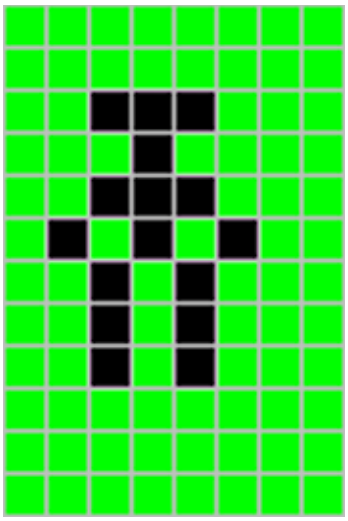
M. David Johnson
http://www.bds-soft.com
info@bds-soft.com



***** WARNING *****

This is working correctly on a CoCo 3
But is not yet working on a CoCo 2

Malky's Warren:
The First Training Quest

by M. David Johnson

2023/04/22

Abstract

The current design state of a simple PMODE 4 maze game is presented. It is planned to be a 64K CoCo 2 game, but is currently only working on a CoCo 3 .

Malky's Warren is a Work-In-Progress and is intended primarily as a Proof-Of-Concept for the ML Foundation System; along with that System's associated False Disk, Graphics Control, and Fake Text Routines.

——

This paper and its associated code are available online at:

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

=====

Table of Contents

MALKYS ASSEMBLY LANGUAGE ROUTINES:

=====

Introduction

You are an Explorer-In-Training.

_____

On Febreuary 23, 303 A.D., Emperor Diocletian of Rome issued an edict prohibiting Christians from assembling for worship and ordered the destruction of their scriptures, liturgical books, and places of worship across the empire. (Wikipedia).

Many Christians, more devoted to Jesus than to the Emperor, hid their scriptures and books in caves; or buried them; or otherwise concealed them rather than destroying them as the edict required.

In the middle of the 20th century, archaeological discoveries at Qumran in Israel, and in the Egyptian desert produced the Dead Sea Scrolls, the Nag Hammadi Library, and other collections of ancient Biblical manuscripts and literature.

_____

In the early years of the 21st century, China quietly began cornering the markets for rare-earth minerals and other rare commodities, and began buying up land and businesses around the world; most notably in the United States of America.

On August 25, 2055 A.D., the United States Congress proposed the 34th Amendment to the Constitution of the United States which read, “All sovereignty over the United States of America and its territories is hereby ceded to the People's Republic of China (PRC)”. The Amendment was ratified by the States on September 29, 2055 A.D.

On January 18, 2056 A.D., the United Nations General Assembly issued Resolution 2056-3, ceding sovereignty over the UN to the PRC; and by mid-2056, the entire world was firmly in China’s grip.

_____

On Febreuary 23, 2063 A.D., Emperor Di Jidu Zhe of China issued an edict prohibiting Christians from assembling for worship and ordered the destruction of their scriptures, liturgical books, and places of worship across the entire world.

Many Christians, more devoted to Jesus than to the Emperor, hid their scriptures and books in caves; or buried them; or otherwise concealed them rather than destroying them as the edict required.

On October 18, 2077 A.D., the world economy suddenly collapsed and civilization was thrown into literal and cultural darkness.

_____

On June 8, 2386 A.D. (June 9, 102 N.C. [New Calendar]), James Malky was digging out a tree stump on his farm (in what used to be Northwest Colorado) when he discovered a small network of subterranean caves and tunnels. Over the next few months, he explored what soon became known locally as Malky's Warren. In addition to various other artifacts, on November 23, 102 N.C., James came upon a bedraggled copy of the Gospel of John.

News of the discovery spread, slowly at first, but then with gathering momentum. By early 116 N.C., the search for additional Biblical documents and other artifacts had intensified worldwide; and Malky's Warren was obtained and refitted as a training center for new explorers.

_____

As a new Explorer-In-Training, your quest is to enter Malky's Warren, find that Gospel of John, and deliver it to the Warehouse at the Warren's exit. Along the way, you may also find some Provisions to sustain you in your quest.

======

This paper describes the 64K CoCo 2 software which implements Malky's Warren to run on top of the ML Foundation System with its associated False Disk Routines, Graphics Control Routines, and Fake Text Routines.

***** WARNING *** As of this writing (2023/04/20; two days before CoCoFest), the Warren is only working on a CoCo 3. It is not yet working on a CoCo 2.**

Malky's Warren is a Work-In-Progress and is intended primarily as a Proof-Of-Concept. As such, the Quest is quite simple and easy to traverse. Future Quests won't be that simple.

_____

A few General Guidelines:

1. The moment you exit the Warren, the game is over. There's no going back at that point. Be careful not to go East from the Warehouse (marked "W") accidentally.

2. Pressing the " X " Key freezes the game and there's no recovery from that either. (This is a Future Feature yet to be implemented; a fancy way of saying it's a Known Bug).

3. North is up on the screen. Press the “ Up-Arrow ” to go North. Press the “ Right-Arrow ” to go East. Press the “ Down-Arrow ” to go South. Press the “ Left-Arrow ” to go West.

4. Press the “ T ” Key to Take something and put it in your Bag. Press the “ L ” Key to take something out of your Bag and Leave it in the Current Cell (including the Warehouse Cell).

4. Press the “ B ” Key for a Bag Contents List. Press the “ I ” Key for a Warehouse Inventory List.

_____

A Note on Numbers: To keep everything simple to understand, and also neatly lined-up, I frequently refer to numbers as decimal bytes with three full digits, e.g. 004, 027, 229, etc. See Appendix A for conversions between the decimal and hexadecimal representations of bytes. The leading zeroes are NOT intended to indicate octal notation. Octal notation is not used anywhere in this paper.

In works of this complexity (at least for me) typos and other errors are bound to sneak in. Please let me know about any you discover so I can note and correct them.

M.D.J. 2023/04/22
info@bds-soft.com

=====

General Methodology

In this paper, the Assembly Language files are presented in plain text form. I didn't have enough time to assemble full code listings between getting the code to actually run (late on the 19th) and the start of the Fest on the 22nd. The text files are on the Development Disks and you can run them through EDTASM if you wish.

Full Assembly Code Listings will (hopefully) be provided at a later date, after dealing with the "It doesn't run on my CoCo 2 !?!" problem.

The programs and routines are presented in their general order of relationship to the system as a whole; not in order of memory location, nor in order of completion date.

The individual programs and routines are fairly well structured internally, but their positions within the memory space are a bit jumbled due to increasing deadline pressures. Hopefully, I'll be able to reorganize the system somewhat moving forward. Meanwhile, use of pejorative nicknames like "Spaghetti Dave" will only be met with a haughtily injured glance.

No Testing is recorded in this paper; nor was any significant testing performed other than the simple running of the game once it was completed. Play the game. Test it for yourself. I'll appreciate any comments or suggestions (as long as they don't cite pasta).

=====

FALSFILE: Reserves the first granule of a disk for Linear Sectors

The BASIC Language program listing:

```
1000 '*****
1010 '*
1020 '* FALSFILE.BAS
1030 '* MDJ 2023/04/19
1040 '*
1050 '* THIS PROGRAM IS BASED
1060 '* UPON THE "FALSINIX.BAS"
1070 '* PROGRAM IN THE FALSE
1070 '* DISK SYSTEM.
1080 '*
1090 '* THIS PROGRAM INITIALIZES
1100 '* A SEMI-FALSE DISK WITH A
1110 '* SINGLE GRANULE (#0) AS A
1120 '* "RESERVED.IMG" FILE.
1130 '*
1140 '* THE 9 SECTORS OF GRANULE 0
1150 '* ARE THEN USED AS LINEAR
1160 '* SECTORS UNDER THE FALSE
1170 '* DISK SYSTEM.
1180 '*
1190 '* THE REMAINING 67 GRANULES
1200 '* ARE AVAILABLE FOR NORMAL
1210 '* PROGRAMS AND FILES.
1220 '*
1210 '* UPON COMPLETION, THE
1220 '* DIRECTORY WILL LOOK
1230 '* LIKE THIS:
1240 '*
1250 '* RESERVED IMG  3 B 1
1260 '*
1270 '* AND WILL PROVIDE
1280 '* THIS RESULT:
1290 '*
1300 '* PRINT FREE(0) --> 67
1310 '*
1320 '*****
1330 '

1500 CLEAR &H1000
1510 '
```

```
1700 PRINT
1710 PRINT "    PUT THE DISK IN DRIVE 0"
1720 PRINT "  ***  ***  WARNING  ***  ***"
1730 PRINT "  *** DISK WILL BE ERASED ***"
1740 PRINT "    PRESS ANY KEY WHEN READY"
1750 A$ = INKEY$
1760 IF A$ = "" GOTO 1750
1770 PRINT
1780 PRINT "WORKING *";
1790 '

2000 'ERASE THE DISK
2010 X$ = "                                "
2020 Z$ = X$+X$+X$+X$
2030 FOR I = 1 TO 128
2040   MID$(Z$,I,1) = CHR$(0)
2050 NEXT I
2060 FOR T = 0 TO 34
2070   FOR S = 1 TO 18
2080     DSKO$ 0,T,S,Z$,Z$
2090   NEXT S
2100   PRINT "*";
2110 NEXT T
2120 '

2200 'GENERATE THE FALSE FAT
2210 F$ = Z$
2220 'SET GRANULE 0
2230 '==> USE ALL 9 SECTORS
2240 MID$(F$,1,1) = CHR$(&HC9)
2250 'SET GRANULES 1 TO 67 = FREE
2260 FOR I = 2 TO 68
2270   MID$(F$,I,1) = CHR$(&HFF)
2280 NEXT I
2290 'PUT TRACK 17, SECTOR 2 TO DISK
2300 DSKO$ 0,17,2,F$,Z$
2310 PRINT "*";
2320 '

2500 'GENERATE THE FALSE DIRECTORY
2510 D$ = Z$
2520 'SET FALSE FILENAME
2531 MID$(D$,1,1) = "R"
2532 MID$(D$,2,1) = "E"
2533 MID$(D$,3,1) = "S"
2534 MID$(D$,4,1) = "E"
```

```
2535 MID$(D$,5,1) = "R"
2536 MID$(D$,6,1) = "V"
2537 MID$(D$,7,1) = "E"
2538 MID$(D$,8,1) = "D"
2539 MID$(D$,9,1) = "I"
2540 MID$(D$,10,1) = "M"
2541 MID$(D$,11,1) = "G"
2600 'SET TYPE =
2610 '  TEXT EDITOR SOURCE
2620 MID$(D$,12,1) = CHR$(3)
2630 'SET FORMAT = BINARY
2640 MID$(D$,13,1) = CHR$(0)
2650 'SET NUMBER OF THE
2660 '  FIRST GRANULE
2670 MID$(D$,14,1) = CHR$(0)
2680 'NUMBER OF BYTES USED
2690 '  IN LAST SECTOR = 256
2700 MID$(D$,15,1) = CHR$(1)
2710 MID$(D$,16,1) = CHR$(0)
2720 'PUT TRACK 17, SECTOR 3 TO DISK
2730 DSKO$ 0,17,3,D$,Z$
2740 PRINT "*";
2750 '

2900 PRINT
2910 PRINT "FALSFILE = DONE"
2920 '

32767 END
```

=====

Sx000001: The Linear Sector Files

There are two 256K Linear Sector Files (SA000001 and SB000001) which were hand-coded to represent the 512 (32x16) character positions of the PMODE 4 Fake Text Screen's representation of Malky's Warren (i.e,. the maze and its reporting fields). This representation is significantly more efficient and economical than a 24-sector graphic representation would require.

There are an additional two 256K Linear Sector Files (SC000001 and SD000001) which are intended to serve as Details and Utilities Sectors for the maze. They are not used in Malky's Warren, but are reserved for future use. At the moment, they are simply dummies (copies of SA000001) and are not presented here.

The Assembly Language text listings:

SA000001:

```
*****
*
* SA000001.ASM
* MDJ 2023/04/07
*
* SCREEN MAKER
*  MAZE    01
*  LEVEL   00
*  SECTION 00
*
* UPPER HALF
*
*****

        ORG     $5500

* LINE 00
LINE00  FCB     32      00
        FCB     32      01
        FCB     32      02
        FCB     32      03
        FCB     32      04
        FCB     32      05
        FCB     32      06
        FCB     32      07
        FCB     32      08
        FCB     105     09
        FCB     98      10
        FCB     103     11
```

```
        FCB     107     12
        FCB     103     13
        FCB     107     14
        FCB     103     15
        FCB     107     16
        FCB     103     17
        FCB     107     18
        FCB     103     19
        FCB     107     20
        FCB     103     21
        FCB     107     22
        FCB     103     23
        FCB     107     24
        FCB     103     25
        FCB     107     26
        FCB     103     27
        FCB     107     28
        FCB     103     29
        FCB     99      30
        FCB     32      31

* LINE 01
        FCB     69      00
        FCB     78      01
        FCB     84      02
        FCB     69      03
        FCB     82      04
        FCB     252     05
        FCB     253     06
        FCB     32      07
        FCB     113     08
        FCB     32      09
        FCB     117     10
        FCB     32      11
        FCB     117     12
        FCB     32      13
        FCB     117     14
        FCB     32      15
        FCB     117     16
        FCB     32      17
        FCB     117     18
        FCB     32      19
        FCB     110     20
        FCB     32      21
        FCB     117     22
        FCB     32      23
        FCB     117     24
```

```
        FCB     32      25
        FCB     117     26
        FCB     32      27
        FCB     117     28
        FCB     32      29
        FCB     111     30
        FCB     32      31

* LINE 02
        FCB     98      00
        FCB     103     01
        FCB     107     02
        FCB     103     03
        FCB     107     04
        FCB     103     05
        FCB     107     06
        FCB     103     07
        FCB     107     08
        FCB     103     09
        FCB     116     10
        FCB     102     11
        FCB     116     12
        FCB     102     13
        FCB     116     14
        FCB     102     15
        FCB     116     16
        FCB     102     17
        FCB     116     18
        FCB     108     19
        FCB     116     20
        FCB     102     21
        FCB     116     22
        FCB     102     23
        FCB     116     24
        FCB     108     25
        FCB     116     26
        FCB     102     27
        FCB     116     28
        FCB     102     29
        FCB     115     30
        FCB     32      31

* LINE 03
        FCB     111     00
        FCB     32      01
        FCB     117     02
        FCB     32      03
```

```
        FCB     117     04
        FCB     32      05
        FCB     117     06
        FCB     32      07
        FCB     117     08
        FCB     32      09
        FCB     117     10
        FCB     32      11
        FCB     117     12
        FCB     32      13
        FCB     117     14
        FCB     32      15
        FCB     117     16
        FCB     32      17
        FCB     117     18
        FCB     32      19
        FCB     110     20
        FCB     32      21
        FCB     117     22
        FCB     32      23
        FCB     117     24
        FCB     32      25
        FCB     117     26
        FCB     32      27
        FCB     117     28
        FCB     32      29
        FCB     111     30
        FCB     32      31

* LINE 04
        FCB     114     00
        FCB     108     01
        FCB     116     02
        FCB     108     03
        FCB     116     04
        FCB     102     05
        FCB     116     06
        FCB     102     07
        FCB     116     08
        FCB     102     09
        FCB     116     10
        FCB     102     11
        FCB     116     12
        FCB     102     13
        FCB     116     14
        FCB     108     15
        FCB     116     16
```

```
        FCB     102     17
        FCB     116     18
        FCB     102     19
        FCB     116     20
        FCB     108     21
        FCB     116     22
        FCB     108     23
        FCB     116     24
        FCB     108     25
        FCB     116     26
        FCB     108     27
        FCB     116     28
        FCB     108     29
        FCB     115     30
        FCB     32      31

* LINE 05
        FCB     111     00
        FCB     32      01
        FCB     110     02
        FCB     32      03
        FCB     110     04
        FCB     32      05
        FCB     117     06
        FCB     32      07
        FCB     117     08
        FCB     32      09
        FCB     110     10
        FCB     32      11
        FCB     117     12
        FCB     32      13
        FCB     110     14
        FCB     32      15
        FCB     110     16
        FCB     32      17
        FCB     117     18
        FCB     32      19
        FCB     117     20
        FCB     32      21
        FCB     110     22
        FCB     32      23
        FCB     110     24
        FCB     32      25
        FCB     110     26
        FCB     32      27
        FCB     110     28
        FCB     32      29
```

```
        FCB     111     30
        FCB     32      31

* LINE 06
        FCB     114     00
        FCB     108     01
        FCB     116     02
        FCB     108     03
        FCB     116     04
        FCB     108     05
        FCB     116     06
        FCB     102     07
        FCB     116     08
        FCB     108     09
        FCB     116     10
        FCB     108     11
        FCB     116     12
        FCB     108     13
        FCB     116     14
        FCB     108     15
        FCB     116     16
        FCB     108     17
        FCB     116     18
        FCB     102     19
        FCB     116     20
        FCB     102     21
        FCB     116     22
        FCB     102     23
        FCB     116     24
        FCB     102     25
        FCB     116     26
        FCB     108     27
        FCB     116     28
        FCB     108     29
        FCB     115     30
        FCB     32      31

* LINE 07
        FCB     111     00
        FCB     32      01
        FCB     110     02
        FCB     32      03
        FCB     110     04
        FCB     32      05
        FCB     117     06
        FCB     32      07
        FCB     110     08
```

```
        FCB     32      09
        FCB     110     10
        FCB     32      11
        FCB     110     12
        FCB     32      13
        FCB     117     14
        FCB     32      15
        FCB     110     16
        FCB     32      17
        FCB     117     18
        FCB     32      19
        FCB     110     20
        FCB     32      21
        FCB     117     22
        FCB     32      23
        FCB     117     24
        FCB     32      25
        FCB     117     26
        FCB     32      27
        FCB     110     28
        FCB     32      29
        FCB     111     30
        FCB     32      31

        END

*****
*
* EOF
*
*****
```

SB000001:

```
*****
*
* SB000001.ASM
* MDJ 2023/04/08
*
* SCREEN MAKER
*  MAZE    01
*  LEVEL   00
*  SECTION 00
*
* LOWER HALF
*
*****
```

```
        ORG     $5600

* LINE 08
        FCB     114     00
        FCB     108     01
        FCB     116     02
        FCB     108     03
        FCB     116     04
        FCB     102     05
        FCB     116     06
        FCB     102     07
        FCB     116     08
        FCB     108     09
        FCB     116     10
        FCB     102     11
        FCB     116     12
        FCB     102     13
        FCB     116     14
        FCB     102     15
        FCB     116     16
        FCB     102     17
        FCB     116     18
        FCB     102     19
        FCB     116     20
        FCB     108     21
        FCB     116     22
        FCB     102     23
        FCB     116     24
        FCB     102     25
        FCB     116     26
        FCB     102     27
        FCB     116     28
        FCB     108     29
        FCB     115     30
        FCB     32      31

* LINE 09
        FCB     111     00
        FCB     32      01
        FCB     110     02
        FCB     32      03
        FCB     117     04
        FCB     32      05
        FCB     117     06
        FCB     32      07
        FCB     117     08
```

```
        FCB     32      09
        FCB     117     10
        FCB     32      11
        FCB     110     12
        FCB     32      13
        FCB     110     14
        FCB     32      15
        FCB     117     16
        FCB     32      17
        FCB     117     18
        FCB     32      19
        FCB     117     20
        FCB     32      21
        FCB     110     22
        FCB     32      23
        FCB     117     24
        FCB     32      25
        FCB     117     26
        FCB     32      27
        FCB     117     28
        FCB     32      29
        FCB     111     30
        FCB     32      31

* LINE 10
        FCB     114     00
        FCB     108     01
        FCB     116     02
        FCB     102     03
        FCB     116     04
        FCB     102     05
        FCB     116     06
        FCB     102     07
        FCB     116     08
        FCB     102     09
        FCB     116     10
        FCB     108     11
        FCB     116     12
        FCB     108     13
        FCB     116     14
        FCB     108     15
        FCB     116     16
        FCB     108     17
        FCB     116     18
        FCB     102     19
        FCB     116     20
        FCB     102     21
```

```
        FCB     116     22
        FCB     103     23
        FCB     106     24
        FCB     103     25
        FCB     106     26
        FCB     103     27
        FCB     106     28
        FCB     103     29
        FCB     101     30
        FCB     32      31

* LINE 11
        FCB     111     00
        FCB     32      01
        FCB     117     02
        FCB     32      03
        FCB     110     04
        FCB     32      05
        FCB     117     06
        FCB     32      07
        FCB     117     08
        FCB     32      09
        FCB     117     10
        FCB     32      11
        FCB     117     12
        FCB     32      13
        FCB     117     14
        FCB     32      15
        FCB     110     16
        FCB     32      17
        FCB     117     18
        FCB     32      19
        FCB     117     20
        FCB     127     21
        FCB     117     22
        FCB     32      23
        FCB     113     24
        FCB     32      25
        FCB     252     26
        FCB     253     27
        FCB     69      28
        FCB     88      29
        FCB     73      30
        FCB     84      31

* LINE 12
        FCB     100     00
```

```
        FCB     103     01
        FCB     106     02
        FCB     103     03
        FCB     106     04
        FCB     103     05
        FCB     106     06
        FCB     103     07
        FCB     106     08
        FCB     103     09
        FCB     106     10
        FCB     103     11
        FCB     106     12
        FCB     103     13
        FCB     106     14
        FCB     103     15
        FCB     106     16
        FCB     103     17
        FCB     106     18
        FCB     103     19
        FCB     106     20
        FCB     103     21
        FCB     101     22
        FCB     105     23
        FCB     32      24
        FCB     32      25
        FCB     32      26
        FCB     32      27
        FCB     32      28
        FCB     32      29
        FCB     32      30
        FCB     32      31

* LINE 13
        FCB     32      00
        FCB     32      01
        FCB     32      02
        FCB     32      03
        FCB     32      04
        FCB     32      05
        FCB     32      06
        FCB     32      07
        FCB     32      08
        FCB     32      09
        FCB     32      10
        FCB     32      11
        FCB     32      12
        FCB     32      13
```

```
        FCB     32      14
        FCB     32      15
        FCB     32      16
        FCB     32      17
        FCB     32      18
        FCB     32      19
        FCB     32      20
        FCB     32      21
        FCB     32      22
        FCB     32      23
        FCB     32      24
        FCB     32      25
        FCB     32      26
        FCB     32      27
        FCB     32      28
        FCB     32      29
        FCB     32      30
        FCB     32      31

* LINE 14
        FCB     32      00
        FCB     32      01
        FCB     32      02
        FCB     32      03
        FCB     32      04
        FCB     32      05
        FCB     32      06
        FCB     32      07
        FCB     32      08
        FCB     32      09
        FCB     32      10
        FCB     32      11
        FCB     32      12
        FCB     32      13
        FCB     32      14
        FCB     32      15
        FCB     32      16
        FCB     32      17
        FCB     32      18
        FCB     32      19
        FCB     32      20
        FCB     32      21
        FCB     32      22
        FCB     32      23
        FCB     32      24
        FCB     32      25
        FCB     32      26
```

```
        FCB     32      27
        FCB     32      28
        FCB     32      29
        FCB     32      30
        FCB     32      31

* LINE 15
        FCB     83      00
        FCB     84      01
        FCB     82      02
        FCB     69      03
        FCB     78      04
        FCB     71      05
        FCB     84      06
        FCB     72      07
        FCB     32      08
        FCB     61      09
        FCB     32      10
        FCB     48      11
        FCB     48      12
        FCB     48      13
        FCB     48      14
        FCB     48      15
        FCB     32      16
        FCB     32      17
        FCB     32      18
        FCB     83      19
        FCB     67      20
        FCB     79      21
        FCB     82      22
        FCB     69      23
        FCB     32      24
        FCB     61      25
        FCB     32      26
        FCB     48      27
        FCB     48      28
        FCB     48      29
        FCB     48      30
        FCB     48      31

        END

*****
*
* EOF
*
*****
```

SMMAKER: Installs the Linear Sector Files on the first granule

The Assembly Language text listing:

```
*****
*
* SMMAKER.ASM
* MDJ 2023/04/19
*
* SCREEN MAZE MAKER
* ASSEMBLY ROUTINE
*
* SAVES THE SA, SB, SC, SD
* FILE CONTENTS, I.E. THE
* SCREEN INFORMATION BUFFERS,
* TO THE "RESERVED.IMG"
* ON A FALSFILE DISK.
*
*****

FLPUT   EQU     $44F2   PUT BUFFER TO FALSE DISK
LINE00  EQU     $5500   START OF SA FILE
LINE08  EQU     $5600   START OF SB FILE
SCDTLS  EQU     $5700   START OF SB FILE
SCUTLS  EQU     $5800   START OF SB FILE

        ORG     $536F

SMMAKE  PSHS    X,Y

* PUT SA FILE CONTENTS TO
* FALSE DISK SECTOR #0
        LDX  #0
        LDY  #LINE00
        JSR  FLPUT

* PUT SB FILE CONTENTS TO
* FALSE DISK SECTOR #1
        LDX  #1
        LDY  #LINE08
        JSR  FLPUT

* PUT SC FILE CONTENTS TO
* FALSE DISK SECTOR #2
```

```
        LDX  #2
        LDY  #SCDTLS
        JSR  FLPUT

* PUT SB FILE CONTENTS TO
* FALSE DISK SECTOR #3
        LDX  #3
        LDY  #SCUTLS
        JSR  FLPUT

        PULS    X,Y

ENDCHK  RTS

        END
```

The BASIC Control Program listing:

```
1000 '*****
1010 '*
1020 '* SMMAKER.BAS
1030 '* MDJ 2023/04/19
1040 '*
1050 '* SCREEN MAZE MAKER
1060 '* BASIC PROGRAM
1070 '*
1080 '* SAVES THE SA, SB, SC, SD
1090 '* FILE CONTENTS TO
1100 '* THE "RESERVED.IMG"
1110 '* ON A FALSFILE DISK.
1120 '*
1130 '*****
1140 '

2000 'SETUP MEMORY
2010 CLEAR 200, &H4000
2020 '

4000 LOADM "MLBASE.BIN"
4010 LOADM "SMMAKER.BIN"
4020 LOADM "SA000001.BIN"
4030 LOADM "SB000001.BIN"
4040 LOADM "SC000001.BIN"
4050 LOADM "SD000001.BIN"
4060 '

5000 PRINT "PLACE FALSFILE DISK IN DRIVE 0"
```

```
5010 PRINT "PRESS ANY KEY WHEN READY >";
5020 A$ = INKEY$
5030 IF A$="" GOTO 5020

6000 EXEC &H536F
6010 '

32767 END
```

=====

SMREADER: Reads the Linear Sectors and displays their bytes for checking

The Assembly Language text listing:

```
*****
*
* SMREADER.ASM
* MDJ 2023/04/10
*
* SCREEN MAZE READER
* ASSEMBLY ROUTINE
*
* GETS THE SCREEN
* INFORMATION BUFFERS
* FROM A FALSE DISK
*
*****

FLGET   EQU     $4533   GET BUFFER FROM FALSE DISK
LINE00  EQU     $5500   START OF LINE00 BUFFER
LINE08  EQU     $5600   START OF LINE08 BUFFER
SCDTLS  EQU     $5700   START OF SCDTLS BUFFER
SCUTLS  EQU     $5800   START OF SCUTLS BUFFER

        ORG     $539C

SMREAD  PSHS    X,Y

* GET LINE00 BUFFER CONTENTS
* FROM FALSE DISK SECTOR #0
        LDX  #0
        LDY  #LINE00
        JSR  FLGET

* GET LINE08 BUFFER CONTENTS
* FROM FALSE DISK SECTOR #1
        LDX  #1
        LDY  #LINE08
        JSR  FLGET

* GET SCDTLS BUFFER CONTENTS
* FROM FALSE DISK SECTOR #2
        LDX  #2
        LDY  #SCDTLS
```

```
        JSR  FLGET

* GET SCUTLS BUFFER CONTENTS
* FROM FALSE DISK SECTOR #3
        LDX  #3
        LDY  #SCUTLS
        JSR  FLGET

        PULS    X,Y

ENDCHK  RTS

        END
```

The BASIC Control Program listing:

```
1000 '*****
1010 '*
1020 '* SMREADER.BAS
1030 '* MDJ 2023/04/19
1040 '*
1050 '* SCREEN MAZE READER
1060 '* BASIC PROGRAM
1070 '*
1080 '* GETS THE CONTENTS OF
1090 '* A FALSE DISK'S
1100 '* SECTORS #0 - #3
1110 '* AND PLACES THE DATA
1120 '* IN FOUR WORKING BUFFERS.
1130 '*
1140 '* IT THEN STEPS THROUGH
1150 '* THE BUFFERS TO ALLOW
1160 '* CHECKING OF THE DATA.
1170 '*
1180 '*****
1190 '

2000 'SETUP MEMORY
2010 CLEAR 200, &H4000
2020 '

4000 LOADM "MLBASE.BIN"
4010 LOADM "SMREADER.BIN"
4040 '

5000 PRINT "PLACE FALSFILE DISK IN DRIVE 0"a
5010 PRINT "PRESS ANY KEY WHEN READY >";
```

```
5020 A$ = INKEY$
5030 IF A$="" GOTO 5020

5200 EXEC &H539C
5210 '

5500 FOR Y = 0 TO 7
5510   FOR X = 0 TO 31
5520     Z = (Y * 32) + X
5530     Z1 = Z + &H5500
5540     C = PEEK(Z1)
5550     PRINT C;
5560   NEXT X
5570 NEXT Y

6000 A$ = INKEY$
6010 IF A$="" GOTO 6000

6500 FOR Y = 8 TO 15
6510   FOR X = 0 TO 31
6520     Z = (Y * 32) + X
6530     Z1 = Z + &H5500
6540     C = PEEK(Z1)
6550     PRINT C;
6560   NEXT X
6570 NEXT Y

7000 A$ = INKEY$
7010 IF A$="" GOTO 7000

7500 FOR Y = 16 TO 23
7510   FOR X = 0 TO 31
7520     Z = (Y * 32) + X
7530     Z1 = Z + &H5500
7540     C = PEEK(Z1)
7550     PRINT C;
7560   NEXT X
7570 NEXT Y

8000 A$ = INKEY$
8010 IF A$="" GOTO 8000

8500 FOR Y = 24 TO 31
8510   FOR X = 0 TO 31
8520     Z = (Y * 32) + X
8530     Z1 = Z + &H5500
8540     C = PEEK(Z1)
```

```
     8550     PRINT C;
     8560   NEXT X
     8570 NEXT Y

     32767 END
```

=====

SMDISPLY: Reads the Linear Sectors and displays the PMODE 4 Screen they represent

The Assembly Language text listing:

```
*****
*
* SMDISPLY.ASM
* MDJ 2023/04/10
*
* SCREEN MAZE DISPLAY
* ASSEMBLY ROUTINE
*
* DISPLAYS THE MAZE
* ON SCREEN, USING THE
* FAKETEXT 32 X 16
* CHARACTER SET FOR
* PMODE 4
*
* ** START NOTE TO MDJ **
* THIS ROUTINE IS SOMEWHAT
* INEFFICIENT, BUT IT'S
* EASY TO UNDERSTAND AND
* IT'S FAST ENOUGH TO BE
* ACCEPTABLE. IT ALSO USES
* $E400-$E7FF IN HIGH MEMORY.
*    ** BOO! HISS! **
*
* FOR FUTURE WORK: REPLACE
* THIS WITH A MORE EFFICIENT
* SMDRAW ROUTINE - SEE THE
* "FUTUREWORK" FOLDER.
* ** END NOTE TO MDJ **
*
*****

PTFCHR  EQU     $5300   FAKE TEXT ROUTINE
LINE00  EQU     $5500   START OF BUFFERS

        ORG     $53C9

SMDISP  JMP     LBL001
```

```
XCOORD  RMB     1
YCOORD  RMB     1
CHRCOD  RMB     2

LBL001  PSHS    A,B,X,Y,U
        LDY     #LINE00
        LDU     #$E400  TEMP CHRCOD STORE

LBL002  LDB     ,Y+     GET THE CHRCOD
        CLRA            EXTEND IT TO 16-BITS
        STD     ,U++    SAVE CHRCOD TO STORE

        CMPY    #$5700  ARE WE DONE?
        BLO     LBL002  GO IF NO

CHROUT  LDU     #$E400  RESET CHRCOD STORE PTR
        LDA     #$FF    SET FIRST XCOORD TO ROLL
        LDB     #$00    SET FIRST YCOORD TO ZERO

LBL003  INCA            INCREMENT XCOORD
        STA     XCOORD
        STB     YCOORD
        CMPA    #32     END OF THE X LINE?
        BLO     LBL004  GO IF NO
        CLRA            SET XCOORD = 0
        STA     XCOORD
        INCB            INCREMENT YCOORD
        STB     YCOORD
        CMPB    #16     END OF SCREEN?
        BLO     LBL004  GO IF NO
        BRA     LBL005  GO IF YES

LBL004  LDX     ,U++    GET CHRCOD FROM STORE
        STX     CHRCOD

        PSHS    A,B,X   PUT CHRCOD TO SCREEN
        LDA     XCOORD
        LDB     YCOORD
        LDX     CHRCOD
        JSR     PTFCHR
        PULS    A,B,X
        BRA     LBL003  RETURN FOR NEXT CHRCOD

LBL005  PULS    A,B,X,Y,U

ENDCHK  RTS
```

```
END
```

The BASIC Control Program listing:

```
1000 '*****
1010 '*
1020 '* SMDISPLY.BAS
1030 '* MDJ 2023/04/19
1040 '*
1050 '* SCREEN MAZE DISPLAY
1060 '* BASIC PROGRAM
1070 '*
1080 '* DISPLAYS THE MAZE
1090 '* ON SCREEN, USING THE
1100 '* FAKETEXT 32 X 16
1110 '* CHARACTER SET FOR
1120 '* PMODE 4
1130 '*
1140 '*****
1150 '

2000 'SETUP MEMORY
2010 CLEAR 200, &H4000
2020 PCLEAR 4
2030 '

4000 LOADM "MLBASE.BIN"
4010 LOADM "SMREADER.BIN"
4020 LOADM "SMDISPLY.BIN"
4050 '

5000 PRINT "PLACE FALSFILE DISK IN DRIVE 0"
5010 PRINT "PRESS ANY KEY WHEN READY >";
5020 A$ = INKEY$
5030 IF A$="" GOTO 5020
5040 '

7000 EXEC &H539C  'SMREADER
7010 '

9500 'SETUP GRAPHICS
9510 PMODE 4,1
9520 PCLS 1
9530 SCREEN 1,0
9540 '
```

```
9610 EXEC &H53C9  'SMDISPLY
9620 '

9700 'HOLD THE SCREEN
9710 GOTO 9710
9720 '

32767 END
```

=====

MKMLBASE: Combines MLCORE.BIN, FLSYS.BIN, MLGC.BIN, and C3216SET.BIN

This combines the ML Foundation, False Disk, Graphics Control, and Fake Text files all into one combined file for easier use during development. The BASIC Language program listing:

```
1000 '*****
1010 '*
1020 '* MKMLBASE.BAS
1030 '* MDJ 2023/04/07
1040 '*
1050 '* MAKES THE
1060 '* MLBASE.BIN FILE
1070 '* BY COMBINING:
1080 '*   MLCORE.BIN
1090 '*   FLSYS.BIN
1100 '*   MLGC.BIN
1110 '*   C3216SET.BIN
1120 '*
1130 '* SO THAT THE SYSTEM
1140 '* CAN BE LOADED AS A
1150 '* SINGLE ENTITY.
1160 '*
1170 '*****
1180 '

1500 CLEAR 200, &H4000
1510 '

2000 LOADM "MLCORE.BIN"
2010 LOADM "FLSYS.BIN"
2020 LOADM "MLGC.BIN"
2030 LOADM "C3216SET.BIN"
2040 '

3000 SAVEM "MLBASE.BIN", &H4000, &H536E, &H4000
3010 '

32767 END
```

=====

MAKEMLKY: Makes the Production File MALKYS.BIN

This is not actually used until all the Assembly Language files have been completed and assembled. It is placed at this position in the document because, like MKMLBASE, it performs an administrative task rather than a game task. The BASIC Language program listing:

```
1000 '*****
1010 '*
1020 '* MAKEMLKY.BAS
1030 '* MDJ 2023/04/19
1040 '*
1050 '* MAKES THE PRODUCTION
1060 '* FILE MALKYS.BIN
1070 '* FROM THE COLLECTION
1080 '* OF INDIVIDUAL
1090 '* SUBROUTINE AND
1100 '* ASSOCIATED FILES.
1110 '*
1120 '*****
1130 '

2000 'SETUP MEMORY
2010 CLEAR 200, &H4000
2020 PCLEAR 4
2030 '

4000 LOADM "MLBASE.BIN"
4010 LOADM "SMREAD.BIN"
4020 LOADM "MCSCCVT.BIN"
4030 LOADM "STAVTR.BIN"
4040 LOADM "PTFCHA.BIN"
4050 LOADM "PTFBYA.BIN"
4060 LOADM "PTFWRA.BIN"
4070 LOADM "GTFLOC.BIN"
4080 LOADM "GTFVAL.BIN"
4090 LOADM "PTFVAL.BIN"
4100 LOADM "SYSVAR.BIN"
4110 LOADM "EASTK.BIN"
4120 LOADM "WESTK.BIN"
4130 LOADM "NORTHK.BIN"
4140 LOADM "SOUTHK.BIN"
4150 LOADM "TCHARK.BIN"
4160 LOADM "LCHARK.BIN"
4170 LOADM "BCHARK.BIN"
```

```
4180 LOADM "ICHARK.BIN"
4190 LOADM "UCHARK.BIN"
4200 LOADM "DCHARK.BIN"
4210 LOADM "PCHARK.BIN"
4220 LOADM "RCHARK.BIN"
4230 LOADM "XCHARK.BIN"
4240 LOADM "CCHARK.BIN"
4250 LOADM "GCHARK.BIN"
4260 LOADM "GMLOOP.BIN"
4270 LOADM "SMGAME.BIN"
4290 LOADM "DECMAL.BIN"
4300 LOADM "RPTSTR.BIN"
4310 LOADM "RPTSCO.BIN"
4320 LOADM "CLRL13.BIN"
4330 LOADM "CLRL14.BIN"
4340 LOADM "CLRSTR.BIN"
4350 LOADM "CLRSCO.BIN"
4360 LOADM "MSG001.BIN"
4370 LOADM "MSG002.BIN"
4380 LOADM "MSG003.BIN"
4390 LOADM "GMOVER.BIN"
4400 LOADM "GMOVED.BIN"
4410 LOADM "RANDOM.BIN"
4420 LOADM "MSG004.BIN:1"
4430 LOADM "MSG005.BIN:1"
4440 LOADM "MSG006.BIN:1"
4450 LOADM "MSG007.BIN:1"
4460 LOADM "MSG008.BIN:1"
4470 LOADM "MSG009.BIN:1"
4480 LOADM "PROCHK.BIN:1"
4490 '

5000 SAVEM "MALKYS.BIN:2", &H4000, &H7FE2, &H4000
5010 '

32767 END
```

=====

SYSVAR: System Variables

The Assembly Language text listing:

```
*****
*
* SYSVAR.ASM
* MDJ 2023/04/15
*
* SYSTEM VARIABLES
*
*****

        ORG     $549C

STRNTH  RMB     2       CURRENT STRENGTH
SCORE   RMB     2       CURRENT SCORE

* GMOK: 1 = RUNNING; 0 = OVER
GMOK    RMB     1       GAME OVER FLAG

* BAG: &20 = EMPTY; $E0 = GOSPEL OF JOHN
BAG     RMB     1       BAG CONTENTS

* WHSE: &20 = EMPTY; $E0 = GOSPEL OF JOHN
WHSE    RMB     1       WAREHOUSE CONTENTS

* DOCVAL: VALUE OF GOSPEL OF JOHN (21 CHAPTERS)
* BASED ON RANDOMLY SELECTED DOCUMENT CONDITION:
*        MINT = 210
*   EXCELLENT = 189
*   VERY GOOD = 168
*        GOOD = 147
*        FAIR = 126
*        POOR = 105
DOCVAL  RMB     1       SCORE VALUE OF DOCUMENT

* PROVAL: NUMBER OF POINTS ADDED TO STRENGTH
* RANDOMLY SELECTED BETWEEN 25 AND 75
PROVAL  RMB     1       PROVISIONS STRENGTH

ENDCHK  NOP

        END
```

=====

SIBUFF: Screen Information Buffers

The first four buffers are loaded from disk, as described in the Sx000001 Chapter. The fifth buffer, the General Utilities Buffer, is uninitialized and is intended to be used as a Screen Information scratchpad. This is for future use: this buffer remains unused in Malky's Warren. The Assembly Language text listing:

```
*****
*
* SIBUFF.ASM
* MDJ 2023/04/14
*
* SCREEN INFORMATION
* BUFFERS
*
*****

        ORG     $5500

* MAZE DESIGN
* UPPER HALF
* $5500 - $55FF
LINE00  RMB     256

* MAZE DESIGN
* LOWER HALF
* $5600 - $56FF
LINE08  RMB     256

* MAZE DESIGN
* SCREEN DETAILS
* $5700 - $57FF
SCDTLS  RMB     256

* MAZE DESIGN
* SCREEN UTILITIES
* $5800 - $58FF
SCUTLS  RMB     256

* GENERAL UTILITY
* BUFFER; UNINITIALIZED
* $5900 - $59FF
GENUTL  RMB     256

        END
=====
```

SMREAD: Gets the Screen Information Buffers from a False Disk

KNOWN BUG: Actually, not from a full False Disk, but rather from the RESERVED.IMG False File granule on the game disk itself, as described in the FALSFILE Chapter. This is just a missed revision of terminology - it doesn't effect gameplay at all.

The Assembly Language text listing:

```
*****
*
* SMREAD.ASM
* MDJ 2023/04/14
*
* SCREEN MAZE READER
* ASSEMBLY ROUTINE
*
* GETS THE SCREEN
* INFORMATION BUFFERS
* FROM A FALSE DISK
*
*****

FLGET   EQU     $4533   GET BUFFER FROM FALSE DISK
LINE00  EQU     $5500   START OF LINE00 BUFFER
LINE08  EQU     $5600   START OF LINE08 BUFFER
SCDTLS  EQU     $5700   START OF SCDTLS BUFFER
SCUTLS  EQU     $5800   START OF SCUTLS BUFFER

        ORG     $536F

SMREAD  PSHS    X,Y

* GET LINE00 BUFFER CONTENTS
* FROM FALSE DISK SECTOR #0
        LDX  #0
        LDY  #LINE00
        JSR  FLGET

* GET LINE08 BUFFER CONTENTS
* FROM FALSE DISK SECTOR #1
        LDX  #1
        LDY  #LINE08
        JSR  FLGET
```

```
      * GET SCDTLS BUFFER CONTENTS
      * FROM FALSE DISK SECTOR #2
              LDX  #2
              LDY  #SCDTLS
              JSR  FLGET

      * GET SCUTLS BUFFER CONTENTS
      * FROM FALSE DISK SECTOR #3
              LDX  #3
              LDY  #SCUTLS
              JSR  FLGET

              PULS    X,Y

      ENDCHK  RTS

              END
```

=====

STAVTR: Sets up the Avatar in its Starting Position

The Assembly Language text listing:

```
*****
*
* STAVTR.ASM
* MDJ 2023/04/14
*
* SETUP AVATAR IN
* STARTING POSITION
*
* STARTING MAZE
* COORDINATES
*   MX = 4
*   MY = 0
*
* ENTRY CONDITIONS:
* NONE
*
* EXIT CONDITIONS:
* NONE
*
*****

* PUT FAKE TEXT ROUTINE
PTFCHR  EQU     $5300

* COORDINATES CONVERTER
MCSCCV  EQU     $539C

        ORG     $53A1

STAVTR  PSHS    A,B,X
        JMP     LBL001

* CURRENT MAZE COORDINATES
MXC     RMB     1
MYC     RMB     1

* CURRENT CELL CONTENTS
* (UNDER THE AVATAR)
CELLCC  RMB     1
```

```
      * NEW MAZE COORDINATES
      * (DUMMIES ON STARTUP)
      MXN     RMB     1
      MYN     RMB     1

      * SET COORDINATES AND
      * AND CONTENTS
      LBL001  LDA     #32
              STA     CELLCC

              LDA     #4
              STA     MXC
              STA     MXN

              CLRB
              STB     MYC
              STB     MYN

      * CONVERT CURRENT
      * COORDINATES
              JSR     MCSCCV

      * PLACE AVATAR ON
      * THE SCREEN, USING ITS
      * CHARACTER CODE EXTENDED
              LDX     #$0000
              JSR     PTFCHR

              PULS    A,B,X
      ENDCHK  RTS

              END

=====
```

MCSCCVT: Maze Coordinates to Screen Coordinates Converter

The Assembly Language text listing:

```
*****
*
* MCSCCVT.ASM
* MDJ 2023/04/14
*
* MAZE COORDINATES TO
* SCREEN COORDINATES
* CONVERTER
*
* THE MAZE IS A 15 X 6
* CELL STRUCTURE,
* SUPERIMPOSED UPON
* THE 32 X 16 SCREEN
*
* FOR CHECKING DOOR
* OPENINGS, OBSTRUCTIONS,
* ETC., THE MAZE COORDS
* NEED TO BE CONVERTED
* TO SCREEN COORDINATES.
*   SX = (MX * 2) + 1
*   SY = (MY * 2) + 1
*
* ENTRY CONDITIONS:
* (MAZE COORDINATES)
* A = MX-COORDINATE
* B = MY-COORDINATE
*
* EXIT CONDITIONS:
* (SCREEN COORDINATES)
* A = SX-COORDINATE
* B = SY-COORDINATE
*
*****

        ORG     $539C

MCSCCV  LSLA            MX * 2
        INCA            + 1

        LSLB            MY * 2
```

```
            INCB            + 1

     ENDCHK  RTS

            END
```

=====

PTFCHA: Put a Fake Text Character to the Screen and Advance the Cursor

The Assembly Language text listing:

```
*****
*
* PTFCHA.ASM
* MDJ 2023/04/14
*
* PUT A FAKE TEXT
* CHARACTER TO THE
* PMODE 4 SCREEN AND
* ADVANCE THE POSITION
*
* ENTRY CONDITIONS:
* A = X-COORDINATE (0-31)
* B = Y-COORDINATE (0-15)
* X = CHARACTER CODE (0-255)
*     EXTENDED TO 16-BITS
*     I.E. ($0000 - $00FF)
*
* EXIT CONDITIONS:
* A = NEW X-COORDINATE
* B = SAME Y-COORDINATE
*
*****

PTFCHR  EQU     $5300

        ORG     $53CB

PTFCHA  PSHS    A,B     SAVE COORDS TO STACK

        JSR     PTFCHR  PUT CHAR TO SCREEN

        PULS    A,B     RETRIEVE COORDS
        INCA            POINT TO NEXT POS

ENDCHK  RTS

        END
```

=====

PTFBYA: Put an 8-bit Hexadecimal Number to the Screen and Advance the Cursor

The Assembly Language text listing:

```
*****
*
* PTFBYA.ASM
* MDJ 2023/04/14
*
* PUTS AN 8-BIT NUMBER
* TO THE PMODE 4 SCREEN
* AS TWO HEXADECIMAL DIGITS
* AND ADVANCES THE POSITION
*
* ENTRY CONDITIONS:
* A = X-COORDINATE (0-31)
* B = Y-COORDINATE (0-15)
* X = 8-BIT NUMBER EXTENDED
*     TO 16-BITS (0-255)
*
* EXIT CONDITIONS:
* A = NEW X-COORDINATE
* B = SAME Y-COORDINATE
*
*****

* SCRATCHPAD VARIABLES
* THE 8-BIT NUMBER
L0076   EQU     $0076

* THE HIGH NIBBLE
L0077   EQU     $0077

* THE LOW NIBBLE
L00F3   EQU     $00F3

* EXTERNAL ROUTINE
* ADDRESS
PTFCHA  EQU     $53CB

        ORG     $53D4

PTFBYA  BRA     LBL001
```

```
XTEMP   RMB     1
YTEMP   RMB     1

LBL001  STA     XTEMP   SAVE THE COORDINATES
        STB     YTEMP
        TFR     X,D     MOVE THE NUMBER TO A
        TFR     B,A

* SAVE THE NUMBER
        STA     L0076

* DIVIDE BY 16
        LSRA
        LSRA
        LSRA
        LSRA

* SAVE THE HIGH NIBBLE
        STA L0077

* MULTIPLY BY 16
        LSLA
        LSLA
        LSLA
        LSLA

* SAVE TEMP RESULT
        STA     L00F3

* GET THE NUMBER AGAIN
        LDA     L0076

* SUBTRACT TEMP RESULT
        SUBA    L00F3

* SAVE LOW NIBBLE
        STA     L00F3

* IS LOW NIBBLE <= 9
        CMPA    #9

* GO IF NO
        BHI     LBL002

* ADD ZERO OFFSET
        ADDA    #48
        BRA     LBL003
```

```
* ADD "A" OFFSET
LBL002  ADDA    #55     (65-10)

* SAVE LOW NIBBLE CHAR
LBL003  STA     L00F3

* GET HIGH NIBBLE
        LDA     L0077

* IS HIGH NIBBLE <= 9
        CMPA    #9

* GO IF NO
        BHI     LBL004

* ADD ZERO OFFSET
        ADDA    #48
        BRA     LBL005

* ADD "A" OFFSET
LBL004  ADDA    #55     (65-10)

LBL005  TFR     A,B     EXTEND CHAR TO X
        CLRA
        TFR     D,X
        LDA     XTEMP   RETRIEVE THE COORDS
        LDB     YTEMP

* PUT HIGH NIBBLE CHAR
* TO THE PMODE 4 SCREEN
        JSR     PTFCHA  (ALSO ADVANCES THE POS)

        STA     XTEMP   SAVE THE COORDINATES
        STB     YTEMP

* GET LOW NIBBLE CHAR
        LDA     L00F3

        TFR     A,B     EXTEND CHAR TO X
        CLRA
        TFR     D,X
        LDA     XTEMP   RETRIEVE THE COORDS
        LDB     YTEMP

* PUT LOW NIBBLE CHAR
* TO THE PMODE 4 SCREEN
```

```
            JSR     PTFCHA  (ALSO ADVANCES THE POS)
     ENDCHK RTS
            END
=====
```

PTFWRA: Put a 16-bit Hexadecimal Number to the Screen and Advance the Cursor

The Assembly Language text listing:

```
*****
*
* PTFWRA.ASM
* MDJ 2023/04/14
*
* PUTS A 16-BIT NUMBER
* TO THE PMODE 4 SCREEN
* AS FOUR HEXADECIMAL DIGITS
* AND ADVANCES THE POSITION
*
* ENTRY CONDITIONS:
* A = X-COORDINATE (0-31)
* B = Y-COORDINATE (0-15)
* X = 16-BIT NUMBER EXTENDED
*
* EXIT CONDITIONS:
* A = NEW X-COORDINATE
* B = SAME Y-COORDINATE
*
*****

* EXTERNAL ROUTINE
* ADDRESS
PTFBYA  EQU     $53D4

        ORG     $5433

PTFWRA  BRA     LBL001

XTEMP   RMB     1
YTEMP   RMB     1
NTEMP   RMB     2

LBL001  STA     XTEMP   SAVE THE COORDINATES
        STB     YTEMP
        STX     NTEMP   SAVE THE NUMBER

        LDD     NTEMP   RETRIEVE THE NUMBER
```

```
            TFR     A,B     GET THE HIGH BYTE TO X
            CLRA
            TFR     D,X
            LDA     XTEMP   RETRIEVE THE COORDINATES
            LDB     YTEMP

      * PRINT THE HIGH BYTE
            JSR     PTFBYA

      * SAVE THE NEW X-COORDINATE
            STA     XTEMP

            LDD     NTEMP   RETRIEVE THE NUMBER
            CLRA            GET THE LOW BYTE TO X
            TFR     D,X
            LDA     XTEMP   RETRIEVE THE COORDINATES
            LDB     YTEMP

      * PRINT THE LOW BYTE
            JSR     PTFBYA

      ENDCHK  RTS

            END
```

=====

GTFLOC: Get the Memory Location of the Current Screen (X,Y) Coordinates

The Assembly Language text listing:

```
*****
*
* GTFLOC.ASM
* MDJ 2023/04/14
*
* RETURNS THE ADDRESS
* OF THE MEMORY LOCATION
* OF THE PMODE 4 SCREEN'S
* (X,Y) COORDINATES
*
* ENTRY CONDITIONS:
* A = SX-COORDINATE
* B = SY-COORDINATE
*
* EXIT CONDITIONS:
* X = LOCATION ADDRESS
*
*****

LINE00  EQU     $5500   START OF BUFFERS

        ORG     $5466

GTFLOC  JMP     LBL001

* TEMPORARY 16-BIT
* EXTENSION OF SX-COORD
XTEMP1  FCB     $00     HIGH BYTE
XTEMP2  RMB     1       LOW BYTE

LBL001  STA     XTEMP2  EXTEND THE SX-COORD
        CLRA            EXTEND THE SY-COORD

* MULTIPLY THE EXTENDED SY-COORDINATE BY 32
* USING THE LSL EQUIVALENT LSLA; ROLB
* FIVE TIMES
        LSLB            *2
        ROLA
        LSLB            *4
        ROLA
```

```
            LSLB            *8
            ROLA
            LSLB            *16
            ROLA
            LSLB            *32
            ROLA

            ADDD    XTEMP1  ADD EXTENDED SX-COORD
            ADDD    #LINE00 ADD BUFFERS START ADDRESS
            TFR     D,X     MOVE ADDRESS TO REG X

    ENDCHK  RTS

            END
```

=====

GTFVAL: Get a Fake Text Character's Value from the Screen Information Buffers

The Assembly Language text listing:

```
*****
*
* GTFVAL.ASM
* MDJ 2023/04/14
*
* GET A FAKE TEXT
* CHARACTER'S VALUE
* FROM THE SCREEN
* INFORMATION BUFFERS
*
* ENTRY CONDITIONS:
* A = SX-COORDINATE
* B = SY-COORDINATE
*
* EXIT CONDITIONS:
* B = CHARACTER CODE
*
*****

* EXTERNAL ROUTINE TO
* CONVERT THE (SX,SY)
* COORDINATES TO THE
* PMODE 4 SCREEN ADDRESS
GTFLOC  EQU     $5466

        ORG     $5482

GTFVAL  PSHS    X

        JSR     GTFLOC  X = LOCATION
        LDB     ,X      B = VALUE

        PULS    X
ENDCHK  RTS

        END
```

=====

PTFVAL: Put a Fake Text Character's Value to the Screen Information Buffers

KNOWN BUG: This file erroneously uses "GTFVAL: instead of "PTFVAL as its entry label. This doesn't affect anything: it just looks bad.

The Assembly Language text listing:

```
*****
*
* PTFVAL.ASM
* MDJ 2023/04/14
*
* PUT A FAKE TEXT
* CHARACTER'S VALUE
* TO THE SCREEN
* INFORMATION BUFFERS
*
* ENTRY CONDITIONS:
* A = SX-COORDINATE
* B = SY-COORDINATE
* X = CHARACTER CODE
*     EXTENDED TO 16-BITS
*
* EXIT CONDITIONS:
* NONE
*
*****

* EXTERNAL ROUTINE TO
* CONVERT THE (SX,SY)
* COORDINATES TO THE
* PMODE 4 SCREEN ADDRESS
GTFLOC  EQU     $5466

        ORG     $548C

GTFVAL  BRA     LBL001

CTEMP1  RMB     1
CTEMP2  RMB     1

LBL001  STX     CTEMP1 TRUNCATE CHAR CODE
        JSR     GTFLOC
        LDB     CTEMP2 GET TRUNCATED CODE
```

```
            STB     ,X

ENDCHK      RTS

            END
```

=====

RANDOM: Returns a Random Number between 0 and R, where R = 1 to R = 65534

The Assembly Language text listing:

```
*****
*
* RANDOM.ASM
* MDJ 2023/04/18
*
* RETURNS A RANDOM
* NUMBER BETWEEN
* 0 AND R INCLUSIVE
* WHERE R IS BETWEEN
* 1 AND 65534 ($FFFE)
*
* WITH R IN REG X, AND
* THE RANDOM NUMBER RETURNED
* BY THE ML FOUNDATION'S
* RNDU16 IN REGY, THE HIGH
* BYTE (REG X) OF MU1616'S
* 32-BIT RESULT IS THE
* DESIRED RANDOM NUMBER HERE.
*
* NO CHECKING: THE
* USER IS RESPONSIBLE
* FOR MAKING SURE THAT
* R IS WITHIN RANGE
*
* ENTRY CONDITIONS:
* D = THE R VALUE
*
* EXIT CONDITIONS:
* D = THE RANDOM NUMBER
*
*****

MU1616  EQU     $429D   16X16 MULTIPLY
RNDU16  EQU     $43E2   MLF'S RNG

        ORG     $7110

RANDOM  PSHS    X,Y
```

```
            ADDD    #1      INCREASE R BY 1
            TFR     D,X     MOVE R TO REG X
            JSR     RNDU16  GET MLF RANDOM #
            TFR     D,Y     MOVE RANDOM # TO Y
            JSR     MU1616  GO MULTIPLY
            TFR     X,D     MOVE RESULT TO D

            PULS    X,Y
    ENDCHK  RTS

            END
```

=====

PROCHK: Provisions Check and Assimilation Subroutine

The Assembly Language text listing:

```
*****
*
* PROCHK.ASM
* MDJ 2023/04/18
*
* PROVISIONS CHECK
* AND ASSIMILATION
* SUBROUTINE.
*
* THIS SUBROUTINE IS CALLED BY:
*   EASTK.ASM
*   WESTK.ASM
*   NORTHK.ASM
*   SOUTHK.ASM
*
* THIS IS DONE THIS WAY
* IN ORDER TO ENSURE THE
* ABOVE FOUR KEY HANDLING
* ROUTINES DO NOT EXCEED
* THEIR ALLOTED 256 BYTES.
*
*****

* COORDINATES CONVERTER
MCSCCV  EQU     $539C

* PUT CHARACTER VALUE TO
* SCREEN INFORMATION BUFFERS
PTFVAL  EQU     $548C

* MAZE COORDINATES
* AND CONTENTS
MXC     EQU     $53A6
MYC     EQU     $53A7
CELLCC  EQU     $53A8

* STRENGTH VARIABLE
STRNTH  EQU     $549C

PROVAL  EQU     $54A4   PROVISIONS VALUE
```

```
          ORG     $7471

* CHECK FOR PROVISIONS
PROCHK    PSHS    A,B,X
          LDA     CELLCC  CURRENT CELL CONTENTS
          CMPA    #$85    IS IT PROVISIONS?
          BNE     LBLPC1  GO IF NO
          LDA     #$20    BLANK SPACE
          STA     CELLCC  TO CURRENT CONTENTS
          LDB     PROVAL  PROVISIONS VALUE
          CLRA            EXTEND IT
          ADDD    STRNTH  ADD IT TO THE STRENGTH
          STD     STRNTH  SAVE THE NEW STRNTH
          LDB     CELLCC  NEW CURRENT CELL CONTENTS
          CLRA            EXTEND IT
          TFR     D,X
          LDA     MXC     MX-COORDINATE
          LDB     MYC     MY-COORDINATE
          JSR     MCSCCV  CONVERT TO SX,SY
          JSR     PTFVAL  PUT TO SCREEN BUFFER
LBLPC1    PULS    A,B,X

ENDCHK    RTS

          END
```

=====

EASTK: East Key (Right Arrow) Event Handler

The Assembly Language text listing:

```
*****
*
* EASTK.ASM
* MDJ 2023/04/18
*
* EAST KEY
* (RIGHT ARROW)
* EVENT HANDLER
*
*****

* PUT FAKE TEXT ROUTINE
PTFCHR  EQU     $5300

* COORDINATES CONVERTER
MCSCCV  EQU     $539C

* GET CHARACTER VALUE FROM
* SCREEN INFORMATION BUFFERS
GTFVAL  EQU     $5482

* MAZE COORDINATES
* AND CONTENTS
MXC     EQU     $53A6
MYC     EQU     $53A7
CELLCC  EQU     $53A8
MXN     EQU     $53A9
MYN     EQU     $53AA

* VERTICAL DOOR CODE
VRTDOR  EQU     $75

* GAME OVER ROUTINES
GMOVER  EQU     $70F2
GMOVED  EQU     $7101

* STRENGTH REPORTING
STRNTH  EQU     $549C
RPTSTR  EQU     $705A
```

```
* SCORE REPORTING
SCORE   EQU     $549E
RPTSCO  EQU     $70A6

PROCHK  EQU     $7471   PROVISIONS CHECK

* BRUTE FORCE MESSAGE ROUTINES
CLRL13  EQU     $7800
CLRL14  EQU     $7945
MSG001  EQU     $7AF8

        ORG     $5A00

EASTK   PSHS    A,B,X

        JSR     CLRL13  CLEAR LINE 13
        JSR     CLRL14  CLEAR LINE 14

* CHECK FOR LEGAL MOVE
        LDA     MXC     MX-COORDINATE
        LDB     MYC     MY-COORDINATE
        JSR     MCSCCV  CONVERT TO SX,SY
        INCA            POINT TO NEXT SX
        JSR     GTFVAL  GET THE FAKE CHAR
        CMPB    #VRTDOR IS IT AN OPENING
        BEQ     LBL002  GO IF YES
        JSR     MSG001  DISPLAY ERROR MESSAGE

* ADJUST STRENGTH VARIABLE
        PSHS    A,B     STRENGTH EFFECT
        LDD     STRNTH
        CMPD    #2      IS IT AT LIMIT
        BHI     LBL001  GO IF NO
        LDD     #0
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH
        PULS    A,B,X
        JMP     GMOVED  YOU DIED

LBL001  SUBD    #2
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH
        LBRA    LBL004  GO (IT'S NOT AN OPENING)

* REVEAL THE CURRENT CELL
```

```
* CONTENTS ON THE SCREEN.
* (FROM UNDER THE AVATAR)
* TESTING = USE SPACE
LBL002  LDB     CELLCC  CONTENTS
        CLRA            EXTEND IT
        TFR     D,X
        LDA     MXC     MX-COORDINATE
        LDB     MYC     MY-COORDINATE
        JSR     MCSCCV  CONVERT TO SX,SY
        JSR     PTFCHR  PUT TO SCREEN

* GO ONE MAZE CELL EAST
        LDA     MXC     CURRENT MX
        INCA
        STA     MXN     NEW MX
        LDB     MYC     CURRENT MY
        STB     MYN     NEW MY

* SAVE NEW CELL SCREEN CONTENTS
        JSR     MCSCCV  CONVERT TO SX,SY
        JSR     GTFVAL  GET CHAR VALUE
        STB     CELLCC  SAVE AS CONTENTS

* PUT AVATAR TO NEW SCREEN LOCATION
        LDA     MXN     MX-COORDINATE
        LDB     MYN     MY-COORDINATE
        JSR     MCSCCV  CONVERT TO SX,SY
        LDX     #$0000  AVATAR CODE EXTENDED
        JSR     PTFCHR  PUT TO SCREEN

* MAKE THE NEW COORDINATES CURRENT
        LDA     MXN     NEW MX
        STA     MXC     CURRENT MX
        LDB     MYN     NEW MY
        STB     MYC     CURRENT MY

* GO CHECK FOR PROVISIONS
        JSR     PROCHK

* ADJUST STRENGTH VARIABLE
        PSHS    A,B     STRENGTH EFFECT
        LDD     STRNTH
        CMPD    #1      IS IT AT LIMIT
        BHI     LBL003  GO IF NO
        LDD     #0
        STD     STRNTH
        PULS    A,B
```

```
            JSR     RPTSTR  REPORT CURRENT STRENGTH
            PULS    A,B,X
            JMP     GMOVED  YOU DIED

LBL003      SUBD    #1
            STD     STRNTH
            PULS    A,B
            JSR     RPTSTR  REPORT CURRENT STRENGTH

* CHECK FOR QUEST COMPLETION
* (CAN ONLY BE ACCOMPLISHED BY AN
*  EAST MOVE INTO CELL (MX=11, MY=5)
            PSHS    A,B
            LDA     MXC
            CMPA    #11
            BNE     LBLSC3
            LDB     MYC
            CMPB    #5
            BNE     LBLSC3

* ADJUST STRENGTH AND SCORE
* TEMPORARILY SKIP LIMIT CHECK
            LDD     SCORE
            ADDD    STRNTH
            STD     SCORE
            LDD     #0
            STD     STRNTH
            JSR     RPTSTR  REPORT CURRENT STRENGTH
            JSR     RPTSCO  REPORT CURRENT SCORE
            PULS    A,B
            PULS    A,B,X
            JMP     GMOVER  QUEST IS COMPLETE

LBLSC3      PULS    A,B

LBL004      PULS    A,B,X
ENDCHK      RTS

            END
```

=====

WESTK: West Key (Left Arrow) Event Handler

The Assembly Language text listing:

```
*****
*
* WESTK.ASM
* MDJ 2023/04/18
*
* WEST KEY
* (LEFT ARROW)
* EVENT HANDLER
*
*****

* PUT FAKE TEXT ROUTINE
PTFCHR  EQU     $5300

* COORDINATES CONVERTER
MCSCCV  EQU     $539C

* GET CHARACTER VALUE FROM
* SCREEN INFORMATION BUFFERS
GTFVAL  EQU     $5482

* MAZE COORDINATES
* AND CONTENTS
MXC     EQU     $53A6
MYC     EQU     $53A7
CELLCC  EQU     $53A8
MXN     EQU     $53A9
MYN     EQU     $53AA

* VERTICAL DOOR CODE
VRTDOR  EQU     $75

* GAME OVER ROUTINE
GMOVED  EQU     $7101

* STRENGTH REPORTING
STRNTH  EQU     $549C
RPTSTR  EQU     $705A

PROCHK  EQU     $7471   PROVISIONS CHECK
```

```
* BRUTE FORCE MESSAGE ROUTINES
CLRL13  EQU     $7800
CLRL14  EQU     $7945
MSG001  EQU     $7AF8

        ORG     $5B00

WESTK   PSHS    A,B,X

        JSR     CLRL13  CLEAR LINE 13
        JSR     CLRL14  CLEAR LINE 14

* CHECK FOR LEGAL MOVE
        LDA     MXC     MX-COORDINATE
        LDB     MYC     MY-COORDINATE
        JSR     MCSCCV  CONVERT TO SX,SY
        DECA            POINT TO NEXT SX
        JSR     GTFVAL  GET THE FAKE CHAR
        CMPB    #VRTDOR IS IT AN OPENING
        BEQ     LBL002  GO IF YES
        JSR     MSG001  DISPLAY ERROR MESSAGE

* ADJUST STRENGTH VARIABLE
        PSHS    A,B     STRENGTH EFFECT
        LDD     STRNTH
        CMPD    #2      IS IT AT LIMIT
        BHI     LBL001  GO IF NO
        LDD     #0
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH
        LBRA   LBL004
        PULS    A,B,X
        JMP     GMOVED  YOU DIED

LBL001  SUBD    #2
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH
        LBRA    LBL004  GO IF NOT AN OPENING

* REVEAL THE CURRENT CELL
* CONTENTS ON THE SCREEN.
* (FROM UNDER THE AVATAR)
* TESTING = USE SPACE
LBL002  LDB     CELLCC  CONTENTS
```

```
        CLRA            EXTEND IT
        TFR     D,X
        LDA     MXC     MX-COORDINATE
        LDB     MYC     MY-COORDINATE
        JSR     MCSCCV  CONVERT TO SX,SY
        JSR     PTFCHR  PUT TO SCREEN

* GO ONE MAZE CELL WEST
        LDA     MXC     CURRENT MX
        DECA
        STA     MXN     NEW MX
        LDB     MYC     CURRENT MY
        STB     MYN     NEW MY

* SAVE NEW CELL SCREEN CONTENTS
        JSR     MCSCCV  CONVERT TO SX,SY
        JSR     GTFVAL  GET CHAR VALUE
        STB     CELLCC  SAVE AS CONTENTS

* PUT AVATAR TO NEW SCREEN LOCATION
        LDA     MXN     MX-COORDINATE
        LDB     MYN     MY-COORDINATE
        JSR     MCSCCV  CONVERT TO SX,SY
        LDX     #$0000  AVATAR CODE EXTENDED
        JSR     PTFCHR  PUT TO SCREEN

* MAKE THE NEW COORDINATES CURRENT
        LDA     MXN     NEW MX
        STA     MXC     CURRENT MX
        LDB     MYN     NEW MY
        STB     MYC     CURRENT MY

* GO CHECK FOR PROVISIONS
        JSR     PROCHK

* ADJUST STRENGTH VARIABLE
        PSHS    A,B     STRENGTH EFFECT
        LDD     STRNTH
        CMPD    #1      IS IT AT LIMIT
        BHI     LBL003  GO IF NO
        LDD     #0
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH
        PULS    A,B,X
        JMP     GMOVED  YOU DIED
```

```
LBL003  SUBD    #1
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH

LBL004  PULS    A,B,X
ENDCHK  RTS

        END
```

=====

NORTHK: North Key (Up Arrow) Event Handler

The Assembly Language text listing:

```
*****
*
* NORTHK.ASM
* MDJ 2023/04/18
*
* NORTH KEY
* (UP ARROW)
* EVENT HANDLER
*
*****

* PUT FAKE TEXT ROUTINE
PTFCHR  EQU     $5300

* COORDINATES CONVERTER
MCSCCV  EQU     $539C

* GET CHARACTER VALUE FROM
* SCREEN INFORMATION BUFFERS
GTFVAL  EQU     $5482

* MAZE COORDINATES
* AND CONTENTS
MXC     EQU     $53A6
MYC     EQU     $53A7
CELLCC  EQU     $53A8
MXN     EQU     $53A9
MYN     EQU     $53AA

* HORIZONTAL DOOR CODE
HORDOR  EQU     $6C

* GAME OVER ROUTINE
GMOVED  EQU     $7101

* STRENGTH REPORTING
STRNTH  EQU     $549C
RPTSTR  EQU     $705A

PROCHK  EQU     $7471   PROVISIONS CHECK
```

```
* BRUTE FORCE MESSAGE ROUTINES
CLRL13  EQU     $7800
CLRL14  EQU     $7945
MSG001  EQU     $7AF8

        ORG     $5C00

NORTHK  PSHS    A,B,X

        JSR     CLRL13  CLEAR LINE 13
        JSR     CLRL14  CLEAR LINE 14

* CHECK FOR LEGAL MOVE
        LDA     MXC     MX-COORDINATE
        LDB     MYC     MY-COORDINATE
        JSR     MCSCCV  CONVERT TO SX,SY
        DECB            POINT TO NEXT SY
        JSR     GTFVAL  GET THE FAKE CHAR
        CMPB    #HORDOR IS IT AN OPENING
        BEQ     LBL002  GO IF YES
        JSR     MSG001  DISPLAY ERROR MESSAGE

* ADJUST STRENGTH VARIABLE
        PSHS    A,B     STRENGTH EFFECT
        LDD     STRNTH
        CMPD    #2      IS IT AT LIMIT
        BHI     LBL001  GO IF NO
        LDD     #0
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH
        PULS    A,B,X
        JMP     GMOVED  YOU DIED

LBL001  SUBD    #2
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH
        LBRA    LBL004  GO IF NOT AN OPENING

* REVEAL THE CURRENT CELL
* CONTENTS ON THE SCREEN.
* (FROM UNDER THE AVATAR)
* TESTING = USE SPACE
LBL002  LDB     CELLCC  CONTENTS
        CLRA            EXTEND IT
```

```
        TFR     D,X
        LDA     MXC     MX-COORDINATE
        LDB     MYC     MY-COORDINATE
        JSR     MCSCCV  CONVERT TO SX,SY
        JSR     PTFCHR  PUT TO SCREEN

* GO ONE MAZE CELL NORTH
        LDA     MXC     CURRENT MX
        STA     MXN     NEW MX
        LDB     MYC     CURRENT MY
        DECB
        STB     MYN     NEW MY

* SAVE NEW CELL SCREEN CONTENTS
        JSR     MCSCCV  CONVERT TO SX,SY
        JSR     GTFVAL  GET CHAR VALUE
        STB     CELLCC  SAVE AS CONTENTS

* PUT AVATAR TO NEW SCREEN LOCATION
        LDA     MXN     MX-COORDINATE
        LDB     MYN     MY-COORDINATE
        JSR     MCSCCV  CONVERT TO SX,SY
        LDX     #$0000  AVATAR CODE EXTENDED
        JSR     PTFCHR  PUT TO SCREEN

* MAKE THE NEW COORDINATES CURRENT
        LDA     MXN     NEW MX
        STA     MXC     CURRENT MX
        LDB     MYN     NEW MY
        STB     MYC     CURRENT MY

* GO CHECK FOR PROVISIONS
        JSR     PROCHK

* ADJUST STRENGTH VARIABLE
        PSHS    A,B     STRENGTH EFFECT
        LDD     STRNTH
        CMPD    #1      IS IT AT LIMIT
        BHI     LBL003  GO IF NO
        LDD     #0
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH
        BRA    LBL004
        PULS    A,B,X
        JMP     GMOVED  YOU DIED
```

```
LBL003  SUBD    #1
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH

LBL004  PULS    A,B,X
ENDCHK  RTS

        END
```

=====

SOUTHK: South Key (Down Arrow) Event Handler

The Assembly Language text listing:

```
*****
*
* SOUTHK.ASM
* MDJ 2023/04/18
*
* SOUTH KEY
* (DOWN ARROW)
* EVENT HANDLER
*
*****

* PUT FAKE TEXT ROUTINE
PTFCHR  EQU     $5300

* COORDINATES CONVERTER
MCSCCV  EQU     $539C

* GET CHARACTER VALUE FROM
* SCREEN INFORMATION BUFFERS
GTFVAL  EQU     $5482

* MAZE COORDINATES
* AND CONTENTS
MXC     EQU     $53A6
MYC     EQU     $53A7
CELLCC  EQU     $53A8
MXN     EQU     $53A9
MYN     EQU     $53AA

* HORIZONTAL DOOR CODE
HORDOR  EQU     $6C

* GAME OVER ROUTINE
GMOVED  EQU     $7101

* STRENGTH REPORTING
STRNTH  EQU     $549C
RPTSTR  EQU     $705A

PROCHK  EQU     $7471   PROVISIONS CHECK
```

```
* BRUTE FORCE MESSAGE ROUTINES
CLRL13  EQU     $7800
CLRL14  EQU     $7945
MSG001  EQU     $7AF8

        ORG     $5D00

SOUTHK  PSHS    A,B,X

        JSR     CLRL13  CLEAR LINE 13
        JSR     CLRL14  CLEAR LINE 14

* CHECK FOR LEGAL MOVE
        LDA     MXC     MX-COORDINATE
        LDB     MYC     MY-COORDINATE
        JSR     MCSCCV  CONVERT TO SX,SY
        INCB            POINT TO NEXT SY
        JSR     GTFVAL  GET THE FAKE CHAR
        CMPB    #HORDOR IS IT AN OPENING
        BEQ     LBL002  GO IF YES
        JSR     MSG001  DISPLAY ERROR MESSAGE

* ADJUST STRENGTH VARIABLE
        PSHS    A,B     STRENGTH EFFECT
        LDD     STRNTH
        CMPD    #2      IS IT AT LIMIT
        BHI     LBL001  GO IF NO
        LDD     #0
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH
        PULS    A,B,X
        JMP     GMOVED  YOU DIED

LBL001  SUBD    #2
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH
        LBRA    LBL004  GO IF NOT AN OPENING

* REVEAL THE CURRENT CELL
* CONTENTS ON THE SCREEN.
* (FROM UNDER THE AVATAR)
* TESTING = USE SPACE
LBL002  LDB     CELLCC  CONTENTS
        CLRA            EXTEND IT
```

```
        TFR     D,X
        LDA     MXC     MX-COORDINATE
        LDB     MYC     MY-COORDINATE
        JSR     MCSCCV  CONVERT TO SX,SY
        JSR     PTFCHR  PUT TO SCREEN

* GO ONE MAZE CELL SOUTH
        LDA     MXC     CURRENT MX
        STA     MXN     NEW MX
        LDB     MYC     CURRENT MY
        INCB
        STB     MYN     NEW MY

* SAVE NEW CELL SCREEN CONTENTS
        JSR     MCSCCV  CONVERT TO SX,SY
        JSR     GTFVAL  GET CHAR VALUE
        STB     CELLCC  SAVE AS CONTENTS

* PUT AVATAR TO NEW SCREEN LOCATION
        LDA     MXN     MX-COORDINATE
        LDB     MYN     MY-COORDINATE
        JSR     MCSCCV  CONVERT TO SX,SY
        LDX     #$0000  AVATAR CODE EXTENDED
        JSR     PTFCHR  PUT TO SCREEN

* MAKE THE NEW COORDINATES CURRENT
        LDA     MXN     NEW MX
        STA     MXC     CURRENT MX
        LDB     MYN     NEW MY
        STB     MYC     CURRENT MY

* GO CHECK FOR PROVISIONS
        JSR     PROCHK

* ADJUST STRENGTH VARIABLE
        PSHS    A,B     STRENGTH EFFECT
        LDD     STRNTH
        CMPD    #1      IS IT AT LIMIT
        BHI     LBL003  GO IF NO
        LDD     #0
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH
        PULS    A,B,X
        JMP     GMOVED  YOU DIED

LBL003  SUBD    #1
```

```
            STD     STRNTH
            PULS    A,B
            JSR     RPTSTR  REPORT CURRENT STRENGTH

     LBL004 PULS    A,B,X
     ENDCHK RTS

            END
```

=====

TCHARK: Take Key (T-Key) Event Handler

The Assembly Language text listing:

```
*****
*
* TCHARK.ASM
* MDJ 2023/04/18
*
* TAKE KEY
* (T-KEY)
* EVENT HANDLER
*
*****

BAG     EQU     $54A1   THE BAG
GMOVED  EQU     $7101   YOU DIED
CLRL14  EQU     $7945   CLEAR LINE 14
MSG004  EQU     $7E13   "NOTHING HERE"
MSG005  EQU     $7EEA   "NO ROOM"
MSG007  EQU     $7124   "BAG CONTENTS"

* PUT FAKE TEXT ROUTINE
PTFCHR  EQU     $5300

* COORDINATES CONVERTER
MCSCCV  EQU     $539C

* PUT CHARACTER VALUE TO
* SCREEN INFORMATION BUFFERS
PTFVAL  EQU     $548C

* MAZE COORDINATES
* AND CONTENTS
MXC     EQU     $53A6
MYC     EQU     $53A7
CELLCC  EQU     $53A8

* STRENGTH REPORTING
STRNTH  EQU     $549C
RPTSTR  EQU     $705A

        ORG     $5E00
```

```
TCHARK  PSHS    A,B,X

        JSR     CLRL14  CLEAR LINE 14

* CHECK FOR GOSPEL OF JOHN
        LDB     CELLCC  CURRENT CELL CONTENTS
        CMPB    #$E0    IS IT JOHN?
        BNE     LBLDC1  GO IF NO
        STB     BAG     PUT IT IN THE BAG
        LDB     #$20    BLANK SPACE
        STB     CELLCC  TO CURRENT CONTENTS
        CLRA            EXTEND IT
        TFR     D,X
        LDA     MXC     MX-COORDINATE
        LDB     MYC     MY-COORDINATE
        JSR     MCSCCV  CONVERT TO SX,SY
        JSR     PTFVAL  PUT TO SCREEN BUFFER
        JSR     MSG007  "BAG CONTENTS"
        BRA     LBL002

LBLDC1  JSR     MSG004  "NOTHING HERE"

* FAILED ACTION: ADJUST STRENGTH VARIABLE
        PSHS    A,B     STRENGTH EFFECT
        LDD     STRNTH
        CMPD    #2      IS IT AT LIMIT
        BHI     LBL001  GO IF NO
        LDD     #0
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH
        PULS    A,B,X
        JMP     GMOVED  YOU DIED

LBL001  SUBD    #2
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH
        BRA     LBLDC2

* COMPLETED ACTION: ADJUST STRENGTH VARIABLE
LBL002  PSHS    A,B     STRENGTH EFFECT
        LDD     STRNTH
        CMPD    #1      IS IT AT LIMIT
        BHI     LBL003  GO IF NO
        LDD     #0
        STD     STRNTH
```

```
            PULS    A,B
            JSR     RPTSTR  REPORT CURRENT STRENGTH
            PULS    A,B,X
            JMP     GMOVED  YOU DIED

     LBL003 SUBD    #1
            STD     STRNTH
            PULS    A,B
            JSR     RPTSTR  REPORT CURRENT STRENGTH

     LBLDC2 PULS    A,B,X
     ENDCHK RTS

            END
```

=====

LCHARK: Leave Key (L-Key) Event Handler

The Assembly Language text listing:

```
*****
*
* LCHARK.ASM
* MDJ 2023/04/18
*
* LEAVE KEY
* (L-KEY)
* EVENT HANDLER
*
*****

BAG     EQU     $54A1   THE BAG
WHSE    EQU     $54A2   THE WAREHOUSE
DOCVAL  EQU     $54A3   DOCUMENT VALUE
GMOVED  EQU     $7101   YOU DIED
CLRL14  EQU     $7945   CLEAR LINE 14
MSG005  EQU     $7EEA   "NO ROOM"
MSG006  EQU     $7F34   "EMPTY BAG"

* PUT FAKE TEXT ROUTINE
PTFCHR  EQU     $5300

* COORDINATES CONVERTER
MCSCCV  EQU     $539C

* PUT CHARACTER VALUE TO
* SCREEN INFORMATION BUFFERS
PTFVAL  EQU     $548C

* MAZE COORDINATES
* AND CONTENTS
MXC     EQU     $53A6
MYC     EQU     $53A7
CELLCC  EQU     $53A8

* STRENGTH REPORTING
STRNTH  EQU     $549C
RPTSTR  EQU     $705A

* SCORE REPORTING
```

```
SCORE   EQU     $549E
RPTSCO  EQU     $70A6

        ORG     $5F00

LCHARK  PSHS    A,B,X

        JSR     CLRL14  CLEAR LINE 14

* CHECK BAG FOR GOSPEL OF JOHN
        LDA     BAG     GET BAG CONTENTS
        CMPA    #$E0    IS IT JOHN?
        BNE     LBLBC2  GO IF NO
        LDB     CELLCC  CURRENT CELL CONTENTS
        CMPB    #$7F    IS IT THE WAREHOUSE?
        BEQ     LBLBC1  GO IF YES
        CMPB    #$20    IS IT A BLANK SPACE?
        BNE     LBLBC3  GO IF NO

* PUT BAG CONTENTS TO CELL
        LDA     #$20    EMPTY THE BAG
        STA     BAG
        LDB     #$E0    PUT JOHN IN THE CELL
        STB     CELLCC
        CLRA            EXTEND IT
        TFR     D,X
        LDA     MXC     MX-COORDINATE
        LDB     MYC     MY-COORDINATE
        JSR     MCSCCV  CONVERT TO SX,SY
        JSR     PTFVAL  PUT TO SCREEN BUFFER
        BRA     LBL002

* PUT BAG CONTENTS TO WAREHOUSE
LBLBC1  LDA     #$20    EMPTY THE BAG
        STA     BAG
        LDA     #$E0    PUT JOHN IN WHSE
        STA     WHSE
        LDB     DOCVAL  GET THE DOCUMENT VALUE
        CLRA            EXTEND IT
        ADDD    SCORE   ADD IT TO THE SCORE
        STD     SCORE   SAVE THE NEW SCORE
        JSR     RPTSCO  REPORT THE NEW SCORE
        BRA     LBL002

LBLBC2  JSR     MSG006  "BAG EMPTY"
        BRA     LBL000
LBLBC3  JSR     MSG005  "NO ROOM"
```

```
* FAILED ACTION: ADJUST STRENGTH VARIABLE
LBL000  PSHS    A,B     STRENGTH EFFECT
        LDD     STRNTH
        CMPD    #2      IS IT AT LIMIT
        BHI     LBL001  GO IF NO
        LDD     #0
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH
        PULS    A,B,X
        JMP     GMOVED  YOU DIED

LBL001  SUBD    #2
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH
        BRA     LBLBC4

* COMPLETED ACTION: ADJUST STRENGTH VARIABLE
LBL002  PSHS    A,B     STRENGTH EFFECT
        LDD     STRNTH
        CMPD    #1      IS IT AT LIMIT
        BHI     LBL003  GO IF NO
        LDD     #0
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH
        PULS    A,B,X
        JMP     GMOVED  YOU DIED

LBL003  SUBD    #1
        STD     STRNTH
        PULS    A,B
        JSR     RPTSTR  REPORT CURRENT STRENGTH

LBLBC4  PULS    A,B,X
ENDCHK  RTS

        END
```

=====

BCHARK: Bag Inventory Key (B-Key) Event Handler

The Assembly Language text listing:

```
*****
*
* BCHARK.ASM
* MDJ 2023/04/18
*
* BAG INVENTORY KEY
* (B-KEY)
* EVENT HANDLER
*
*****

BAG     EQU     $54A1   THE BAG
GMOVED  EQU     $7101   YOU DIED
CLRL14  EQU     $7945   CLEAR LINE 14
MSG006  EQU     $7F34   "EMPTY BAG"
MSG007  EQU     $7124   "BAG CONTENTS"

        ORG     $6000

BCHARK  PSHS    A

        JSR     CLRL14  CLEAR LINE 14

        LDA     BAG     BAG CONTENTS
        CMPA    #$E0    IS IT JOHN?
        BNE     LBL001  GO IF NO
        JSR     MSG007  "BAG CONTENTS"
        BRA     LBL002

LBL001  JSR     MSG006  "EMPTY BAG"

LBL002  PULS    A
ENDCHK  RTS

        END
```

=====

ICHARK: Warehouse Inventory Key (I-Key) Event Handler

The Assembly Language text listing:

```
*****
*
* ICHARK.ASM
* MDJ 2023/04/18
*
* WAREHOUSE INVENTORY KEY
* (I-KEY)
* EVENT HANDLER
*
*****

WHSE    EQU     $54A2   THE WAREHOUSE
GMOVED  EQU     $7101   YOU DIED
CLRL14  EQU     $7945   CLEAR LINE 14
MSG008  EQU     $724B   "EMPTY WHSE"
MSG009  EQU     $7336   "WHSE INVENTORY"

        ORG     $6100

ICHARK  PSHS    A

        JSR     CLRL14  CLEAR LINE 14

        LDA     WHSE    WHSE CONTENTS
        CMPA    #$E0    IS IT JOHN?
        BNE     LBL001  GO IF NO
        JSR     MSG009  "WHSE INVENTORY"
        BRA     LBL002

LBL001  JSR     MSG008  "EMPTY WHSE"

LBL002  PULS    A
ENDCHK  RTS

        END
```

=====

XCHARK: Exit Key (X-Key) Event Handler

KNOWN BUG: The X-Key freezes the system.

The Assembly Language text listing:

```
*****
*
* XCHARK.ASM
* MDJ 2023/04/18
*
* EXIT KEY
* (X-KEY)
* EVENT HANDLER
*
* NOT IMPLEMENTED FOR
* MALKY'S WARREN
*
*****

        ORG     $6600

XCHARK  RTS

        END
```

=====

Unimplemented Key Handlers

These are dummies; reserved for future use. They are not implemented in Malky's Warren, but their skeletons are presented here to avoid confusion in the GMLOOP Chapter.

The Assembly Language text listings:

```
*****
*
* UCHARK.ASM
* MDJ 2023/04/11
*
* GO UP KEY
* (U-KEY)
* EVENT HANDLER
*
* NOT IMPLEMENTED FOR
* MALKY'S WARREN
*
*****

        ORG     $6200

UCHARK  RTS

        END

*****
*
* DCHARK.ASM
* MDJ 2023/04/11
*
* GO DOWN KEY
* (D-KEY)
* EVENT HANDLER
*
* NOT IMPLEMENTED FOR
* MALKY'S WARREN
*
*****

        ORG     $6300

DCHARK  RTS
```

```
        END

*****
*
* PCHARK.ASM
* MDJ 2023/04/11
*
* PAUSE KEY
* (P-KEY)
* EVENT HANDLER
*
* NOT IMPLEMENTED FOR
* MALKY'S WARREN
*
*****

        ORG     $6400

PCHARK  RTS

        END

*****
*
* RCHARK.ASM
* MDJ 2023/04/11
*
* RESUME KEY
* (R-KEY)
* EVENT HANDLER
*
* NOT IMPLEMENTED FOR
* MALKY'S WARREN
*
*****

        ORG     $6500

RCHARK  RTS

        END

*****
*
* CCHARK.ASM
* MDJ 2023/04/11
```

```
*
* CONFIRM KEY
* (C-KEY)
* EVENT HANDLER
*
* NOT IMPLEMENTED FOR
* MALKY'S WARREN
*
*****

        ORG     $6700

CCHARK  RTS

        END

*****
*
* GCHARK.ASM
* MDJ 2023/04/11
*
* NEW GAME KEY
* (G-KEY)
* EVENT HANDLER
*
* NOT IMPLEMENTED FOR
* MALKY'S WARREN
*
*****

        ORG     $6800

GCHARK  RTS

        END
```

=====

CLRL13: Clear Line 13 of the Screen

The Assembly Language text listing:

```
*****
*
* CLRL13.ASM
* MDJ 2023/04/15
*
* BRUTE FORCE
* CLEAR LINE 13
*
* FUTURE WORK:
* REPLACE WITH
* PMODE 4 FAKE
* STRING OPS
*
* ENTRY CONDITIONS
* NONE
*
* EXIT CONDITIONS:
* NONE
*
*****

PTFCHR  EQU     $5300

        ORG     $7800

MSG001  PSHS    A,B,X

        LDA     #0
        LDB     #13
        LDX     #$0020
        JSR     PTFCHR
        LDA     #1
        LDB     #13
        LDX     #$0020
        JSR     PTFCHR
        LDA     #2
        LDB     #13
        LDX     #$0020
        JSR     PTFCHR
        LDA     #3
        LDB     #13
        LDX     #$0020
```

```
JSR     PTFCHR
LDA     #4
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #5
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #6
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #7
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #8
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #9
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #10
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #11
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #12
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #13
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #14
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #15
LDB     #13
```

```
LDX     #$0020
JSR     PTFCHR
LDA     #16
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #17
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #18
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #19
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #20
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #21
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #22
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #23
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #24
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #25
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #26
LDB     #13
LDX     #$0020
JSR     PTFCHR
LDA     #27
```

```
            LDB     #13
            LDX     #$0020
            JSR     PTFCHR
            LDA     #28
            LDB     #13
            LDX     #$0020
            JSR     PTFCHR
            LDA     #29
            LDB     #13
            LDX     #$0020
            JSR     PTFCHR
            LDA     #30
            LDB     #13
            LDX     #$0020
            JSR     PTFCHR
            LDA     #31
            LDB     #13
            LDX     #$0020
            JSR     PTFCHR

            PULS    A,B,X
ENDCHK      RTS

            END
```

=====

CLRL14: Clear Line 14 of the Screen

The Assembly Language text listing:

```
*****
*
* CLRL14.ASM
* MDJ 2023/04/15
*
* BRUTE FORCE
* CLEAR LINE 14
*
* FUTURE WORK:
* REPLACE WITH
* PMODE 4 FAKE
* STRING OPS
*
* ENTRY CONDITIONS
* NONE
*
* EXIT CONDITIONS:
* NONE
*
*****

PTFCHR  EQU     $5300

        ORG     $7945

MSG001  PSHS    A,B,X

        LDA     #0
        LDB     #14
        LDX     #$0020
        JSR     PTFCHR
        LDA     #1
        LDB     #14
        LDX     #$0020
        JSR     PTFCHR
        LDA     #2
        LDB     #14
        LDX     #$0020
        JSR     PTFCHR
        LDA     #3
        LDB     #14
        LDX     #$0020
```

```
JSR     PTFCHR
LDA     #4
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #5
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #6
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #7
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #8
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #9
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #10
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #11
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #12
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #13
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #14
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #15
LDB     #14
```

```
LDX     #$0020
JSR     PTFCHR
LDA     #16
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #17
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #18
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #19
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #20
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #21
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #22
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #23
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #24
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #25
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #26
LDB     #14
LDX     #$0020
JSR     PTFCHR
LDA     #27
```

```
            LDB     #14
            LDX     #$0020
            JSR     PTFCHR
            LDA     #28
            LDB     #14
            LDX     #$0020
            JSR     PTFCHR
            LDA     #29
            LDB     #14
            LDX     #$0020
            JSR     PTFCHR
            LDA     #30
            LDB     #14
            LDX     #$0020
            JSR     PTFCHR
            LDA     #31
            LDB     #14
            LDX     #$0020
            JSR     PTFCHR

            PULS    A,B,X
ENDCHK      RTS

            END
```

=====

CLRSTR: Clear the Screen's Strength Field

The Assembly Language text listing:

```
*****
*
* CLRSTR.ASM
* MDJ 2023/04/15
*
* BRUTE FORCE
* CLEAR THE
* STRENGTH FIELD
*
* FUTURE WORK:
* REPLACE WITH
* PMODE 4 FAKE
* STRING OPS
*
* ENTRY CONDITIONS
* NONE
*
* EXIT CONDITIONS:
* NONE
*
*****

PTFCHR  EQU     $5300

        ORG     $7A8A

MSG001  PSHS    A,B,X

        LDA     #11
        LDB     #15
        LDX     #$0020
        JSR     PTFCHR
        LDA     #12
        LDB     #15
        LDX     #$0020
        JSR     PTFCHR
        LDA     #13
        LDB     #15
        LDX     #$0020
        JSR     PTFCHR
```

```
            LDA     #14
            LDB     #15
            LDX     #$0020
            JSR     PTFCHR
            LDA     #15
            LDB     #15
            LDX     #$0020
            JSR     PTFCHR

            PULS    A,B,X
ENDCHK      RTS

            END
```

=====

CLRSCO: Clear the Screen's Score Field

The Assembly Language text listing:

```
*****
*
* CLRSCO.ASM
* MDJ 2023/04/15
*
* BRUTE FORCE
* CLEAR THE
* SCORE FIELD
*
* FUTURE WORK:
* REPLACE WITH
* PMODE 4 FAKE
* STRING OPS
*
* ENTRY CONDITIONS
* NONE
*
* EXIT CONDITIONS:
* NONE
*
*****

PTFCHR  EQU     $5300

        ORG     $7AC1

MSG001  PSHS    A,B,X

        LDA     #27
        LDB     #15
        LDX     #$0020
        JSR     PTFCHR
        LDA     #28
        LDB     #15
        LDX     #$0020
        JSR     PTFCHR
        LDA     #29
        LDB     #15
        LDX     #$0020
        JSR     PTFCHR
```

```
            LDA     #30
            LDB     #15
            LDX     #$0020
            JSR     PTFCHR
            LDA     #31
            LDB     #15
            LDX     #$0020
            JSR     PTFCHR

            PULS    A,B,X
    ENDCHK  RTS

            END
```

=====

DECMAL: Get the ASCII Decimal Representation of a 16-bit Unsigned Integer

The Assembly Language text listing:

```
*****
*
* DECMAL.ASM
* MDJ 2023/04/15
*
* GET THE ASCII
* REPRESENTATION
* OF A DECIMAL
* NUMBER BETWEEN
* 0 AND 65535
*
* ENTRY CONDITIONS
* D = NUMBER
*
* EXIT CONDITIONS:
* DIGIT4 THROUGH DIGIT0
* HOLD THE REPRESENTATION
*
*****

        ORG     $7000

DECMAL  PSHS    A,B     SAVE THE NUMBER
        BRA     LBL001

DIGIT4  RMB     1
DIGIT3  RMB     1
DIGIT2  RMB     1
DIGIT1  RMB     1
DIGIT0  RMB     1

* PRELOAD THE DIGITS WITH ASCII "0"
LBL001  LDA     #$30    = DECIMAL 48 = "0"
        STA     DIGIT4
        STA     DIGIT3
        STA     DIGIT2
        STA     DIGIT1
        STA     DIGIT0

        PULS    A,B     RETRIEVE THE NUMBER
```

```
        * FORM DIGIT 4
        LBLDG4  CMPD    #10000
                BLO     LBLDG3
                INC     DIGIT4
                SUBD    #10000
                BRA     LBLDG4

        * FORM DIGIT 3
        LBLDG3  CMPD    #1000
                BLO     LBLDG2
                INC     DIGIT3
                SUBD    #1000
                BRA     LBLDG3

        * FORM DIGIT 2
        LBLDG2  CMPD    #100
                BLO     LBLDG1
                INC     DIGIT2
                SUBD    #100
                BRA     LBLDG2

        * FORM DIGIT 1
        LBLDG1  CMPD    #10
                BLO     LBLDG0
                INC     DIGIT1
                SUBD    #10
                BRA     LBLDG1

        * FORM DIGIT 0
        LBLDG0  ADDB    #$30
                STB     DIGIT0

        ENDCHK  RTS

                END
```

=====

RPTSTR: Strength Reporter

The Assembly Language text listing:

```
*****
*
* RPTSTR.ASM
* MDJ 2023/04/15
*
* BRUTE FORCE
* STRENGTH REPORTER
*
* FUTURE WORK:
* REPLACE WITH
* PMODE 4 FAKE
* STRING OPS
*
* ENTRY CONDITIONS
* NONE
*
* EXIT CONDITIONS:
* NONE
*
*****

PTFCHR  EQU     $5300
STRNTH  EQU     $549C
DECMAL  EQU     $7000
DIGIT4  EQU     $7004
DIGIT3  EQU     $7005
DIGIT2  EQU     $7006
DIGIT1  EQU     $7007
DIGIT0  EQU     $7008

        ORG     $705A

RPTSTR  PSHS    A,B,X

        LDD     STRNTH  GET CURRENT STRENGTH
        JSR     DECMAL  EXPRESS AS A DECIMAL

        LDB     DIGIT4  GET THE DIGIT
        CLRA            EXTEND IT
        TFR     D,X     MOVE IT TO REG X
        LDA     #11     X-COORDINATE
        LDB     #15     Y-COORDINATE
```

```
            JSR     PTFCHR  REPORT IT

            LDB     DIGIT3  GET THE DIGIT
            CLRA            EXTEND IT
            TFR     D,X     MOVE IT TO REG X
            LDA     #12     X-COORDINATE
            LDB     #15     Y-COORDINATE
            JSR     PTFCHR  REPORT IT

            LDB     DIGIT2  GET THE DIGIT
            CLRA            EXTEND IT
            TFR     D,X     MOVE IT TO REG X
            LDA     #13     X-COORDINATE
            LDB     #15     Y-COORDINATE
            JSR     PTFCHR  REPORT IT

            LDB     DIGIT1  GET THE DIGIT
            CLRA            EXTEND IT
            TFR     D,X     MOVE IT TO REG X
            LDA     #14     X-COORDINATE
            LDB     #15     Y-COORDINATE
            JSR     PTFCHR  REPORT IT

            LDB     DIGIT0  GET THE DIGIT
            CLRA            EXTEND IT
            TFR     D,X     MOVE IT TO REG X
            LDA     #15     X-COORDINATE
            LDB     #15     Y-COORDINATE
            JSR     PTFCHR  REPORT IT

            PULS    A,B,X
ENDCHK      RTS

            END
```

=====

RPTSCO: Score Reporter

The Assembly Language text listing:

```
*****
*
* RPTSCO.ASM
* MDJ 2023/04/15
*
* BRUTE FORCE
* SCORE REPORTER
*
* FUTURE WORK:
* REPLACE WITH
* PMODE 4 FAKE
* STRING OPS
*
* ENTRY CONDITIONS
* NONE
*
* EXIT CONDITIONS:
* NONE
*
*****

PTFCHR  EQU     $5300
SCORE   EQU     $549E
DECMAL  EQU     $7000
DIGIT4  EQU     $7004
DIGIT3  EQU     $7005
DIGIT2  EQU     $7006
DIGIT1  EQU     $7007
DIGIT0  EQU     $7008

        ORG     $70A6

RPTSTR  PSHS    A,B,X

        LDD     SCORE   GET CURRENT STRENGTH
        JSR     DECMAL  EXPRESS AS A DECIMAL

        LDB     DIGIT4  GET THE DIGIT
        CLRA            EXTEND IT
        TFR     D,X     MOVE IT TO REG X
        LDA     #27     X-COORDINATE
        LDB     #15     Y-COORDINATE
```

```
            JSR     PTFCHR  REPORT IT

            LDB     DIGIT3  GET THE DIGIT
            CLRA            EXTEND IT
            TFR     D,X     MOVE IT TO REG X
            LDA     #28     X-COORDINATE
            LDB     #15     Y-COORDINATE
            JSR     PTFCHR  REPORT IT

            LDB     DIGIT2  GET THE DIGIT
            CLRA            EXTEND IT
            TFR     D,X     MOVE IT TO REG X
            LDA     #29     X-COORDINATE
            LDB     #15     Y-COORDINATE
            JSR     PTFCHR  REPORT IT

            LDB     DIGIT1  GET THE DIGIT
            CLRA            EXTEND IT
            TFR     D,X     MOVE IT TO REG X
            LDA     #30     X-COORDINATE
            LDB     #15     Y-COORDINATE
            JSR     PTFCHR  REPORT IT

            LDB     DIGIT0  GET THE DIGIT
            CLRA            EXTEND IT
            TFR     D,X     MOVE IT TO REG X
            LDA     #31     X-COORDINATE
            LDB     #15     Y-COORDINATE
            JSR     PTFCHR  REPORT IT

            PULS    A,B,X
ENDCHK      RTS

            END
```

=====

MSG001: “You Can’t Go That Way”

The Assembly Language text listing:

```
*****
*
* MSG001.ASM
* MDJ 2023/04/15
*
* BRUTE FORCE
* MESSAGE NO. 001
*
* FUTURE WORK:
* REPLACE WITH
* PMODE 4 FAKE
* STRING OPS
*
* ENTRY CONDITIONS
* NONE
*
* EXIT CONDITIONS:
* NONE
*
*****

PTFCHR  EQU     $5300

        ORG     $7AF8

MSG001  PSHS    A,B,X

        LDA     #0
        LDB     #14
        LDX     #$0059  "Y"
        JSR     PTFCHR
        LDA     #1
        LDB     #14
        LDX     #$004F  "O"
        JSR     PTFCHR
        LDA     #2
        LDB     #14
        LDX     #$0055  "U"
        JSR     PTFCHR
        LDA     #3
        LDB     #14
        LDX     #$0020  SPACE
```

```
JSR     PTFCHR
LDA     #4
LDB     #14
LDX     #$0043  "C"
JSR     PTFCHR
LDA     #5
LDB     #14
LDX     #$0041  "A"
JSR     PTFCHR
LDA     #6
LDB     #14
LDX     #$004E  "N"
JSR     PTFCHR
LDA     #7
LDB     #14
LDX     #$0027  "'"
JSR     PTFCHR
LDA     #8
LDB     #14
LDX     #$0054  "T"
JSR     PTFCHR
LDA     #9
LDB     #14
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #10
LDB     #14
LDX     #$0047  "G"
JSR     PTFCHR
LDA     #11
LDB     #14
LDX     #$004F  "O"
JSR     PTFCHR
LDA     #12
LDB     #14
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #13
LDB     #14
LDX     #$0054  "T"
JSR     PTFCHR
LDA     #14
LDB     #14
LDX     #$0048  "H"
JSR     PTFCHR
LDA     #15
LDB     #14
```

```
            LDX     #$0041  "A"
            JSR     PTFCHR
            LDA     #16
            LDB     #14
            LDX     #$0054  "T"
            JSR     PTFCHR
            LDA     #17
            LDB     #14
            LDX     #$0020  SPACE
            JSR     PTFCHR
            LDA     #18
            LDB     #14
            LDX     #$0057  "W"
            JSR     PTFCHR
            LDA     #19
            LDB     #14
            LDX     #$0041  "A"
            JSR     PTFCHR
            LDA     #20
            LDB     #14
            LDX     #$0059  "Y"
            JSR     PTFCHR
            LDA     #21
            LDB     #14
            LDX     #$002E  "."
            JSR     PTFCHR

            PULS    A,B,X
ENDCHK      RTS

            END
```

=====

MSG002: “ ** Game Over: You Died!”

The Assembly Language text listing:

```
*****
*
* MSG002.ASM
* MDJ 2023/04/16
*
* BRUTE FORCE
* GAME OVER MESSAGE
* YOU DIED!
*
* FUTURE WORK:
* REPLACE WITH
* PMODE 4 FAKE
* STRING OPS
*
* ENTRY CONDITIONS
* NONE
*
* EXIT CONDITIONS:
* NONE
*
*****

PTFCHR  EQU     $5300

        ORG     $7BD9

MSG002  PSHS    A,B,X

        LDA     #0
        LDB     #13
        LDX     #$0020  SPACE
        JSR     PTFCHR
        LDA     #1
        LDB     #13
        LDX     #$0020  SPACE
        JSR     PTFCHR
        LDA     #2
        LDB     #13
        LDX     #$002A  "*"
        JSR     PTFCHR
        LDA     #3
        LDB     #13
```

```
LDX     #$002A  "*"
JSR     PTFCHR
LDA     #4
LDB     #13
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #5
LDB     #13
LDX     #$0047  "G"
JSR     PTFCHR
LDA     #6
LDB     #13
LDX     #$0041  "A"
JSR     PTFCHR
LDA     #7
LDB     #13
LDX     #$004D  "M"
JSR     PTFCHR
LDA     #8
LDB     #13
LDX     #$0045  "E"
JSR     PTFCHR
LDA     #9
LDB     #13
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #10
LDB     #13
LDX     #$004F  "O"
JSR     PTFCHR
LDA     #11
LDB     #13
LDX     #$0056  "V"
JSR     PTFCHR
LDA     #12
LDB     #13
LDX     #$0045  "E"
JSR     PTFCHR
LDA     #13
LDB     #13
LDX     #$0052  "R"
JSR     PTFCHR
LDA     #14
LDB     #13
LDX     #$003A  ":"
JSR     PTFCHR
LDA     #15
```

```
            LDB     #13
            LDX     #$0020  SPACE
            JSR     PTFCHR
            LDA     #16
            LDB     #13
            LDX     #$0059  "Y"
            JSR     PTFCHR
            LDA     #17
            LDB     #13
            LDX     #$004F  "O"
            JSR     PTFCHR
            LDA     #18
            LDB     #13
            LDX     #$0055  "U"
            JSR     PTFCHR
            LDA     #19
            LDB     #13
            LDX     #$0020  SPACE
            JSR     PTFCHR
            LDA     #20
            LDB     #13
            LDX     #$0044  "D"
            JSR     PTFCHR
            LDA     #21
            LDB     #13
            LDX     #$0049  "I"
            JSR     PTFCHR
            LDA     #22
            LDB     #13
            LDX     #$0045  "E"
            JSR     PTFCHR
            LDA     #23
            LDB     #13
            LDX     #$0044  "D"
            JSR     PTFCHR
            LDA     #24
            LDB     #13
            LDX     #$0021  "!"
            JSR     PTFCHR

            PULS    A,B,X
ENDCHK      RTS

            END
```

=====

MSG003:
“ ** Game Over: Quest Complete.”

The Assembly Language text listing:

```
*****
*
* MSG003.ASM
* MDJ 2023/04/15
*
* BRUTE FORCE
* GAME OVER MESSAGE
* QUEST COMPLETE.
*
* FUTURE WORK:
* REPLACE WITH
* PMODE 4 FAKE
* STRING OPS
*
* ENTRY CONDITIONS
* NONE
*
* EXIT CONDITIONS:
* NONE
*
*****

PTFCHR  EQU     $5300

        ORG     $7CD8

MSG003  PSHS    A,B,X

        LDA     #0
        LDB     #13
        LDX     #$0020  SPACE
        JSR     PTFCHR
        LDA     #1
        LDB     #13
        LDX     #$0020  SPACE
        JSR     PTFCHR
        LDA     #2
        LDB     #13
        LDX     #$002A  "*"
        JSR     PTFCHR
```

```
LDA     #3
LDB     #13
LDX     #$002A  "*"
JSR     PTFCHR
LDA     #4
LDB     #13
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #5
LDB     #13
LDX     #$0047  "G"
JSR     PTFCHR
LDA     #6
LDB     #13
LDX     #$0041  "A"
JSR     PTFCHR
LDA     #7
LDB     #13
LDX     #$004D  "M"
JSR     PTFCHR
LDA     #8
LDB     #13
LDX     #$0045  "E"
JSR     PTFCHR
LDA     #9
LDB     #13
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #10
LDB     #13
LDX     #$004F  "O"
JSR     PTFCHR
LDA     #11
LDB     #13
LDX     #$0056  "V"
JSR     PTFCHR
LDA     #12
LDB     #13
LDX     #$0045  "E"
JSR     PTFCHR
LDA     #13
LDB     #13
LDX     #$0052  "R"
JSR     PTFCHR
LDA     #14
LDB     #13
LDX     #$003A  ":"
```

```
JSR     PTFCHR
LDA     #15
LDB     #13
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #16
LDB     #13
LDX     #$0051  "Q"
JSR     PTFCHR
LDA     #17
LDB     #13
LDX     #$0055  "U"
JSR     PTFCHR
LDA     #18
LDB     #13
LDX     #$0045  "E"
JSR     PTFCHR
LDA     #19
LDB     #13
LDX     #$0053  "S"
JSR     PTFCHR
LDA     #20
LDB     #13
LDX     #$0054  "T"
JSR     PTFCHR
LDA     #21
LDB     #13
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #22
LDB     #13
LDX     #$0043  "C"
JSR     PTFCHR
LDA     #23
LDB     #13
LDX     #$004F  "O"
JSR     PTFCHR
LDA     #24
LDB     #13
LDX     #$004D  "M"
JSR     PTFCHR
LDA     #25
LDB     #13
LDX     #$0050  "P"
JSR     PTFCHR
LDA     #26
LDB     #13
```

```
            LDX     #$004C  "L"
            JSR     PTFCHR
            LDA     #27
            LDB     #13
            LDX     #$0045  "E"
            JSR     PTFCHR
            LDA     #28
            LDB     #13
            LDX     #$0054  "T"
            JSR     PTFCHR
            LDA     #29
            LDB     #13
            LDX     #$0045  "E"
            JSR     PTFCHR
            LDA     #30
            LDB     #13
            LDX     #$002E  "."
            JSR     PTFCHR

            PULS    A,B,X
ENDCHK      RTS

            END
```

=====

MSG004: “There’s Nothing Here.”

The Assembly Language text listing:

```
*****
*
* MSG004.ASM
* MDJ 2023/04/17
*
* BRUTE FORCE
* NOTHING HERE MESSAGE
*
* FUTURE WORK:
* REPLACE WITH
* PMODE 4 FAKE
* STRING OPS
*
* ENTRY CONDITIONS
* NONE
*
* EXIT CONDITIONS:
* NONE
*
*****

PTFCHR  EQU     $5300

        ORG     $7E13

MSG001  PSHS    A,B,X

        LDA     #0
        LDB     #14
        LDX     #$0054  "T"
        JSR     PTFCHR
        LDA     #1
        LDB     #14
        LDX     #$0048  "H"
        JSR     PTFCHR
        LDA     #2
        LDB     #14
        LDX     #$0045  "E"
        JSR     PTFCHR
        LDA     #3
        LDB     #14
        LDX     #$0052  "R"
```

```
JSR     PTFCHR
LDA     #4
LDB     #14
LDX     #$0045  "E"
JSR     PTFCHR
LDA     #5
LDB     #14
LDX     #$0027  "'"
JSR     PTFCHR
LDA     #6
LDB     #14
LDX     #$0053  "S"
JSR     PTFCHR
LDA     #7
LDB     #14
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #8
LDB     #14
LDX     #$004E  "N"
JSR     PTFCHR
LDA     #9
LDB     #14
LDX     #$004F  "O"
JSR     PTFCHR
LDA     #10
LDB     #14
LDX     #$0054  "T"
JSR     PTFCHR
LDA     #11
LDB     #14
LDX     #$0048  "H"
JSR     PTFCHR
LDA     #12
LDB     #14
LDX     #$0049  "I"
JSR     PTFCHR
LDA     #13
LDB     #14
LDX     #$004E  "N"
JSR     PTFCHR
LDA     #14
LDB     #14
LDX     #$0047  "G"
JSR     PTFCHR
LDA     #15
LDB     #14
```

```
            LDX     #$0020  SPACE
            JSR     PTFCHR
            LDA     #16
            LDB     #14
            LDX     #$0048  "H"
            JSR     PTFCHR
            LDA     #17
            LDB     #14
            LDX     #$0045  "E"
            JSR     PTFCHR
            LDA     #18
            LDB     #14
            LDX     #$0052  "R"
            JSR     PTFCHR
            LDA     #19
            LDB     #14
            LDX     #$0045  "E"
            JSR     PTFCHR
            LDA     #20
            LDB     #14
            LDX     #$002E  "."
            JSR     PTFCHR

            PULS    A,B,X
ENDCHK      RTS

            END
```

=====

MSG005: “No Room.”

The Assembly Language text listing:

```
*****
*
* MSG005.ASM
* MDJ 2023/04/17
*
* BRUTE FORCE
* NO ROOM MESSAGE
*
* FUTURE WORK:
* REPLACE WITH
* PMODE 4 FAKE
* STRING OPS
*
* ENTRY CONDITIONS
* NONE
*
* EXIT CONDITIONS:
* NONE
*
*****

PTFCHR  EQU     $5300

        ORG     $7EEA

MSG001  PSHS    A,B,X

        LDA     #0
        LDB     #14
        LDX     #$004E  "N"
        JSR     PTFCHR
        LDA     #1
        LDB     #14
        LDX     #$004F  "O"
        JSR     PTFCHR
        LDA     #2
        LDB     #14
        LDX     #$0020  SPACE
        JSR     PTFCHR
        LDA     #3
        LDB     #14
        LDX     #$0052  "R"
```

```
            JSR     PTFCHR
            LDA     #4
            LDB     #14
            LDX     #$004F  "O"
            JSR     PTFCHR
            LDA     #5
            LDB     #14
            LDX     #$004F  "O"
            JSR     PTFCHR
            LDA     #6
            LDB     #14
            LDX     #$004D  "M"
            JSR     PTFCHR
            LDA     #7
            LDB     #14
            LDX     #$002E  "."
            JSR     PTFCHR

            PULS    A,B,X
ENDCHK      RTS

            END
```

=====

MSG006: “The Bag is Empty.”

The Assembly Language text listing:

```
*****
*
* MSG006.ASM
* MDJ 2023/04/17
*
* BRUTE FORCE
* EMPTY BAG MESSAGE
*
* FUTURE WORK:
* REPLACE WITH
* PMODE 4 FAKE
* STRING OPS
*
* ENTRY CONDITIONS
* NONE
*
* EXIT CONDITIONS:
* NONE
*
*****

PTFCHR  EQU     $5300

        ORG     $7F34

MSG001  PSHS    A,B,X

        LDA     #0
        LDB     #14
        LDX     #$0054  "T"
        JSR     PTFCHR
        LDA     #1
        LDB     #14
        LDX     #$0048  "H"
        JSR     PTFCHR
        LDA     #2
        LDB     #14
        LDX     #$0045  "E"
        JSR     PTFCHR
        LDA     #3
        LDB     #14
        LDX     #$0020  SPACE
```

```
JSR     PTFCHR
LDA     #4
LDB     #14
LDX     #$0042  "B"
JSR     PTFCHR
LDA     #5
LDB     #14
LDX     #$0041  "A"
JSR     PTFCHR
LDA     #6
LDB     #14
LDX     #$0047  "G"
JSR     PTFCHR
LDA     #7
LDB     #14
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #8
LDB     #14
LDX     #$0049  "I"
JSR     PTFCHR
LDA     #9
LDB     #14
LDX     #$0053  "S"
JSR     PTFCHR
LDA     #10
LDB     #14
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #11
LDB     #14
LDX     #$0045  "E"
JSR     PTFCHR
LDA     #12
LDB     #14
LDX     #$004D  "M"
JSR     PTFCHR
LDA     #13
LDB     #14
LDX     #$0050  "P"
JSR     PTFCHR
LDA     #14
LDB     #14
LDX     #$0054  "T"
JSR     PTFCHR
LDA     #15
LDB     #14
```

```
            LDX     #$0059  "Y"
            JSR     PTFCHR
            LDA     #16
            LDB     #14
            LDX     #$002E  "."
            JSR     PTFCHR

            PULS    A,B,X
    ENDCHK  RTS

            END
```

=====

MSG007:
“Bag Contents: Gospel of John.”

The Assembly Language text listing:

```
*****
*
* MSG007.ASM
* MDJ 2023/04/17
*
* BRUTE FORCE
* BAG CONTENTS MESSAGE
*
* FUTURE WORK:
* REPLACE WITH
* PMODE 4 FAKE
* STRING OPS
*
* ENTRY CONDITIONS
* NONE
*
* EXIT CONDITIONS:
* NONE
*
*****

PTFCHR  EQU     $5300

        ORG     $7124

MSG001  PSHS    A,B,X

        LDA     #0
        LDB     #14
        LDX     #$0042  "B"
        JSR     PTFCHR
        LDA     #1
        LDB     #14
        LDX     #$0041  "A"
        JSR     PTFCHR
        LDA     #2
        LDB     #14
        LDX     #$0047  "G"
        JSR     PTFCHR
        LDA     #3
```

```
LDB     #14
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #4
LDB     #14
LDX     #$0043  "C"
JSR     PTFCHR
LDA     #5
LDB     #14
LDX     #$004F  "O"
JSR     PTFCHR
LDA     #6
LDB     #14
LDX     #$004E  "N"
JSR     PTFCHR
LDA     #7
LDB     #14
LDX     #$0054 "T"
JSR     PTFCHR
LDA     #8
LDB     #14
LDX     #$0045  "E"
JSR     PTFCHR
LDA     #9
LDB     #14
LDX     #$004E  "N"
JSR     PTFCHR
LDA     #10
LDB     #14
LDX     #$0054  "T"
JSR     PTFCHR
LDA     #11
LDB     #14
LDX     #$0053  "S"
JSR     PTFCHR
LDA     #12
LDB     #14
LDX     #$003A  ":"
JSR     PTFCHR
LDA     #13
LDB     #14
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #14
LDB     #14
LDX     #$0047  "G"
JSR     PTFCHR
```

```
LDA     #15
LDB     #14
LDX     #$004F  "O"
JSR     PTFCHR
LDA     #16
LDB     #14
LDX     #$0053  "S"
JSR     PTFCHR
LDA     #17
LDB     #14
LDX     #$0050  "P"
JSR     PTFCHR
LDA     #18
LDB     #14
LDX     #$0045  "E"
JSR     PTFCHR
LDA     #19
LDB     #14
LDX     #$004C  "L"
JSR     PTFCHR
LDA     #20
LDB     #14
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #21
LDB     #14
LDX     #$004F  "O"
JSR     PTFCHR
LDA     #22
LDB     #14
LDX     #$0046  "F"
JSR     PTFCHR
LDA     #23
LDB     #14
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #24
LDB     #14
LDX     #$004A  "J"
JSR     PTFCHR
LDA     #25
LDB     #14
LDX     #$004F  "O"
JSR     PTFCHR
LDA     #26
LDB     #14
LDX     #$0048  "H"
```

```
            JSR     PTFCHR
            LDA     #27
            LDB     #14
            LDX     #$004E  "N"
            JSR     PTFCHR
            LDA     #28
            LDB     #14
            LDX     #$002E  "."
            JSR     PTFCHR

            PULS    A,B,X
    ENDCHK  RTS

            END
```

=====

MSG008: “The Warehouse is Empty.”

The Assembly Language text listing:

```
*****
*
* MSG008.ASM
* MDJ 2023/04/17
*
* BRUTE FORCE
* EMPTY WHSE MESSAGE
*
* FUTURE WORK:
* REPLACE WITH
* PMODE 4 FAKE
* STRING OPS
*
* ENTRY CONDITIONS
* NONE
*
* EXIT CONDITIONS:
* NONE
*
*****

PTFCHR  EQU     $5300

        ORG     $724B

MSG001  PSHS    A,B,X

        LDA     #0
        LDB     #14
        LDX     #$0054  "T"
        JSR     PTFCHR
        LDA     #1
        LDB     #14
        LDX     #$0048  "H"
        JSR     PTFCHR
        LDA     #2
        LDB     #14
        LDX     #$0045  "E"
        JSR     PTFCHR
        LDA     #3
        LDB     #14
        LDX     #$0020  SPACE
```

```
JSR     PTFCHR
LDA     #4
LDB     #14
LDX     #$0057  "W"
JSR     PTFCHR
LDA     #5
LDB     #14
LDX     #$0041  "A"
JSR     PTFCHR
LDA     #6
LDB     #14
LDX     #$0052  "R
JSR     PTFCHR
LDA     #7
LDB     #14
LDX     #$0045  "E"
JSR     PTFCHR
LDA     #8
LDB     #14
LDX     #$0048  "H"
JSR     PTFCHR
LDA     #9
LDB     #14
LDX     #$004F  "O"
JSR     PTFCHR
LDA     #10
LDB     #14
LDX     #$0055  "U"
JSR     PTFCHR
LDA     #11
LDB     #14
LDX     #$0053  "S"
JSR     PTFCHR
LDA     #12
LDB     #14
LDX     #$0045  "E"
JSR     PTFCHR
LDA     #13
LDB     #14
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #14
LDB     #14
LDX     #$0049  "I"
JSR     PTFCHR
LDA     #15
LDB     #14
```

```
            LDX     #$0053  "S"
            JSR     PTFCHR
            LDA     #16
            LDB     #14
            LDX     #$0020  SPACE
            JSR     PTFCHR
            LDA     #17
            LDB     #14
            LDX     #$0045  "E"
            JSR     PTFCHR
            LDA     #18
            LDB     #14
            LDX     #$004D  "M"
            JSR     PTFCHR
            LDA     #19
            LDB     #14
            LDX     #$0050  "P"
            JSR     PTFCHR
            LDA     #20
            LDB     #14
            LDX     #$0054  "T"
            JSR     PTFCHR
            LDA     #21
            LDB     #14
            LDX     #$0059  "Y"
            JSR     PTFCHR
            LDA     #22
            LDB     #14
            LDX     #$002E  "."
            JSR     PTFCHR

            PULS    A,B,X
ENDCHK      RTS

            END
```

=====

MSG009:
“Whse Inventory: Gospel of John.”

The Assembly Language text listing:

```
*****
*
* MSG009.ASM
* MDJ 2023/04/17
*
* BRUTE FORCE
* WHSE INVENTORY MESSAGE
*
* FUTURE WORK:
* REPLACE WITH
* PMODE 4 FAKE
* STRING OPS
*
* ENTRY CONDITIONS
* NONE
*
* EXIT CONDITIONS:
* NONE
*
*****

PTFCHR  EQU     $5300

        ORG     $7336

MSG001  PSHS    A,B,X

        LDA     #0
        LDB     #14
        LDX     #$0057  "W"
        JSR     PTFCHR
        LDA     #1
        LDB     #14
        LDX     #$0048  "H"
        JSR     PTFCHR
        LDA     #2
        LDB     #14
        LDX     #$0053  "S"
        JSR     PTFCHR
        LDA     #3
```

```
LDB     #14
LDX     #$0045  "E"
JSR     PTFCHR
LDA     #4
LDB     #14
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #5
LDB     #14
LDX     #$0049  "I"
JSR     PTFCHR
LDA     #6
LDB     #14
LDX     #$004E  "N"
JSR     PTFCHR
LDA     #7
LDB     #14
LDX     #$0056  "V"
JSR     PTFCHR
LDA     #8
LDB     #14
LDX     #$0045  "E"
JSR     PTFCHR
LDA     #9
LDB     #14
LDX     #$004E  "N"
JSR     PTFCHR
LDA     #10
LDB     #14
LDX     #$0054  "T"
JSR     PTFCHR
LDA     #11
LDB     #14
LDX     #$004F  "O"
JSR     PTFCHR
LDA     #12
LDB     #14
LDX     #$0052  "R"
JSR     PTFCHR
LDA     #13
LDB     #14
LDX     #$0059  "Y"
JSR     PTFCHR
LDA     #14
LDB     #14
LDX     #$003A  ":"
JSR     PTFCHR
```

```
LDA     #15
LDB     #14
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #16
LDB     #14
LDX     #$0047  "G"
JSR     PTFCHR
LDA     #17
LDB     #14
LDX     #$004F  "O"
JSR     PTFCHR
LDA     #18
LDB     #14
LDX     #$0053  "S"
JSR     PTFCHR
LDA     #19
LDB     #14
LDX     #$0050  "P"
JSR     PTFCHR
LDA     #20
LDB     #14
LDX     #$0045  "E"
JSR     PTFCHR
LDA     #21
LDB     #14
LDX     #$004C  "L"
JSR     PTFCHR
LDA     #22
LDB     #14
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #23
LDB     #14
LDX     #$004F  "O"
JSR     PTFCHR
LDA     #24
LDB     #14
LDX     #$0046  "F"
JSR     PTFCHR
LDA     #25
LDB     #14
LDX     #$0020  SPACE
JSR     PTFCHR
LDA     #26
LDB     #14
LDX     #$004A  "J"
```

```
            JSR     PTFCHR
            LDA     #27
            LDB     #14
            LDX     #$004F  "O"
            JSR     PTFCHR
            LDA     #28
            LDB     #14
            LDX     #$0048  "H"
            JSR     PTFCHR
            LDA     #29
            LDB     #14
            LDX     #$004E  "N"
            JSR     PTFCHR
            LDA     #30
            LDB     #14
            LDX     #$002E  "."
            JSR     PTFCHR

            PULS    A,B,X
    ENDCHK  RTS

            END
```

=====

GMOVER: Game Over - Quest Complete

The Assembly Language text listing:

```
*****
*
* GMOVER.ASM
* MDJ 2023/04/16
*
* GAME OVER
* QUEST COMPLETE
*
*****

GMOK    EQU     $54A0   GAME STATUS FLAG
MSG003  EQU     $7CD8   QUEST COMPLETE MSG

        ORG     $70F2

GMOVER  PSHS    A

        LDA     #0      GAME OVER CODE
        STA     GMOK    PUT TO STATUS FLAG
        JSR     MSG003  DISPLAY MESSAGE

* HOLD THE SCREEN
LBL001  BRA     LBL001

        PULS    A
ENDCHK  RTS

        END
```

=====

GMOVED: Game Over - You Died

The Assembly Language text listing:

```
*****
*
* GMOVED.ASM
* MDJ 2023/04/16
*
* GAME OVER
* YOU DIED
*
*****

GMOK    EQU     $54A0   GAME STATUS FLAG
MSG002  EQU     $7BD9   YOU DIED MSG

        ORG     $7101

GMOVED  PSHS    A

        LDA     #0      GAME OVER CODE
        STA     GMOK    PUT TO STATUS FLAG
        JSR     MSG002  DISPLAY MESSAGE

* HOLD THE SCREEN
LBL001  BRA     LBL001

        PULS    A
ENDCHK  RTS

        END
```

=====

GMLOOP: The Game Loop

The Assembly Language text listing:

```
*****
*
* GMLOOP.ASM
* MDJ 2023/04/14
*
* GAME LOOP
*
*****

* MLF POLCAT
POLCAT  EQU     $4142

* KEY EVENT HANDLERS
EASTK   EQU     $5A00
WESTK   EQU     $5B00
NORTHK  EQU     $5C00
SOUTHK  EQU     $5D00
TCHARK  EQU     $5E00
LCHARK  EQU     $5F00
BCHARK  EQU     $6000
ICHARK  EQU     $6100
UCHARK  EQU     $6200
DCHARK  EQU     $6300
PCHARK  EQU     $6400
RCHARK  EQU     $6500
XCHARK  EQU     $6600
CCHARK  EQU     $6700
GCHARK  EQU     $6800

        ORG     $6900

GMLOOP  PSHS    A

* GET A KEYPRESS
LBL001  JSR     POLCAT
        BEQ     LBL001

* EASTK (RIGHT ARROW)
        CMPA    #$09
        BNE     LBL002
        JSR     EASTK
        BRA     LBL001
```

```
* WESTK (LEFT ARROW)
LBL002  CMPA    #$08
        BNE     LBL003
        JSR     WESTK
        BRA     LBL001

* NORTHK (UP ARROW)
LBL003  CMPA    #$5E
        BNE     LBL004
        JSR     NORTHK
        BRA     LBL001

* SOUTHK (DOWN ARROW)
LBL004  CMPA    #$0A
        BNE     LBL005
        JSR     SOUTHK
        BRA     LBL001

* TCHARK (T-KEY = TAKE)
LBL005  CMPA    #$54
        BNE     LBL006
        JSR     TCHARK
        BRA     LBL001

* LCHARK (L-KEY = LEAVE)
LBL006  CMPA    #$4C
        BNE     LBL007
        JSR     LCHARK
        BRA     LBL001

* BCHARK (B-KEY = BAG INVENTORY)
LBL007  CMPA    #$42
        BNE     LBL008
        JSR     BCHARK
        BRA     LBL001

* ICHARK (I-KEY = WAREHOUSE INVENTORY)
LBL008  CMPA    #$49
        BNE     LBL009
        JSR     ICHARK
        BRA     LBL001

* UCHARK (U-KEY = UP TO NEXT LEVEL ABOVE)
LBL009  CMPA    #$55
        BNE     LBL010
        JSR     UCHARK
```

```
        BRA     LBL001

* DCHARK (D-KEY = DOWN TO NEXT LEVEL BELOW)
LBL010  CMPA    #$44
        BNE     LBL011
        JSR     DCHARK
        BRA     LBL001

* PCHARK (P-KEY = PAUSE GAME)
LBL011  CMPA    #$50
        BNE     LBL012
        JSR     PCHARK
        BRA     LBL001

* RCHARK (R-KEY = RESUME GAME)
LBL012  CMPA    #$50
        BNE     LBL013
        JSR     RCHARK
        BRA     LBL001

* XCHARK (X-KEY = EXIT GAME)
LBL013  CMPA    #$58
        BNE     LBL014
        JSR     XCHARK
        BRA     GMEXIT

* CCHARK (C-KEY = CONFIRM?)
LBL014  CMPA    #$43
        BNE     LBL015
        JSR     CCHARK
        LBRA    LBL001

* GCHARK (G-KEY = NEW GAME)
LBL015  CMPA    #$47
        BNE     LBL016
        JSR     GCHARK
        LBRA    LBL001

* ANY OTHER KEYPRESS
LBL016  LBRA    LBL001

GMEXIT  PULS    A
ENDCHK  RTS

        END
```

=====

SMGAME: Displays the Maze and Starts the Game

The Assembly Language text listing:

```
*****
*
* SMGAME.ASM
* MDJ 2023/04/15
*
* SCREEN MAZE GAME
* ASSEMBLY ROUTINE
*
* DISPLAYS THE MAZE
* ON SCREEN, USING THE
* FAKETEXT 32 X 16
* CHARACTER SET FOR
* PMODE 4, AND THEN
* STARTS THE GAME.
*
*****

POLCAT  EQU     $4142   GET A KEYPRESS
RSEED   EQU     $43D7   SET RANDOM SEED
PTFCHR  EQU     $5300   FAKE TEXT ROUTINE
LINE00  EQU     $5500   START OF BUFFERS
MCSCCV  EQU     $539C   COORDINATE CONVERTER
STAVTR  EQU     $53A1   AVATAR INITIAL SETUP
GTFVAL  EQU     $5482   GET CELL CONTENTS
PTFVAL  EQU     $548C   PUT CELL CONTENTS
STRNTH  EQU     $549C   STRENGTH SYSTEM VARIABLE
SCORE   EQU     $549E   SCORE SYSTEM VARIABLE
GMOK    EQU     $54A0   GAME STATUS FLAG
BAG     EQU     $54A1   BAG CONTENTS
WHSE    EQU     $54A2   WAREHOUSE CONTENTS
DOCVAL  EQU     $54A3   DOCUMENT VALUE
PROVAL  EQU     $54A4   PROVISIONS VALUE
GMLOOP  EQU     $6900   GAME LOOP
RPTSTR  EQU     $705A   REPORT STRENGTH
RPTSCO  EQU     $70A6   REPORT SCORE
RANDOM  EQU     $7110   RANDOM NUMBER GENERATOR

        ORG     $6996

SMGAME  JMP     LBL001
```

```
XCOORD  RMB     1
YCOORD  RMB     1
CHRCOD  RMB     2
RANGE   RMB     1

LBL001  PSHS    A,B,X,Y,U

* INITIALIZE SYSTEM VARIABLES
        JSR     RSEED   SET RANDOM SEED
        LDX     #100
        STX     STRNTH  INITIAL STRENGTH
        LDX     #0
        STX     SCORE   INITIAL SCORE
        LDA     #1      GAME IS RUNNING
        STA     GMOK    GAME STATUS FLAG
        LDA     #$20    "EMPTY" CODE
        STA     BAG     BAG CONTENTS
        STA     WHSE    WAREHOUSE CONTENTS
        LDD     #5      MAXIMUM FOR RANDOM
        JSR     RANDOM  RANDOM # GENERATOR
        LDA     #21     # OF JOHN CHAPTERS
        MUL
        ADDD    #105    MINIMUM DOCVAL
        STB     DOCVAL  DOCUMENT VALUE
        LDD     #50     MAXIMUM FOR RANDOM
        JSR     RANDOM  RANDOM # GENERATOR
        ADDD    #25     MINIMUM PROVAL
        STB     PROVAL  PROVISIONS VALUE

* INITIALIZE THE DOCUMENT LOCATION
        LDX     #$00E0  CHRCOD FOR JOHN
        STX     CHRCOD
        LDD     #5      MAXIMUM MY
        JSR     RANDOM  RANDOM # GENERATOR
        STB     YCOORD  MY-COORDINATE
        CMPB    #0      IS IT FIRST ROW?
        BEQ     LBL005  GO IF YES
        CMPB    #5      IS IT LAST ROW?
        BEQ     LBL006  GO IF YES
        LDD     #14     RANGE FOR ROWS 1-4
        JSR     RANDOM  RANDOM # GENERATOR
        BRA     LBL007
LBL005  LDD     #9      RANGE FOR ROW 0
        JSR     RANDOM  RANDOM # GENERATOR
        ADDD    #5      OFFSET FOR ROW 0
        BRA     LBL007
```

```
LBL006  LDD     #9      RANGE FOR ROW 5
        JSR     RANDOM  RANDOM # GENERATOR
LBL007  STB     XCOORD  MX-COORDINATE
        LDA     XCOORD  MX-COORDINATE
        LDB     YCOORD  MY-COORDINATE
        JSR     MCSCCV  CONVERT TO (SX,SY)
        LDX     CHRCOD  CHARACTER CODE
        JSR     PTFVAL  PUT TO SCREEN BUFFER

* INITIALIZE THE PROVISIONS LOCATION
        LDX     #$0085  CHRCOD FOR PROVISIONS
        STX     CHRCOD
        LDD     #5      MAXIMUM MY
        JSR     RANDOM  RANDOM # GENERATOR
        STB     YCOORD  MY-COORDINATE
        CMPB    #0      IS IT FIRST ROW?
        BEQ     LBL008  GO IF YES
        CMPB    #5      IS IT LAST ROW?
        BEQ     LBL009  GO IF YES
        LDD     #14     RANDOM RANGE ROWS 1-4
        JSR     RANDOM  RANDOM # GENERATOR
        BRA     LBL010
LBL008  LDD     #9      RANDOM RANGE ROW 0
        JSR     RANDOM  RANDOM # GENERATOR
        ADDD    #5      OFFSET FOR ROW 0
        BRA     LBL010
LBL009  LDD     #9      RANDOM RANGE ROW 5
        JSR     RANDOM  RANDOM # GENERATOR
LBL010  STB     XCOORD  MX-COORDINATE
LBL011  LDA     XCOORD  MX-COORDINATE
        LDB     YCOORD  MY-COORDINATE
        JSR     MCSCCV  CONVERT TO (SX,SY)
        JSR     GTFVAL  GET CURRENT CONTENTS
        CMPB    #$20    IS IT A BLANK SPACE?
        BEQ     LBL015  GO IF YES
        LDA     XCOORD  MX-COORDINATE
        LDB     YCOORD  MY-COORDINATE
        INCA            INCREMENT MX
        STA     XCOORD  SAVE IT
        CMPB    #5      IS IT LAST ROW?
        BNE     LBL012  GO IF NO
        LDB     #10     LAST ROW RANGE
        STB     RANGE
        BRA     LBL013
LBL012  LDB     #15     OTHER ROWS RANGE
LBL013  CMPA    RANGE   END OF ROW?
        BNE     LBL011  CHECK NEXT CELL IF NO
```

```
        LDB     YCOORD  MY-COORDINATE
        INCB            INCREMENT MY
        STB     YCOORD  SAVE IT
        CMPB    #6      END OF SCREEN
        BEQ     LBL014  GO IF YES
        LDA     #0      MX-COORDINATE
        STA     XCOORD  SAVE IT
        BRA     LBL011  GO CHECK NEXT CELL
LBL014  LDA     #5      MX-COORDINATE
        STA     XCOORD  SAVE IT
        CLRB            MY-COORDINATE
        STB     YCOORD  SAVE IT (=0)
        BRA     LBL011  GO CHECK NEXT CELL
LBL015  LDA     XCOORD  MX-COORDINATE
        LDB     YCOORD  MY-COORDINATE
        JSR     MCSCCV  CONVERT TO (SX,SY)
        LDX     CHRCOD  CHARACTER CODE
        JSR     PTFVAL  PUT TO SCREEN BUFFER

* INITIALIZE THE SCREEN
        LDY     #LINE00 POINT TO BUFFERS

        LDA     #$FF    SET FIRST XCOORD TO ROLL
        STA     XCOORD
        LDB     #$00    SET FIRST YCOORD TO ZERO
        STB     YCOORD

LBL002  LDA     XCOORD
        LDB     YCOORD
        INCA            INCREMENT XCOORD
        STA     XCOORD
        STB     YCOORD
        CMPA    #32     END OF THE X LINE?
        BLO     LBL003  GO IF NO
        CLRA            SET XCOORD = 0
        STA     XCOORD
        INCB            INCREMENT YCOORD
        STB     YCOORD
        CMPB    #16     END OF SCREEN?
        BLO     LBL003  GO IF NO
        BRA     LBL004  GO IF YES

LBL003  LDA     XCOORD
        LDB     YCOORD
        JSR     GTFVAL  GET CELL CONTENTS
        CLRA            EXTEND IT
        TFR     D,X     MOVE IT TO REG X
```

```
            STX     CHRCOD  SAVE IT

            PSHS    A,B,X   PUT CHRCOD TO SCREEN
            LDA     XCOORD
            LDB     YCOORD
            LDX     CHRCOD
            JSR     PTFCHR
            PULS    A,B,X
            LBRA    LBL002  RETURN FOR NEXT CHRCOD

    * INITIAL AVATAR SETUP
    LBL004  JSR     STAVTR

    * REPORT CURRENT STRENGTH
            JSR     RPTSTR

    * REPORT CURRENT SCORE
            JSR     RPTSCO

    * GAME LOOP
            JSR     GMLOOP

    * HOLD THE SCREEN
    LBL016  BRA     LBL016

            PULS    A,B,X,Y,U
    ENDCHK  RTS

            END
```

=====

MALKYS.BAS: Sets General Parameters, enters ALLRAM Mode, and then Executes the SMGAME Routine

The BASIC Language program listing:

```
1000 '*****
1010 '*
1020 '* MALKYS.BAS
1030 '* MDJ 2023/04/19
1040 '*
1050 '* MALKY'S WARREN: THE
1060 '* FIRST TRAINING QUEST
1070 '*
1080 '* SCREEN MAZE GAME
1090 '* BASIC PROGRAM
1100 '*
1110 '* DISPLAYS THE MAZE
1120 '* ON SCREEN, USING THE
1130 '* FAKETEXT 32 X 16
1140 '* CHARACTER SET FOR
1150 '* PMODE 4, AND THEN
1160 '* STARTS THE GAME.
1170 '*
1180 '*****
1190 '

1500 PRINT
1510 PRINT "WORKING ***";

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

4000 LOADM "MALKYS.BIN"
4010 PRINT "***";
4020 '

7000 EXEC &H536F  'SMREAD
7010 PRINT "***"
7020 '
```

```
9500 'SETUP GRAPHICS
9510 PMODE 4,1
9520 PCLS 1
9530 SCREEN 1,0
9540 '

9600 'SMGAME.ASM RUN ADDRESS = &H6996
9610 'PUT IT TO THE ML FOUNDATION'S
9620 'REGPC (AT $H400A)
9630 POKE &H400A, &H69
9640 POKE &H400B, &H96
9650 'GO START THE GAME IN ALLRAM MODE
9660 EXEC &H4403  'STRTUP
9670 '

32767 END
```

=====

Results

Well, the Warren works. Subject to the limitations of the “CoCo 3 only” problem and the “ X “Key bug, the game is playable and does indeed serve as the Proof-of-Concept it was primarily intended to be.

=====

Conclusions and Future Work

In addition to correction of the “CoCo 3 only” problem and the “ X “ Key bug, the following are areas to be addressed in future work:

1. Minor Bugs, as noted herein.

2. A re-arrangement and re-organization of the code, so as to ever hereafter avoid the stigma of the “Spaghetti Dave” moniker.

3. All messages and reporting during the game are now performed by Brute Force, i.e., by putting individual characters to individual positions on the screen. I need to find the time to develop a proper “Put a String to the Screen” function for the Fake Text system. The time to do that was not available prior to CoCoFest.

4. DECMAL is also a bit Brute Force-ish. I may want to investigate the possibility of doing something more elegant. However, I suspect that the Brute Force method may actually be the most efficient here.

5. The False Disk System should yield itself to the creation of mazes with many Levels and Sections; and the Fake Text System was specifically designed to provide significant possibilities beyond the simple One Level, One Section scheme of Malky’s Warren. Many mazes await to be designed. Perhaps partially random maze generation may be considered.

=====

Appendix A
Decimal to Hexadecimal Conversions

DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX
000	00	032	20	064	40	096	60
001	01	033	21	065	41	097	61
002	02	034	22	066	42	098	62
003	03	035	23	067	43	099	63
004	04	036	24	068	44	100	64
005	05	037	25	069	45	101	65
006	06	038	26	070	46	102	66
007	07	039	27	071	47	103	67
008	08	040	28	072	48	104	68
009	09	041	29	073	49	105	69
010	0A	042	2A	074	4A	106	6A
011	0B	043	2B	075	4B	107	6B
012	0C	044	2C	076	4C	108	6C
013	0D	045	2D	077	4D	109	6D
014	0E	046	2E	078	4E	110	6E
015	0F	047	2F	079	4F	111	6F
016	10	048	30	080	50	112	70
017	11	049	31	081	51	113	71
018	12	050	32	082	52	114	72
019	13	051	33	083	53	115	73
020	14	052	34	084	54	116	74
021	15	053	35	085	55	117	75
022	16	054	36	086	56	118	76
023	17	055	37	087	57	119	77
024	18	056	38	088	58	120	78
025	19	057	39	089	59	121	79
026	1A	058	3A	090	5A	122	7A
027	1B	059	3B	091	5B	123	7B
028	1C	060	3C	092	5C	124	7C
029	1D	061	3D	093	5D	125	7D
030	1E	062	3E	094	5E	126	7E
031	1F	063	3F	095	5F	127	7F

DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX
128	80	160	A0	192	C0	224	E0
129	81	161	A1	193	C1	225	E1
130	82	162	A2	194	C2	226	E2
131	83	163	A3	195	C3	227	E3
132	84	164	A4	196	C4	228	E4
133	85	165	A5	197	C5	229	E5
134	86	166	A6	198	C6	230	E6
135	87	167	A7	199	C7	231	E7
136	88	168	A8	200	C8	232	E8
137	89	169	A9	201	C9	233	E9
138	8A	170	AA	202	CA	234	EA
139	8B	171	AB	203	CB	235	EB
140	8C	172	AC	204	CC	236	EC
141	8D	173	AD	205	CD	237	ED
142	8E	174	AE	206	CE	238	EE
143	8F	175	AF	207	CF	239	EF
144	90	176	B0	208	D0	240	F0
145	91	177	B1	209	D1	241	F1
146	92	178	B2	210	D2	242	F2
147	93	179	B3	211	D3	243	F3
148	94	180	B4	212	D4	244	F4
149	95	181	B5	213	D5	245	F5
150	96	182	B6	214	D6	246	F6
151	97	183	B7	215	D7	247	F7
152	98	184	B8	216	D8	248	F8
153	99	185	B9	217	D9	249	F9
154	9A	186	BA	218	DA	250	FA
155	9B	187	BB	219	DB	251	FB
156	9C	188	BC	220	DC	252	FC
157	9D	189	BD	221	DD	253	FD
158	9E	190	BE	222	DE	254	FE
159	9F	191	BF	223	DF	255	FF

=====

Appendix B: My CoCo Philosophy

The CoCo community enjoys a great diversity of interests.

Some choose to concentrate on hardware innovations and modifications such as interfacing with VGA and HDMI monitors, SD Card data storage, and 104-key keyboards. This interest is at least partly born of necessity, since composite monitors, floppy diskettes, and CoCo spare parts are no longer manufactured and are in increasingly short supply.

Others concentrate on expanding the software horizons of the CoCo 3, using NitrOS-9 and other operating systems to make the multitasking CoCo behave ever closer to modern Windows, Mac, and Linux machines.

Still others are devoted to emulating the CoCo on other platforms by developing emulators such as VCC, OVCC, MAME, and XRoar.

And some just love retro gaming.

My personal interest is twofold:

1. To see VCC increasingly used as a learning tool for budding software developers.

2. To see just how much I can cram into a 64K CoCo 2.

First, VCC: Today's Grade School, Junior High, and High School students have a wealth of available learning tools. Micro-bits, Arduinos, and Raspberry Pi supermicro devices provide highly affordable entry-level introductions to computer programming and interfacing. Maker-Spaces and Innovation Centers in our schools and libraries help foster growth and experience.

But these devices do have limitations. Even these simple(?) computers can have rather steep learning curves, and their low initial cost can quickly expand as new peripherals and experimental equipment and supplies are added.

VCC is free, and can be used on any Windows computer: just download it, install it, and it runs. If you don't own a Windows computer, your school, library, or a friend probably does. The included BASIC language is easy to learn and can readily serve as a stepping-stone towards more complex programming languages. (And, no, learning structured programming does not require a language that enforces structure. In fact, I think learning to structure your programs is actually more effective when you do so on your own.)

I prefer VCC to the other emulators for these purposes because its setup is trivial: Again, just download it, install it, and it runs. OVCC, MAME, and XRoar have their advantages, but ease of setup is not one of them. Even with their available Windows binary packages, they require pre-installation of other bits and pieces of software before they can be downloaded,

installed, and run. This may not be a major problem for a reasonably adept aficionado, but it forms a significant barrier for the newbie. And, it's the newbie whom we're trying to reach, interest, and encourage here; the newbie who may not yet recognize even the tiniest awakening of interest in things computational.

But, for these purposes, VCC has one glaring weakness: its instruction manual is woefully terse. I would like to see VCC bundled with a selection of tutorials, manuals, and examples suited to guiding even the most newbie of newbies into the wonders of computing.

Second, The Stuffed CoCo: I'm simply fascinated by the challenge of seeing how much functional capability I can sandwich into the nooks and crannies of the 64K space. Whether it's working in the available RAM left by the 32K ROM and the dedicated RAM that supports that ROM, or whether it's jumping right into ALLRAM mode and just filling the entire 64K to near-overflowing; it's an investigative gauntlet which goes right to the heart of my enchantment with puzzles in general.

It's great fun!

M.D.J. 2021/08/29

=====

Appendix C: New BDS Software License

This New Software License applies to all software found on the BDS Software site, and supersedes all previous copyright notices and licensing provisions which may appear in the software itself or in any documentation therefor.

All software which has previously been placed in the public domain remains in the public domain.

All other software, programs, experiments and reports, documentation, and any other material on this site (other than that attributed to outside sources) is hereby copyright © 2018 (or later if so marked) by M. David Johnson.

All software, documentation, and other information on the BDS Software site is available for you to freely download without cost.

Whether you downloaded such items directly from this site, or you obtained them by any other means, you are hereby licensed to copy them, to sell or give away such copies, to use them, and to excerpt from them, in any way whatsoever, so long as nothing you do with them would denigrate the name of our Lord and Savior, Jesus Christ.

I make absolutely no warranty whatsoever for any of these items. You use them entirely at your own risk.

If they don't work for you, I commiserate.

If they crash your system, I sympathize.

But I accept no responsibility whatsoever for any such consequences. Under no circumstances will BDS Software or M. David Johnson be liable for any negative results of any kind which you may experience from downloading or using these items.

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

Wikipedia “Diocletianic Persecution”.
https://en.wikipedia.org/wiki/Diocletianic_Persecution . 2023. Online.

=====