M. David Johnson
http://www.bds-soft.com
info@bds-soft.com

Preliminary Thoughts on a CoCo 2 High Memory Stack Engine

by M. David Johnson

2024/05/03

Abstract

Some preliminary thoughts and code are presented. I hope to be able to expand this code into a complete High Memory Stack Engine for the CoCo 2 during the coming year.

____

This paper is available online at:

http://www.bds-soft.com/cocoPapers.php .

=====

Table of Contents

Introduction

Ever since I wrote CF83 Forth back in 1991, I've had it in the back of my mind to write a generalized, lean and mean Stack Engine for the CoCo. The past couple of weeks, I finally got down to creating some proof-of-concept code which I may expand into just such an engine over the coming year.

Probably my favorite advantage of the Stack Engine concept is its minimum memory use. This fits right in with my oft-expressed goal of stuffing just as much into the 96K of the "64K" CoCo 2 as I possibly can.

[Google] presents the advantages and disadvantages of stacks as:

Advantages of Stacks:

· Simplicity: Stacks are a simple and easy-to-understand data structure, making them suitable for a wide range of applications.

· Efficiency: Push and pop operations on a stack can be performed in constant time (O(1)), providing efficient access to data.

· Last-in, First-out (LIFO): Stacks follow the LIFO principle, ensuring that the last element added to the stack is the first one removed. This behavior is useful in many scenarios, such as function calls and expression evaluation.

· Limited memory usage: Stacks only need to store the elements that have been pushed onto them, making them memory-efficient compared to other data structures.

Disadvantages of Stacks:

· Limited access: Elements in a stack can only be accessed from the top, making it difficult to retrieve or modify elements in the middle of the stack.

· Potential for overflow: If more elements are pushed onto a stack than it can hold, an overflow error will occur, resulting in a loss of data.

· Not suitable for random access: Stacks do not allow for random access to elements, making them unsuitable for applications where elements need to be accessed in a specific order

.

· Limited capacity: Stacks have a fixed capacity, which can be a limitation if the number of elements that need to be stored is unknown or highly variable.

Ideally, I hope to have the Stack Engine eventually reside in high RAM, between $8000 and $FF00. For the moment, the proof-of-concept code is still located below $7FFF for ease of development during the tight two-week horizon I had available running up to CoCoFEST this year.

The code I developed over the past two weeks is presented herein. Everything was built and tested on Vcc 2.1.0.7. Since the tests all rely heavily on printer output, no test has been attempted on an actual CoCo 2 yet.

Everything seems to work except that the testing code is just a bit wonky: Sometimes (for no reason I've been able to ascertain) a test will just jam in the middle of its output. If it does, simply doing a cold start (F9-F9 on the Vcc) and re-running the test will result in the test's proper completion. Go figure !?!

M.D.J. 2024/05/01
info@bds-soft.com

=====

General Methodology

For the 2024 CoCoFEST, all I've done is:

1. Establish the stack.

2. Provide some elementary PUSH and POP operations.

3. Build some 8-bit Stack Manipulation Routines.

The steps I would expect to pursue in continuing the development of the Stack Engine would be as follows:

4. The next task to be approached would be to develop the methodology for moving existing code from the disk to high RAM (above $7FFF) from within the Machine Language Foundation (MLF) code space; a sort of substitute "LOADM" if you will. This is not the same as position-independent code: The concept is to write and initially test the code in low memory, yet with the expectation that it will eventually be moved to, and retested within, high memory. This will involve three considerations:

    A. References to Low Memory addresses will be absolute.
    B. References to a routine's internal locations will be relative.
    C. References between routines will require re-addressing before Transfer to high memory (not a trivial task).

5. Develop 8-bit routines for:

    A. 8-bit Memory Access.
    B. 8-bit Logic.
    C. 8-bit Unsigned Integer Math.

6. Develop 16-bit routines similar to the routines of Items 3 and 5 above.

7. Develop character and string handling routines.

8. Develop flow-control routines.

9. Possibly develop some I/O routines and/or tie I/O to MLF code.

10. Develop a few system control routines (very few seem indicated at this point).

11. Consider the possibility of the following enhancements:

A. 32-bit, 64-bit, and 128-bit routines similar to the routines of Items 3 and 5 above.
B. Signed Integer routines.
C. IEEE Floating Point routines.
D. Complex Number routines.
E. Overlays ?
F. Other stuff ???

____

This paper is available online at:

http://www.bds-soft.com/cocoPapers.php .

=====

STKMAP: System Map

The Assembly Language listing:

```
*****
*
* STKMAP.ASM
* MDJ 2024/04/26
*
* U-STACK
* STACK ENGINE
*
* SYSTEM MAP
*
* FOR REFERENCE ONLY
*
*****

* THIS STACK ENGINE IS INTENDED FOR EXPERT USERS.

* THE 64K COCO2 IS VERY OLD, VERY SMALL, AND VERY
* SLOW - IT NEEDS ALL THE HELP WE CAN GIVE IT.

* IN THE INTEREST OF MAXIMIZING SPEED AND MINIMIZING
* MEMORY USAGE, THIS ENGINE ITSELF DOES NOT INCLUDE
* ANY ERROR CHECKING. ANY NECESSARY ERROR CHECKING
* IS TO BE PROVIDED BY THE USER.

* THE USER'S PROCEDURE FOR SETTING-UP AND RUNNING
* THE STACK ENGINE IS GENERALLY TO:
*   1. MAKE SURE ALL OF THE USER'S .BIN FILES,
*        AND ANY OTHER NEEDED FILES (INCLUDING
*        STKENGIN.BAS) ARE RESIDENT ON DISK.
*   2. SPECIFY THE BOTTOM OF THE STACK ADDRESS
*        AND THE STACK SIZE IN STKSSVAR.ASM.
*   3. RE-ASSEMBLE STKSSVAR.ASM.
*   4. SPECIFY THE START ROUTINE NAME (2 PLACES)
*        AND ADDRESS (1 PLACE) IN STKRUN.ASM.
*   5. RE-ASSEMBLE STKRUN.ASM.
*   6. MAKE SURE STKENGIN.BAS IS MODIFIED SO AS
*        TO LOADM ALL OF THE USER'S STACK
*        ENGINE .BIN FILES.
*   7. "RUN STKENGIN.BAS" FROM THE "OK" PROMPT.
* SCRATCHPAD VARIABLES
T1BYT   EQU     $536F   TEMPORARY 8-BIT VARIABLE
T2BYT   EQU     $5370   TEMPORARY 16-BIT VARIABLE
T2BYT2  EQU     $5372   TEMPORARY 16-BIT VARIABLE

* STACK ENGINE VARIABLES
ASTPTR  EQU     $53F0   AUXILIARY STACK POINTER
ASTBOT  EQU     $53F2   BOTTOM OF THE AUXILIARY STACK
ASTTOP  EQU     $53F4   TOP OF THE AUXILIARY STACK
ASTSIZ  EQU     $53F6   SIZE OF THE AUXILIARY STACK IN BYTES

STKPTR  EQU     $53F8   HOLDING VARIABLE FOR U-STACK POINTER
STKBOT  EQU     $53FA   BOTTOM OF THE U-STACK
STKTOP  EQU     $53FC   MAXIMUM TOP OF THE U-STACK
STKSIZ  EQU     $53FE   SIZE OF THE U-STACK IN BYTES

* ELEMENTARY STACK PUSHES AND POPS
PSHA    EQU     $5400   PUSH REGISTER A
PSHB    EQU     $5403   PUSH REGISTER B
PSHD    EQU     $5406   PUSH REGISTER D
PSHX    EQU     $5409   PUSH REGISTER X
PSHY    EQU     $540C   PUSH REGISTER Y
PSHMX   EQU     $540F   PUSH MEMORY BYTE ,X
PSHMY   EQU     $5414   PUSH MEMORY BYTE ,Y
PSHMXI  EQU     $5419   PUSH MEMORY BYTE ,X+
PSHMYI  EQU     $541E   PUSH MEMORY BYTE ,Y+
PSHMXD  EQU     $5423   PUSH MEMORY BYTE ,X-
PSHMYD  EQU     $542A   PUSH MEMORY BYTE ,Y-
POPA    EQU     $5431   POP REGISTER A
POPB    EQU     $5434   POP REGISTER B
POPD    EQU     $5437   POP REGISTER D
POPX    EQU     $543A   POP REGISTER X
POPY    EQU     $543D   POP REGISTER Y
POPMX   EQU     $5440   POP MEMORY BYTE ,X
POPMY   EQU     $5445   POP MEMORY BYTE ,Y
POPMXI  EQU     $544A   POP MEMORY BYTE ,X+
POPMYI  EQU     $544F   POP MEMORY BYTE ,Y+
POPMXD  EQU     $5454   POP MEMORY BYTE ,X-
POPMYD  EQU     $545B   POP MEMORY BYTE ,Y-

* 8-BIT STACK MANIPULATION ROUTINES
NOOP    EQU     $5470   NO OPERATION
STBOT   EQU     $5472   STACK BOTTOM
STTOP   EQU     $5478   STACK TOP
STSIZ   EQU     $547E   STACK SIZE
SPCUR   EQU     $5484   CURRENT STACK POINTER
STCLR   EQU     $5489   CLEAR THE STACK
```

```
DPTH08  EQU     $548F   STACK DEPTH IN BYTES
TOA08   EQU     $549B   8-BIT U-TO-S TRANSFER
FMA08   EQU     $54AC   8-BIT S-TO-U TRANSFER
AAT08   EQU     $54BD   8-BIT S-TO-U COPY
DROP08  EQU     $54D0   8-BIT DROP
DUP08   EQU     $54D3   8-BIT DUPLICATE
QDUP08  EQU     $54DA   8-BIT DUP IF NOT ZERO
SWAP08  EQU     $54E5   8-BIT SWAP
OVER08  EQU     $54EE   8-BIT OVER
ROT08   EQU     $54F9   8-BIT ROT
PICK08  EQU     $550E   8-BIT PICK
ROLL08  EQU     $5515   8-BIT ROLL

* STACK ENGINE SETUP ROUTINES
STKSSV  EQU     $5550   SET SYSTEM VARIABLES
STKRUN  EQU     $55A0   START THE STACK ENGINE
STKSUP  EQU     $55B0   ENTRY POINT

* PRINTER-SPECIFIC EXTERNAL ROUTINES
PTPCHR  EQU     $7F00   PUT CHARACTER TO PRINTER
PTCRLF  EQU     $7F20   PUT CRLF TO PRINTER
PTPBYT  EQU     $7F40   PUT BYTE TO PRINTER
PTBYTS  EQU     $7F80   PUT BYTE + SPACE TO PRINTER
PTPS00  EQU     $7F90   PUT 0TERM STRING TO PRINTER
PTPWRD  EQU     $7FA0   PUT WORD TO PRINTER
PTWRDS  EQU     $7FB0   PUT WORD + SPACE TO PRINTER
PTPDEC  EQU     $7FC0   PUT DECIMAL TO PRINTER
PTDECS  EQU     $7FF0   PUT DECIMAL + SPACE TO PRINTER
```

=====

STKSKPAD: Scratchpad Variables

The Assembly Language text listing:

```
                   00100 *****
                   00110 *
                   00120 * STKSKPAD.ASM
                   00130 * MDJ 2024/03/19
                   00140 *
                   00150 * U-STACK
                   00160 * STACK ENGINE
                   00170 *
                   00180 * SCRATCHPAD - 129 BYTES
                   00190 *
                   00200 * THE SCRATCHPAD
                   00210 * OCCUPIES $536F - $53EF
                   00220 * INCLUSIVE
                   00230 *
                   00240 * ROUTINES USE THIS FOR
                   00250 * STORAGE OF VARIABLES
                   00260 * WHICH ARE WHOLLY
                   00270 * INTERNAL TO THAT
                   00280 * ROUTINE, OR WHICH ARE
                   00290 * INVOLVED IN IMMEDIATE
                   00300 * TRANSFER OF DATA FROM
                   00310 * ONE ROUTINE TO ANOTHER.
                   00320 *
                   00330 *****
                   00340
536F               00350         ORG     $536F
                   00360
536F               00370 STKPAD  RMB     128
53EF               00380 ENDCH2  RMB     1
                   00390
          0000     00400         END
```

=====

STKSVARS: System Variables

The Assembly Language text listing:

```
00100 *****
00110 *
00120 * STKSVARS.ASM
00130 * MDJ 2024/04/29
00140 *
00150 * U-STACK
00160 * STACK ENGINE
00170 *
00180 * SYSTEM VARIABLES
00190 *
00200 * THESE VARIABLES OCCUPY $53F0 - $53FF INCLUSIVE.
00210 *
00220 * THIS STACK ENGINE INCORPORATES TWO SEPARATE STACKS:
00230 *   1. THE MAIN U-STACK
00240 *   2. AN AUXILIARY STACK, AKA THE A-STACK
00250 *
00260 * THESE ARE BOTH SEPARATE FROM THE SYSTEM OR S-STACK.
00270 *
00280 * THE BOTTOM ADDRESS OF A STACK IS NOT INCLUDED
00290 * AS PART OF THE STACK ITSELF. THE TOP ADDRESS OF
00300 * THAT STACK, HOWEVER, IS INCLUDED AS PART OF THE
00310 * STACK, I.E.:
00320 *
00330 * THE U-STACK IS INCLUSIVE OF STKTOP, BUT EXCLUSIVE
00340 * OF STKBOT, AND:
00350 *
00360 * THE A-STACK IS INCLUSIVE OF ASTTOP, BUT EXCLUSIVE
00370 * OF ASTBOT.
00380 *
00390 * MAXIMUM ADDRESS FOR THE U-STACK'S STKBOT = $FF00
```

```
                    * WHICH IS THE BEGINNING OF THE COCO 2'S I/O AREA.
                    *
                    * THE AUXILIARY STACK LOCATION DEFAULTS TO JUST
                    * BELOW THE U-STACK IN MEMORY. IN THAT LOCATION,
                    * ASTBOT = STKTOP.
                    *
                    * AND IT FOLLOWS THAT THE STACK SIZES ARE:
                    *   1. STKSIZ = STKBOT - STKTOP
                    *   2. ASTSIZ = ASTBOT - ASTTOP
                    *
                    * NOTE: THE U-STACK GROWS DOWNWARD FROM STKBOT
                    * USING REGISTER U AS THE STACK POINTER
                    *
                    * THE A-STACK GROWS DOWNWARD FROM ASTBOT. THE
                    * A-STACK ALSO USES REGISTER U AS THE STACK
                    * POINTER; HENCE THE TWO STACK POINTER TEMPORARY
                    * HOLDING VARIABLES, STKPTR AND ASTPTR, DESIGNED
                    * TO FACILITATE SWITCHING BETWEEN THE U-STACK
                    * AND THE A-STACK.
                    *
                    * *** IMPORTANT ***
                    * THE USER SHOULD GENERALLY NOT TRY TO ACCESS
                    * THE A-STACK DIRECTLY. IT SHOULD ONLY BE
                    * ACCESSED VIA THE TOAXX, FMAXX, AND AATXX
                    * ROUTINES (I.E. TOA08, FMA08, AAT08;
                    * TOA16, FMA16, AAT16, ETC.) PROCEED OTHERWISE
                    * AT YOUR OWN RISK.
                    *
                    *****

53F0                        ORG     $53F0

```

```
          53F0     00720 ASTPTR  EQU     $53F0   HOLDING
VARIABLE FOR A-STACK PO
INTER
          53F2     00730 ASTBOT  EQU     $53F2   BOTTOM OF THE
A-STACK
          53F4     00740 ASTTOP  EQU     $53F4   TOP OF THE A-
STACK
          53F6     00750 ASTSIZ  EQU     $53F6   SIZE OF THE A-
STACK IN BYTES
                   00760
          53F8     00770 STKPTR  EQU     $53F8   HOLDING
VARIABLE FOR U-STACK PO
INTER
          53FA     00780 STKBOT  EQU     $53FA   BOTTOM OF THE
U-STACK
          53FC     00790 STKTOP  EQU     $53FC   TOP OF THE U-
STACK
          53FE     00800 STKSIZ  EQU     $53FE   SIZE OF THE U-
STACK IN BYTES
                   00810
          0000     00820         END
```

=====

STKELEMT: Elementary Stack Pushes and Pops

The Assembly Language text listing:

```
00100 *****
00110 *
00120 * STKELEMT.ASM
00130 * MDJ 2024/04/20
00140 *
00150 * U-STACK
00160 * STACK ENGINE
00170 *
00180 * ELEMENTARY STACK
00190 * PUSHES AND POPS
00200 *
00210 * REGISTERS ARE NOT PRESERVED
00220 *
00230 *****
00240
00250 * IN THE LIGHT OF COMMON EXPERIENCE,
00260 * AND IN THE INTERESTS OF EFFICIENCY,
00270 * THESE ELEMENTARY STACK ROUTINES
00280 * ONLY INCLUDE PUSHING AND POPPING OF
00290 * REGA, REGB, REGD, REGX, REGY, AND
00300 * MEMORY BYTES.
00310
00320 * PUSHING AND POPPING OF REGDP, REGCC,
00330 * REGS, REGU, AND REGPC CAN BE
00340 * IMPLEMENTED BY THE USER IF AND
00350 * WHEN NECESSARY.
00360
00370 *****
00380 *
00390 * FOR PSHA, PSHB, PSHMX, PSHMY,
00400 *     PSHMXI, PSHMYI. PSHMXD, AND PSHMYD:
00410 *
00420 * U-STACK STATE ON ENTRY:
00430 *   ---
00440 *
00450 * U-STACK STATE ON EXIT:
00460 *    ,U = 8-BIT BYTE
```

```
                   00470 *
                   00480 *****
                   00490
                   00500 *****
                   00510 *
                   00520 * FOR PSHD, PSHX, AND PSHY:
                   00530 *
                   00540 * U-STACK STATE ON ENTRY:
                   00550 *   ---
                   00560 *
                   00570 * U-STACK STATE ON EXIT:
                   00580 *   1,U = 16-BIT WORD'S LOW BYTE
                   00590 *    ,U = 16-BIT WORD'S HIGH BYTE
                   00600 *
                   00610 *****
                   00620
                   00630 *****
                   00640 *
                   00650 * FOR POPA, POPB, POPMX, POPMY,
                   00660 *     POPMXI, POPMYI. POPMXD, AND
POPMYD:
                   00670 *
                   00680 * U-STACK STATE ON ENTRY:
                   00690 *    ,U = 8-BIT BYTE
                   00700 *
                   00710 * U-STACK STATE ON EXIT:
                   00720 *   ---
                   00730 *
                   00740 *****
                   00750
                   00760 *****
                   00770 *
                   00780 * FOR POPD, POPX, AND POPY:
                   00790 *
                   00800 * U-STACK STATE ON ENTRY:
                   00810 *   1,U = 16-BIT WORD'S LOW BYTE
                   00820 *    ,U = 16-BIT WORD'S HIGH BYTE
                   00830 *
                   00840 * U-STACK STATE ON EXIT:
                   00850 *   ---
                   00860 *
                   00870 *****
                   00880
5400               00890         ORG     $5400
                   00900
5400 36   02       00910 PSHA    PSHU    A       PUSH REGISTER A
5402 39            00920         RTS
```

```
                   00930
5403 36   04       00940 PSHB    PSHU    B          PUSH REGISTER B
5405 39            00950         RTS
                   00960
5406 36   06       00970 PSHD    PSHU    A,B        PUSH REGISTER D
5408 39            00980         RTS
                   00990
5409 36   10       01000 PSHX    PSHU    X          PUSH REGISTER X
540B 39            01010         RTS
                   01020
540C 36   20       01030 PSHY    PSHU    Y          PUSH REGISTER Y
540E 39            01040         RTS
                   01050
540F A6   84       01060 PSHMX   LDA     ,X         PUSH THE MEMORY
BYTE WHOSE
5411 36   02       01070         PSHU    A            ADDRESS IS IN
REGX
5413 39            01080         RTS
                   01090
5414 A6   A4       01100 PSHMY   LDA     ,Y         PUSH THE MEMORY
BYTE WHOSE
5416 36   02       01110         PSHU    A            ADDRESS IS IN
REGY
5418 39            01120         RTS
                   01130
5419 A6   80       01140 PSHMXI  LDA     ,X+        PUSH THE MEMORY
BYTE WHOSE
541B 36   02       01150         PSHU    A            ADDRESS IS IN
REGX,
541D 39            01160         RTS                  AND INCREMENT
REGX
                   01170
541E A6   A0       01180 PSHMYI  LDA     ,Y+        PUSH THE MEMORY
BYTE WHOSE
5420 36   02       01190         PSHU    A            ADDRESS IS IN
REGY,
5422 39            01200         RTS                  AND INCREMENT
REGY
                   01210
5423 A6   84       01220 PSHMXD  LDA     ,X         PUSH THE MEMORY
BYTE WHOSE
5425 36   02       01230         PSHU    A            ADDRESS IS IN
REGX,
5427 30   1F       01240         LEAX    -1,X         AND DECREMENT
REGX
5429 39            01250         RTS
                   01260
```

```
542A A6   A4        01270 PSHMYD  LDA     ,Y      PUSH THE MEMORY
BYTE WHOSE
542C 36   02        01280         PSHU    A         ADDRESS IS IN
REGY,
542E 31   3F        01290         LEAY    -1,Y      AND DECREMENT
REGY
5430 39             01300         RTS
                    01310
5431 37   02        01320 POPA    PULU    A       POP REGISTER A
5433 39             01330         RTS
                    01340
5434 37   04        01350 POPB    PULU    B       POP REGISTER B
5436 39             01360         RTS
                    01370
5437 37   06        01380 POPD    PULU    A,B     POP REGISTER D
5439 39             01390         RTS
                    01400
543A 37   10        01410 POPX    PULU    X       POP REGISTER X
543C 39             01420         RTS
                    01430
543D 37   20        01440 POPY    PULU    Y       POP REGISTER Y
543F 39             01450         RTS
                    01460
5440 37   02        01470 POPMX   PULU    A       POP THE MEMORY
BYTE WHOSE
5442 A7   84        01480         STA     ,X        ADDRESS IS IN
REGX
5444 39             01490         RTS
                    01500
5445 37   02        01510 POPMY   PULU    A       POP THE MEMORY
BYTE WHOSE
5447 A7   A4        01520         STA     ,Y        ADDRESS IS IN
REGY
5449 39             01530         RTS
                    01540
544A 37   02        01550 POPMXI  PULU    A       POP THE MEMORY
BYTE WHOSE
544C A7   80        01560         STA     ,X+       ADDRESS IS IN
REGX,
544E 39             01570         RTS               AND INCREMENT
REGX
                    01580
544F 37   02        01590 POPMYI  PULU    A       POP THE MEMORY
BYTE WHOSE
5451 A7   A0        01600         STA     ,Y+       ADDRESS IS IN
REGY,
5453 39           01610        RTS                 AND INCREMENT
REGY
                  01620
5454 37   02      01630 POPMXD  PULU    A        POP THE MEMORY
BYTE WHOSE
5456 A7   84      01640        STA     ,X          ADDRESS IS IN
REGX,
5458 30   1F      01650        LEAX    -1,X        AND DECREMENT
REGX
545A 39           01660        RTS
                  01670
545B 37   02      01680 POPMYD  PULU    A        POP THE MEMORY
BYTE WHOSE
545D A7   A4      01690        STA     ,Y          ADDRESS IS IN
REGY,
545F 31   3F      01700        LEAY    -1,Y        AND DECREMENT
REGY
5461 39           01710 ENDCHK  RTS
                  01720
5462              01730 SPARE   RMB     13
546F              01740 ENDCH2  RMB     1
                  01750
          0000    01760        END
```

=====

STKMNP08: 8-bit Stack Manipulation

The Assembly Language text listing:

```
                   00100 *****
                   00110 *
                   00120 * STKMNP08.ASM
                   00130 * MDJ 2024/04/30
                   00140 *
                   00150 * U-STACK
                   00160 * STACK ENGINE
                   00170 *
                   00180 * 8-BIT
                   00190 * STACK MANIPULATION
                   00200 * ROUTINES
                   00210 *
                   00220 *****
                   00230
                   00240 * SCRATCHPAD VARIABLES
          536F     00250 T1BYT   EQU     $536F   TEMPORARY 8-BIT
VARIABLE
          5370     00260 T2BYT   EQU     $5370   TEMPORARY 16-
BIT VARIABLE
          5372     00270 T2BYT2  EQU     $5372   TEMPORARY 16-
BIT VARIABLE
                   00280
                   00290 * STACK ENGINE VARIABLES
          53F0     00300 ASTPTR  EQU     $53F0   AUXILIARY STACK
POINTER
          53F2     00310 ASTBOT  EQU     $53F2   BOTTOM OF THE
AUXILIARY STACK
          53F4     00320 ASTTOP  EQU     $53F4   TOP OF THE
AUXILIARY STACK
          53F6     00330 ASTSIZ  EQU     $53F6   SIZE OF THE
AUXILIARY STACK IN
BYTES
                   00340
          53F8     00350 STKPTR  EQU     $53F8   HOLDING
VARIABLE FOR U-STACK PO
INTER
          53FA     00360 STKBOT  EQU     $53FA   BOTTOM OF THE
U-STACK
          53FC     00370 STKTOP  EQU     $53FC   MAXIMUM TOP OF
THE U-STACK
```

```
          53FE      STKSIZ  EQU     $53FE   SIZE OF THE U-
STACK IN BYTES

5470                        ORG     $5470

                   *****
                   *
                   * NOOP
                   * NO OPERATION
                   *
                   * DOES NOTHING EXCEPT
                   * WASTE TIME AND SPACE
                   *
                   *****

5470 12             NOOP    NOP             DO NOTHING
5471 39                     RTS

                   *****
                   *
                   * STBOT
                   * U-STACK BOTTOM
                   *
                   * PUT THE ADDRESS OF THE
                   * BOTTOM OF THE U-STACK
                   * ON THE STACK
                   *
                   * U-STACK STATE ON ENTRY:
                   *   ---
                   *
                   * U-STACK STATE ON EXIT:
                   *   1,U = LOW BYTE OF THE 16-BIT
ADDRESS
                   *    ,U = HIGH BYTE OF THE 16-BIT
ADDRESS
                   *
                   *****

5472 FC   53FA      STBOT   LDD     STKBOT  BOTTOM OF THE
U-STACK
5475 36   06                PSHU    D       PUT IT TO THE
U-STACK
5477 39                     RTS

                   *****
                   *
                   * STTOP
                   00800 * U-STACK TOP
                   00810 *
                   00820 * PUT THE ADDRESS OF THE
                   00830 * TOP OF THE U-STACK
                   00840 * ON THE U-STACK
                   00850 *
                   00860 * U-STACK STATE ON ENTRY:
                   00870 *   ---
                   00880 *
                   00890 * U-STACK STATE ON EXIT:
                   00900 *   1,U = LOW BYTE OF THE 16-BIT
ADDRESS
                   00910 *    ,U = HIGH BYTE OF THE 16-BIT
ADDRESS
                   00920 *
                   00930 *****
                   00940
5478 FC   53FC     00950 STTOP   LDD     STKTOP  TOP OF THE U-
STACK
547B 36   06       00960         PSHU    D       PUT IT TO THE
U-STACK
547D 39            00970         RTS
                   00980
                   00990 *****
                   01000 *
                   01010 * STSIZ
                   01020 * U-STACK SIZE
                   01030 *
                   01040 * PUT THE SIZE OF THE
                   01050 * STACK ON THE U-STACK
                   01060 *
                   01070 * U-STACK STATE ON ENTRY:
                   01080 *   ---
                   01090 *
                   01100 * U-STACK STATE ON EXIT:
                   01110 *   1,U = LOW BYTE OF THE 16-BIT U-
STACK SIZE
                   01120 *    ,U = HIGH BYTE OF THE 16-BIT U-
STACK SIZE
                   01130 *
                   01140 *****
                   01150
547E FC   53FE     01160 STSIZ   LDD     STKSIZ  SIZE OF THE U-
STACK
5481 36   06       01170         PSHU    D       PUT IT TO THE
U-STACK
5483 39            01180         RTS
```

```

                     *****
                     *
                     * SPCUR
                     * CURRENT U-STACK POINTER
                     *
                     * PUT THE ADDRESS OF THE
                     * CURRENT U-STACK POINTER
                     * ON THE U-STACK
                     *
                     * SPCUR IS DEFINED AS THE ADDRESS OF
                     * THE U-STACK POINTER JUST BEFORE
                     * SPCUR WAS EXECUTED
                     *
                     * U-STACK STATE ON ENTRY:
                     *   ---
                     *
                     * U-STACK STATE ON EXIT:
                     *   1,U = LOW BYTE OF THE 16-BIT
ADDRESS
                     *    ,U = HIGH BYTE OF THE 16-BIT
ADDRESS
                     *
                     *****

5484 1F   30         SPCUR   TFR     U,D     CURRENT U-STACK
POINTER
5486 36   06                 PSHU    D       PUT IT TO THE
U-STACK
5488 39                      RTS

                     *****
                     *
                     * STCLR
                     * CLEAR THE U-STACK
                     *
                     * MOVE THE U-STACK POINTER
                     * TO THE BOTTOM OF THE U-STACK
                     *
                     * U-STACK STATE ON ENTRY:
                     *    MAY OR MAY NOT HAVE ANY CONTENTS
                     *
                     * U-STACK STATE ON EXIT:
                     *   EMPTY
                     *
                     *****

```

```
5489 FC   53FA       STCLR   LDD     STKBOT  U-STACK BOTTOM
548C 1F   03                 TFR     D,U     PUT IT TO U-
STACK POINTER
548E 39                      RTS

                     *****
                     *
                     * DPTH08
                     * U-STACK DEPTH IN BYTES
                     *
                     * PUT THE DEPTH, IN BYTES,
                     * ONTO THE U-STACK
                     *
                     * THE DEPTH IS DEFINED AS THE NUMBER
                     * OF BYTES WHICH WERE ON THE U-STACK
                     * BEFORE THE DEPTH VALUE WAS ADDED
                     * TO THE U-STACK
                     *
                     * U-STACK STATE ON ENTRY:
                     *   ---
                     *
                     * U-STACK STATE ON EXIT:
                     *   1,U = LOW BYTE OF THE 16-BIT NUMBER
                     *    ,U = HIGH BYTE OF THE 16-BIT
NUMBER
                     *
                     *****

548F 1F   30         DPTH08  TFR     U,D     CURRENT U-STACK
POINTER
5491 36   06                 PSHU    D       PUT IT TO THE
U-STACK
5493 FC   53FA               LDD     STKBOT  BOTTOM OF THE
U-STACK
5496 A3   C1                 SUBD    ,U++    SUBTRACT, THEN
CLEAN U-STACK
5498 36   06                 PSHU    D       PUT NUMBER TO
THE U-STACK
549A 39                      RTS

                     *****
                     *
                     * NOTE: STBOT, STTOP, STSIZ, SPCUR,
STCLR, & DPTH08
                     *       DEFINE AND DESCRIBE THE STATE
OF THE U-STACK.
                     *
```

```
02000 *       SIMILAR ROUTINES TO DEFINE AND DESCRIBE THE
02010 *       STATE OF THE A-STACK ARE NOT CONSIDERED TO
02020 *       BE NECESSARY. SUCH CAN BE ADDED BY THE USER
02030 *       IF DESIRED.
02040 *
02050 *****
02060
02070 *****
02080 *
02090 * ++ GENERAL CAUTION:
02100 *
02110 * EVERY TOA08 SHOULD GENERALLY BE BALANCED
02120 * WITH A CORRESPONDING FMA08; BOTH WITHIN
02130 * DEFINED ROUTINES, AND ALSO WITHIN LOOPS.
02140 *
02150 *****
02160
02170 *****
02180 *
02190 * TOA08
02200 * 8-BIT U-TO-A TRANSFER
02210 *
02220 * POP ONE BYTE FROM THE
02230 * U-STACK AND PUSH IT
02240 * TO THE A-STACK
02250 *
02260 * STACK STATES ON ENTRY:
02270 *   U-STACK
02280 *      ,U = 8-BIT BYTE
02290 *   A-STACK
02300 *     ---
02310 *
02320 * STACK STATES ON EXIT:
02330 *   U-STACK
02340 *     ---
02350 *   A-STACK
02360 *      ,ASTPTR = 8-BIT BYTE
02370 *
02380 *****
02390
```

```
549B 37   02        02400 TOA08   PULU    A       GET BYTE FROM
U-STACK
549D FF   53F8      02410         STU     STKPTR  SAVE U-STACK
POINTER
54A0 FE   53F0      02420         LDU     ASTPTR  GET A-STACK
POINTER
54A3 36   02        02430         PSHU    A       PUT BYTE TO A-
STACK
54A5 FF   53F0      02440         STU     ASTPTR  SAVE A-STACK
POINTER
54A8 FE   53F8      02450         LDU     STKPTR  RESTORE U-STACK
POINTER
54AB 39             02460         RTS
                    02470
                    02480 *****
                    02490 *
                    02500 * FMA08
                    02510 * 8-BIT A-TO-U TRANSFER
                    02520 *
                    02530 * POP ONE BYTE FROM THE
                    02540 * A-STACK AND PUSH IT
                    02550 * TO THE U-STACK
                    02560 *
                    02570 * STACK STATES ON ENTRY:
                    02580 *   U-STACK
                    02590 *     ---
                    02600 *   A-STACK
                    02610 *       ,ASTPTR = 8-BIT BYTE
                    02620 *
                    02630 * STACK STATES ON EXIT:
                    02640 *   U-STACK
                    02650 *       ,U = 8-BIT BYTE
                    02660 *   A-STACK
                    02670 *     ---
                    02680 *
                    02690 *****
                    02700
54AC FF   53F8      02710 FMA08   STU     STKPTR  SAVE U-STACK
POINTER
54AF FE   53F0      02720         LDU     ASTPTR  GET A-STACK
POINTER
54B2 37   02        02730         PULU    A       GET BYTE FROM
A-STACK
54B4 FF   53F0      02740         STU     ASTPTR  SAVE A-STACK
POINTER
54B7 FE   53F8      02750         LDU     STKPTR  RESTORE U-STACK
POINTER
```

```
54BA 36   02        02760          PSHU    A       PUT BYTE TO U-
STACK
54BC 39             02770          RTS
                    02780
                    02790 *****
                    02800 *
                    02810 * AAT08
                    02820 * 8-BIT A-TO-U COPY
                    02830 *
                    02840 * COPY ONE BYTE FROM THE
                    02850 * A-STACK AND PUSH IT
                    02860 * TO THE U-STACK
                    02870 *
                    02880 * STACK STATES ON ENTRY:
                    02890 *   U-STACK
                    02900 *     ---
                    02910 *   A-STACK
                    02920 *       ,ASTPTR = 8-BIT BYTE
                    02930 *
                    02940 * STACK STATES ON EXIT:
                    02950 *   U-STACK
                    02960 *       ,U = 8-BIT BYTE
                    02970 *   A-STACK
                    02980 *       ,ASTPTR = 8-BIT BYTE
                    02990 *
                    03000 *****
                    03010
54BD FF   53F8      03020 AAT08   STU     STKPTR  SAVE U-STACK
POINTER
54C0 FE   53F0      03030          LDU     ASTPTR  GET A-STACK
POINTER
54C3 37   02        03040          PULU    A       GET BYTE FROM
A-STACK
54C5 36   02        03050          PSHU    A       PUT IT BACK
54C7 FF   53F0      03060          STU     ASTPTR  SAVE A-STACK
POINTER
54CA FE   53F8      03070          LDU     STKPTR  RESTORE U-STACK
POINTER
54CD 36   02        03080          PSHU    A       PUT BYTE TO U-
STACK
54CF 39             03090          RTS
                    03100
                    03110 *****
                    03120 *
                    03130 * DROP08
                    03140 * 8-BIT DROP
                    03150 *
```

```
                   * ONE BYTE IS REMOVED FROM THE STACK
                   *
                   * U-STACK STATE ON ENTRY:
                   *    ,U = THE BYTE
                   *
                   * U-STACK STATE ON EXIT:
                   *   ---
                   *
                   *****

54D0 33   41       DROP08  LEAU    1,U     REGU = REGU + 1
54D2 39                    RTS

                   *****
                   *
                   * DUP08
                   * 8-BIT DUPLICATE
                   *
                   * THE BYTE ON THE TOP OF
                   * THE STACK IS DUPLICATED
                   *
                   * U-STACK STATE ON ENTRY:
                   *    ,U = THE BYTE
                   *
                   * U-STACK STATE ON EXIT:
                   *   1,U = THE BYTE
                   *    ,U = THE COPY OF THE BYTE
                   *
                   *****

54D3 37   02       DUP08   PULU    A       POP THE BYTE
54D5 36   02               PSHU    A       PUT IT BACK
54D7 36   02               PSHU    A         TWICE
54D9 39                    RTS

                   *****
                   *
                   * QDUP08
                   * DUPLICATE ONE BYTE
                   * IF THAT BYTE IS NON-ZERO
                   *
                   * THE BYTE ON THE TOP OF
                   * THE STACK IS DUPLICATED
                   * IF IT IS NOT ZERO
                   *
                   * -----
                   * IF NON-ZERO
                   * U-STACK STATE ON ENTRY:
                   *    ,U = THE BYTE
                   *
                   * U-STACK STATE ON EXIT:
                   *   1,U = THE BYTE
                   *    ,U = THE COPY OF THE BYTE
                   * -----
                   *
                   * -----
                   * IF ZERO
                   * U-STACK STATE ON ENTRY:
                   *    ,U = THE BYTE = 0
                   *
                   * U-STACK STATE ON EXIT:
                   *    ,U = THE BYTE = 0
                   * -----
                   *
                   *****

54DA 37   02       QDUP08  PULU    A       POP THE BYTE
54DC 81   00               CMPA    #0      IS IT ZERO?
54DE 27   02               BEQ     QDP08Z  GO IF YES
54E0 36   02               PSHU    A       PUT IT BACK
54E2 36   02       QDP08Z  PSHU    A       PUT IT BACK
54E4 39                    RTS

                   *****
                   *
                   * SWAP08
                   * 8-BIT SWAP
                   *
                   * THE TOP TWO BYTES ON
                   * THE STACK ARE SWAPPED
                   *
                   * U-STACK STATE ON ENTRY:
                   *   1,U = BYTE B
                   *    ,U = BYTE A
                   *
                   * U-STACK STATE ON EXIT:
                   *   1,U = BYTE A
                   *    ,U = BYTE B
                   *
                   *****

54E5 37   02       SWAP08  PULU    A       POP BYTE A
54E7 37   04               PULU    B       POP BYTE B
54E9 36   02               PSHU    A       PUSH BYTE A
```

```
54EB 36   04        04100         PSHU    B       PUSH BYTE B
54ED 39             04110         RTS
                    04120
                    04130 *****
                    04140 *
                    04150 * OVER08
                    04160 * 8-BIT OVER
                    04170 *
                    04180 * THE SECOND BYTE ON
                    04190 * THE STACK IS COPIED TO
                    04200 * THE TOP OF THE STACK
                    04210 *
                    04220 * U-STACK STATE ON ENTRY:
                    04230 *   1,U = BYTE B
                    04240 *    ,U = BYTE A
                    04250 *
                    04260 * U-STACK STATE ON EXIT:
                    04270 *   2,U = BYTE B
                    04280 *   1,U = BYTE A
                    04290 *    ,U = BYTE B
                    04300 *
                    04310 *****
                    04320
54EE 37   02        04330 OVER08  PULU    A       POP BYTE A
54F0 37   04        04340         PULU    B       POP BYTE B
54F2 36   04        04350         PSHU    B       PUT B BACK
54F4 36   02        04360         PSHU    A       PUT A BACK
54F6 36   04        04370         PSHU    B       PUSH B AGAIN
54F8 39             04380         RTS
                    04390
                    04400 *****
                    04410 *
                    04420 * ROT08
                    04430 * 8-BIT ROTATE
                    04440 *
                    04450 * THE TOP THREE BYTES ON
                    04460 * THE STACK ARE ROTATED,
                    04470 * BRINGING THE DEEPEST TO
                    04480 * THE TOP OF THE STACK
                    04490 *
                    04500 * U-STACK STATE ON ENTRY:
                    04510 *   2,U = BYTE C
                    04520 *   1,U = BYTE B
                    04530 *    ,U = BYTE A
                    04540 *
                    04550 * U-STACK STATE ON EXIT:
                    04560 *   2,U = BYTE B
```

```
                   04570 *   1,U = BYTE A
                   04580 *    ,U = BYTE C
                   04590 *
                   04600 *****
                   04610
54F9 37   06       04620 ROT08   PULU    D       POP BYTES A AND
B
54FB FD   5370     04630         STD     T2BYT   PUT TO 16-BIT
TEMPORARY
54FE 37   02       04640         PULU    A       POP BYTE C
5500 B7   536F     04650         STA     T1BYT   PUT TO 8-BIT
TEMPORARY
5503 FC   5370     04660         LDD     T2BYT   GET 16-BIT
TEMPORARY
5506 36   06       04670         PSHU    D       PUSH BYTES A
AND B
5508 B6   536F     04680         LDA     T1BYT   GET 8-BIT
TEMPORARY
550B 36   02       04690         PSHU    A       PUSH BYTE C
550D 39            04700         RTS
                   04710
                   04720 *****
                   04730 *
                   04740 * PICK08
                   04750 * 8-BIT PICK
                   04760 *
                   04770 * BYTE N IS COPIED TO THE
                   04780 * TOP OF THE STACK
                   04790 *
                   04800 * N IS A 16-BIT UNSIGNED INTEGER
                   04810 *
                   04820 * BYTE N IS THE NTH BYTE
                   04830 * ON THE STACK, NOT COUNTING
                   04840 * THE TWO BYTES OF N ITSELF
                   04850 *
                   04860 * FOR EASE OF REFERENCE, (M)
                   04870 * REPRESENTS THE RELATIVE STACK
                   04880 * POSITIONS WITHOUT N
                   04890 *
                   04900 * U-STACK STATE ON ENTRY:
                   04910 *   N+2,U = BYTE N      (M)
                   04920 *     .
                   04930 *     .
                   04940 *     .
                   04950 *   3,U = BYTE B        (1)
                   04960 *   2,U = BYTE A        (0)
```

```
                  *   1,U = LOW BYTE OF THE 16-BIT NUMBER N
                  *    ,U = HIGH BYTE OF THE 16-BIT NUMBER N
                  *
                  * U-STACK STATE ON EXIT:
                  *   N+1,U = BYTE N      (M+1)
                  *     .
                  *     .
                  *     .
                  *   2,U = BYTE B        (2)
                  *   1,U = BYTE A        (1)
                  *    ,U = BYTE N        (0)
                  *
                  *****

550E 37   06      PICK08  PULU    D       POP N
5510 A6   CB              LDA     D,U     COPY BYTE N
5512 36   02              PSHU    A       PUT IT TO STACK
5514 39                   RTS

                  *****
                  *
                  * ROLL08
                  * 8-BIT ROLL
                  *
                  * BYTE N IS MOVED TO THE
                  * TOP OF THE STACK
                  *
                  * N IS A 16-BIT UNSIGNED INTEGER
                  *
                  * BYTE N IS THE NTH BYTE
                  * ON THE STACK, NOT COUNTING
                  * THE TWO BYTES OF N ITSELF
                  *
                  * BYTE N IS FIRST REMOVED AND THEN
                  * TRANSFERRED TO THE TOP OF THE STACK,
                  * SHIFTING THE REMAINING BYTES DOWN
                  * TO FILL IN THE VACATED POSITION.
                  *
                  * FOR EASE OF REFERENCE, (M)
                  * REPRESENTS THE RELATIVE STACK
                  * POSITIONS WITHOUT N
                  *
                  * U-STACK STATE ON ENTRY:
                  *   N+2,U = BYTE N      (M)
                  *   N+1,U = BYTE N-1    (M-1)
```

```
                   05420 *     N,U = BYTE N-2    (M-2)
                   05430 *    .
                   05440 *    .
                   05450 *    .
                   05460 *   3,U = BYTE B        (1)
                   05470 *   2,U = BYTE A        (0)
                   05480 *   1,U = LOW BYTE OF THE 16-BIT NUMBER
N
                   05490 *    ,U = HIGH BYTE OF THE 16-BIT
NUMBER N
                   05500 *
                   05510 * U-STACK STATE ON EXIT:
                   05520 *   N+1,U = BYTE N-1    (M)
                   05530 *     N,U = BYTE N-2    (M-1)
                   05540 *    .
                   05550 *    .
                   05560 *    .
                   05570 *   2,U = BYTE B        (2)
                   05580 *   1,U = BYTE A        (1)
                   05590 *    ,U = BYTE N        (0)
                   05600 *
                   05610 * CF. "ROLL" IN CF83
                   05620 *
                   05630 *****
                   05640
5515 37   10       05650 ROLL08  PULU    X       POP N FROM THE
U-STACK
5517 1F   12       05660         TFR     X,Y     COPY N TO REGY
                   05670
5519 8C   0000     05680 ROL001  CMPX    #0      REGX (N) = 0?
551C 27   08       05690         BEQ     ROL002  GO IF YES
551E 37   02       05700         PULU    A       POP 8-BITS FROM
THE U-STACK
5520 34   02       05710         PSHS    A       PUSH IT TO THE
S-STACK
5522 30   1F       05720         LEAX    -1,X    REGX (N) = REGX
- 1
5524 20   F3       05730         BRA     ROL001  RETURN FOR NEXT
BYTE
                   05740
5526 37   04       05750 ROL002  PULU    B       POP 8-BITS FROM
THE U-STACK
5528 1F   21       05760         TFR     Y,X     RESTORE
ORIGINAL N TO REGX
                   05770
552A 8C   0000     05780 ROL003  CMPX    #0      REGX (N) = 0?
552D 27   08       05790         BEQ     ROL004  GO IF YES
```

```
552F 35   02        05800         PULS    A       POP 8-BITS FROM
THE S-STACK
5531 36   02        05810         PSHU    A       PUSH IT TO THE
U-STACK
5533 30   1F        05820         LEAX    -1,X    REGX (N) = REGX
- 1
5535 20   F3        05830         BRA     ROL003  RETURN FOR NEXT
BYTE
                    05840
5537 36   04        05850 ROL004  PSHU    B       PUSH 8-BITS TO
U-STACK FROM REG
B
                    05860
5539 39             05870 ENDCHK  RTS
                    05880
553A                05890 SPARE   RMB     21
554F                05900 ENDCH2  RMB     1
                    05910
          0000      05920         END
```

=====

STKSSVAR: Set System Variables

The Assembly Language text listing:

```
                   00100 *****
                   00110 *
                   00120 * STKSSVAR.ASM
                   00130 * MDJ 2024/04/29
                   00140 *
                   00150 * U-STACK
                   00160 * STACK ENGINE
                   00170 *
                   00180 * SET SYSTEM VARIABLES
                   00190 *
                   00200 * SEE STKSVARS.ASM FOR DESCRIPTION OF
THE STACKS,
                   00210 * AND FOR THE RULES FOR STACK SYSTEM
VARIABLES.
                   00220 *
                   00230 *****
                   00240
                   00250 * STACK SYSTEM VARIABLES
          53F0     00260 ASTPTR  EQU     $53F0   AUXILIARY STACK
POINTER
          53F2     00270 ASTBOT  EQU     $53F2   BOTTOM OF THE
AUXILIARY STACK
          53F4     00280 ASTTOP  EQU     $53F4   TOP OF THE
AUXILIARY STACK
          53F6     00290 ASTSIZ  EQU     $53F6   SIZE OF THE
AUXILIARY STACK IN
BYTES
                   00300
          53F8     00310 STKPTR  EQU     $53F8   HOLDING
VARIABLE FOR U-STACK PO
INTER
          53FA     00320 STKBOT  EQU     $53FA   BOTTOM OF THE
U-STACK
          53FC     00330 STKTOP  EQU     $53FC   TOP OF THE U-
STACK
          53FE     00340 STKSIZ  EQU     $53FE   SIZE OF THE U-
STACK IN BYTES
                   00350
5550               00360         ORG     $5550
                   00370
```

```
5550 34   06        00380 STKSSV  PSHS    A,B
                    00390
                    00400 * THE STACK ENGINE DEVELOPER SHOULD
                    00410 * SET THE BOTTOM OF THE U-STACK ADDRESS
                    00420 * ON THE FOLLOWING CODE LINE BEFORE
                    00430 * RE-ASSEMBLING THIS FILE.
                    00440 * DEFAULT STKBOT ADDRESS = #$FF00
5552 CC   FF00      00450         LDD     #$FF00  DESIRED ADDRESS
                    00460
                    00470 * SET THE BOTTOM OF THE U-STACK
5555 FD   53FA      00480         STD     STKBOT
                    00490
                    00500 * THE STACK ENGINE DEVELOPER SHOULD
                    00510 * SET THE TOP OF THE U-STACK ON THE
                    00520 * FOLLOWING CODE LINE. THE DEFAULT
                    00530 * PROVIDES A STKSIZ = 512 BYTES.
                    00540 * DEFAULT STKTOP ADDRESS = #$FD00
5558 CC   FD00      00550         LDD     #$FD00  DESIRED ADDRESS
                    00560
                    00570 * SET THE TOP OF THE U-STACK
555B FD   53FC      00580         STD     STKTOP
                    00590
                    00600 * SET U-STACK SIZE, DEFINED AS:
                    00610 * STKSIZ = STKBOT - STKTOP
555E FC   53FA      00620         LDD     STKBOT
5561 B3   53FC      00630         SUBD    STKTOP
5564 FD   53FE      00640         STD     STKSIZ
                    00650
                    00660 * THE STACK ENGINE DEVELOPER SHOULD
                    00670 * SET THE BOTTOM OF THE A-STACK ADDRESS
                    00680 * ON THE FOLLOWING CODE LINE. THE
                    00690 * DEFAULT IS ASTBOT = STKTOP.
                    00700 * DEFAULT ASTBOT ADDRESS = #$FD00
5567 CC   FD00      00710         LDD     #$FD00  DESIRED ADDRESS
                    00720
                    00730 * SET THE BOTTOM OF THE A-STACK
556A FD   53F2      00740         STD     ASTBOT
                    00750
                    00760 * THE STACK ENGINE DEVELOPER SHOULD
                    00770 * SET THE TOP OF THE A-STACK ON THE
                    00780 * FOLLOWING CODE LINE. THE DEFAULT
                    00790 * PROVIDES AN ASTSIZ = 128 BYTES.
                    00800 * DEFAULT ASTTOP ADDRESS = #$FC80
556D CC   FC80      00810         LDD     #$FC80  DESIRED ADDRESS
                    00820
                    00830 * SET THE TOP OF THE A-STACK
5570 FD   53F4      00840         STD     ASTTOP
```

```

                    * SET A-STACK SIZE, DEFINED AS:
                    * ASTSIZ = ASTBOT - ASTTOP
5573 FC   53F2              LDD     ASTBOT
5576 B3   53F4              SUBD    ASTTOP
5579 FD   53F6              STD     ASTSIZ

                    * SET THE POINTER HOLDING VARIABLES
557C FC   53FA              LDD     STKBOT
557F FD   53F8              STD     STKPTR
5582 FC   53F2              LDD     ASTBOT
5585 FD   53F0              STD     ASTPTR

                    * SET THE U-STACK POINTER
5588 FE   53FA              LDU     STKBOT

558B 35   06                PULS    A,B

558D 39             ENDCHK  RTS

558E                SPARE   RMB     17
559F                ENDCH2  RMB     1

          0000              END
```

=====

STKRUN: Run the Start Routine

The Assembly Language text listing:

```
                        00100 *****
                        00110 *
                        00120 * STKRUN.ASM
                        00130 * MDJ 2024/04/20
                        00140 *
                        00150 * U-STACK
                        00160 * STACK ENGINE
                        00170 *
                        00180 * SPECIFY THE START ROUTINE
                        00190 * AND THEN JSR TO THAT ROUTINE
                        00200 *
                        00210 *****
                        00220
                        00230 * USER'S START ROUTINE.
                        00240 * THE STACK ENGINE DEVELOPER
                        00250 * SHOULD SET THE DESIRED STACK
                        00260 * START ROUTINE AND ADDRESS
                        00270 * ON THE FOLLOWING LINE BEFORE
                        00280 * RE-ASSEMBLING THIS FILE:
          7000          00290 TESTER  EQU     $7000
                        00300
55A0                    00310         ORG     $55A0
                        00320
                        00330 * CALL THE START ROUTINE.
                        00340 * THE ROUTINE MUST ALREADY
                        00350 * BE COMPLETE AND LOADED
                        00360 * INTO MEMORY BEFORE THE
                        00370 * FOLLOWING LINE IS
                        00380 * EXECUTED. THE ROUTINE'S
                        00390 * NAME IS TO BE SPECIFIED
                        00400 * BY THE USER ON THE
                        00410 * FOLLOWING LINE:
55A0 BD   7000          00420 STKRUN  JSR     TESTER
                        00430
55A3 39                 00440 ENDCHK  RTS
                        00450
55A4                    00460 SPARE   RMB     11
55AF                    00470 ENDCH2  RMB     1
                        00480
          0000          00490         END
```

STKSETUP: System Entry Point

The Assembly Language text listing:

```
                  *****
                  *
                  * STKSETUP.ASM
                  * MDJ 2024/04/23
                  *
                  * U-STACK
                  * STACK ENGINE
                  *
                  * THIS IS THE ENTRY POINT
                  * FOR SETTING UP THE
                  * STACK ENGINE.
                  *
                  * BEFORE EXECUTING THIS
                  * ROUTINE, THE DESIRED
                  * BOTTOM OF THE STACK
                  * AND STACK SIZE SHOULD
                  * BE SPECIFIED IN
                  * STKSSVAR.ASM
                  *
                  * THIS STACK ENGINE IS
                  * INTENDED FOR EXPERT USERS.
                  *
                  * IN THE INTEREST OF
                  * MAXIMIZING SPEED AND
                  * MINIMIZING MEMORY USAGE,
                  * THIS ENGINE ITSELF DOES
                  * NOT INCLUDE ANY ERROR
                  * CHECKING. ANY NECESSARY
                  * ERROR CHECKING IS TO BE
                  * PROVIDED BY THE USER.
                  *
                  *****

                  * EXTERNAL STACK ROUTINES
          5550    STKSSV  EQU     $5550   SET SYSTEM
VARIABLES
          55A0    STKRUN  EQU     $55A0   START THE STACK
ENGINE

55B0                      ORG     $55B0
```

```
                    00480
55B0 BD   5550      00490 STKSUP  JSR     STKSSV  SET SIZE AND
MAX TOP
                    00500
55B3 BD   55A0      00510         JSR     STKRUN  GO START THE
ENGINE
                    00520
55B6 39             00530 ENDCHK  RTS
                    00540
55B7                00550 SPARE   RMB     8
55BF                00560 ENDCH2  RMB     1
                    00570
          0000      00580         END
```

=====

TESTA: Elementary Stack Operations

The Assembly Language text listing:

```
                  00100 *****
                  00110 *
                  00120 * TESTA.ASM
                  00130 * MDJ 2024/05/02
                  00140 *
                  00150 * U-STACK
                  00160 * STACK ENGINE
                  00170 *
                  00180 * ELEMENTARY STACK OPERATIONS
                  00190 * TEST ROUTINE
                  00200 *
                  00210 * ASSEMBLE AS TESTER.BIN
                  00220 *
                  00230 *****
                  00240
                  00250 * LOW MEMORY VARIABLE
         0088     00260 CURPOS  EQU     $0088
                  00270
                  00280 * MLF ROUTINES
         400E     00290 VIDCLS  EQU     $400E
         40D7     00300 CRLF    EQU     $40D7
         4142     00310 POLCAT  EQU     $4142
         41A2     00320 COLD    EQU     $41A2
         420A     00330 PRTS00  EQU     $420A
                  00340
                  00350 * SCRATCHPAD VARIABLES
         536F     00360 TMEM01  EQU     $536F   8-BIT TEMPORARY
#1
         5370     00370 TMEM02  EQU     $5370   8-BIT TEMPORARY
#2
         5371     00380 TMEMXX  EQU     $5371   16-BIT
TEMPORARY #3 LOW BYTE
         5372     00390 TMEM03  EQU     $5372   16-BIT
TEMPORARY #3 HIGH BYTE
                  00400
                  00410 * STACK ENGINE VARIABLES
         53FA     00420 STKBOT  EQU     $53FA   BOTTOM OF THE
U-STACK
         53FC     00430 STKTOP  EQU     $53FC   MAXIMUM TOP OF
THE U-STACK
```

```
          53FE     00440 STKSIZ  EQU     $53FE   SIZE OF THE U-
STACK IN BYTES
                   00450
                   00460 * ELEMENTARY STACK PUSHES AND POPS
          5400     00470 PSHA    EQU     $5400   PUSH REGISTER A
          5403     00480 PSHB    EQU     $5403   PUSH REGISTER B
          5406     00490 PSHD    EQU     $5406   PUSH REGISTER D
          5409     00500 PSHX    EQU     $5409   PUSH REGISTER X
          540C     00510 PSHY    EQU     $540C   PUSH REGISTER Y
          540F     00520 PSHMX   EQU     $540F   PUSH MEMORY
BYTE ,X
          5414     00530 PSHMY   EQU     $5414   PUSH MEMORY
BYTE ,Y
          5419     00540 PSHMXI  EQU     $5419   PUSH MEMORY
BYTE ,X+
          541E     00550 PSHMYI  EQU     $541E   PUSH MEMORY
BYTE ,Y+
          5423     00560 PSHMXD  EQU     $5423   PUSH MEMORY
BYTE ,X-
          542A     00570 PSHMYD  EQU     $542A   PUSH MEMORY
BYTE ,Y-
          5431     00580 POPA    EQU     $5431   POP REGISTER A
          5434     00590 POPB    EQU     $5434   POP REGISTER B
          5437     00600 POPD    EQU     $5437   POP REGISTER D
          543A     00610 POPX    EQU     $543A   POP REGISTER X
          543D     00620 POPY    EQU     $543D   POP REGISTER Y
          5440     00630 POPMX   EQU     $5440   POP MEMORY BYTE
,X
          5445     00640 POPMY   EQU     $5445   POP MEMORY BYTE
,Y
          544A     00650 POPMXI  EQU     $544A   POP MEMORY BYTE
,X+
          544F     00660 POPMYI  EQU     $544F   POP MEMORY BYTE
,Y+
          5454     00670 POPMXD  EQU     $5454   POP MEMORY BYTE
,X-
          545B     00680 POPMYD  EQU     $545B   POP MEMORY BYTE
,Y-
                   00690
                   00700 * STACK ENGINE SETUP ROUTINES
          5550     00710 STKSSV  EQU     $5550   SET SYSTEM
VARIABLES
          55A0     00720 STKRUN  EQU     $55A0   START THE STACK
ENGINE
          55B0     00730 STKSUP  EQU     $55B0   ENTRY POINT
                   00740
                   00750 * PRINTER-SPECIFIC EXTERNAL ROUTINES
          7F00     00760 PTPCHR  EQU     $7F00   PUT CHARACTER
TO PRINTER
          7F20     00770 PTCRLF  EQU     $7F20   PUT CRLF TO
PRINTER
          7F40     00780 PTPBYT  EQU     $7F40   PUT BYTE TO
PRINTER
          7F80     00790 PTBYTS  EQU     $7F80   PUT BYTE +
SPACE TO PRINTER
          7F90     00800 PTPS00  EQU     $7F90   PUT 0TERM
STRING TO PRINTER
          7FA0     00810 PTPWRD  EQU     $7FA0   PUT WORD TO
PRINTER
          7FB0     00820 PTWRDS  EQU     $7FB0   PUT WORD +
SPACE TO PRINTER
          7FC0     00830 PTPDEC  EQU     $7FC0   PUT DECIMAL TO
PRINTER
          7FF0     00840 PTDECS  EQU     $7FF0   PUT DECIMAL +
SPACE TO PRINTER
                   00850
7000               00860         ORG     $7000
                   00870
7000 34   36       00880 TELEMN  PSHS    A,B,X,Y
                   00890
7002 BD   400E     00900         JSR     VIDCLS
7005 CC   0400     00910         LDD     #$0400
7008 DD   88       00920         STD     CURPOS
700A 7E   707E     00930         JMP     TESTR0
                   00940
700D      53       00950 TMSGR0  FCC     'STACK ENGINE -
ELEMENTARY STACK OPERAT
IONS TESTS'
          54
          41
          43
          4B
          20
          45
          4E
          47
          49
          4E
          45
          20
          2D
          20
          45
          4C
```

```
             45
             4D
             45
             4E
             54
             41
             52
             59
             20
             53
             54
             41
             43
             4B
             20
             4F
             50
             45
             52
             41
             54
             49
             4F
             4E
             53
             20
             54
             45
             53
             54
             53
703D         00        00960           FCB      0
703E         52        00970 TMSGR1    FCC      'READY THE PRINTER'
             45
             41
             44
             59
             20
             54
             48
             45
             20
             50
             52
             49
             4E
             54
```

```
          45
          52
704F      00        00980          FCB     0
7050      50        00990 TMSGR2   FCC     'PRESS ANY KEY WHEN
READY'
          52
          45
          53
          53
          20
          41
          4E
          59
          20
          4B
          45
          59
          20
          57
          48
          45
          4E
          20
          52
          45
          41
          44
          59
7068      00        01000          FCB     0
7069      52        01010 TMSGR3   FCC     'RUN COMPLETE'
          55
          4E
          20
          43
          4F
          4D
          50
          4C
          45
          54
          45
7075      00        01020          FCB     0
7076      54        01030 TMSG01   FCC     'TEST 01'
          45
          53
          54
          20
          30
          31
707D      00        01040         FCB      0
                    01050
707E 8E   700D      01060 TESTR0  LDX      #TMSGR0
7081 BD   420A      01070         JSR      PRTS00
7084 BD   40D7      01080         JSR      CRLF
7087 BD   40D7      01090         JSR      CRLF
                    01100
708A 8E   703E      01110         LDX      #TMSGR1
708D BD   420A      01120         JSR      PRTS00
7090 BD   40D7      01130         JSR      CRLF
                    01140
7093 8E   7050      01150         LDX      #TMSGR2
7096 BD   420A      01160         JSR      PRTS00
7099 BD   40D7      01170         JSR      CRLF
709C BD   40D7      01180         JSR      CRLF
                    01190
709F BD   4142      01200 LBL001  JSR      POLCAT
70A2 27   FB        01210         BEQ      LBL001
                    01220
                    01230 *TEST00
70A4 8E   700D      01240         LDX      #TMSGR0
70A7 BD   7F90      01250         JSR      PTPS00
70AA BD   7F20      01260         JSR      PTCRLF
70AD BD   7F20      01270         JSR      PTCRLF
                    01280
                    01290 *TEST01
70B0 8E   7076      01300         LDX      #TMSG01
70B3 BD   7F90      01310         JSR      PTPS00
70B6 BD   7F20      01320         JSR      PTCRLF
                    01330
70B9 1F   30        01340         TFR      U,D      STACK ADDRESS
PRE-CHECK
70BB BD   7FB0      01350         JSR      PTWRDS
                    01360
70BE C6   01        01370         LDB      #$01     PSHB/POPA TEST
70C0 BD   5403      01380         JSR      PSHB     SHOULD PUTPUT
01
70C3 BD   5431      01390         JSR      POPA
70C6 BD   7F80      01400         JSR      PTBYTS
                    01410
70C9 86   02        01420         LDA      #$02     PSHA/POPB TEST
70CB BD   5400      01430         JSR      PSHA     SHOULD PUTPUT
02
70CE BD   5434      01440         JSR      POPB
70D1 1F   98        01450         TFR      B,A
```

```
70D3 BD   7F80      01460          JSR      PTBYTS
                    01470
70D6 CC   0003      01480          LDD      #$0003  PSHD/POPD TEST
70D9 BD   5406      01490          JSR      PSHD    SHOULD PUTPUT
0003
70DC BD   5437      01500          JSR      POPD
70DF BD   7FB0      01510          JSR      PTWRDS
                    01520
70E2 8E   0004      01530          LDX      #$0004  PSHX/POPY TEST
70E5 BD   5409      01540          JSR      PSHX    SHOULD PUTPUT
0004
70E8 BD   543D      01550          JSR      POPY
70EB 1F   20        01560          TFR      Y,D
70ED BD   7FB0      01570          JSR      PTWRDS
                    01580
70F0 108E 0005      01590          LDY      #$0005  PSHY/POPX TEST
70F4 BD   540C      01600          JSR      PSHY    SHOULD PUTPUT
0005
70F7 BD   543A      01610          JSR      POPX
70FA 1F   10        01620          TFR      X,D
70FC BD   7FB0      01630          JSR      PTWRDS
                    01640
70FF 86   06        01650          LDA      #$06    PSHMX/POPMY
TEST
7101 B7   536F      01660          STA      TMEM01  SHOULD PUTPUT
06
7104 8E   536F      01670          LDX      #TMEM01
7107 BD   540F      01680          JSR      PSHMX
710A 108E 5370      01690          LDY      #TMEM02
710E BD   5445      01700          JSR      POPMY
7111 B6   5370      01710          LDA      TMEM02
7114 BD   7F80      01720          JSR      PTBYTS
                    01730
7117 86   07        01740          LDA      #$07    PSHMY/POPMX
TEST
7119 B7   536F      01750          STA      TMEM01  SHOULD PUTPUT
07
711C 108E 536F      01760          LDY      #TMEM01
7120 BD   5414      01770          JSR      PSHMY
7123 8E   5370      01780          LDX      #TMEM02
7126 BD   5440      01790          JSR      POPMX
7129 B6   5370      01800          LDA      TMEM02
712C BD   7F80      01810          JSR      PTBYTS
                    01820
712F 86   08        01830          LDA      #$08    PSHMXI/POPMYD
TEST
```

```
7131 B7   536F     01840          STA     TMEM01  SHOULD PUTPUT
0809
7134 4C            01850          INCA
7135 B7   5370     01860          STA     TMEM02
7138 8E   536F     01870          LDX     #TMEM01
713B BD   5419     01880          JSR     PSHMXI
713E BD   5419     01890          JSR     PSHMXI
7141 108E 5370     01900          LDY     #TMEM02
7145 BD   545B     01910          JSR     POPMYD
7148 BD   545B     01920          JSR     POPMYD
714B B6   536F     01930          LDA     TMEM01
714E BD   7F40     01940          JSR     PTPBYT
7151 B6   5370     01950          LDA     TMEM02
7154 BD   7F80     01960          JSR     PTBYTS
                   01970
7157 86   0A       01980          LDA     #$0A    PSHMYI/POPMXD
TEST
7159 B7   536F     01990          STA     TMEM01  SHOULD PUTPUT
0A0B
715C 4C            02000          INCA
715D B7   5370     02010          STA     TMEM02
7160 108E 536F     02020          LDY     #TMEM01
7164 BD   541E     02030          JSR     PSHMYI
7167 BD   541E     02040          JSR     PSHMYI
716A 8E   5370     02050          LDX     #TMEM02
716D BD   5454     02060          JSR     POPMXD
7170 BD   5454     02070          JSR     POPMXD
7173 B6   536F     02080          LDA     TMEM01
7176 BD   7F40     02090          JSR     PTPBYT
7179 B6   5370     02100          LDA     TMEM02
717C BD   7F80     02110          JSR     PTBYTS
                   02120
717F 1F   30       02130          TFR     U,D     STACK ADDRESS
POST-CHECK
7181 BD   7FB0     02140          JSR     PTWRDS
                   02150
7184 BD   7F20     02160          JSR     PTCRLF
7187 BD   7F20     02170          JSR     PTCRLF
                   02180
                   02190 *TSTEND
718A 8E   7069     02200          LDX     #TMSGR3
718D BD   7F90     02210          JSR     PTPS00
7190 BD   7F20     02220          JSR     PTCRLF
7193 BD   7F20     02230          JSR     PTCRLF
                   02240
7196 7E   41A2     02250          JMP     COLD   GO DO COLD START
                   02260
```

```
7199 35   36       02270         PULS    A,B,X,Y
                   02280
                   02290 *ENDCHK
719B 39            02300         RTS
                   02310
          0000     02320         END
```

=====

TESTB: First 8-bit Manipulation

The Assembly Language text listing:

```
                00100 *****
                00110 *
                00120 * TESTB.ASM
                00130 * MDJ 2024/05/02
                00140 *
                00150 * U-STACK
                00160 * STACK ENGINE
                00170 *
                00180 * TEST OF 8-BIT
                00190 * STACK MANIPULATIONS:
                00200 *   NOOP
                00210 *   STBOT
                00220 *   STTOP
                00230 *   STSIZ
                00240 *   SPCUR
                00250 *   DPTH08
                00260 *   STCLR
                00270 *   TOA08
                00280 *   FMA08
                00290 *   AAT08
                00300 *
                00310 * ASSEMBLE AS TESTER.BIN
                00320 *
                00330 *****
                00340
                00350 * LOW MEMORY VARIABLE
     0088       00360 CURPOS  EQU     $0088
                00370
                00380 * MLF ROUTINES
     400E       00390 VIDCLS  EQU     $400E
     40D7       00400 CRLF    EQU     $40D7
     4142       00410 POLCAT  EQU     $4142
     41A2       00420 COLD    EQU     $41A2
     420A       00430 PRTS00  EQU     $420A
                00440
                00450 * STACK ENGINE VARIABLES
     53F0       00460 ASTPTR  EQU     $53F0   AUXILIARY STACK
POINTER
     53F2       00470 ASTBOT  EQU     $53F2   BOTTOM OF THE
AUXILIARY STACK
```

```
          53F4     00480 ASTTOP  EQU     $53F4   TOP OF THE
AUXILIARY STACK
          53F6     00490 ASTSIZ  EQU     $53F6   SIZE OF THE
AUXILIARY STACK IN
BYTES
                   00500
          53F8     00510 STKPTR  EQU     $53F8   HOLDING
VARIABLE FOR U-STACK PO
INTER
          53FA     00520 STKBOT  EQU     $53FA   BOTTOM OF THE
U-STACK
          53FC     00530 STKTOP  EQU     $53FC   MAXIMUM TOP OF
THE U-STACK
          53FE     00540 STKSIZ  EQU     $53FE   SIZE OF THE U-
STACK IN BYTES
                   00550
                   00560 * 8-BIT STACK MANIPULATION ROUTINES
          5470     00570 NOOP    EQU     $5470   NO OPERATION
          5472     00580 STBOT   EQU     $5472   STACK BOTTOM
          5478     00590 STTOP   EQU     $5478   STACK TOP
          547E     00600 STSIZ   EQU     $547E   STACK SIZE
          5484     00610 SPCUR   EQU     $5484   CURRENT STACK
POINTER
          5489     00620 STCLR   EQU     $5489   CLEAR THE STACK
          548F     00630 DPTH08  EQU     $548F   STACK DEPTH IN
BYTES
          549B     00640 TOA08   EQU     $549B   8-BIT U-TO-S
TRANSFER
          54AC     00650 FMA08   EQU     $54AC   8-BIT S-TO-U
TRANSFER
          54BD     00660 AAT08   EQU     $54BD   8-BIT S-TO-U
COPY
                   00670
                   00680 * STACK ENGINE SETUP ROUTINES
          5550     00690 STKSSV  EQU     $5550   SET SYSTEM
VARIABLES
          55A0     00700 STKRUN  EQU     $55A0   START THE STACK
ENGINE
          55B0     00710 STKSUP  EQU     $55B0   ENTRY POINT
                   00720
                   00730 * PRINTER-SPECIFIC EXTERNAL ROUTINES
          7F00     00740 PTPCHR  EQU     $7F00   PUT CHARACTER
TO PRINTER
          7F20     00750 PTCRLF  EQU     $7F20   PUT CRLF TO
PRINTER
          7F40     00760 PTPBYT  EQU     $7F40   PUT BYTE TO
PRINTER
```

```
         7F80     00770 PTBYTS  EQU     $7F80   PUT BYTE +
SPACE TO PRINTER
         7F90     00780 PTPS00  EQU     $7F90   PUT 0TERM
STRING TO PRINTER
         7FA0     00790 PTPWRD  EQU     $7FA0   PUT WORD TO
PRINTER
         7FB0     00800 PTWRDS  EQU     $7FB0   PUT WORD +
SPACE TO PRINTER
         7FC0     00810 PTPDEC  EQU     $7FC0   PUT DECIMAL TO
PRINTER
         7FF0     00820 PTDECS  EQU     $7FF0   PUT DECIMAL +
SPACE TO PRINTER
                  00830
7000              00840         ORG     $7000
                  00850
7000 34  36       00860 TMNP08  PSHS    A,B,X,Y
                  00870
7002 BD  400E     00880         JSR     VIDCLS
7005 CC  0400     00890         LDD     #$0400
7008 DD  88       00900         STD     CURPOS
700A 7E  709A     00910         JMP     TESTR0
                  00920
700D     53       00930 TMSGR0  FCC     'STACK ENGINE - 8-BIT
STACK MANIPULATIO
NS TESTS'
         54
         41
         43
         4B
         20
         45
         4E
         47
         49
         4E
         45
         20
         2D
         20
         38
         2D
         42
         49
         54
         20
         53
         54
```

```
          41
          43
          4B
          20
          4D
          41
          4E
          49
          50
          55
          4C
          41
          54
          49
          4F
          4E
          53
          20
          54
          45
          53
          54
          53
703B      00        00940          FCB     0
703C      52        00950 TMSGR1   FCC     'READY THE PRINTER'
          45
          41
          44
          59
          20
          54
          48
          45
          20
          50
          52
          49
          4E
          54
          45
          52
704D      00        00960          FCB     0
704E      50        00970 TMSGR2   FCC     'PRESS ANY KEY WHEN
READY'
          52
          45
          53
          53
          20
          41
          4E
          59
          20
          4B
          45
          59
          20
          57
          48
          45
          4E
          20
          52
          45
          41
          44
          59
7066      00        00980          FCB      0
7067      52        00990 TMSGR3   FCC      'RUN COMPLETE'
          55
          4E
          20
          43
          4F
          4D
          50
          4C
          45
          54
          45
7073      00        01000          FCB      0
7074      54        01010 TMSG01   FCC      'TEST 01 - STACK OPS
PRE-CHECK'
          45
          53
          54
          20
          30
          31
          20
          2D
          20
          53
          54
          41
          43
          4B
          20
          4F
          50
          53
          20
          50
          52
          45
          2D
          43
          48
          45
          43
          4B
7091      00      01020          FCB     0
7092      54      01030 TMSG02   FCC     'TEST 02'
          45
          53
          54
          20
          30
          32
7099      00      01040          FCB     0
                  01050
709A 8E   700D    01060 TESTR0   LDX     #TMSGR0
709D BD   420A    01070          JSR     PRTS00
70A0 BD   40D7    01080          JSR     CRLF
70A3 BD   40D7    01090          JSR     CRLF
                  01100
70A6 8E   703C    01110          LDX     #TMSGR1
70A9 BD   420A    01120          JSR     PRTS00
70AC BD   40D7    01130          JSR     CRLF
                  01140
70AF 8E   704E    01150          LDX     #TMSGR2
70B2 BD   420A    01160          JSR     PRTS00
70B5 BD   40D7    01170          JSR     CRLF
70B8 BD   40D7    01180          JSR     CRLF
                  01190
70BB BD   4142    01200 LBL001   JSR     POLCAT
70BE 27   FB      01210          BEQ     LBL001
                  01220
                  01230 *TEST00
70C0 8E   700D    01240          LDX     #TMSGR0
70C3 BD   7F90    01250          JSR     PTPS00
```

```
70C6 BD   7F20             JSR      PTCRLF
70C9 BD   7F20             JSR      PTCRLF

                     *TEST01
70CC 8E   7074             LDX      #TMSG01
70CF BD   7F90             JSR      PTPS00
70D2 BD   7F20             JSR      PTCRLF

70D5 1F   30               TFR      U,D      STACK ADDRESS
PRE-CHECK
70D7 BD   7FB0             JSR      PTWRDS
70DA 86   FC               LDA      #$FC
70DC 36   02               PSHU     A
70DE 1F   30               TFR      U,D
70E0 BD   7FB0             JSR      PTWRDS   THIS LINE
SHOULD REPORT XXXX-1
70E3 A6   C4               LDA      ,U
70E5 BD   7F80             JSR      PTBYTS   THIS LINE
SHOULD REPORT FC
70E8 37   02               PULU     A
70EA 1F   30               TFR      U,D
70EC 1F   30               TFR      U,D      STACK ADDRESS
POST-CHECK
70EE BD   7FB0             JSR      PTWRDS
70F1 BD   7F20             JSR      PTCRLF
70F4 BD   7F20             JSR      PTCRLF

                     *TEST02
70F7 8E   7092             LDX      #TMSG02
70FA BD   7F90             JSR      PTPS00
70FD BD   7F20             JSR      PTCRLF

                     *NOOP
7100 BD   5470             JSR      NOOP     NOOP TEST
7103 86   0B               LDA      #$0B     SHOULD OUTPUT
0B
7105 BD   7F80             JSR      PTBYTS   BUT THAT JUST
INDICATES
                     *                       WE GOT PAST THE
NOOP
                     *                       WITHOUT
CRASHING THE SYSTEM

                     *STBOT
7108 BD   5472             JSR      STBOT    BOTTOM OF U-
STACK TEST
```

```
710B 37   06        01630          PULU    D       SHOULD REPORT
FF00
710D BD   7FB0      01640          JSR     PTWRDS
                    01650
                    01660 *STTOP
7110 BD   5478      01670          JSR     STTOP   TOP OF U-STACK
TEST
7113 37   06        01680          PULU    D       SHOULD REPORT
FD00
7115 BD   7FB0      01690          JSR     PTWRDS
                    01700
                    01710 *STSIZ
7118 BD   547E      01720          JSR     STSIZ   U-STACK SIZE
TEST
711B 37   06        01730          PULU    D       SHOULD REPORT
0200
711D BD   7FB0      01740          JSR     PTWRDS
7120 BD   7F20      01750          JSR     PTCRLF
                    01760
                    01770 *SPCUR
7123 1F   30        01780          TFR     U,D     STACK ADDRESS
PRE-CHECK
7125 BD   7FB0      01790          JSR     PTWRDS
7128 4F             01800          CLRA            STACK POINTER
TEST
7129 36   02        01810          PSHU    A       SHOULD REPORT
FEFB
712B 36   02        01820          PSHU    A
712D 36   02        01830          PSHU    A
712F 36   02        01840          PSHU    A
7131 36   02        01850          PSHU    A
7133 BD   5484      01860          JSR     SPCUR
7136 37   06        01870          PULU    D
7138 BD   7FB0      01880          JSR     PTWRDS
713B 1F   30        01890          TFR     U,D     STACK ADDRESS
POST-CHECK
713D BD   7FB0      01900          JSR     PTWRDS
7140 BD   7F20      01910          JSR     PTCRLF
                    01920
                    01930 *DPTH08
7143 1F   30        01940          TFR     U,D     STACK ADDRESS
PRE-CHECK
7145 BD   7FB0      01950          JSR     PTWRDS
7148 BD   548F      01960          JSR     DPTH08  DPTH08 TEST
714B 37   06        01970          PULU    D       SHOULD REPORT
0005
714D BD   7FB0      01980          JSR     PTWRDS
```

```
7150 1F   30        01990         TFR     U,D     STACK ADDRESS
PRE-CHECK
7152 BD   7FB0      02000         JSR     PTWRDS
7155 BD   7F20      02010         JSR     PTCRLF
                    02020
                    02030 *STCLR
7158 1F   30        02040         TFR     U,D     STACK ADDRESS
PRE-CHECK
715A BD   7FB0      02050         JSR     PTWRDS
715D BD   5489      02060         JSR     STCLR   STCLR TEST
7160 BD   5484      02070         JSR     SPCUR   SHOULD REPORT
FF00
7163 37   06        02080         PULU    D       FOLLOWED BY
0000
7165 BD   7FB0      02090         JSR     PTWRDS
7168 BD   548F      02100         JSR     DPTH08
716B 37   06        02110         PULU    D
716D BD   7FB0      02120         JSR     PTWRDS
7170 1F   30        02130         TFR     U,D     STACK ADDRESS
POST-CHECK
7172 BD   7FB0      02140         JSR     PTWRDS
7175 BD   7F20      02150         JSR     PTCRLF
                    02160
                    02170 *TOA08/FMA08
7178 1F   30        02180         TFR     U,D     STACK ADDRESS
PRE-CHECK
717A BD   7FB0      02190         JSR     PTWRDS
717D 86   0B        02200         LDA     #$0B    TOA08/FMA08
TEST
717F 36   02        02210         PSHU    A       SHOULD REPORT
0D
7181 4C             02220         INCA            FOLLOWED BY 0C
7182 36   02        02230         PSHU    A       FOLLOWED BY 0D
7184 4C             02240         INCA
7185 36   02        02250         PSHU    A
7187 BD   7F80      02260         JSR     PTBYTS
718A BD   549B      02270         JSR     TOA08
718D 37   02        02280         PULU    A
718F 36   02        02290         PSHU    A
7191 BD   7F80      02300         JSR     PTBYTS
7194 BD   54AC      02310         JSR     FMA08
7197 37   02        02320         PULU    A
7199 36   02        02330         PSHU    A
719B BD   7F80      02340         JSR     PTBYTS
719E 37   02        02350         PULU    A       CLEAN THE STACK
71A0 37   02        02360         PULU    A
71A2 37   02        02370         PULU    A
```

```
71A4 1F   30       02380          TFR     U,D     STACK ADDRESS
POST-CHECK
71A6 BD   7FB0     02390          JSR     PTWRDS
71A9 BD   7F20     02400          JSR     PTCRLF
                   02410
                   02420 *AAT08
71AC 1F   30       02430          TFR     U,D     STACK ADDRESS
PRE-CHECK
71AE BD   7FB0     02440          JSR     PTWRDS
71B1 FF   53F8     02450          STU     STKPTR  AAT08 TEST
71B4 FE   53F0     02460          LDU     ASTPTR  SHOULD REPORT
0E 0E
71B7 86   0E       02470          LDA     #$0E
71B9 36   02       02480          PSHU    A
71BB FF   53F0     02490          STU     ASTPTR
71BE FE   53F8     02500          LDU     STKPTR
71C1 BD   54BD     02510          JSR     AAT08
71C4 37   02       02520          PULU    A
71C6 BD   7F80     02530          JSR     PTBYTS
71C9 BD   54AC     02540          JSR     FMA08   CLEAR THE A-
STACK
71CC 37   02       02550          PULU    A
71CE BD   7F80     02560          JSR     PTBYTS
71D1 1F   30       02570          TFR     U,D     STACK ADDRESS
POST-CHECK
71D3 BD   7FB0     02580          JSR     PTWRDS
71D6 BD   7F20     02590          JSR     PTCRLF
71D9 BD   7F20     02600          JSR     PTCRLF
                   02610
                   02620 *TSTEND
71DC 8E   7067     02630          LDX     #TMSGR3
71DF BD   7F90     02640          JSR     PTPS00
71E2 BD   7F20     02650          JSR     PTCRLF
71E5 BD   7F20     02660          JSR     PTCRLF
                   02670
71E8 7E   41A2     02680          JMP     COLD   GO DO COLD START
                   02690
71EB 35   36       02700          PULS    A,B,X,Y
                   02710
                   02720 *ENDCHK
71ED 39            02730          RTS
                   02740
          0000     02750          END
```

=====

TESTC: Second 8-bit Manipulation

The Assembly Language text listing:

```
                00100 *****
                 00110 *
                 00120 * TESTC.ASM
                 00130 * MDJ 2024/05/02
                 00140 *
                 00150 * U-STACK
                 00160 * STACK ENGINE
                 00170 *
                 00180 * TEST OF 8-BIT
                 00190 * STACK MANIPULATIONS:
                 00200 *   DROP08
                 00210 *   DUP08
                 00220 *   QDUP08
                 00230 *   SWAP08
                 00240 *   OVER08
                 00250 *   ROT08
                 00260 *   PICK08
                 00270 *
                 00280 * ASSEMBLE AS TESTER.BIN
                 00290 *
                 00300 *****
                 00310
                 00320 * LOW MEMORY VARIABLE
          0088   00330 CURPOS  EQU     $0088
                 00340
                 00350 * MLF ROUTINES
          400E   00360 VIDCLS  EQU     $400E
          40D7   00370 CRLF    EQU     $40D7
          4142   00380 POLCAT  EQU     $4142
          41A2   00390 COLD    EQU     $41A2
          420A   00400 PRTS00  EQU     $420A
                 00410
                 00420 * STACK ENGINE VARIABLES
          53F0   00430 ASTPTR  EQU     $53F0   AUXILIARY STACK
POINTER
          53F2   00440 ASTBOT  EQU     $53F2   BOTTOM OF THE
AUXILIARY STACK
          53F4   00450 ASTTOP  EQU     $53F4   TOP OF THE
AUXILIARY STACK
```

```
          53F6     00460 ASTSIZ  EQU     $53F6   SIZE OF THE
AUXILIARY STACK IN
BYTES
                   00470
          53F8     00480 STKPTR  EQU     $53F8   HOLDING
VARIABLE FOR U-STACK PO
INTER
          53FA     00490 STKBOT  EQU     $53FA   BOTTOM OF THE
U-STACK
          53FC     00500 STKTOP  EQU     $53FC   MAXIMUM TOP OF
THE U-STACK
          53FE     00510 STKSIZ  EQU     $53FE   SIZE OF THE U-
STACK IN BYTES
                   00520
                   00530 * 8-BIT STACK MANIPULATION ROUTINES
          54D0     00540 DROP08  EQU     $54D0   8-BIT DROP
          54D3     00550 DUP08   EQU     $54D3   8-BIT DUPLICATE
          54DA     00560 QDUP08  EQU     $54DA   8-BIT DUP IF
NOT ZERO
          54E5     00570 SWAP08  EQU     $54E5   8-BIT SWAP
          54EE     00580 OVER08  EQU     $54EE   8-BIT OVER
          54F9     00590 ROT08   EQU     $54F9   8-BIT ROT
          550E     00600 PICK08  EQU     $550E   8-BIT PICK
          5515     00610 ROLL08  EQU     $5515   8-BIT ROLL
                   00620
                   00630 * STACK ENGINE SETUP ROUTINES
          5550     00640 STKSSV  EQU     $5550   SET SYSTEM
VARIABLES
          55A0     00650 STKRUN  EQU     $55A0   START THE STACK
ENGINE
          55B0     00660 STKSUP  EQU     $55B0   ENTRY POINT
                   00670
                   00680 * PRINTER-SPECIFIC EXTERNAL ROUTINES
          7F00     00690 PTPCHR  EQU     $7F00   PUT CHARACTER
TO PRINTER
          7F20     00700 PTCRLF  EQU     $7F20   PUT CRLF TO
PRINTER
          7F40     00710 PTPBYT  EQU     $7F40   PUT BYTE TO
PRINTER
          7F80     00720 PTBYTS  EQU     $7F80   PUT BYTE +
SPACE TO PRINTER
          7F90     00730 PTPS00  EQU     $7F90   PUT 0TERM
STRING TO PRINTER
          7FA0     00740 PTPWRD  EQU     $7FA0   PUT WORD TO
PRINTER
          7FB0     00750 PTWRDS  EQU     $7FB0   PUT WORD +
SPACE TO PRINTER
```

```
          7FC0      00760 PTPDEC  EQU     $7FC0   PUT DECIMAL TO
PRINTER
          7FF0      00770 PTDECS  EQU     $7FF0   PUT DECIMAL +
SPACE TO PRINTER
                    00780
7000                00790         ORG     $7000
                    00800
7000 34   36        00810 TMNP08  PSHS    A,B,X,Y
                    00820
7002 BD   400E      00830         JSR     VIDCLS
7005 CC   0400      00840         LDD     #$0400
7008 DD   88        00850         STD     CURPOS
700A 7E   70D2      00860         JMP     TESTR0
                    00870
700D      53        00880 TMSGR0  FCC     'STACK ENGINE - 8-BIT
STACK MANIPULATIO
NS TESTS'
          54
          41
          43
          4B
          20
          45
          4E
          47
          49
          4E
          45
          20
          2D
          20
          38
          2D
          42
          49
          54
          20
          53
          54
          41
          43
          4B
          20
          4D
          41
          4E
          49
          50
          55
          4C
          41
          54
          49
          4F
          4E
          53
          20
          54
          45
          53
          54
          53
703B      00       00890          FCB     0
703C      52       00900 TMSGR1   FCC     'READY THE PRINTER'
          45
          41
          44
          59
          20
          54
          48
          45
          20
          50
          52
          49
          4E
          54
          45
          52
704D      00       00910          FCB     0
704E      50       00920 TMSGR2   FCC     'PRESS ANY KEY WHEN
READY'
          52
          45
          53
          53
          20
          41
          4E
          59
          20
          4B
          45
          59
          20
          57
          48
          45
          4E
          20
          52
          45
          41
          44
          59
7066      00       00930        FCB     0
7067      52       00940 TMSGR3 FCC     'RUN COMPLETE'
          55
          4E
          20
          43
          4F
          4D
          50
          4C
          45
          54
          45
7073      00       00950        FCB     0
7074      44       00960 TMSG03 FCC     'DROP08 TEST'
          52
          4F
          50
          30
          38
          20
          54
          45
          53
          54
707F      00       00970        FCB     0
7080      44       00980 TMSG04 FCC     'DUP08 TEST'
          55
          50
          30
          38
          20
          54
          45
          53
```

```
        54
708A    00      00990          FCB    0
708B    51      01000 TMSG05   FCC    'QDUP08 TEST'
        44
        55
        50
        30
        38
        20
        54
        45
        53
        54
7096    00      01010          FCB    0
7097    53      01020 TMSG06   FCC    'SWAP08 TEST'
        57
        41
        50
        30
        38
        20
        54
        45
        53
        54
70A2    00      01030          FCB    0
70A3    4F      01040 TMSG07   FCC    'OVER08 TEST'
        56
        45
        52
        30
        38
        20
        54
        45
        53
        54
70AE    00      01050          FCB    0
70AF    52      01060 TMSG08   FCC    'ROT08 TEST'
        4F
        54
        30
        38
        20
        54
        45
        53
```

```
             54
70B9         00        01070          FCB      0
70BA         50        01080 TMSG09   FCC      'PICK08 TEST'
             49
             43
             4B
             30
             38
             20
             54
             45
             53
             54
70C5         00        01090          FCB      0
70C6         52        01100 TMSG10   FCC      'ROLL08 TEST'
             4F
             4C
             4C
             30
             38
             20
             54
             45
             53
             54
70D1         00        01110          FCB      0
                       01120
70D2 8E      700D      01130 TESTR0   LDX      #TMSGR0
70D5 BD      420A      01140          JSR      PRTS00
70D8 BD      40D7      01150          JSR      CRLF
70DB BD      40D7      01160          JSR      CRLF
                       01170
70DE 8E      703C      01180          LDX      #TMSGR1
70E1 BD      420A      01190          JSR      PRTS00
70E4 BD      40D7      01200          JSR      CRLF
                       01210
70E7 8E      704E      01220          LDX      #TMSGR2
70EA BD      420A      01230          JSR      PRTS00
70ED BD      40D7      01240          JSR      CRLF
70F0 BD      40D7      01250          JSR      CRLF
                       01260
70F3 BD      4142      01270 LBL001   JSR      POLCAT
70F6 27      FB        01280          BEQ      LBL001
                       01290
                       01300 *TEST00
70F8 8E      700D      01310          LDX      #TMSGR0
70FB BD      7F90      01320          JSR      PTPS00
```

```
70FE BD   7F20     01330          JSR     PTCRLF
7101 BD   7F20     01340          JSR     PTCRLF
                   01350
                   01360 *TEST03
                   01370 *DROP08
7104 8E   7074     01380          LDX     #TMSG03 DROP08 TEST
7107 BD   7F90     01390          JSR     PTPS00  SHOULD REPORT
11 10 0F
710A BD   7F20     01400          JSR     PTCRLF
710D 1F   30       01410          TFR     U,D     STACK ADDRESS
PRE-CHECK
710F BD   7FB0     01420          JSR     PTWRDS
7112 86   0F       01430          LDA     #$0F
7114 36   02       01440          PSHU    A
7116 4C            01450          INCA
7117 36   02       01460          PSHU    A
7119 4C            01470          INCA
711A 36   02       01480          PSHU    A
711C BD   7F80     01490          JSR     PTBYTS
711F BD   54D0     01500          JSR     DROP08
7122 A6   C4       01510          LDA     ,U
7124 BD   7F80     01520          JSR     PTBYTS
7127 BD   54D0     01530          JSR     DROP08
712A A6   C4       01540          LDA     ,U
712C BD   7F80     01550          JSR     PTBYTS
712F BD   54D0     01560          JSR     DROP08
7132 1F   30       01570          TFR     U,D     STACK ADDRESS
POST-CHECK
7134 BD   7FB0     01580          JSR     PTWRDS
7137 BD   7F20     01590          JSR     PTCRLF
713A BD   7F20     01600          JSR     PTCRLF
                   01610
                   01620 *TEST04
                   01630 *DUP08
713D 8E   7080     01640          LDX     #TMSG04 DUP08 TEST
7140 BD   7F90     01650          JSR     PTPS00  SHOULD REPORT
12 12
7143 BD   7F20     01660          JSR     PTCRLF
7146 1F   30       01670          TFR     U,D     STACK ADDRESS
PRE-CHECK
7148 BD   7FB0     01680          JSR     PTWRDS
714B 86   12       01690          LDA     #$12
714D 36   02       01700          PSHU    A
714F BD   54D3     01710          JSR     DUP08
7152 37   02       01720          PULU    A
7154 BD   7F80     01730          JSR     PTBYTS
7157 37   02       01740          PULU    A
```

```
7159 BD   7F80      01750          JSR     PTBYTS
715C 1F   30        01760          TFR     U,D     STACK ADDRESS
POST-CHECK
715E BD   7FB0      01770          JSR     PTWRDS
7161 BD   7F20      01780          JSR     PTCRLF
7164 BD   7F20      01790          JSR     PTCRLF
                    01800
                    01810 *TEST05
                    01820 *QDUP08
7167 8E   708B      01830          LDX     #TMSG05 QDUP08 TEST
716A BD   7F90      01840          JSR     PTPS00  SHOULD REPORT
13 13 0F
716D BD   7F20      01850          JSR     PTCRLF  FOLLOWED BY 00
0F
7170 1F   30        01860          TFR     U,D     STACK ADDRESS
PRE-CHECK
7172 BD   7FB0      01870          JSR     PTWRDS
7175 86   0F        01880          LDA     #$0F    END FLAG
7177 36   02        01890          PSHU    A
7179 86   13        01900          LDA     #$13    THE NON-ZERO
VALUE
717B 36   02        01910          PSHU    A
717D BD   54DA      01920          JSR     QDUP08
7180 37   02        01930          PULU    A       FIRST #$13
7182 BD   7F80      01940          JSR     PTBYTS
7185 37   02        01950          PULU    A       SECOND #$13
7187 BD   7F80      01960          JSR     PTBYTS
718A 37   02        01970          PULU    A       #$0F FLAG
718C BD   7F80      01980          JSR     PTBYTS
718F 1F   30        01990          TFR     U,D     STACK ADDRESS
MID-CHECK
7191 BD   7FB0      02000          JSR     PTWRDS
7194 86   0F        02010          LDA     #$0F    END FLAG
7196 36   02        02020          PSHU    A
7198 86   00        02030          LDA     #$00    THE ZERO VALUE
719A 36   02        02040          PSHU    A
719C BD   54DA      02050          JSR     QDUP08
719F 37   02        02060          PULU    A       THE #$00
71A1 BD   7F80      02070          JSR     PTBYTS
71A4 37   02        02080          PULU    A       #$0F FLAG
71A6 BD   7F80      02090          JSR     PTBYTS
71A9 1F   30        02100          TFR     U,D     STACK ADDRESS
POST-CHECK
71AB BD   7FB0      02110          JSR     PTWRDS
71AE BD   7F20      02120          JSR     PTCRLF
71B1 BD   7F20      02130          JSR     PTCRLF
                    02140
```

```
                   02150 *TEST06
                   02160 *SWAP08
71B4 8E   7097     02170          LDX     #TMSG06 SWAP08 TEST
71B7 BD   7F90     02180          JSR     PTPS00  SHOULD REPORT
14 15
71BA BD   7F20     02190          JSR     PTCRLF
71BD 1F   30       02200          TFR     U,D     STACK ADDRESS
PRE-CHECK
71BF BD   7FB0     02210          JSR     PTWRDS
71C2 86   14       02220          LDA     #$14
71C4 36   02       02230          PSHU    A
71C6 4C            02240          INCA
71C7 36   02       02250          PSHU    A
71C9 BD   54E5     02260          JSR     SWAP08
71CC 37   02       02270          PULU    A
71CE BD   7F80     02280          JSR     PTBYTS
71D1 37   02       02290          PULU    A
71D3 BD   7F80     02300          JSR     PTBYTS
71D6 1F   30       02310          TFR     U,D     STACK ADDRESS
POST-CHECK
71D8 BD   7FB0     02320          JSR     PTWRDS
71DB BD   7F20     02330          JSR     PTCRLF
71DE BD   7F20     02340          JSR     PTCRLF
                   02350
                   02360 *TEST07
                   02370 *OVER08
71E1 8E   70A3     02380          LDX     #TMSG07 OVER08 TEST
71E4 BD   7F90     02390          JSR     PTPS00  SHOULD REPORT
16 17 16
71E7 BD   7F20     02400          JSR     PTCRLF
71EA 1F   30       02410          TFR     U,D     STACK ADDRESS
PRE-CHECK
71EC BD   7FB0     02420          JSR     PTWRDS
71EF 86   16       02430          LDA     #$16
71F1 36   02       02440          PSHU    A
71F3 4C            02450          INCA
71F4 36   02       02460          PSHU    A
71F6 BD   54EE     02470          JSR     OVER08
71F9 37   02       02480          PULU    A
71FB BD   7F80     02490          JSR     PTBYTS
71FE 37   02       02500          PULU    A
7200 BD   7F80     02510          JSR     PTBYTS
7203 37   02       02520          PULU    A
7205 BD   7F80     02530          JSR     PTBYTS
7208 1F   30       02540          TFR     U,D     STACK ADDRESS
POST-CHECK
720A BD   7FB0     02550          JSR     PTWRDS
```

```
720D BD   7F20      02560           JSR     PTCRLF
7210 BD   7F20      02570           JSR     PTCRLF
                    02580
                    02590 *TEST08
                    02600 *ROT08
7213 8E   70AF      02610           LDX     #TMSG08 ROT08 TEST
7216 BD   7F90      02620           JSR     PTPS00  SHOULD REPORT
18 1A 19
7219 BD   7F20      02630           JSR     PTCRLF
721C 1F   30        02640           TFR     U,D     STACK ADDRESS
PRE-CHECK
721E BD   7FB0      02650           JSR     PTWRDS
7221 86   18        02660           LDA     #$18
7223 36   02        02670           PSHU    A
7225 4C             02680           INCA
7226 36   02        02690           PSHU    A
7228 4C             02700           INCA
7229 36   02        02710           PSHU    A
722B BD   54F9      02720           JSR     ROT08
722E 37   02        02730           PULU    A
7230 BD   7F80      02740           JSR     PTBYTS
7233 37   02        02750           PULU    A
7235 BD   7F80      02760           JSR     PTBYTS
7238 37   02        02770           PULU    A
723A BD   7F80      02780           JSR     PTBYTS
723D 1F   30        02790           TFR     U,D     STACK ADDRESS
POST-CHECK
723F BD   7FB0      02800           JSR     PTWRDS
7242 BD   7F20      02810           JSR     PTCRLF
7245 BD   7F20      02820           JSR     PTCRLF
                    02830
                    02840 *TEST09
                    02850 *PICK08
7248 8E   70BA      02860           LDX     #TMSG09 PICK08 TEST
724B BD   7F90      02870           JSR     PTPS00  WITH N = #$0004
724E BD   7F20      02880           JSR     PTCRLF  SHOULD REPORT
1B 1F 1E 1D 1C 1B
 1A
7251 1F   30        02890           TFR     U,D     STACK ADDRESS
PRE-CHECK
7253 BD   7FB0      02900           JSR     PTWRDS
7256 86   1A        02910           LDA     #$1A
7258 36   02        02920           PSHU    A
725A 4C             02930           INCA
725B 36   02        02940           PSHU    A
725D 4C             02950           INCA
725E 36   02        02960           PSHU    A
```

```
7260 4C             02970          INCA
7261 36   02        02980          PSHU    A
7263 4C             02990          INCA
7264 36   02        03000          PSHU    A
7266 4C             03010          INCA
7267 36   02        03020          PSHU    A
7269 CC   0004      03030          LDD     #$0004
726C 36   06        03040          PSHU    D
726E BD   550E      03050          JSR     PICK08
7271 37   02        03060          PULU    A
7273 BD   7F80      03070          JSR     PTBYTS
7276 37   02        03080          PULU    A
7278 BD   7F80      03090          JSR     PTBYTS
727B 37   02        03100          PULU    A
727D BD   7F80      03110          JSR     PTBYTS
7280 37   02        03120          PULU    A
7282 BD   7F80      03130          JSR     PTBYTS
7285 37   02        03140          PULU    A
7287 BD   7F80      03150          JSR     PTBYTS
728A 37   02        03160          PULU    A
728C BD   7F80      03170          JSR     PTBYTS
728F 37   02        03180          PULU    A
7291 BD   7F80      03190          JSR     PTBYTS
7294 1F   30        03200          TFR     U,D     STACK ADDRESS
POST-CHECK
7296 BD   7FB0      03210          JSR     PTWRDS
7299 BD   7F20      03220          JSR     PTCRLF
729C BD   7F20      03230          JSR     PTCRLF
                    03240
                    03250 *TSTEND
729F 8E   7067      03260          LDX     #TMSGR3
72A2 BD   7F90      03270          JSR     PTPS00
72A5 BD   7F20      03280          JSR     PTCRLF
72A8 BD   7F20      03290          JSR     PTCRLF
                    03300
72AB 7E   41A2      03310          JMP     COLD   GO DO COLD START
                    03320
72AE 35   36        03330          PULS    A,B,X,Y
                    03340
                    03350 *ENDCHK
72B0 39             03360          RTS
                    03370
          0000      03380          END
```

=====

TESTD: Third 8-bit Manipulation

The Assembly Language text listing:

```
                 00100 *****
                 00110 *
                 00120 * TESTD.ASM
                 00130 * MDJ 2024/05/02
                 00140 *
                 00150 * U-STACK
                 00160 * STACK ENGINE
                 00170 *
                 00180 * TEST OF 8-BIT
                 00190 * STACK MANIPULATIONS:
                 00200 *   ROLL08
                 00210 *
                 00220 * ASSEMBLE AS TESTER.BIN
                 00230 *
                 00240 *****
                 00250
                 00260 * LOW MEMORY VARIABLE
        0088     00270 CURPOS  EQU     $0088
                 00280
                 00290 * MLF ROUTINES
        400E     00300 VIDCLS  EQU     $400E
        40D7     00310 CRLF    EQU     $40D7
        4142     00320 POLCAT  EQU     $4142
        41A2     00330 COLD    EQU     $41A2
        420A     00340 PRTS00  EQU     $420A
                 00350
                 00360 * STACK ENGINE VARIABLES
        53F0     00370 ASTPTR  EQU     $53F0   AUXILIARY STACK
POINTER
        53F2     00380 ASTBOT  EQU     $53F2   BOTTOM OF THE
AUXILIARY STACK
        53F4     00390 ASTTOP  EQU     $53F4   TOP OF THE
AUXILIARY STACK
        53F6     00400 ASTSIZ  EQU     $53F6   SIZE OF THE
AUXILIARY STACK IN
BYTES
                 00410
        53F8     00420 STKPTR  EQU     $53F8   HOLDING
VARIABLE FOR U-STACK PO
INTER
```

```
         53FA     00430 STKBOT  EQU     $53FA   BOTTOM OF THE
U-STACK
         53FC     00440 STKTOP  EQU     $53FC   MAXIMUM TOP OF
THE U-STACK
         53FE     00450 STKSIZ  EQU     $53FE   SIZE OF THE U-
STACK IN BYTES
                  00460
                  00470 * 8-BIT STACK MANIPULATION ROUTINES
         54D0     00480 DROP08  EQU     $54D0   8-BIT DROP
         54D3     00490 DUP08   EQU     $54D3   8-BIT DUPLICATE
         54DA     00500 QDUP08  EQU     $54DA   8-BIT DUP IF
NOT ZERO
         54E5     00510 SWAP08  EQU     $54E5   8-BIT SWAP
         54EE     00520 OVER08  EQU     $54EE   8-BIT OVER
         54F9     00530 ROT08   EQU     $54F9   8-BIT ROT
         550E     00540 PICK08  EQU     $550E   8-BIT PICK
         5515     00550 ROLL08  EQU     $5515   8-BIT ROLL
                  00560
                  00570 * STACK ENGINE SETUP ROUTINES
         5550     00580 STKSSV  EQU     $5550   SET SYSTEM
VARIABLES
         55A0     00590 STKRUN  EQU     $55A0   START THE STACK
ENGINE
         55B0     00600 STKSUP  EQU     $55B0   ENTRY POINT
                  00610
                  00620 * PRINTER-SPECIFIC EXTERNAL ROUTINES
         7F00     00630 PTPCHR  EQU     $7F00   PUT CHARACTER
TO PRINTER
         7F20     00640 PTCRLF  EQU     $7F20   PUT CRLF TO
PRINTER
         7F40     00650 PTPBYT  EQU     $7F40   PUT BYTE TO
PRINTER
         7F80     00660 PTBYTS  EQU     $7F80   PUT BYTE +
SPACE TO PRINTER
         7F90     00670 PTPS00  EQU     $7F90   PUT 0TERM
STRING TO PRINTER
         7FA0     00680 PTPWRD  EQU     $7FA0   PUT WORD TO
PRINTER
         7FB0     00690 PTWRDS  EQU     $7FB0   PUT WORD +
SPACE TO PRINTER
         7FC0     00700 PTPDEC  EQU     $7FC0   PUT DECIMAL TO
PRINTER
         7FF0     00710 PTDECS  EQU     $7FF0   PUT DECIMAL +
SPACE TO PRINTER
                  00720
7000              00730         ORG     $7000
                  00740
```

```
7000 34   36       00750 TMNP08  PSHS    A,B,X,Y
                   00760
7002 BD   400E     00770         JSR     VIDCLS
7005 CC   0400     00780         LDD     #$0400
7008 DD   88       00790         STD     CURPOS
700A 7E   7080     00800         JMP     TESTR0
                   00810
700D      53       00820 TMSGR0  FCC     'STACK ENGINE - 8-BIT
STACK MANIPULATIO
NS TESTS'
          54
          41
          43
          4B
          20
          45
          4E
          47
          49
          4E
          45
          20
          2D
          20
          38
          2D
          42
          49
          54
          20
          53
          54
          41
          43
          4B
          20
          4D
          41
          4E
          49
          50
          55
          4C
          41
          54
          49
          4F
          4E
          53
          20
          54
          45
          53
          54
          53
703B      00        00830          FCB      0
703C      52        00840 TMSGR1   FCC      'READY THE PRINTER'
          45
          41
          44
          59
          20
          54
          48
          45
          20
          50
          52
          49
          4E
          54
          45
          52
704D      00        00850          FCB      0
704E      50        00860 TMSGR2   FCC      'PRESS ANY KEY WHEN
READY'
          52
          45
          53
          53
          20
          41
          4E
          59
          20
          4B
          45
          59
          20
          57
          48
          45
          4E
          20
          52
          45
          41
          44
          59
7066      00        00870             FCB     0
7067      52        00880 TMSGR3      FCC     'RUN COMPLETE'
          55
          4E
          20
          43
          4F
          4D
          50
          4C
          45
          54
          45
7073      00        00890             FCB     0
7074      52        00900 TMSG10      FCC     'ROLL08 TEST'
          4F
          4C
          4C
          30
          38
          20
          54
          45
          53
          54
707F      00        00910             FCB     0
                    00920
7080 8E   700D      00930 TESTR0      LDX     #TMSGR0
7083 BD   420A      00940             JSR     PRTS00
7086 BD   40D7      00950             JSR     CRLF
7089 BD   40D7      00960             JSR     CRLF
                    00970
708C 8E   703C      00980             LDX     #TMSGR1
708F BD   420A      00990             JSR     PRTS00
7092 BD   40D7      01000             JSR     CRLF
                    01010
7095 8E   704E      01020             LDX     #TMSGR2
7098 BD   420A      01030             JSR     PRTS00
709B BD   40D7      01040             JSR     CRLF
709E BD   40D7      01050             JSR     CRLF
                    01060
70A1 BD   4142      01070 LBL001      JSR     POLCAT
```

```
70A4 27   FB        01080          BEQ     LBL001
                    01090
                    01100 *TEST00
70A6 8E   700D      01110          LDX     #TMSGR0
70A9 BD   7F90      01120          JSR     PTPS00
70AC BD   7F20      01130          JSR     PTCRLF
70AF BD   7F20      01140          JSR     PTCRLF
                    01150
                    01160 *TEST10
                    01170 *ROLL08
70B2 8E   7074      01180          LDX     #TMSG10 ROLL08 TEST
70B5 BD   7F90      01190          JSR     PTPS00  WITH N = #$0004
70B8 BD   7F20      01200          JSR     PTCRLF  SHOULD REPORT
1B 1F 1E 1D 1C 1A
70BB 1F   30        01210          TFR     U,D     STACK ADDRESS
PRE-CHECK
70BD BD   7FB0      01220          JSR     PTWRDS
70C0 86   1A        01230          LDA     #$1A
70C2 36   02        01240          PSHU    A
70C4 4C             01250          INCA
70C5 36   02        01260          PSHU    A
70C7 4C             01270          INCA
70C8 36   02        01280          PSHU    A
70CA 4C             01290          INCA
70CB 36   02        01300          PSHU    A
70CD 4C             01310          INCA
70CE 36   02        01320          PSHU    A
70D0 4C             01330          INCA
70D1 36   02        01340          PSHU    A
70D3 CC   0004      01350          LDD     #$0004
70D6 36   06        01360          PSHU    D
70D8 1F   30        01370          TFR     U,D     STACK ADDRESS
MID-CHECK #1
70DA BD   7FB0      01380          JSR     PTWRDS
70DD BD   5515      01390          JSR     ROLL08
70E0 1F   30        01400          TFR     U,D     STACK ADDRESS
MID-CHECK #2
70E2 BD   7FB0      01410          JSR     PTWRDS
70E5 37   02        01420          PULU    A
70E7 BD   7F80      01430          JSR     PTBYTS
70EA 37   02        01440          PULU    A
70EC BD   7F80      01450          JSR     PTBYTS
70EF 37   02        01460          PULU    A
70F1 BD   7F80      01470          JSR     PTBYTS
70F4 37   02        01480          PULU    A
70F6 BD   7F80      01490          JSR     PTBYTS
70F9 37   02        01500          PULU    A
```

```
70FB BD   7F80     01510        JSR     PTBYTS
70FE 37   02       01520        PULU    A
7100 BD   7F80     01530        JSR     PTBYTS
7103 1F   30       01540        TFR     U,D     STACK ADDRESS
POST-CHECK
7105 BD   7FB0     01550        JSR     PTWRDS
7108 BD   7F20     01560        JSR     PTCRLF
710B BD   7F20     01570        JSR     PTCRLF
                   01580
                   01590 *TSTEND
710E 8E   7067     01600        LDX     #TMSGR3
7111 BD   7F90     01610        JSR     PTPS00
7114 BD   7F20     01620        JSR     PTCRLF
7117 BD   7F20     01630        JSR     PTCRLF
                   01640
711A 7E   41A2     01650        JMP     COLD   GO DO COLD START
                   01660
711D 35   36       01670        PULS    A,B,X,Y
                   01680
                   01690 *ENDCHK
711F 39            01700        RTS
                   01710
          0000     01720        END
```

=====

TESTER.BAS: Testing Control Program

The BASIC Language listing:

```
1000 '*****
1010 '*
1020 '* TESTER.BAS
1030 '* MDJ 2024/04/30
1040 '*
1050 '* U-STACK
1060 '* STACK ENGINE
1070 '*
1080 '* TESTING CONTROL PROGRAM
1090 '*
1100 '*****
1105 '

1500 PRINT
1510 PRINT "WORKING *";
1520 '

2000 'SETUP MEMORY
2010 CLEAR 0,&H4000
2020 PCLEAR 4
2030 PRINT "*";
2040 '

2100 LOADM "MLBASE.BIN"      'ML FOUNDATION
2110 LOADM "STKSKPAD.BIN"    'SCRATCHPAD
2120 LOADM "STKSVARS.BIN"    'STACK ENGINE SYSTEM VARIABLES
2130 LOADM "STKSSVAR.BIN"    'SET SYSTEM VARIABLES
2140 LOADM "STKRUN.BIN"      'START THE STACK ENGINE
2150 LOADM "STKSETUP.BIN"    'ENTRY POINT
2160 PRINT "*";
2165 '

3000 LOADM "STKELEMT.BIN"    'ELEMENTARY STACK PUSHES AND POPS
3010 LOADM "STKMNP08.BIN"    '8-BIT STACK MANIPULATION ROUTINES
3990 PRINT "*";
3995 '

4000 LOADM "TESTER.BIN"      'THE TEST SUITE
4010 PRINT "*";
4015 '
```

```
4100 LOADM "PTPPKG.BIN"     'PRINTER CONTROL PACKAGE
4110 PRINT "*";
4115 '

4998 'GRAPHICS SETUP NOT REQUIRED
4999 'FOR THE STACK ENGINE
5000 'SETUP GRAPHICS
5010 'PMODE 4,1
5020 'PCLS 1
5030 'SCREEN 1,0
5040 '

7000 'STKSETUP.BIN RUN ADDRESS = &H55B0
7010 'PUT IT TO THE ML FOUNDATION'S
7020 'REGPC (AT $H400A)
7030 POKE &H400A, &H55
7040 POKE &H400B, &HB0
7050 'GO START THE RUN IN ALLRAM MODE
7060 EXEC &H4403  'STRTUP
7070 '

32767 END
```

=====

Appendix A: New BDS Software License

BDS Software's former mail address at P.O. Box 485 in Glenview, IL is no longer valid. Any mail sent to that address will be rejected by the U.S. Postal Service. See my Contact page.

M.D.J. 2018/06/08

=====

Works Cited

[Google]
https://www.geeksforgeeks.org/applications-advantages-and-disadvantages-of-stack/ . 2024. Online.

=====