will allow you to access ANY protected file in either DOS or Disk Basic when using DOS 2.1, DOS 2.2, or NEWDOS+. Thank you Manny Garcia. Every lock has a key and "F3GUM" will allow you to LOAD, KILL, transfer or do whatever you wish with any disk file/program whether SIP (system-invisible-protected) or otherwise.

| BASIC FUNCTION | HEX CALL ADDRESS | BASIC FUNCTION | HEX CALL ADDRESS |
|-------------------|---------------------|-------------------|---------------------|
| ABS | 0977 | &H | 4194 |
| ASC | 2A0F | ATN | 15BD |
| AUTO | 2008 | BREAK | 400C |
| CDBL | 0ADB | CHR\$ | 2A1F |
| CINT | 0A7F | CLEAR | 1E7A |
| CLOAD | 2C1F | CLOSE | 4185 |
| CLS | 01C9 | CMD | 4173 |
| CONT | 1DE4 | COS | 1541 |
| CSAVE | 2BF5 | CSNG | 0AB1 |
| CVD | 415E | CVI | 4152 |
| CVS | 4158 | DATA | 1F05 |
| DEF | 415B | DEFDBL | 1E09 |
| DEFINT | 1E03 | DEFSNG | 1E06 |
| DEFSTR | 1E00 | DELETE | 2BC6 |
| DIM | 2608 | EDIT | 2E60 |
| ELSE | 1F07 | END | 1DAE |
| EOF | 4161 | ERL | 24DD |
| ERR | 24CF | ERROR | 1FF4 |
| EXP | 1439 | FIELD | 417C |
| FIX | 0B26 | FOR | 1CA1 |
| FN | 4155 | FRE | 27D4 |
| GET | 417F | GOSUB | 1EB1 |
| GOTO | 1EC2 | IF | 2039 |
| INKEY\$ | 019D | INP | 2AEF |
| INPUT | 219A | INSTR | 419D |
| INT | 0B37 | KILL | 4191 |
| LEFT\$ | 2A61 | LEN | 2A03 |
| LET | 1F21 | LINE | 41A3 |
| LIST | 2B2E | LOAD | 4188 |

| BASIC FUNCTION | HEX CALL ADDRESS | BASIC FUNCTION | HEX CALL ADDRESS |
|-------------------|---------------------|-------------------|---------------------|
| LOC | 4164 | LOF | 4167 |
| LOG | 0809 | LLIST | 2B29 |
| LPRINT | 2067 | LSET | 4197 |
| MASTER | F3GUM | MEM | 27C9 |
| MERGE | 418B | MID\$ | 2A9A |
| MKD\$ | 4170 | MKI\$ | 416A |
| MKS\$ | 416D | NAME | 418E |
| NEW | 1B49 | NEXT | 22B6 |
| NOT | 25C4 | ON | 1F6C |
| OPEN | 4179 | OUT | 2AFB |
| PEEK | 2CAA | POINT | 0133 |
| POKE | 2CB1 | POS | 27F5 |
| PRINT | 206F | PUT | 4218 |
| RANDOM | 01D3 | READ | 21EF |
| REM | 1F07 | RESET | 0138 |
| RESTORE | 1D91 | RETURN | 1EDE |
| RESUME | 1FAF | RIGHT\$ | 2A91 |
| RND | 14C9 | RSET | 419A |
| RUN | 1EA3 | SAVE | 41A0 |
| SET | 0135 | SGN | 098A |
| SIN | 1547 | SQR | 13E7 |
| STOP | 1DA9 | STR\$ | 2836 |
| STRING\$ | 2A2F | SYSTEM | 02B2 |
| TAN | 15A8 | TIME\$ | 4176 |
| TROFF | 1DF8 | TRON | 1DF7 |
| USR | 27FE | VARPTR | 24EB |
| VAL | 2AC5 | | |

NOTE: If you are doing a considerable amount of assembly language programming, we suggest you Xerox these two pages and paste-up a single sheet with the entire CALL functions and addresses on it as they would not fit on a single typed page.

* footnote:

ELSE = 1F07 hex is questionable even though ROM points to this address. ROM also points to 1F07 for the REM function which appears correct. It may be only an improperly shadow-masked bit on this particular Level II ROM chip.

| | | | | | | | | | |
|------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|---------------------|
| 0000 | F3AF | C374 | 06C3 | 0040 | C300 | 40E1 | E9C3 | 9F06 |@...@..... |
| 0010 | C303 | 40C5 | 0601 | 182E | C306 | 40C5 | 0602 | 1826 | ..@.....@.....& |
| 0020 | C309 | 40C5 | 0604 | 181E | C30C | 4011 | 1540 | 18E3 | !..@.....@...@.. |
| 0030 | C30F | 4011 | 1D40 | 18E3 | C312 | 4011 | 2540 | 18DB | ..@...@.....@.8@.. |
| 0040 | C3D9 | 05C9 | 0000 | C3C2 | 03CD | 2B00 | B7C0 | 18F9 |+..... |
| 0050 | 0D0D | 1F1F | 0101 | 5B1B | 0A1A | 0818 | 0919 | 2020 | |
| 0060 | 0B78 | B120 | FBC9 | 3100 | 063A | EC37 | 3CFE | 02D2 |1...:7<... |
| 0070 | 0000 | C3CC | 0611 | 8040 | 21F7 | 1801 | 2700 | EDB0 |@!.....!... |
| 0080 | 21E5 | 4136 | 3A23 | 7023 | 362C | 2322 | A740 | 112D | !.A6:#. #6, #" .@.- |
| 0090 | 0106 | 1C21 | 5241 | 36C3 | 2373 | 2372 | 2310 | F706 | ...!RA6.#.#.#... |
| 00A0 | 1536 | C923 | 2323 | 10F9 | 21E8 | 4270 | 31F8 | 41CD | .6.###...!.B.1.A. |
| 00B0 | 8F1B | CDC9 | 0121 | 0501 | CDA7 | 28CD | B31B | 38F5 |!.....(...8. |
| 00C0 | D7B7 | 2012 | 214C | 4323 | 7CB5 | 281B | 7E47 | 2F77 |!LC#...(..G/. |
| 00D0 | BE70 | 28F3 | 1811 | CD5A | 1EB7 | C297 | 19EB | 2B3E | ..(.....Z.....+> |
| 00E0 | 8F46 | 77BE | 7020 | CE2B | 1114 | 44DF | DA7A | 1911 | .F.....+..D..... |
| 00F0 | CEFF | 22B1 | 4019 | 22A0 | 40CD | 4D1B | 2111 | 01CD | ..".@."@.M.!... |
| 0100 | A728 | C319 | 1A4D | 454D | 4F52 | 5920 | 5349 | 5A45 | .(...MEMORY.SIZE |
| 0110 | 0052 | 4144 | 494F | 2053 | 4841 | 434B | 204C | 4556 | .RADIO.SHACK.LEV |
| 0120 | 454C | 2049 | 4920 | 4241 | 5349 | 430D | 001E | 2CC3 | EL.II.BASIC...., |
| 0130 | A219 | D7AF | 013E | 8001 | 3E01 | F5CF | 28CD | 1C2B |>..>...(+ |
| 0140 | FE80 | D24A | 1EF5 | CF2C | CD1C | 2BFE | 30D2 | 4A1E | ...J.....,+0.J. |
| 0150 | 16FF | 14D6 | 0330 | FBC6 | 034F | F187 | 5F06 | 027A |0...O..... |
| 0160 | 1F57 | 7B1F | 5F10 | F879 | 8F3C | 47AF | 378F | 10FD | .W.....<G.7... |
| 0170 | 4F7A | F63C | 571A | B7FA | 7C01 | 3E80 | 47F1 | B778 | O.<W.....>.G... |
| 0180 | 2810 | 12FA | 8F01 | 792F | 4F1A | A112 | CF29 | C9B1 | (...../O.....)... |
| 0190 | 18F9 | A1C6 | FF9F | E5CD | 8D09 | E118 | EFD7 | E53A |:..... |
| 01A0 | 9940 | B720 | 06CD | 5803 | B728 | 11F5 | AF32 | 9940 | .@.....X..(...2.@ |
| 01B0 | 3CCD | 5728 | F12A | D440 | 77C3 | 8428 | 2128 | 1922 | <.W(*.@...(!(" |
| 01C0 | 2141 | 3E03 | 32AF | 40E1 | C93E | 1CCD | 3A03 | 3E1F | !A>.2.@..>...>. |
| 01D0 | <i>C33A</i> | <i>63E0</i> | <i>5F32</i> | <i>AB40</i> | <i>C921</i> | <i>61FC</i> | <i>C021</i> | <i>6246</i> |2.@.!...!.. |
| 01E0 | 0B10 | FE21 | 02FC | CD21 | 0206 | 0B10 | FE21 | 00FC | ...!...!...!...! |
| 01F0 | CD21 | 0206 | 5C10 | FEC9 | E521 | 00FB | 181B | 7ED6 | !.....!..... |
| 0200 | 233E | 0020 | 0DCD | 012B | CF2C | 7BA2 | C602 | D24A | #>.....+.,.....J |
| 0210 | 1E3D | 32E4 | 37E5 | 2104 | FFCD | 2102 | E1C9 | 2100 | .=2.7.!...!...!. |
| 0220 | FF3A | 3D40 | A4B5 | D3FF | 323D | 40C9 | 3A3F | 3CEE | .:=@.....2=@.:?<. |
| 0230 | 0A32 | 3F3C | C9C5 | E506 | 08CD | 4102 | 10FB | E1C1 | .2?<.....A..... |
| 0240 | C9C5 | F5DB | FF17 | 30FB | 0641 | 10FE | CD1E | 0206 |0..A..... |
| 0250 | 7610 | FEDB | FF47 | F1CB | 1017 | F5CD | 1E02 | F1C1 |G..... |
| 0260 | C9CD | 6402 | E5C5 | D5F5 | 0E08 | 57CD | D901 | 7A07 |W..... |
| 0270 | 5730 | 0BCD | D901 | 0D20 | F2F1 | D1C1 | E1C9 | 0687 | W0..... |
| 0280 | 10FE | 18F2 | CDFE | 0106 | FFAF | CD64 | 0210 | FB3E |> |
| 0290 | A518 | D1CD | FE01 | E5AF | CD41 | 02FE | A520 | F93E |A.....> |
| 02A0 | 2A32 | 3E3C | 323F | 3CE1 | C9CD | 1403 | 22DF | 40CD | *2><2?<.....".@. |
| 02B0 | F801 | CDE2 | 4131 | 8842 | CDFE | 203E | 2ACD | 2A03 |A1.B....>*.* |
| 02C0 | CDE3 | 1BDA | CC06 | D7CA | 9719 | FE2F | 284F | CD93 |/(O.. |
| 02D0 | 02CD | 3502 | FE55 | 20F9 | 0606 | 7EB7 | 2809 | CD35 | ..5..U.....(..5 |
| 02E0 | 02BE | 20ED | 2310 | F3CD | 2C02 | CD35 | 02FE | 7828 |#.....5... (|
| 02F0 | B8FE | 3C20 | F5CD | 3502 | 47CD | 1403 | 854F | CD35 | ..<...5.G....O.5 |

| | | | | | | | | | |
|------|----------------------|------|------|------|------|------|------|---------|-------------------------------|
| 0300 | 0277 | 2381 | 4F10 | F7CD | 3502 | B928 | DA3E | 4332 | ..# ⁴ .O...5..(>C2 |
| 0310 | 3E3C | 18D6 | CD35 | 026F | CD35 | 0267 | C9EB | 2ADF | ><...5...5....*. |
| 0320 | 40EB | D7C4 | 5A1E | 208A | EBE9 | C54F | CDC1 | 413A | @...Z.....O..A: |
| 0330 | 9C40 | B779 | C1FA | 6402 | 2062 | D5CD | 3300 | F5CD | .@.....3... |
| 0340 | 4803 | 32A6 | 40F1 | D1C9 | 3A3D | 40E6 | 083A | 2040 | H.2.@.....=@...: @ |
| 0350 | 2803 | 0FE6 | 1FE6 | 3FC9 | CDC4 | 41D5 | CD2B | 00D1 | (.....?...A..+.. |
| 0360 | C9AF | 3299 | 4032 | A640 | CDAF | 41C5 | 2AA7 | 4006 | ..2.@2.@..A.*.@. |
| 0370 | F0CD | D905 | F548 | 0600 | 0936 | 002A | A740 | F1C1 |H...6.*.@.. |
| 0380 | 2BD8 | AFC9 | CD58 | 03B7 | C018 | F9AF | 329C | 403A | +....X.....2.@: |
| 0390 | 9B40 | B7C8 | 3E0D | D5CD | 9C03 | D1C9 | F5D5 | C54F | .@..>.....O |
| 03A0 | 1E00 | FE0C | 2810 | FE0A | 2003 | 3E0D | 4FFE | 0D28 |(.....>.O..(|
| 03B0 | 053A | 9B40 | 3C5F | 7B32 | 9B40 | 79CD | 3B00 | C1D1 | ..@<..2.@..;... |
| 03C0 | F1C9 | E5DD | E5D5 | DDE1 | D521 | DD03 | E54F | 1AA0 |!...O.. |
| 03D0 | B8C2 | 3340 | FE02 | DD6E | 01DD | 6602 | E9D1 | DDE1 | ..3@..... |
| 03E0 | E1C1 | C921 | 3640 | 0101 | 3816 | 000A | 5FAE | 73A3 | ...!6@..8..... |
| 03F0 | 2008 | 142C | CB01 | F2EB | 03C9 | 5F7A | 0707 | 0757 |,.....W |
| 0400 | 0E01 | 79A3 | 2005 | 14CB | 0118 | F73A | 8038 | 477A |:8G. |
| 0410 | C640 | FE60 | 3013 | CB08 | 3031 | C620 | 573A | 4038 | .@..0...01..W:@8 |
| 0420 | E610 | 2828 | 7AD6 | 6018 | 22D6 | 7030 | 10C6 | 40FE | ..((....."0..@. |
| 0430 | 3C38 | 02EE | 10CB | 0830 | 12EE | 1018 | 0E07 | CB08 | <8.....0..... |
| 0440 | 3001 | 3C21 | 5000 | 4F06 | 0009 | 7E57 | 01AC | 0DCD | 0.<!P.O....W.... |
| 0450 | 6444 7AFE | 01C0 | EFC9 | DD6E | 03DD | 6604 | 383A |8: | |
| 0460 | DD7E | 05B7 | 2801 | 7779 | FE20 | DA06 | 05FE | 8030 |(.....0 |
| 0470 | 35FE | 4038 | 08D6 | 40FE | 2038 | 02D6 | 20CD | 4105 | 5.@8..@..8....A. |
| 0480 | 7CE6 | 03F6 | 3C67 | 56DD | 7E05 | B728 | 05DD | 7205 |<.V....(.... |
| 0490 | 365F | DD75 | 03DD | 7404 | 79C9 | DD7E | 05B7 | C07E | 6..... |
| 04A0 | C97D | E6C0 | 6FC9 | FEC0 | 38D3 | D6C0 | 28D2 | 473E |8...(.G> |
| 04B0 | 20CD | 4105 | 10F9 | 18C8 | 7EDD | 7705 | C9AF | 18F9 | ..A..... |
| 04C0 | 2100 | 3C3A | 3D40 | E6F7 | 323D | 40D3 | FFC9 | 2B3A | !.<:=@..2=@...+: |
| 04D0 | 3D40 | E608 | 2801 | 2B36 | 20C9 | 3A3D | 40E6 | 08C4 | =@..(+6...:=@... |
| 04E0 | E204 | 7DE6 | 3F2B | C011 | 4000 | 19C9 | 237D | E63F |?+..@...#..? |
| 04F0 | C011 | C0FF | 19C9 | 3A3D | 40F6 | 0832 | 3D40 | D3FF |:=@..2=@.. |
| 0500 | 237D | E6FE | 6FC9 | 1180 | 04D5 | FE08 | 28C0 | FE0A | #.....~(... |
| 0510 | D8FE | 0E38 | 4F28 | A1FE | 0F28 | A2FE | 1728 | D7FE | ...80(...(....(. |
| 0520 | 1828 | B7FE | 1928 | C5FE | 1A28 | BCFE | 1B28 | C2FE | .(....(....(....(. |
| 0530 | 1C28 | 8DFE | 1DCA | A104 | FE1E | 2837 | FE1F | 283C | .(.....(7..(< |
| 0540 | C977 | 233A | 3D40 | E608 | 2801 | 237C | FE40 | C011 | ..#:=@..(.#..@.. |
| 0550 | C0FF | 19E5 | 1100 | 3C21 | 403C | C501 | C003 | EDB0 |<!@<..... |
| 0560 | C1EB | 1819 | 7DE6 | C06F | E511 | 4000 | 197C | FE40 |@.....@ |
| 0570 | 28E2 | D1E5 | 547D | F63F | 5F13 | 1804 | E511 | 0040 | (...T..?.....@ |
| 0580 | 3620 | 237C | BA20 | F97D | BB20 | F5E1 | C979 | B728 | 6.#.....(|
| 0590 | 40FE | 0B28 | 0AFE | 0C20 | 1BAF | DDB6 | 0328 | 15DD | @..(.....(. |
| 05A0 | 7E03 | DD96 | 0447 | CDD1 | 0520 | FB3E | 0A32 | E837 |G.....>.2.7 |
| 05B0 | 10F4 | 1818 | F5CD | D105 | 20FB | F132 | E837 | FE0D |2.7.. |
| 05C0 | C0DD | 3404 | DD7E | 04DD | BE03 | 79C0 | DD36 | 0400 | ..4.....6.. |
| 05D0 | C93A | E837 | E6F0 | FE30 | C9E5 | 3E0E | CD33 | 0048 | ..:7...0..>..3.H |
| 05E0 | CD49 | 00FE | 2030 | 25FE | 0DCA | 6206 | FE1F | 2829 | .I...0&.....() |
| 05F0 | FE01 | 286D | 11E0 | 05D5 | FE08 | 2834 | FE18 | 282B | ..(.....(4..(+ |

```

0600 FE09 2842 FD19 2839 FE0A C0D1 7778 B728 ..(B..(9.....(
0610 CF7E 23CD 3300 0518 C7CD C901 41E1 E5C3 ..#.3.....A...
0620 E005 CD30 062B 7E23 FE0A C878 B920 F3C9 ...0.+.#.....
0630 78B9 C82B 7EFE 0A23 C82B 3E08 CD33 0004 ...+...#.+>..3..
0640 C93E 17C3 3300 CD48 03E6 072F 3CC6 085F .>..3..H.../<...
0650 78B7 C83E 2077 23D5 CD33 00D1 051D C818 ...>..#..3.....
0660 EF37 F53E 0D77 CD33 003E 0FCD 3300 7990 .7.>...3.>..3...
0670 47F1 E1C9 D3FF 21D2 0611 0040 0136 00ED G.....!.....@.6..
0680 B03D 3D20 F106 2712 1310 FC3A 4038 E604 .==...'.@8..
0690 C275 0031 7D40 3AFC 373C FE02 DA75 003E ...1.@:..7<.....>
06A0 0132 E137 21EC 3711 EF37 3603 0100 00CD .2.7!.7..76.....
06B0 6000 CB46 20FC AF32 EE37 0100 423E 8C77 ...F...2.7..B>..
06C0 CB4E 28FC 1A02 0C20 F7C3 0042 0118 1AC3 .N(.....B.....
06D0 AE19 C396 1CC3 781D C390 1CC3 D925 C900 .....%...
06E0 00C9 00C9 FBC9 0001 E303 00C9 004B 4907 .....KI.
06F0 5804 003C 0044 4F06 8D05 4300 0050 52C3 X..<.DO...C..PR.
0700 0050 C700 003E 00C9 2180 13CD C209 1806 .P...>..!.....
0710 CDC2 09CD 8209 78B7 C83A 2441 B7CA B409 .....:SA....
0720 9030 0C2F 3CEB CDA4 09EB CDB4 09C1 D1FE .0./<.....
0730 19D0 F5CD DF09 67F1 CDD7 07B4 2121 41F2 .....!!A.
0740 5407 CDB7 07D2 9607 2334 CAB2 072E 01CD T.....#4.....
0750 EB07 1842 AF90 477E 9B5F 237E 9A57 237E ...B..G...#.W#.
0760 994F DCC3 0768 63AF 4779 B720 184A 5465 .O.....G....JT.
0770 6F78 D608 FEE0 20F0 AF32 2441 C905 297A .....2$A..).
0780 1757 798F 4FF2 7D07 785C 45B7 2808 2124 .W..O.....E.(!$
0790 4186 7730 E3C8 7821 2441 B7FC A807 4623 A..0...!$A....F#
07A0 7EE6 80A9 4FC3 B409 1CC0 14C0 0CC0 0E80 ....O.....
07B0 34C0 1E0A C3A2 197E 835F 237E 8A57 237E 4.....#.W#.
07C0 894F C921 2541 7E2F 77AF 6F90 477D 9B5F .O.!$A./.....G...
07D0 7D9A 577D 994F C906 00D6 0838 0743 5A51 ..W..O.....8.CZQ
07E0 0E00 18F5 C609 6FAF 2DC8 791F 4F7A 1F57 .....-...O..W
07F0 7B1F 5F78 1F47 18EF 0000 0081 03AA 5619 .....G.....V.
0800 80F1 2276 8045 AA38 82CD 5509 B7EA 4A1E ..".E.8..U..J.
0810 2124 417E 0135 8011 F304 90F5 70D5 C5CD !$A..5.....
0820 1607 C1D1 04CD A208 21F8 07CD 1007 21FC .....!.....!
0830 07CD 9A14 0180 8011 0000 CD16 07F1 CD89 .....
0840 0F01 3180 1118 72CD 5509 C82E 00CD 1409 ..1.....U.....
0850 7932 4F41 EB22 5041 0100 0050 5821 6507 .2OA."PA...PX!..
0860 E521 6908 E5E5 2121 417E 23B7 2824 E52E .!.....!!A.#.($..
0870 081F 6779 300B E52A 5041 19EB E13A 4F41 ....0..*PA...:OA
0880 891F 4F7A 1F57 7B1F 5F78 1F47 2D7C 20E1 ..O..W.....G-...
0890 E1C9 435A 514F C9CD A409 21D8 0DCD B109 ..CZQO.....!.....
08A0 C1D1 CD55 09CA 9A19 2EFF CD14 0934 342B ...U.....44+
08B0 7E32 8940 2B7E 3285 402B 7E32 8140 41EB .2.@+.2.@+.2.@A.
08C0 AF4F 575F 328C 40E5 C57D CD80 40DE 003F .OW.2.@.....@..?
08D0 3007 328C 40F1 F137 D2C1 E179 3C3D 1FFA 0.2.@..7.....<=..
08E0 9707 177B 175F 7A17 5779 174F 2178 1747 .....W..O)..G
08F0 3A8C 4017 328C 4079 B2B3 40CB E521 2441 :.@.2.@.....!$A

```

| | | | | | | | | | |
|------|------|------|------|------|------|------|----------------------|------------------|--------------------|
| 0900 | 35E1 | 20C3 | C3B2 | 073E | FF2E | AF21 | 2D41 | 4E23 | 5.....>...!-AN# |
| 0910 | AE47 | 2E00 | 78B7 | 281F | 7D21 | 2441 | AE80 | 471F | .G....(!!\$A..G. |
| 0920 | A878 | F236 | 09C6 | 8077 | CA90 | 08CD | DF09 | 772B | ...6.....+. |
| 0930 | C9CD | 5509 | 2FE1 | B7E1 | F278 | 07C3 | B207 | CDBF | ..U./..... |
| 0940 | 0978 | B7C8 | C602 | DAB2 | 0747 | CD16 | 0721 | 2441 |G...!\$A |
| 0950 | 34C0 | C3B2 | 073A | 2441 | B7C8 | 3A23 | 41FE | 2F17 | 4.....:\$A...#A./. |
| 0960 | 9FC0 | 3CC9 | 0688 | 1100 | 0021 | 2441 | 4F70 | 0600 | ..<.....!\$AO... |
| 0970 | 2336 | 8017 | C362 | 07CD | 9409 | F0E7 | FA5B | 0CCA | #6..... |
| 0980 | F60A | 2123 | 417E | EE80 | 77C9 | CD94 | 096F | 179F | ..!#A..... |
| 0990 | 67C3 | 9A0A | E7CA | F60A | F255 | 092A | 2141 | 7CB5 |U.*!A.. |
| 09A0 | C87C | 18BB | EB2A | 2141 | E3E5 | 2A23 | 41E3 | E5EB |*!A.*#A... |
| 09B0 | C9CD | C209 | EB22 | 2141 | 6069 | 2223 | 41EB | C921 |"!A.."#A..! |
| 09C0 | 2141 | 5E23 | 5623 | 4E23 | 4623 | C911 | 2141 | 0604 | !A.#V#N#F#...!A.. |
| 09D0 | 1805 | EB3A | AF40 | 471A | 7713 | 2305 | 20F9 | C921 |@G...#.....! |
| 09E0 | 2341 | 7E07 | 371F | 773F | 1F23 | 2377 | 7907 | 371F | #A..7..?..##...7. |
| 09F0 | 4F1F | AEC9 | 2127 | 4111 | D209 | 1806 | 2127 | 4111 | O...!'A.....!'A. |
| 0A00 | D309 | D511 | 2141 | E7D3 | 111D | 41C9 | 78B7 | CA55 |!A....A....U |
| 0A10 | 0921 | 5E09 | E5CD | 5509 | 79C8 | 2123 | 41AE | 79F8 | .!.....U...!#A... |
| 0A20 | CD26 | 0A1F | A9C9 | 2378 | BEC0 | 2B79 | BEC0 | 2B7A | ..&.....#...+...+. |
| 0A30 | BEC0 | 2B7B | 96C0 | E1E1 | C97A | AC7C | FA5F | 09BA | ..+.....~..... |
| 0A40 | C260 | 097D | 93C2 | 6009 | C921 | 2741 | CDD3 | 0911 |!'A.... |
| 0A50 | 2E41 | 1AB7 | CA55 | 0921 | 5E09 | E5CD | 5509 | 1B1A | .A...U.!...U... |
| 0A60 | 4FC8 | 2123 | 41AE | 79F8 | 1323 | 0608 | 1A96 | C223 | O.!#A....#.....# |
| 0A70 | 0A1B | 2B75 | 20F6 | C1C9 | CD4F | 0AC2 | 5E09 | C9E7 | ..+.....O..... |
| 0A80 | 2A21 | 41F8 | CAF6 | 0AD4 | B90A | 21B2 | 07E5 | 3A24 | *!A.....!...:\$ |
| 0A90 | 41FE | 9030 | 0ECD | FB0A | EBD1 | 2221 | 413E | 0232 | A..0....."!A>.2 |
| 0AA0 | AF40 | C901 | 8090 | 1100 | 00CD | 0C0A | C061 | 6A18 | .@..... |
| 0AB0 | E8E7 | E0FA | CC0A | CAF6 | 0ACD | BF09 | CDEF | 0A78 | |
| 0AC0 | B7C8 | CDDF | 0921 | 2041 | 46C3 | 9607 | 2A21 | 41CD |!AF...*!A. |
| 0AD0 | EF0A | 7C55 | 1E00 | 0690 | C369 | 09E7 | D0CA | F60A | ...U..... |
| 0AE0 | FCCC | 0A21 | 0000 | 221D | 4122 | 1F41 | 3E08 | 013E | ...!.."A".A>..> |
| 0AF0 | 04C3 | 9F0A | E7C8 | 1E18 | C3A2 | 1947 | 4F57 | 5FB7 |GOW.. |
| 0B00 | C8E5 | CDBF | 09CD | DF19 | AE67 | FC1F | 0B3E | 9890 |>.. |
| 0B10 | CDD7 | 077C | 17DC | A807 | 0600 | DCC3 | 07E1 | C91B | |
| 0B20 | 7AA3 | 3CC0 | 0BC9 | E7F8 | CD55 | 09F2 | 370B | CD82 | ..<.....U..7... |
| 0B30 | 09CD | 370B | C37B | 09E7 | F830 | 1E28 | B9CD | 8E0A | ..7.....0.(.... |
| 0B40 | 2124 | 417E | FE98 | 3A21 | 41D0 | 7ECD | FB0A | 3698 | !\$A...:!A.....6. |
| 0B50 | 7BF5 | 7917 | CD62 | 07F1 | C921 | 2441 | 7EFE | 90DA |!\$A.... |
| 0B60 | 7F0A | 2014 | 4F2B | 7EEE | 8006 | 062B | B605 | 20FB |O+.....+.... |
| 0B70 | B721 | 0080 | CA9A | 0A79 | FEB8 | D0F5 | CDBF | 09CD | ..!..... |
| 0B80 | DF09 | AE2E | 36B8 | F5FC | A00B | 2123 | 413E | B890 | ...+6.....!#A>.. |
| 0B90 | CD69 | 0DF1 | FC20 | 0DAF | 321C | 41F1 | D0C3 | D80C |2.A..... |
| 0BA0 | 211D | 417E | 35B7 | 2328 | FAC9 | E521 | 0440 78B7 | !A.5.#(....!.... | |
| 0BB0 | 2812 | 3E10 | 29DA | 3D27 | EB29 | EB30 | 0409 | DA3D | (.>.)='.)0...= |
| 0BC0 | 273D | 20F0 | EBE1 | C97C | 179F | 47CD | 510C | 7998 | '=.....G.Q.... |
| 0BD0 | 1803 | 7C17 | 9F47 | E57A | 179F | 1988 | 0FAC | F299 |G..... |
| 0BE0 | 0AC5 | EBCD | CF0A | F1E1 | CDA4 | 09EB | CD6B | 0CC3 | |
| 0BF0 | 8F0F | 7CB5 | CA9A | 0AE5 | D5CD | 450C | C544 | 4D21 |E..DM! |

| | | | | | | | | |
|------|-----------------|------|------------|------------|------------|------------|------|--------------------------|
| 0C00 | <i>444</i> 3E10 | 2938 | 1FEB | 29EB | 3004 | 09DA | 260C | ..>.)8..).0...&. |
| 0C10 | 3D20 | F1C1 | D17C | B7FA | 1F0C | D178 | C34D | 0CEE =.....M.. |
| 0C20 | 80B5 | 2813 | EB01 | C1E1 | CDCF | 0AE1 | CDA4 | 09CD ..(..... |
| 0C30 | CF0A | C1D1 | C347 | 0878 | B7C1 | FA9A | 0AD5 | CDCFG..... |
| 0C40 | 0AD1 | C382 | 097C | AA47 | CD4C | 0CEB | 7CB7 | F29AG.L..... |
| 0C50 | 0AAF | 4F95 | 6F79 | 9C67 | C39A | 0A2A | 2141 | CD51 ..O.....*!A.Q |
| 0C60 | 0C7C | EE80 | B5C0 | EBCD | EF0A | AF06 | 98C3 | 6909 |
| 0C70 | 212D | 417E | EE80 | 7721 | 2E41 | 7EB7 | C847 | 2B4E !-A....!.A...G+N |
| 0C80 | 1124 | 411A | B7CA | F409 | 9030 | 162F | 3CF5 | 0E08 .\$.A.....0./<... |
| 0C90 | 23E5 | 1A46 | 7778 | 121B | 2B0D | 20F6 | E146 | 2B4E #..F.....+.....F+N |
| 0CA0 | F1FE | 39D0 | F5CD | DF09 | 2336 | 0047 | F121 | 2D41 ..9.....#6.G.!-A |
| 0CB0 | CD69 | 0D3A | 2641 | 321C | 4178 | B7F2 | CF0C | CD33:&A2.A.....3 |
| 0CC0 | 0DD2 | 0E0D | EB34 | CAB2 | 07CD | 900D | C30E | 0DCD4..... |
| 0CD0 | 450D | 2125 | 41DC | 570D | AF47 | 3A23 | 41B7 | 201E E.!%A.W..G:#A... |
| 0CE0 | 211C | 410E | 0856 | 777A | 230D | 20F9 | 78D6 | 08FE !.A..V..#..... |
| 0CF0 | C020 | E6C3 | 7807 | 0521 | 1C41 | CD97 | 0DB7 | F2F6!.A..... |
| 0D00 | 0C78 | B728 | 0921 | 2441 | 8677 | D278 | 07C8 | 3A1C ...(!\$.A.....:. |
| 0D10 | 41B7 | FC20 | 0D21 | 2541 | 7EE6 | 802B | 2BAE | 77C9 A.....!%A...++... |
| 0D20 | 211D | 4106 | 0734 | C023 | 0520 | FA34 | CAB2 | 072B !.A..4.#...4...+ |
| 0D30 | 3680 | C921 | 2741 | 111D | 410E | 07AF | 1A8E | 1213 6..!'A..A..... |
| 0D40 | 230D | 20F8 | C921 | 2741 | 111D | 410E | 07AF | 1A9E #.....!'A..A..... |
| 0D50 | 1213 | 230D | 20F8 | C97E | 2F77 | 211C | 4106 | 08AF ..#...../!.A... |
| 0D60 | 4F79 | 9E77 | 2305 | 20F9 | C971 | E5D6 | 0838 | 0EE1 O...#.....8.. |
| 0D70 | E511 | 0008 | 4E73 | 592B | 1520 | F918 | EEC6 | 0957N.Y+.....W |
| 0D80 | AFE1 | 15C8 | E51E | 087E | 1F77 | 2B1D | 20F9 | 18F0 |
| 0D90 | 2123 | 4116 | 0118 | ED0E | 087E | 1777 | 230D | 20F9 !#A.....#... |
| 0DA0 | C9CD | 5509 | C8CD | 0A09 | CD39 | 0E71 | 1306 | 071A ..U.....9..... |
| 0DB0 | 13B7 | D528 | 170E | 08C5 | 1F47 | DC33 | 0DCD | 900D ...(.G.3.... |
| 0DC0 | 78C1 | 0D20 | F2D1 | 0520 | E6C3 | D80C | 2123 | 41CD |
| 0DD0 | 700D | 18F1 | <i>444</i> | <i>444</i> | <i>444</i> | <i>108</i> | 11D4 | 0D21 |
| 0DE0 | 2741 | CDD3 | 093A | 2E41 | B7CA | 9A19 | CD07 | 0934 'A.....A.....4 |
| 0DF0 | 34CD | 390E | 2151 | 4171 | 4111 | 4A41 | 2127 | 41CD 4.9.!QA.A.JA!'A. |
| 0E00 | 4B0D | 1A99 | 3F38 | 0B11 | 4A41 | 2127 | 41CD | 390D K...?8..JA!'A.9. |
| 0E10 | AFDA | 1204 | 3A23 | 413C | 3D1F | FA11 | 0D17 | 211D:#A<=.....!. |
| 0E20 | 410E | 07CD | 990D | 214A | 41CD | 970D | 78B7 | 20C9 A.....!JA..... |
| 0E30 | 2124 | 4135 | 20C3 | C3B2 | 0779 | 322D | 412B | 1150 !\$A5.....2-A+.P |
| 0E40 | 4101 | 0007 | 7E12 | 711B | 2B05 | 20F8 | C9CD | FC09 A.....+..... |
| 0E50 | EB2B | 7EB7 | C8C6 | 02DA | B207 | 77E5 | CD77 | 0CE1 .+..... |
| 0E60 | 34C0 | C3B2 | 07CD | 7807 | CDEC | 0AF6 | AFEB | 01FF 4..... |
| 0E70 | 0060 | 68CC | 9A0A | EB7E | FE2D | F5CA | 830E | FE2B |
| 0E80 | 2801 | 2BD7 | DA29 | 0FFE | 2ECA | E40E | FE45 | 2814 (.+..).....E(.) |
| 0E90 | FE25 | CAEE | 0EFE | 23CA | F50E | FE21 | CAF6 | 0EFE .%.....#.....!..... |
| 0EA0 | 4420 | 24B7 | CDFB | 0EE5 | 21BD | 0EE3 | D715 | FECE D.\$.....!..... |
| 0EB0 | C8FE | 2DC8 | 14FE | CDC8 | FE2B | C82B | F1D7 | DA94 ..-.....+.+..... |
| 0EC0 | 0F14 | 2003 | AF93 | 5FE5 | 7B90 | F40A | 0FFC | 180F |
| 0ED0 | 20F8 | E1F1 | E5CC | 7B09 | E1E7 | E8E5 | 2190 | 08E5 |
| 0EE0 | CDA3 | 0AC9 | E70C | 20DF | DCFB | 0EC3 | 830E | E7F2 |
| 0EF0 | 9719 | 2318 | D2B7 | CDFB | 0E18 | F7E5 | D5C5 | F5CC ..#..... |

| | | | | | | | | | |
|------|------|------|------|------------|------------|-----------------|------------|------|---------------------|
| 0F00 | B10A | F1C4 | DB0A | C1D1 | E1C9 | C8F5 | E7F5 | E43E |> |
| 0F10 | 09F1 | EC4D | 0EF1 | 3DC9 | D5E5 | F5E7 | F5E4 | 9708 | ...M..=..... |
| 0F20 | F1EC | DC0D | F1E1 | D13C | C9D5 | 7889 | 47C5 | E57E |<....G... |
| 0F30 | D630 | F5E7 | F25D | 0F2A | 2141 | 11CD | 0CDF | 3019 | .0.....*!A.....0. |
| 0F40 | 545D | 2929 | 1929 | F14F | 097C | B7FA | 570F | 2221 | T.))..).O....W."! |
| 0F50 | 41E1 | C1D1 | C383 | 0E79 | F5CD | CC0A | 3730 | 1801 | A.....70.. |
| 0F60 | 7494 | 1100 | 24CD | 0C0A | F274 | 0FCD | 3E09 | F1CD |\$.>.... |
| 0F70 | 890F | 18DD | CDE3 | 0ACD | 4D0E | CDFC | 09F1 | CD64 |M..... |
| 0F80 | 09CD | E30A | CD77 | 0C18 | C8CD | A409 | CD64 | 09C1 | |
| 0F90 | D1C3 | 1607 | 7BFE | 0A30 | 0907 | 0783 | 0786 | D630 |0.....0 |
| 0FA0 | 5FFA | 1E32 | C3BD | 0EE5 | 2124 | 19CD | A728 | E1CD | ...2....!\$.<.. |
| 0FB0 | 9A0A | AFCD | 3410 | B6CD | D90F | C3A6 | 28AF | CD34 |4.....(.4 |
| 0FC0 | 10E6 | 0828 | 0236 | 2BEB | CD94 | 09EB | F2D9 | 0F36 | ...(.6+.....6 |
| 0FD0 | 2DC5 | E5CD | 7B09 | E1C1 | B423 | 3630 | 3AD8 | 4057 | -.....#60:..@W |
| 0FE0 | 173A | AF40 | DA9A | 10CA | 9210 | FE04 | D23D | 1001 | ...@.....=.. |
| 0FF0 | 0000 | CD2F | 1321 | 3041 | 460E | 203A | D840 | 5FE6 | .../..!0AF....@.. |
| 1000 | 2028 | 0778 | B90E | 2A20 | 0141 | 71D7 | 2814 | FE45 | .(....*..A..(.E |
| 1010 | 2810 | FE44 | 280C | FE30 | 28F0 | FE2C | 28EC | FE2E | (..D(..0(..,(... |
| 1020 | 2003 | 2B36 | 307B | E610 | 2803 | 2B36 | 247B | E604 | ..+60...(+6\$.>.. |
| 1030 | C02B | 70C9 | 32D8 | 4021 | 3041 | 3620 | C9FE | 05E5 | +.2.@!0A6..... |
| 1040 | DE00 | 1757 | 14CD | <i>112</i> | <i>116</i> | <i>352-FA57</i> | <i>114</i> | | ...W.....W.. |
| 1050 | BA30 | 043C | 473E | 02D6 | 02E1 | F5CD | 9112 | 3630 | .0.<G>.....60 |
| 1060 | CCC9 | 09CD | A412 | 2B7E | FE30 | 28FA | FE2E | C4C9 |+..0(..... |
| 1070 | 09F1 | 281F | F5E7 | 3E22 | 8F77 | 23F1 | 362B | F285 | ..(....>"..#.6+.. |
| 1080 | 1036 | 2D2F | 3C06 | 2F04 | D60A | 30FB | C63A | 2370 | .6-/<./...0...:#. |
| 1090 | 2377 | 2336 | 00EB | 2130 | 41C9 | 23C5 | FE04 | 7AD2 | #.6..!0A.#..... |
| 10A0 | 0911 | 1FDA | A311 | 0103 | 06CD | 8912 | D17A | D605 | |
| 10B0 | F469 | 12CD | 2F13 | 7BB7 | CC2F | 093D | F469 | 12E5 |/...../=.... |
| 10C0 | CDF5 | 0FE1 | 2802 | 7023 | 3600 | 212F | 4123 | 3AF3 |(..#6.!/A#:.>.. |
| 10D0 | 4095 | 92C8 | 7EFE | 2028 | F4FE | 2A28 | F02B | E5F5 | @.....(.*(+.. |
| 10E0 | 01DF | 10C5 | D7FE | 2DC8 | FE2B | C8FE | 24C8 | C1FE |-..+..\$.>.. |
| 10F0 | 3020 | 0F23 | D730 | 0B2B | 012B | 77F1 | 28FB | C1C3 | 0..#.0.+..+..(... |
| 1100 | CE10 | F128 | FDE1 | 3625 | C9E5 | 1FDA | AA11 | 2814 | ...(..6%...>..(. |
| 1110 | 1184 | 13CD | 490A | 1610 | FA32 | 11E1 | C1CD | BD0F |I....2..... |
| 1120 | 2B36 | 25C9 | 010E | B611 | CA1B | CD0C | 0AF2 | 1B11 | +6%..... |
| 1130 | 1606 | CD55 | 09C4 | 0112 | E1C1 | FA57 | 11C5 | 5F78 | ...U.....W.... |
| 1140 | 9293 | F469 | 12CD | 7D12 | CDA4 | 12B3 | C477 | 12B3 | |
| 1150 | C491 | 12D1 | C3B6 | 105F | 79B7 | C416 | 0F83 | FA62 | |
| 1160 | 11AF | C5F5 | FC18 | 0FFA | 6411 | C17B | 90C1 | 5F82 | |
| 1170 | 78FA | 7F11 | 9293 | F469 | 12C5 | CD7D | 1218 | 11CD | |
| 1180 | 6912 | 79CD | 9412 | 4FAF | 9293 | CD69 | 12C5 | 474F |O.....GO |
| 1190 | CDA4 | 12C1 | B120 | 032A | F340 | 833D | F469 | 1250 |*.e.=...P |
| 11A0 | C3BF | 10E5 | D5CD | CC0A | D1AF | CAB0 | 111E | 1001 | |
| 11B0 | 1E06 | CD55 | 0937 | C401 | 12E1 | C1F5 | 79B7 | F5C4 | ...U.7..... |
| 11C0 | 160F | 804F | 7AE6 | 04FE | 019F | 5781 | 4F93 | F5C5 | ...O.....W.O...>.. |
| 11D0 | FC18 | 0FFA | D011 | C1F1 | C5F5 | FADE | 11AF | 2F3C |/< |
| 11E0 | 803C | 8247 | 0E00 | CDA4 | 12F1 | F471 | 12C1 | F1CC | .<.G..... |
| 11F0 | 2F09 | F138 | 0383 | 9092 | C5CD | 7410 | EBD1 | C3BF | /..8..... |

| | | | | | | | | | |
|------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-------------------|
| 1200 | 10D5 | AFF5 | E7E2 | 2212 | 3A24 | 41FE | 91D2 | 2212 |"::\$A..." |
| 1210 | 1164 | 1321 | 2741 | CDD3 | 09CD | A10D | F1D6 | 0AF5 | ...!'A..... |
| 1220 | 18E6 | CD4F | 12E7 | 300B | 0143 | 9111 | F94F | CD0C | ...O..0..C...O.. |
| 1230 | 0A18 | 0611 | 6C13 | CD49 | 0AF2 | 4B12 | F1CD | 0B0F |I..K..... |
| 1240 | F518 | E2F1 | CD18 | 0FF5 | CD4F | 12F1 | B7D1 | C9E7 |O..... |
| 1250 | EA5E | 1201 | 7494 | 11F8 | 23CD | 0C0A | 1806 | 1174 |#..... |
| 1260 | 13CD | 490A | E1F2 | 4312 | E9B7 | C83D | 3630 | 2318 | ..I...C....=60#. |
| 1270 | F920 | 04C8 | CD91 | 1236 | 3023 | 3D18 | F67B | 823C |60#=...< |
| 1280 | 473C | D603 | 30FC | C605 | 4F3A | D840 | E640 | C04F | G<..0...O::@.@.O |
| 1290 | C905 | 2008 | 362E | 22F3 | 4023 | 48C9 | 0DC0 | 362C |6.".@#H...6, |
| 12A0 | 230E | 03C9 | D5E7 | E2EA | 12C5 | E5CD | FC09 | 217C | #.....!. |
| 12B0 | 13CD | F709 | CD77 | 0CAF | CD7B | 0BE1 | C111 | 8C13 | |
| 12C0 | 3E0A | CD91 | 12C5 | F5E5 | D506 | 2F04 | E1E5 | CD48 | >...../....H |
| 12D0 | 0D30 | F8E1 | CD36 | 0DEB | E170 | 23F1 | C13D | 20E2 | .0...6....#...=. |
| 12E0 | C5E5 | 211D | 41CD | B109 | 180C | C5E5 | CD08 | 073C | ..!A.....< |
| 12F0 | CDFB | 0ACD | B409 | E1C1 | AF11 | D213 | 3FCD | 9112 |?... |
| 1300 | C5F5 | E5D5 | CDBF | 09E1 | 062F | 047B | 965F | 237A |/.....#. |
| 1310 | 9E57 | 2379 | 9E4F | 2B2B | 30F0 | CDB7 | 0723 | CDB4 | .W#..O++0....#.. |
| 1320 | 09EB | E170 | 23F1 | C138 | D313 | 133E | 0418 | 06D5 |#..8...>.... |
| 1330 | 11D8 | 133E | 05CD | 9112 | C5F5 | E5EB | 4E23 | 46C5 | ...>.....N#F. |
| 1340 | 23E3 | EB2A | 2141 | 062F | 047D | 936F | 7C9A | 6730 | #..*!A./.....0 |
| 1350 | F719 | 2221 | 41D1 | E170 | 23F1 | C13D | 20D7 | CD91 | .."!A...#...=.... |
| 1360 | 1277 | D1C9 | 0000 | 0000 | F902 | 15A2 | FDFE | 9F31 |1 |
| 1370 | A95F | 63B2 | FEFF | 03BF | C91B | 0EB6 | 0000 | 0000 | |
| 1380 | 0000 | 0000 | 0000 | 04BF | C91B | 0EB6 | 0080 | C6A4 | |
| 1390 | 7E8D | 0300 | 407A | 10F3 | 5A00 | 00A0 | 724E | 1809 |@...Z....N.. |
| 13A0 | 0000 | 10A5 | D4E8 | 0000 | 00E8 | 7648 | 1700 | 0000 |H.... |
| 13B0 | E40B | 5402 | 0000 | 00CA | 9A3B | 0000 | 0000 | E1F5 | ..T.....;..... |
| 13C0 | 0500 | 0000 | 8096 | 9800 | 0000 | 0040 | 420F | 0000 |@B... |
| 13D0 | 0000 | A086 | 0110 | 2700 | 1027 | E803 | 6400 | 0A00 |'...'..... |
| 13E0 | 0100 | 2182 | 09E3 | E9CD | A409 | 2180 | 13CD | B109 | ..!.....!..... |
| 13F0 | 1803 | CDB1 | 0AC1 | D1CD | 5509 | 7828 | 3CF2 | 0414 |U..(<... |
| 1400 | B7CA | 9A19 | B7CA | 7907 | D5C5 | 79F6 | 7FCD | BF09 |:..... |
| 1410 | F221 | 14D5 | C5CD | 400B | C1D1 | F5CD | CC0A | E17C | ..!....@..... |
| 1420 | 1FE1 | 2223 | 41E1 | 2221 | 41DC | E213 | CC82 | 09D5 | .."#A."!A..... |
| 1430 | C5CD | 0908 | C1D1 | CD47 | 08CD | A409 | 0138 | 8111 |G.....8.. |
| 1440 | 3BAA | CD47 | 083A | 2441 | FE88 | D231 | 09CD | 400B | ;..G.:\$A...1..@. |
| 1450 | C680 | C602 | DA31 | 09F5 | 21F8 | 07CD | 0B07 | CD41 |1..!.....A |
| 1460 | 08F1 | C1D1 | F5CD | 1307 | CD82 | 0921 | 7914 | CDA9 |!..... |
| 1470 | 1411 | 0000 | C14A | C347 | 0808 | 402E | 9474 | 704F |J.G..@....O |
| 1480 | 2E77 | 6E02 | 887A | E6A0 | 2A7C | 50AA | AA7E | FFFF |*.P..... |
| 1490 | 7F7F | 0000 | 8081 | 0000 | 0081 | CDA4 | 0911 | 320C |2. |
| 14A0 | D5E5 | CDBF | 09CD | 4708 | E1CD | A409 | 7E23 | CDB1 |G.....#.. |
| 14B0 | 0906 | F1C1 | D13D | C8D5 | C5F5 | E5CD | 4708 | E1CD |=.....G... |
| 14C0 | C209 | E5CD | 1607 | E118 | E9CD | 7F0A | 7CB7 | FA4A |J |
| 14D0 | 1EB5 | CAFO | 14E5 | CDF0 | 14CD | BF09 | EBE3 | C5CD | |
| 14E0 | CF0A | C1D1 | CD47 | 0821 | F807 | CD0B | 07C3 | 400B |G.I.....@. |
| 14F0 | 2190 | 40E5 | 1100 | 004B | 2603 | 2E08 | EB29 | EB79 | l.@....K&....) |

| | | | | | | | | | |
|------|------|------|------|------|------|------|------|------|--------------------|
| 1500 | 174F | E37E | 0777 | E3D2 | 1615 | E52A | AA40 | 19EB | .O!.....*.@.. |
| 1510 | 3AAC | 4089 | 4FE1 | 2DC2 | FC14 | E323 | E325 | C2FA | ::@.O.-.....#.\$.. |
| 1520 | 14E1 | 2165 | B019 | 22AA | 40CD | EF0A | 3E05 | 8932 | ..!....".@...>..2 |
| 1530 | AC40 | EB06 | 8021 | 2541 | 702B | 704F | 0600 | C365 | .@...!\$A.+..O.... |
| 1540 | 0721 | 8B15 | CD0B | 07CD | A409 | 0149 | 8311 | DB0F | ..!.....I.... |
| 1550 | CDB4 | 09C1 | D1CD | A208 | CDA4 | 09CD | 400B | C1D1 |@.... |
| 1560 | CD13 | 0721 | 8F15 | CD10 | 07CD | 5509 | 37F2 | 7715 | ...!.....U.7... |
| 1570 | CD08 | 07CD | 5509 | B7F5 | F482 | 0921 | 8F15 | CD0B |U.....!.... |
| 1580 | 07F1 | D482 | 0921 | 9315 | C39A | 14DB | 0F49 | 8100 |!.....I.. |
| 1590 | 0000 | 7F05 | BAD7 | 1E86 | 6426 | 9987 | 5834 | 2387 |&..X4#. |
| 15A0 | E05D | A586 | DA0F | 4983 | CDA4 | 09CD | 4715 | C1E1 |I.....G... |
| 15B0 | CDA4 | 09EB | CDB4 | 09CD | 4115 | C3A0 | 08CD | 5509 |A.....U. |
| 15C0 | FCE2 | 13FC | 8209 | 3A24 | 41FE | 8138 | 0C01 | 0081 |:\$A..8.... |
| 15D0 | 5159 | CDA2 | 0821 | 1007 | E521 | E315 | CD9A | 1421 | QY...!...!.....! |
| 15E0 | 8B15 | C909 | 4AD7 | 3B78 | 026E | 847B | FEC1 | 2F7C |J.;...../. |
| 15F0 | 7431 | 9A7D | 843D | 5A7D | C87F | 917E | E4BB | 4C7E | .1...=Z.....L. |
| 1600 | 6CAA | AA7F | 4444 | 4451 | 8A09 | 370B | 7709 | D427 |7.....' |
| 1610 | EF2A | F527 | E713 | C914 | 0908 | 3914 | 4115 | 4715 | .*.'.....9.A.G. |
| 1620 | A815 | BD15 | AA2C | 5241 | 5841 | 5E41 | 6141 | 6441 |,RAXA.A.A.A |
| 1630 | 6741 | 6A41 | 6D41 | 7041 | 7F0A | B10A | DB0A | 260B | .A.A.A.A.....&. |
| 1i40 | 032A | 3628 | C52A | 0F2A | 1F2A | 612A | 912A | 9A2A | .*6(*.*.*.*.*.* |
| 1650 | C54E | 44C6 | 4F52 | D245 | 5345 | 54D3 | 4554 | C34C | .ND.OR.ESET.ET.L |
| 1660 | 53C3 | 4D44 | D241 | 4E44 | 4F4D | CE45 | 5854 | C441 | S.MD.ANDOM.EXT.A |
| 1670 | 5441 | C94E | 5055 | 54C4 | 494D | D245 | 4144 | CC45 | TA.NPUT.IM.EAD.E |
| 1680 | 54C7 | 4F54 | 4FD2 | 554E | C946 | D245 | 5354 | 4F52 | T.OTO.UN.F.ESTOR |
| 1690 | 45C7 | 4F53 | 5542 | D245 | 5455 | 524E | D245 | 4DD3 | E.OSUB.ETURN.EM. |
| 16A0 | 544F | 50C5 | 4C53 | 45D4 | 524F | 4ED4 | 524F | 4646 | TOP.LSE.RON.ROFF |
| 16B0 | C445 | 4653 | 5452 | C445 | 4649 | 4E54 | C445 | 4653 | .EFSTR.EFINT.EFS |
| 16C0 | 4E47 | C445 | 4644 | 424C | CC49 | 4E45 | C544 | 4954 | NG.EFDBL.INE.DIT |
| 16D0 | C552 | 524F | 52D2 | 4553 | 554D | 45CF | 5554 | CF4E | .RROR.ESUME.UT.N |
| 16E0 | CF50 | 454E | C649 | 454C | 44C7 | 4554 | D055 | 54C3 | .PEN.IELD.ET.UT. |
| 16F0 | 4C4F | 5345 | CC4F | 4144 | CD45 | 5247 | 45CE | 414D | LOSE.OAD.ERGE.AM |
| 1700 | 45CB | 494C | 4CCC | 5345 | 54D2 | 5345 | 54D3 | 4156 | E.ILL.SET.SET.AV |
| 1710 | 45D3 | 5953 | 5445 | 4DCC | 5052 | 494E | 54C4 | 4546 | E.YSTEM.PRINT.EF |
| 1720 | D04F | 4B45 | D052 | 494E | 54C3 | 4F4E | 54CC | 4953 | .OKE.RINT.ONT.IS |
| 1730 | 54CC | 4C49 | 5354 | C445 | 4C45 | 5445 | C155 | 544F | T.LIST.ELETE.UTO |
| 1740 | C34C | 4541 | 52C3 | 4C4F | 4144 | C353 | 4156 | 45CE | .LEAR.LOAD.SAVE. |
| 1750 | 4557 | D441 | 4228 | D44F | C64E | D553 | 494E | 47D6 | EW.AB(.O.N.SING. |
| 1760 | 4152 | 5054 | 52D5 | 5352 | C552 | 4CC5 | 5252 | D354 | ARPTR.SR.RL.RR.T |
| 1770 | 5249 | 4E47 | 24C9 | 4E53 | 5452 | D04F | 494E | 54D4 | RING\$.NSTR.OINT. |
| 1780 | 494D | 4524 | CD45 | 4DC9 | 4E4B | 4559 | 24D4 | 4845 | IME\$.EM.NKEY\$.HE |
| 1790 | 4ECE | 4F54 | D354 | 4550 | ABAD | AAAF | DBC1 | 4E44 | N.OT.TEP.....ND |
| 17A0 | CF52 | BEBD | BCD3 | 474E | C94E | 54C1 | 4253 | C652 | .R....GN.NT.BS.R |
| 17B0 | 45C9 | 4E50 | D04F | 53D3 | 5152 | D24E | 44CC | 4F47 | E.NP.OS.QR.ND.OG |
| 17C0 | C558 | 50C3 | 4F53 | D349 | 4ED4 | 414E | C154 | 4ED0 | .XP.OS.IN.AN.TN. |
| 17D0 | 4545 | 4BC3 | 5649 | C356 | 53C3 | 5644 | C54F | 46CC | EEK.VI.VS.VD.OF. |
| 17E0 | 4F43 | CC4F | 46CD | 4B49 | 24CD | 4B53 | 24CD | 4B44 | OC.OF.KI\$.KS\$.KD |
| 17F0 | 24C3 | 494E | 54C3 | 534E | 47C3 | 4442 | 4CC6 | 4958 | \$.INT.SNG.DBL.IX |

| | | | | | | | | | |
|------|------|------|------|------|------|------|------|------|----------------------|
| 1800 | CC45 | 4ED3 | 5452 | 24D6 | 414C | C153 | 43C3 | 4852 | .EN.TR\$.AL.SC.HR |
| 1810 | 24CC | 4546 | 5424 | D249 | 4743 | 5424 | CD49 | 4424 | \$.EFT\$.IGHT\$.ID\$ |
| 1820 | A780 | AE1D | A11C | 3801 | 3501 | C901 | 7341 | D301 |8.5.....A.. |
| 1830 | B622 | 051F | 9A21 | 0826 | EF21 | 211F | C21E | A31E | ."....!.&!!..... |
| 1840 | 3920 | 91mD | B11E | DE1E | 071F | A91D | 071F | F71D | 9..... |
| 1850 | F81D | 001E | 031E | 061E | 091E | A341 | 602E | F41F |A..... |
| 1860 | AF1F | FB2A | 6C1F | 7941 | 7C41 | 7F41 | 8241 | 8541 | ...*...A.A.A.A.A |
| 1870 | 8841 | 8B41 | 8E41 | 9141 | 9741 | 9A41 | A041 | B202 | .A.A.A.A.A.A.A.. |
| 1880 | 6720 | 5B41 | B12C | 6F20 | E41D | 2E2B | 292B | C62B | ...A.,.....+)++.+ |
| 1890 | 0820 | 7A1E | 1F2C | F52B | 491B | 7979 | 7C7C | 7F50 |,+I.....P |
| 18A0 | 46DB | 4A41 | 447F | 0AF4 | 0AB1 | 0A77 | 0C70 | 0CA1 | F..... |
| 18B0 | 0DE5 | 0D78 | 0A16 | 0713 | 0747 | 08A2 | 080C | 0AD2 |G..... |
| 18C0 | 0BC7 | 0BF2 | 0B90 | 2439 | 0A4E | 4653 | 4E52 | 474F |\$9.NFSNRGO |
| 18D0 | 4446 | 434F | 564F | 4D55 | 4C42 | 5344 | 442F | 3049 | DFCOVOMULBSDD/OI |
| 18E0 | 4454 | 4D4F | 534C | 5353 | 5443 | 4E4E | 5252 | 5755 | DTMOSLSSTCNNRRWU |
| 18F0 | 454D | 4F46 | 444C | 33D6 | 006F | 7CDE | 0067 | 78DE | EMOFDL3..... |
| 1900 | 0047 | 3E00 | C94A | 1E40 | E64D | DB00 | C9D3 | 00C9 | .G>..J.@.M..... |
| 1910 | 4444 | 4444 | 4030 | 004C | 43FE | FFE9 | 4220 | 4572 |@0.LC...B.E. |
| 1920 | 726F | 7200 | 2069 | 6E20 | 0052 | 4541 | 4459 | 0D00 |READY.. |
| 1930 | 4272 | 6561 | 6B00 | 2104 | 0039 | 7E23 | FE81 | C04E | B.....!.9.#...N |
| 1940 | 2346 | 23E5 | 6960 | 7AB3 | EB28 | 02FB | DF01 | 0E00 | #F#.....(..... |
| 1950 | E1C8 | 0918 | E5CD | 6C19 | C5E3 | C1DF | 7E02 | C80B | |
| 1960 | 2B18 | F8E5 | 2AFD | 4006 | 0009 | 093E | E53E | C695 | +...*.@....>.>.. |
| 1970 | 6F3E | FF9C | 3804 | 6739 | E1D8 | 1E0C | 1824 | 2AA2 | .>..8..9.....\$*. |
| 1980 | 407C | A53C | 2808 | 3AF2 | 40B7 | 1E22 | 2014 | C3C1 | @..<(.:@.."..... |
| 1990 | 1D2A | DA40 | 22A2 | 401E | 0201 | 1E14 | 011E | 0001 | .*@".@..... |
| 19A0 | 1E24 | 2AA2 | 4022 | EA40 | 22EC | 4001 | B419 | 2AE8 | .\$*@"@"@...* |
| 19B0 | 40C3 | 9A1B | C17B | 4B32 | 9A40 | 2AE6 | 4022 | EE40 | @.....K2.@*@"@ |
| 19C0 | EB2A | EA40 | 7CA5 | 3C28 | 0722 | F540 | EB22 | F740 | .*@..<(."@."@ |
| 19D0 | 2AF0 | 407C | B5EB | 21F2 | 4028 | 08A6 | 2005 | 35EB | *.@...!.@(...5. |
| 19E0 | C336 | 1DAF | 7759 | CDF9 | 2021 | C918 | CDA6 | 4157 | .6...Y...!....AW |
| 19F0 | 3E3F | CD2A | 0319 | 7ECD | 2A03 | D7CD | 2A03 | 211D | >?.*...*...*!. |
| 1A00 | 19E5 | 2AEA | 40E3 | CDA7 | 28E1 | 11FE | FFDF | CA74 | ..*.@... (..... |
| 1A10 | 067C | A53C | C4A7 | 0F3E | C1CD | 8B03 | CDAC | 41CD | ...<...>...A. |
| 1A20 | F801 | CDF9 | 2021 | 2919 | CDA7 | 283A | 9A40 | D602 |!)...(:@.. |
| 1A30 | CC53 | 2E21 | FFFF | 22A2 | 403A | E140 | B728 | 372A | .S.!.."@:..@.(7* |
| 1A40 | E240 | E5CD | AF0F | D1D5 | CD2C | 1B3E | 2A38 | 023E | @.....,>*8.> |
| 1A50 | 20CD | 2A03 | CD61 | 03D1 | 3006 | AF32 | E140 | 18B9 | ..*.....0..2.@.. |
| 1A60 | 2AE4 | 4019 | 38F4 | D511 | F9FF | DFD1 | 30EC | 22E2 | *.@.8.....0." |
| 1A70 | 40F6 | FFC3 | EB2F | 3E3E | CD2A | 03CD | 6103 | DA33 | @...../>>.*.....3 |
| 1A80 | 1AD7 | 3C3D | CA33 | 1AF5 | CD5A | 1E2B | 7EFE | 2028 | ..<=.3...Z.+... (|
| 1A90 | FA23 | 7EFE | 20CC | C909 | D5CD | C01B | D1F1 | 22E6 | .#....." |
| 1AA0 | 40CD | B241 | D25A | 1DD5 | C5AF | 32DD | 40D7 | B7F5 | @..A.Z.....2.@... |
| 1AB0 | EB22 | EC40 | EBCD | 2C1B | C5DC | E42B | D1F1 | D528 | ."@.....+... (|
| 1AC0 | 27D1 | 2AF9 | 40E3 | C109 | E5CD | 5519 | E122 | F940 | '.*@.....U..".@ |
| 1AD0 | EB74 | D1E5 | 2323 | 7323 | 7223 | EB2A | A740 | EB1B |##.#.#.*@.. |
| 1AE0 | 1B1A | 7723 | 13B7 | 20F9 | D1CD | FC1A | CDB5 | 41CD |#.....A. |
| 1AF0 | 5D1B | CDB8 | 41C3 | 331A | 2AA4 | 40EB | 626B | 7E23 |A.3.*@.....# |

| | | | | | | | | | |
|------|-----------|------|------|------|------|------|------------|---------------------|---------------------|
| 1B00 | B6C8 | 2323 | 23AF | BE23 | 20FC | EB73 | 2372 | 18EC | ..###..#.....#..... |
| 1B10 | 1185 74D5 | 2809 | D1CD | 4F1E | D528 | 0BCF | CE11 |(....O....(.... | |
| 1B20 | FAFF | C44F | 1EC2 | 9719 | EBD1 | E3E5 | 2AA4 | 4044 | ...O.....*.@D |
| 1B30 | 4D7L | 23B6 | 2BC8 | 2323 | 7E23 | 666F | DF60 | 697E | M.#.+##.#..... |
| 1B40 | 2366 | 6F3F | C83F | D018 | E6C0 | CDC9 | 012A | A440 | #..??.#.....*.@ |
| 1B50 | CDF8 | 1D32 | E140 | 7723 | 7723 | 22F9 | 402A | A440 | ...2.@.#.#".@*.@ |
| 1B60 | 2B22 | DF40 | 061A | 2101 | 4136 | 0423 | 10FB | AF32 | +".@..!.A6.#...2 |
| 1B70 | F240 | 6F67 | 22F0 | 4022 | F740 | 2AB1 | 4022 | D640 | .@..".@".@*.@".@ |
| 1B80 | CD91 | 1D2A | F940 | 22FB | 4022 | FD40 | CDBB | 41C1 | ...*.@"@"@"@.A. |
| 1B90 | 2AA0 | 402B | 2B22 | E840 | 2323 | F921 | B540 | 22B3 | *.@++".@##.!.@". |
| 1BA0 | 40CD | 8B03 | CD69 | 21AF | 676F | 32DC | 40E5 | C52A | @.....!.#...2.@..* |
| 1BB0 | DF40 | C93E | 3FCD | 2A03 | 3E20 | CD2A | 03C3 | 6103 | .@.>?.*.>..*.... |
| 1BC0 | AF32 | B040 | 4FEB | 2AA7 | 402B | 2BEB | 7EFE | 20CA | .2.@O.*.@++..... |
| 1BD0 | 5B1C | 47FE | 22CA | 771C | B7CA | 7D1C | 3AB0 | 40B7 | ..G.".....:.@. |
| 1BE0 | 7EC2 | 5B1C | FE3F | 3EB2 | CA5B | 1C7E | FE30 | 38s5 |?>.....08. |
| 1BF0 | FE3C | DA5B | 1CD5 | 114F | 16C5 | 013D | 1CC5 | 067F | .<.....O....=..... |
| 1C00 | 7EFE | 6138 | 07FE | 7B30 | 03E6 | 5F77 | 4EEB | 23B6 | ...8...0....N.#. |
| 1C10 | F20E | 1C04 | 7EE6 | 7FC8 | B920 | F3EB | E513 | 1AB7 | |
| 1C20 | FA39 | 1C4F | 78FE | 8D20 | 02D7 | 2B23 | 7EFE | 6138 | .9.O.....+#...8 |
| 1C30 | 02E6 | 5FE9 | 28E7 | E118 | D348 | F1EB | C9EB | 79C1 |(....H..... |
| 1C40 | D1EB | FE95 | 363A | 2002 | 0C23 | FEFB | 200C | 363A |6:....#....6: |
| 1C50 | 2306 | 9370 | 23EB | 0C0C | 181D | EB23 | 1213 | 0CD6 | #...#.....#..... |
| 1C60 | 3A28 | 04FE | 4E20 | 0332 | B040 | D659 | C2CC | 1B47 | :(..N..2.@.Y...G |
| 1C70 | 7EB7 | 2809 | B828 | E423 | 120C | 1318 | F321 | 0500 | ..(..(#.....!.. |
| 1C80 | 4409 | 444D | 2AA7 | 402B | 2B2B | 1213 | 1213 | 12C9 | D.DM*.@+++..... |
| 1C90 | 7C92 | C07D | 93C9 | 7EE3 | BE23 | E3CA | 781D | C397 |#..... |
| 1CA0 | 193E | 6432 | DC40 | CD21 | 1FE3 | CD36 | 19D1 | 2005 | .>.2.@.!.#...6.... |
| 1CB0 | 09F9 | 22E8 | 40EB | 0E08 | CD63 | 19E5 | CD05 | 1FE3 | ..".@..... |
| 1CC0 | E52A | A240 | E3CF | BDE7 | CAF6 | 0AD2 | F60A | F5CD | *.@..... |
| 1CD0 | 3723 | F1E5 | F2EC | 1CCD | 7F0A | E311 | d144 7E FE | 7# | |
| 1CE0 | CCCC | 012B | D5E5 | EBCD | 9E09 | 1822 | CDB1 | 0ACD | ...+....."..... |
| 1CF0 | BF09 | E1C5 | D501 | 0081 | 515A | 7EFE | CC3E | 0120 |QZ...>.. |
| 1D00 | 0ECD | 3823 | E5CD | B10A | CDBF | 09CD | 5509 | E1C5 | ..8#.....U... |
| 1D10 | D54F | E747 | C5E5 | 2ADF | 40E3 | 0681 | C53v | CD58 | .O.G.*.@...3.X |
| 1D20 | 03B7 | C4A0 | 1D22 | E640 | ED73 | E840 | 7EFE | 3A28 |".@...@...:(|
| 1D30 | 29B7 | C297 | 1923 | 7E23 | B6CA | 7E19 | 235E | 2356 |).....#.#.....#.#V |
| 1D40 | EB22 | A240 | 3A1B | 41B7 | 280F | D53E | 3CCD | 2A03 | ..".@:.A.(..><.*. |
| 1D50 | CDAF | 0F3E | 3ECD | 2A03 | D1EB | D711 | 1E1D | D5C8 | ...>>.*..... |
| 1D60 | D680 | DA21 | 1FFE | 3CD2 | E72A | 074F | 0600 | EB21 | ...!..<..*.O...! |
| 1D70 | 2218 | 094E | 2346 | C5EB | 237E | FE3A | D0FE | 20CA | "..N#F..#...:.... |
| 1D80 | 781D | FE0B | 3005 | FE09 | D278 | 1DFE | 303F | 3C3D |0.....0?<= |
| 1D90 | C9EB | 2AA4 | 402B | 22FF | 40EB | C9CD | 5803 | B7C8 | ..*.@+"@...X... |
| 1DA0 | FE60 | CC84 | 0332 | 9940 | 3DC0 | 3CC3 | B41D | C0F5 |2.@=.<..... |
| 1DB0 | CCBB | 41F1 | 22E6 | 4021 | B540 | 22B3 | 4021 | F6FF | ..A."@!.@"@!.. |
| 1DC0 | C12A | A240 | E5F5 | 7DA4 | 3C28 | 0922 | F540 | 2AE6 | *.@.....<("@"@*. |
| 1DD0 | 4022 | F740 | CD8B | 03CD | F920 | F121 | 3019 | C206 | @".@.....!0... |
| 1DE0 | 1AC3 | 181A | 2AF7 | 407C | B51E | 20CA | A219 | EB2A |*.@.....* |
| 1DF0 | F540 | 22A2 | 40EB | C93E | AF32 | 1B41 | C9F1 | E1C9 | .@"@..>.2.A.... |

| | | | | | | | | | |
|------|------|------|------|------|------|------|------|------|------------------------|
| 1E00 | 1E03 | 011E | 0201 | 1E04 | 011E | 08CD | 3D1E | 0197 |=... |
| 1E10 | 19C5 | D8D6 | 414F | 47D7 | FECE | 2009 | D7CD | 3D1E |AOG.....=. |
| 1E20 | D8D6 | 4147 | D778 | 91D8 | 3CE3 | 2101 | 4106 | 0009 | ..AG....<!.A... |
| 1E30 | 7323 | 3D20 | FBE1 | 7EFE | 2CC0 | D718 | CE7E | FE41 | ..#=.,.....A |
| 1E40 | D8FE | 5B3F | C9D7 | CD02 | 2BF0 | 1E08 | C3A2 | 197E | ...?.+..... |
| 1E50 | FE2E | EB2A | EC40 | EBCA | 781D | 2B11 | 0000 | D7D0 | ...*. @.....+..... |
| 1E60 | E5F5 | 2198 | 19DF | DA97 | 1962 | 6B19 | 2919 | 29F1 | ..!.).). |
| 1E70 | D630 | 5F16 | 0019 | EBE1 | 18E4 | CA61 | 1BCD | 461E | .0.....F. |
| 1E80 | 2BD7 | C0E5 | 2AB1 | 407D | 935F | 7C9A | 57DA | 7A19 | +...*. @.....W... |
| 1E90 | 2AF9 | 4001 | 2844 | 01DF | D27A | 19EB | 22A0 | 40E1 | *. @. (.....". @. |
| 1EA0 | C361 | 1BCA | 5D1B | CDC7 | 41CD | 611B | 011E | 1D18 |A..... |
| 1EB0 | 100E | 03CD | 6319 | C1E5 | E52A | A240 | E33E | 91F5 |*. @.>.. |
| 1EC0 | 33C5 | CD5A | 1ECD | 071F | E52A | A240 | DFE1 | 23DC | 3..Z.....*. @..#. |
| 1ED0 | 2F1B | D42C | 1B60 | 692B | D81E | 0EC3 | A219 | C016 | /.....+..... |
| 1EE0 | FFCD | 3619 | F922 | E840 | FE91 | 1E04 | C2A2 | 19E1 | ..6..". @..... |
| 1EF0 | 22A2 | 4023 | 7CB5 | 2007 | 3ADD | 40B7 | C218 | 1A21 | ". @#.....: @.....! |
| 1F00 | 1E1D | E33E | E101 | 3144 | 0706 | 0479 | 4847 | 7EB7 | ...>.....HG.. |
| 1F10 | C8B8 | C823 | FE22 | 28F3 | D68F | 20F2 | B88A | 5718 | ...#. "(.....W. |
| 1F20 | EDCD | 0D26 | CFD5 | EB22 | DF40 | EBD5 | E7F5 | CD37 | ...&...". @.....7 |
| 1F30 | 23F1 | E3C6 | 03CD | 1928 | CD03 | 0AE5 | 2028 | 2A21 | #.....(.....(*! |
| 1F40 | 41E5 | 235E | 2356 | 2AA4 | 40DF | 300E | 2AA0 | 40DF | A.#.#V*. @.0.*. @. |
| 1F50 | D130 | 0F2A | F940 | DF30 | 093E | D1CD | F529 | EBCD | .0.*. @.0.>.....) |
| 1F60 | 4328 | CDF5 | 29E3 | CDD3 | 09D1 | E1C9 | FE9E | 2025 | C(..).....% |
| 1F70 | D7CF | 8DCD | 5A1E | 7AB3 | 2809 | CD2A | 1B50 | 59E1 |Z... (. *. PY. |
| 1F80 | D2D9 | 1EEB | 22F0 | 40EB | D83A | F240 | B7C8 | 3A9A |". @.....@..... |
| 1F90 | 405F | C3AB | 19CD | 1C2B | 7E47 | FE91 | 2803 | CF8D | @.....+. G.. (... |
| 1FA0 | 2B4B | 0D78 | CA60 | 1DCD | 5B1E | FE2C | C018 | F311 | +K.....,..... |
| 1FB0 | F240 | 1AB7 | CAA0 | 193C | 329A | 4012 | 7EFE | 8728 | . @.....<2. @.....(|
| 1FC0 | 0CCD | 5A1E | C07A | B3C2 | C51E | 3C18 | 02D7 | C02A | ..Z.....<.....* |
| 1FD0 | EE40 | EB2A | EA40 | 22A2 | 40EB | C07E | B720 | 0423 | . @.*. @". @.....# |
| 1FE0 | 2323 | 2323 | 7AA3 | 3CC2 | 051F | 3ADD | 403D | CABE | ####<.....: @=.. |
| 1FF0 | 1DC3 | 051F | CD1C | 2BC0 | B7CA | 4A1E | 3D87 | 5FFE |+.....J.=... |
| 2000 | 2D38 | 021E | 26C3 | A219 | 110A | 0405 | 2817 | CD4F | -8...&.....(..O |
| 2010 | 1EEB | E328 | 11EB | CF2C | EB2A | E440 | EB28 | 06CD | ... (....., *. @. (... |
| 2020 | 5A1E | C297 | 19EB | 7CB5 | CA4A | 1E22 | E440 | 32E1 | Z.....J." @2. |
| 2030 | 40E1 | 22E2 | 40C1 | C333 | 1ACD | 3723 | 7EFE | 2CCC | @." @..3..7#...,, |
| 2040 | 781D | FECA | CC78 | 1D2B | E5CD | 9409 | E128 | 07D7 |+.....(.. |
| 2050 | DAC2 | 1EC3 | 5F1D | 1601 | CD05 | 1FB7 | C8D7 | FE95 | |
| 2060 | 20F6 | 1520 | F318 | E83E | 0132 | 9C40 | C39B | 20CD |>.2. @..... |
| 2070 | CA41 | FE40 | 2019 | CD01 | 2BFE | 04D2 | 4A1E | E521 | .A. @.....+...J..! |
| 2080 | 003C | 1922 | 2040 | 7BE6 | 3F32 | A640 | E1CF | 2CFE | .<." @..?2. @..... |
| 2090 | 2320 | 08CD | 8402 | 3E80 | 329C | 402B | D7CC | FE20 | #.....>.2. @+..... |
| 20A0 | CA69 | 21FE | BFCA | BD2C | FEBC | CA37 | 21E5 | FE2C | ..!.....,....7!..., |
| 20B0 | CA08 | 21FE | 3BCA | 6421 | C1CD | 3723 | E5E7 | 2832 | ..!.;...!..7#..(2 |
| 20C0 | CDBD | 0FCD | 6528 | CDCD | 412A | 2141 | 3A9C | 40B7 |(..A*!A:..@. |
| 20D0 | FAE9 | 2028 | 083A | 9B40 | 86FE | 8418 | 093A | 9D40 | ... (.: @.....: @ |
| 20E0 | 473A | A640 | 86B8 | D4FE | 20CD | AA28 | 3E20 | CD2A | G: @..... (>..* |
| 20F0 | 03B7 | CCAA | 28E1 | C39B | 203A | A640 | B7C8 | 3E0D | (.....: @..>. |

| | | | | | | | | | |
|------|------|------|------|------------------|------|-----------------------|-------|---------------|-------------------|
| 2100 | CD2A | 03CD | D041 | AFC9 | CDD3 | 413A | 9C40 | B7F2 | .*.A.A:@.. |
| 2110 | 1921 | 3E2C | CD2A | 0318 | 4B28 | 083A | 9B40 | FE70 | !>,*.K(..@.. |
| 2120 | C32B | 213A | 9E40 | 473A | A640 | B8D4 | FE20 | 3034 | .+!:@G:@...04 |
| 2130 | D610 | 30FC | 2F18 | 23CD | 1B2B | E63F | 5FCF | 292B | ..0./.#..+?.)+ |
| 2140 | E5CD | D341 | 3A9C | 40B7 | FA4A | 1ECA | 5321 | 3A9B | ...A:@..J..S!: |
| 2150 | 4018 | 033A | A640 | 2F83 | 300A | 3C47 | 3E20 | CD2A | @...:/0.<G>.* |
| 2160 | 0305 | 20FA | E1D7 | C3A0 | 203A | 9C40 | B7FC | F801 |@..... |
| 2170 | AF32 | 9C40 | CDBE | 41C9 | 3F52 | 4544 | 4F0D | 003A | .2.@..A.?REDO..: |
| 2180 | DE40 | B7C2 | 9119 | 3AA9 | 40B7 | 1E2A | CAA2 | 19C1 | .@.....@..*..... |
| 2190 | 2178 | 21CD | A728 | 2AE6 | 40C9 | CD28 | 287E | CDD6 | !.!..(*.@..((...) |
| 21A0 | 41D6 | 2332 | A940 | 7E20 | 20CD | 9302 | E506 | FA2A | A.#2.@.....*.. |
| 21B0 | A740 | CD35 | 0277 | 23FE | 0D28 | 0210 | F52B | 3600 | .@.5..#..(+6.. |
| 21C0 | CD66 | 012A | A740 | 2B18 | 2201 | DB21 | C5FE | 22C0 | ...*.@+."..l.." |
| 21D0 | CD66 | 28CF | 3BE5 | CDA4 | 28E1 | C9E5 | CDB3 | 1BC1 | ..(;...(... |
| 21E0 | DABE | 1D23 | 7EB7 | 2BC5 | CA04 | 1F36 | 2C18 | 05E5 | ...#..+.....6,.. |
| 21F0 | 2AFF | 40F6 | AF32 | DE40 | E318 | 02CF | 2CCD | 0D26 | *.@..2.@.....,& |
| 2200 | E3D5 | 7EFE | 2C28 | 263A | DE40 | B7C2 | 9622 | 3AA9 |(&:@...": |
| 2210 | 40B7 | 1E06 | CAA2 | 193E | 3FCD | 2A03 | CDB3 | 1BD1 | @.....>?.*..... |
| 2220 | C1DA | BE1D | 237E | B72B | C5CA | 041F | D5CD | DC41 | ...#..+.....A |
| 2230 | E7F5 | 2019 | D757 | 47FE | 2228 | 0516 | 3A06 | 2C2B |WG."(..:,+ |
| 2240 | CD69 | 28F1 | EB21 | 5A22 | E3D5 | C333 | 1FD7 | F1F5 | ..(!Z"...3.... |
| 2250 | 0143 | 22C5 | DA6C | 0ED2 | 650E | 2BD7 | 2805 | FE2C | .C".....+.(,.. |
| 2260 | C27F | 21E3 | 2BD7 | C2FB | 21D1 | <i>φφφφ φφφφ φφ3φ</i> | | | ..!+...!.....: |
| 2270 | DE40 | B7EB | C296 | 1DD5 | CDDF | 41B6 | 2186 | 22C4 | @.....A.!." |
| 2280 | A728 | E1C3 | 6921 | 3F45 | 7874 | 7261 | 2069 | 676E | .(...!E..... |
| 2290 | 6F72 | 6564 | 0D00 | CD05 | 1FB7 | 2012 | 237E | 23B6 |#.#. |
| 22A0 | 1E06 | CAA2 | 1923 | 5E23 | 56EB | 22DA | 40EB | D7FE |#.#V."@... |
| 22B0 | 8820 | E3C3 | 2D22 | <i>11φφ φφc4</i> | 0D26 | 22DF | 40CD |-".&".@. | |
| 22C0 | 3619 | C29D | 19F9 | 22E8 | 40D5 | 7E23 | F5D5 | 7E23 | 6.....".@..#...# |
| 22D0 | B7FA | EA22 | CDB1 | 09E3 | E5CD | 0B07 | E1CD | CB09 | ...". |
| 22E0 | E1CD | C209 | E5CD | 0C0A | 1829 | 2323 | 2323 | 4E23 |)###N# |
| 22F0 | 4623 | E35E | 2356 | E569 | 60CD | D20B | 3AAF | 40FE | F#..#V.....@. |
| 2300 | 04CA | B207 | EBE1 | 722B | 73E1 | D55E | 2356 | 23E3 |+.....#V#. |
| 2310 | CD39 | 0AE1 | C190 | CDC2 | 0928 | 09EB | 22A2 | 4069 | .9.....(".@. |
| 2320 | 60C3 | 1A1D | F922 | E840 | 2ADF | 407E | FE2C | C21E |".@*.@.,.. |
| 2330 | 1DD7 | CDB9 | 22CF | 282B | 1600 | D50E | 01CD | 6319 |".(+..... |
| 2340 | CD9F | 2422 | F340 | 2AF3 | 40C1 | 7E16 | 00D6 | D438 | ..\$".@*.@.....8 |
| 2350 | 13FE | 0330 | 0FFE | 0117 | AABA | 57DA | 9719 | 22D8 | ...0.....W..." |
| 2360 | 40D7 | 18E9 | 7AB7 | C2EC | 237E | 22D8 | 40D6 | CDD8 | @.....#."@... |
| 2370 | FE07 | D05F | 3AAF | 40D6 | 03B3 | CA8F | 2921 | 9A18 |@.....)!.. |
| 2380 | 1978 | 56BA | D0C5 | 0146 | 23C5 | 7AFE | 7FCA | D423 | ..V....F#.....# |
| 2390 | FE51 | DAE1 | 2321 | 2141 | B73A | AF40 | 3D3D | 3DCA | .Q..#!A...@==. |
| 23A0 | F60A | 4E23 | 46C5 | FAC5 | 2323 | 4E23 | 46C5 | F5B7 | ..N#F...#N#F... |
| 23B0 | E2C4 | 23F1 | 2338 | 0321 | 1D41 | 4E23 | 4623 | C54E | ..#.#8.!AN#F#N |
| 23C0 | 2346 | C506 | F1C6 | 034B | 47C5 | 0106 | 24C5 | 2AD8 | #F.....KG...\$.* |
| 23D0 | 40C3 | 3A23 | CDB1 | 0ACD | A409 | 01F2 | 1316 | 7F18 | @.:#..... |
| 23E0 | ECD5 | CD7F | 0AD1 | E501 | E925 | 18E1 | 78FE | 64D0 |&..... |
| 23F0 | C5D5 | 1104 | 6421 | B825 | E5E7 | C295 | 232A | 2141 |l.%.....#!A |

| | | | | | | | | | |
|------|------|------|------|------|------|------|------|------|--------------------|
| 2400 | E501 | 8C25 | 18C7 | C179 | 32B0 | 4078 | FE08 | 2828 | ..!%...2.@...((|
| 2410 | 3AAF | 40FE | 08CA | 6024 | 5778 | FE04 | CA72 | 247A | ::@...\$W...\$. |
| 2420 | FE03 | CAF6 | 0AD2 | 7C24 | 21BF | 1806 | 0009 | 094E |\$!.....N |
| 2430 | 2346 | D12A | 2141 | C5C9 | CDD8 | 0ACD | FC09 | E122 | #F.*!A....." |
| 2440 | 1F41 | E122 | 1D41 | C1D1 | CDB4 | 09CD | DB0A | 21AB | .A."A.....!. |
| 2450 | 183A | B040 | 07C5 | 4F06 | 44C1 | C17E | 2366 | 6FE9 | ::@..O.....#... |
| 2460 | C5CD | FC09 | F132 | AF40 | FE04 | 28DA | E122 | 2141 |2.@..(.."!A |
| 2470 | 18D9 | CDB1 | 0AC1 | D121 | B518 | 18D5 | E1CD | A409 |!..... |
| 2480 | CDCF | 0ACD | BF09 | E122 | 2341 | E122 | 2141 | 18E7 |"#A."!A.. |
| 2490 | E5EB | CDCF | 0AE1 | CDA4 | 09CD | CF0A | C3A0 | 08D7 | |
| 24A0 | 1E28 | CAA2 | 19DA | 6C0E | CD3D | 1ED2 | 4025 | FECD | .(.....=@%.. |
| 24B0 | 28ED | FE2E | CA6C | 0EFE | CECA | 3225 | FE22 | CA66 | (.....2%.".. |
| 24C0 | 28FE | CBCA | C425 | FE26 | CA94 | 41FE | C320 | 0AD7 | (.....%&..A..... |
| 24D0 | 3A9A | 40E5 | CDF8 | 27E1 | C9FE | C220 | 0AD7 | E52A | ::@...!.....* |
| 24E0 | EA40 | CD66 | 0CE1 | C9FE | C020 | 14D7 | CF28 | CD0D | .@.....(.. |
| 24F0 | 26CF | 29E5 | EB7C | B5CA | 4A1E | CD9A | 0AE1 | C9FE | &.).....J..... |
| 2500 | C1CA | FE27 | FEC5 | CA9D | 41FE | C8CA | C927 | FEC7 | ...'.....A.....' |
| 2510 | CA76 | 41FE | C6CA | 3201 | FEC9 | CA9D | 01FE | C4CA | ..A...2..... |
| 2520 | 2F2A | FEBE | CA55 | 41D6 | D7D2 | 4E25 | CD35 | 23CF | /*...UA...N%5#. |
| 2530 | 29C9 | 167D | CD3A | 232A | F340 | E5CD | 7B09 | E1C9 |).....:##*@..... |
| 2540 | CD0D | 26E5 | EB22 | 2141 | E7C4 | F709 | E1C9 | 0600 | ..&.."!A..... |
| 2550 | 074F | C5D7 | 79FE | 4138 | 16CD | 3523 | CF2C | CDF4 | .O....A8..5#.,.. |
| 2560 | 0AEB | 2A21 | 41E3 | E5EB | CD1C | 2BEB | E318 | 14CD | ..*!A.....+..... |
| 2570 | 2C25 | E37D | FE0C | 3807 | FE1B | E5DC | B10A | E111 | ,%.....8..... |
| 2580 | 3E25 | D501 | 0816 | 094E | 2366 | 69E9 | CDD7 | 297E | >%.....N#.....). |
| 2590 | 234E | 2346 | D1C5 | F5CD | DE29 | D15E | 234E | 2346 | #N#F.....).#N#F |
| 25A0 | E17B | B2C8 | 7AD6 | 01D8 | AFBB | 3CD0 | 151D | 0ABE |<..... |
| 25B0 | 2303 | 28ED | 3FC3 | 6009 | 3C8F | C1A0 | C6FF | 9FCD | #.(?...<..... |
| 25C0 | 8D09 | 1812 | 165A | CD3A | 23CD | 7F0A | 7D2F | 6F7C |Z.:#...../.. |
| 25D0 | 2F67 | 2221 | 41C1 | C346 | 233A | AF40 | FE08 | 3005 | /.!"!A..F#::@..0. |
| 25E0 | D603 | B737 | C9D6 | 03B7 | C9C5 | CD7F | 0AF1 | D101 | ...7..... |
| 25F0 | FA27 | C5FE | 4620 | 067B | B56F | 7CB2 | C97B | A56F | !'..F..... |
| 2600 | 7CA2 | C92B | D7C8 | CF2C | 0103 | 26C5 | F6AF | 32AE | ...+...&...2. |
| 2610 | 4046 | CD3D | 1EDA | 9719 | AF4F | D738 | 05CD | 3D1E | @F.=.....O.8..=. |
| 2620 | 3809 | 4FD7 | 38FD | CD3D | 1E30 | F811 | 5226 | D516 | 8.O.8..=.O..R&.. |
| 2630 | 02FE | 25C8 | 14FE | 24C8 | 14FE | 21C8 | 1608 | FE23 | ..%...\$...!.....# |
| 2640 | C878 | D641 | E67F | 5F16 | 00E5 | 2101 | 4119 | 56E1 | ...A.....!..A.V. |
| 2650 | 2BC9 | 7A32 | AF40 | D73A | DC40 | B7C2 | 6426 | 7ED6 | +..2.@::@...&.. |
| 2660 | 28CA | E926 | AF32 | DC40 | E5D5 | 2AF9 | 40EB | 2AFB | (..&2.@..*.@.*. |
| 2670 | 40DF | E128 | 191A | 6FBC | 1320 | 0B1A | B920 | 0713 | @..(..... |
| 2680 | 1AB8 | CACC | 263E | 1313 | E524 | 9419 | 18DF | 7CE1 |&>...&..... |
| 2690 | E3F5 | D511 | F124 | DF28 | 3611 | 4325 | DFD1 | 2835 |\$. (6.C%..(5 |
| 26A0 | F1E3 | E5C5 | 4F06 | 00C5 | 0303 | 032A | FD40 | E509 |O.....*.@.. |
| 26B0 | C1E5 | CD55 | 19E1 | 22FD | 4060 | 6922 | FB40 | 2B36 | ...U..".@..".@+6 |
| 26C0 | 00DF | 20FA | D173 | 23D1 | 7323 | 72EB | 13E1 | C957 |#.#.....W |
| 26D0 | 5FF1 | F1E3 | C932 | 2441 | C167 | 6F22 | 2141 | E720 |2\$A...!"!A.. |
| 26E0 | 0621 | 2819 | 2221 | 41E1 | C9E5 | 2AAE | 40E3 | 57D5 | !(!!"!A...*.@.W. |
| 26F0 | C5CD | 451E | C1F1 | EBE3 | E5EB | 3C57 | 7EFE | 2C28 | ..E.....<W...,(|

2680

| | | | | | | | | | |
|------|------|------|------|------|-----------|-----------|-----------|------|--------------------|
| 2700 | EECF | 2922 | F340 | E122 | AE40 | D52A | FB40 | 3E19 | ..).".@."@.*.@>. |
| 2710 | EB2A | FD40 | EBDF | 3AAF | 4028 | 27BE | 2320 | 087E | .*.@...@('.#... |
| 2720 | B923 | 2004 | 7EB8 | 3E23 | 235E | 2356 | 2320 | E03A | .#...>##.#V#...: |
| 2730 | AE40 | B71E | 12C2 | A219 | F196 | CA95 | 271E | 10C3 | .@.....'.... |
| 2740 | A219 | 7723 | 5F16 | 00F1 | 7123 | 7023 | 4FCD | 6319 | ...#.....#.#O... |
| 2750 | 2323 | 22D8 | 4071 | 233A | AE40 | 1779 | 010B | 0030 | ##"@.#:..@.....0 |
| 2760 | 02C1 | 0371 | 2370 | 23F5 | CDAA | 0BF1 | 3D20 | EDF5 | ...#.#.....=... |
| 2770 | 424B | EB19 | 38C7 | CD6C | 1922 | FD40 | 2B36 | 00DF | BK..8.....".@+6.. |
| 2780 | 20FA | 03'7 | 2AD8 | 405E | EB29 | 09EB | 2B2B | 7323 | ...W*.@..)..++.# |
| 2790 | 7223 | F138 | 3047 | 4F7E | 2316 | E15E | 2356 | 23E3 | .#.#80GO.#...#V#. |
| 27A0 | F5DF | D23D | 27CD | AA0B | 19F1 | 3D44 | 4D20 | EB3A | ...='.....=DM...: |
| 27B0 | AF40 | 444D | 29D6 | 0438 | 0429 | 2806 | 29B7 | E2C2 | .@DM)..8.)(..) |
| 27C0 | 2709 | C109 | EB2A | F340 | C9AF | E532 | AF40 | CDD4 | '.....*.a....2.@.. |
| 27D0 | 27E1 | D7C9 | 2AFD | 40EB | 2146 4439 | E720 (DC) | | | '...*.@.!..9.... |
| 27E0 | DA29 | CDE6 | 282A | A040 | EB2A | D640 | 7D93 | 6F7C | ..).(*.@.*.@..... |
| 27F0 | 9A67 | C366 | 0C3A | A640 | 6FAF | 67C3 | 9A0A | CDA9 |:@..... |
| 2800 | 41D7 | CD2C | 25E5 | 2190 | 08E5 | 3AAF | 40F5 | FE03 | A.,%,!.....:@... |
| 2810 | CCDA | 29F1 | EB2A | 8E40 | E9E5 | E607 | 21A1 | 184F | ..).*.@.....!..O |
| 2820 | 0600 | 09CD | 8625 | E1C9 | E52A | A240 | 237C | B5E1 |&...*.@#... |
| 2830 | C01E | 16C3 | A219 | CDBD | 0FCD | 6528 | CDDA | 2901 |(..). |
| 2840 | 2B2A | C57E | 23E5 | CDBF | 28E1 | 4E23 | 46CD | 5A28 | +*..#...(.N#F.Z(|
| 2850 | E56F | CDCE | 29D1 | C9CD | BF28 | 21D3 | 40E5 | 7723 |)(!.@..# |
| 2860 | 7323 | 72E1 | C92B | 0622 | 50E5 | 0EFF | 237E | 0CB7 | .#...+."P...#... |
| 2870 | 2806 | BA28 | 03B8 | 20F4 | FE22 | CC78 | 1DE3 | 23EB | (..(.....".....#. |
| 2880 | 79CD | 5A28 | 11D3 | 403E | D52A | B340 | 2221 | 413E | ..Z(..@>.*.@"!A> |
| 2890 | 0332 | AF40 | CDD3 | 0911 | D640 | DF22 | B340 | E17E | .2.@.....@."@.. |
| 28A0 | C01E | 1EC3 | A219 | 23CD | 6528 | CDDA | 29CD | C409 |#..(..) |
| 28B0 | 1415 | C80A | CD2A | 03FE | 0DCC | 0321 | 0318 | F2B7 |*.....!..... |
| 28C0 | 0EF1 | F52A | A040 | EB2A | D640 | 2F4F | 06FF | 0923 | ...*.@.*.@/O...# |
| 28D0 | DF38 | 0722 | D640 | 23EB | F1C9 | F11E | 1ACA | A219 | .8."@#..... |
| 28E0 | BFF5 | 01C1 | 28C5 | 2AB1 | 4022 | D640 | 2146 04E5 | |(*.@".@!... |
| 28F0 | 2AA0 | 40E5 | 21B5 | 40EB | 2AB3 | 40EB | DF01 | F728 | *.@.!@.*.@.....(|
| 2900 | C24A | 292A | F940 | EB2A | FB40 | EBDF | 2813 | 7E23 | .J)*.@.*.@..(..# |
| 2910 | 2323 | FE03 | 2004 | CD4B | 29AF | 5F16 | 0019 | 18E6 | ##.....K)..>..... |
| 2920 | C1EB | 2AFD | 40EB | DFCA | 6B29 | 7E23 | CDC2 | 09E5 | ..*.@.....).#..... |
| 2930 | 09FE | 0320 | EB22 | D840 | E14E | 0600 | 0909 | 23EB |".@.N...#. |
| 2940 | 2AD8 | 40EB | DF28 | DA01 | 3F29 | C5AF | B623 | 5E23 | *.@..(..?)...#.# |
| 2950 | 5623 | C844 | 4D2A | D640 | DF60 | 69D8 | E1E3 | DFE3 | V#.DM*.@..... |
| 2960 | E560 | 69D0 | C1F1 | F1E5 | D5C5 | C9D1 | E17D | B4C8 | |
| 2970 | 2B46 | 2B4E | E52B | 6E26 | 0009 | 5059 | 2B44 | 4D2A | +F+N.+.&..PY+DM* |
| 2980 | D640 | CD58 | 19E1 | 7123 | 7069 | 602B | C3E9 | 28C5 | .@.X...#...+..(. |
| 2990 | E52A | 2141 | E3CD | 9F24 | E3CD | F40A | 7EE5 | 2A21 | *!A...\$.....*! |
| 29A0 | 41E5 | 861E | 1CDA | A219 | CD57 | 28D1 | CDDE | 29E3 | A.....W(.....). |
| 29B0 | CDDD | 29E5 | 2AD4 | 40EB | CDC6 | 29CD | C629 | 2149 | ..).*.@.....)!I |
| 29C0 | 23E3 | E5C3 | 8428 | E1E3 | 7E23 | 4E23 | 466F | 2C2D | #.....(....#N#F.,- |
| 29D0 | C80A | 1203 | 1318 | F8CD | F40A | 2A21 | 41EB | CDF5 |*!A... |
| 29E0 | 29EB | C0D5 | 5059 | 1B4E | 2AD6 | 40DF | 2005 | 4709 |)...PY.N*.@...G. |
| 29F0 | 22D6 | 40E1 | C92A | B340 | 2B46 | 2B4E | 2BDF | C022 | ".@...*.@+F+N+.." |

| | | | | | | | | | |
|------|------|------|------------------|-------------|------|-------------|-------------|-------------|---------------------|
| 2A00 | B340 | C901 | F827 | C5CD | D729 | AF57 | 7EB7 | C901 | .@...'....).W.... |
| 2A10 | F827 | C5CD | 072A | CA4A | 1E23 | 5E23 | 561A | C93E | .'...*.J.#.#V..> |
| 2A20 | 01CD | 5728 | CD1F | 2B2A | D440 | 73C1 | C384 | 28D7 | ..W(..+*.@.....(. |
| 2A30 | CF28 | CD1C | 2BD5 | CF2C | CD37 | 23CF | 29E3 | E5E7 | .(...+...,.7#.).... |
| 2A40 | 2805 | CD1F | 2B18 | 03CD | 132A | D1F5 | F57B | CD57 | (...+...*.W.....W |
| 2A50 | 285F | F11C | 1D28 | D42A | D440 | 7723 | 1D20 | FB18 | (.....(*.@.#..... |
| 2A60 | CACD | DF2A | AFE3 | 4F3E | E5E5 | 7EB8 | 3802 | 7811 | ...*.O>.....8.... |
| 2A70 | 0E00 | C5CD | BF28 | C1E1 | E523 | 4623 | 6668 | 0600 |(....#F#..... |
| 2A80 | 0944 | 4DCD | 5A28 | 6FCD | CE29 | D1CD | DE29 | C384 | .DM.Z(...).).... |
| 2A90 | 28CD | DF2A | D1D5 | 1A90 | 18CB | EB7E | CDE2 | 2A04 | (.*.....).....* |
| 2AA0 | 05CA | 4A1E | C51E | FFFE | 2928 | 05CF | 2CCD | 1C2B | ..J.....)(...+. |
| 2AB0 | CF29 | F1E3 | 0169 | 2AC5 | 3DBE | <i>0169</i> | <i>0169</i> | <i>7E91</i> | .).....*=.....O.. |
| 2AC0 | BB47 | D843 | C9CD | 072A | CAF8 | 275F | 237E | 2366 | .G.C...*..'.#.#. |
| 2AD0 | 6FE5 | 1946 | 72E3 | C57E | CD65 | 0EC1 | E170 | C9EB | ...F..... |
| 2AE0 | CF29 | C1D1 | C543 | C9FE | 7AC2 | 9719 | C3D9 | 41CD | .)...C.....A. |
| 2AF0 | 1F2B | 3294 | 40CD | 9340 | C3F8 | 27CD | 0E2B | C396 | .+2.@..@..'.+.. |
| 2B00 | 40D7 | CD37 | 23E5 | CD7F | 0AEB | E17A | B7C9 | CD1C | @..7#..... |
| 2B10 | 2B32 | 9440 | 3297 | 40CF | 2C18 | 01D7 | CD37 | 23CD | +2.@2.@,..7#. |
| 2B20 | 052B | C24A | 1E2B | D77B | C93E | 0132 | 9C40 | C1CD | +.J.+...>.2.@.. |
| 2B30 | 101B | C521 | FFFF | 22A2 | 40E1 | D14E | 2346 | 2378 | ...!..".@..N#F#. |
| 2B40 | B1CA | 191A | CDDF | 41CD | 9B1D | C54E | 2346 | 23C5 |A.....N#F#. |
| 2B50 | E3EB | DFC1 | DA18 | 1AE3 | E5C5 | EB22 | EC40 | CDAF |".@.. |
| 2B60 | 0F3E | 20E1 | CD2A | 03CD | 7E2B | 2AA7 | 40CD | 752B | .>...*...+*.@..+ |
| 2B70 | CDFE | 2018 | BE7E | B7C8 | CD2A | 0323 | 18F7 | E52A |*.#...* |
| 2B80 | A740 | 444D | E116 | FF18 | 0303 | 15C8 | 7EB7 | 2302 | .@DM.....#. |
| 2B90 | C8F2 | 892B | FEFB | 2008 | 0B0B | 0B0B | 1414 | 1414 | ...+..... |
| 2BA0 | FE95 | CC24 | 0BD6 | 7FE5 | 5F21 | 5016 | 7EB7 | 23F2 | ...\$.!P...#. |
| 2BB0 | AC2B | 1D20 | F7E6 | 7F02 | 0315 | CAD8 | 287E | 23B7 | +......(.#. |
| 2BC0 | F2B7 | 2BE1 | 18C6 | CD10 | 1BD1 | C5C5 | CD2C | 1B30 | ..+.....,.0 |
| 2BD0 | 0554 | 5DE3 | E5DF | D24A | 1E21 | 2919 | CDA7 | 28C1 | .T.....J.!)...(. |
| 2BE0 | 21E8 | 1AE3 | EB2A | F940 | 1A02 | 0313 | DF20 | F960 | !.....*..... |
| 2BF0 | 6922 | F940 | C9CD | 8402 | CD37 | 23E5 | CD13 | 2A3E | ".@.....7#...*> |
| 2C00 | D3CD | 6402 | CD61 | 021A | CD64 | 022A | A440 | EB2A |?*.@.* |
| 2C10 | F940 | 1A13 | CD64 | 02DF | 20F8 | CDF8 | 01E1 | C9CD | .@..... |
| 2C20 | 9302 | 7ED6 | B228 | 02AF | 012F | 23F5 | 2BD7 | 3E00 |(.../#.+.>. |
| 2C30 | 2807 | CD37 | 23CD | 132A | 1A6F | F1B7 | 6722 | 2141 | (..7#..*....."!A |
| 2C40 | CC4D | 1B2A | 2141 | EB06 | 03CD | 3502 | D6D3 | 20F7 | .M.*!A....5..... |
| 2C50 | 10F7 | CD35 | 021C | 1D28 | 03BB | 2037 | 2AA4 | 4006 | ...5...(...7*.@. |
| 2C60 | 03CD | 3502 | 5F96 | A220 | 2173 | CD6C | 197E | B723 | ..5.....!.....# |
| 2C70 | 20ED | CD2C | 0210 | EA22 | F940 | 2129 | 19CD | A728 | ...,.#.@!)... (|
| 2C80 | CDF8 | 012A | A440 | E5C3 | E81A | 21A5 | 2CCD | A728 | ...*.@.....!.,..(|
| 2C90 | C318 | 1A32 | 3E3C | 0603 | CD35 | 02B7 | 20F8 | 10F8 | ...2><...5..... |
| 2CA0 | CD96 | 0218 | <i>A242 4144</i> | <i>0D60</i> | CD7F | 0A7E | C3F8 | |BAD..... |
| 2CB0 | 27CD | 022B | D5CF | 2CCD | 1C2B | D112 | C9CD | 3823 | '..+...+.....8# |
| 2CC0 | CDF4 | 0ACF | 3BEB | 2A21 | 4118 | 083A | DE40 | B728 |;.*!A...@.(|
| 2CD0 | 0CD1 | EBE5 | AF32 | DE40 | BAF5 | D546 | B0CA | 4A1E |2.@...F..J. |
| 2CE0 | 234E | 2366 | 6918 | 1C58 | E50E | 027E | 23FE | 25CA | #N#.....X.....#.#. |
| 2CF0 | 172E | FE20 | 2003 | 0C10 | F2E1 | 433E | 25CD | 492E |C>#.I. |

| | | | | | | | | | |
|------|------|------|-------|------|------|------|------|------|------------------------------------|
| 2D00 | CD2A | 03AF | 5F57 | CD49 | 2E57 | 7E23 | FE21 | CA14 | . * . . . W . I . W . # . ! . . |
| 2D10 | 2EFE | 2328 | 3705 | CAFE | 2DFE | 2B3E | 0828 | E72B | .. # (7 . . . - . + > . (. + |
| 2D20 | 7E23 | FE2E | 2840 | FE25 | 28BD | BE20 | D0FE | 2428 | .. # . . (@ . 8 (. . . . \$ (|
| 2D30 | 14FE | 2A20 | C878 | FE02 | 2338 | 037E | FE24 | 3E20 | .. * # 8 . . . \$ > . |
| 2D40 | 2007 | 051C | FEAF | C610 | 231C | 8257 | 1C0E | 0005 | # . . W |
| 2D50 | 2847 | 7E23 | FE2E | 2818 | FE23 | 28F0 | FE2C | 201A | (G . # . . (. . # (. . , . . |
| 2D60 | 7AF6 | 4057 | 18E6 | 7EFE | 233E | 2E20 | 900E | 0123 | .. @ W # > # |
| 2D70 | 0C05 | 2825 | 7E23 | FE23 | 28F6 | D511 | 972D | D554 | .. (8 . # . # (. . . . - . T |
| 2D80 | 5DFE | 5BC0 | BEC0 | 23BE | C023 | BEC0 | 2378 | D604 | # . # . # |
| 2D90 | D8D1 | D147 | 1423 | CAEB | D17A | 2B1C | E608 | 2015 | .. . G . # + |
| 2DA0 | 1D78 | B728 | 107E | D62D | 2806 | FEFE | 2007 | 3E08 | .. . (. . . - (. . . . > . |
| 2DB0 | C604 | 8257 | 05E1 | F128 | 5C05 | D5CD | 3723 | D1C1 | .. . W (P . . . 7 # . . |
| 2DC0 | C5E5 | 4378 | 81FE | 19D2 | 4A1E | 7AF6 | 80CD | BE0F | .. C J |
| 2DD0 | CDA7 | 28E1 | 2BD7 | 3728 | 0D32 | DE40 | FE3B | 2805 | .. (. + . 7 (. 2 . @ . ; (. |
| 2DE0 | FE2C | C297 | 19D7 | C1EB | E1E5 | F5D5 | 7E90 | 234E | .. , # N |
| 2DF0 | 2366 | 6916 | 005F | 1978 | B7C2 | 032D | 1806 | CD49 | # - . . . I |
| 2E00 | 2ECD | 2A03 | E1F1 | C2CB | 2CDC | FE20 | E3CD | DD29 | .. * ,) |
| 2E10 | E1C3 | 6921 | 0E01 | 3EF1 | 05CD | 492E | E1F1 | 28E9 | .. ! . ! . > I . . . (. |
| 2E20 | C5CD | 3723 | CDF4 | 0AC1 | C5E5 | 2A21 | 4141 | 0E00 | .. 7 # * ! A A . . |
| 2E30 | C5CD | 682A | CDA A | 282A | 2141 | F196 | 473E | 2004 | .. * (* ! A . . G > . . |
| 2E40 | 05CA | D32D | CD2A | 0318 | F7F5 | 7AB7 | 3E2B | C42A | .. - . * > + . * |
| 2E50 | 03F1 | C932 | 9A40 | 2AEA | 40B4 | A53C | EBC8 | 1804 | .. . 2 . @ * . @ . . < |
| 2E60 | CD4F | 1EC0 | E1EB | 22EC | 40EB | CD2C | 1BD2 | D91E | .. O " . @ . . , |
| 2E70 | 6069 | 2323 | 4E23 | 4623 | C5CD | 7E2B | E1E5 | CDAF | .. # V N # F # + |
| 2E80 | 0F3E | 20CD | 2A03 | 2AA7 | 403E | 0ECD | 2A03 | E50E | .. > . . * . * . @ > . . * . . . |
| 2E90 | FF0C | 7EB7 | 2320 | FAE1 | 4716 | 00CD | 8403 | D630 | # G 0 |
| 2EA0 | 380E | FE0A | 300A | 5F7A | 0707 | 8207 | 8357 | 18EB | 8 0 W |
| 2EB0 | E521 | 992E | E315 | 14C2 | BB2E | 14FE | D8CA | D22F | .. ! / |
| 2EC0 | FEDD | CAE0 | 2FFE | F028 | 41FE | 3138 | 02D6 | 20FE | / . . (A . 1 8 |
| 2ED0 | 21CA | F62F | FE1C | CA40 | 2FFE | 2328 | 3FFE | 19CA | ! . . / . . . @ / . # (? . . . |
| 2EE0 | 7D2F | FE14 | CA4A | 2FFE | 13CA | 652F | FE15 | CAE3 | .. / J / / |
| 2EF0 | 2FFE | 28CA | 782F | FE1B | 281C | FE18 | CA75 | 2FFE | / . (. . / . . (. / . |
| 2F00 | 11C0 | C1D1 | CDFE | 20C3 | 652E | 7EB7 | C804 | CD2A | * |
| 2F10 | 0323 | 1520 | F5C9 | E521 | 5F2F | E337 | F5CD | 8403 | .. # ! . / . 7 |
| 2F20 | 5FF1 | F5DC | 5F2F | 7EB7 | CA3E | 2FCD | 2A03 | F1F5 | / > / . * . . . |
| 2F30 | DCA1 | 2F38 | 0223 | 047E | BB20 | EB15 | 20E8 | F1C9 | .. / 8 . # |
| 2F40 | CD75 | 2BCD | FE20 | C1C3 | 7C2E | 7EB7 | C83E | 21CD | .. + > ! . |
| 2F50 | 2A03 | 7EB7 | 2809 | CD2A | 03CD | A12F | 1520 | F33E | * (. . * . . . / . . . > |
| 2F60 | 21CD | 2A03 | C97E | B7C8 | CD84 | 0377 | CD2A | 0323 | ! . * * . # |
| 2F70 | 0415 | 20F1 | C93C | 4443 | 16FF | CD0A | 2FCD | 8403 | 6 . H / |
| 2F80 | B7CA | 7D2F | FE08 | 280A | FE0D | CAE0 | 2FFE | 1BC8 | .. . / . . (. / |
| 2F90 | 201E | 3E08 | 0504 | 281F | CD2A | 032B | 0511 | 7D2F | .. > . . . (. . * . + . . . / |
| 2FA0 | D5E5 | 0D7E | B737 | CA90 | 0823 | 7E2B | 7723 | 18F3 | 7 . . . # . + . # . . . |
| 2FB0 | F579 | FEFF | 3803 | F118 | C490 | 0C04 | C5EB | 6F26 | 8 & |
| 2FC0 | 0019 | 444D | 23CD | 5819 | C1F1 | 77CD | 2A03 | 23C3 | .. DM # . X * . # . . |
| 2FD0 | 7D2F | 78B7 | C805 | 2B3E | 08CD | 2A03 | 1520 | F3C9 | .. / + > . . * |
| 2FE0 | CD75 | 2BCD | FE20 | C1D1 | 7AA3 | 3C2A | A740 | 2BC8 | .. + < * . @ + . |
| 2FF0 | 3723 | F523 | 981A | C1D1 | C319 | 1ADE | C3C3 | 4402 | 7 # D . |
| 3000 | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | |

```

500 CLS: CLEAR200
510 CMD" T"
520 PRINT
530 ON ERROR GOTO 540
540 RESUME 550
550 PRINT" W 4 U C H   N U M B E R   C O N V E R S I O N   P R O
    G R A M"
560 PRINT
570 PRINT"           - DECIMAL TO BINARY ENTER D"
580 PRINT"           - BINARY TO DECIMAL ENTER B"
590 PRINT"           - HEXADECIMAL TO BINARY ENTER HB"
600 PRINT"           - DECIMAL TO HEXADECIMAL ENTER DH":
610 PRINT"           - HEXADECIMAL TO DECIMAL ENTER HD"
620 PRINT"           - SPLIT DECIMAL TO DECIMAL ENTER SP"
630 PRINT"           - DECIMAL TO SPLIT HEXADECIMAL ENTER DS"
640 PRINT"           - SPLIT HEXADECIMAL TO DECIMAL ENTER SD ";
650 INPUT AA$: CLS
660 IF AA$="HB" GOTO 1550
670 IF AA$="DH" THEN QB$=""
680 IF AA$="DS" THEN AA$="DH": QB$="DS"
690 IF AA$="SD" GOTO 1500
700 IF AA$="HD" GOTO 1550
710 IF AA$="B" GOTO 860
720 REM DECIMAL TO BINARY CONVERSION
730 CLS: INPUT" DECIMAL NO. "; X: IF X>65535 GOTO 730
740 A=INT(X/2): AA=X-2*A: B=INT(A/2): BB=A-2*B: C=INT(B/2): CC=B-2*C
750 D=INT(C/2): DD=C-2*D: E=INT(D/2): EE=D-2*E: F=INT(E/2): FF=E-2*F
760 G=INT(F/2): GG=F-2*G: H=INT(G/2): HH=G-2*H: I=INT(H/2): II=H-2*I
770 J=INT(I/2): JJ=I-2*J: K=INT(J/2): KK=J-2*K: L=INT(K/2): LL=K-2*L
780 M=INT(L/2): MM=L-2*M: N=INT(M/2): NN=M-2*N: O=INT(N/2): OO=N-2*O
790 PP=O-2*INT(O/2)
800 Y$=STR$(PP)+STR$(OO)+STR$(NN)+STR$(MM)+STR$(LL)+STR$(KK)+STR
$(JJ)+STR$(II)+STR$(HH)+STR$(GG)+STR$(FF)+STR$(EE)+STR$(DD)+STR$
(CC)+STR$(BB)+STR$(AA)
810 IF AA$="SP" THEN NO=NO+1: QB$="DS": GOTO 1200
820 IF AA$="DH" AND QB$="" THEN PRINT: PRINT" HEXADECIMAL ";: GOTO 1200
830 IF QB$="DS" GOTO 1200
840 PRINT" BINARY #"Y$: INPUT R: GOTO 730
850 REM BINARY TO DECIMAL CONVERSION
860 CLS: INPUT" 16, 8, OR 4 DIGIT BINARY NO. "; AA
870 CLS: PRINT AA: "DIGIT BINARY NO. ";: INPUT X$
880 IF LEN(X$)<>A GOTO 870
890 FOR Z=1 TO AA: X=VAL(MID$(X$, Z, 1)): IF Z=1 THEN A=X
900 IF Z=2 THEN B=X
910 IF Z=3 THEN C=X
920 IF Z=4 THEN D=X
930 IF AA=8 GOTO 970 ELSE IF AA=16 GOTO 970
940 NEXT
950 Y=A*8+B*4+C*2+D*1
960 PRINT"           DECIMAL"Y: INPUT R: GOTO 870
970 IF Z=5 THEN E=X
980 IF Z=6 THEN F=X
990 IF Z=7 THEN G=X
1000 IF Z=8 THEN H=X

```

```
1010 IFAA=16GOTO1050
1020 NEXT
1030 Y=A*128+B*64+C*32+D*16+E*8+F*4+G*2+H*1
1040 PRINT"          DECIMAL"Y:INPUTR:GOTO870
1050 IFZ=9THENI=X
1060 IFZ=10THENJ=X
1070 IFZ=11THENK=X
1080 IFZ=12THENL=X
1090 IFZ=13THENM=X
1100 IFZ=14THENN=X
1110 IFZ=15THENO=X
1120 IFZ=16THENP=X
1130 NEXT
1140 Y=A*32768+B*16384+C*8192+D*4096+E*2048+F*1024+G*512+H*256+I
*128+J*64+K*32+L*16+M*8+N*4+O*2+P*1
1150 PRINT:PRINT"          DECIMAL";Y:INPUTR
1160 IFAA$="SP"GOTO730
1170 IFAA$="B"GOTO870
1180 IFAA$="HD"GOTO1540
1190 IFAA$="SD"GOTO1500
1200 DH$=MID$(Y$,1,8):GOSUB1250
1210 DH$=MID$(Y$,9,8):GOSUB1250
1220 DH$=MID$(Y$,17,8):GOSUB1250
1230 DH$=MID$(Y$,25,8):GOSUB1250
1240 PRINT"          ";:INPUTR:GOTO730
1250 DH=VAL(DH$):IFDH=0THENDH$="0"
1260 IFDH=1THENDH$="1"
1270 IFDH=10THENDH$="2"
1280 IFDH=11THENDH$="3"
1290 IFDH=100THENDH$="4"
1300 IFDH=101THENDH$="5"
1310 IFDH=110THENDH$="6"
1320 IFDH=111THENDH$="7"
1330 IFDH=1000THENDH$="8"
1340 IFDH=1001THENDH$="9"
1350 IFDH=1010THENDH$="A"
1360 IFDH=1011THENDH$="B"
1370 IFDH=1100THENDH$="C"
1380 IFDH=1101THENDH$="D"
1390 IFDH=1110THENDH$="E"
1400 IFDH=1111THENDH$="F"
1410 IFQB$="DS"THENQQ=QQ+1
1420 IFQQ=1THENRR$=DH$:RETURN
1430 IFQQ=2THENS$=DH$:RETURN
1440 IFQQ=3THENTT$=DH$:RETURN
1450 IFQQ=4THENUU$=DH$
1460 IFQB$="DS"ANDAA$="DH"THENPRINT:PRINT" SPLIT HEX      ";TT$;UU$
;RR$;S$:QQ=0:INPUTR:GOTO730
1470 IFQB$="DS"ANDAA$="SP"ANDNO=1THENFF$ =TT$+UU$:QQ=0:GOTO730
1480 IFQB$="DS"ANDAA$="SP"ANDNO=2THENS$=TT$+UU$:X$=S$+FF$:NO=0
:QQ=0:PRINT" HEXADECIMAL ";X$:GOTO1560
1490 PRINTDH$;:RETURN
1500 'SPLIT HEX TO DECIMAL CONVERSION
1510 CLS:INPUT"SPLIT HEX NO.      ";SH$
1520 SI$=LEFT$(SH$,2):SJ$=RIGHT$(SH$,2)
```

```
1530 X$="":X$=SJ$+SI$:GOTO1560
1540 REM HEX TO DECIMAL AND HEX TO BINARY CONVERSION
1550 CLS:X$="":INPUT"HEXADECIMAL NO. ";X$
1560 IFLEN(X$)<>4GOTO1550
1570 FOR XX=1TO4:A$=MID$(X$,XX,1)
1580 IFA$="0"THENNA$="0000"
1590 IFA$="1"THENNA$="0001"
1600 IFA$="2"THENNA$="0010"
1610 IFA$="3"THENNA$="0011"
1620 IFA$="4"THENNA$="0100"
1630 IFA$="5"THENNA$="0101"
1640 IFA$="6"THENNA$="0110"
1650 IFA$="7"THENNA$="0111"
1660 IFA$="8"THENNA$="1000"
1670 IFA$="9"THENNA$="1001"
1680 IFA$="A"THENNA$="1010"
1690 IFA$="B"THENNA$="1011"
1700 IFA$="C"THENNA$="1100"
1710 IFA$="D"THENNA$="1101"
1720 IFA$="E"THENNA$="1110"
1730 IFA$="F"THENNA$="1111"
1740 NN=NN+1
1750 IFNN=1GOTO1760ELSEIFNN=2GOTO1770ELSEIFNN=3GOTO1780ELSEIFNN=
4GOTO1790
1760 BB$=A$:          NEXTXX:GOTO1570
1770 CC$=A$:          NEXTXX:GOTO1570
1780 DD$=A$:          NEXTXX:GOTO1570
1790 EE$=A$:
1800 X$=BB$+CC$+DD$+EE$:NN=0
1810 IFAA$="HB"THENPRINTX$;" BINARY ";:INPUTR:CLS:GOTO1550
1820 AA=16:GOTO890
1830 END
1840 '
1850 ' NOTE: PROGRAM UTILIZES 3919 BYTES OF MEM
```

```

00100 ; SOURCE CODE PROGRAM TO PRINT ALL ZEROS WITH A SLASH
00110
00120 ; WITH AUTO CARRIAGE RETURN @ 64 CHARACTERS WHEN LLIST
00130
00140      ORG      7F15H      ;=32533 DECIMAL ORIGIN
00150 COUNT    EQU      7F15H      ;MEM LOCATION FOR CHAR. COUNTER
00160      DEFW    COUNT      ;ASSEMBLER SAVES 2 BYTES @ 7F15H
00170 START    PUSH    AF          ;SAVE REGISTERS IN STACK MEMORY
00180      PUSH    BC
00190      PUSH    DE
00200      PUSH    HL
00210      LD     A,C          ;NEXT CHARACTER TO PRINT IN C REG
00220      CP     0DH          ;IS IT A CARRIAGE RETURN?
00230      JR     Z,NOINC      ;EXIT W/O INCREMENTING COUNTER
00240      LD     A,(COUNT)    ;COUNTER VALUE FROM MEM
00250      CP     040H        ;IS IT 64 DECIMAL = END OF LINE?
00260      JR     Z,CARRET     ;IF SO, GOTO CARR. RETURN
00270 FINIS    LD     A,(COUNT) ;IF RETURN FM CARRET
00280      INC     A           ;ADD +1 CHARACTER ON LINE
00290      LD     (COUNT),A    ;NEW COUNT NUMBER TO MEM
00300      LD     A,C          ;NEXT CHARACTER TO PRINT IN C REG
00310      CP     30H          ;30H=ASCII ZERO. IS IT A ZERO?
00320      CALL   Z,ZERO       ;IF SO, GOTO ZERO SUBROUTINE
00330 ELFIN    POP     HL       ;RESTORE REGISTERS TO ORIG. CONDX
00340      POP     DE
00350      POP     BC
00360      POP     AF
00370      JP     058DH        ;GOTO ROM STD. PRINTER ROUTINE
00380 NOINC    LD     A,00H     ;RESET CHARACTER COUNTER-
00390      LD     (COUNT),A    ;IN MEMORY
00400      JR     ELFIN        ;QUICK DEPARTURE
00410 RESET    LD     A,00H     ;RESET CHARACTER-
00420      LD     (COUNT),A    ;COUNTER IN MEMORY.
00430      JR     FINIS       ;ALL DONE
00440 CARRET   CALL   TEST     ;TEST IF PRINTER READY?
00450      LD     A,0DH        ;0DH = ASCII CARRIAGE RETURN
00460      LD     (37E8H),A     ;DO CARRIAGE RETURN
00470      JR     RESET       ;GOTO RESET CHARACTER COUNTER
00480 TEST     LD     A,(37E8H) ;PRINTER MEM LOCATION
00490      BIT     7,A         ;PRINTER READY HANDSHAKE?
00500      JR     NZ,TEST      ;LOOP TILL HANDSHAKE
00510      RET                    ;RETURN TO LINE AFTER CALL
00520 ZERO     CALL   TEST     ;IS PRINTER READY?
00530      LD     A,2FH        ;2FH=ASCII SLASH
00540      LD     (37E8H),A     ;PRINT '/' SLASH
00550      CALL   TEST     ;IS PRINTER READY?
00560      CALL   DELAY      ;20 MILLISECOND DELAY
00570      LD     A,08H        ;08H = ASCII BACKSPACE
00580      LD     (37E8H),A     ;DO BACKSPACE
00590 DELAY    LD     C,0AH     ;INITIALIZE DELAY-
00600 DELAY1    LD     B,0H     ;LOOPS TO ALLOW SELECTRIC-
00610 DELAY2    DJNZ   DELAY2  ;PRINT HEAD TIME TO SETTLE DOWN-
00620      DEC     C           ;FROM BACKSPACE VIBRATION AND-
00630      JP     NZ,DELAY1    ;FOR TRACK TO LOCK.
00640      RET                    ;RETURN TO LINE AFTER CALL DELAY.
00650      ORG     4026H        ;PUT ADDRESS OF START IN PRINTER-
00660      DEFW    START      ;DRIVER ADDRESS AT 4026H MEMORY.
00670      ORG     COUNT      ;CHARACTER COUNTER MEM ADDRESS-
00680      DEFB    00H        ;INITIALIZE AT ZERO.
00690      END     COUNT      ;FIRST LINE OF SUBROUTINE

```


OBJECT CODE: LPRINT ALL ZEROS WITH SLASH

| | | | | | |
|------|--------|-------|--------|------|-----------|
| 7F15 | | 00140 | | ORG | 7F15H |
| 7F15 | | 00150 | COUNT | EQU | 7F15H |
| 7F15 | 157F | 00160 | | DEFW | COUNT |
| 7F17 | F5 | 00170 | START | PUSH | AF |
| 7F18 | C5 | 00180 | | PUSH | BC |
| 7F19 | D5 | 00190 | | PUSH | DE |
| 7F1A | E5 | 00200 | | PUSH | HL |
| 7F1B | 79 | 00210 | | LD | A,C |
| 7F1C | FE0D | 00220 | | CP | 0DH |
| 7F1E | 281B | 00230 | | JR | Z,NOINC |
| 7F20 | 3A157F | 00240 | | LD | A,(COUNT) |
| 7F23 | FE40 | 00250 | | CP | 040H |
| 7F25 | 2822 | 00260 | | JR | Z,CARRET |
| 7F27 | 3A157F | 00270 | FINIS | LD | A,(COUNT) |
| 7F2A | 3C | 00280 | | INC | A |
| 7F2B | 32157F | 00290 | | LD | (COUNT),A |
| 7F2E | 79 | 00300 | | LD | A,C |
| 7F2F | FE30 | 00310 | | CP | 30H |
| 7F31 | CC5B7F | 00320 | | CALL | Z,-ZERO |
| 7F34 | E1 | 00330 | ELFIN | POP | HL |
| 7F35 | D1 | 00340 | | POP | DE |
| 7F36 | C1 | 00350 | | POP | BC |
| 7F37 | F1 | 00360 | | POP | AF |
| 7F38 | C38D05 | 00370 | | JP | 058DH |
| 7F3B | 3E00 | 00380 | NOINC | LD | A,00H |
| 7F3D | 32157F | 00390 | | LD | (COUNT),A |
| 7F40 | 18F2 | 00400 | | JR | ELFIN |
| 7F42 | 3E00 | 00410 | RESET | LD | A,00H |
| 7F44 | 32157F | 00420 | | LD | (COUNT),A |
| 7F47 | 18DE | 00430 | | JR | FINIS |
| 7F49 | CD537F | 00440 | CARRET | CALL | TEST |
| 7F4C | 3E0D | 00450 | | LD | A,0DH |
| 7F4E | 32E837 | 00460 | | LD | (37E8H),A |
| 7F51 | 18EF | 00470 | | JR | RESET |
| 7F53 | 3AE837 | 00480 | TEST | LD | A,(37E8H) |
| 7F56 | CE7F | 00490 | | BIT | 7,A |
| 7F58 | 20F9 | 00500 | | JR | NZ,TEST |
| 7F5A | C9 | 00510 | | RET | |
| 7F5B | CD537F | 00520 | ZERO | CALL | TEST |
| 7F5E | 3E2F | 00530 | | LD | A,2FH |
| 7F60 | 32E837 | 00540 | | LD | (37E8H),A |
| 7F63 | CD537F | 00550 | | CALL | TEST |
| 7F66 | CD6E7F | 00560 | | CALL | DELAY |
| 7F69 | 3E08 | 00570 | | LD | A,08H |
| 7F6B | 32E837 | 00580 | | LD | (37E8H),A |
| 7F6E | 0E0A | 00590 | DELAY | LD | C,0AH |
| 7F70 | 0600 | 00600 | DELAY1 | LD | B,0H |
| 7F72 | 10FE | 00610 | DELAY2 | DJNZ | DELAY2 |
| 7F74 | 0D | 00620 | | DEC | C |
| 7F75 | C2707F | 00630 | | JP | NZ,DELAY1 |
| 7F78 | C9 | 00640 | | RET | |
| 4026 | | 00650 | | ORG | 4026H |
| 4026 | 177F | 00660 | | DEFW | START |
| 7F15 | | 00670 | | ORG | COUNT |
| 7F15 | 00 | 00680 | | DEFB | 00H |
| 7F15 | | 00690 | | END | COUNT |

CHAPTER 9
- SELF-TEST QUESTIONS -

The following pages of self-test questions cover a number of important points in the preceding Chapters and demonstration programs. If you understand the logic, program flow, and rationale of each Chapter/program's contents, you should have little difficulty in answering the questions. If a question's answer is not clear, re-read/study the appropriate Chapter and/or demonstration program till osmosis occurs, as in time it will indeed permeate. If all else fails, try putting this handbook under your pillow and sleeping on it. (Ed.)

SELF-TEST QUESTIONS FOR CHAPTER 1:

1. What is meant by masking the most significant bit (MSB)?

2. What are the decimal and hexadecimal locations in ROM for the Level II functions' NAMES?

From _____ / _____ to _____ / _____.

3. Why are disk BASIC's functions' names and coupling addresses in non-disk Level II BASIC ROM?

4. How would you mask the MSB of any MEM location in a simple program written in BASIC?.

5. How many MEM groups (separate address segments) do the BASIC functions' CALL addresses occupy? Beginning locations?

A. _____ B. _____ C. _____ D. _____

6. Name the 2 geniuses at Microsoft who wrote Level II and disk BASIC (luckily they did not write DOS 2.1 or 2.2).

A _____ G _____

7. Why are the BASIC functions' CALL addresses in Level II ROM expressed LSB first and MSB second?

8. What are the decimal values of PEEK 5666 & 5667? What Level II function's CALL address do they represent?

A. _____ / _____ B. _____ / _____

9. What are the MEM locations in decimal and hex of the following PEEK values? A. 194-30 B. 15-42 C. 255-255 D. 0-255

A. _____ / _____ B. _____ / _____ C. _____ / _____ D. _____ / _____

SELF-TEST QUESTIONS - CHAPTER 2:

1. On the average, how much faster will efficiently written assembly language programs run than BASIC ? MEM required?

A. _____ B. _____

2. What do we call the 2 RAM arithmetic storage locations?

A. _____ B. _____

3. What is the NT (number type) for the following numbers?

A. 10000 _____ B. 33000 _____ C. 1.011 _____ D. 1.1111111 _____

4. Where is the NT (number type) stored in RAM memory? _____

5. What type of numbers are represented by NT (number type) =?

A. 2 _____ B. 3 _____ C. 4 _____ D. 8 _____

6. In Figure 7, what is accomplished by line 310?

7. In Figure 7, what is accomplished by line 320?

8. In Figure 9, what is accomplished by lines 340/350?

9. In Figure 11, what is accomplished by line 310?

10. How many significant digits in a CDBL argument?

NOTES:

SELF-TEST QUESTIONS - CHAPTER 3:

1. Does Level II ROM ever use the the Z-80 alternate register pairs AF' - BC' - DE' - HL' ?

2. Where is the ROM CALL location to move "B" register bytes from MEM location XXXX to MEM location YYYY?

3. How many MEM locations in the ACCUM and CDBL store? Where?

4. How is the NT (number type converted to ASCII)?

5. What CALL converts the ACCUM to an ASCII string?

6. What function does Figure 13's line 420 perform? Why?

7. What is the difference between RANDOM and RND?

8. What is the difference between CINT and INT?

9. How is the conversion MEM location CALLED in Figure 13?

10. How is the RETURN MEM location CALLED in Figure 13?

NOTE: Pages 17, 19, & 21 add a zero Operand AFTER "DEFB".
See pages 18, 20, & 22 for correct placement.

SELF-TEST QUESTIONS - CHAPTER 4:

1. What is the value of PEEK (14464) if one shift key is pressed? If both shift keys are pressed?

2. What is the value of PEEK (14338) if H, I, J, K, L, M, N, and O keys are pressed simultaneously?

3. What is the major difference between CALL 002BH and 0049H?

4. What data is stored at MEM location 409CH?

5. What is the simplest way to store an integer and single precision number in RAM (not counting the ACCUM & CDBL store).

6. Where are data comparison results stored? What values?

7. What is the value of the "A" register if CALL 0994H is used and the ACCUM contains -1.9999999999?

8. What CALL is used to change any value to an integer? To a single precision number? To a double precision number?

9. What is the value of NT RAM storage for a string?

10. What value will MEM location 37E8H contain when the line printer is ready (handshake)? What MEM location is loaded with the NEXT character to be printed?

11. Write a brief assembly language program to output to the video display a message of ANY length using the CALL 28A7H subroutine.

NOTE: This subroutine will output a string of up to 63 characters with non-disk Level II. By concatenating the strings with additional DEFM OPCODES, the message length is only limited by available memory.

SELF-TEST QUESTIONS - CHAPTER 5:

Fill in decimal CALL locations for these Level II functions using Chapter 7's Multi-Base Conversion Program:

| BASIC FUNCTION | CALL ADDRESS | BASIC FUNCTION | CALL ADDRESS |
|-------------------|-----------------|-------------------|-----------------|
| ABS | _____ | ASC | _____ |
| ATN | _____ | AUTO | _____ |
| CDBL | _____ | CHR\$ | _____ |
| CINT | _____ | CLEAR | _____ |
| CLOAD | _____ | CLS | _____ |
| CONT | _____ | COS | _____ |
| CSAVE | _____ | CSNG | _____ |
| DATA | _____ | DEFDBL | _____ |
| DEFINT | _____ | DEFSNG | _____ |
| DEFSTR | _____ | DELETE | _____ |
| DIM | _____ | EDIT | _____ |
| ELSE | _____ | END | _____ |
| ERR | _____ | ERL | _____ |
| ERROR | _____ | EXP | _____ |
| FIX | _____ | FOR | _____ |
| FRE | _____ | GOSUB | _____ |
| GOTO | _____ | IF | _____ |
| INKEY\$ | _____ | INP | _____ |
| INPUT | _____ | INSTR | _____ |
| INT | _____ | LEFT\$ | _____ |
| LEN | _____ | LIST | _____ |
| LOG | _____ | LLIST | _____ |
| LPRINT | _____ | MEM | _____ |
| MID\$ | _____ | NEW | _____ |
| NEXT | _____ | NOT | _____ |
| ON | _____ | OUT | _____ |
| PEEK | _____ | POINT | _____ |
| POKE | _____ | POS | _____ |
| PRINT | _____ | RANDOM | _____ |
| READ | _____ | REM | _____ |
| RESET | _____ | RESTORE | _____ |
| RESUME | _____ | RIGHT\$ | _____ |
| RND | _____ | RUN | _____ |
| SET | _____ | SGN | _____ |
| SIN | _____ | SQR | _____ |
| STOP | _____ | STR\$ | _____ |
| STRING\$ | _____ | SYSTEM | _____ |
| TAN | _____ | TROFF | _____ |
| TRON | _____ | USR | _____ |
| VARPTR | _____ | VAL | _____ |

SELF-TEST QUESTIONS - CHAPTERS 6, 7 & 8:

1. A. Each MEM location on page 37 holds how many hex bytes?
B. Each Line? C. Total page?
-

2. A. On page 37, location 011FH contain hex? B. = ASCII?
-

3. On page 37, just above line 0000, fill-in LSB of each hexadecimal memory location, zero through F.
-

4. A. In Chapter 7's "Number Conversion Program," is line 510 only for disk BASIC? B. Should it be deleted for non-disk?
-

5. How many fundamental number conversions are performed by Chapter 7's "Number Conversion Program"?
-

6. How is the "SPLIT DECIMAL TO DECIMAL" conversion performed?
-

7. In Chapter 8's program, rewrite lines 170-200 and 330-360 to use EX AF,AF' & EXX OPCODES instead of the stack for storage. Be VERY careful.
-

8. A. In line 490, rewrite this line to use the CP (compare) OPCODE instead of BIT 7,A. B. What is the difference?
-

9. Why are lines 590-630 necessary with IBM Selectric printers?
-

10. If you had a line printer with 130 characters per line capability (most IBM Selectric Printer Terminals have it) how would you modify this program to use all 130 positions?

CHAPTER 10

- BIBLIOGRAPHY AND ANSWERS TO SELF-TEST QUESTIONS -

BIBLIOGRAPHY and BOOKS AVAILABLE:

Using the consumers' scale of: BEST buy, GOOD buy, FAIR buy, POOR buy, and utterly WRETCHED, there is nothing on the market today for the TRS-80 assembly programmer that rates much higher than FAIR in this reviewers opinion, with two exceptions. They are both for the advanced assembly language programmer and are:

BEST: Andrew Hildebrand's "Software Technical Manual", @ \$40.
Houston Micro Computer Technologies
5313 Bissonnet Street
Bellaire, Texas 77401

GOOD: Lance Leventhal's "Z-80 Assembly Language Pgmng", @ \$15.
Adam Osborne & Associates
630 Bancroft Way
Berkeley, California 94710

Both of the above books assume that the reader has at least a few years of experience of 8080A assembly language programming experience before jumping into the Z-80, in our opinion.

There is really nothing truly worthwhile available for the beginning assembly language programmer. This includes Radio Shack's "TRS-80 Assembly language Programming," at \$3.95. Even at \$3.95 it is grossly OVER PRICED and in our kindest moments deserves our WRETCHED rating. Since there is nothing else for the beginner but garbage, garbage is the only ENTRE available and if one is truly hungry enough and holds one's nose, garbage is better than starving to death.

The following books have received our FAIR consumers' rating because of considerable generosity on our reviewer's part.

FAIR: "Z-80 Programming For Logic Design"
great for typewriter mechanics

FAIR: "The Z-80 Microcomputer Handbook"
the cover should be a clue to the buyer

FAIR: "Z-80 Software Gourmet Guide & Cookbook"
cooked tripe is still tripe

What the 200,000+ TRS-80 owners above the 5th grade level are awaiting is W6OVP's new book on the subject of assembly language programming. W6OVP is Dr. David Lien, Dean of San Diego University, a professional educator of the first water.

Dave Lien wrote the "User's Manual for Level 1," which is certainly the finest tutorial for beginning BASIC programmers from age 12 to 80 ever published, and followed it up fall '79 with "Learning Level II," which also deserves an excellent rating. If and when Dave publishes tutorials on disk BASIC and assembly language programming, they will indeed be on the best-seller list.

- ANSWERS TO QUESTIONS ON CHAPTERS' 1 TO 8 -

NOTE:

Advanced assembly language programmers' will undoubtedly think many of the questions naive to the point of ridiculousness. Indeed, many questions are absurdly simple to DRIVE HOME points to those readers who may NOT be as advanced as others. It is a difficult balancing act, so we ask for your patience and understanding.

ANSWERS TO CHAPTER 1:

1. Assume that PEEK(5712) = 197 decimal. 5712 is MEM location 1650H. Now, 197 decimal = 11000101 binary. The left most bit is a 1 which is masked (eliminated). It now = 1000101 binary = 69 decimal = ASCII "E". Since 10000000 binary = 128 decimal this same bit may be masked by simply subtracting 128 from the PEEK decimal value.
2. From 5712 decimal/1650H to 6172 decimal/181CH.
3. They are all CALLED by non-disk Level II ROM. If no disk is present or operating = L3/ERROR.
4. Subtract 128 from the PEEK value.
5. Two.
6. From 6178 decimal/1822H to 6297 decimal/1899H, and from 5642 decimal/160AH to 5711 decimal/164FH.
7. Convenience + allowing modest upward compatability between the earlier 8080 microprocessor and the newer Z-80, both invented/designed by Dr. Federico Faggin (President of Zilog).
8. 189 and 21 decimal.
9. ATN = arc tangent with output value in radians.
10. A. 7874 decimal/1EC2H B. 10767 decimal/2A0FH
 C. 65535 decimal/FFFFH D. 65280 decimal/FF00H

ANSWERS TO QUESTIONS ON CHAPTER 2:

1. A. 300 to 350+ times faster. B. 1/10th as much memory.
2. ACCUM and CDBL Store; ("CS" abbreviation).
3. A. = 2 B. = 4 (>32767) C. = 4 D. = 8 NOTE: 3 = string
4. NT MEM location at 40AFH.
5. A. 2 = integer B. 4 = single precision C. 8 = double prec.
6. CALL 0E6CH changes an arithmetic string to minimum NT and stores it in the ACCUM in the appropriate format, plus storing the NT in MEM location 40AFH.
7. CALL 0A7FH = CINT which changes ANY number in the range of +32767 to -32768 to an integer.
8. Uses stack MEM in RAM to store the single precision value of registers BCDE while BCDE are used in following CALLs.
9. See answer to question 6, above. CALL 0E65H is similar, but stores the number in the ACCUM in double precision format and sets the NT in MEM location 40AFH to 8.
10. THIS MAY SURPRISE YOU. Answer: only as many digits as contained in the argument. Up to 17 digits maximum. If the argument contains 10 significant digits, the last 7 digits will be meaningless. See Radio Shack's "Microcomputer News-Letter," October '79, page 2 for an excellent explanation.

ANSWERS TO QUESTIONS ON CHAPTER 3:

1. NO, they are never used in Level II ROM as this Microsoft BASIC is sort of the son of an earlier 8080 BASIC and the 8080 has neither OPCODE. This allows YOU to use it whenever desired and save 4+ bytes compared with PUSH and POP.
2. CALL 09D7H/2519 decimal.
3. Eight in each one.
4. ADD A,48 ;converts to ASCII.
5. CALL 0FBDBH converts ACCUM to a string (address in HL), and NT in MEM location 40AFH = 3.
6. A. Converts any number in CINT range in ACCUM to integer.
B. All ROM CALL locations are integers, so acts as a trap for erroneous input.

ANSWERS TO QUESTIONS ON CHAPTER 3 - CONTINUED:

7. RANDOM re-initializes the Z-80 random number generator. RND with NT = 2 = integer, generates a pseudo-random number between 1 and the integer value in the ACCUM. RND with NT = 4 = a single precision number, generates a pseudo random single precision number between zero and the number in the ACCUM which must = or < 1.
8. INT returns a round number (no decimal points) for ANY number. CINT changes any number with up to 7 digits to an integer within the range of +32767 and -32768.
9. Line 600's OPCODE is a RET which effectively POPs the top number off the stack, thus CALLing the CONV MEM location.
10. Since each listed CONVersion subroutine ends with a RET, this effects a POP the top number off of the stack, which is the RETURN address. This is a very useful ploy at times.

ANSWERS TO QUESTIONS ON CHAPTER 4:

1. A. 1 B. 1 2. 255 decimal
3. CALL 002BH scans the keybaord ONCE. CALL 0049H scans the keyboard till NOT zero (till any key is pressed). Virtually the same as INKEY\$ in BASIC.
4. MEM location 409CH stores "output" directions for CALL 1BB3H (amongst others). Initialized at Zero = video display, +1 = line printe, and -1 = cassette.
5. A. PUSH DE or PUSH HL B. PUSH BC and PUSH DE C. Move 8 bytes from ACCUM to MEM via CALL 09D6H or 09D7H (A OR B reg).
6. A. In the "A" register. B. Zero C. +1 D. 255 (OFFH)
7. 255 decimal - 11111111 binary - OFFH hex
8. A. CALL 0B37H/2871 decimal = INT; CINT's range is limited.
B. CALL 0AB1H/2737 decimal = CSNG C. CALL 0ADBH/2779 decimal = CDBL.
9. 3 decimal
10. A. 63 decimal = ASCII ? (what next?). B. 37E8H
11. W4UCH EQU 7D00H ;MESSAGE PROGRAM
 ORG W4UCH ;7D00H = 32000 DECIMAL
 LD HL,STRING ;STRING MEM ADDRESS
 CALL 28A7H ;DISPLAY \$ SUBROUTINE
 STRING DEFM 'USING LEVEL II ROM SUBROUTINES'
 DEFB 0 ;END OF MESSAGE DELIMITER
 END W4UCH ;CONCATENATE ANY LENGTH
;NOTE-you may concatenate up to 240 bytes per CALL 28A7H.

ANSWERS TO SELF-TEST QUESTIONS

- CHAPTER 5 -

LEVEL II BASIC FUNCTION CALL ADDRESSES IN DECIMAL

| BASIC FUNCTION | CALL ADDRESS | BASIC FUNCTION | CALL ADDRESS |
|-------------------|-----------------|-------------------|-----------------|
| ABS | 2423 | ASC | 10767 |
| ATN | 5565 | AUTO | 8200 |
| CDBL | 2779 | CHR\$ | 10783 |
| CINT | 2687 | CLEAR | 7802 |
| CLOAD | 11295 | CLS | 457 |
| CONT | 7652 | COS | 5441 |
| CSAVE | 11253 | CSNG | 2737 |
| DATA | 7941 | DEFDBL | 7689 |
| DEFINT | 7683 | DEFSNG | 7686 |
| DEFSTR | 7680 | DELETE | 11206 |
| DIM | 9736 | EDIT | 11872 |
| ELSE | 7943 | END | 7598 |
| ERL | 9437 | ERR | 9423 |
| ERROR | 8180 | EXP | 5177 |
| FIX | 2854 | FOR | 7329 |
| FRE | 10196 | GOSUB | 7857 |
| GOTO | 7874 | IF | 8249 |
| INKEY\$ | 413 | INP | 10991 |
| INPUT | 8602 | INSTR | 16787 |
| INT | 2871 | LEFT\$ | 10849 |
| LEN | 10755 | LIST | 11054 |
| LOG | 2057 | LLIST | 11049 |
| LPRINT | 8295 | MEM | 10185 |
| MID\$ | 10906 | NEW | 6985 |
| NEXT | 8886 | NOT | 9668 |
| ON | 8044 | OUT | 11003 |
| PEEK | 11434 | POINT | 307 |
| POKE | 11441 | POS | 10229 |
| PRINT | 8303 | RANDOM | 467 |
| READ | 8687 | REM | 7943 |
| RESET | 312 | RESTORE | 7569 |
| RESUME | 8111 | RIGHT\$ | 10897 |
| RND | 5321 | RUN | 7843 |
| SET | 309 | SGN | 2442 |
| SIN | 5447 | SQR | 5095 |
| STOP | 7593 | STR\$ | 10294 |
| STRING\$ | 10799 | SYSTEM | 690 |
| TAN | 5544 | TROFF | 7672 |
| TRON | 7671 | USR | 10238 |
| VARPTR | 9461 | VAL | 10949 |

ANSWERS TO SELF-TEST QUESTIONS CHAPTERS 6, 7, 8:

1. A. One B. 16 bytes per line C. 48 lines X 16/line = 768

2. A. 56H = 86 decimal B. = ASCII "V"

3. 0 1 2 3 4 5 6 7 8 9 A B C D E F 0123456789ABCDEF
 0000 F3AF C374 06C3 0040 C300 40E1 E9C3 9F06@...@.....

4. A. YES: to speed-up disk a bit B. YES: remove from non-disk

5. A. To round-off division by 2 to an integer = a short-cut in the decimal to binary conversion equation.

B. 4 fundamental conversions - rest are only massages.

- decimal to binary : lines 740 - 840

- binary to decimal : lines 860 - 1150

- binary to hexadecimal : lines 1200 - 1490

- hexadecimal to binary : lines 1550 - 1810

6. The first decimal number is converted to hex and stored in FF\$ in line 1470. The second decimal number is then converted to hex and stored in SS\$ in line 1480. FF\$ and SS\$ are then reversed and made = to X\$ in line 1480. The correct hex number which is X\$ is then displayed on video in line 1480. This hex number, X\$, is then converted to binary, and next converted to decimal, and then displayed on video. Though very SLOW when written in BASIC, it still beats a Hewlett-Packard calculator.

```
7. 00170 START EX AF,AF' ;EXCHANGE ALTERNATE REGS.
    00180 LD A,C ;NEXT CHAR. TO "A" REGISTER
    00190 EXX ;EXCHANGE BC DE HL ALT. REG
delete lines 00200 and 00210
```

NOTE: Since the NEXT character to be printed is in the "C" register, we MUST load it into "A" BEFORE exchange.

```
00330 ELFIN EX AF,AF' ;EXCHANGE ALTERNATE REGS.
00340 EXX ;EXCHANGE BD DE HL ALT. REG
delete lines 00350 and 00360
```

NOTE: This little exercise deleted 4 lines and 4+ bytes.
 A very decided improvement & good technique.

8. 00490 CP 3FH ;is it ASCII ? = 63 decimal

9. Backspace vibrates most IBM Selectric printers quite severely. Try your printer without these lines. Use only if necessary.

10. Change line 250 to read:

```
00250 CP 082H ;is it 130 decimal = end of line?
```

FINAL REQUEST:

Please send ANY errors you find to:

RICHCRAFT ENGINEERING LTD
 DRAWER 1065
 CHAUTAUQUA, NY 14722

Commands

| Commands | Definition |
|----------|--|
| A | Displays alteration possibilities. |
| A/A | Alters A account names. |
| A/C | Alters Category names. |
| A/D | Alters data. |
| A/E | Alters Notepad names. |
| A/F | Alters data diskette title. |
| A/Y | (Month Level only) Alters data diskette year. |
| B | Sets or clears buzzer. If check is turned on. |
| C | Selects from display of categories or cancels category. |
| D | (Day Level only) Deletes entry. |
| D/D | (Month Level only) Deletes all non-permanent entries. |
| F | Displays flag possibilities. |
| F/F | Feed Flag. Turns automatic carriage return line feed off and on. |
| F/L | Lower Case Flag. Displays lower case letters. |
| F/Q | Quiet Flag. Causes tone to sound, or remain quiet, when days are changed at either the Month or Day Level. |

| Commands | Definition |
|----------|---|
| F/B | Recover Flag. Uses flags from data diskette, not those currently selected. |
| F/W | Write Flag. Records flags currently selected onto data diskette. |
| F/X | Expert Flag. Causes Time Manager to not display error messages unless requested by X key. Typing X again will cause program to display error messages normally. |
| I | Calculates and displays interval from marked day to selected data. |
| K | Selects or cancels keyword. |
| L | Selects prioritises for display. |
| M | (Day Level only) Displays move possibilities. |
| M/D | Moves entry to (D) day. |
| M/C | Move/Clear. Clears move buffer. |
| M/G | Move/Get. Extracts entry from move buffer. |
| M/S | Move/Save. Stores entry in move buffer. |
| N | Displays Notepad Directory. |
| N/A | Displays Notepads A through H. |
| P | Prints entries. |

| Commands | Definition |
|----------|--|
| R | Displays status of flags, number of entries, buffer contents, number of entries on data diskette, Keyword or Category Selection, and Priority Selection. |
| R/D | (Day Level only) Repeats entry in move buffer. |
| R/S | (Day Level only) Scans for entry. |
| T | Displays account sums and totals. |
| T/C | Clears individual accounts sums to zero. |
| T/S | Calculates account sums. |
| T/M | Sets account multipliers. |
| T/R | Turns off running account sums. |
| T/B | Turns on running account sums. |
| T/S | Sets account sums. |
| S | (Day Level only) Writes new entries on data diskette before adding other entries or performing other tasks. |
| X | (Month Level only) Copies data to new diskette. |
| Y | (Month Level only) Moves calendar to indicated year. |
| Z | (Month Level only) Creates new data diskette. |

Account Descriptions

| Account Number | Symbol | Record Your Description |
|----------------|--------|-------------------------|
| 1 | ! | |
| 2 | ~ | |
| 3 | - | |
| 4 | ^ | |
| 5 | ^ | |
| 6 | ^ | |
| 7 | ^ | |
| 8 | ^ | |
| 9 | ^ | |