BEYOND "REPACK"

BY MICHAEL WHITE

FIRST OFF, THERE ARE A FEW THINGS NEEDED TO BE CLEARIFIED. WHILE SENDING OUT A TAPE RECENTLY, I REMEMBERED ANOTHER POINT FROM MY ORIGINAL FILE SEARCH ARTICLE, OMMITTED AND LATER FORGOTTEN. AT 300 BAUD ONLY, WHEN YOU SAVE A PROGRAM USING [:PRINT ;LIST], AND THEN LOAD IT IN OVER ANOTHER PROGRAM, LIKE FILE SEARCHER, THE INCOMMING PROGRAM LOADS AS IF YOU'RE KEYING IT IN "LINE BY LINE". THIS MEANS THAT IF YOUR LINE NUMBERS DON'T ALL MATCH, YOU WON'T WRITE OVER ALL OF THE EXHISTING LINES, AND YOUR SLAVE PROGRAM WILL EITHER CRASH UPON RUNNING INTO A BOGUS LINE (LEFT OVER), OR RUN OUT OF MEMORY IF IT USED ALL 1800 BYTES. YOU CAN RUN A STRING OF NUMBERS OFF OF THE TAPE THAT SIMULATES LINE NUMBER ENTRIES, BUT THAT TAKES MORE TIME AND MORE TAPE! THE ONLY SURE-FIRE CURE IS TO USE THE DAVE IBACH TAPING METHOD (ARCADIAN VOL.3 PG.24+25), BUT THAT EXPOSES YOU TO THE "INTERFACE OPENING PROBLEM" THAT I DISCUSSED IN **"FILE SEARCH"** (SEE BALLYALLEY.COM).

SECONDLY, IN MY LAST TUTORIAL, I SAID "---THE TEXT (WHICH OCCUPIES THE ODD BITS) AND THE PICTURE (WHICH RESIDES IN THE EVEN ONES)", AND YET JOHN PERKINS (ARCADIAN VOL.1 PG.40) STATES THE OPPOSITE! JOHN WAS USING THE SCIENTIFIC ANNOTATION, I USE THE MATHAMATICAL. THE DISCREPENCY LIES IN THE DATA LINE "D0". THE PICTURE USES DATA LINES "D1+D3+D5+D7" (ODD), WHICH ARE THE 2ND+4TH+6TH+8TH DIGITS (EVEN). THE TEXT USES DATA LINES "D0+D2+D4+D6" (EVEN), WHICH ARE THE 1ST+3RD+5TH+7TH DIGITS (ODD). CLEAR AS MUD, RIGHT? FOR PROGRAMMING PURPOSES, THE TEXT BITS ARE IN THE BITS OCCUPIED BY HEX 55, AND THE PICTURE IS IN THE BITS OCCUPIED BY HEX AA. IT'S EASIER FOR ME TO SEE IT LIKE I WAS TAUGHT IN MATH CLASS, HOW ABOUT YOU?

THE OTHER THING IS SOMEWHAT MY FAULT AS I NEVER UNDERSTOOD THE VIPER SYSTEM COMPLETELY, NEVER HAVING OWNED ONE! VIPERSOFT BASIC GIVES AN SZ OF 32K IN A 16K BLUE RAM, BUT NOT IN A VIPER 1. SO THE !A000 AREA CAN'T BE USED. THE SOLUTION? (ARCADIAN VOL.7 PG.73+74)! THE PICTURE ONLY NEED BE IN MEMORY A SHORT TIME, AND IF THE TAPE LOADING DOESN'T FAULT, NO MESSAGE NEED APPEAR BEFORE THE MACHINE CODE "REPACK" ROUTINE RUNS. THEREFORE, CHANGE LINE 40 TO:

```
>40LC=0;GET%(16384);A=-32584;IF LCGOSUB 190;GOTO 40
```

ALSO, YOU'LL HAVE TO CHANGE THE MACHINE CODE TO LOAD "DE" WITH !4000 RATHER THAN !A000. IF YOU BLUE RAM OWNERS WANT TO TRY THIS AS OPPOSED TO USING !8000, **GO FOR IT!** THIS ALLOWS YOU TO SEE THE PICTURE LOADING, AND THEN BEING REPACKED! (DON'T BLINK!)

WHAT THE MACHINE CODE PART DOES IS SET THE REGISTERS "DE" WITH THE 1ST ADDRESS OF OUR PICTURE (!4000?), "BC" WITH 1808 (BYTES TO "REPACK"), AND "HL" WITH THE 1ST ADDRESS OF THE TEXT (!9000). NOTICE THAT "EXX" WAS CHOSEN OVER "PUSH DE" TO SAVE OUR BASIC TEXT PROGRAM POINTER (MORE ON THIS LATER). THE PART BEGINNING WITH THE LABEL "LOOP" RUNS 1808 TIMES! FIRST IT GETS THE 1ST BYTE OF TEXT AND USES LOGIC "AND" TO CLEAR OUT THE PICTURE BITS! THE RULE OF "AND" (OR INTERSECTION) IS: 0+0=0, 0+1=0, 1+0=0, 1+1=1. THEREFORE, WHEN WE "AND !55" WITH OUR TEXT, THE PICTURE CLEARS! SEE THE FOLLOWING EXAMPLES:

	#1	#2	#3	#4	#5	#6
BIT NUMBERS >	76543210	76543210	76543210	76543210	76543210	76543210
EXAMPLES >	00000000	01010101	10101010	11111111	11001100	00110011
"AND 55 HEX" >	01010101	01010101	01010101	01010101	01010101	01010101
TEXT RESULTS>	00000000	01010101	00000000	01010101	01000100	00010001

AS YOU CAN SEE, ONLY THE BITS THAT ARE ANY PART (OR ALL) OF THE 55 HEX PASS INTO THE "TEXT RESULTS". AFTER PUTTING THE CLEARED TEXT BYTE BACK TO !9000, THE FIRST BYTE OF PICTURE IS FETCHED. AGAIN WE USE "AND", ONLY THIS TIME WITH HEX AA. HERE ARE THOSE EXAMPLES:

```
                    #1       #2       #3       #4       #5       #6
  BIT  NUMBERS > 76543210 76543210 76543210 76543210 76543210 76543210
    EXAMPLES   > 00000000 01010101 10101010 11111111 00110011 11001100
  "AND AA HEX" > 10101010 10101010 10101010 10101010 10101010 10101010
PICTURE RESULTS> 00000000 00000000 10101010 10101010 00100010 10001000
```

DOING IT THIS WAY, ONLY THE PICTURE BITS END UP AS "PICTURE RESULTS". THEN WE USE LOGIC "OR" TO CREATE OUR COMPOSITE. THE RULE FOR "OR" (OR UNION) IS: 0+0=0, 0+1=1, 1+0=1, 1+1=1. HOWEVER, SINCE WE CLEARED OUT BOTH THE TEXT AND THE PICTURE, THERE WILL BE NO 1+1 HITS **ANYWHERE!!** HERE IS THE COMPOSITE CHART OF OUR PREVIOUS TWO RESULTS:

```
                    #1       #2       #3       #4       #5       #6
  BIT  NUMBERS > 76543210 76543210 76543210 76543210 76543210 76543210
  TEXT  RESULTS> 00000000 01010101 00000000 01010101 01000100 00010001
PICTURE RESULTS> 00000000 00000000 10101010 10101010 00100010 10001000
   COMPOSITES  > 00000000 01010101 10101010 11111111 01100110 10011001
```

NOTICE THAT I SWAPPED EXAMPLE #5 WITH #6 FOR THE PICTURE. THIS IS TO SIMULATE THE EXCHANGING OF "TITLE SCREENS" AS "REPACK" NORMALLY DOES. THE COMPOSITE BYTE IS THEN STORED AT !9000. THE NEXT INSTRUCTION IS A "SEMI-AUTOMATIC" (TWO BYTE) INSTRUCTION THAT BASICALLY ACCOMPLISHES THE FOLLOWING:

```
LD (DE),(HL)
INC DE
INC HL
DEC BC
SET FLAG (PE (1) IF BC#0, PO (0) IF BC=0)
```

THERE ARE OTHER "SEMI-AUTOMATIC" INSTRUCTIONS IN THE Z80 INSTRUCTION SET (ALL DO MULTIPLE FUNCTIONS), BUT THIS ONE WAS CHOSEN BECAUSE IT ACCOMPLISHES ALL OUR NEEDED TASKS. ONLY THE FIRST PART IS AN EXTRA, AND THAT'S THE PART THIS INSTRUCTION WAS MADE FOR! IF YOU HAVE THE PICTURE LOADED TO !4000, YOU WILL SEE THE COMPOSITE APPEAR BECAUSE OF THE FIRST PART OF THIS INSTRUCTION. AT THIS POINT ALL OF OUR REGISTERS HAVE BEEN ADVANCED AND WE'RE READY TO TEST THE FLAG AND LOOP BACK, (SORT OF LIKE ENCOUNTERING THE "NEXT " IN A "FOR TO NEXT " LOOP IN BASIC). HOWEVER, THERE IS NO "JUMP RELATIVE" ON CONDITION OF THE PARITY FLAG IN THE Z80 INSTRUCTION SET! THERE IS, THOUGH, A "RETURN " ON THIS CONDITION! BUT, BEFORE WE RETURN TO BASIC, THE BASIC TEXT PROGRAM POINTER NEEDS TO BE RESTORED, AND DOING THIS FROM THE STACK (USING "POP DE") CLOBBERS OUR CURRENT "DE" (WORKING PICTURE ADDRESS) THAT WE NEED TO LOOP BACK WITH! ANSWER?--"EXX"! WHEN WE USE "EXX", WE (SORT OF) "EXCHANGE" THE CONTENTS OF "BC", "DE", AND "HL" WITH THIER ALTERNATE COUNTERPARTS (SOMETIMES CALLED "SACRED REGISTERS"), BY "BANK SWITCHING" BETWEEN TWO REGISTER SETS! SO WHEN WE BRING UP OUR BASIC TEXT PROGRAM POINTER, THE REGISTERS THAT HELD OUR ADDRESSES AND ETC. (FOR REPACKING) ARE PRESERVED, AND CAN BE REINSTATED WITH ANOTHER "EXX" 1808 TIMES! (WITHOUT LOSING OUR BASIC TEXT PROGRAM POINTER EITHER!) AFTER ALL THAT, WE "JUMP RELATIVE" BACK TO WHERE THE "LOOP" LABEL WAS, AND REPEAT UNTIL DONE. ANY QUESTIONS? WRITE AND ASK! (SPECIAL THANKS TO MARION NALEPA OF SPECTRE SYSTEMS FOR THIS ROUTINE)

"BANK SWITCHING" (MENTIONED ABOVE) IS SIMPLY A WAY FOR A COMPUTER (ANY COMPUTER) TO ACCESS MORE MEMORY THAN IT HAS ADDRESS LINES FOR. IN THE CASE OF THE PROPOSED (BUT NEVER BUILT) "ADD ON" FOR THE ARCADE, IT IS POSSIBLE TO BUILD A MEMORY EXPANSION THAT HAS 256K, OR MAYBE EVEN MORE, **BUT NOT IN 1984 FOR LESS THAN $1000.00!!** THE WAY IT WOULD HAVE WORKED WAS THAT EACH "BANK" (GROUP OF RAM OR ROM CHIPS THAT MAKE UP 32K OR 64K OF MEMORY) WOULD BE TIED TO THE SAME ADDRESS LINES. MEANING THAT A READ OR A WRITE TO ONE OF THOSE ADDRESSES, **ACCESSES THAT ADDRESS IN EVERY "BANK"!!!** HOWEVER, A "DECODER" WOULD BE INSTALLED, WHICH ALLOWS ONLY ONE "BANK" TO RESPOND! THE COMPUTER SEE'S ONLY THIS ONE "BANK", BUT CAN SWITCH "BANKS" BY SENDING A DIFFERENT VALUE TO THE DECODER, WHICH IS CONTROLLED AS AN OUTPUT PORT (USUALLY A HIGH

NUMBER, SAY 200). TO "BANK SWITCH" SIMPLY SAY "%(200)=X" AND ANOTHER "BANK" RESPONDS! THE COMPUTER WOULD ONLY BE ABLE TO ACCESS THE ENTIRE MEMORY ONE "BANK" AT A TIME, LIKE VIEWING PAGES IN A BOOK, BUT IT CAN "BANK SWITCH" TO ANY CHAPTER THE BOOK HAS.

BELIEVE IT OR NOT, WE CAN ALSO DO THIS IN BLUE RAM BASIC 1.1! %(!6FFC) AND %(!6FFE) ARE THE BEGINNING AND ENDING POINTERS OF OUR TEXT AREA. TO SWITCH TO A DIFFERENT "BANK" OF MEMORY, SIMPLY POKE NEW VALUES INTO THESE ADDRESSES AND THE SIZE OR LOCATION OF THE BASIC TEXT AREA CHANGES! ALSO, AS STATED (ARCADIAN VOL.4 PG.103) CONCERNING "GOTO " AND "GOSUB ", "MACHINE GOES TO THE BEGINNING AND SEARCHES FOR THE NEW LINE NUMBER---SHOULD BE NEAR THE BEGINNING TO SAVE TIME", BUT, WITH A POKE AT %(!6FFC) THE "BEGINNING" CHANGES AND BRB PROGRAMS CAN RUN EVEN FASTER! BY USING "GOSUB X,%(!6FFC),Y" FOR EVERY SUBROUTINE CALL, WE GET TERRIFIC SPEED (FOR BASIC). ALAS, THERE IS NO PROGRAM, NOT EVEN **"QUADRA"**, THAT TAKES IT TO THIS LIMIT! HOWEVER, BASIC CAN NOW RUN IN BOTH THE UPPER AND LOWER MEMORY "BANKS" IN THE SAME PROGRAM!! THIS **"QUADRA"** DOES DO BY CHANGING BOTH VALUES AND THE "TXTUNF" AS WELL (16K VERSION ONLY). ONLY "GOSUB ", "GOTO ", OR "RUN " REFERS TO THE "BEGINNING" VALUE (%(!6FFC) IN BRB). THEREFORE, ONLY A "GOTO " NEED FOLLOW, AND OUR "BANK SWITCH" IS COMPLETE! NOW, IF YOU'RE WONDERING IF "GOSUB " WERE USED, WOULD "RETURN " BRING YOU BACK? TRY IT!!

ALSO, LAST TIME (IN THE "REPACK" TUTORIAL) I ASKED YOU TO TRY TO USE THE MACHINE CODE "REPACK" ROUTINE IN AN AB PROGRAM. DID YOU GET SOMETHING LIKE:

```
>10 CLEAR ;A=24576;CALL20258;PRINT "  AWAITING INPUT ";PRINT "   0F TEXT;
    :INPUT %(A)
>20 PRINT "  TEXT L0ADED!";B=1864;FOR C=28346TO 28516STEP 2;IF %(C)B=(C-A)÷2
>30 NEXT C
>40 CLEAR ;PRINT "  READY";PRINT "  T0 L0AD PICTURE";:INPUT %(28672)
>50 PRINT "   PICTURE L0ADED";FOR D=0TO 999;NEXT D
>60 CALL20268;CLEAR ;PRINT "  D0 Y0U WANT";PRINT "   FILE SEARCH?";GOSUB 130;
    D=20018
>70 C=KP-50;IF (C<-1)+(C>0)GOTO 70
>80 CLEAR ;IF CFOR E=72TO 89;TV=32;TV=E;INPUT "="%(D);D=D+2;NEXT E;PRINT
    "  IS THIS 0K?";GOSUB 130;IF KP-49GOTO 80
>90 CLEAR ;PRINT "  START TAPE REC0RDING";PRINT "  AND HIT ANY KEY
>100 IF KPIF C:PRINT H,18
>110 :PRINT %(A),B;PRINT "  PLAY TAPE BACK IN";PRINT "  T0 VERIFY";:LIST ;
    IF C:LIST
>120 PRINT ;PRINT "  T0 RERUN UTILITY";PRINT "   PRESS [G0]";IF KPRUN
>130 PRINT "   1=YES  2=N0";RETURN

ENTER> FOR A=20258TO 20293STEP 2;PRINT "%(",#5,A,;INPUT ")="%(A);NEXT A
```

THEN KEY IN THESE NUMBERS BY READING EACH COLUMN DOWN (NOT ACROSS) WHEN ENTERING:

```
-43      32     3608    24576    -6330    -9824
 27  -13871       17    -6530   -18774    -9760
 96     473     8560    30549    -4745    -4072
```

AND FINALLY, SAVE WITH SOMETHING LIKE:

```
ENTER> PRINT ";RUN ";CLEAR ;CY=4;CX=-15;PRINT "REPACK";CX=-9;PRINT "FAST";
    :PRINT %(16384),2000 [REC]
```

THE ONLY ADVANTAGE TO THIS LISTING OVER THE ONE I INCLUDED LAST TIME (BESIDES LESSON VALUE) IS SPEED! THEY BOTH DO THE SAME THING!

LAST TIME ALSO, THERE WERE A FEW OTHER THINGS I SAID WE WOULD LOOK INTO. ONE WAS THE "CC" VARIABLE IN EB. THIS CONTROLS THE "SCREEN WRITE" THAT WE "PRINT " OR "TV=" IN. IT GOES:

COLOR	XOR	OVERLAY
BC	CC=0	CC=4
FA	CC=1	CC=5
FB	CC=2	CC=6
FC	CC=3	CC=7

NOTICE (FROM THIS CHART), THAT "CC=0" WOULD CAUSE INVISIBLE (USELESS) PRINTING. HOWEVER, A LOOK AT **"SNOOP CAMERA"** (ARCADIAN VOL.7 PG.64) REVEALS "DATA NT,3,-76,48,0,0,13094" IN LINE 93. THE SEQUENCE IS: NT,CX,CY,CC,LC,CF. THIS SETS "CC" TO 0, SO THAT WHEN WE RETURN FROM A CARTRIDGE WITH A SCREEN WE WISH TO KEEP, IT WON'T PRINT "BLUE RAM BASIC 1.1" ON THE SCREEN AND RUIN OUR PICTURE! ALL WE NEED DO THEN IS KEY IN "RUN [GO]", THE UTILITY RESTARTS, "SNAPS" THE PICTURE, AND GOES ON FROM THERE.

ANOTHER "TRICK" I HAD SLATED FOR MY SERIES "TRICKS OF THE TRADE", BUT NEVER GOT PRINTED, WAS THE LINE INPUT BUFFER RELOADING, AS SEEN ON SOME WAVEMAKERS TAPES, SUCH AS "DUNGEONS OF DRACULA". THE PROGRAM BEGINS SO:

```
>1.M.PEACE
>0:INPUT %(R);%(A)=B
```

LINE #0 WILL NOT LIST ON THE SCREEN WITHOUT EITHER A LINE NUMBER CHANGE, (USE "%(-24565)=5)", OR A "FOR TO NEXT " LOOP, (LIKE: FOR C=-24576TO -24500; TV=%(C);NEXT C). THE VARIABLES WERE SET "R=20160;A=-24563;B=46", AND THEN THE PROGRAM WAS TAPED WITH:

```
PRINT ";RUN ";:PRINT %(16384),1888 [GO]
```

THE FOLLOWING "BURSTS" WERE TAPED FROM A SEPARATE PROGRAM, THUS:

```
>10 ;CLEAR ;CY=9;PRINT " D0N'T ST0P THE REC0RDER   THANKS F0R Y0UR SUPP0RT";
    PRINT ;CX=0;PRINT "MIKE PEACE";BC=&(28)x8-1;&(9)=-1;RUN
>20 :PRINT %(-24574),66;A=KP;GOTO 20
```

"GOTO 20" SENT THIS TO TAPE. THEN LINE 10 WAS CHANGED TO:

```
>10 ;CLEAR ;CY=0;CX=-44;PRINT "\/\/AVE/\/\AKERS";E=-2;FOR D=-68TO 68STEP 5;
    E=E+D÷20;BOX D,E,20,2,1;BOX -D,-E,20,2,1;NEXT D;R=20200;RUN
```

"GOTO 20" TAPED THIS AS THE SECOND "BURST" AND THE MACHINE CODE (FOR THE PROGRAM), BECAME THE THIRD "BURST". THE "R" VALUE WAS SET TO THE ADDRESS OF THE 7TH BYTE OF THE LINE INPUT BUFFER, SO THAT WHEN THE NEXT "BURST" LOADS, THE LINE INPUT BUFFER RELOADS WITH THE NEW DATA AND EXECUTES ACCORDING TO THE SYSTEM SHOWN IN MY TUTORIAL "FILE SEARCH II". "R" GETS CHANGED FOR THE LAST "BURST", AND BECAUSE THE LINE INPUT BUFFER DOESN'T GET AN "OVERWRITE", LINE #0 FINALLY FINISHES BY POKING A 46 (PERIOD) INTO ITSELF CAUSING IT TO BECOME A "REM STATEMENT" AND OVERLOOKED (FROM THEN ON) BY THE COMPUTER. NICE HAVING OUR PROGRAM IN RAM ISN'T IT? NOW, CAN YOU IMPROVE ON THIS? THINK ABOUT IT!

ONE THING THAT EVERYBODY IS DYING TO KNOW, IS EXACTLY HOW TO PUT A BASIC PROGRAM INTO A ROM AND RUN IT! WELL, BASICARTS [AS THEY WERE CALLED BY KEN LILL (ARCADIAN VOL.7 PG.59)] DO NOT RUN FROM A ROM AT ALL! IN FACT, THEY RUN NO DIFFERENT THAN TAPED PROGRAMS! LEROY'S IDEA WAS NOT TO RUN BASIC FROM ROM (ALTHOUGH A REQUEST WAS MADE FOR THAT), BUT TO SIMPLY ELIMINATE THE TAPE RECORDER! THE PROGRAM ARRIVES IN SCREEN RAM **(JUST AS IF IT WAS TAPED)** VIA THE "MOVE" SUBROUTINE #94 (+1)! BY TAPING YOUR PROGRAM WITH A "TITLE SCREEN" AND THE "AUTO RUN" TECHNIQUE AS I'VE DISCUSSED (ARCADIAN VOL.7 PG.22+23, OR NIAGARA BUG BULLETIN VOL.2 PG.37+38), YOU'VE SAVED EVERYTHING NEEDED TO MAKE THE PROGRAM RUN **REGARDLESS** OF HOW IT ARRIVES IN MEMORY! ALL THAT REMAINS IS TO BURN AN 8Kx8 EPROM (LIKE A 2764) WITH A COPY OF ASTRO BASIC IN THE 1ST 4K AND YOUR BASIC PROGRAM IN THE 2ND 4K SO THAT THE ARCADE SEES ASTRO BASIC IN THE 2000 THROUGH 2FFF HEX ADDRESSES AND THE BASIC PROGRAM IN THE REST, WITH A SHORT MACHINE CODE ROUTINE ADDED AT 3FC9 HEX! NOTE: THIS (WOULD BE) ADDRESS IS USED FOR THIS MACHINE CODE BECAUSE THE STACK CANNOT BE SAVED AND RELOADED BY ANY MEANS! THE MACHINE CODE ROUTINE IS:

```
START)31 .LD.SP,4FEA
      EA [SET STACK POINTER]
      4F  [FOR AB]
      FF .RST.38
      00 [INTERPRET CALLS]
      5F [MOVE +1]
      00 [DESTINATION ADDRESS]
      40  [TO DE]
      C8 [BYTES TO MOVE]
      0F  [TO BC]
      00 [SOURCE ADDRESS]
      30  [TO HL]
      17 [SET OUT +1]
      B0 [TO &(10)]
      2C [TO &(9)]
      18 [TO &(14)]
      19 [COL SET +1]
      F8 [COLOR TABLE ADDRESS]
      3F  [TO HL]
      02 [END CALLS]
      3E .LD.A,20
      20 [INTERRUPT VECTOR H]
      ED .LD.I,A
      47 [SET I]
      3E .LD.A,22
      22 [INTERRUPT VECTOR L]
      D3 .OUT.13,A
      0D [TO &(13)]
      3E .LD.A,C8
      C8 [INTERRUPT TRIGGER]
      D3 .OUT.15,A
      0F [TO &(15)]
      FB .EI.
      FF .RST.38
      51 [PAWS +1]
      7F [2 SECONDS]
      CD CALL,2D06
      06 [CALL SUBROUTINE]
      2D  [IN AB]
      11 .LD.DE,4EBD
      BD [SET DE]
      4E  [FOR AB]
      D5 PUSH DE
      1B DEC DE
      C3 .JP.2563
      63 [WARM START ADDRESS]
      25  [IN AB]
COLOR TABLE AT 3FF8
      00 [TO &(7)]
      00 [TO &(6)]
      07 [TO &(5)]
      07 [TO &(4)]
      5A [TO &(3)]
      84 [TO &(2)]
      06 [TO &(1)]
      00 [TO &(0)]
```

YOU'LL HAVE TO CHANGE THE 2ND AND 3RD BYTES OF ASTRO BASIC (!2001 FROM !F7 TO !C9 AND !2002 FROM !24 TO !3F). THIS WILL CAUSE A COLD START JUMP TO THIS MACHINE CODE ROUTINE FROM A [RESET]!

THIS ROUTINE USES TWO "ON BOARD" SUBROUTINES THAT HAVEN'T BEEN DISCUSSED YET, SET OUT AND COL SET. THESE TWO SET &(9) AND ITS "COUSINS". FOR THE BEST DISCRIPTION OF THE COLOR PORTS (ARCADIAN VOL.1 PG.40) IS STILL IT! COL SET (COLOR SET) SETS PORTS 0 THROUGH 7 BY SENDING IT THROUGH THE "COLOR BLOCK TRANSFER PORT" &(11). THE CALL IS 24DEC (+1) FOLLOWED BY AN ADDRESS (LOADING TO THE HL REGISTER PAIR) THAT POINTS TO AN 8 BYTE COLOR TABLE. THAT APPEARS REVERSED [THE 1ST BYTE ENDS UP IN &(7) AND THE LAST IN &(0)]! SET OUT [22 DEC (+1)] SETS 3 PORTS, &(10), &(9), AND &(14). &(10) AND &(9) DO THE SAME AS IN BASIC. THE BORDER SHOWS THE &(4) COLOR FOR &(9)=0-63, &(5) FOR &(9)=64-127, &(6) FOR &(9)=128-191, AND &(7) FOR &(9)=192-255. &(0), &(1), &(2), AND &(3) NEVER CONTROL BORDER COLOR! NOTE: IN AB (OR BB) &(4) AND &(5) ARE SET TO "BC" AND &(6) AND &(7) ARE SET TO "FC" EVERY SCREEN INTERRUPT! IN EB &(4) IS SET TO "BC", &(5) TO "FA", &(6) TO "FB", AND &(7) TO "FC" EVERY SCREEN INTERRUPT AS WELL!! &(14) SETS THE INTERRUPT MODE IN THE ADDRESS CHIP. THAT'S 18 HEX IN AB, SOMETHING EVEN THE NUTTING MANUEL DOESN'T CLEARIFY!

THE MACHINE CODE BEGINS BY SETTING THE STACK POINTER (WHERE AB SETS IT), THEN ENTERS AN INTERPRETED STRING (ARCADIAN VOL.5 PG.132+133) WHERE IT CALLS "MOVE", "SET OUT", "COL SET", AND THEN EXITS VIA THE END SUBROUTINE #02! THE NEXT 13 BYTES INITIALIZE THE SCREEN INTERRUPT (MORE ON THIS NEXT TIME). AFTER THAT IT CALLS "PAWS" (ARCADIAN VOL.5 PG.100) WHICH IS A TIME DELAY TO DISPLAY YOUR "TITLE SCREEN". NEXT IS A CALL TO A ROUTINE IN AB THAT I'M NOT SURE WHAT IT DOES, BUT IT'S INCLUDED TO BE SAFE. THEN "DE" LOADS WITH 20157 AND GETS "PUSHED" TO THE STACK. WITHOUT THIS, THE TAPE INTERFACE WON'T WORK (IF YOU WISHED TO SWAP TO AN ORIGIONAL ASTRO BASIC AND USE IT). THEN "DE" DECREMENTS TO 20156 WHICH **IS** THE ADDRESS FOR "AUTO RUN" (SEE FILE SEARCH II), AND INTO AB'S "WARM START" ADDRESS (2563 HEX) IT GOES! ANY QUESTIONS? WRITE AND ASK!

VIPERSOFT BASIC PROGRAMS CANNOT "AUTO RUN" FROM TAPE, BUT **DO** AS BASICARTS! IF ANYONE WANTS TO SEE THIS, CONTACT ME! BRB IS LIKE AB EXCEPT THE ADDRESSES CHANGE, 2 SPACES ARE ADDED TO THE LINE INPUT BUFFER, "CALL2164" STARTS THE SCREEN INTERRUPT, AND A "SWAP LOOP" LETS YOU CHANGE THE CARTRIDGE BECAUSE EB IS 8K, NOT 4K! THUS, TWO 4K PROGRAMS CAN BE PUT INTO AN 8Kx8 EPROM AND ACCESSED VIA THE "ON BOARD MENU". (SEE THE MULTICART) WHERE AB AS WELL AS BRB PROGRAMS GOT "DOUBLED UP". THE "MENU" IS A RST 38 SUBROUTINE THAT CAN BE BROUGHT UP BY "CALL3177" FROM ANY BASIC, **ON ANY ARCADE!** NEXT TIME WE'LL LOOK AT THE PROPER WAY, PLUS BRB BASICARTS, INTERRUPTS, AND MULTIPLE LOADING TECNIQUES! KEEP BUGGIN'!!