See “How to create DOSBOOT” in Google documents.

1/1/2022:

* Backspace event sequence:
  + PutIntoBuffer #8 (by FormKeyPress)
  + TakeFromBuffer #8
  + READKEY0 #8
  + READKEY #8
  + (should be all processed at this point?)
  + PutPrefixedKey(kf\_BackSpace)
    - (cur character has been erased)
* Does the compiler itself need to be re-compiled to get the floating point constants correct?
* The compiler calls WRITELINKERINFO.
  + FREADREA specifies 2 params
  + FWRITERE specifies 5 params
  + DECOPS specifies 0 parameters
* Use LIBMAP to display linker info.
* See “library map of DTEST.TXT”
* Why is there a reference to “L”? This is a “global” variable that can be accessed by external code.Likewise, changing to a real variable “X”, it is also added to the linker information/
* DECOPS:
  + Extproc = 9?
  + Pnumoff = 2
  + NParams = 0?
* Does the compiler know about SEPARATE or INTRINSIC? It does know about SEPARATE.
* SYSCOMP is set on for {$U-}
* Do I have a newer version of the compiler somewhere? Later than 11/16/1978. Maybe: F:\NDAS-I\d7\Projects\pSystem\Volumes\UCSDII.VOL.
* No. That is version II compiler which is “word addressed”.

1/3/2022:

* LIBMAP
  + Read the data dictionary
  + “Not a code file” if codeaddr or codeleng is negative or segname is not alphabetic (OR CODEADDR > 300).
  + Process SEGTBL.SEGKINDS
  + GETENTRY starts reading at NEXTBLK. NEXTBLK is the block *after* the CODEADDR.
* UIOERR - goes into an infinite loop once hit
* Compiler error:(IORESULT = 8→ “No Room on Volume”) 'Dynamic CallStack: PASCALCO.PRINTLIN @ 38, PASCALCO.CHECKEND @ 71, PASCALCO.INSYMBOL @ 535, PASCALCO.BLOCK @ 301, PASCALCO.PASCALCO @ 80, COMPILER.PASCALCO @ 6, PASCALSY. @ 101, PASCALSY.PASCALCO @ 8'
* Is DTEST an “external” proc or a “separate” proc?
* LINK3A.TEXT contains RESOLVE.
* Proc DECOPS param mismatch
* defsym^.entry.nparams <> nparams: 8, 0

1/4/2022:

* Save named breakpoints
* Lientry:
  + Words @ $06C0 [ 1728]: 4544 4F43 5350 2020 0009 0002 0000 0000 0000 0000 0000 01C8 0000 0000 06F0 5244 5741 494C 454E 000B 0001 0008 06D2 0000 0000
  + lientry @ $06C0 [ 1728]: name=DECOPS , litype=9 {EXTPROC?}, srcproc=0, nparams=2, place=0
* I need to update LINK3A source in the database (or just RESOLVE)
* I am getting a range check error 'Static CallStack: PASCALSY.FPUT @ 299, PASCALSY.VARIABLE @ 51' while compiling the linker. Segment #0, P#8, RefIPC:299-- probably occurring on this

IF (FNXTBYTE >= FBLKSIZE-127) AND NOT ODD(FNXTBLK) THEN

Statement in PASCALSY.FPUT.

Doing a NOT on the number 3 gives 2? I think that this was a bug in LNOT.

* Where does the window position get saved? FormCloseQuery.
* The problem is a bad value for: defsym^.entry.nparams. Where does it get set?

1/5/2022:

* Defsym = 1752, littype = 9 (EXTPROC) coming from (or is is 1752+18
* Is symp somehow getting *place* (8) mixed up with *nparams* (0)?
* Check out the source/p-Code for LINKER.LINKER. The offsets do not make sense.

1/6/2022:

* Code is indexing 9 words from the start of lientry. Code ought to be looking 6 words from the start.
* Location is written to by UNITREAD. Also written to by READLINKINFO @ line 293 (syp^.entry := entry).
* When the entry is read from the disk, it is stored at $B074 and this is the value: lientry @ $B074 [45172]: name=DECOPS , litype=9, nparams=0, srcproc=2, place=0. Which is correct! (I think)
* This is loaded for DECOPS:
  + lientry @ $B074 [45172]: name=DECOPS , litype=SEPPROC, nparams=0, srcproc=1, place=0
* This is loaded for PASCALIO:
  + lientry @ $B074 [45172]: name=PASCALIO, litype=PUBLREF, format=0, nrefs=1, nwords=0, reflist=0
* After line 293 in LINKER.READLINK, syp gets set to:
  + Symbol @ $06BA [ 1722]: llink=0, rlink=0, slink=0, name=DECOPS , litype=EXTPROC, nparams=0, srcproc=2, place=0
* The DECOPS lientry appears to be located at $B074
  + lientry @ $B074 [45172]: name=DECOPS , litype=EXTPROC, nparams=0, srcproc=2, place=0
* *(procedure 45 is now LINKSEGMENT)*
* something thinks that the linker has two additional procedures (47 and 48)
* I need to take the debugger out of edit mode when…?
* Where does DEFSYM get initialized? Maybe in sepsrch.
* The address of “inlist^.defsym := syp;” 2270

1/7/2022:

* p-Code in RESOLVE is off by 1 byte
* When I try to run SYSTEM.WRK.CODE (which has linked successfully?) I get a crash (“I/O error 105: Segment 1: ProcNum:13 @ #Unknown, Ofs:13 (maybe because DECOPS is not implemented).
* Code to write reals seems to be working
* Even after I close the frmPsysWindow, something is trying to execute p-Code
* What is closing the window? The ESYSTEMHALT exception.
* Try to compile WHETSTONE (can compile but not run)
* Can compile DTEST but get I/O error 105 when run @ S#1, ProcNum:1, O#13
* Does DECOPS get copied into WHETSTONE? iF NOT WORK ON whetstone.
* Why does SEARCH require DECOPS? Fix it.

1/8/2022:

* CLOSEPRINT to close the printer file.
* Whetstone compiles and starts to run but gets an AV. Whetstone wants DECOPS? Where is the reference coming from? I don’t see it in the decoded or in the LIBMAP.
* Search wants DECOPS and refers to it as P#13.
* When dis-assembling SEARCH.CODE and trying to list opcode statistics causes memory corruption and a crash.
* SEARCH references DECOPS via its use of the STR function. I can substitute my own STR function.
* I’m getting a very long hang that breaks in GetSegStuff. Things never return to the p-System.
* Speed is being adversely affected by all the calls to OutPutDebugStrFmt.
* After SEARCH.CODE finishes, I get an “Unknown SEGTOP $DAF8 (EXEQERR)” message which appears to be coming from DECREF. Search never finishes with my new STR function.
* And it seems like keyboard is echoing.
* I am getting lots of garbage in fFilesLoadedList[fn] for various files-- looks like it was never initialized
  + SYSTEM.MICRO is listed as a “code” file
    - it has garbage from the very first time that it is seen
  + SYSTEM.COMPILER seems OK in the dump for about the first 56 (or so) dumps. The everything in SYSTEM.COMPILER gets corrupted (including the starting block)

1/10/2022:

* Duplicate entry in pCodeProcsTable? DB does have some duplicate entries.
* ErrorLogFile “LogFile (2).txt” shows 278,503 entries? File was always being appended to. Never rewritten.
* READSEG did seem to manage to add the linker to the fFilesLoadedList but a bit later it could not be found.
* The crash occurs when it is trying to write X
* Crash occurs when trying to decrement 56060. I just freed it in GetSegStuff because it overlapped with the “old” block.
  + FREADREA external proc P #3, nparams 2, place 0
  + FWRITERE external proc P #10, nparams 5, place 0
* Debugger gets confused about procedure 42: in “COMPILER”. RelIPC = 8341. I’m getting an “EIntOverflow: integer overflow” in segment 0, proc #42. It may be that the compiler is trying to exit. I think that both the segment name (“COMPILER”) and the procedure name (“TERM”) are probably nonsense. Something is trashing memory.
* I tried commenting out the code that “free” old segment info things because it “overlaps” new stuff. This ignores the proper use of REFCNT.
* Takes a LONG time to go from line 905 to 995 in BIGGY. Crashes when it trys to execute LOOP11.
* AddLineSeperator: OpCode is ALWAYS 206? wAS GETTING “OpCode” from the interpreter rather than from the decoder.

1/11/2022:

* Somehow the WHETSTON.WHETSTON variables got stored in the BIGGY variables? Maybe this had something to do with me “Locking” the local variables?
* Crashes in LOOP11 when it tries to write 0.75 via FWRITERE.
* SEARCH.CODE crashes when exiting in DECREF (Can’t find SEGTOP).
* The reference to SegInfoRecP gets set in ReadSeg.
* Getting overwritten in OverrideSysCall
* Initial setting of SEARCH.CODE (FN=6, SegIdx=1, BlockNr=796) occurs in OVERRIDE SYSCALL.
* I can get the same crash after re-compiling PASCALIO and running a trivial test (RTEST.TEXT).
* Executing RTEST gets to the Write statement without invoking the debugger.
  + Could it be related to missing “EXTERNAL UNIT”?

1/12/2022:

* Watch for exact/inexact matches of Segment info.
* Sequence:
  + READSEG loads the segment info and sets TheSegTop. This is quickly followed by DECREF setting it back to 0.
  + Immediately after loading RTEST.CODE, it cannot be found (of course).
* Reminder: PASCALIO seems to want to use DECOPS
* Can I link my DECOPS into system.library?
* DecReF never uses SEGIDX.
* Linker says “bad litype”
* DISASM doesn’t create an output file when writing to a text file. DISASM source is on 15sys2.raw
* After a segment’s ref counts go to zero, why would it be needed?

1/13/2022:

1. Why does PASCALIO need DECOPS?
2. If I removed the dependency, would PASCALIO function?
3. Compare PASCALIO from
   1. Decoded / libmap from original
   2. Compiled by Z80
   3. Compiled by Delphi

* Getting an “odd address” error when I try to run the SEARCH program. (“Unit #”)
* GDIRP = 622 ($26E)
* The stuff currently on or below the SP does seem to match what was popped off? (HL=$E801).
* Where did the $E801 come from?
* 'Dynamic CallStack: PASCALSY.FETCHDIR @ 61'
  + gDirp=622 ($026E)
  + DirEntry @ $E801 [59393]: DFirstBlk=0, DLastBlk=6, DFKind=0, Time=00:00, DVID=1.5SYS2, DeovBlk=6, DNumFiles=23, DLASTBOOT=12/27/2004
  + IF (DFIRSTBLK = 0) AND
  + Execute what file? \*SEARCH
    - SEARCH. V1.18
    - Search for filename only? N
    - Search string: CLOSE(
    - -----> CLOSE(
    - Search string:
    - Search all units: N
    - Unit #11
    - Unit #
  + 59: SLDO9
  + 60: SIND0
  + Is the SYSCOM^ OK? No. I don’t think so. Why did interpreter seem to skip over the 1st instruction of LOD?
  + The pointer to SYSCOM is located at $FC72. (0?)
  + “WITH SYSCOM^ DO” got a bad value for the local/global copy. Something has overwritten it. The change to $7EFC seems to occur at RelIOC = 0 when FETCHDIR is being called on a CBP opcode. BC = 6? One place that this occurs is BLD3. BC updated on the way out. CBPXNL is also changing BC and then restoring it on the way out. Is the SYSCOM ptr located at $FC70 or $FC72? GETIA dosen’t seem to be loading from the correct address. GETIA seems a bit hinky.

1/14/2022:

* I am starting off (GETIA) with LASTMP pointing to an MSCW that has BOTH static and dynamic links set to nil.
* Nothing is breaking on anything in SEARCH?
* Skipping over “Do all text files on each unit?”
* Also skipping over “Listing file (CONSOLE:):”
* The stack appears to be all zero in SEARCH? This occurs following the call to ADJ, SRO.
* Units is set of 0..MAX\_STANDARD\_UNIT. MAX\_STANDARD\_UNIT is now 12 (which should fit into 1 (2?) word). I am setting UNITS to [4,5,9..MAX\_STANDARD\_UNIT] which should require 1 word but there is still stuff getting left on the stack. I am now getting the odd address error again.
* SEARCH is now working but I am still getting the ODD address error. 'Static CallStack: PASCALSY.FRESET @ 10'

Which occurs after after SEARCH finishes but before the command line prompt reappears. LOD has executed 1087 times. The error is occurringin FRESET. The LOD command *never* breaks on HL = 0. See the file: “F:\NDAS-I\d7\Projects\pSystem\Reports\History (4).txt” for the history prior to the crash. FRESET does not seem to have a call stack (although it was probably called by GETCMD). Likely candidates are STM and INT. STM is only called before SEARCH is called.

1/15/2022:

* I also get the AV when I try to let DISASM decode RTEST and display the statistics.
* DiskAddr = 381

1/17/2022:

* Trace through LINKER (or LIBMAP?) to determine why “bad litype” is generated. Can I compile a dummy routine to replace DECOPS?

1/18/2022:

* In PME-debug.txt, native code is called via NATEXTERNAL (I think);
* In my DECOPS, I am putting garbage onto the stack and then taking it off.
* Look at ProcessNative ucsdpsys\_vm\native6502.c
* look at old 1.5 sources

1/19/2022:

* Is their SEG the same as my SEGp?
* EnterIC is the same as my ProcBase (which is the same as their ProcBase)

1/20/2022:

* Is KP (SEGBASE) the same as SEGBOTTOM? I think so!
* Deleting some watch items and then attempting to add another led to AV
* The SYSCOM variable named SEG never seems to change.
* Is LASTMP the same as MP?

1/21/2022:

* Inline? NO. Does not exist.
* SegBottom and KP cannot be the same thing. Whoa? Why not?
* What is Seg? I think it is the same at SEGTOP? Same as SEGp?
* BASE changed to Globals.LowMem.Syscom.STKBASE
* I think the IPCBASE is equivalent to what I called PROCBASE (address of 1st word of procedure).

1/22/2022:

* KP\_TOP 0xfe80 ?
* JTAB location is mucked up-- is it either a local variable (property) or does directly access GLOBALS.LOWMEM.SYSCOM.JTAB

1/24/2022:

* Variable name equivalences (best guess):

|  | Z80 Interpreter | C Interpreter |  |
| --- | --- | --- | --- |
|  | HeapTop | Np |  |
|  | STKBASE | Base |  |
|  | ProcBase | IPCBase |  |
|  | LastMP | MP |  |
|  | SegBottom | KP (SegBase) |  |
|  | SegP | SEG |  |
| HEAP\_BOT | NP |  | 256 |
|  |  | KP |  |

* Is the SegDict that is defined in the interpreter supposed to be the same thing that is defined in SYSCOM?

1/25/2022:

* Where does SegDict[0] get initialized? Maybe in CspLoadSegment?

1/26/2022:

* Why is SP getting set to a low memory address?

1/27/2022:

* I may not have copied the hidden files/folders from the CD ROM
* How do I get BASH to print anything?
  + "$major: " $major "
  + "$minor: " $minor "
  + "$build: " $build "
* generates
  + "5: " 5 The colons (:) were in quotes
  + "10: " 10
  + ": "
* The glibc version: ldd (Debian GLIBC 2.31-13+deb11u2) 2.31

**1/28/2022:**

* Gets lost trying to compile “main.c” in directory “bin2c” on this line:

#include <libexplain/fflush.h>home

**1/29/2022:**

* Gcc compiler searches for include files here:

/usr/include/c++/10

/usr/include/x86\_64-linux-gnu/c++/10

/usr/include/c++/10/backward

/usr/lib/gcc/x86\_64-linux-gnu/10/include

/usr/local/include

/usr/include

* Line 145 in the Makefile
  + What is ($CC)
  + What is $(CPPFLAGS)
  + What is $(CFLAGS)
* The only file in bin2c is main.c
* Bin2c is located in pays, i.e., I am compiling /home/psys/bin2c/main.c
* The compiler is looking for the header files in /home/psys/bin2c/main.c/libexplain/\*.h
* Starts by looking for getopt.h
  + Where is getopt.h located?
* Using this command:
  + find /home -name getopt.h
  + I found this:
    - /home/dand/psys/libexplain-1.4/libexplain/ac/getopt.h
  + When i “make”, I get this error:

In file included from bin2c/main.c:20:

/home/dand/psys/libexplain-1.4/libexplain/ac/getopt.h:28:10: fatal error: libexplain/config.h: No such file or directory

28 | #include <libexplain/config.h>

* Maybe the preprocessor path should be

/home/dand/psys/libexplain/ac. No.

* After re-installing libexplain, getopt.h no longer shows up on a find command (except in trash)

**2/1/2022**

* Trying to get cross compiler to work
* When I try to compile any of the main.cc programs I get this error:
  + main.cc:19:10: fatal error: lib/config.h: No such file or directory
  + 19 | #include <lib/config.h>
  + | ^~~~~~~~~~~~~~
* I appear to have a few copies of config.h but I don’t think any of the are the one being specified
* If I search everything for reference to config.h see c:\temp\?.txt.
* Could the file *config.h.in* have anything to do with creating *config.h*?

**2/3/2022:**

| Suse Linux | dan | rapanui |  |
| --- | --- | --- | --- |
| Suse Linux | Root | killer→Sna99le! | keyring pw: “killer” |

* Mount CD image:

mount /media/dvd /mnt

* cd /media/dvd/
* ./VBoxLinuxAdditions.run
* vi /etc/modprobe.d/10-unsupported-modules
  + allow\_unsupported\_modules 1
  + I don’t think that even exists on my version os suse
* Goals
  + Change the password
  + Get VBox guest additions to work
    - Get the kernel headers installed into suse
  + Get networking to work
  + be able to get workable resolution
    - see: <right-ctrl><home> (switch to/from scaled mode)

**2/8/2022:**

* Could not find the task ‘C/C++:g++ build active file’
* The following command seems to be going in the right direction:
  + gcc -v -I ~/psys/ucsd-psystem-vm-0.11/ ucsdpsys\_vm/main.c
  + This at least find the included files and generates a lot of output
  + See the file: \\xps-8930\NDAS-I\d7\Projects\pSystem\Debian work\First Compile.txt
* make --help
* FetchUB appears to be defined on line 283 the first reference is line 1089, 1091, 1095, 1106, 1108 (which do not generate errors)

**2/9/2022:**

* Exactly which source files am I modifying on VPC?
* Include paths:
  + $(BCB)\Projects\psys\_VM\ucsd-psystem-vm-0.11\lib
  + C:\Program Files\Borland\CBuilder3\Projects\psys\_VM\ucsd-psystem-vm-0.11\ucsdpsys\_vm
* Library paths:
  + $(BCB)\lib
  + c:\Program Files\Borland\CBuilder3\Projects\psys\_VM\ucsd-psystem-vm-0.11\ucsdpsys\_vm\lib
* Project folder:
  + c:\Program Files\Borland\CBuilder3\Projects\psys\_VM\ucsd-psystem-vm-0.11\ucsdpsys\_vm\_psysvm.bpr
* \\VPC-Surfacepro\psysvm
* Cannot find PopInteger. First reference occurs at line 1695 in psysvm.cpp. This is actually defined in stack.h which is included at line 64 in psysvm.cpp. If I change the “#include <stack.h>” to “#include <crapstack.h>” the compiler complains. There are both stack.c and stack.h located in Y:\ucsdpsys\_vm\. If I rename the file from “Y:\ucsdepsys\_vm\stack.h” to “Y:\ucsdepsys\_vm\CRAPstack.h”, the compiler does not complain. If I change BOTH the “#include” and the actual file name to “CRAPstack.h” the compiler does open the file.
* Network drive:
  + \\VPC-SURFACEPRO\pSysVM
* I have mapped this path:
  + \\VPC-SurfacePro\psysvm
  + Y:
* Compiler option to dis(?)able warning about significant digits:
  + -wsigdi

**2/10/2022:**

* This command at least tries to compile psysvm.cpp:link
  + Y:\>bcc32 -Ilib\ ucsdpsys\_vm\psysvm.cpp -o obj\psysvm.o
* This
  + Y:\>bcc32 -Ilib\ lib\progname.c -oY:\ucsdpsys\_vm\obj\progname.o
    - generates and .obj and a .exe and a .tds (progname.exe, progname.obj, progname.tds) in the Y: folder
* and this:
  + Y:\>bcc32 -Ilib\ lib\progname.c -oY:\ucsdpsys\_vm\obj\progname
    - does exactly the same thing
* The above files are ending up in the “.lib” folder
* All of the above sems to ignore anything that I specify in the “-o” parameter.
* ilink32 is the linker`
* What are the instructions that came with the C++Builder that I downloaded?
* I can compile, link and run a trivial program (hello.c)?

**2/11/2022:**

* My source code is here: ~/psys/ucsd-psystem-vm-0.11/ucsdpsys\_vm
* Also here: ~/psys/ucsd-psystem-vm-0.11/lib
* The compiler is “gcc”
* When I execute “configure” this way:
  + ./configure CPPFLAGS=-I/usr/local/include LIBS="-L/usr/local/lib ~/psys/ucsd-psystem-vm-0.11/lib" There is also a YACC (YFLAGS?) setting
  + the parameters at least make it into “config.log”
* and this
  + ./configure CPPFLAGS=-I/usr/local/include LIBS="-L/usr/local/lib"
* kind of looks like it worked.
* This is the first complaint from the compiler about FetchUB (line 1155):
* /home/dand/psys/ucsd-psystem-vm-0.11/ucsdpsys\_vm/main.c:1155: undefined reference to `FetchUB'
* but FetchUB is actually referenced previously:
  + Defined on line 283
  + referenced on line 1089, 1091, 1095, 1106, 1108, 1117, 1128, 1130, 1149, 1151, 1153. None of these are generating a compile error?
* ProcDataSize is defined on line 398
* What does the makefile do
  + Compiles “bin2c” and puts the results in the “bin2c” folder (ok? main.o @ 12:56)
  + builds various manuals and supposedly puts the results intp /etc. I don’t see any evidence of anything having been put there?
  + Build float\_conv and move the output (main.o) to float\_conv/main.o (ok? main.o @ 12:56)
  + Build lib/byte\_sex and move the result to lib/byte\_sex.o (ok? byte\_sex.o @12:56)
  + Build lib/diskio.o (here is a typical “build” command:
  + lib/diskio.o: lib/diskio.c lib/byte\_sex.h lib/diskio.h lib/gcc\_attributes.h lib/memory.h lib/psystem.h
    - $(CC) $(CPPFLAGS) $(CFLAGS) -I. -c lib/diskio.c
    - mv diskio.o lib/diskio.o (ok? 12:56)
  + Build lib/diskio.o
  + Build lib/memory.o
  + Build lib/progname.o
  + Build lib/version.o
  + Build lib/version\_stamp.o
  + Build manuals and put them into the ~/psys/ucsd-psystem-vm-0.11/man/ directory. I was able to display the ucsdpsys\_svolio program using the nroff program with this command:
    - **dand@debian:~/psys/ucsd-psystem-vm-0.11/man/man1$ nroff -man ucsdpsys\_svolio**

**2/12/2022:**

* **Here is where the executables (ucsdpsys\_svolcvt ucsdpsys\_svolio) go: /usr/local/bin**
* **Here are the manuals: ~/psys/ucsd-psystem-vm-0.11/man/**
* **Here us the volume copied from XPS-8930: ./psys/ucsd-psystem-vm-0.11/volumes/UCSDI4.VOL**
* **makefile got as far as:**
  + **bin2c.o (main.o)**
  + **float\_conv (main.o)**
  + **lib (byte\_sex.o)**
  + **lib (diskio.o)**
  + **lib (memory.o)**
  + **lib (progname.o)**
  + **lib (version.o)**
  + **lib (version\_stamp.o)**
  + **man1 (ucsdpsys\_svolcvt.1)**
  + **man1 (ucsdpsys\_svolio.1)**
  + **man1 (ucsdpsys\_vm.1)**
  + **man1 (ucsdpsys\_vm\_license.1)**
  + **man1 (ucsdpsys\_xturtleserver.1)**
  + **script (test\_prelude)**
* **make sure**
  + **PATH=`pwd`/bin:$PATH /bin/sh test/00/t0001a.sh**
  + **test/00/t0001a.sh: 57: ucsdpsys\_compile: not found**
  + **NO RESULT for test of XJP**
  + **make: \*\*\* [Makefile:231: t0001a] Error 2**
  + ***Seems to be expecting a executable named “ucsdpsys\_compile” to exist. Maybe this was supposed to be output from the cross compiler?***
* **I need to build the cross compiler**
  + **boost library**
    - **sudo apt-get install libboost-all-dev**
  + **libexplain**
    - **sudo apt-get update (already installed)**
  + **GNU Groff**
    - **sudo apt-get update**
    - **sudo apt-get -y install groff**
  + **dand@debian:~/psys/ucsd-psystem-xc-0.13$ ./configure CPPFLAGS=-I/usr/local/include LIBS=-L/usr/local/lib**
  + **appears that the cross compiler is being built with C++ whereas the virtual machine was built with “c” compiler.**
  + **g++ -I/usr/local/include -g -O2 -I. -c lib/bitmap/png.cc**
    - **lib/bitmap/png.cc:20:10: fatal error: png.h: No such file or directory (:20 refers to line 20, :10 refers to column 10 in the png.cc file: #include <p). There are two includes referencing png.h.**
    - **20 | #include <png.h>**
    - ***This file does seem to exist? It does exist in lib/bitmap***
    - ***I don’t see any place that PNG\_COLOR\_TYPE\_GRAY (or any of the other PNG\_COLOR\_TYPE) gets defined?***
    - ***Found:***
      * ***anonymous\_name.h***
      * ***bit\_address.h***
      * ***bitmap.cc***
      * ***bitmap/factory.cc***
      * ***bitmap/invert.cc***
      * ***bitmap/png.cc***
    - **I NEED TO RESTORE png.cc and png.h. Maybe there is some other source file that I need. Should define: png\_structp, png\_ptr, png\_info, png\_create\_info\_struct, etc.**
    - **Currently, I am here:~/psys/ucsd-psystem-xc-0.13**
      * **trying to run “make”**
      * **~/psys/ucsd-psystem-xc-0.13/lib/bitmap**

**2/14/2022:**

* **I have two versions of png.h**
  + **/usr/include/png.h (Sept 16, 2020) 14-Sept-2016**
  + **/ucsd-psystem-xc-0.13/lib/png.h 11-Nov-2012**
* **I renamed the files in ucsd-psystem-xc-0.13/lib/bitmap png.cc-saved and png.h-saved**
* **I need to find a copy of png.cc**

**Is png.cc something that is specific to the cross compiler?**

* **All of the following got compiled (see the file GotCompiled.txt in \\xps-8930\NDAS-I\d7\Projects\pSystem\Debian work):**
  + **This takes me to about line 1727 in Makefile**
  + **lib/pascal/grammar.yacc.cc is being created by Makefile**

**2/14/2022:**

* **YFLAGS passed into yacc {here}**
* ./configure CPPFLAGS=-I/usr/locd ..cal/include LIBS="-L/usr/local/lib ~/psys/ucsd-psystem-vm-0.11/lib" YFLAGS=-y
* Configure now barfs at the parameter: ~/psys/ucsd-psystem-xc-0.13/lib
* Latest call to configure:
  + ./configure LIBS="~/psys/ucsd-psystem-xc-0.13/lib"
  + This gets me to
    - lib/pascal/grammar.y:63.22-34: warning: POSIX Yacc does not support string literals [-Wyacc]
    - 63 | %token END\_OF\_FILE 0 "end of file"
  + Once I use the YFLAGS=-y the compiler gets sick and complains about checking whether the C++ compiler works... no
  + configure: error: in `/home/dand/backups/ucsd-psystem-xc-0.13':
  + configure: error: C++ compiler cannot create executables
* After editing grammar.y, I reran “configure” (with no parameters) which takes me back to errors in grammar.y related to yyerror:
  + lib/pascal/grammar.y: In function ‘int pascal\_grammar\_parse()’:
  + lib/pascal/grammar.y:454:61: error: expression cannot be used as a function
  + 454 | { yyerror("that \":=\" should be \"=\""); }
* I also reran configure with YFLAGS=
* Look at ytab.c

**2/15/2022:**

* **I am currently working in this directory: ~/backups/ucsd-psystem-xc-0.13**
* **I replaced AC\_PROG\_YACC with AC\_CHECK\_PROG in configure.ac**
* **Somehow, my changing yyerror → yyerror2 caused pascal\_grammar\_error to get changed to pascal\_grammar\_error2?**
* **After changing yyerror to yyerror99, I now get messages like:**
  + **yyerror99("packed files are not supported")**
  + **error: ‘pascal\_grammar\_error99’ was not declared in this scope; did you mean ‘pascal\_grammar\_error’?**
* **In grammar.yacc.cc (in lib.cc) pascal\_grammar\_error = 256**
* **but in lex.h, it is defined as a procedure:** 
  + void pascal\_grammar\_error(const char \*text);
* **yyerror, when used, becomes “pascal\_grammar\_error”**
* **I changed “pascal\_grammar\_error” to “PASCAL\_GRAMMAR\_ERROR” in grammar.yacc.h**
* **Do I still need to get yyerror defined in grammar.y?**
* **void pascal\_lex\_error(const location &loc, const char \*fmt, ...)**
* **It appears as if pascal\_grammar\_error is a function call, but PASCAL\_GRAMMAR\_ERROR should be an integer constant (256)**
* **I reran ./configure. All of the changes that I made to turn pascal\_grammar\_error into PASCAL\_GRAMMAR\_ERROR (constant 256) have disappeared.**
* **Inputs to lib/pascal/grammar.yacc.cc**
  + **lib/pascal/grammar.yacc.h-- PASCAL\_GRAMMAR\_ERROR got reset to pascal\_grammar\_error?**
  + **lib/pascal/grammar.y**
    - **lib/pascal/grammar.h**
    - **lib/pascal/lex.h**
    - **lib/pascal/grammar.yacc.h (make this read-only)**
* **make clean-obj deletes long\_integer/grammar.yacc.h**
* **The makefile blows up on this line**
  + **sed -e 's/[yY][yY]/pascal\_grammar\_/g' -e \ 's/Y\_TAB\_H/lib\_pascal\_grammar\_YACC\_H/g' y.tab.h > \ lib/pascal/grammar.yay.tab.hls y.tab.\***
  + **This is trying to replace all occurrences of** 
    - **“yy” → “grammar\_” in y.tab.c**
    - **and “<stdio.h>” → “<stdlib.h>**
    - **and to delete all lines containing “<stdio.h>” and <stdlib.h> in the file y.tab.c and output the result to ucsdpsys\_assemble/grammar.yacc.cc**
  + **sed -e 's/[yY][yY]/pascal\_grammar\_/g' \**
    - **-e 's/Y\_TAB\_H/lib\_pascal\_grammar\_YACC\_H/g' y.tab.h > \**
    - **lib/pascal/grammar.yacc.h**
  + **This is trying to replace “yy” → “pascal\_grammar\_”, and**
    - **“Y\_TAB\_H” → “lib\_pascal\_grammar\_YACC-H” in the file y.tab.h**
    - **and output the result to lib/pascal/grammar.yacc.h**
  + **After being processed, y.tab.c, y.tab.h and y.output all get deleted**
  + **y.tab.h is getting created somewhere**
  + **grammar.yacc.cc is created by “yacc”**
  + **Where do y.tab.h and y.tab.cc get created?**
  + **How do I exit a makefile cleanly?**

**2/17/2022:**

* **lib/pascal/grammar.yacc.cc ⇐ lib/pascal/grammar.yacc.h, lib/pascal/grammar.y**
  + **This will generate y.tab.c & y.tab.h, which then:**
* **lib/pascal/grammar.yacc.cc ⇐(sed)== y.tab.c**
* **lib/pascal/grammar.yacc.h ⇐(sed)== y.tab.h**
* **lib/pascal/grammar.y**
  + **y.tab.c, y.tab.h are by products of yacc**
    - **They get operated on by “sed” to create “lib/pascal/grammar.yacc.cc” and “lib/pascal/grammar.yacc.h”**
  + **references to yymsg occur in y.tab.c**
  + **Should “sed” also be working on grammar.y?**
* **I need to remove the “echo” lines from makefile make clean**

**4/12/2022:**

* **I installed Samba using these instructions:**
  + [**https://computingforgeeks.com/how-to-configure-samba-share-on-debian/**](https://computingforgeeks.com/how-to-configure-samba-share-on-debian/)
  + **User = “sambauser” password = “quito”**
* **Also see:**
  + **https://www.oreilly.com/openbook/samba/book/ch04\_06.html**
* **I was able to access XPS-8930 from Debian using this command:**
  + **smbclient '\\xps-8930\ndas-i' -U '**[**dhdorrough2@hotmail.com**](mailto:dhdorrough2@hotmail.com)**' (password “duluth03”)**
* **192.168.1.107 DEBIAN [ DEVELOPMENT ]**
* **I used the “findsmb” command to get the list of servers**
* **I don’t have permission to write into the “public” directory.**

**4/13/2022:**

* **The text editor is gedit**
* **I can create into ‘public’ from XPS-8930 but I cannot make any changes to the file in Debian**
* **The ‘private’ folder is U: on XPS-8930**
* **The ‘public’ folder is W: on XPS-8930**

## **Creating Symlink To a File** [**#**](https://linuxize.com/post/how-to-create-symbolic-links-in-linux-using-the-ln-command/#creating-symlink-to-a-file)

* **Using svolio to get a directory listing:**
  + **bin/ucsdpsys\_svolio -e public/15SYS1.VOL**
  + **bin/ucsdpsys\_svolio -l public/15SYSsvolio1.VOL**
  + **bin/ucsdpsys\_svolio -t public/15SYS1.VOL link0.text**
  + **bin/ucsdpsys\_svolio -V**
  + **looking for ucsdpsys\_mkfs in github**

**4/14/2022:{here}**

* **Here is how I got into the p-System:**
  + **~/bin/ucsdpsys\_vm -w ~/Downloads/system.vol**

**4/15/2022:**

* **ssh-keygen -t rsa**
* **/home/dand/.ssh/id\_rsa**
* **This is how I lost Volumes:**
  + **root@debian:/public# mv Volumes ~/psys\***
* **Possibly here:**
  + **/home/dand/psys/ucsd-psystem-vm-0.11/psys/**
* **I still have a bunch of files here () that I cannot remove**

**4/18/2022:**

1. **It appears that the gcc compiler is doing the compiling**
2. **I think that things are getting compiled with the “-g” debugging option**
3. **The compiler is not finding any of the things in “lib\**
4. **The environment variable LIBS should reference the local libraries**
5. **Did I specify “XXX” somewhere?**
6. **I think that I need to set CFLAGS or CPPFLAGS**

**4/19/2022:**

* **Can I compare DISKIO.C to some other version of DISKIO.C? I did. Nothing useful.**
* **After creating a new copy of ~/bin/ucsdpsys\_vm -w ~/Downloads/system.vol**
  + **./configure**
  + **make**
    - **This did not change (/home/dand/bin/ucsdpsys\_vm) which is still dated Apr 13, 2022**
* **Can I just run the debugger on the already compiled version? I.e.,** 
  + **gdb ~/bin/ucsdpsys\_vm -w ~/Downloads/system.vol (gdb does not like the -w parameter) but starts up anyways**
  + **gdb ~/bin/ucsdpsys\_vm**

**or**

* + **gdb /home/dand/bin/ucsdpsys\_vm**
  + **(gdb) run -w ~/Downloads/system.vol (runs w/o any breakpoint)**

**4/20/2022:**

* **Notes from debugging C system**
  + **ucsdpsys\_vm/main.c 3294**
  + **Np (HEAP\_BOT) starts off at 512**
  + **Kp (KP\_TOP) stars off at 0 and is then set to $fdd6 using the WordIndexed function**
  + **Syscom is set from Kp to $fdd6**
  + **Sp starts out at 0**
  + **After loading the system, Mp = $ed9d**
  + **After loading the system, Sp = $1ff and after WordIndexed, it is still $200**
* **Starting values:**

| **Register Name** | **Meaning** | **Initial Value** |
| --- | --- | --- |
| **Np** | **HEAP\_BOT** | **512** |
| **Kp** | **HEAP\_TOP**  **(Seg bottom?)** | **$fdd6** |
| **Syscom** |  | **$fdd6** |
| **Sp** | **Stack** | **0, $200** |
| **Mp** |  | **$fdd6** |
| **BaseMp** |  | **$ed9d** |

* **Appears to be doing word addressing. Sp is 511**
* **Is the stack located in low memory? Yes. I think so and it grows towards 0.**
* **I am directly editing Makefile.in to try and force in the include directory that I want ("-I/home/dand/psys/ucsd-psystem-vm-0.11")**
* **I then re-run ./configure**
* **Things that got changed**
  + **bin**
  + **bin2c**
  + **config.status**
  + **configure**
  + **install-sh**
  + **Makefile**
  + **Makefile.in**

**4/21/2022:**

* **After restoring the changed files, setting CPPFLAGS=-I/home/dand/psys/ucsd-psystem-vm-0.11, ./configure again and make, I get a bunch of usr/bin/ld: /home/dand/psys/ucsd-psystem-vm-0.11/ucsdpsys\_vm/main.c:1159: undefined reference to `FetchUB'**
* **ld is the linker**
* **LDFLAGS?**

**Things that I think that I did when building bin/ucsdpsys\_vm [20220421 @10:44] (not guaranteed and may not be relevant) {backed up to ESD-USB (J:) - 32 Gb card):**

1. ***REMEMBER THAT THERE IS A CONFIG(?) OPTION TO RESTORE THE ENVIRONMENT.***
2. **LIBS=-L/usr/local/lib**
3. **CPPFLAGS=-I/home/dand/psys/ucsd-psystem-vm-0.11 -I/usr/local/include**
   1. **CPPFLAGS=”-I/home/dand/psys/ucsd-psystem-vm-0.11 -I/usr/local/include”**
4. **./configure**
   1. **./configure CPPFLAGS=-I/usr/local/include LIBS="-L/usr/local/lib"**
5. **commented out Applehack2 in main.c**
6. **make**
7. **bin/ucsdpsys\_vm -w ~/Downloads/system.vol**

**BACKUP COPY**

**Things that I think that I did when building bin/ucsdpsys\_vm [20220421 @10:44] (not guaranteed and may not be relevant) {backed up to ESD-USB (J:) - 32 Gb card):**

1. ***REMEMBER THAT THERE IS A CONFIG(?) OPTION TO RESTORE THE ENVIRONMENT.***
2. **LIBS=-L/usr/local/lib**
3. **CPPFLAGS=-I/home/dand/psys/ucsd-psystem-vm-0.11**
4. **./configure**
5. **commented out Applehack2 in main.c**
6. **make clean**
7. **make**
8. **bin/ucsdpsys\_vm -w ~/Downloads/system.vol**

**4/21/2022 (back to Delphi):**

* **Interp\_Common.RealFormat is basing its result on the VersionNr which is incorrect.**
* **Initialize\_Interp is also making assumptions about real size (search for CREALSIZE)**
* **TIIPsystemInterpreter.INCR is basing its result on the version number rather than WORD\_ADDRESSED**

**4/22/2022:**

* **Is IpcBase the same as ProcBase?**
* **Range check error in FetchUB**
  + **Never makes it to MemRdByte**
  + **IPC is very large $FF6A**
  + **IPC comes from IPCSAV**
  + **IPCSAV gets set in StartPME and it is an absolute memory address taken from the JTAB EntryIC.**
  + **In C interpreter, IPC starts out at 0 and is always relative to IpcBase.It is what I would refer to as RelIPC.**

**4/23/2022:**

* **Assume that IPCSAV is used to store IPCRel. NO. I think it should be storing the IPCAbs!**
* **LCA (Delphi) == LSA [C]**

**4/25/2022:**

* **MSIPC is an absolute address**
* **SEG comes from** 
  + **Seg := MemRd(WordIndexed(OldMp, MS\_SEG)) and it gets set during the “ret” call.**
  + **Seg is a memory address.**
  + **Is SEGNUM the same as SEG? Perhaps I can check it on Linux?**
* **IPC comes from** 
  + **Ipc = MemRd(WordIndexed(OldMp, MS\_IPC)) and it gets set during the return call.**

**4/26/2022:**

* **Register equivalences:**

| **Meaning** | **C interpreter** | **Delphi** |
| --- | --- | --- |
| **Global Variables base** | **Base** | **GlobVar**  **BASED0** |
|  | **Np** | **HEAP\_BOT** |
| **bottom of the current segment?** | **Kp** | **HEAP\_TOP**  **SEGBOTTOM** |
|  | **Syscom** |  |
| **Stack Pointer** | **Sp** | **Stack** |
| **current MSCW address**  **(dynamic link)** | **Mp** | **LastMP**  **LocalVar?** |
|  | **BaseMp** |  |
| **starting address of procedure** | **IpcBase** | **ProcBase** |
|  | **Seg** | **SegTop** |
|  | **JTAB** | **JTAB** |

* **Line 6873: byte sex issues- see byte\_sex.h**

**4/28/2022:**

* **Starting gdb:**

**gdb ~/bin/ucsdpsys\_vm**

* **After gdb starts:**
  1. **break load**
  2. **layout src**
  3. **run -w ~/Downloads/system.vol -w ~/Downloads/utility.vol**
* **config.status: Run this file to recreate the current configuration.**

**4/30/2022:**

* **UseCInterp is always set to true**

**5/2/2022:**

* **Start with the ability to run the debugger**

**5/4/2022:**

**FilerMain**

1. **procedure TfrmFiler.OpenDebugger(Version: TVersionNr; UseCInterp: boolean);**
2. **function TfrmFiler.LoadFromUnit(UnitNr: integer; VersionNr: TVersionNr; UseCInterp: boolean): TCustomPsystemInterpreter;**
3. **procedure TfrmFiler.BootFromMounted(Sender: TObject);**
4. **function GetBootItem(UnitNumber: integer; const VolumeName, FileName: string): TBootItem;**
5. **procedure DebugFromMounted(Sender: TObject);**
6. **procedure TfrmFiler.BootInterpreter(BootItem: TBootItem);**

**pCodeDebugger2**

1. **procedure TfrmPCodeDebugger.LoadFromUnit(aVersionNr: TVersionNr; UseCInterp: boolean; Boot\_Unit: integer);**

**Events:**

1. **DebugFromMounted called from menuitem**
2. **Interpreter gets created in LoadFromUnit**
3. **OpenDebugger gets called from DebugFromMounted**
4. **The debugger gets created**
5. **TFrmFiler.OpenDebugger calls TfrmPCodeDebugger.LoadFromUnit**

* **OpenDebugger frees fThePsysWindow and the interpreter (deleted)**
* **When the Debugger window is created, the interpreter is gobbleddygook. When OPenDebugger is called, fInterpreter is already gunk. LoadFromUnit isn’t doing anything that is useful. fInterpreter does not yet exist. Is LoadFromUnit getting called twice? Once in the debugger? Yes! UpdateDebuggerDisplay getting called before Load\_Psystem**
* **Procedure names currently displayed in the debugger are wrong**
* **“Step Into” (F7) is not working or maybe the debugger source is not listing the correct procedure.**
* **ProcNames[] looks like it got loaded correctly**

**5/5/2022:**

* **Appears to be dying in PASCALSY.DECSIZE which is a compiler routine.**
* **The database entries (VI.5) for PASCALSY.\* appear to be compiler entries? The files stored in “opsys-src” sometimes seem to be OK if the file was created prior to 10/04/2021 but also could belong to the compiler or filer.**
* **The listing could be in:** 
  + **I5-SYSLIST.TXT**
  + **System-I5-Listing.TXT**
  + **I5-SYSTEM-LISTING.txt**

**5/6/2022:**

* **The version II database appears to contain the source code**
* **I should be able to extract the PASCALSY records from the version II DB and insert into the I.5 DB.**
* **Getting hung in BLD3**
  + **Dies on this line: “ A := Bytes[NEWJTB]; {get proc#, if 0 then assembly}”**
* **I am in INITIALIZE.INITIALIZE (RelIPC = 8) when I get the DECOPS call**
* **I appear to have returned to the start of INITIALI.INITIALI following the call to FWRITEIN.**
* **This is also where I get 5: I/O error 103 (EInOutError): Segment:0, ProcNum:1, RelIPC:6 Was caused by a call to WriteLn where fLogFile had not been opened (problem with $IfDef SegInfoRec)**
* **I appear to be using a different version of “F:\NDAS-I\d7\Projects\pSystem\Z80EM2010\UCSDI5.RAW” on the two machines.**

**5/7/2022:**

* **NoteSegTopChange being called by DECREC. needed to be inside “{$IfDef SegInfoRec}”**
* **NoteSegTopChange first appeared 20220112**
* **“Debug From Mounted” is getting loaded up with duplicates.**
* **The Version IV db does not show up in pCodesDatabaseFileNameS. See: ConstructStringListForVersion**

**5/9/2022:**

* **There is currently no way to set a default VersionNr**
* **I need to be able to back up for the currently selected VersionNr**
* **SaveMemoToFile should be a method of a TpCodeProcsTable**
* **Getting the message “*No files were copied to F:\NDAS-I\d7\Projects\pSystem\DB Contents\V-IV\*” four times**

**5/10/2022:**

* **I need to pass in “Enable External Pool”. I have to pass the “EnableExternalPool” back to the caller.**
* **InterpIV (7127):** 
  + **Assert(realsize = CREALSIZE, 'SYSTEM.MISCINFO has incompatible**
  + **realsize');**
* **RecentBootsList contains junk in the .ini file**

**5/11/2022:**

* **I managed to get the version IV on SurfacePro (drive C:) running. This version appears to date from about 10/25/2021.**
* **IV error(s?) if EnableExternalPool not enabled**
* **What database is getting used on SurfacePro? It would appear to be the same database.**
* **Exiting Load\_Psystem the registers appear to be identical.**
* **After the first instruction has executed, the registers do not match:**
  + **LOCALVAR**
  + **SP**
* **By the time that BLDFRM is entered, SP is already off by 2 -- may be caused by PUSH(MP)**
  + **SurfacePro refers to a local variable in InterpIV**
  + **XPS-8930 refers to a variable in Interp\_Common**
* **The registers are getting changed in BLDFRM**
* **Range check error on line 1727 in GetBig**
  + **AX := CBW(Bytes[DS+SI]); // get byte and convert to word (sign extend AL)**

**5/12/2022:**

* **Occurs at '26: USERPROG.INITSYSC'-- trying to assign a negative value to AX which is a word. Instruction is LCO.**
* **LCO is getting 128 rather than 96**
* **External Pool is true**
* **I/O Error loading kernel- possibly in USERPROG.LOAD**
* **SYSRBOOT should be leaving the IO result in AX?**

**5/13/2022:**

**When writing out “SEARCH” this is what actually gets sent to the terminal: “'SEA'#$11'RC'#$11'H'” ??? Occurs in PASCALIO.WRITE, Offset: 48**

**5/16/2022:**

* **The UPPERCASE function seems to screw the string up**
* **Memo.Lines.Count = 53,294 ??**

**5/17/2022:**

* **Use PSYS2.BAT to boot on HP Laptop**
* **This boots version IV.2.1 R3.3**
* **Mounts PSYSTEM.VOL, TESTING.VOL and TESTED.VOL**
* **SEARCH.TEXT [v1.12] (4/19/2022) found on UTILS.VOL**
* **Later version of search is [v1.18] and is located on 15SYS1.VOL**
* **Undefined operator 82 in DebugOpsTable**
* **Opcodes 65..95 do not seem to be defined in Version IV**
* **“bytes” created in INTERPC.**
* **What is the decoder trying to decode?**
* **The debugger “registers” window shows the correct value for the OpCode (134: LAO).**
* **Changed the “Watch Address” field in fires whenever the watch address is changed**
* **fpCodeDecoder is defined in pCodeDebugger**
* **XfpCodeDecoder is constructed when the TfrmDecodeWindow is created but the TfrmDecodeWindow is never created. But it should be when ExternalDecoderWindow1Click is called.**
* **Sequence:**
  + **fpCodeDecoder gets created when the debugger gets created**
  + **frmDecodeWindow already exists? Was created when the Filer was created?**
* **“Paste External p-Code” does not do anything.**
* **‘Range Check Error” on Memo.Lines.Count immediately after entering UpString()**
* **I am apparently debugging Version 1.12 (not the latest)**
* **Trying to decode the RZERO code goes crazy. Why is RZERO even getting called?**
  + **It is actually UPPERCASE that is getting called.**
  + **“SEARCH” is already crapped up before UPPERCASE is entered.**
* **Note that the call stack is wonky:**
  + **Dynamic Call Stack Report. Generated on 05/17/2022 2:59:49 PM**
  + **MSCW Proc @IPC**
  + **FF3A 20: PASCALIO.CHARDEVG 379**
  + **FFBA 10: PASCALIO.FREADSTR 128**
  + **FFD0 1: PASCALIO.PASCALIO 129**
  + **FFDA 51: SEARCH. 40 ⇐======**
  + **FFEA 1: SEARCH.SEARCH 5 ⇐======**
  + **FFF4 2: SEARCH.UPPER 6215**
  + **170 1: SEARCH.SEARCH 2356**
* **'Dynamic: 20: PASCALIO.CHARDEVG @ 266, 10: PASCALIO.FREADSTR @ 1034, 1: PASCALIO.PASCALIO @ 1965, Integer overflow @ 51: SEARCH.'**
* **Try putting a breakpoint into “char\_dev\_get” when the ReadLn is about to be executed.**

**5/18/2022:**

* **Pressing <enter> does not pass the #13 through?**
* **Should I be using the FILER.ACCDB (798 recs, 6/23/2021) or SYSTEM4.ACCDB (445 recs, 4/15/2021)?**
* **If the debugger references a non-existant ACCDB, you cannot get into the debugger to fix the settings problem. Maybe I should allow the Debugger settings to be edited in FilerMain???**
* **Does IXA need to multiply by 2?**
* **Reg BP not displaying properly in MemDumpDF**
* **ReadKey0 erratic**
  + **PutPrefixed puts the #17. It thinks that the “a” in “Search” is prefixed (key to move cursor up) and the “c” is a prefixed (key to move cursor up).**

**5/19/2021:**

* **This is the one that I want:**
  + **E:\psysvolumes\UCSDI5 (49).RAW**

**5/20/2022:**

* **The version II sources are located “F:\NDAS-I\Floppy Diskette\PSysSrcs2\Z80Interp20”**
* **GlobalAddr uses WordIndexed**
* **WordIndexed:**
  + **#ifdef WORD\_MEMORY**
  + **#define WordIndexed(Pointer, Offset) ((Pointer) + (Offset))**
  + **#else**
  + **#define WordIndexed(Pointer, Offset) ((Pointer) + 2\*(Offset))**
  + **#endif**
* **MS\_VARw = 5**
* **I think that my “LocalVar” is the same as C interpreters “MP”**
* **I think that my “GlobalVar” is the same as C interpreters “Base”-- Is this true? There is a 5 word discrepancy present (MS\_VARW).**
* **I think that my “GetBig” is the same as C interpreters “FetchB”**
* **I am currently trying to reconcile “LAO”. I think that my “LocalVar” and “GlobVar” already include the MS\_VARW offset …? (I.e., offsets that index past the MSCW.)**

**5/21/2022:**

* **GETIA starts with the ptr based on MP (latest MSCW)**
* **Seems to be using the SAME database**
  + **E:\d7\Projects\pSystem\AccDb\VersionI-5.accdb**
  + **F:\NDAS-I\d7\Projects\pSystem\AccDb\VersionI-5.accdb**
* **Both have 953 records**

**5/24/2022:**

* **PASCALSY.WRITELIN vs FETCHDIR**
* **44 is really FETCHDIR -- not WRITELINE**
* **CSP\_MVL / CSPMove**
  + **They seem to be popping 4 words. I only see 2?**
  + **Could this be a version I.5 vs version ii.0 change?**
* **Here are the sources to the Version II.0 interpreter: F:\NDAS-I\Floppy Diskette\PSysSrcs2\Z80Interp20**
* **I need to the the implementation of MoveLeft in the C interpreter**
* **I am getting LOTS of duplicate entries when loading the debugging database - see the logfiles**

**5/26/2022:**

* **LOCALVAR = MP + 10**
* **Need carefully to watch the loading of segments and verify that the segment has loaded correctly and that the segment dictionary is correct.**

**5/27/2022:**

* **I put some backups of the FILER\_DEBUGGER.EXE files into F:\NDAS-I\Shared Programs\Filer\_Debugger**
* **I am now backing up the EXE file to “Other Sources” - need a separate folder to backup the EXE files**
* **I need to figure out why the number of procedures in Segment 0 is wrong.**

**5/28/2022:**

* **I interchanged InterpII and InterpC**
  + **renamed the files**
  + **changed the unit names**
  + **changed to interpreter names**
    - **TIIPsystemInterpreter ←> TCPsystemInterpreter**
* **This had problems so I completely restored InterpII from an archived version (made on 1/18/2022)**
* **This version I.5 appears to be working**
* **WORKING SOURCES BACKED UP TO “p-system Sources 5/28/2022” in round CD case #42.**
* **The procedure count comes from the last byte of the segment**
* **Segment PascalSy starts at block 1 of SYSTEM.PASCAL**
* **SYSTEM.PASCAL is 3992 words (7984 = $1f30 bytes) long**
* **SYSTEM.PASCAL should end just before $2030**

**5/30/2022:**

* **TfrmPCodeDebuggerII.IdentifierValue**
* **I need to add newly created BootItem to the list**
* **Load\_pSystem is called by** 
  + **FilerMain.BootInterpreter**
  + **pCodeDebugger.DebuggerLoadFromUnit**
* **It does not get called from** 
  + **FilerMain.DebugInterpreter**
* **The segment dictionary gets loaded to $100 (1 block)**
* **If I inherit the interpreter from the custom class (TCustomPsystemInterpreter) I will have to duplicate a lot of the code from the Version II interpreter.**
* **If I inherit from the version II interpreter and I call the inherited version class (Load\_PSystem), I may be calling things that are not appropriate for the “C” interpreter.**
* **I think that I need a parent class that both InterpII and InterpC derive from but which only contains the things that are common to both.**
* **The introductory code is getting displayed twice: (Initialize\_Interp)**
  + **Once when the version II interpreter is created in the inherited**
  + **It gets called again when the interpC in returns from the Version II interpreter**
* **How many things do TCPsystemInterpreter and TIIPsystemInterpreter really have in common?**
* **Maybe the MemDumpDW wt\_SegDict stuff should just be moved to a shared unit?**

**5/31/2022:**

* **aCodeAddr is zero. It ought to be 381?**
  + **NP = 256 ($100)**
  + **WordIndexed = 256 ($100)**
  + **The parameter passed to GetWordAt is 636. Why isn’t it 256 ($100)?**
  + **The InterpC.GetWordAt should have been calling the inherited GetWordAt. I got rid of InterpC.GetWordAt (and InterpC.SetWordAt).**
  + **Kp is reduced by the segment size.**
* **aCodeAddr keeps coming out to be zero?**
* **aCodeLeng seems to be OK?**
* **Segments loaded:**

| **Segment Name** | **UnitNr** | **Length(bytes)** | **Starting Block** | **Address** |
| --- | --- | --- | --- | --- |
| **PascalSY** | **4** | **7984** | **381** | **$DD50** |
| **UserProg** | **4** | **54** | **397** | **$DD1A** |
| **Debugger** | **4** | **7270** | **398** | **$C0BA** |
| **PrintErr** | **4** | **988** | **413** |  |
| **Initiali** | **4** | **2418** | **415** | **$B366** |
| **GetCmd** | **4** | **2102** | **420** | **$AB30** |

* It appears that JTAB might contain both the procedure number (in the low-order byte) and the LexLevel (in the high-order byte)
* LOGFILE.TXT still getting created

6/1/2022:

* Calling Segment 4 (Initialize) Procedure 1 (FOpen).
* The SEGTOP from the SegDict is 0.
* Globals.LowMem.SysCom.SEGTBL is all zero. This never gets initialized because SegInfo = 0.
* **Trying to run the Debian version again:**
  + **bin/ucsdpsys\_vm -w ~/Downloads/system.vol**
* **from this directory directory:**
  + **~/psys/ucsd-psystem-vm-0.11**

**6/3/2022:**

* **Loads 7768 bytes from #4 @56872 starting at block 203**

**6/6/2022:**

* **F:\NDAS-I\d7\Projects\pSystem\Volumes\V1-5\UTILS.SVOL**
* **I would like to see a SEGMAP of SYSTEM.PASCAL on the version II.0 boot disk (SYSTEM.VOL).**
* **The SysRd is trashing memory. Kp probably is not initialized.**
* **Kp initialized to KP\_TOP (= 65162) - SYSCOM\_SIZE (256)**
* **Is Kp a *word* value?**
* **CodeLeng could be word OR byte value depending on the mode (may not be true)**

**6/7/2022:**

* **Boschell puts segments onto the stack.**
* **Did Version II change how segment lengths are stored?**
* **Not displaying “Code\_Size”**

**6/8/2022:**

* **Is Word\_Memory true or false?**
* **Segments get loaded at the following addresses:**

| **Segment Name** | **SegBottom** | **CodeLeng (Bytes)** | **SegTop** | **BlockNr** |
| --- | --- | --- | --- | --- |
| **PASCALSY** | **56866** | **7768** |  | **203** |
| **USERPROG** |  | **54** |  | **187** |
| **DEBUGGER** |  |  |  |  |

* **On Debian I am still running a version compiled on April 13**
* **A more recent version appears to be located in bin/ucsdpsys\_vm**
* **I can run it with** 
  + **~/psys/ucsd-psystem-vm-0.11$ bin/ucsdpsys\_vm -w ~/Downloads/system.vol**
  + **gdb bin/ucsdpsys\_vm**
  + **break load**
  + **layout src**
  + **run -w ~/Downloads/system.vol -w ~/Downloads/utility.vol**
* **Variables are optimized out and cannot always be seen. I think I need to pass CFLAGS to configure.**
* **“load” - SegNo (line 5392) is only zero on the first loop through the code.**

**6/9/2022:**

* **CODE\_LENG appears to be a byte count.**
* **I think that KP is a word index**
* **I think that SP is also a word index**
* **SEGTOP is a word index**
* **KP initially set to 65152 and then adjusted 64896 (to allow room for SYSCOM - unnecessary)**
* **SegDict[0]**
  + **OldKp 65152**
  + **SegTop 65151**
  + **SegBase 61268**
  + **The segment (0?) is loaded at 61268 → 65151 = 3883 words?7768**
* **Loading segment 0 @ 61268 → (61268 + 3884) (= 65152). CodeAddr = 203** 
  + *remember that these are word values*
* **After loading the segment, the SegDict[0] is crapped up.**
* **Only InterpIV seems to be creating New(Bytes) ?? InterpC is using a “GetMem” rather than “New()”**

**6/10/2022:**

* **PASCALSY starts at block 17 in SYSTEM.PASCAL**
* **This is $2200 in the file:**
* **SYSRD does seem to read the right stuff**
* **+7768 should be found at $4058 in the file**
* **$4058 appears to be just beyond the data in the segment.**
* **The word prior to this is $0032. The high byte contains $32 = 50. Could there be 50 procedures? VI.5 has 48 so 50 is plausible for V2.0+.**

**6/13/2022:**

**Appears to be executing segment 0, procedure 1**

**LDCN**

**STL 54**

**CXP 4**

**6/14/2022:**

* **There are two versions of MemRdByte - one for word\_memory and one for not (I think).**
* **MemDumpDF is always assuming that it is passed a *word* address (and doubles it). Changed so that it is now expecting a byte address.**
* **Would be nice to be able to decode better**

**6/15/2022:**

* **The interpreter is created in LoadFromUnit**
* **fInterpreter is OK when the debugger is created**
* **Is “base” the same as GlobalVar?**
* **TheSegTop should get set when a segment is loaded**

**6/16/2022:**

* **Not booting from the specified unit.**
* **GetAccDbAndSegNameIdx is still unable to fetch the SegNameIdx or AccDbNr**

**6/17/2022:**

* **Globals.LowMem.Syscom.SegTbl[] has not been set by the call to SysRd**
* **It only gets set on the 2nd pass through the loop when Mtype is <> 0 but SysRd only gets called on the 1st pass through the loop when SegNo = 0.**

**6/20/2022:**

* **Run3Click**
* **Fetch ← Run\_Psystem ← DebugInterpreter**
* **The debugger never gets control. How do I load a volume in the debugger that is NOT the LastBootedUnit?**
* **The debugger already knows what it is supposed to load from.**
* **Does the Debugger really need “Load” and “Load From Last”? Can these be folded into “Run”?**

**6/21/2022:**

* **DebuggerLoadFromUnit**
* **I need to see the Segment/ProcNames w/o getting into the debugger**
* **SegInfo[0].TheSegTop has gotten set to 69416 somewhere? This is occurring in UpdateSegInfo on this line:**
  + **TheSegTop := SEGBOT + SegLen - 2;**
    - **SegBot = 61650**
    - **SegLen = 7768**
* **CurrentDataSize is accessing JTab which hasn’t been set yet.**
* **GetByteFromMemory needs to adapt to Word\_Memory**
  + **MemRdByte(IpcBase, IPC)**

**6/22/2022:**

* **Change GetWordAt to function correctly for VersionC?**
* **Do I need TfrmPCodeDebuggerC?**
* **ProcNum is wrong. Debugger thinks that it is zero. Decoded thinks that it is 1.**
* **Decoder doing the long jumps incorrectly.**
* **SegTop is incorrect**
* **GlobVar is wrong**
* **LocalVar is wrong**
* **Instruction sequence NEQI, FJP → NFJ**
* **Instruction sequence EQUI, FJP → EFJ**
* **Is EFJ OpCode 210 or 211?** 
  + **C interpreter does not have OpCode 210 and expects EFJ = 211.**
  + **The decoder says that 211 = EFJ and that 210 = BYT (?)**
  + **Laurie does not implement EFJ or NFJ**
  + **Version IV uses 210=EFJ & 211=NFJ**
* **It seems likely that Version I.5 opcodes do not all match those for version II. In particular see “EFJ”, “NFJ” and other opcodes.**
* **MOV and LDO are reversed between Miller and Laurie**
* **LDO < - > MVB**
* **LDO < - > LAE**
* **CGA < - > S1P**

**6/23/2022:**

* **It would be nice to be able to do a DISASSEMBL.CODE of Version II.0. Debian p-System crashes with an I/O error whenever I try to run it.**

**System IO error**

**S# 1, P# 1, I# 4810**

**Type <space> to continue**

* **To run the Debian version of the p-System:**
  + **CD ~/psys/ucsd-psystem-vm-0.11**
  + **bin/ucsdpsys\_vm -w ~/Downloads/system.vol -w ~/Downloads/utility.vol**
* **[I.5] SaveSegInfoForFile ⇐ LoadSegDict ⇐ Load\_Psystem**
* **[I.4] GetSegStuff ⇐ ReadSeg ⇐ GetSeg ⇐ StartPME ⇐ Load\_Psystem**
* **CrtInfo.FunctionPrefixes: #8,#10,#27,'A','H','K','|']**
* **Msg: 'Command: E(dit, R(un, F(ile, C(ompile, X(ecute, D(ebug, I(nit, H(alt [I.4b]'**
* **Version I.4 does not have an “IsPrefixed” property.**

**6/24/2022:**

* **To do today**
  + **Reconcile the OpCode tables with the “C” interpreter.**
* **DISASSEMBL.CODE does not work using the Z80 interpreter. It just hangs.**
* **Be able to display the name of which database is being used.**
* **A segment got loaded without updating the fFilesLoaded list.**
  + **GetSegNum ⇐ GetSegNameIdx ⇐ UpdateDebuggerDisplay**

**6/25/2022:**

* **MountListOfVolumes**
* **TVolume.BlockRead⇐BlockReadRel⇐SYSRD⇐load⇐Load\_Psystem**
* **CspLoadSegment loads a segment but does not update the debug info.Unit 4, Block 192, Size 2318, @ 60491. Can I use UpdateSegInfo?**

**6/27/2022:**

* **LDCN not working?**
* **MemDumpDW(Addr, wt\_HexWords) is assuming that the address is a byte address when it may actually be a word address.**
* **GlobVar is probably wrong**
* **LocalVar is probably wrong**

**6/28/2022:**

* **I need to figure out why the stack display does not agree with a MemDumpDW(SP, wt\_HexWords…)**
* **My MemDumpDW is probably right and the debugger call stack is wrong.**
* **The LocalVariables Addr are incorrect because *LocalVar* is incorrect.**
* ***GlobVar* is also incorrect.**
* ***call* never gets called when the system is started**
* **In Debian, trace through to see what the initial call to *call* does.I.e.,**
* ***ProcBase* is used as a property and as a function.**

**6/29/2022:**

* **VariableList.VarSize needs to be dependent upon Word\_Memory**
* **Column 1 in LocalVariables sgVariables is *ParamAddr***
* **LTITLE is 20 (actually 21) words long but the display doesn’t reflect that**

**6/30/2022:**

* **The Wd# & Addr are incorrect. The “Words” (size) seems to be correct**
* **Code stores ‘???’ at $E66C**
* **Procedure INITIALI.INITHEAP**
* **SYSCOM^.GDIRP := NIL**
  + **0: LOD 2 1 // SYSCOM^ ? What address does the interpreter think it has for SYSCOM?**
  + **3: INC 8**
  + **5: LDCN ⇐ CRASH**
  + **6: STO**
* **Maybe SYSCOM address needed to be pushed onto the stack during initialization? But things seem to work when I use the old code for LOD.**
* **In FETCHB, *MemRdByte(IpcBase, IPC)* gives a different result than *Bytes[ByteIndexed(IpcBase)+IPC]*. They ought to give the same thing. Maybe they do. All three methods give $36. *Remember that IPC has already been bumped.***

**7/1/2022:**

* **RBP and RNP are trying to return a value. NO! They are actually calling a function named *ret.***
* **I need to see what BaseMp gets set to. BaseMp gets set to Mp immediately after the call to load\_system, Mp gets set just before~ the call to load\_system. It gets set to Kp.**
* **SysCom gets loaded into HIGH memory (64982)**
* **I think that they start off by setting the address of SysCom as a local variable.**

**7/2/2022:**

* **Crashing with a *range check error* in FJP ('Procedure name: 4:INITIALI.INITFNAM') at the call to jump. DbgCnt = 468.**

**7/4/2022:**

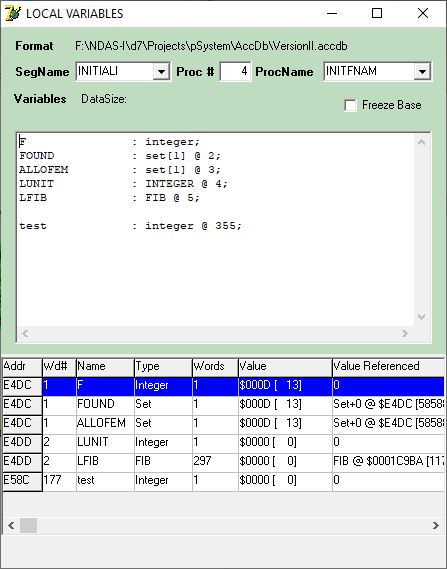
* **The jump instructions (particularly backwards jumps) are dubious.**
* **SearchFor: MyStrPos alters fFindStart when the MyStrPos function is called. Which I don’t think should happen because it is not passed as a VAR parameter to MyStrPos. Result seems OK when MyStrPos is exited. F3 seems to work OK but the original find does not.**

**7/5/2022:**

* **Getting an access error in MemDumpDF**
* **Halting in INITFNAM (@ipc=420) because something is NIL. Looks kind of like INITUNITTABLE:**
  + **FOR LUNIT := 0 TO MAXUNIT DO**
  + **WITH UNITABLE[LUNIT] DO**
  + **BEGIN UVID := '';**
  + **UISBLKD := LUNIT IN [4,5,9..12];**
  + **IF UISBLKD THEN UEOVBLK := MMAXINT;**
  + **UNITCLEAR(LUNIT);**
  + **END;**
  + **UNITABLE[1].UVID := 'CONSOLE';**
  + **UNITABLE[2].UVID := 'SYSTERM';**
  + **SYVID := '';**
  + **LUNIT := VOLSEARCH(SYVID,TRUE,LDIR);**
  + **SYVID := UNITABLE[SYSCOM^.SYSUNIT].UVID;**
  + **IF LENGTH(SYVID) = 0 THEN HALT;**
* **Getting a memory leak when …?**
* **Maybe gdb doesn’t recognize ProcNumber because it is *inline*? Yes. I commented out all of the *inline* directives.**

**7/6/2022:**

* **Currently halts in INITFNAM at the CSP 39 (halt) opcode. Appears to follow a call to VOLSEARCH.**
* **LocalVariables offsets are wrong:**

****

**7/7/2022:**

* **The breakpoint in EFJ**

| **DbgCnt** | **Proc Number** | **IPC Before** | **IPC After** | **ok?** |  |
| --- | --- | --- | --- | --- | --- |
| **110** | **3** | **32** | **53** | **ok** |  |
| **167** | **3** | **46** | **46** | **ok** |  |
| **217** | **3** | **46** | **46** | **ok** |  |
| **267** | **3** | **46** | **46** | **ok** |  |
| **311** | **3** | **32** | **53** | **ok** |  |
| **358** | **3** | **32** | **53** | **ok** |  |
| **429** | **3** | **32** | **53** | **ok** |  |
| **593** | **4** | **198** | **198->389** |  | **INITIALI.INITFNAM (doing a UNITCLEAR(4)). Corrected. UNITTBL was not getting initialized.** |
| **609** | **42** | **20** | **20** |  | **ok. Gets range check error at DbgCnt 621.** |
| **637** | **42** | **68** | **68** |  |  |
| **660** | **42** | **112** | **112** |  |  |
| **667** | **42** | **121** | **121** |  |  |
| **1388** | **4** | **224** | **224** |  |  |

* **For some reason the C compiler is optimizing out my DbgCnt variable. Fixed by changing the compiler optimization level to -O0.**
* **UNITBL[4].Driver is not set. Fixed. The inherited LOAD\_PSYSTEM was not being called.**
* **INITFNAM{E} also appears to include INITUNIT.**
* **CXP to FETCHDIR smashed the unit number on the stack. This is done by *call.* The SEGTOP is 65533 ($FFFD). The SP is $F0CC. The NewJTab is $F4CE. The stack pointer (SP) gets messed just after “copy parameters onto stack”. Moving 6 bytes from word 61644 ⇒ 58572 (byte 128288 ⇒ 117144).**
* **I think that the Debian stack starts in low memory and grows towards high memory? NO. Sp starts out at 512 and grows towards 0.**
* **MoveLeft is moving 6 bytes from 508 to 58396**
* **DbgCnt only exists outside of the main loop?**
* **On Debian, MoveLeft:**

| **ProcNumber** | **Dst** | **Src** | **Len** |
| --- | --- | --- | --- |
|  | **60835** | **512** | **0** |
|  | **58814** | **512** | **0** |
|  | **58436** | **60567** | **4** |
| **1** | **58438** | **60572** | **4** |
| **1** | **58444** | **60587** | **4** |
| **1** | **58448** | **60597** | **4** |
|  | **58450** | **60602** | **4** |

* **I now have a variable called CurrentProcNumber**
* **In call at line 612**

| **DbgCnt** | **CurrentProcNumber** | **Ipc** | **Sp** |
| --- | --- | --- | --- |
| **0** | **0** | **0** | **512** |
| **2** | **1** | **6** | **512** |
| **72** | **1** | **171** | **511** |
| **84** | **7** | **22** | **508** |
| **133** | **7** | **41** | **508** |
| **183** | **7** | **60** | **508** |
| **233** | **7** | **79** | **508** |

* **When CALLing, only room is made for parameters (ParamSize) and not for local data (DataSize)?**

**7/8/2022:**

* **Where is SYSCOM located now?**
* **I think I need to change references to SYSCOM to something like**

**with SysComPtr(@Bytes[ByteIndexed(Syscom)])^ do**

* **I need to move the MiscInfo stuff to follow when Syscom gets set**
* **load\_system changes Mp**
* **Does load\_psystem need to call the *inherited* load\_psystem? Should load\_psystem update the passed UnitNr? Probably not.**
* **Somewhen before DbgCnt=621 (in *call)*, Delphi goes off of the rails in CallIO when attempting to pop the UNUM (line 1189). 'Procedure name:**

**7/11/2022:**

* **The dis-assembled code for FETCHDIR only goes to 358**
* **The directory gets loaded to (word address) 695**

**7/12/2022:**

* **SYSCOM resides at $FFD6**
* **GDIRP^ 695**
* **SysCom offsets appear to be byte offsets**
* **CNEQ (183) is getting a zero comparison type. PascalSy.FetchDir, RelIPC = 167: OpCode = 183, DbgCnt = 703**
* **SOURCE CODE IS NOT GETTING BACKED UP !**

**7/13/2022:**

* **set\_is\_proper\_subset vs set\_is\_improper\_subset**
* **Something is erasing SYSTEM.VOL: 'Procedure name: 42:PASCALSY.FETCHDIR', IPC = 294.**
* **Trying to delete a directory entry because something was wrong with it? Possibly the year >= 100? Or possibly DLASTBYTE <= 0?**
* **The clean up directory function should list any problems that it finds.**

**7/14/2022:**

* **Line 231: (DLASTBYTE <= 0) OR**
  + **Compares 0 > 0 which returns FALSE (which jumps over the DACCESS.YEAR >= 100 code and goes to “OK := FALSE” code. DACCESS.YEAR = 0 (from the packed date) - “Unpack was incorrect”**
* **“Find Again” (F3) is not working.**
* **LocalVariables: “Value Referenced” is not working for simple types (like integers).**
* **NFJ is getting a DISP value of 246 which is out of the range -128..127. Disabled range check errors to fix.**
* **SP\_UPR has not gotten set (or debugger is displaying incorrectly)**
* **Getting a “range check error” in 3:PASCALSYS.FINIT @ I:23. Opcode = 154 (“STO”).**
* **“Untested LDC” when I try to halt the system (“Untested: LDC: P#GETCMD.GETCMD O#314”**

**7/15/2022:**

* **Currently tracing through GETCMD.GETCMD. The p-Code needs to be updated from the external Decode Window.**
* **None of the system commands work.**

**7/18/2022:**

* **3:GETCMD.ASSOCIAT @ 16**
* **CSPLoadSegment**
* **F(iler does load the directory block**
* **The original main.c had a bug in the DIF opcode. (Needs Set.Data[i++] = 0;)**
* **File U134.4\_OS\_SOURCE.raw: sectors appear to be 128 bytes long**
  + **Did I once write a program to try and guess the layout of a volume?**
* **See: Volume F:\NDAS-I\d7\Projects\pSystem\Z80EM2010\U134.4\_OS\_SOURCE.raw: mounted onto Unit# 9. 14 files**
* **Files were extracted to: F:\NDAS-I\d7\Projects\pSystem\Sources\II.0 Sources\U134.4**
* **I may be able to compile the system II using UCSDPSYS\_VM under Linux**

**7/19/2022:**

* **Possibly recompile Vers II using and then use LIBRARIAN to install it into SYSTEM.PASCAL. This was done on UCSDII0.RAW. Just hangs.**
* **reverting to saved volume UCSDII0.RAW**

**58: WITH UNITABLE[LUNIT] DO**

**69: IF UISBLKD THEN**

* **Why does SIND0 not show up in PHITS? Maybe because it is being called SIND1?**
* **The UNITABLE does not include the disk volumes?**
* **Currently synchronizing GETCMD.GETCMD @ about line 336 in the p-code**
* **i need to be able to compile and run a complete version II SYSTEM.PASCAL**

**7/20/2022:**

* **Is the INN function working correctly? For example: 2 in [0,1]. I think that is OK. Only the debugger was displaying it incorrectly.**
* **UNITABLE seems to be incorrect. Indexing is all wrong.**
* **Figure out the correct size for a unittable entry! 12 Bytes.**
* **LTITLE: Len=21, 'SYSTEM::SYSTEM.EDITOR'**
* **The LDC for BADCMD := NOT (CH IN ['E','R','F','C','L','X','A','D','U','I','H','?']); goes into the wilderness**
  + **MemDumpDF calculates the AbsIPC (byteaddress) by doubling the IpcBase giving 121319**
  + **Then, MemDumpDW doubles it again giving 242638**
* **LDC**
  + **314: LDC 6 292 4986 -32768 0 0 0**
  + **'Words @ $0001D9E7 [121319]: 2406 7A01 0013 0080 0000 0000'**
  + **The Ipc should be advanced to 321**
  + **FetchW is going to increment the Ipc**
  + **The final Ipc after the LDC is 328 but I end up in some other unrelated code (GETCMD.ASSOCIAT), 328 is what it ought to be. This IS the first time that LDC has been called. The correct words are loaded and pushed except that -32768 shows up as (+)32767. But I think that although it is not correctly displayed, it is working OK.**
* **I don’t think that the INN operation is working. It is returning something (64) that is neither true or false; I think that it is now OK.**
* **GetChar call in GETCMD (p-Code line 238) seems to return the correct character (‘F’) but the SRO that follows it doesn’t seem to store it or it is not displayed correctly. It gets stored to 59519. The calculator won’t let me enter this address. It is $E87F in hex. I think that it was stored correctly but just not displayed correctly.**
* **Does BADCMD code correctly identify that ‘F’ is legal? The INN function returns 64 (which is TRUE). But the NOT function changes it to 1 (which is still true).**

**7/21/2022:**

* **Trying to execute “F” for filer, actually executes “E” for editor (but cannot find :SYSTEM.EDITOR).**
* **Would be helpful to editor the system source so that each variable is on a separate line and then get a new compiler listing.**
* **Although tracing through the source code implies that ASSOCIATE is getting called to locate the editor, the actual parameter that is passed is “:SYSTEM.FILER” which may (or may not) be correct.**
* **“No file: :SYSTEM.FILER”**
* **Getting a BAD\_TITLE error in FOPEN @ IPC = 341 in INITUNIT**
* **The callstack in the debugger is not working.**
* **Put a breakpoint in CONCAT and see what is getting concatenated.**
* **Would be nice to be able to do MemDumpDF without worrying about Word\_Memory**
* **The .accdb does not contain an entry for CONCAT (was SCANNER)**

**7/22/2022:**

* **DisplayLocalParameters is calling the inherited version of MemDumpDW (not the InterpC) version**
* **I need a MemDumpDF that skips the ByteIndexed stuff for use in the Delphi debugger**

**7/23/2022:**

* **25:PASCALSY.SCOPY is moving 5 (?) characters from ???**
* **Copying 16 characters gives:**
  + **' @ $E3D9 [58329]: \_:SYSTEM.ASSMBLER'**
  + **' @ $E417 [58391]: \_SYSTEM:SYSTEM.AS'**

**which makes no sense? Maybe this is actually trying to append “SYSTEM.ASSEMBLER” to “SYSTEM:” which is ok.**

* **What is the actual address for LTITLE (which is declared at the INITIALIZE level)?**
* **PASCALSYS.SPOS is ?**
* **Does it find SYSTEM:SYSTEM.COMPILER?**
* **SCANTITLE is not doing its job. Possibly fixed.**
* **LEQI not working?**
* **SCANTITL (@ LINE 192) exited because LENGTH(FVID) = 0 but there is a colon (“SYSTEM:SYSTEM.COMPILER”). The Pos function did not find the ‘:’. See the SPOS function. The POS function now finds the ‘:’.**
* **Does the attempt to extract the file name extension work? (Line 405 in SCANTITL). Yes. I think that it is working.**
* **Goes off into an infinite loop now.**

**7/24/2022:**

* **Call stacks are not working**
* **Repeatedly searching for SYSTEM.ASSMBLER, SYSTEM.COMPILER, SYSTEM.STARTUP. It is only finding SYSTEM.COMPILER.**
* **SGS doing something weird? Maybe screwing up the IPC?**
* **SetPop is popping a 31 word set in ADJ at IPC 370 in INITIALI.INITUNIT**
* **SGS called 367 in INITIALI.INITUNIT**
* **trying to fix the call stack**

**7/25/2022:**

* **Look at UCSDExit for clues about how to display the call stack**
* **Seems like JTab got munged in the filer? (Delphi 60611 vs Debian). The CSP\_XIT code was combining variable assignments with non-zero tests and had not been translated to Pascal correctly.**
* **I need to recompile the SYSTEM.FILER. The listing is stored in II.0.FILER-LISTING.TXT**
* **SaveSegInfoForFile needs to be called when the .CODE file is loaded. I am currently calling it on every call to UNITREAD for a blocked volume..**
* **ASSOCIATE loads SYSTEM.FILER**
* **Getting a range check error in TCPsystemInterpreter.GetSegNum because it cannot find the answer in fFilesLoadedList and returns a -1 when a word value is required.**

**7/26/2022:**

* **Is *FindSegInfoRec* doing the same thing as *GetSegInfoRec*?**
* **Here are the routines:**
  + **In UCSDInterpreter:**
    - **SegNameFromSegTop**
    - **SegIdxFromSegTop**
    - **GetSegStuff**
    - **SaveSegInfoForFile**
    - **FindSegInfoRec**
    - **GetSegInfoRec**
  + **In InterpC**
    - **SegIdxFromSegTop**
* **I need to be able to put a breakpoint into the filer. Does the segment info get loaded?**
* **fLogFile never gets closed**
* **Only garbage is getting saved for SYSTEM.FILER. TheAbsFileStartingBlock= 2. ValidDictionary wasn’t always correct.**
* **I need a breakpoint when the filer gets loaded**
  + **CXP loads the segment and correctly sets the values into SEGDICT**
  + **Both GETCMD and FILEHAND now both claim 60819 as their SEGTOP.**
  + **Renamed GetSegStuff to UpdateSegStuff**
  + **TCPsystemInterpreter.GetSegNum calls SegIdxFromSegTop which returns -1 which cannot be assigned to a word and gives a range check error.**
    - **GetSegNum <- GetSegNameIdx <- UpdateDebuggerDisplay**
    - **The debugger is looking for aSegTop 64981 (which probably hasn’t been set yet).**
    - **UpdateSegStuff gets called to update the fFilesLoadedList**
    - **When UpdateSegStuff gets called Globals.LowMem.Syscom.SegTbl[SEGNUM] has not yet been initialized.**
    - **UpdateSegStuff is getting called twice in UpdateSegInfo**
    - **CspLoadSegment initializes SEGTBL[SegNo] but LOAD does not.**

**7/28/2022:**

* **UpdateSegStuff <- UpdateSegInfo <- load. SEGTOP = 64981. But Globals.LowMem.Syscom.SegTbl[SEGNUM] has not yet been set so THESEGTOP does not get set.**
* **Getting a “range check error” after loading:**
  + **GetSegNum <- GetSegNameIdx <- UpdateDebuggerDisplay <- DebuggerLoadFromUnit**
* **The *load* function doesn't update the low memory SEGTBL which may be needed later to update fFilesLoadedList**
  + ***load* calls UpdateSegInfo which tries to use the DiskAddr which was not saved to SEGTBL above.**
* **Deleted the READSEG function.**
* **FILEHAND get loaded with SEGTOP = 60819 but it does not show in the call stack nor does it hit any breakpoint.**
* **“Step-into” is not working. Actually INITIALIZE.INITIALIZE is executing. JTab hasn’t gotten set yet.**
* **Still thinks that the PROCNUM is 1 after entering INITIALIZE. OK but the segnum is 4.**
* **The “Do you want to update the directory…” never gets displayed?**
* **Loads the Filer without going through *load* (Uning UNITREAD)**
* **MemDumpDW(0, wt\_DynamicCallStack) not working correctly. JTab = 0?**
* **JTab gets set to 60229, 62737, etc**
* **PASCALSYS.FETCHDIR, Offset=43**
* **GETCMD and FILEHAND both seem to have a SEGTOP of 60819**
* **CSPLoadSegment gets called to update the segment info for FILEHAND**
* **The low memory SEGTBL (Globals.LowMem.Syscom.SegTbl[SEGNUM]) still shows (CODEUNIT:0; DISKADDR:0; CODELENG:0) so the old SEGTOP never gets cleared.**
* **Look for differences to the backup files**
* **Change made at the line “7/28/2022”**
* **Back to getting a range check error in GetSegNum (presumably because the segment cannot be found). UpdateSegStuff has not been called yet. Put a breakpoint in for SaveSegInfoForFile which has been called 6 times.**

**7/29/2022:**

* **GETCMD and FILEHAND both seem to have a SEGTOP of 60819**
* **GETCMD gets set to SegTop = 60819**
* **When the filer is loaded, UpdateSegStuff ignores GETCMD because TheREFCOUNT = 0. TheREFCOUNT never gets updated.**
* **History is not getting updated? Because TheAccDbFileNumber is almost always -1!**
* **Functions based on JTab:**
  + **ProcExitIpc(JTab: word)**
  + **ProcBase(JTab: word)**
  + **ProcParamSize(JTab: word)**
  + **ProcLexLevel(JTab: word)**
  + **GetEnterIC(JTab: word)**
  + **ProcNumber(JTab: word)**
  + **ProcDataSize(JTab: word)**
* **I should try to see if I can recompile the filer and use it because I might then have an exact match between p-Code and source code!**
* **Try to run the BuildDebugDatabase program on filer.code which is located on F:\NDAS-I\d7\Projects\pSystem\Volumes\Linux Vols\system.vol**
* **When I rename FILER.CODE to SYSTEM.FILER, the “FILEHAND” breakpoint never gets fired.The SYSTEM.FILER does not have PASCALSYS as segment 0.**
  + **F:\NDAS-I\d7\Projects\pSystem\Volumes\system-Debian.vol**
    - **SYSTEM:FILER.CODE5**
    - **SYSTEM:SYSTEM.FILER**

**8/1/2022:**

* **CALLIO can be moved to a more Global level**
* **FILEHAND does not have a TheSegTop listed**
* **Segment marked at block 234**
* **Segment is named as USERPROG**
* **Try to update SYSTEM.FILER to remove segment 0**
* **The error message is generated on line 147 of MAKECALL by WRITEANDCLEAR**
* **I’m getting a HUGE XJP whenI try to decode MESSAGES which is procedure number 5. I need to try and decode SYSTEM.FILER on UCSDII0.RAW.**
* **DISASSEMBLE.CODE does not work on either Z80 or on Debian. The error number (in ERR) is 9 ('No such vol on-line').**
* **This error appears to be coming from**

**MAKECALL(9,NOVOL); { No such vol was on-line } in SCANINPUT.**

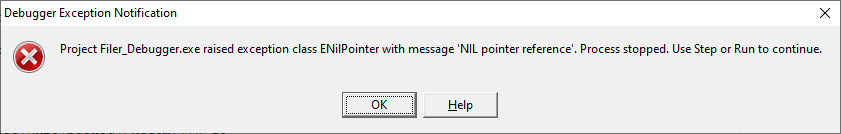
* **The file handler (FILEHAND) breakpoints are not breaking.**
* **Look at *DELETE(FTITLE,1,I)* in SCANTITLE. This is where I stopped debugging. Does the delete work?**

**8/2/2022:**

* **VOLSEARCH returns 0 in SCANINPUT**
* **Set of digits [‘0’..’9’] shows in the debugger as [' '..')']. LDM has only been called 1 time. ‘4’ is not (?) is the set [‘0’..’9’]. The LDM code looks solid. But INN has been called 35 times -- could it be wrong?**
* **The set of digits [‘0’..’9’] is stored in DIGITS @ $EE1C**
* **The VOLSEARCH function seems to have lost one of the words belonging to DIGITS 4,0,0,$3FF,0 should be 4,0,0,0,$3FF**
* **LDM is probably not working correctly. Log the addresses used.**
* **Disassembling MESSAGES gives a gigantic, BAD XJP statement.**

**8/3/2022:**

* **Don’t forget the XJP problem in disassembler.**
* **Getting a NIL pointer reference in PASCALSY.FWRITELN, Ipc=5**
* **The 2nd call to FWRITELN finds the FIB closed. This is when I get the NIL pointer. This is occurring when trying to E(dit #10:SEARCH.**

****

* **Does the editor ever get loaded?**

**Do I need the source code for the editor?**

**8/22/2022:**

* **Why don’t my references to the XJP code seem to be aligned? Am I misusing the debugger addresses (MemDumpDFWB)?**
* **Remember that the XJP instruction FOLLOWS all of the cases that it calls!**
* **The decoder is ignoring its parameters! It is only looking**
* **TpCodeDecoder defined in pCodeDecoderUnit**
  + **TGetBaseAddressFunc = function *{GetLongWordAt}*: longword of object;**
  + **TGetByteFunc = function {GetByteAt}(p: longword): byte of object;**
  + **TGetByteFromMemoryBased = function{GetByteFromMemoryBased}(base: word; offset: word): byte;**
  + **fGetBaseAddressFunc : TGetBaseAddressFunc**
  + **fOnGetByte3 : TGetByteFunc;**
  + **fGetByteFromMemoryBased: TGetByteFromMemoryBased;**
* **TpCodeDecoderII is defined in pCodeDecoderII**
  + **fGetBaseAddressFunc: TGetBaseAddressFunc;**
  + **fGetBaseAddressFunc is not set ($7D41DD0)**
* **TDecodeToMemDump defined in DecodeToMemDumpUnit**
  + **fGetBaseAddressFunc : TGetBaseAddressFunc;**
  + **OnGetByteBased: TGetByteFromMemoryBased**
  + **read fGetByteFromMemoryBased**
  + **write fGetByteFromMemoryBased;**
  + **Things that get set INTO TDecodeToMemDump (BUT NOT INTO THE DECODER!)**
    - **OnGetBaseAddress**
    - **OnGetByteBased**
    - **OnGetWord2**
    - **OnGetJtab**
* **OnGetBaseAddress gets set into fDecodeToMemDump**
* **OnGetByteBased gets set into fDecodeToMemDump**
* **TDecodeToMemDump uses the TpCodeDecoder that is passed in in the Constructor**

**8/23/2022:**

* **When I try to e(Xecute #5:setup, nothing happens- no errors- nothing.**
* **When I try to e(Xecute #5:radix, program runs but CRT is all messed up.**
* **YALOE might be working.**
* **Where are the test programs for the “C” interpreter? They are located in “F:\NDAS-I\d7\Projects\pSystem\Z80EM2010\TEST.RAW” (or TEST.VOL) T0001A.text, etc**
* **T0001A.code gives a range check error**
* **T0004A.text does not exist**
* **T0010A.text will not compile**
* **T0015A - T0019A all give “unimplemented instruction”**

**8/24/2022:**

* **Trying to execute DISASSEMBL on Z80 gives:** 
  + **IO error: file not found**
  + **S# 7, P#1, I# 38”**
* **Trying to execute DISASSEMBL on Debian gives:** 
  + **System IO error**
  + **S# 1, P#1, I# 4810**
* **Could I use the source code for DISASM.TEXT on F:\NDAS-I\Floppy Diskette\PSysSrcs2\AAA-Volumes\15SYS2.VOL version 2?**
* **\*OPCODES.I5**
* **Do a SEGMAP on T0002A.code.**
* **Look at the binary for DISASSEMBL.CODE using UEDIT32.EXE. Trying to open “OPCODES.II.0”? This is located on “F:\NDAS-I\d7\Projects\pSystem\Volumes\UTIL1.VOL”. 8 bytes for each opcode.**
* **After running #9:disassem.code, I get an assertion failure when trying to unload the segment 31.**
* **NIL pointer reference when I try to invoke the editor**
* **Getting a RCE (Range Check Error) when I try to execute T0002A.TEXT:**
  + **Fetch**
  + **CSP**
  + **CSPMove**
  + **MoveLeft(60691, 9690, 60691, 9689, 55847)**
  + **MemWrByte(83, -18704, 4)**
* **It is possible that CSPMove is being asked to move 65535 bytes which is -1 bytes. Problem goes away when I treat “len” as an integer rather than as a word.**
* **I am writing T0005A\_OUT.TEXT to UTILITY: on *public***
* **IDSEARCH SymBufP, RetnInfoP need to make use of ByteIndexed and/or WordIndexed.**

**8/25/2022:**

* **I believe that SymBufP is correct ($02b7 \* 2 → $1390)**
* **Initial entry to IDSEARCH:**
  + **' @ $01FA [ 506]: 02B7 E2D8 2020 2020 2020 2020 0000 0001 0001 0000 0000 0000 0003 040A 0006 090E 000B 0A24 0000 0000 0000 0000 0000 0000 0000'**
* **SymBufP passes a pointer**
* **$E2D8 = 58072**
* **When I do a ByteIndexed on $E2D8, the result is $1C5B0 which is too big to store in a word!**

**8/26/2022:**

* **Memory gets trashed following a call to CSP in T0005A.IDS @ 14 which is a call to FILLCHAR. This occurs BEFORE the T0005A program does anything. BUFADR is getting an address of zero.**
* **SYMBUFP = $02B7. Value stored at that location in $0000?**
* **@SYMBUFP = $E2E8**
* **The value stored at $E2E8 is $2B7**
* **Getting a call to ABI?** 
  + **'42217=ABI (T0005A.EXAMPLE @14),42216=BPT (T0005A.EXAMPLE @13),42215=RBP (PASCALSY.FINIT @76)**
  + **Need to get a listing for T0006A and a disassembled**
* **DISASSEM.CODE is located on F:\NDAS-I\d7\Projects\pSystem\Z80EM2010\UTILITY-II0.RAW**
* **I need T0012A which is on TEST.RAW. It is also on F:\NDAS-I\d7\Projects\pSystem\Z80EM2010\UCSDII0.RAW**
* **Looking at ROUNDing. See *runnit* in file T0012A.TEXT. Put breakpoint in CSPRound and examine things on a case-by-case basis. See the SetRoundMode procedure.**

**8/27/2022:**

* **When I pop real numbers off of the stack, the UCSDREAL2 value is absurd, but when I do the calculation (SIN or COS) the result looks ok? Is probably a problem with having been compiled under a different system.**
* **The SINE calculations do not agree with with either the Debian version of the calculator.**
* **i need to restore UCSDII.RAW**

**8/29/2022:**

* **GoToXY - starts with 30 ($1E)**
* **p-System window is not closed when system is HALTed**
* **SetDefaultTermType(tt\_PETERMILLER, '#27,\*J,\*K,\*A,\*C,#10,#8,,\*J,\*K,,\*H,?,#30');**
* **What does SetTermType actually do? It is used in HandleGoToXY and in HandleEscapeSequence. SetDefaultTermType does NOT set cf\_GotoXY.**
* **#30 never becomes one of the CrtInfo.FunctionPrefixes**
* **CrtFuncInfo[cf\_GotoXY].ch = #0**
* **Where does CrtFuncInfo get set?**
* **CRTInfo is initialized from SYSTEM.MISCINFO in Load\_PSystem. This does NOT initialize cf\_GotoXY.**
* **CrtFuncInfo[cf\_GoToXY] needs to be initialized**
* **SetTermType gets called in load\_system**
* **Where does the ttPeterMIller TermType Esc char get changed from #30 to #27? Its OK on the initial call to SetTermType.**
* **Do I just need to add #30 to the FunctionPrefixes?**
* **I cannot compile TEST.TEXT. Acts like IDSearch might not be working. Re-run the IDSEARCH test program.**

**8/30/2022:**

* **Trying to boot the UCSDII0.raw volume leads to lots of “UNTESTED” when USECOUNT > 0 and then straight to SYSHALT. UCSDII0 works fine in the Z80 emulator.**
* **Trying to boot from UCSDII.VOL on Debian leads to “Stack underflow” and then SIGABRT. This occurs in *load* after 16 loops through the “ Create the Segment Dictionary” code. The crash occurred at DbgCnt 9438. It looks like the crash went off into the wilderness. CurrentProcNumber = 2. CurrentIpc = 3. FixupSeg0 does get called. FixupSeg0(LoadAddr=60923) ← load(Unit=4, BlockNo=260) ← load\_system(root\_unit\_p=xxxxx, file\_name=”system.pascal”) ← main**
* **FixupSeg0:**
  + **seg = 64981**
  + **SegBase = 60923**

| **I** | **JTab** |
| --- | --- |
| **0** | **0** |
| **1** | **0** |
| **2** | **0** |
| **3** | **0** |

**SegNumProc(seg) = 50**

**“No fixup needed”**

**Final non crash pass through switch(OpCode) occurs at DbgCnt = 9435.**

* **Procedure *Error* is in procedure 2 in segment 1**
* **SegmentInfo.SegmentNumber has no SegmentNumber = 1**
* **@SY = 56669**
* **The source code for INSYMBOL does not match the listing that I have**

**8/31/2022:**

* **The compiler source that I have is for version II.0g. SYSTEM.COMPILER is for version 1.3.D001.**
* **It appears that version II.1 allows 32 segments but previous versions only allowed 16 segments.**
* **Error 101 (identifier declared twice) in ENTSTDPROCS**
* **S1P opcode ???**
* **“Run Until Return” does not work in debugger.**
* **I need to move SYSTEM.COMPILER from UCSDII0.RAW to my usual boot disk system-DebianBackup.vol**

**9/1/2022:**

* **Error occurs in ENTSTDNAMES called from COMPINIT. Appears to occur when entering the ID “TRUE”.**
* **Problem might be TREESEARCH**
* **IXA problem occurs at '1:DECLARAT.DECLARAT' @ 12 related to**
* “UNITWRITE(3,DUMMYVAR[-1600],35); (\*ADJUST DISPLAY OF STACK AND HEAP\*)”
* **Not finding basic types like INTEGER, REAL, BOOLEAN… Check that they are getting entered by ENTERID.**
* **Getting an Assertion Failure in CspLoadSegment “Assert(not odd(aSegSize))”**
* **Can compile and run (a few times?) on Z80. At some point I get an error:**
* **If I compile on Delphi and then try to run on Z80, I get an “unknown run-time error”: S#0, P#43, I# 101.**
* **I need to be using the exact same compiler when compiling. Try the one dated 08-Feb-79.**
* **Although both code files are the same length (2 blocks), they are not identical.**
* **My internal SEGMAP doesn’t seem to work for either code file.**

**9/2/2022:**

* **I am using BuildDbDb to update the VersionII.accdb from the following volume: F:\NDAS-I\d7\Projects\pSystem\Volumes\system-DebianBackup.vol. Referring to SYSTEM.COMPILER dated 31-Aug-22. Source code is on COMP: (Z80EM2010 comp-II.RAW**
* **Compiling TESTIO.TEXT under Delphi p-System, leads to immediate Assertion failure (Assert not odd(aSegSize)) when trying to execute the TESTIO.CODE file. Something is causing the compiler to generate bad code (or to write bad code). When compiled using the Z80 compiler, TESTIO seems to work fine (generate good data files) when run on the Delphi system**

**9/3/2022:**

* **HELLOZ: “HELLO” byte 74**
* **HELLOD: “HELLO” byte 103**
* **(\*$1+\*) changes to II.1g compiler RSK 12-Apr-80 }**
* **LocalVariable should be able to handle CHAR[80]**
* **Compiler claims to be a Version II.1 program**

| **Block** | **Offset** | **Segment** |
| --- | --- | --- |
| **257** | **6** | **COMPINIT** |
| **265** | **14** | **DECLARAT** |
| **288** | **37** | **BODYPART** |
| **317** | **66** | **FINISHUP** |

* **I put a breakpoint in *call* on line 604 on Debian** 
  + **if (!ProcNumber(NewJTab))**
* **and a breakpoint in *call* on line 1474 in Delphi**
  + **if (ProcNumber(NewJTab) = 0) then**
* **I need to put a breakpoint in CspLoadSegment and proceed until DECLARAT gets loaded before breaking in *call*.**

**9/5/2022:**

* **Is my LNOT function correct? See GEN0, O #23. I think that it is OK.**
* **Delphi writing to block 355, Debian to block 340**
* **Debian wants to write 9 bytes to Block 0?**
* **Where does the compiler write block 320 (which is the segment dictionary)? PASCALSY.FBLOCKIO ipc=124. I think that it is getting written in FINISHUP.FINISHUP @ line 1056.**

**9/6/2022:**

* **There is also BLOCKWRITE in COMPINIT. 1st write is t block 321 (byte 28200 in SYSTEM.VOL).**
* **Is the ODD function working? See GENWORD. No. It wasn’t.**
* **STB is being passed $FFFF as a byte to store.**

**9/7/2022:**

* **When I try to boot theUCSDII boot disk, I get this: “UnTested: load - UseCount > 0”.**
* **I also get this:** 
  + **Error = 2, EnterIC = $FD91 (64913), LexLevel = 0, S#0, P#0, O#6**
* **Compiling and running UNITCLEAR gives:**
  + **Error = 2, EnterIC = $FD1C (64796), LexLevel = 0, S#0, P#0, O#107**
* **Compiling and running COMPSTR gives:**
  + **Error = 2, EnterIC = $FD1C (64796), LexLevel = 0, S#0, P#0, O#107**
* **Compiling HELLO2 stops with a compiler error ESYIOERR: I/O error: 9. Appears to occur following a call (in CXP) to CspLoadSegment. Occurring in FINISHUP.FINISHUP 68 or 70. The IOResult appears to be the result x#9:hello2of trying to write to the non-existent unit #3. fOnPutIOResult is not assigned.**
  + **It appears that OnPutIOResult has never been used.**
  + **I need to initialize it somewhere early on in the interpreter.**
* **Compiling and running HELLO2 now gives:**
  + **Error = 2, EnterIC = $FD1C (64796), LexLevel = 0, S#0, P#0, O#107**
* **OnPutIoResult should be moved to a high place in the hierarchy**
* **The Error = 2 seems to be occurring in COMMAND @ 105. It is the result of an exception (ENOPROC) being raised.**
* **My source code for LISTDIR does not seem to be correct. Its not really LISTDIR.**
* **PASCALSY.#43 appears to be CHECKDEL. No. it is really PASCALSY.COMMAND.**
* **I don’t see CspLoadSegment getting called?**
* **CXP02 is not using ByteIndexed**
* **PASCALSY.#43 is really COMMAND.**
* **currently the error looks like this:** 
  + **'Error = 2, EnterIC = $FD1C (64796), LexLevel = 0, PASCALSY.COMMAND, O#105'**

**9/8/2022:**

* **ENOPROC is getting raised but the breakpoint is getting skipped somehow.**
* **When i try to execute HELLO, it says must Link first?**
* **Does SYSRD expect length to be in bytes or words?**
* **Does the SEGTBL store the length in words or bytes? I need to be careful of the calls to SYSRD!**
* **Currently dying in PASCALSY.FETCHDIR at IPC 314 with range check error on an INC opcode**
* **FETCHDIR @ 683 - range check error**
* **Just about any program that I run gives a range check error or ENOPROC.**
* **I get “bin/ucsdpsys\_vm This was using the “new” compiler dated 31-Aug-22.-w /public/system.vol” when I try to compile and run HELLO.CODE on Debian.**
* **For some reason I am not seeing the shared system.vol synchronized between debian and windows.**
* **When I switched to the old compiler, HELLO did seem to compile and run. I am getting an “illegal call to native code” (or an “illegal instruction”)**
* **The problem occurs right after printing “DONE”. Maybe it is related to trying to write a real number.**

**9/9/2022:**

* **Getting an IO error 8 when it tries to open the output file.**
* **Caught in a loop of run-time errorsR**
* **'Seeking Invalid block number (1030) in volume “TESTS2"',**

**9/12/2022:**

* **Program EXTCHARS is non-compatible source code. Cannot compile.**
* **Passed:**
  + **UNITCLEAR**
  + **COMPSTR**
  + **SETTEST**
* **Failed:**
  + **HELLO2 - Illegal call to native code - probably when attempting to write a real number**
  + **SEARCH fails on call to native code - probably the STR function.**
    - **DECOPS undefined when I try to link**
  + **Link with SYSTEM.LIBRARY**
  + **Run-time errors are not being handled well**
  + **Output to screen not handling tabs or DLE?**
  + **SEARCH.CODE is also searching .CODE files?**
  + **Definite memory leak if I try to execute the debugger when it is already running. Also trouble if I try to reuse the debugger rather than quitting it and starting over.**

**9/13/2022:**

* **Executing LIBRARY.CODE on Debian gives; “No proc in seg-table; S# 0, P#43, I# 961; Type <space> to continue**
* **Executing LIBRARY.CODE on Delphi gives: “Error = 2, EnterIC = $FDIC (64796), LexLevel = 0, PASCALSY.COMMAND, O#105; press any key to continue”.**
* **Synchronization problems between “\\debian\public\system.vol” from Windows side and “/public/system.vol” from the Debian side?**
  + **Delphi says the volume has 2008 blocks**
  + **Debian says that it has 2000 blocks**
  + **Stopped happening for no apparent reason? Possibly related to BOOTINFO having the wrong path name?**
* **Trying to execute SEGMAP.CODE on SEGMAP.CODE access violation on Delphi. Doing so on Debian seems OK (i.e., does not crash but claims that it is a Version III segment).**
* **-Or- SEGMAP.CODE active on SYSTEM.EDITOR in Delphi leads to NIL pointer ref.**
* **p-System window does not remember where it was last at.**
* **'49909=STB (PASCALSY.FWRITELN @5),49908=SLDC13 (PASCALSY.FWRITELN @4),49907=SLDC0 (PASCALSY.FWRITELN @3),49906=SIND0 (PASCALSY.FWRITELN @2),49905=SLDO1 (PASCALSY.FWRITELN @1),49904=CXP (USERPROG. @59),49903=LAO (USERPROG. @57),49902=UJP (USERPROG. @46),49901=RNP (USERPROG. @27),49900=CSP (USERPROG. @25),49899=SLDC0 (USERPROG. @24),49898=SLDC0 (USERPROG. @23),49897=SLDC1 (USERPROG. @22),49896=IXS (USERPROG. @21),49895=SLDC1 (USERPROG. @20),49894=LLA (USERPROG. @18),49893=SLDC1 (USERPROG. @17),49892=STB (USERPROG. @16),49891=SLDC11 (USERPROG. @15),49890=IXS (USERPROG. @14)'**
* **Try to get a listing for SEGMAP and a dis-assembly.**
* **FWINDOW is NIL. Should have gotten set in the call to FINIT. Exception occurs when MAP is called.**
* **OPENCODEFILE seems to have been able to open the file but never sets the file name or FWINDOW**
* **Crash occurs when MENU is exited.**
* **DTID in FWRITELN is gobbledy gook?**

**9/14/2022:**

* **I get the NIL pointer ref in PASCALSY.FWRITELN @ 5 immediately after exiting INITIALI.INITIALI -- in STB**
* **When looking at a FIB, the DTID seems to be a mess.**
* **Upon reaching DirEntryFormat, Addr = 0;**
* **The FWRITELN call from MENU @ O # 28 appears to be passing a bad FIB. This could be a call to the default OUTPUT file.**
* **The OUTPUTFIB Window is at $25B and the FIB is at $023D**
* **The FIB that is being written to is at $E02F**
* **Does IsUserProg ever get used? Yes. If IsUserProg is set CurrentSegName returns “USERPROG”.**
* **MemChanged breakpoint is not breaking. Probably problem with ByteIndexed. Yes. That is the problem. Quick and Dirty fix: double the address. Later fix: pass another parameter *OldDebugger* to fetch ByteIndexed bytes.**
* **The FIB gets opened by a call to FOPEN coming from offset 371 in SEGMAP. FIB F is passed into FOPEN. This is probably a REWRITE in SEGMAP.INITIALIZE.**
* **Version II DB thinks that procedure 49 is ZEROVOLUME. It is actually RESETER.**
* **Can I break on USERPROG.OPENCODE? tHE PROBLEM appears to be occurring following a *CLOSE( CodeFile )* immediately followed by *RESET( CodeFile, fn ) ;***

**9/15/2022:**

* **fFilesLoadedList[FileIdx].SegInfo[aSegNum] = (TheREFCOUNT:0; TheSEGTOP:0; TheSEGNAME:'SEGMAP '; TheCodeAddr:1; TheCODEleng:6164)**
* **Somehow TheSegName is getting set to USERPROG but TheSEGTOP is 0.**
* **‘USERPROG’ initially gets set into the FilesLoadedList when SYSTEM.PASCAL is loaded. Does SEGTOP get overwritten when SEGMAP gets loaded? Put a break in CSPLoadSegment when SEGMAP is about to be loaded.**
* **TheSegName gets set to “SEGMAP” when SEGMAP.CODE gets loaded.**
* **60819 still thinks that the SegName is USERPROG. SegName passed in is “”. This is just a local parameter in UpdateSegInfo.**
  + **CSPUnitRead initially saves the segment info for the file SEGMAP.CODE using SaveSegInfoForFile**
  + **SegInfoRecFromTop is used to clear the info when CSPUnloadSegment is called.**
  + **CSPLoadSegment is called. At this point fFilesLoadedList[1].SegInfo[1] knows that TheSegName is ‘SEGMAP~~bb~~’ but TheSEGTOP is not yet known.**
* **CODEFILE starts off with FWINDOW = 0**
* **Death occurs in SEGMAP.MAP @ 59 When it is trying to do a WRITELN to ‘o’ (the output file).**
* **‘o’ should get initialized in *Initialize.***
* **At some point I wrote a test program to test GOTOXY. Maybe it was on the HP laptop or the Win XP virtual machine.**
  + **See: SCTEST.TEXT F:\NDAS-I\d7\Projects\pSystem\Volumes\CONFIG.VOL**
* **keyword search for multiple keywords does not work**
* **Possibly search for SCREENTEST.CODE and see when the volume was last accessed. No clues as to the location of the SCREENTEST source program.**
* **What I wanted was CHARTEST.TEXT**
* **Also possibly look at TESTING.VOL more carefully**

**9/16/2022:**

* **Could the formatting problems be related to CR not being sent?**
* **#0 Msg? i.E., FWINDOW is nil and is being ignored by MemWrByte (exception ignored). Occurring in FWRITECHAR {FWINDOW^[0] := ' '}, FWRITELN {F.FWINDOW^[0] := CHR(EOL)},**
* **Debian is also giving a NIL pointer ref sometimes (but not always) when I try to execute SEGMAP**
* **The editor truncates text files some times (only on Delphi so far). This was dues to a bad MOVELEFT.**

**9/19/2022**

* **Look for the CRT testing program-- cannot find it. What I wanted was CHARTEST.TEXT**
* **get SEGMAP to work with just “writeln(” rather than “writeln(o”**
* **write a program to test FILLCHAR, MOVELEFT, MOVERIGHT**
* **Install UEDIT32 on SurfacePro**

**9/20/2022:**

* **Try to boot with the Laurie Boschell UCSDII0.RAW**
  + **Trying to boot it gives** 
    - Error = 2, EnterIC = $FD91 (64913), LexLevel = -1, PASCALSY.PASCALSY, O#4
    - Error = 2, EnterIC = $FD16 (64790), LexLevel = 0, PASCALSY.COMMAD, O#87
    - **Untested: load - UseCount > 0**
* **Trying to execute LIBRARY.CODE gives NOPROC error**
* **Could this be a version IV library? Doing a SEGMAP on it seems so.**
* **Trying to boot from UCSDII.VOL on Debian gives “stack underflow”**
* **To do:**
  + **I need the ability to be able to write real numbers**
  + **Ability to link with SYSTEM.LIBRARY**
  + **Ability to boot UCSDII.VOL**
  + **Fix the problem with closing the pSysWindow**
  + **Need the ability to compile PASCALIO.TEXT.**
  + **Would be nice to have at least a primitive version of SCREENOPS**
* **Getting a “native code” error when I try to run SEARCH.CODE. No longer occurring?**
* **Compiling SEARCH.TEXT is supposed to send the listing to PRINTER:. It does not. The “{$Lprinter:}” was incorrect.**
* **When I run SEARCH.CODE, I am getting “NIL pointer reference in segment PASCALSY, procedure 22 @ IPC: 5”. This is probably the same problem that I was seeing previously related to FWINDOW not being initialized in the FIB.**

**9/21/2022:**

* **Trying to load the UCSDII0 volume completely whacks the Delphi debugger. Seems to occur even before I reach the ‘load’ procedure?**
* **Things seem OK when I mount and run \\Debian\public\system.vol**
* **The BootInfo list is not working.**
* **SEGINFO is 0 for all of the segments in SYSTEM.PASCAL**
* **The Debian version of SYSTEM.PASCAL lists the Version as “II.1”, the “machine” as “PCode-”, the “kind” as “Linked executable” and lists segment numbers 1..5.**
* **The Z80 version of SYSTEM.PASCAL lists the version as “Volition”, nothing for the machine, kind or Seg.**
* **When loading from the Debian SYSTEM.VOL, the SegNo contains the segment number. When loading from the Z80 version :) the SegNo is 0.**
* **============================================================================**
* **SegMap[0.1f] M(ap, O(utput file, P(refix, Q(uit**
* **File: (\\DEBIAN\public\system.vol):SYSTEM.PASCAL**
* **# Name Addr Len Version Machine Kind Seg Text**
* **0 PASCALSY 17 3884 Volition**
* **1 USERPROG 1 27 II.1 PCode- Linked excutable 1**
* **2 DEBUGGER 2 30 II.1 PCode- Linked excutable 2**
* **3 PRINTERR 3 517 II.1 PCode- Linked excutable 3**
* **4 INITIALI 6 1159 II.1 PCode- Linked excutable 4**
* **5 GETCMD 11 1298 II.1 PCode- Linked excutable 5**

**============================================================================**

* **SegMap[0.1f] M(ap, O(utput file, P(refix, Q(uit**
* **File: (F:\NDAS-I\d7\Projects\pSystem\Volumes\UCSDII0.vol):SYSTEM.PASCAL**
* **# Name Addr Len Version Machine Kind Seg Text**
* **0 PASCALSY 1 4059 Volition**
* **1 USERPROG 17 28 Volition**
* **2 DEBUGGER 18 31 Volition**
* **3 PRINTERR 19 526 Volition**
* **4 INITIALI 22 1200 Volition**
* **5 GETCMD 27 1344 Volition**

**9/22/2022:**

* **I can successfully compile, link and run REALTEST using the Z80 simulator.**
* **I can compile it under Delphi system but (even if linked) it gets a NOPROC error.**
* **I can compile and try to link it on Debian but I get proc FWRITERE undefined in the link process.**
* **I can compile and run DZ80 on the Delphi system. Problems with uninitialized FWINDOW in WRITELNs.**
* **I would like to find the source code that matches the output from DZ80. A possible source for the interpreter is INTERP.TEXT from UCSDII0.**

**9/23/2022:**

* **ENOPROC seems to be a poison pill. System cannot recover.**
* **Missing DECOPS**
* **Does the DEBIAN version have DECOPS?**
* **How did I handle DECOPS IN version 1.5?**
  + **I overrode system calls in CXP and handled SYSCALL 49 (DECOPS).**
* **Priorities:**
  + **Try to get PERUSE4.6 to work**
    - **I probably need DECOPS to do that**
  + **Compile a simple floating point number with a WRITE statement.**
  + **Disassemble the code to see what was generated.**
  + **Compile a simple long integer number with a WRITE statement.**
  + **Disassemble the code to see what was generated.**
  + **I need to alter CGP 0 2 to call the DECOPS procedures**
  + **“LDCI 31415” did not seem to work**
    - **LDC getting called rather than LDCI?**
    - **OpCode should be 199 (LDCI) but it actually is 179 (LDC)**

**9/24/2022:**

* **Trying to figure out the “CGP 3”**
* **The TOS contains 0, B0A1, 12B9**
  + **The B0A1, 12B9 is a long integer-- presumably 314159265**
  + **I don’t know what the 0 is for nor do I know how this becomes the information necessary to allow writing to a file (FIB is @ 573 = $023D) which is not on the stack.**
* **Long integers are stored with the LSW on the right**
* **The FIB address is still on the stack @ $23D buried beneath a bunch of an adjusted (10 words) longword value**

**9/25/2022:**

* **Using DISASM.II located on \\Debian\public\system.vol**
* **CGP 3 is preceded by a push of the requested field width x:width**
* **[15SYS1]LIBMAP - Gets an ASSERTion error in LSA - “odd address”**
* **SEGMAP - F:\NDAS-I\d7\Projects\pSystem\Volumes\V1-5\TESTED.VOL**
* **Trying to execute SEGMAP gives a “No Proc in Seg Table”**
* **Trying to execute LIBRARY gives a ENOPROC error**
* **Linking REALTEST and then trying to run it gives ENOPROC error**
* **Compiling and trying to link LONGTEST gives proc “DECOPS ” undefined.**
* **SEGMAP says that SYSTEM.LIBRARY contains PASCALIO.**
* **Trying to recompile LIBRARY.TEXT on 15SYS1.VOL**
* **LONGTEST.CODE does have external references to**
  + **2. DECOPS**
  + **3. FWRITEDE**
* **REALTEST.CODE claims that it is completely linked**
  + **but when I try to run it, I get ENOPROC**
* **Compiling REALTEST again, LIBMAP states that the external procedure FWRITERE will be needed.**
* **PASCALIO.TEXT is located on 15SYS1.VOL**
* **I think that I need to do {$I GLOBALS.TEXT} -- SEE SYSTEM.TEXT on 15SYS1.VOL**
  + **{$I GLOBALS }**
  + **{$I SYSSEGS }**
  + **{$I SYSTEM.B }**
  + **{$I SYSTEM.C }**
* **i can compile PASCALIO.TEXT by compiling PIO.TXT on 1.5SYS1: (F:\NDAS-I\d7\Projects\pSystem\Volumes\15SYS1.VOL)**
* **Trying to link in SYSTEM.LIBRARY (which contains PASCALIO) fails with FWRITERE undefined.**
* **Trying to link in DECOPS may require the same kind of kludge that I used in version IV.**
* **Are the names stored in PIO.CODE longer than 8 characters?**
* **Try to debug the linker**

**9/27/2022:**

* **Source code to the linker can be found on 15SYS1.VOL**
* **Unable to open #01:LINK3B.TEXT**
  + **I am getting ABSTRACT ERRORS on PutIOResult**
  + **I am getting ODD address errors**
  + **Division by 0.**
  + **- apparently I was trying to run a Laurence Boshell version**
* **Why does my linker listing end abruptly? I have no listing for LINKER.LINKER. It was because there was a “{$L” in the source code.**
* **The “undefined” appears to be occurring in procedure *resolve.***

**9/28/2022:**

* **FWRITERE is an EXTPROC with 0? params, SRCPROC = 2, PLACE = 5**
* **WP^.DEFSYM is NIL which leads to “Proc undefined”**
* **When I run LIBMAP on the original SYSTEM.LIBRARY, this is what I get:**
* **When I run LIBMAP on my compiled version of PIO.CODE, I get errors and gobbeldy gook.**
* **When I try to link my version of REALTEST.CODE with SYSTEM.LIBRARY, it says “Unit PASCALIO not found”.**
* **Linker Proc #22 is actually VALIDATE**
* **Executing “OLD.LINKER” (VII.0) but what I compiled is VI.5 (this is contained in SYSTEM.LINKER).**
* **LITYPES appears to have different definitions:**
* **LITYPES = ({0}EOFMARK, {1}MODDULE, {2}GLOBREF, {3}PUBBLIC, {4}PRIVVATE, {5}CONNSTANT, {6}GLOBDEF, {7}PUBLICDEF, {8}CONSTDEF, {9}EXTPROC, {10}EXTFUNC, {11}SSEPPROC, {12}SSEPFUNC, {13}SEPPREF, {14}SEPFREF);**
* ***sepsrc* is looking for oktype=SSEPPROC but is actually EXTPROC .**
* **DefSym should either get set in *sepsrch* or in *resolve* .**
* **Trying to simply load PASCALIO into REALTEST.CODE via LIBRARY.CODE just leads to ENOPROC.**
* **Try to link with SYSTEM.LIBRARY from the Z80 emulator boot disk.**

**10/3/2022:**

* **Debian:**
  + **Appears to be running version II.0**
  + **Trying to execute SETUP.CODE still gives** 
    - **“No proc in seg-table; S #0, P#43, I# 961**
  + **Executing TESTLONG gives** 
    - **“Unimplemented instruction; S# 1, P# 1, I# 4935**
  + **Trying to link LONGTEST.CODE with OLD.LIBRARY gives:**
    - **Linker [I.5]**
    - **Host file? longtest**
    - **Opening longtest.CODE**
    - **Lib file? old.library**
    - **Opening old.library**
    - **Lib file?**
    - **Map name?**
    - **Reading LONGTEST**
    - **Reading PASCALIO**
    - **Unit PASCALIO not found**
    - **Type <sp>(continue), <esc>(terminate)**
  + **SYSTEM.PASCAL is dated 18-May-11**
  + **OLD.LIBRARY is dated 17-Apr-79**
  + **SETUP.CODE is dated 14-May-79**
* **Z80 Emulator also claims to be version II.0**
  + **SYSTEM.PASCAL is dated 10-Feb-79**
  + **SYSTEM.LIBRARY is dated 17-Apr-79**
  + **SETUP.CODE is dated 14-May-79 -- It works!**
* **The version II sources are located “F:\NDAS-I\Floppy Diskette\PSysSrcs2\Z80Interp20”**
* **F:\NDAS-I\d7\Projects\pSystem\Volumes\UCSDII.VOL**
* **Trying to boot to UCSDII**
  + **Immediately goes to EOPROC (DbgCnt = 2)**
  + **'2=CXP (PASCALSY.PASCALSY @4),1=STL (PASCALSY.PASCALSY @2),0=LDCN (PASCALSY.PASCALSY @1)'**
  + **and appears to lock up Delphi**

**10/4/2022:**

* **SYSTEM.VOL Sequence of events**
  1. **Enter the Filer**
  2. **{Debug} \\Debian\public\system.volume**
  3. **Enter *DebugClick***
  4. **Load Version**
     + **Load Peter Miller Derived Version**
     + **Version II**
     + **#4:system**
     + **File Name: \\Debian\public\system.vol**
     + **Boot**
  5. ***DebugInterpreter***
  6. ***LoadFromUnit***
     + **dismount all subsidiary volumes**
  7. **fInterpreter := TCPsystemInterpreter.Create**
  8. ***GetThePSyswindow***
  9. ***FindNamedWindow***
  10. ***TfrmPSysWindow.Create***
  11. ***LoadWindowInfo***
  12. ***FindNamedWindow***
  13. ***TfrmPCodeDebuggerC.Create***
  14. ***TfrmPCodeDebuggerII.Create***
  15. ***TfrmPCodeDebugger.Create***
      + ***TfrmPCodeDebuggerCustom.Create***
      + ***DebuggerSettings.LoadFromFile(DebuggerSettingsFileName)***
      + ***InitDebugUnit***
      + **Open ALL of the p-Code procedure tables (in TfrmPCodeDebugger.Create)**
      + **LoadProcedureNames**
      + ***DisplayBreakPoints***
      + ***DisplayWatches***
      + ***TpCodeDecoderII.Create***
      + **{DebugInterpreter}*.DebuggerLoadFromUnit***
        - ***Load\_PSystem***
        - ***ResetVolume(4)***
  16. ***---- At this point the p-System is loaded and ready to run (F9)***
* **Volume F:\NDAS-I\d7\Projects\pSystem\Volumes\UCSDII0.vol Sequence of events**
  1. **Enter the filer**
  2. **Enter DebugClick**
  3. **Load Peter Miller Derived Version**
     + **Version II**
     + **#4:system**
     + **File Name: #4 F:\NDAS-I\D7\PROJECTS\VOLUMES\UCSDII0.VOL**
     + **Boot**
  4. **DebugInterpreter**
     + **LoadFromUnit**
       - **dismount all subsidiary volumes**
       - **TCPsystemInterpreter.Create**
       - **GetThePSyswindow**
     + ***TfrmPCodeDebuggerC.Create***
     + **TfrmPCodeDebuggerII.Create**
     + **TfrmPCodeDebugger.Create**
     + **TfrmPCodeDebuggerCustom.Create**
       - ***DebuggerSettings.LoadFromFile(DebuggerSettingsFileName)***
       - ***InitDebugUnit***
         * **LoadProcedureNames**
         * ***DisplayBreakPoints***
         * ***DisplayWatches***
         * ***TpCodeDecoderII.Create***
         * ***TpCodeDecoder.Create***
         * ***InitDebugOpsTable***

***Initialise***

* + - ***Load\_PSystem is doing some screwy things with UnitNr:*** 
      * ***It is a passed parameter that is saved to fBootUnit***
      * ***It is then used as a local variable that is used to ResetVolume for all of the mounted volumes***
    - **TCPsystemInterpreter.Load\_PSystem**
    - **SysCom seems to duplicate SyscomAddr**
    - **Load\_pSystem has a FillChar which could be dangerous**
    - ***load\_system***
    - ***FindFileFirstBlockOnUnit***
    - ***Load***
      * ***SysRd(4, 512, 512, 260)***
      * **How do I know if a segment dictionary has 32 segments?**
    - ***---- At this point the p-System is loaded and ready to run (F9)***

**10/05/2022:**

* **Latest version of pSysDebugWindow is in “F:\NDAS-I\Source Code\Other Sources\pr20200809\pSystem\Debugger”**

**10/6/2022:**

* **See pSysDebugWindow for DumpCodeSeg**
* **See TIIPsystemInterpreter.Create for how to allocate the memory**
* **Where do I get the segment length from?**
* **SEGMAP gives an IO error 103 when trying to process SYSTEM.PASCAL on [SYSTEM4]**
* **Assume that the segment dictionary contains word counts for length**

**10/7/2022:**

* **How many procedures in some known .code file?**
* **SEARCH.CODE appears to have about 17 procedures.**
* **The file size is 8 blocks == 4096 (bytes) = $1000 (bytes) = 2048 (bytes) = $800 (words).**
* **The code\_leng is 3258 (words) or 6516 (bytes) → $1974 (bytes)**
* **$1974 + $200 should put the last byte @ $1B74 from start of the file (which is beyond the end of the file?)**
* **If I assume that 3258 is bytes (rather than words), that would put the end of the segment at 512 + 3258 → 3770 (bytes) = $0EBA (bytes) which does point to a byte containing $11 = 17 (17 procedures?).**

**10/8/2022:**

* **DiskFormat is not initialized**
  + **Breakpoint Expression VolumeName: 'COMP-II'**
  + **Breakpoint Expression VolumeName: 'COMP'**
  + **Breakpoint Expression VolumeName: 'CONSTANTES'**
* **Trying to process a “.PAS” file.**

**10/10/2022:**

* **Why are some segments UCSD and some SofTech?**
* **Figure out the “Invalid buffer offset” (mostly version IV?)**
* **Find “end of segment” in version IV**
* **Figure out the “Range Check Error”**

**10/11/2022:**

* **TVolume.ProcedureInfoScan**
* **TVolume.ScanForProcedureInfo**
* **{pSysVolumes} MajorVersion := ExtractMajorVersion(SegDic.Dict.Seg\_Info[0].SegInfo);**
* **{SegMap} ExtractMajorVersion**
* **Do a WriteLn even if there is “Invalid buffer offset”**
* **NUmber of words in segment**
* **I need the length of a segment for Version IV. For version IV, SEG\_LENG is in words**
* **Where do I get the segment number in Version IV?**
* **Version IV allows more than 16 segments per code file.**
* **What does SEGMAP show for the segment length of VERIFY.CODE on volume CODE4?**
* **‘Flip” is screwing up the .CSV file**

**10/12/2022:**

* **FlipSegDic is expecting a SegDicRec:**

**SegDicRec = RECORD**

**Flipped : BOOLEAN ;**

**case integer of**

**0: (Dict : TSeg\_Dict);**

**1: (aBlock : block) }**

**END ;**

* **Seems to be going off the rails immediately after flipping VERIFY.CODE**
* **Invalid segment names are causing problems.**

**10/13/2022:**

* **Is ItIsFlipped getting a pointer to the right thing?**
* **ByteOffset is wacky.**
* **VERIFY.CODE has one segment. Its length is 433 and it is flipped.**
* **None of the .SVOL are being searched.**
* **DOSSRC.SVOL**

**10/17/2022:**

|  | **Debian** | **Z80** |
| --- | --- | --- |
| **VOLUME Name** | **system.vol** | **USCDII0.RAW** |
| **SYSTEM.PASCAL version** | **II.0** | **II.0** |
| **SYSTEM.PASCAL date** | **18-May-2011** | **10-Feb-1979** |
| **SYSTEM.PASCAL file size** | **33 blocks** | **33 blocks** |
| **SYSTEM.PASCAL byte sex** | **LSB** |  |
| **SYSTEM.PASCAL word addressed** | **TRUE** |  |
| **Notes** | **1, 2** | **1, 2** |
|  |  |  |

1. **For Version II, Code Leng displayed appears to be number of bytes**
2. **For Version IV, Code Leng appears to be number of words.**

**Comparison decoded files:**

* **Running onPeter Miller System (\\Debian\System.vol, DISASM.II.CODE)**
* **V2 SystemPascalDecoded(fromDebian).txt**
  + **decoding SYSTEM.PASCAL on the boot volume, decoding to When disassembling to V2 SystemPascalDecoded(fromDebian), I get (ERangeError, Seg 0, ProcNUm 10, RelIPC 187, Ofs 187)**
* **V2 SystemPascalDecoded(fromUCSDII0).txt:**
  + **decoding SYSTEM.PASCAL from UCSDII)**

| **Debian SYSTEM.VOL** | **Z80 (UCSDII0)** |
| --- | --- |
| **V2 SystemPascalDecoded(fromDebian).txt** | **V2 SystemPascalDecoded(fromUCSDII0).txt** |
| **CBP 19** | **CXP WRITESTR** |
| **CBP 13** | **CXP WRITEINT** |
| **generating NFJ** | **generating FJP** |
| **LSA ‘,’; SLDC 0; CBP 19** | **SLDC 44; SLDC 0; CXP WRITCHAR** |
| **CBP 16** | **CXP READCHAR** |
| **CBP 11** | **CXP EOLN** |
| **CBP 22** | **CXP WRITELN** |
| **CBP 25** | **CXP COPY** |
| **CBP 26** | **CXP DELETE** |
| **CBP 27** | **CXP POS** |
| **CBP 24** | **CXP INSERT** |
| **CXP OPEN** | **CXP OPEN** |
| **CXP EOF** | **CXP EOF** |
| **CXP CLOSE** | **CXP CLOSE** |
| **CXP INIT** | **CXP INIT** |
| **CXP CONCAT** | **CXP CONCAT** |

**10/18/2022:**

* **Look at INTERP.TXT to see implementation of CXP**
* **UCSDII is always forcing to an even address after the LSA**
* **I need to make a copy of VERSIONII.ACCDB**
* **InterpII does not demand a word (even address). (LSA) - *for the start of the string***
* **InterpC does demand a word address. (LCA) - *for the start of the string***
* **How do I select the database that I want to use?**

**10/19/2022:**

* **I DO need SegNamesInDB but do I really need table SegmentInfo?**
* **I need to figure out how many procedures are in CATALOG4.CODE. I should be able to do this by decoding it.**
* **It has about 45 procedures. More like 31 or 32 procedures.**
* **CATALOG4.CODE is listed with a CodeLen of 4247 words (SMS Version IV) = 8494 bytes + 512 bytes would put the end at around $232E.**
* **CATALOG2.CODE is listed with a CodeLen of 6004 (UCSD Version II?)**
* **CATALOG.CODE is listed with a CodeLen of 4389 (UCSD Version ??)**
* **RE SEGMAP: UCSD versions 1.5, 2 store CODELENG as words, but display it as bytes.**
* **COMPINIT appears to have 10 procedures**
* **COMPINIT (on PASCAL.VOL) claims to be flipped but has nonsensical values NrProcs (2560) which should have flipped to 10 (which is correct).**
* **NrProcs seems to be OK for version IV segments (unless flipped)**
* **Code flipping appears to be based on the overall file rather than on individual segments. All of the flipped segments appear to have nonsensical values for *NrProcs***
* **Look at PASCAL.VOL, SYSTEM.COMPILER, all segments**

**10/20/2022:**

* **Did NrProcs get flipped? Flipping needs to go as far as 455 words**
* **FlipSegDic is ONLY flipping the segment dictionary. It is not flipping the procedure dictionary.**
* **'MISCCOD.VOL':'LISTDATA.CODE' ItIsFlipped is returning FALSE.**
* **QUICKSTART.CODE on FORT\_SYS.VOL - Is this really Version IV? SEGMAP says that it is. QUICKSTART procedure directory appears to be flipped EVEN THOUGH the dictionary is not flipped? KLUDGE- try it with the bytes flipped- There must be a clue somewhere that would indicate that this needs to be done! See the “Internal Architecture Guide”. Sex is stored in word 6 of the segment file.**
* **Version II is not setting NrProcs**

**10/21/2022:**

* **Figure how how to get the NrProcs for version < IV**
* **TEST.CODE has 204 bytes: 512 + 204 → 716 ($2CC).**
* **Version II.1 always has something in the “Info” field.**
* **Version IV always has $8000 + in the info field.**
  + **$8900 + might indicate assembly language program**
* **Where does SEGTBL[] get loaded?**
* **Debian SYSTEM.VOL/SYSTEM.PASCAL has info in the INFO field ($4201..4205). UCSDII0/SYSTEM.PASCAL does not have any information in the INFO field. Is this a difference between version II.1 and Version II.0?**

**10/22/2022:**

* **VT-52 TermTypes[cf\_EraseToEndOfScreen] says that it is prefixed, but CrtFuncInfo[cf\_EraseToEndOfScreen] is not set. It never got assigned because of a conflict with “Erase Screen”. Same for “Erase to end of line”**
* **“Move cursor up” is miscinfo as <esc>A BUT TermTypes[tt\_VT52] says that it is #10**
* **I have somehow lost the codes for a SOROC terminal (and other terminals).**

**10/24/2022:**

* **Would be nice to have the ability to directly load .DSK (apple volumes)**

**10/25/2022:**

* **The only interleaves tested are 1..6**
* **The Guess function does not work for .RAW**
* **Volume “RAMDRV” could not be mounted (Directory entry 9 (SYSTEM.START.1.1) in volume “” is marked as bad [DLASTBYTE > BLOCKSIZE])**
* **VT52 is shown twice in the list of possible terminals. H19 with the wrong name.**
* **Filer/Volume is not properly displaying the screen. Appears to have a spurious GOTOXY(52,51). (52,51)= “43”?**
* **Problem occurs because “Y” has been declared to be a *function prefix.* This was occuring when SetTermType set type to VT52**

**10/26/2022:**

* **UCSDII0.RAW does NOT have a backup directory**
* **Move the .DSK files into the correct pSysSrcs2 folder (psyssrcs2\disk images) Currently in C:\Temp\Apple II?**
* **ImageDisk (C language) located in F:\NDAS-I\Floppy Diskette\PSysSrcs2\Imagedisk**

**10/28/2022:**

* **I looked at using ImageDisk (\\xps-8930\ndas-i\Floppy Diskette\PSysSrcs2\Imagedisk) to read “.IMG” and/or “.IMAGE” files (such as those Macintosh files” (\\xps-8930\ndas-i\Floppy Diskette\PSysSrcs2\Disk Images\Mac). Converting these files looks like a very difficult process. I am not going to attempt to do it now.**
* **Use CreateVolume to mount any volume**
* **How do I force display of the optional non-standard parameters dialog?**
* **StandardVolumeFormat seems to be returning incorrect results.**
* **Vols on-line:**

**# VolumeName DeovBlk DOSPath**

**# 4: UCSDII0: [ 2166] F:\NDAS-I\d7\Projects\pSystem\Volumes\UCSDII0.vol**

**--> # 5: UCSD1.5: [ 280] F:\NDAS-I\d7\Projects\pSystem\Volumes\Apple2\I56502.DSK**

* **I need the ability to save newly created disk formats into the settings file.**
* **I still have to finish the APPLE.DSK stuff. Needs to be tested. Might be working (at least for Read)**

**10/29/2022:**

* **ScanSegmentFile never finds the strings that it is looking for. Was stopping search at #0.**
* **I want to find out which segment of UCSD1.5:SYSTEM.PASCAL contains “Welcome…”.**
  + **It ought to start in block 204 within 4 blocks**
  + **Skipping over SYSTEM.PASCAL with an exception**
  + **INITIALIZE is really segment #4**
  + **Block 204 is byte 2048+512 within the file (=$A00)**
  + **“Welcome” should be found at $1104 (4356)**
  + **“U.C.S.D.” should be found at $1140 (4416)**
  + **BUFENDOFFSET = 4096 (too small!)**
  + **I am trashing the end of memory block somewhere. Possibly with the BLOCKREAD. Trying to read in NrBlocks2 (15360 bytes) which is bigger than the allocated blocksize (14848)-- probably occurring because i am doubling the byte count (aCode\_Leng)**

**10/30/2022:**

* **Base Code\_Leng on UCSD-vs-SOFTTECH (i.e. Version pre\_IV -vs- IV)**

**10/31/2022:**

* **When I try to do F(iler E#4:DIS=**

**UCSDII0:**

**Value range error**

**S# 1, P# 5, I# 13061**

**Type <space> to continue**

* **Occurring in SRS**
  + **P1 = 1023**
  + **P2 = 1017**
  + **both greater than 512**
  + **#5 FILEHAND.MESSAGES @ ipc=868 (not 13061)**
* **Trying to boot from I6502 craps out IMMEDIATELY in the debugger. Appears to be caught in a tight loop. Possibly one of the MEMDUMP commands has gotten caught in a loop.**
* **ODD address immediately. Hangs when I try to tab to the second page of Delphi watches. Disable all of the Delphi watches.**
* **Getting the ODD address error as soon as I finish “Specify…” in the Debug menu.**
* **StackOverFlow: RegDumpHex**
* **Something has already mashed memory by the time that I get to the exception handler in MemDumpDW. Cannot even display the call stack.**
* **Occurring because the MemDumpDW function is being called within the Delphi debugger**
* **Goes overboard when I try to display the Delphi call stack. I need to disable the Delphi watches BEFORE they can get called. Done.**
* **Source code never gets displayed**
* **Changed the active database to be the Version I.5 database. This allowed the debugger to display PASCALSY.PASCALSY.**
* **immediately get a stack overlow: UCSDName(‘INITIALI’); SegNameFromTop(56784); CurrentSegName; PointerCheck(0); ProcNumber(0)**
  + **PointerCheck tries to print the CurrentSegName (which causes the stackoverflow).**
* **Trying to boot with the Laurence Boschell version gives an “Abstract error” because fOnPutIOResult is not implemented. The interpreters need a “PutIOResult” function implemented.**
  + **PutIOResult is *Abstract* in Interp\_Common (TCustomPsystemInterpreter)**
  + **PutIOResult seems to be implemented in TCPsystemInterpreter**

**11/1/2022:**

* **F:\NDAS-I\d7\Projects\pSystem\Volumes\Apple2\I56502.DSK (aka UCSD1.5:) appears to be version I.5**
* **I don’t seem to even be able to boot F:\NDAS-I\d7\Projects\pSystem\Z80EM2010\UCSDI5.RAW. Now I can.**

**11/2/2022:**

* **Booting (version II) system.vol and trying to run SETUP.CODE:**
  + **On Delphi gives “No proc in seg-table; s#0, P#43, I# 962; Type <space> to continue**
  + **On Debian gives “No proc in seg-table; s#0, P#43, I# 961; Type <space> to continue**
    - **Debian appears to be getting the CRT characters right. Delphi does not.**
* **“Load Version” doesn’t display the mounted volumes list.**
* **I can boot “F:\NDAS-I\d7\Projects\pSystem\Z80EM2010\UCSDI5.RAW:**
* **BootInfo should contain information about which .ACCDB should be used.**
* **What is happening to force the frmLoadVersion to be shown prematurely? Seems to be a direct result of creating it? Apparently it was caused because the *visible*** **property of the form was true.**
* **I need to disable the *Maintain* function for BootItem.**
* **The *Browse*  function need to list other file types (\*.RAW) etc.**
* **Should remember which *BootParams* was last used.**

**11/3/2022:**

* **Need to be able to set the debugger database file name.**
* **Where does AccDbFileNumber get set?**
  + **Debugger Constructor**
  + **UpdateDebuggerDisplay ?? based on the currently active segment?**
* **What is BootParams.FileName used for? I think that is used to store TheVolume.DOSFileName --not-- DebugDbToUse !**
* **Even when I specify that I want to boot the Version I.4 file, it is still loading the UCSDI51 file.**
* **The BootParams should be saved in FilerSettings.RecentBootsList**
* **Currently working on TfrmLoadVersion.btnOkClick:**
* **FilerSettings.RecentBootsList is not getting updated**
* **Trying to boot version I.4 shows gobbledy gook for the call stacks and for the “Unit Table”. Also shows “Invalid Acc Db file # -1”**

**11/4/2022:**

* **Does fRecentBootParams get freed?**
  + **Loading the FilerSettings file creates 1 BootParams**
* **LoadVersion is not displaying DebugDBToUse**

**11/5/2022:**

* **Where does fRecentBootParams get created? It gets ASSIGNED in TfrmLoadVersion.SetRecentBootParams. fRecentBootParams is LOCAL to LoadVersion.**
* **TFilerSettings.RecentBootsList gets created in TFilerSettings.GetRecentBootsList. Then it gets loaded from the FilerSettings file.**
* **LoadVersion updates fRecentBootParams. If the settings do not match anything already in FilerSettings.RecentBootsList, an Item is added to FilerSettings.RecentBootsList and the newly changed stuff replaces it. If no changes are detected, nothing is added to FilerSettings.RecentBootsList**
* **TfrmFiler.GetBootParams adds a new (blank) TBootParams and passes it into frmLoadVersion. This gets updated *in situ* in LoadVersion and should not get changed.**
* **Using BootParams to set a database should set it to be “Active”**
* **WindowsList should discard anything too old.**
* **Load Version: “Debug DB To Use” is not getting set.**
* **Trying to run another p-System session after the first has been closed always leads to memory failure!**
* **tt\_PeterMiller GoToXY prefix (#$1E = #30) is not found in CrtInfo.FunctionPrefixes**

**11/7/2022:**

* **fThePsysWindow (in FilerMain) is already nil when SaveCrtSettings gets called.**
* **Closing down:**
  + **H(alt the p-System**
  + **Close the debugger**
  + **FilerMain gets notified**
    - **sets frmPCodeDebugger to nil**
    - **FreeAndNil(fInterpreter)**
    - **FreeAndNil(fThePSysWindow)**
      * **fOnSaveSettings(self) == TfrmFiler.SaveCrtSettings**
    - **fThePsysWindow is already nil so FilerSettings.TermType does not get set**
* **I should be able to simplify DefaultCRTInfo**
* **Need to re-Display “Vols on-line:” after mounting boot volume**
* **fFilesLoadedList[0].TheSegName is garbage (in version I.4 at least)**
* **'F:\ndas-i\d7\Projects\pSystem\Reports\DumpDebugInfo.txt'**
* **“Invalid ProcNum: -1” coming from ProcNamesF.** 
  + **aSegTop = 0.**
  + **aProcNr = 0 and**
  + **SegP = 0**
* **ProcNameFromSegTop MsProc = 0 & aSegTop = 0. This is while trying to display the dynamic call stack. The call stack still looks like Gobbledygook:**
* **MSCW Proc @IPC**
* **X ÓW.Invalid AccDb file #: -1 0**
* **TheSegTop = 0 for all segments**
* **filerSettingsForm is getting shrunk?**

**11/8/2022:**

* **SEGP starts out at 0 and is getting used by the debugger when displaying the stack**
* **aSegTop = 64612 : SegName = 'Q'#0#4#0'ÓW' (fn = 0; SegIdx = 0)**
* **TheSEGNAME is getting garbled from NewSegName which is a parameter to UpdateSegStuff called from TIIPsystemInterpreter.ReadSeg**
  + **Which is taken from** 
    - **Move(SD.DICT.SegNamesII[SEGNUM], SegName, IDLEN)**
* **Switching to a different terminal doesn’t seem to take effect until you reboot.**
* **CRTInfo.CRTFuncInfo is an array[TCRTFuncs] of TFuncInfo.**
  + **Each TFuncInfo holds just one Pfxed/ch value;**
* **Is KeyInfo coming ONLY from SYSTEM.MISCINFO?**
* **Changing to vt52 does not get handled. <Esc> did not make it into CrtInfo.FunctionPrefixes**
* **When I try to guess, CRTFuncInfo seems to be messed up**
* **After initial call to SetTerminalType(tt\_Vt52) in Load\_pSystem :**

**CRTFuncInfo: ((False, #0), (False, #27), (False, #24), (False, 'P'), (True, 'J'), (True, 'K'), (True, 'A'), (True, 'C'), (True, 'B'), (True, 'D'), (True, 'E'), (True, 'I'), (True, 'H'), (True, #0), (False, #0), (False, #0), (False, #0), (False, #0), (False, #0), (False, #0), (False, #0), (False, #0), (False, #0), (False, #0), (False, #0), (False, #0), (False, #0), (False, #0))**

* **Version I.5 does not display the call stacks or current source code**
* **Do I need to re-enable DumpDebugInfo? No. I don’t think so.**

**11/9/2022:**

* **Where does pCodeDatabaseFileNameS get initialized? It gets initialized in TfrmPCodeDebugger.Create. Info should be coming from the debugger settings but this is a PUBLIC setting-- not a PUBLISHED one.**
* **SegNamesInDB gets populated by LoadSegmentNames (which is no longer called) by scanning the SegmentInfoTable.**
* **I think I either need to revise the procedure which populates from the debugger database or add another procedure to be able to populate the SegNamesInDB…**
  + **LoadSegmentProcName??**
  + **ScanSegmentProcNames -- LoadSegmentName**
  + **I think that maybe LoadSegmentName *should be/is* doing this-- not sure.**
  + **LoadProcedureNames never gets called? Maybe because pCodesDatabaseFileNameS.Count = 0? Correct.**
  + **Where does pCodesDatabaseFileNameS get created? Make sure it is not in the debugger.** 
    - **GetpCodesDatabaseFileNameS never seems to get created. It gets created in ConstructStringListForVersion (called by SetVersion(vn\_VersionI\_5) and because there are no version I.5 databases, nothing gets put into fpCodesDatabaseFileNameS.**
    - **I think that BootInfo should allow a database to be used in multiple versions.**
    - **TListView / TListItems**
* **lvVersions has the following items: Succ(Low(TVersionNr)) to High(TVersionNr)**
* **The version numbers being displayed in lvVersions are all wrong**
* **No TListItem was created**

**11/10/2022:**

* **Volume Conversion is not working**

**11/11/2022:**

* **THIS is the volume that I want to convert:**

**F:\NDAS-I\d7\Projects\pSystem\Volumes\15SYS1.VOL**

* **Any other copies of 15SYS1.VOL are crap**
* **Now I would like to compile the system under version I.4 a get a listing**
* **UCSDI4.RAW: trying to compile SETUPI3.TEXT does not run under version I.4**
* **Field ‘PARAMSIZE’ not found (in PSYSTEMX.ACCDB). Database = pSystemX.AccDB**
* **Nothing gets saved from PSYSTEMX.ACCDB because there is nothing in any of the memo fields. Do I have a later and better DB for version IV somewhere? Possibly System4.accdb-saved or a CD and/or DVD**

**11/14/2022:**

* **Getting an “Unspecified Error” when it tries to open “F:\NDAS-I\Old Databases\ACCDB\20220521\FilerPro.accdb”-- but I can open it in MSAccess. I can also open in CatalogACCDBDatabases when I start at a lower level in the tree.**
* **Remember to fix backup to do “F:\NDAS-I\Old Databases\ACCDB”**
* **The earliest version of INTERPII.PAS appears to be 1/12/2021**
* **Possibly use V4-Filer.accdb** 
  + **stored in F:\NDAS-I\Old Databases\ACCDB\20210505\AccDb\ which has a 2/17/2021 date**
* **Only databases updated prior to 1/1/2021**
* **PSYSTEMX,ACCDB has 346 procs**
* **F:\NDAS-I\d7\Projects\pSystem\AccDb\VersionIV.accdb has 530 procs (many are not OS procs)**

**11/15/2022:**

* **Cannot save memos from: 'F:\NDAS-I\d7\Projects\pSystem\AccDb\PSYSTEMX.accdb'**
* **ENilPointerRef in CXP 0:INITIALI.XXX, DbgCnt = 103**
  + **History is all INITIALI.XXX @ nnn**
  + **Where is the “XXX” coming from?**
  + **This is procedure # 0 in INITIALI? Laurie Boschell version I.4 seems to work fine.**
  + **Call stack:**
    - **PASCALSY.PASCALSY @6 → INITIALI.INITIALI @ 207 → INITIALI.INITHEAP @ 3**
* **'NIL pointer reference: JTab = 0, @ IPC: 22**
  + **INITIALIZE.XXX**
    - **Appears to be trying to call procedure 0 in INITIALIZE**
  + **CXP -> StaticLInk -> ProcLexLevel(0)**
  + **DbgCnt = 103**
  + **Trying to do SYSCOM^.GDIRP = NIL**
    - **Occurs in the STO opcode**
    - **Trying to store 0 at $FD3C (= word $7E9E)**
    - **Gets into 5:INITIALI.INITHEAP correctly**
    - **After the STO the next apparent instruction is an LDA**
    - **Problem occurs when the debugger attempts to display the dynamic call stack? Or does it?**
    - **ProcLexLevel(0)**
    - **DisplayCallStack is getting JTab = 9 from SysComIIPtr^.JTab and that is where the NIL pointer is coming from.**
    - **Has JTab ever been changed from 0? In the Debugger registers dump, JTab shows as $D3EE**
    - **Occurring in the call ProcBase(JTab) in RegValues. Maybe it should be using the values from the MSCW?**
    - **JTab should get set on the very first *call***

**11/16/2022:**

* **GetJTab is getting called BEFORE SetJTab. Call coming from the decoder?**
  + **Call coming from pCodeDecoder DisplayWatches**
  + **JTab gets set in *call*  to 64712 (or is it 54254?)**
  + **The debugger appears BEFORE JTab gets set--?**
  + **But JTab has somehow gotten set to 64712?**
    - **Syscom = 64812, @JTab = @SysCom + 16 = Bytes[64812+16]: 64712**
    - **Something is setting JTab to zero? Perhaps when MISCINFO gets loaded? No. SysCom.Miscinfo.JTab is zero before and after**
    - **JTab is located at Words[32414]**
    - **gdirp should be located at SYSCOM+8 -> 64812+8 (64820) -> Words[32410]**
    - **fLastJtab = 54252 BUT JTab = 0? How can that happen? Something is setting JTab to 0? Returning from a procedure perhaps?**
    - **SAME ADDRESS FOR GDIRP AS FOR JTAB!**
      * **Is SYSCOM loaded at $FD2C? That is the value that SYSCOM has. When trying to zero GDIRP, the code is doing an INC of 8 but it is adding 8 words rather than 8 bytes.**
      * **The base address is 64812**
  + **debugger seems to be loading info from the wrong database**
* **The List SegNames/ProcNames is listing the wrong database. Temporary? Possibly due to creating new BOOTPARAMS -vs- using pre-existing BOOTPARAM<S.**
* **Getting an exception in RBP when exiting FINIT: Base $242 out of range ($C340..$D808) Segment:0, ProcNum: 3, RelIPC : 82 at #PASCALS.FINIT**
  + **Where did BASE get pushed onto the stack?**
    - **Maybe in *call***
    - **Call: Base = 0 pushed onto stack when SP = $200**
    - **Call: Base = $D808 pushed onto stack when SP = $200**
    - **Call: Base = $C368 pushed onto stack when , SP = $01FE**
    - **RBP: Base = $C368 pushed onto stack when, SP = $01FE**
    - **Call: Base = $C368, SP = $01FE**
    - **RBP: Pops Base off of stack. Base = $0242?, SP=$01FE**
    - **$0242 is pushed onto the stack in PASCALSYS.FINIT, RelIPC = 6. This is immediately following an INC. It is trying to calculate the address of the FSTATE field within a FIB. It assumes that the FIB is located at $0200. This is occurring on the call to FINIT LWINDOW**
    - **The FSTATE word is BYTE offset = 6 or WORD offset = 3 in a FIB.**
* **I need to allow the LocalVariables window to display using byte offsets or word offsets.**

**11/17/2022:**

* **Base has been set to $0242 by the time it reaches DbgCnt 206**
* **Problem occurs on (during ?) the second call to FINIT.**
* **Why am I not getting a breakpoint on the call to FINIT? Expecting that FINIT is procedure #3. PASCALSY.FINIT**
* **Stepping over the CXP even though it has a breakpoint set in it. This occurs because STEP-OVER keeps stepping until it gets back to the original procedure.**
* **The RBP changes the SP from 506 to 508 (or is it 510 ($1FE) -> 508 ($1FC)?)**
* **The FIB is located at $23C**
* **RBP is getting the bad base from $01FC**
* **$01FC is frequently on the top of the stack**
* **I never see location $1FC getting changed to contain $0242**
* **The SP does not get changed by Sp := WordAt[WordIndexed(Mp, MS\_SPw)];**
* **$0242 gets put onto the stack by FSTATE := FJANDW and is left on the stack at $01FA**
* **$0242 again gets put onto stack by FSTATE := FNEEDCHAR and is popped off by the STO instruction, however, it is still stored in location $01FC.**

**11/18/2022:**

* **The first time through the RBP, SP is not changed. SP remains at $01FC**

**(STATLINK:$D808; DYNLINK:$C358; MSJTAB:$D3EE; MSSEG:$D804; MSIPC:$16; LOCALDATA:($1FC))**

**'Regs(hex): E712:52, RelIPC=$0052, Kp (SegBottom)=$C340, Sp=$01FC, Mp (MSCW)=$C342, Np (HeapTop)=$023C, LocalVar=$C34C, GlobVar=$C342, SegP (Segtop)=$FD2A, JTab=$E76E, IpcBase=$E712, Syscom=$FD2C'**

**DbgCnt = 146**

* **The second time through RBP, SP gets changed from $01FE to $1FC**

**(STATLINK:$D808; DYNLINK:$C358; MSJTAB:$D3EE; MSSEG:$D804; MSIPC:$29; LOCALDATA:($1FC))**

**'Regs(hex): E712:52, RelIPC=$0052, Kp (SegBottom)=$C340, Sp=$01FE, Mp (MSCW)=$C342, Np (HeapTop)=$027A, LocalVar=$C34C, GlobVar=$C342, SegP (Segtop)=$FD2A, JTab=$E76E, IpcBase=$E712, Syscom=$FD2C'**

**DbgCnt = 206**

* **On the second entry, the SP has changed from $01FC to $01FE**
* **A total of 60 instructions have been executed**
* **Using the Version II interpreter (first time):**

**(STATLINK:$ED9D; DYNLINK:$E420; MSJTAB:$EB8B; MSSEG:$ED9B; MSIPC:$16; LOCALDATA:($1FE))**

* **Using the Version II interpreter (second time) SP does not get changed (SP = $01FE):**

**(STATLINK:$ED9D; DYNLINK:$E420; MSJTAB:$EB8B; MSSEG:$ED9B; MSIPC:$29; LOCALDATA:($1FE))**

* **The Peter MIller Version 2.0 boots and runs from F:\NDAS-I\d7\Projects\pSystem\Volumes\Linux Vols\system.vol**
  + **although the CRT controls are not good.**
* **The Peter MIller Version 2.0 boots and runs from**

**F:\NDAS-I\d7\Projects\pSystem\Z80EM2010\UCSDII0.RAW**

* **In the Version II system, SP is $01FE on entry both times.**
* **When does SP get changed to $01FC?**
  + **Many times.**
* **Has the problem already occurred before RBP is called the first time?**
* **CXP takes one word off the stack**
* **CLP left the stack untouched? DbgCnt = 91, INITIALI.INITHEAP, RelIPC = 0. That is OK, I think because there were no parameters.**
* **The INC @ INITIALI.INITHEAP SP, IPC = 3 seems to have done something mysterious. DbgCnt = 94-- pushed something onto the stack and did not increment anything. Cannot reproduce. Pops 64812 ($FD2C), gets a byte 8, adds the 8, pushes the sum ($FD34). On the 2nd pass through (DbgCnt = 107), it only added 6 rather than 8. That is OK, the 2nd pass is coming from PASCALSY.FINIT which is only adding 6. I can't reproduce any of this. Looks like this was actually a LDCN rather than an INC.**

**11/19/2022:**

* **When LoadVersion “Boot Volume FilePath” gets changed, it may find a “Laurence Boschell” version rather than a “Peter Miller” version.**
* **FILER.INI is losing some BOOTPARAMS items.**
* **Am I using DEBUGSETTINGS when I should be using FILERSETTINGS?**
* **Specify sets Tag to -1 which causes a new TBootParam to be created**
* **GetBootParams is returning NIL when SPECIFYing**

**11/21/2022:**

* **The RBP has popped something off of the stack-- presumably the original BASE which had been pushed onto the stack when the FINIT procedure was called**
* **The NGI in INITHEAP did not seem to properly negate 0001. DbgCnt = 102. Its fine.**
* **CXP (to FINIT?) took three things off of the stack:0000 0278 023C and put one on $C368**
* **Maybe put a break into *call* and see when *base* gets pushed onto the stack.**
* **On the second pass through FINIT, the SP starts out at $01FC and finishes at $01FE**
* **FWINDOW is $0278. See where it gets set in FINIT. It gets passed into FINIT on the 2nd call to FINIT.**
* **The STB in FINIT is getting passed a WINDOW (from LWINDOW) that is probably bogus. It is only sized to contain 1 byte. LWINDOW = $0278 and we are trying to put something into FWINDOW^[1]**
* **This was all the result of STB popping 3 parameters but version 1.4 only knew about two**

**11/22/2022:**

* **STB leaving three things on the stack?**
* **STB only needs two things from the stack: the address and the byte to store**
* **Getting an odd IPC in a LSA**
  + **'Procedure name: 3:INITIALI.INITUNIT'**
  + **'Regs(hex): D148:15, RelIPC=$0015, Kp (SegBottom)=$C350, Sp=$01FC, Mp (MSCW)=$C352, Np (HeapTop)=$036E, LocalVar=$C35C, GlobVar=$C368, SegP (Segtop)=$D804, JTab=$D236, IpcBase=$D148, Syscom=$FD2C'**
  + **'Dynamic CallStack: INITIALI.INITUNIT @ 21**
* **I’m getting a divide by 0 error in FWRITEINT @ O #114.**
* **I’m getting format errors related to “%s”**

**11/23/2022:**

* **The element size used in IXA seems to be a size measured in words.**
* **I get another “Base out of Range” error in RBP in PASCALSY.COMMAND @ 174**
* **FWRITECH is trying to write a character to a FIB that is not open**
* **I seem to be lost in a series of calls to FWRITECH**
* **The BASE out of range error occurs when attempting to exit COMMAND**

**11/25/2022:**

* **I need to assign both fGetBaseAddressFunc and fGetByteFromMemoryBased**
* **I’m still decoding the wrong thing somehow.**
* **Decoder gets the first procedure correctly**
* **For procedure #2, appears to get the correct (or at least plausible) values for these: EnterIC, ExitIC, ParamSize, DataSize, Lastcode**
* **GetByteAt is called as GetByteAt(64678) from TpCodeDecoderII.Decode(64678,false,dfShortFormat)**
  + **which calls GetByteFromMemoryBased(64678,64678)**
  + **fGetBaseAddressFunc IS assigned, but,**
  + **fOnGetByte3 is NOT Assigned.**
  + **Call is coming from MemDumpDW(0, wt\_OpCodesDecoded, 0, ‘’)**
    - **which is coming from the debugger DisplayWatches**
* **Be aware of DecodeToMemoryDump (dmd). dmd assigns the OnGetByteBased procedure which is the root cause.**
  + **This is created in TCPsystemInterpreter.DecodedRange.**

**11/26/2022:**

* **TDecodeToMemDump is created by DecodedRange(0, 50)**
* **TCPsystemInterpreter.DecodedRange called by MemDumpDW**
* **see: “OnGetBaseAddress” which gets assigned in TCPsystemInterpreter.DecodedRange**
* **OnGetBaseAddress gets assigned to the Decoder in TDecodeToMemDump.Create**
* **Both**
  + **Assigned(self.fOnGetByte2) = false**
  + **Assigned(self.fOnGetWord2) = false**
    - **where self = TDecodeToMemDump during creation**
* **Somehow TpCodeDecoder is getting used rather than TpCodeDecoderII**
* **pCodeDecoder is getting created by the interpreter by GetpCodeDecoder when DecodeToMemDump gets created-- needed by DecodedRange- needed by MemDumpDW-- debugger DisplayWatches**
* **pCodeDecoder gets created again when the debugger is created. Debugger.UpdateDebuggerDisplay ← Debugger.DisplayWatches ← MemDumpDW(64678 ← DecodedRange(64678 ← Decode(64678 ← GetByteAt(64678 ← GetByteFromMemoryBased(64678, 64678) BLAM**
* **This all originates from:**
  + **WatchName = 'MemDump(AbsIPC, 'O','Opcodes Decoded', 0)'**
* **GetByteFromMemory(56706,56706) ← TpDecoderII.GetByteAt(56706) ← TpCodeDecoder(56706,0,...)** 
  + **Note that this is NOT TpCodeDecoderII(56706,0,...)**
  + **Using XfpCodeDecoder**
  + **The call to Decode(... is using the inherited version of Decode(... which does not have the BaseAddr parameter.**
  + **XfpCodeDecoder is created in TfrmDecodeWindow.Create and OnGetByte3, OnGetWord3 are assigned there.**
    - **I should be able to assign OnGetBaseAddress and OnGetByteBased**
* **It is dubious whether or not the Decode Window will always be using the correct version of the BaseAddress. Right now it is always using IPCBASE which may not always be correct.**
* **FWRITECH is always getting called on a FIB which is not open. Appears to be the FIB for SYSTERM.**
* **SYSTERM ought to get opened in INITFILES**
* **At DbgCnt ???? I hit 'RBP: Base $C368 out of range ($D7F4 .. $D808)'**

**11/28/2022:**

* **Why does LAE have TWO calls to FETCHUB()?**
* **LDO(V1.4, V1.5?) ⇐> LAE (V2?)?**
* **The call to FOPEN goes off of the rails**
  + **Does the SCAN function have different params with VERSION II?**

**11/29/2022:**

* **Does SYSTERM ever get opened? Yes,**
* **Finally got version I.4b to boot under the Peter Miller derived OS.**
* **Got 'RBP: Base $C368 out of range ($D7F4 .. $D808)' error when I tried to halt the OS.**
  + **Appears to be occurring in PASCALSY.COMMAND @ 174**
  + **CASE STATE OF (in COMMAND) appears to be using a value of**
* **UNITBUSY removing too much from the stack? No. I don’t think so.**
* **This should be the set [UPROGNOU,UPROGUOK] (but it is not)** 
  + **Set+2 @ $01FA [ 506]: 1 words: [3..4]**
* **Stack when entering COMMAND.**
  + **Words @ $01FE [ 510]: C368 0000 0001 0001 0000 0000 0000 0026 1C66 0035 03DC 0037 07A4 003B 05D4 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000**
* **Stack just before exiting COMMAND**
  + **Words @ $01FE [ 510]: C368 0000 0001 0001 0000 0000 0000 0026 1C66 0035 03DC 0037 07A4 003B 05D4 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000**
* **I could try running it with Version 1.5 or Version 2 or using the Laurie Boschell version.**
* **Backup is backing up a bunch of “DB Contents” to “Other Sources\pr20221129” that probably didn’t need to be backed up (“.pcode”, “.var”, “.pas”)**

**11/30/2022:**

* **Prior to calling command:** 
  + **Words @ $0200 [ 512]: 0000 0001 0001 0000 0000 0000 0026 1C66 0035 03DC 0037 07A4 003B 05D4 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000**
* **The *Base* that gets pushed is 50024 ($C368).**
  + **At the time that it gets pushed:**
    - **Kp = 55302 ($D806)**
    - **BaseMp = 55304 ($D808)**
* **Even after Kp ($D7F4) and BaseMP ($D808) get updated, Base is still out of range.**
* **“Load Version” does not remember the version number**
* **Trying to boot to Peter Miller Version I.5 leads to:**
  + **'RBP: Base $ 0 out of range ($BC12 .. $DBE2)'**
  + **Appears to be trying to exit INITIALI.INITIALI**
* ***Comment* does not get passed through**
* **Version I.5 mashes *base* somewhere in *INITIALIZE.***
* **Peter Miller Version seems to get as far as the *halt* (but CRT controls do not work) and the Assert(not odd(Ipc)) in LSA trips.**
* **Peter Miller CRT lead-in to screen is #$19**
* **Peter MIller not recognizing that #$1E is the GOTOXY flag for Peter Miller**
* **Is GoToXYPrefixChar actually being used?**
* **This assumption: “*for VI.4 assume that if it is one of the movement characters, it must be prefixed by the LeadIn char*” may not always be true**

**12/1/2022:**

* **The TermType gets set to VT-52 from the FILERSETTINGS which is going to mess up the Guess function.**
  + **Need to call the LoadMiscInfo function in the MiscInfoUnit.**
  + **And then call LoadCrtKeyInfo(:InBufPtr, CrtInfo, KeyInfo)**
  + **And THEN do the *Guess* stuff.**
* **When I close down the Laurence Boschell version I.5, I get an “Odd Address” error in PASCALSY.PASCALSY @ IPC = 36**

**12/2/2022:**

* **The CRT info reports looked OK as of 10/24/2022 (“CrtInfo (6).CSV”)**
* **The CRT info reports looked OK as of 11/02/2022 11:21 AM (“CrtInfo (12).CSV”)**
* **The CRT info reports look wrong by 12/2/2022 11:45 AM (“CrtInfo (15).CSV”)**
* **SetFunctionPrefixes is screwing everything up.**

**12/3/2022:**

* **Version I.5 (Peter Miller):**
* **Just prior to CXP to call INITIALIZE.INITIALIZE:**
  + **Words @ $0200 [ 512]: 0001 1F30 0011 0036 0012 1C66 0021 03DC 0023 0972 0028 0836 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000**
* **Immediately following entrance to INITIALIZE.INITIALIZE**

**Words @ $01FE [ 510]: DBE2 0001 1F30 0011 0036 0012 1C66 0021 03DC 0023 0972 0028 0836 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000**

* **Seems like INITCHARSET is getting called regardless of the value of SYSCOM^.MISCINFO.HAS8510A (only sometimes?) (The decoded p-Code was wrong)**
* **Following the first call to INITIALIZE.INITIALIZE, the SP = $200 rather than $1FE ($200 - $1FE = $002) which is expected.**
* **Seems like the statement if SYSCOM^.MISCINFO.HAS8510A took something off of the stack**
  + **Three things get put onto the stack for LDP**
    1. **the word address (=$FD2C)**
    2. **the bits / element (=1)**
    3. **and the bit number (=1)**
  + **and the result gets put back onto the stack ($0000).**
  + **and that result gets popped off by the FJP (all seems OK).**
* **INITIALIZE.INITIALIZE seems OK after the first pass through except that the SP may be incorrect -- because the BASE (50024 = $C368) gets popped off of the stack**
* **RBP gets called a bunch of times. The PushBase and PopBase seem to be synchronized until around DbgCnt = 13779 and fNrPushes = 120. This is occurring on the RBP that is exiting INITIALI.INITIALI.**
* **Seems like something got popped off of the BaseStack that shouldn’t have.**
* **The Peter Miller Version II.0 does not incur this problem.**
* **Weak evidence implicating FWRITEIN in the problem (P #13, IPC:150)**

**12/5/2022:**

* **The call to FWRITESTRING may be mis-positioning the stack.**
* **Exiting FWRITEINT pops the stack removing the needed *base.***
* **There are two calls to FWRITEIN:**
  + **SP after entry to FWRITEIN = $01FC**
  + **SP just prior to exiting FWRITEIN = $01FC**
* **ProcLexLevel(NewJTab) <= 0 is suspicious. Any chance of ProcLexLevel returning an incorrect negative result?**
* **Exiting INITIALI.INITIALI, something ($0368) has been popped off of the stack. Trying to mousetrap the POP of $C368 occurs in PASCALSY.FINIT IPC 146**
* **The LoadVersion function should have an option to Browse for a debug DataBase.**
* **I can boot the version I.4 system on SurfacePro and it doesn’t crash until I am trying to exit the system? This went away when I turned off optimization on the Surface Pro.**

**12/6/2022:**

* **Running on Surface Pro was not compiling “Nice place for a break” lines because optimization was turned on.**
* **With optimization turned off, both versions (SurfacePro and XPS) behave identically-- messed up *base* value on the stack.**
* **How come there are TWO calls to close on the Miscinfo file? The second call to CLOSE unceremoniously exited to INITSYSCOM.**

**12/7/2022:**

* **The procedure/IPC number do not get fixed until ret() gets called.**
* **Network access to SurfacePro might not work because I changed drive H: to drive F:**

**12/8/2022:**

* **Try to break entering/exiting FWRITEIN and log the current stack**
* **Version I.4 currently runs but crashes with an** 
  + **'RBP: Base $C368 out of range ($D7F4 .. $D808) (Exception): Segment:0, ProcNum:42, RelIPC:174'**
* **I want to display the Breakpoint watch info based on a register value.**
* **See how Inspector Info address gets shown**
* **H(alting the system to exit gets the RBP: Base out of Range Error…**
* **H(alting also manages to skip the *CloseLogFile***
* **Displaying FGET rather than FWRITEIN**
* **BREAKPOINTINFO not displaying the hex value correctly**
* **BREAKPOINTINFO not remembering the IDENT correctly**
* **WordAt[LowAddr] → WatchAddrFromExpression(AddrExpr)**
* **revert xLowAddr back to LowAddr**
* **I may not be saving BreakPointInfo: Indirect, DisplayAs, NrBytes, PassCount and/or Param. Also make sure that the *Assign* is copying these values.**

**12/9/2022:**

* **TfrmPCodeDebuggerCustom.WatchAddrFromExpression(AddrExpr: string)**
* **It is asking about do you want to log to a file twice.**
* **LogFileName is kept in TBreakList**
* **EditBreakPointInfo → SetAccDbFileName → SetSegIdx(0) → RepopulateProcNames -- Crash with AV: ProcNamesInDB[AccDbFileNumber, Value, i]**
* **Do you want to log to a file? Asking about a .LOG file rather than a .CSV file.**
* **After selecting an output file name, the Debugger display does not update.**
  + **TBreakInfo.LoggingToAFile means that the file has been opened and Logging is occurring.**
    - **It comes from TBreakList.fLoggingToAFile**
  + **TBreakList.fLoggingToAFile is set to true when TBreakList.OpenLogFile is called.**
  + **TBreakList.fLoggingToAFile is set to false when TBreakList.CloseLogFile is called.**
  + **TBreakList.LogFileOpen comes directly from TBreakList.fLoggingToAFile**
  + **TBreakList.SetLogFileOpen(const Value: boolean) opens OR closes the log file**
  + **TBreakInfo.OpenLogFile calls TBreakList.LogFileOpen which calls SetLogFileOpen(const Value: boolean);**
  + **Is TBreakInfo.LogToAFile the SAME as TBreakInfo.LoggingToAFile?** 
    - **TBreakInfo.LoggingToAFile comes from TBreakList.fLoggingToAFile**
  + **BrkItem.LogToAFile is NOT the same as BrkItem.LoggingToAFile**
    - **LogToAFile: this is TRUE if the users WANTS to log to a file**
    - **LoggingToAFile: This is true if we are actually currently logging to a file**
* **Not asking for the LogFileName nor is it opening the created file when finished.**

**12/10/2022:**

* **RepopulateProcNames -- Crash with AV: ProcNamesInDB[AccDbFileNumber, Value, i]**
* **Length(ProcNamesInDb) = 0**
* **Length(DebuggerSettings.pCodesDatabaseFileNameS.Count) = 0**
* **Debugger Settings: 'F:\NDAS-I\D7\Projects\pSystem\DEBUGGER.INI'**
* **Where does *pCodesDatabaseFileNameS* get loaded?**
  + **Probably during the construction of the debugger. No. Not true.**
  + **It is constructed via *ConstructStringListForVersion***
  + **The proposed database file name is probably incorrect.**
    - **Selecting an alternate (proposed) name probably needs a separate pass over the database list.**
* **I’ve got a memory leak. Stopped happening for no good reason.**

**12/12/2022:**

* **Are the TCollection exceptions getting hidden?**
* **Which database should I be using for version II?**
* **Getting a memory leak on SurfacePro running Version II. Not dependable.**
* **pCodeDebugger @ Breakpoint #-1**
* **“Any unseen procedure” is not breaking**
* **Seems to be forgetting “Break Kind” on “ANY UNSEEN”**
  + **This is occurring on “Brk := dbUnknown;” in OldDebugger**

**12/13/2022:**

* **Occurring between DbgCnt 10,574 → 12,656**
* **$D808 getting popped off in INITIALI.INITSYSCOM, DbgCnt 10878, IPC = 43, OpCode = CSP : CSPFLC**
* **Which is the correct database for Version II?**
  + **Most of the records with the latest update are in F:\NDAS-I\d7\Projects\pSystem\AccDb\VersionII - UCSDII-RAW.accdb**

**12/14/2022:**

* **The segment name USERPROG was found in two different databases…**
* **The database VERSIONIV.ACCDB seems to be the real version even though PSYSTEMX.ACCDB has later dates.**
* **Getting the “Odd Address” error in the debugger when attempting to display stack.**
* **The decoder is screwing up. Might be better to revert to a few weeks ago.**
  + **The AbsIPC = $F2C0 but the BaseAddress is $17EC**
  + **Look at the addresses being used by the interpreter**

**12/15/2022:**

* **Does POOLOUTSIDE need to be set before entering OS?**
* **Trying to boot “F:\NDAS-I\d7\Projects\pSystem\Volumes\SYSTEM4.VOL” quickly goes to SYSTEM HALT at DBGCNT=362: USERPROG.FILEPRES @40 (FilePresent(‘SYSTEM.MENU’, dummy, dummy)**
* **Getting a POOL OVERFLOW when I try to edit anything in “\\Hplaptop\psys\psystemy.vol”**
* **F:\NDAS-I\d7\Projects\pSystem\Volumes\SYSTEM4.VOL: FILE\_PRESENT: neither source of p-Code matches the database.**
* **The source code to FILE\_PRESENT is in P403\_1F.VOL: INITUTILS.TEXT. Neither source or p-Code matches the database.*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****
* **There is a listing in F:\NDAS-I\d7\Projects\pSystem\Volumes\UTILITY.VOL: KERNELL.TEXT. This code also does not match. Chances are that the code that I am stepping through isn’t really FUNCTION FILE\_PRESENT**
* **Procedure NUmber 2 in Kernel is Exec\_Error. Procedure number 2 in USERPROG is OS\_INIT.**
* **None of the records in the database actually match the code that I am trying to step through. Is there another version of the DB that matches better?**
* **CPL is trying to call procedure #2 (OS\_INIT) but actually calling procedure #4 (BOOT\_ERROR). The (newly) decoded p-Code shows CPL 4 -- it does not show CPL 2.**
* **What do the BAT files do?**
  + **PSYS.BAT: PSYSTEM.COM PSYSTEM.VOL**
  + **psys2.bat: psystem.com psystem.vol**
  + **PSYST.BAT: psystem.com psystemY.vol**
  + **PSYSX.BAT: psystem.com psystem.vol**
  + **TD.BAT: TD.EXE DOSBOOT.COM PSYSTEMY.VOL TESTING.VOL**
  + **td2.bat: td.exe dosboot.com utilcod.vol rl0.vol system4.vol**
  + **TD4.BAT: TD.EXE PSYSTEM.EXE**
  + **TESTED.VOL UTILCOD.VOL RL0.VOL**
  + **PSYSTEM.EXE : IV.12 B 12/31/1979**
  + **PSYSTEM.COM : IV.2.1 R3.3 04/21/1988**

**12/16/2022:**

* **The question is: Why is the initial code going directly to BOOTERROR? I think that is really NOT BOOTERROR.**
* **Not using the BOOT item that is passed in.**
* **Why is the decoder going wild?**
  + **What is the BASEADDR that gets passed in?**
* **Search for an OS\_INIT listing having a procedure number of 4.**
  + **Use: F:\NDAS-I\d7\Projects\pSystem\Volumes\UTILITY.VOL - KERNELL.TEXT**
* **Maybe its really not OS\_INIT that I am in.**
* **See: F:\ndas-i\d7\projects\psystem\decoded\Decode of System.pascal from psystem.exSYSTEM4-VOL.txt**
* **Procedure #4 in segment USERPROG does appear to be OSINIT in [IV.12 B]**
* **Decoder is still not always decoding the right stuff**
* **The system halt is coming from inside “CPL 6”. Caused by a UNITREAD(0); (System Halt in CallIO). USERPROG.Procedure 6 is currently listed (incorrectly) in in the database as FILE\_PRES[ENT] at IPC: 132. I cannot find anywhere in the source codes that I have available “UNITREAD(0)”. Somewhat similar to BOOTERROR.]**

**12/17/2022:**

* **Can I do segmap of SYSTEM4 SYSTEM.PASCAL?**
* **I am getting a ProcAddr of 0 which leads to RangeCheck exception**
  + **ExitIC := GetWordAt(ProcAddr-2) \* 2;**

**12/19/2022:**

1. **There seem to 49 CSP operators**
2. **See InterpIV line 7862 for the initializations of the CSP operators**
3. **Can I use LCO to get the string constant values?**

**12/20/2022:**

* **GetByteAt is using the base address relative to the procedure rather than the base address relative to the constant pool.**
* **Do I need to use the fGetBaseAddressFunc?**
* **BuildDbDB calls fpCodeDecoderII.Decode (^Q0)**
* **BuildDbDb ProcessSoftechSegment calls** 
  + **fPCodeDecoder.OnGetByte3 (^Q4)**
  + **fPCodeDecoder.OnGetWord3 := GetWordAt (^Q4)**
* **TpCodeDEcoder:**
  + **fGetByteFromMemoryBased: TGetByteFromMemoryBased;**

**12/21/2022:**

* **Where did the code for pDecoderUnt come from? Did I create it from scratch or is it based on something else? I think that I created it.**
* **Trying to flip something. Why?**
* **Getting a range check violation on CodePoolBase := FullAddressToLongint(Base);**

**when booting without “Enabled External Pool”**

* **Version IV.0 says that the pools is outside tdx**
* **P#4 is OSINIT**
* **TIVPsystemInterpreter.MOV does not include “SRC” in line 2666. Does that imply that it is assumed to be 0? This used to refer to “SS” which I think is always 0. On 20210128 the “SS” was removed.**
* **Attempting to call userprog P #4 but the p-Code for P#2 (BOOTERRO) IS BEGIN SHOWN. Somehow the wrong database is being displayed. INitialization code in the debugger (line 1077) is opening the wrong DB.**
* **DB 'F:\NDAS-I\d7\Projects\pSystem\DB Contents\V-4.0\V-4.12\' is listed as active and has the correct version number. TDebuggerSettings.SetActivePSystemVersionNr should set the active version number. See “fSelectedAccDbIndex”-- See “TDebuggerSettings.SetActiveDBName”**
* **Still displaying the BOOTERROR proc #2 when it ought to be displaying the OSINIT proc #4**
* **I’m getting a memory leak error**

**12/22/2022:**

* **See the file “F:\NDAS-I\d7\Projects\pSystem\Docs\INT 21h - The general function dispatcher.html” for descriptions of the DOS interrupt functions.**
* **I am pretty sure that I once had a user manual for Turbo Debugger.**

**12/23/2022:**

* **The Decompiler project was initiated in order to decompile MS-DOS EXE and COM binaries. The project has both a command-line and a GUI tool:**
* [**https://sourceforge.net/projects/decompiler/**](https://sourceforge.net/projects/decompiler/)
* **It looks like the program that I am tracing through is USERPROG procedure number 6 near IPC 1164,**
* **i need to figure out how to display the IPC and the segment name.**

**12/24:2022:**

* **The mystery procedure number 6 appears to be called from OS\_INIT (and only from OS\_INIT). Could it be INIT\_SYSCOM or INIT\_ENVIRONMENT?**
* **PSYSTEM.EXE SYSTEM4.VOL also openS UTILITY.VOL using PSYSTEM.CFG**

**12/27/2022:**

**SYSCOM: Located at $0038**

| **Addr (Hex)** | **Contents** |
| --- | --- |
| **0008** | **CS** |
| **000A** | **DS** |
| **000C** | **SI (Saved IPC)** |
| **0012** | **ES (Saved ES)** |
| **0014** | **current MSCW** |
| **00B8** | **CurProc (usually?)** |
| **0038** | **IORESULT / SYSCOM** |
| **003A** | **(was APOOLSIZE)**  **(I’m going to use for DbgCnt)** |
| **003C** | **SYSUNIT** |
| **004C** | **GDIRP** |
| **0082** | **CRTINFO** |
| **2212** | **IOR** |
| **00B8** | **CURPROC** |
| **0265** | **GetBigB→ CX (Len)** |
| **CSP Table start at 1688** |  |
| **1348** | **PoolBase** |
| **BX div 2** | **Opcode in FETCH** |

* **PME-Debug addresses are $86 bytes**
* **The jump table gets loaded along with the interpreter (located at CS:1488)**
* **The jump to IOR may be located $16C4 (which contains $2212)**
* **The IOR seems to be looking for IORESULT at [0038]**
* **iS HEIGHT in the CRTINFO? Seems to be. I don’t know why it dosen’t seem to be present in the MISCINFO that got loaded.**
* **What was the name of that program that would display a page full of photos/documents and would let you convert to scanned text? (So I can convert the jump tables to a list of addresses and/or opcode names?) PAPERPORT**
* **Maybe I can get TD to save the jump table window to a log file.**

**12/29/2022:**

* **The SCXG is at 0D03**
* **Code Sequence:**

| **PROC NUM** | **OPCODE** |  |
| --- | --- | --- |
| **0** | **CPL 4** |  |
| **4** | **SLDC 1** |  |
| **4** | **SCXG** |  |
| **4** | **LAO** |  |

* **Procedure sequence on VPC interpreter:**
  + **0, 4, 2, 6, 6, …, 9, 7**
* **Delphi**
  + **MOVE(Bytes[SrcO], Bytes[Dst+DstO], Len)**
    - **Proc = 26: InitSysCom, SrcO = 55698, Dst = 0, DstO = 262**
  + **Proc = 2: USERPROG:BOOTERROR; Src = 59296, SrcO = 2, Dst = 0, DstO = 59170 (Why No history? Using invalid value for OpCode) DbgCnt = 25**
  + **The 1st instruction encountered is a SLDC0 but it should be CPL 4? Cannot reproduce. I see 1st instruction is SLDC 1**
  + **SCIP1 3-- procedure = 2**
  + **I changed the MOV command to use SRC (previously omitted) -- now getting a system HALT in CALLIO.**
  + **I need to set ExternalPool setting in SysCom based on BootParams. Maybe this is supposed to be done in SETUP?**
  + **PoolOutSide gets changed when SYSTEM.MISCINFO gets loaded**

**12/30/2022:**

* **SYSTEM4.VOL MiscInfo shows POOLOUTSIDE = TRUE**
* **PSYSTEM.VOL Miscinfo ALSO shows POOLOUTSIDE = TRUE**
* **Is POOLOUTSIDE the same as EXTENDED memory?**
* **Can I boot both SYSTEM4.VOL and PSYSTEM.VOL and determine which version SYSTEM.MISCINFO indicates is being run?**
* **The only version numbers found in pmachver are:**
  + **pre\_iv\_1**
  + **iv\_2**
  + **random**
* **Neither SYSTEM4.VOL or PSYSTEM.VOL contains valid information in PMACHVER**
* **In the VPC version of PSYS, I was able to boot with** 
  + **psystem.com volumes\psystem.vol (but then crashed because it couldn’t load \*LOADFILES.TEXT )**
  + **Has Extended memory: Yes**
  + **Code Pool Size: 64**
  + **First subsidiary volume: 20**
  + **Max number of subsidiary Vol = 40**
  + **Max number of serial volumes = 4**

**12/31/2022:**

* **F:\NDAS-I\d7\Projects\pSystem\Temp\DS3N20.DSK**
  + **BPS: 512, SPT: 16, Int: 1, Trk0: 2, Skew: 0, NrTracks: 127**
* **PSYSTEMX (before booting): PoolSize: 72; PoolBaseAddr: $fb580019; Resolution: 17024; PoolOutside: true. This does NOT agree with the SysCom page in the debugger -- BUT it is looking at the FILE SYSCOM.MISCINFO-- not at memory which gets set during the boot process.**