```
************************************************************
*                                                          *
*              CalendarMY - An AppleWorks Init             *
*              ********************************            *
*          - Adds a multi-year perpetual calendar          *
*             display to the <OA-Q> DeskTop Index          *
*             accessible from any screen in                *
*                     AppleWorks 5.1 -                     *
*                                                          *
*          Version 5.2 (for AppleWorks Version 5.1)        *
*        (c) 2024 by Hugh Hood (perpetual/multi-year)      *
*                                                          *
************************************************************
*                                                          *
*                    PRIOR VERSIONS                        *
*                    --------------                        *
*    (a)     Single year version 1.0 (c) 1996 by           *
*        Christian Serreau / The AppleWorks Gazette        *
*                                                          *
*    (b)     Unfinished and unimplemented version          *
*             included in AppleWorks 5.1 (c) 1995          *
*             as a blanked screen reading simply:          *
*                                                          *
*       AppleWorks Monthly Calendar by Randy Brandt        *
*              Will it ever do anything?                   *
*                                                          *
************************************************************
*                                                          *
*        - After entering the Desktop Index by             *
*              pressing <OA-Q>, press <OA-M>               *
*                to display the calendar -                 *
*                                                          *
*    - Press <Left Arrow>/<Right Arrow> to change Month    *
*                                                          *
*    - Press <Up Arrow>/<Down Arrow> to change Year        *
*                                                          *
*    - Press <Return> to enter a Year from 1582 to 9999    *
*                  [Gregorian Calendar]                    *
*                                                          *
*      - Calendar may then be printed, copied or           *
*         imaged to the clipboard by accessing             *
*         the standard <OA-H> dialog -                     *
*      - <OA-H> is modified during the routine             *
*          to omit the screen headers and footers          *
*          and only print or copy the Calendar and         *
*          to insert 0.00" left and right margins          *
*                                                          *
*    - Seg $27 is patched to allow <OA-Q><OA-M> to call    *
*       Seg $2D (Desktop Calendar), and Seg $2D is         *
*       patched to generate and display the perpetual/     *
*       multi-year calendar.                               *
*                                                          *
*    - NOTE: Since the length of the new code for $5200+   *
*             exceeds the $400 byte maximum for the        *
*             SEG Patch Manager memory ($BB00-$BEFF),      *
*             the code is first stored in Desktop          *
*             memory when the init is run and its          *
*             pointer is saved within the patch code       *
```

```
*             so that it may be retrieved and run at       *
*             the proper address ($5200).                  *
*                                                          *
*                                                          *
************************************************************

               TR             ADR            ; truncate bank address

               XC                            ; enable 65C02 code


* Internal AppleWorks Equates
ErrFlag        EQU            $03            ; ProDOS error occur?
FoundCmd       EQU            $86            ; non-zero if command found
FoundEsc       EQU            $88            ; non-zero if <ESC> pressed
MReg           EQU            $91            ; MultByte/MultWord/DivWord result here
MWrkX          EQU            $93            ;

AArg           EQU            $9A            ; $9A/$9B - contain address of Year in ASCII
BArg           EQU            $9C            ; $9C/$9D - used for conversion calculation
CArg           EQU            $9E            ; $9E/$9F - will contain FullYear in HEX

DVEndMSB       EQU            $F2            ; $F2/$F3 - DivWord remainder here
HardCopyTop    EQU            $AAB           ; top line for <OA-H> (default=0)
HardCopyBtm    EQU            $AAC           ; bottom line for <OA-H> (default=$18/24)
DTCurOpen      EQU            $C54           ; current file # (if any)
DTCtOnDesk     EQU            $C55           ; number of files on current Desktop
SaveScreen1    EQU            $CBB           ; pointer to SaveScr screen 1
SaveScreen2    EQU            $CBD           ; pointer to SaveScr screen 2
NewStr         EQU            $D85           ; 128 bytes (used by many routines)
OrgInMem       EQU            $E86           ; $00 = in file / $01 = in menu /
                                             ;   $40 = in file list
StrWork5       EQU            $FD7           ; 5-byte pString used by newWord2Str
MiscSet2       EQU            $FFE           ; Bit 3 is Mousetext Screens

* AppleWorks Host Equates
AWVersion      EQU            $1003          ; $33/51 = 5.1 / $28/40 = 4.0  /
                                             ;   $1E/30 = 3.0
StrWork3       EQU            $1018          ; 3-byte pString used by newConv1TP
MainInMem      EQU            $101C          ; $01/DB; $15/WP; $1D/SS
PrevSeg        EQU            $1021          ; SEG in use (last loaded by CallSeg)
TOActive       EQU            $10E6          ; $00 = enabled / $01 = disabled <OA-ESC>
CursorOnSw     EQU            $10F2          ; $00 = cursor off / $01 = cursor on
CallSeg        EQU            $10FA          ; AW segment loading routine
                                             ;  - LDA with segment # before calling
                                             ;     ($01/01 - $2F/47)
                                             ;  - see segment chart
                                             ;  - add $80/128 to load only (not JMP)
ClearDA        EQU            $1100          ; clears display area on screen
DivWord        EQU            $1112          ; routine to divide (2) words
                                             ;  followed by:
                                             ;   (i) address of dividend, and
                                             ;  (ii) address of divisor
                                             ;  - result/quotient in MReg / $91-$92
                                             ;  - remainder in DVEndMSB / $F2-$F3
DoBell         EQU            $1115          ; ring bell and return 'A' = 1 / true
DoGoToXY       EQU            $1118          ; gotoXY (cursor on screen)
HiLight        EQU            $1133          ; inverse characters on screen
```

```
                                        ;  X & Y = screen position to start
                                        ;  accum = # of characters (if 0 = normal)
MultByte        EQU             $1142   ; multiply byte ('X') times byte ('Y')
                                        ;    (result in MReg / $91-$92)
MultWord        EQU             $1145   ; multiply (2) addresses (result in MReg+)
MvLeftRtn       EQU             $1148   ; move in memory routine (negative/left)
                                        ; - followed by destination/source/length
MvRightRtn      EQU             $114B   ; move in memory routine (positive/right)
                                        ; - followed by destination/source/length
PopStack        EQU             $114E   ; removes the last Escape level
PressAny        EQU             $1151   ; 'Press space to continue'
PushStack       EQU             $1157   ; add a level to Escape road map
                                        ; - followed by address of text pString
ReadKB          EQU             $115A   ; read a character from keyboard
RestCursor      EQU             $115D   ; restore to before WriteCom
StrMvRtn        EQU             $116C   ; copy a pString
                                        ; - followed by destination/source
StrWrRtn        EQU             $116F   ; writes a string at a fixed location
                                        ;  on the screen
                                        ;  (follow JSR with col/row/string address)
WriteOne        EQU             $1181   ; write one character / precede with LDX
                                        ;  (also accepts console commands)
WritePRtn       EQU             $1187   ; propagates a character (repeats)
                                        ;  (X) times of (Y) character
WriteCom        EQU             $118A   ; writes a string on last line
                                        ;  (follow JSR with string address)
StateByte0      EQU             $11B0   ; one (of many) 'state' save bytes
DrawBox         EQU             $11BA   ; draws rectangle mousetext box on screen
                                        ; - followed by column/row/width/height
BoxDone         EQU             $11BD   ; restores some $1Exx stuff
SaveScr         EQU             $11C3   ; JMP $1A62 / 4C 62 1A
RestoreScr      EQU             $11C6
CommandJump     EQU             $1DF5   ; JMP $1E02 / execute loaded command
CommandGo       EQU             $1E02   ; JMP here to start $1E00+ command

* AppleWorks SEG $27 (Organizer Main) Equates
OAQMenuCnt      EQU             $4611   ; # of <OA-Q> <OA-x> selections


* AppleWorks Subhost Equates
DCentury        EQU             $AB13   ; # of century (e.g. - 19 or 20)
DYear           EQU             $AB14   ; # of year (0 - 127)
DMonth          EQU             $AB15   ; # of month
DDay            EQU             $AB16   ; # of day
ReadClock       EQU             $AB2E   ; ReadClock / write to DYear/DMonth/DDay
NewGetStr       EQU             $AB34   ; routine to read string into NewStr
                                        ;  - put length in 'A'
WriteText       EQU             $AB37   ; writes multiple strings
                                        ;  - follow with address of string 'block'
                                        ;  - block contains:   (i) pString for 1
                                        ;                      (ii) column for 1
                                        ;                     (iii) row for 1
                                        ;                      (iv) repeat i/ii/iii
                                        ;                       (v) $00 = endbyte
newCC2S         EQU             $AB40   ; concatenate (2) strings
                                        ;  (first String has second added
                                        ;    at end)
DoGetMonth      EQU             $AB52   ; LDA with # of month;
```

```
                                        ;  pointer to month string in 'Y' and 'A'
newWord2Str     EQU     $AB61           ; convert HEX word to decimal string
                                        ;  (result in StrWork5) [5 character max]
newConv1TP      EQU     $AB64           ; convert HEX byte to decimal string
                                        ;  (result in StrWork3) [3 character max]
GetStrLowGet    EQU     $B246           ; default low character is $20 <space>
GetStrHiGet     EQU     $B24A           ; default low character is $7F <del>
SMGetBlock      EQU     $D005           ; load a block into main memory
                                        ; (pointer in main / 'gets' to main)
                                        ; - follow with:
                                        ;    (i) address in main to block pointer
                                        ;   (ii) address in main to write data
                                        ;  [CArg/ZReg00 $9E/$9F is returned with
                                        ;     number of bytes retrieved]
SMPutBlock      EQU     $D011           ; puts a block into desktop memory from main
                                        ; - follow with:
                                        ;    (i) address in main to block pointer
                                        ;     [if possible, uses existing pointer;
                                        ;       otherwise new pointer stored at
                                        ;       this address]
                                        ;   (ii) address in main to read data
                                        ;  (iii) length of area to read
SMRelBlock      EQU     $D01A           ; releases a Desktop memory block
                                        ;  - follow with address of pointer
DispEsc         EQU     $D026           ; display Escape situation & help
GetMenuBar      EQU     $D032           ; inverse horizontal menu bar at bottom
                                        ; - follow with:
                                        ;   (i) DA for <ESC> address;
                                        ;  (ii) DB (number of items);
                                        ; (iii) description #x;
                                        ;  (iv) routine #x;
                                        ;   (v) description #x+;
                                        ;  (vi) routine #x+;
ReleaseKB       EQU     $D041           ;
UndoHelp        EQU     $D047           ;
ZapPtrs         EQU     $D06D           ; LDX with module; +$80 adds clipboard
MoveToAux       EQU     $D076           ; follow with destination/source/length

* AppleWorks Init Manager Equates
imSavePatch     EQU     $3006           ; Patch Manager save routine in SEG.IM
InitAdr         EQU     $4000           ; load address for Init files
PatchAdr        EQU     $BB00           ; initial load address for patch code
                                        ;  (NOTE: uses ProDOS I/O buffer -
                                        ;          1K max length -
                                        ;           $BB00 - $BEFF)

* SEG $2D / CalendarMY Equates
CalCodeStart    EQU     $5200           ; Relocate calendar code here
                                        ;



* Literals

BaseColumn      EQU     $06
Offset          EQU     $14             ; 20
SundayCol       EQU     0*4+BaseColumn+Offset         ; $1A/26
MondayCol       EQU     1*4+BaseColumn+Offset         ; $1E/30
TuesdayCol      EQU     2*4+BaseColumn+Offset         ; $22/34
```

```
WednesdayCol    EQU             3*4+BaseColumn+Offset         ; $26/38
ThursdayCol     EQU             4*4+BaseColumn+Offset         ; $2A/42
FridayCol       EQU             5*4+BaseColumn+Offset         ; $2E/46
SaturdayCol     EQU             6*4+BaseColumn+Offset         ; $32/50


************************************************************

                ORG             InitAdr         ; ($4000)
                TYP             $06             ; create binary file


*********************************
*          Init Header          *
*********************************
START
                JMP             IStart          ; skip over header

**-------------------------------------------------------------

                ASC             'mb'            ; Init ID Bytes (AW 5.1)
                DB              $34             ; Init Version - programmer assigned
                                                ;  e.g. - $0A/1.0 $0B/1.1 $34/5.2
                STR             'CalendarMY'    ; Init Screen Name

                HEX             0000            ; Init Header End Bytes

**-------------------------------------------------------------

IStart

                LDA             AWVersion       ; AppleWorks version #
                CMP             #$33            ; Is it Version 5.1?
                BNE             Done            ; disregard - wrong version

StoreCode       JSR             SMPutBlock      ; store calendar code in DT memory
                DA              CalCodePtrAdr   ; save pointer here for retrieval ($4xxxx)
                DA              Code2End        ; start of code to store ($4xxx)
                DA              CalCodeEnd-CalCodeStart       ; length of code to store


PatchH27        JSR             imSavePatch     ; call patch manager
                DW              Code1           ; beginning of patch1 code ($40xx)
                DW              Code2-Code1     ; length of patch code
                DW              $0027           ; SEG number to patch
                                                ;  ($27 = Organizer Main SEG)


PatchH2D1       JSR             imSavePatch     ; call patch manager
                DW              Code2           ; beginning of patch2 code ($40xx)
                DW              Code2End-Code2  ; length of patch code
                DW              $002D           ; SEG number to patch
                                                ;  ($2D = Calendar SEG)


Done            RTS                             ; back to Init Manager

```

```
**-------------------------------------------------------------

Code1           EQU             *               ; (will be $40xx)

                ORG             PatchAdr        ; (Patching Code is moved and run
                                                ;   @ $BB00 by Init Manager)

                HEX             0000            ; Init begin bytes for SEG $27 Patch


                LDA             #$0D            ; 13
                STA             OAQMenuCnt      ; increase # of <OA-Q><OA-x> valid
                                                ;  commands from $0C/12 to $0D/13
                                                ;   (<OA-M> is 13th command in list)

                RTS

**-------------------------------------------------------------
                ORG                             ; ($4xxx+)
Code2

                ORG             PatchAdr        ; (Patching Code is moved and run
                                                ;   @ $BB00 by Init Manager)

                HEX             0000            ; Init begin bytes for SEG $2D Patch

                JSR             SMGetBlock      ; retrieve calendar code from DT memory

                DA              CalCodePA       ; address of pointer stored by SMPutBlock
                                                ;  at running address ($BBxx+)
                DA              CalCodeStart    ; locate code here ($5200)

                RTS                             ;

RunningPtr      EQU             *               ; ($BBxx=+)

                ORG                             ; ($4xxx+)
CalCodePtrAdr                                   ; pointer stored here by SMPutBlock above

                ORG             RunningPtr      ; return to $BBxx addressing

CalCodePA       DA              $0000           ; pointer to stored code block ($BBxx+)

**-------------------------------------------------------------

                ORG                             ; ($4xxx+)
Code2End


**-------------------------------------------------------------

                ORG             CalCodeStart    ; ($5200)

CalCode         DA              CalCodeEnd      ; SEG $2D end address

                JMP             Begin           ; start routine

**-------------------------------------------------------------
```

```
349
350 TitleStr       STR             'Multi-Year Calendar'
351
352 DIStr          STR             'Desktop Index'
353
354 CalOptions     STR             'Calendar Options'
355
356 SelectionStr   STR             'Selection'
357
358 EnterYear1Str  STR             'Enter year'
359
360 AboutStr       STR             'About Calendar'
361
362 SpaceBarStr    STR             'Press spacebar'
363
364 EnterYearStr   STR             'Enter year (1582 - 9999): '
365
366 YearStr        STR             '20xx'
367
368                 DB              $00             ; allow for 5-character year string
369
370 DaysStr        STR             'Sun Mon Tue Wed Thu Fri Sat'
371
372 UseArrowsStr   STR             'Use arrow keys to scroll month and year or press Return for
…  more options '
373
374 MonthNum       DB              $01             ; Month written here (January default)
375
376 DaysInMonth    DB              $00             ; table of days in month
377
378                 DB              31              ; January # of Days (Month #1)
379 FebDays        DB              28              ; February # of Days (non-leap)
380                 DB              31              ; March # of Days
381                 DB              30              ; April # of Days
382                 DB              31              ; May # of Days
383                 DB              30              ; June # of Days
384                 DB              31              ; July # of Days
385                 DB              31              ; August # of Days
386                 DB              30              ; September # of Days
387                 DB              31              ; October # of Days
388                 DB              30              ; November # of Days
389                 DB              31              ; December # of Days (Month #12)
390
391
392 HCTopEntry     DB              $00             ; saved setting for <OA-H> top line
393 HCBtmEntry     DB              $18             ; saved setting for <OA-H> bottom line
394
395                 DS              14
396
397 **--------------------------------------------------------------
398
399 Begin
400
401                 INC             TOActive        ; disable TimeOut until we un-patch
402                                                 ;  several routines on exit
403
404                 LDA             CursorOnSw      ; to restore on exit
405                 STA             CursorEntry
```

```
406
407                LDA             StateByte0      ; ($11B0) to restore on exit
408                STA             State0Entry
409
410                JSR             ClearDA         ; clear work area on screen
411
412                JSR             PushStack       ; set title and add level to Esc road map
413                DA              DIStr           ; "Desktop Index"
414
415                JSR             PushStack       ; set title and add level to Esc road map
416                DA              TitleStr        ; "Multi-Year Calendar"
417
418
419 **--------------------------------------------------------------
420 * Patch top and bottom line settings for PRINTED <OA-H> routine
421
422                LDA             HardCopyTop     ; setting for <OA-H> top line
423                STA             HCTopEntry      ; save to restore on exit
424                LDA             #$02            ; discard top (2) lines
425                STA             HardCopyTop     ;
426                LDA             HardCopyBtm     ; setting for <OA-H> bottom line
427                STA             HCBtmEntry      ; save to restore on exit
428                LDA             #$16            ; discard bottom (2) lines
429                STA             HardCopyBtm     ;
430
431 **--------------------------------------------------------------
432 * Patch $1E00 loader routine JMP to change <OA-H> and DrawBox settings
433
434                LDA             #<Change1ECmds ; ($52xx+)
435                STA             CommandJump+1  ; $1DF6
436                LDA             #>Change1ECmds ; ($52xx+)
437                STA             CommandJump+2  ; $1DF7
438
439 **--------------------------------------------------------------
440 * Start drawing calendar
441
442 ReBegin        JSR             DrawBox         ; draw calendar outline rectangle
443                DB              $18             ; column - 24
444                DB              $06             ; row - 6
445                DB              $20             ; width - 32
446                DB              $0B             ; height - 11
447
448
449 **--------------------------------------------------------------
450
451 * Write Month separator dotted line
452
453                LDX             #$1A            ; column - 26
454                LDY             #$07            ; row - 7
455                JSR             DoGoToXY
456
457                LDY             #$2D            ; Normal '-' character
458                LDX             #$1C            ; 28 characters wide
459                JSR             WritePRtn       ; write 'Y' char 'X' times
460
461 * Read and store CURRENT month #
462
463                JSR             ReadClock       ; ReadClock - writes to DMonth
```

```
464                 LDA             DMonth          ; Load Month
465                 STA             MonthNum        ; Store Current Month # here
466                 STZ             NewStr          ; initialize NewStr
467
468 *  Calculate and save actual CURRENT FullYear based on AppleWorks stored date
469
470                 LDX             DCentury        ; 2-digit century in (1) hex byte
471                 LDY             #$64            ; 100
472                 JSR             MultByte        ; result in MReg/MReg+1
473
474                 LDA             DYear           ; (0-127)
475                 CMP             #$64            ; 100 or more?
476                 BCC             :R
477                 SEC
478                 SBC             #$64            ; subtract 100 years to get year
479                 CLC
480
481 :R
482
483 *               addnum -  add 1-byte number (Year) to 2-byte number (Century)
484
485 *               CLC                             ; carry already cleared
486                 ADC             MReg            ; holds full century (e.g. 2000)
487                 STA             MReg            ; write back low byte
488                 BCC             :S              ; high byte not changed
489                 INC             MReg+1
490
491 :S              LDA             MReg            ; save MReg contents for re-use
492                 STA             FullYear        ;
493                 STA             ThisYear        ; save for day highlight routine
494                 LDA             MReg+1          ;
495                 STA             FullYear+1      ;
496                 STA             ThisYear+1      ; save for day highlight routine
497
498 *
499
500 YearChange      JSR             CalcCalNum      ; gets Year in ASCII and Calendar Number
501
502
503 WriteDays       LDA             MonthNum        ; # of month
504                 BNE             :B              ; Branch if Month Defined by ReadClock
505
506                 LDA             #$01            ; otherwise define as January
507
508 :B              JSR             DoGetMonth      ; lda #month - ptr to string in Y,A
509                 STY             :C
510                 STA             :D
511                 JSR             StrMvRtn        ; move Month Str to NewStr
512                 DA              NewStr          ;
513 :C              DB              $00             ; low byte of Month Str
514 :D              DB              $00             ; high byte of Month Str
515
516
517                 LDA             CalendarNum     ; gets calendar # from 0 - 13
518                                                 ;  (0-6 = non-leap; 7-13 = leap)
519                 CMP             #$07            ;
520                 BCS             :T              ; IS leap year (29 Days in February)
521
```

```
                LDA             #28             ; 28 days
                STA             FebDays
                BRA             :U

:T              LDA             #29             ; 29 days
                STA             FebDays

:U              LDX             NewStr
                INX                             ; increase length of NewStr by one
                LDA             #$20            ; load 'space' character
                STA             NewStr,X        ; add space to end of Month Str
                STX             NewStr          ; increase length of NewStr by one
                JSR             newCC2S         ; add 2nd string to 1st
                DA              NewStr          ; NewStr contains Month Str
                DA              YearStr         ; ASCII Year Str
                JSR             StrWrRtn
                DB              $FF             ; centered (columns)
                DB              $06             ; row
                DA              NewStr          ; now contains month and year

                LDA             #$20            ; width
                LDX             #$18            ; start at column
                LDY             #$06            ; row
                JSR             HiLight         ; inverse month and year row

                JSR             StrWrRtn
                DB              $1A             ; column 26
                DB              $08             ; row 8
                DA              DaysStr         ; Sun Mon Tue Wed Thu Fri Sat

                JSR             BuildCalendar   ; write calendar guts

**---------------------------------------------------------------

ReadKeys

                LDA             #$60            ; RTS
                STA             UndoHelp        ; don't display 'k Avail' message here
                                                ;  to allow for additional text in WriteCom

                JSR             WriteCom        ; write str on last line
                DA              UseArrowsStr    ; "Use arrows ..."

                LDA             #$4C            ; JMP
                STA             UndoHelp        ; re-enable 'k Avail' message here

                LDA             FromAbout       ; coming from 'About' screen?
                BEQ             :RK
                JSR             SaveScr         ;
                STZ             FromAbout       ;

:RK             JSR             ReadKB
                CMP             #$1B            ; check for 'escape' key
                BNE             :E              ; examine keypress

**---------------------------------------------------------------
* Exit on <esc>

```

```
580               LDA             HCTopEntry      ; saved setting for <OA-H> top line
581               STA             HardCopyTop     ; restore on exit
582               LDA             HCBtmEntry      ; saved setting for <OA-H> bottom line
583               STA             HardCopyBtm     ; restore on exit
584
585               LDA             #<CommandGo     ; restore normal JMP $1E02
586               STA             CommandJump+1   ;  on exit
587               LDA             #>CommandGo     ;
588               STA             CommandJump+2   ;
589
590
591               STZ             $1DC2           ; initialize 'Y' command byte to
592                                               ;  require reload of patched <OA-H>
593                                               ; {not confirmed the necessity here}
594                                               ; {BoxDone may re-set this at $04, anyway}
595
596
597               JSR             PopStack        ; removes "Multi-Year Calendar"
598
599               JSR             PopStack        ; removes "Desktop Index"
600
601
602               JSR             BoxDone         ; $11BD (restores some $1Exx stuff)
603
604               STZ             TOActive        ; re-enable TimeOut after un-patches
605                                               ;  and forced re-loads
606
607 ************************************************************
608
609 * Experimental - determine cause of <OA-Q><OA-C> hang/crash in DejaIIx if
610 *  that keystroke combo is used prior to ANY word processor file being
611 *  added to the desktop
612
613 * UPDATE: Cleanest way to fix is to add a macro to the startup macro that
614 *          adds a new word processor file from scratch and then removes it.
615 *         Place the following in <ba-[> between sa-% : and msg ' Default Macros
616 *         Sucessfully Installed - Press any key ' :
617
618
619 * // addition to initialize AWP settngs so that items copied to word
620 * // processor clipboard and then accessed via OA-Q and OA-C will not
621 * // crash or hang if an AWP document was not previously accessed
622
623 * t(2) = peek #totalfiles :
624 * if t(2) = 0 :
625 * display 0 :
626 * >1< rtn >3< rtn >1< rtn >n< rtn : esc >4< rtn : rtn > 3< rtn >y<
627 * display 1 :
628 * endif :
629 * t(2) = 0 :
630
631 * // end addition to prevent crash or hang
632
633 ************************************************************
634
635               RTS                             ; return if 'escape' pressed
636
637 **-------------------------------------------------------------
```

```
638
639 :E             CMP             #$0D            ; check for Return <CTRL-M>
640                 BNE             :EN
641                 JMP             MoreMenu        ; year/about/side-by-side
642 **--------------------------------------------------------------
643 :EN             CMP             #$15            ; check for right arrow <CTRL-U>
644                 BEQ             MonthPlus       ;
645 **--------------------------------------------------------------
646                 CMP             #$2B            ; check for '+' key
647                 BEQ             MonthPlus       ;
648 **--------------------------------------------------------------
649                 CMP             #$08            ; check for left arrow <CTRL-H>
650                 BEQ             MonthMinus      ;
651 **--------------------------------------------------------------
652                 CMP             #$2D            ; check for '-' key
653 **--------------------------------------------------------------
654                 BEQ             MonthMinus      ;
655 **--------------------------------------------------------------
656                 CMP             #$0A            ; check for down arrow <CTRL-J>
657                 BEQ             YearMinus       ;
658 **--------------------------------------------------------------
659                 CMP             #$0B            ; check for up arrow <CTRL-K>
660                 BEQ             YearPlus        ;
661 **--------------------------------------------------------------
662                 JSR             DoBell          ; ring bell - try again
663                 JMP             ReadKeys        ; go back - keep reading
664
665 **--------------------------------------------------------------
666 MonthPlus       LDX             MonthNum        ; load month #
667                 CPX             #$0C            ; December? / 12
668                 BCC             :F              ; Jan-Nov?
669 **--------------------------------------------------------------
670                 LDX             #$01            ; roll to January / 01
671                 STX             MonthNum
672                 BRA             YearPlus
673
674 **--------------------------------------------------------------
675 :F              INX                             ; increase by 1 month
676                 BRA             NewMonth        ;
677
678 **--------------------------------------------------------------
679 MonthMinus     LDX             MonthNum        ; load month #
680                 CPX             #$02            ; February? / 02
681                 BCS             :G              ; Feb-Dec?
682
683 **--------------------------------------------------------------
684                 LDX             #$0C            ; roll to December / 12
685                 STX             MonthNum
686                 BRA             YearMinus
687
688 **--------------------------------------------------------------
689 :G              DEX                             ; decrease by 1 month
690
691 **--------------------------------------------------------------
692 NewMonth        STX             MonthNum        ; store new month # selected
693
694                 JSR             WipeHeader
695
```

```
                JMP             WriteDays       ; re-generate days

**--------------------------------------------------------------
YearPlus        LDA             FullYear
                CMP             GregEnd         ; 9999 is maximum year
                BNE             :YP             ; OK to increment year
                LDA             FullYear+1
                CMP             GregEnd+1
                BEQ             :YQ             ; Do NOT increment year

:YP             INC             FullYear        ; increase year by 1
                BNE             NewYear
                INC             FullYear+1
:YQ             BRA             NewYear
**--------------------------------------------------------------
YearMinus       LDA             FullYear
                CMP             GregStart       ; 1582 is minimum year
                BNE             :FF             ; OK to decrement year
                LDA             FullYear+1
                CMP             GregStart+1
                BEQ             NewYear         ; Do NOT decrement year

:FF             LDA             FullYear
                BNE             :GG
                DEC             FullYear+1
:GG             DEC             FullYear        ; decrease year by 1
**--------------------------------------------------------------
NewYear         JSR             WipeHeader

                JMP             YearChange      ; re-generate year calendar

**--------------------------------------------------------------
WipeHeader      LDX             #$1B            ; column 27
                LDY             #$06            ; row 6
                JSR             DoGoToXY
                LDY             #$20            ; 'space' character
                LDX             #$1B            ; 27
                JSR             WritePRtn       ; write 'Y' char 'X' times
                RTS

**--------------------------------------------------------------
EnterYear
                JSR             PopStack        ; remove Calendar Options
                JSR             PushStack       ; set title and add level to Esc road map
                DA              EnterYear1Str   ; "Enter year"

                JSR             WriteCom        ; write on command (bottom) line
                DA              EnterYearStr    ; prompt to enter year
                LDA             #00
                STA             NewStr          ; init Newstr

* patch NewGetStr routine to accept only numbers
:PP             LDA             #$30            ; "0"
                STA             GetStrLowGet    ; limit to numbers
                LDA             #$3A            ; "9" + 1
                STA             GetStrHiGet     ; limit to numbers

                LDA             #4              ; limit year length to (4) digits
```

```
754                JSR             NewGetStr       ; put Year string at NewStr ($0D85)
755
756 * un-patch NewGetStr routine to accept all characters
757                LDA             #$20            ; <space>
758                STA             GetStrLowGet    ; restore to defaults
759                LDA             #$7F            ; <delete>
760                STA             GetStrHiGet     ; restore to defaults
761
762
763                LDA             FoundEsc        ; if <ESC> key, exit
764                BNE             :HH             ; Yes, escaped out of EnterYear
765
766                LDA             FoundCmd        ; if first character is not #, try again
767                BEQ             :QQ             ; Cmd not set -- IS a number, proceed
768                JSR             DoBell          ; ring error bell and return 'A' = 1 / true
769                BNE             :PP             ; try again
770
771 * have entry - now convert ASCII year to Hex Year
772 :QQ            JSR             Asc2Hex         ; convert ASCII year in NewStr to
773                                                ;  2-byte Hex in CArg/CArg+1
774
775
776 * check for years 1582 - 9999
777                LDA             CArg+1
778                CMP             GregEnd+1       ; 9999 is maximum year
779                BCC             :YMin           ; high byte is less, now check for too low
780                BNE             EnterYear       ; high byte too high, try to enter again
781
782                LDA             CArg            ; High Byte = GregEnd High Byte
783                CMP             GregEnd         ; 9999 is maximum year
784                BCC             :YMin           ; low byte is less, now check for too low
785                BNE             EnterYear       ; low byte too high, try to enter again
786
787
788 :YMin          LDA             CArg+1
789                CMP             GregStart+1     ; 1582 is minimum year
790                BCC             EnterYear       ; high byte too low, try again
791                BNE             :YOK            ; high byte higher, proceed
792
793                LDA             CArg            ; High Byte = GregStart High Byte
794                CMP             GregStart
795                BCC             EnterYear       ; low byte too low, try to enter again
796
797 * Year is OK - Transfer to FullYear and Go Back
798 :YOK           LDA             CArg            ;
799                STA             FullYear        ;
800                LDA             CArg+1          ;
801                STA             FullYear+1
802
803
804                JSR             PopStack        ;
805                JMP             NewYear         ;
806
807 :HH            JSR             PopStack        ; remove Enter Year
808
809                JMP             ReadKeys        ;
810
811
```

```
**-------------------------------------------------------
MoreMenu        JSR             PushStack       ;
                DA              CalOptions      ;

                JSR             WriteCom        ; write on command (bottom) line
                DA              SelectionStr    ; prompt to enter year

                JSR             GetMenuBar      ;
                DA              GoBack          ;
                DB              2               ;
                DA              EnterYear1Str   ;
                DA              EnterYear       ;
                DA              AboutStr        ;
                DA              About           ;


GoBack          JSR             PopStack        ;

                JMP             ReadKeys        ;



About


                JSR             PopStack        ; Remove CalOptions
                JSR             PushStack       ;
                DA              AboutStr        ;


                JSR             DrawBox         ; draw about rectangle
                DB              $03             ; column - 3
                DB              $03             ; row - 3
                DB              $49             ; width - 73
                DB              $12             ; height - 18

                JSR             WriteText       ;
                DA              AboutStrings    ;



                JSR             PressAny        ;
                JSR             PopStack        ;
                JSR             ClearDA         ;


                INC             FromAbout       ;

                JMP             ReBegin         ;

FromAbout       DB              $00             ;

AboutStrings    DB              :AB-*-1         ; Leading Length Byte
                DB              $0A             ; Inverse
                ASC             ' Multi-Year Perpetual Calendar for AppleWorks 5.1 '
                DB              $0B             ; Normal
:AB             DB              $FF             ; centered
                DB              $04             ; row 4
```

```
                STR             "SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS"
                                        ; MouseText middle bar
                DB              $0F     ; column 15
                DB              $05     ;
                STR             'Version 5.2 Copyright (c) 2024 Hugh Hood '
                DB              $FF     ;
                DB              $06     ;
                STR             'Based upon an unfinished and unimplemented version included
…with'
                DB              $FF     ;
                DB              $08     ;
                STR             'AppleWorks 5.1 in 1995 as a blanked screen reading simply:'
                DB              $FF     ;
                DB              $09     ;
                STR             'AppleWorks Monthly Calendar by Randy Brandt'
                DB              $FF     ;
                DB              $0B     ;
                STR             'Will it ever do anything?'
                DB              $FF     ;
                DB              $0C     ;
                STR
…"LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL"
                                        ; MouseText top bar
                DB              $FF     ;
                DB              $0E     ;
                STR             'Type: Gregorian Calendar spanning years 1582 - 9999'
                DB              $0F     ;
                DB              $0F     ;
                DB              :AC-*-1 ; Leading Length Byte
                ASC             'To access: <'
                ASC             "A"     ; MouseText open-apple
                ASC             '-Q> <'  ;
                ASC             "A"     ;
                ASC             '-M> from anywhere within AppleWorks'
:AC             DB              $0A     ; column 10
                DB              $10     ; row 16
                DB              :AD-*-1 ; Leading Length Byte
                ASC             'To print: <'  ;
                ASC             "A"     ; MouseText open-apple
                ASC             '-H> selects ',27,'Print screen',27,' option modified to'
:AD             DB              $0B     ; column 11
                DB              $11     ;
                STR             'exclude headings and footings and include 0.00"'
                DB              $16     ; column 22
                DB              $12     ; line 18
                STR             'left and right screen margins'
                DB              $16     ; column 22
                DB              $13     ; line 19

                DB              $00     ; endbyte

**---------------------------------------------------------------
BuildCalendar   LDA             #$0A    ; row $0A / 10
                STA             :I+1    ;

**---------------------------------------------------------------
:H              LDX             #$1B    ; column $1B / 27
:I              LDY             #$00    ; start at Row $0A/10
```

```
926                JSR             DoGoToXY
927                LDX             #$1B            ; write 27 spaces
928                LDY             #$20            ; 'space' character
929                JSR             WritePRtn       ; write 'Y' char 'X' times
930                INC             :I+1            ; increase Row #
931                LDA             :I+1            ;
932                CMP             #$10            ; row # hit $10/16 yet?
933                BCC             :H              ; repeat until blank out 16 rows
934
935 **------------------------------------------------------------
936 * already calculated before JSR to BuildCalendar routine
937 **------------------------------------------------------------
938
939                LDA             CalendarNum     ;
940                ASL                             ; Multiply by (2)
941                TAX                             ; Index into CalFDoMTables
942                LDA             CalFDoMTables,X
943                STA             :R+1
944                LDA             CalFDoMTables+1,X
945                STA             :R+2
946
947                LDY             MonthNum        ; load month # as table offset
948 :R             LDA             CalendarA,Y     ; default to CalendarA
949                STA             :K              ; column
950
951                LDA             DaysInMonth,Y   ; load number of days in month
952                STA             :J+1            ;
953
954
955                LDA             #$01            ; start days @ 1
956                STA             :M+1            ;
957                LDA             #$0A            ; row # to start 1st DOM
958                STA             :L              ; row
959
960 **------------------------------------------------------------
961 :M             LDA             #$00            ; start days @ 1 then increment
962                JSR             newConv1TP      ; convert Hex to days str
963                LDA             MonthNum        ; load HEX # of month
964                CMP             DMonth          ; DMonth
965                BNE             :N              ; not this month; proceed
966 **------------------------------------------------------------
967                LDA             :M+1            ; day # being written
968                CMP             DDay            ; DDay / today?
969                BNE             :N              ; not this day; proceed
970 **------------------------------------------------------------
971                LDA             ThisYear        ; compare year before hilighting
972                CMP             FullYear        ;
973                BNE             :N              ; year is different, skip ahead
974                LDA             ThisYear+1      ;
975                CMP             FullYear+1      ;
976                BNE             :N              ; year is different, skip ahead
977 **------------------------------------------------------------
978                LDX             #$0A            ; enter console Inverse mode
979                JSR             WriteOne        ; Hilight IF today's date
980 **------------------------------------------------------------
981 :N             JSR             StrWrRtn        ; Write day of month on calendar
982 :K             DB              $00             ; column # to write DOM
983 :L             DB              $00             ; row # to write DOM
```

```
                DA              StrWork3        ;
                LDX             #$0B            ; back to console Normal mode
                JSR             WriteOne
                INC             :M+1            ; proceed to 2,3,etc...
                LDA             :M+1

:J              CMP             #$00            ; # of days in month put here
                BEQ             :O              ; still more days to write
**--------------------------------------------------------------
                BCC             :O              ; still more days to write
**--------------------------------------------------------------
                RTS                             ; no more days - wait for keypress

**--------------------------------------------------------------
:O              LDA             :K              ; column # written
                CMP             #$32            ; column 50 or more?
                BCS             :P              ; yes, column reset to left

**--------------------------------------------------------------
                CLC
                ADC             #$04            ; skip 4 spaces for next write
                STA             :K              ; column #
                BRA             :Q              ;

**--------------------------------------------------------------
:P              LDA             #$1A            ; reset back to column 26
                STA             :K              ; column #
                INC             :L              ; increase row # by 1

**--------------------------------------------------------------
:Q              JMP             :M              ; repeat to write next day

**--------------------------------------------------------------


CursorEntry     DB              $01             ; default is 'On'

State0Entry     DB              $01             ; default is $01

CalendarNum     DB              $00             ; default is Calendar A/1

FullYear        DW              2000            ; default is year 2000

ThisYear        DW              2022            ; default is year 2022

GregStart       DW              1582            ; start year for Gregorian Calendar

GregEnd         DW              9999            ; maximum year for THIS application

CenturyDiv      DW              100             ;

CenturyX4Div    DW              400             ;

YearDiv         DW              4               ;

DaysPerWeek     DW              7               ;

LeapsPerCntry   DW              24              ; 24 leaps years in each Century
```

```
1042
1043 LeapYear       DB              $00             ; $00 = No Leap / $07 = Leap
1044
1045 LeapsThisCent  DW              $0000           ;
1046
1047 NumOfCenturies DW              $0000           ;
1048
1049 YearAndLeaps   DW              $0000           ; total Day-of-Week advances
1050
1051 TotalLeaps     DW              $0000           ; total leaps since beginning
1052
1053 **-------------------------------------------------------------
1054
1055 * Routine to calculate which of (14) possible calendars applies to year given
1056
1057 CalcCalNum
1058                STZ             LeapYear        ; initialize LeapYear for tables
1059                                                ;   (can be 0 or 7)
1060
1061
1062 *  Convert year word to ASCII string for Calendar Display and copy to YearStr
1063                JSR             newWord2Str     ; convert hex year to ASCII year
1064                DA              FullYear        ;  (result in StrWork5)
1065
1066                JSR             StrMvRtn        ; move ASCII year result to YearStr
1067                DA              YearStr         ;
1068                DA              StrWork5        ;
1069
1070
1071 *  Is current year (generally) a Leap Year? (evenly divisible by 4)
1072                JSR             DivWord         ;
1073                DA              FullYear        ; dividend
1074                DA              YearDiv         ; divisor / 4 years
1075
1076                LDA             DVEndMSB        ; remainder of FullYear/4
1077                BNE             NotLeap         ; Every 4 years (generally) is leap year
1078                                                ; (don't need to check DVEndMSB+1 since
1079                                                ;   anything there will be 256 x n, and
1080                                                ;   265 IS evenly divisible by 4)
1081
1082
1083 Leap           LDA             #$07            ;
1084                STA             LeapYear        ;
1085                BRA             CalcLeaps       ;
1086
1087
1088 NotLeap        STZ             LeapYear        ;
1089
1090
1091 *  Calculate # of Leap Years FOLLOWING this Century's century year (xxx1+)
1092 CalcLeaps      JSR             DivWord         ; divide years by 100
1093                DA              FullYear        ;
1094                DA              CenturyDiv      ; 100 years
1095
1096                LDA             MReg            ; # of centuries / quotient
1097                STA             NumOfCenturies  ; to calculate leaps in prior centuries
1098
1099                LDA             DVEndMSB        ; remainder of FullYear/100
```

```
1100                BNE             :AA             ; yes, there IS a remainder
1101
1102 * if an even 100-year century mark is NOT leap (unless 400th year - handled below)
1103                STZ             LeapYear        ; not a leap year / 'F' = 0
1104                STZ             LeapsThisCent   ; no leap years since Century / 'L' = 0
1105                BRA             :BB
1106
1107 * if NOT an even 100-year century mark
1108 :AA            DEC             DVEndMSB        ; subtract 1 year (current year)
1109                                                ;  (no effect on 1st day of THIS year calc)
1110                LDA             DVEndMSB        ;
1111                STA             MWrkX           ;
1112                LDA             DVEndMSB+1      ;
1113                STA             MWrkX+1         ;
1114
1115
1116                JSR             DivWord         ; divide years since Century (less current)
1117                DA              MWrkX           ;  by (4) to get leap years since Century
1118                DA              YearDiv         ; 4 years
1119
1120                LDA             MReg            ; quotient / q
1121                STA             LeapsThisCent   ; # of leap years since Century / 'L' = q
1122
1123 *  Calculate # of Leap Years in all prior Centuries
1124 :BB            JSR             MultWord        ;
1125                DA              NumOfCenturies  ;
1126                DA              LeapsPerCntry   ; 24 leaps per Century
1127
1128 *  Add # of Leap Years in previous Centuries to # of Leap Years since this Century
1129                LDA             LeapsThisCent   ; # of leap years since Century
1130                CLC                             ; prepare to add
1131                ADC             MReg            ; leap years in previous Centuries
1132                STA             TotalLeaps
1133                LDA             LeapsThisCent+1                ;
1134                ADC             MReg+1
1135                STA             TotalLeaps+1
1136
1137 *  Calculate # of Leap Years in every previous 4th century
1138                JSR             DivWord         ; divide years by 400
1139                DA              FullYear        ;
1140                DA              CenturyX4Div    ; 400 years
1141
1142 *  Add 1 extra Leap Year every 4th century to total # Leap Years
1143                LDA             TotalLeaps      ; # of total leap years so far
1144                CLC                             ; prepare to add
1145                ADC             MReg            ; leap years in every previous 4th century
1146                STA             TotalLeaps      ;
1147                LDA             TotalLeaps+1    ;
1148                ADC             MReg+1          ;
1149                STA             TotalLeaps+1    ;
1150
1151 *  If an even 400-year century mark current year IS a leap (subtract it out)
1152                LDA             DVEndMSB        ; remainder from division by 400
1153                BNE             :EE             ; yes, there is a remainder
1154
1155 *  Current 400th year IS a leap so subtract it from # of total leap years so far
1156                LDA             TotalLeaps      ;
1157                BNE             :FF             ;
```

```
1158                DEC             TotalLeaps+1    ; subtract 1 year (current year)
1159 :FF            DEC             TotalLeaps      ;  (no effect on 1st day of THIS year calc)
1160
1161                LDA             #$07            ; mark as leap year
1162                STA             LeapYear        ;
1163
1164
1165 ************************************************************
1166 * IF desired that multiples of 4000 NOT be leap years,
1167 * in spite of their being evenly divisible by 400, add that
1168 * code HERE.
1169 *
1170 * NOTE: Most models treat multiples of 4000 as leap years,
1171 *        but some consider their exclusion as
1172 *        being more accurate.
1173 ************************************************************
1174 * Add multiples of 4000 exclusion code here:
1175 *
1176 * Code ...
1177 *
1178 * End multiples of 4000 exclusion code.
1179 ************************************************************
1180
1181 *   Calculate # of Day-of-Week advances in all prior years back to 0001
1182 *    by adding the number of previous years to the number of previous leap years
1183 *    and dividing by 7, the number of possible days per week. The remainder then
1184 *    gives the day-of-week for January 1st of the current year.
1185 *
1186 *  NOTE: Each passing year adds 1 Day-of-Week to starting Day-of-Week for year.
1187 *         (365 days / 7 days per week equals 52 weeks with a 1 day remainder)
1188
1189 :EE            LDA             FullYear
1190                CLC                             ; prepare to add
1191                ADC             TotalLeaps
1192                STA             YearAndLeaps    ;
1193                LDA             FullYear+1      ;
1194                ADC             TotalLeaps+1    ;
1195                STA             YearAndLeaps+1  ;
1196
1197                JSR             DivWord         ;
1198                DA              YearAndLeaps    ;
1199                DA              DaysPerWeek     ; 7
1200
1201                LDA             DVEndMSB        ; remainder
1202                STA             CalendarNum     ; will yield Calendars 0-6
1203
1204                CLC                             ; prepare to add
1205                ADC             LeapYear        ; will be either '0' or '7'
1206                STA             CalendarNum     ; will yield Calendars 0-13
1207
1208                RTS
1209
1210 **-------------------------------------------------------------
1211
1212 * Routine to convert 1-4 digit ASCII year in NewStr to 2-byte HEX FullYear
1213
1214 Asc2Hex
1215                LDA             #<NewStr        ; copy location of ASCII string
```

```
                STA             AArg            ;  to AArg
                LDA             #>NewStr        ;
                STA             AArg+1          ;

                STZ             CArg            ; initialize CArg, which will
                STZ             CArg+1          ;  contain computed result

                LDX             NewStr          ; get length byte
                STZ             NewStr+1,X      ; zero terminate after last character

                LDY             #$00            ;

Parse           INY                             ; start at 1 ($0D86) and progress
                LDA             (AArg),Y        ;
                CMP             #$30            ; ASCII '0'
                BCC             ResultDone      ; will find $00 at end of ASCII string

                CMP             #$3A            ;
                BCS             ResultDone      ;

                AND             #%11001111      ; will limit to $00 - $09
                                                ;  {drop bits 4 & 5}
                STA             BArg+1          ;
                LDA             CArg+1          ;
                STA             BArg            ;
                LDA             CArg            ;
                ASL                             ; multiply 'A' x 2 / Bit 7 to Carry
                ROL             BArg            ; multiply x 2 / Carry to Bit 0
                ASL                             ; multiply 'A' x 2 / Bit 7 to Carry
                ROL             BArg            ; multiply x 2 / Carry to Bit 0
                ADC             CArg            ; add CArg to 'A'
                STA             CArg            ; store result at CArg
                LDA             BArg            ;
                ADC             CArg+1          ;
                STA             CArg+1          ;
                ASL             CArg            ; multiply CArg x 2 / Bit 7 to Carry
                ROL             CArg+1          ; multiply x 2 / Carry to Bit 0
                LDA             CArg
                ADC             BArg+1          ; add BArg+1 to 'A'
                STA             CArg            ;
                BCC             Parse           ; get next digit

                INC             CArg+1          ;
                BRA             Parse           ; now get next digit

**--------------------------------------------------------------

ResultDone      RTS                             ; go back ... done

**--------------------------------------------------------------

*  Dynamic patches to internal AppleWorks routines loaded at $1E00
*    <OA-H> and DrawBox

*  Change top and bottom lines for <OA-H> Copy and Image to Clipboard
Change1ECmds
                PHA                             ; save to restore

```

```
1274                LDA             $1E01           ; '$FC' stored here for <OA-H> routine
1275                CMP             #$01            ; is it $1E command?
1276                BEQ             :DBN
1277                JMP             DoneHere
1278
1279 :DBN           LDA             $1E00           ; '$FC' stored here for <OA-H> routine
1280                CMP             #$FC            ; is it $1E command?
1281                BEQ             :X              ; process <OA-H> changes
1282
1283                CMP             #$44            ; '$44' stored here for DrawBox routine
1284                BEQ             :DBC
1285                JMP             DoneHere
1286
1287 * Make Changes to DrawBoxCommand {for a better MT box routine}
1288
1289 :DBC           LDA             #$EA            ; NOP out SaveScr Routine
1290                STA             $1E59           ;
1291                STA             $1E5A           ;
1292                STA             $1E5B           ;
1293
1294
1295
1296                LDA             #$DA            ; MT Right Hand Bar '|'
1297                STA             $1E98           ; was MT Left Hand Bar '|'
1298                LDA             #$DF            ; MT Left Hand Bar '|'
1299                STA             $1EA5           ; was MT Right Hand Bar '|'
1300                LDA             #$EA            ; NOP
1301                STA             $1E7A           ;  {was INX}
1302                STA             $1E7B           ;  {was INX}
1303                STA             $1E94           ; don't write space '_' first
1304                STA             $1E95           ;
1305                STA             $1E96           ;
1306                STA             $1EC0           ;  {was INY}
1307                STA             $1EC7           ;  {was INX}
1308                STA             $1EC8           ;  {was INX}
1309                LDA             #$5F            ; underscore '_' for box bottom
1310                STA             $1ECA           ;  {was MT Top Bar / #$CC}
1311                BNE             DoneHere
1312
1313 * Make Changes to <OA-H> Command
1314
1315 :X             LDA             #$17            ; change from #$18 total lines
1316                STA             $1ED4           ;  to #$15 plus (2) margin commands
1317                                                ;  (will be stored @ $C50 / KBHighest
1318
1319                LDA             #$15            ;  (less 2 at top and 1 at bottom)
1320                STA             $1F0A           ; # of screen lines in copy loop
1321
1322                LDA             #$4F            ; change copy to clipboard length
1323                STA             $1EB0           ;  from $52/82 to $4F/79 (gross #)
1324                LDA             #$CD            ; change copy to clipboard length
1325                STA             $1EAE           ;  from $D0/80 to $CD/77
1326                                                ;   (actual characters)
1327                LDA             #$EA            ; NOP
1328                STA             $1E1F           ; disable STZ $A2 instruction
1329                STA             $1E20           ;
1330                LDA             #$02            ; change from #$00 as top line
1331                STA             $A2             ;
```

```
1332
1333 * Experimental to add Zero Margins at top of Clipboard
1334
1335                 LDA             #<ZeroMargins  ;
1336                 STA             $1ECC          ; was $41 ($D041 / ReleaseKB)
1337                 LDA             #>ZeroMargins
1338                 STA             $1ECD          ; was $D0 ($D041 / ReleaseKB)
1339
1340                 BRA             DoneHere
1341
1342
1343 ZeroMargins     JSR             ReleaseKB      ; original command at $1ECB
1344                 LDA             #$08           ; original Clipboard start is $01/0804
1345                 STA             $1ED9          ;  (change to $01/0808)
1346                 JSR             MoveToAux      ; move margin commands to top of Clipboard
1347                 DA              $0804          ; clipboard start (in Aux Mem)
1348                 DA              MarginCmds     ;
1349                 DW              $04            ; two words (4 bytes)
1350
1351 ************************************************************
1352 * Experimental code was inserted here to work around a confirmed Deja IIx bug
1353 * that can be duplicated by doing an <OA-H> (I)mage copy of the Main Menu,
1354 * opening an AWP file, copying part of it to the Clipboard, going back to
1355 * the Main Menu, and doing another <OA-H> (I)mage copy of the Main Menu.
1356 * Deja IIx itself will crash (not the emulated AppleWorks) and reports
1357 * a problem with its NativeCommonPutBlock and NativeAuxPutBlock routines.
1358 * All attempts at a workaround were unsucessful, but it is noted that if
1359 * the Deja IIx Debug Memory window is open, there is no crash, but anomalies
1360 * are still present if Clipboard contains any Printer Options (e.g. - LM/RM).
1361 ************************************************************
1362
1363                 RTS
1364
1365 MarginCmds      DW              $D900          ; LM = 0.00
1366                 DW              $DA00          ; RM = 0.00
1367
1368 CRCmds          DW              $D000          ;
1369                 DW              $D000          ;
1370                 DW              $D000          ;
1371                 DW              $D000          ;
1372
1373 ************************************************************
1374
1375 DoneHere
1376                 PLA                            ; now restore accumulator
1377                 JMP             CommandGo      ; execute command
1378
1379
1380 **-------------------------------------------------------------
1381 ** Address table for calendar first day of month tables
1382
1383 CalFDoMTables   DA              CalendarA      ; #0
1384                 DA              CalendarB      ; #1
1385                 DA              CalendarC      ; #2
1386                 DA              CalendarD      ; #3
1387                 DA              CalendarE      ; #4
1388                 DA              CalendarF      ; #5
1389                 DA              CalendarG      ; #6
```

```
                DA              CalendarH       ; #7
                DA              CalendarI       ; #8
                DA              CalendarJ       ; #9
                DA              CalendarK       ; #10
                DA              CalendarL       ; #11
                DA              CalendarM       ; #12
                DA              CalendarN       ; #13

** 1st Day of Month Tables for 14 possible calendars (A-N / 0-13)

CalendarA       DB              $00
                DB              SundayCol       ; January
                DB              WednesdayCol    ; February
                DB              WednesdayCol    ; March
                DB              SaturdayCol     ; April
                DB              MondayCol       ; May
                DB              ThursdayCol     ; June
                DB              SaturdayCol     ; July
                DB              TuesdayCol      ; August
                DB              FridayCol       ; September
                DB              SundayCol       ; October
                DB              WednesdayCol    ; November
                DB              FridayCol       ; December

CalendarB       DB              $00
                DB              MondayCol       ; January
                DB              ThursdayCol     ; February
                DB              ThursdayCol     ; March
                DB              SundayCol       ; April
                DB              TuesdayCol      ; May
                DB              FridayCol       ; June
                DB              SundayCol       ; July
                DB              WednesdayCol    ; August
                DB              SaturdayCol     ; September
                DB              MondayCol       ; October
                DB              ThursdayCol     ; November
                DB              SaturdayCol     ; December

CalendarC       DB              $00
                DB              TuesdayCol      ; January
                DB              FridayCol       ; February
                DB              FridayCol       ; March
                DB              MondayCol       ; April
                DB              WednesdayCol    ; May
                DB              SaturdayCol     ; June
                DB              MondayCol       ; July
                DB              ThursdayCol     ; August
                DB              SundayCol       ; September
                DB              TuesdayCol      ; October
                DB              FridayCol       ; November
                DB              SundayCol       ; December

CalendarD       DB              $00
                DB              WednesdayCol    ; January
                DB              SaturdayCol     ; February
                DB              SaturdayCol     ; March
                DB              TuesdayCol      ; April
                DB              ThursdayCol     ; May
```

```
1448                DB              SundayCol       ; June
1449                DB              TuesdayCol      ; July
1450                DB              FridayCol       ; August
1451                DB              MondayCol       ; September
1452                DB              WednesdayCol    ; October
1453                DB              SaturdayCol     ; November
1454                DB              MondayCol       ; December
1455
1456 CalendarE       DB              $00
1457                DB              ThursdayCol     ; January
1458                DB              SundayCol       ; February
1459                DB              SundayCol       ; March
1460                DB              WednesdayCol    ; April
1461                DB              FridayCol       ; May
1462                DB              MondayCol       ; June
1463                DB              WednesdayCol    ; July
1464                DB              SaturdayCol     ; August
1465                DB              TuesdayCol      ; September
1466                DB              ThursdayCol     ; October
1467                DB              SaturdayCol     ; November
1468                DB              TuesdayCol      ; December
1469
1470 CalendarF       DB              $00
1471                DB              FridayCol       ; January
1472                DB              MondayCol       ; February
1473                DB              MondayCol       ; March
1474                DB              ThursdayCol     ; April
1475                DB              SaturdayCol     ; May
1476                DB              TuesdayCol      ; June
1477                DB              ThursdayCol     ; July
1478                DB              SundayCol       ; August
1479                DB              WednesdayCol    ; September
1480                DB              FridayCol       ; October
1481                DB              MondayCol       ; November
1482                DB              WednesdayCol    ; December
1483
1484 CalendarG       DB              $00
1485                DB              SaturdayCol     ; January
1486                DB              TuesdayCol      ; February
1487                DB              TuesdayCol      ; March
1488                DB              FridayCol       ; April
1489                DB              SundayCol       ; May
1490                DB              WednesdayCol    ; June
1491                DB              FridayCol       ; July
1492                DB              MondayCol       ; August
1493                DB              ThursdayCol     ; September
1494                DB              SaturdayCol     ; October
1495                DB              TuesdayCol      ; November
1496                DB              ThursdayCol     ; December
1497
1498 CalendarH       DB              $00
1499                DB              SundayCol       ; January
1500                DB              WednesdayCol    ; February
1501                DB              ThursdayCol     ; March
1502                DB              SundayCol       ; April
1503                DB              TuesdayCol      ; May
1504                DB              FridayCol       ; June
1505                DB              SundayCol       ; July
```

```
1506                    DB              WednesdayCol    ; August
1507                    DB              SaturdayCol     ; September
1508                    DB              MondayCol       ; October
1509                    DB              ThursdayCol     ; November
1510                    DB              SaturdayCol     ; December
1511
1512 CalendarI          DB              $00
1513                    DB              MondayCol       ; January
1514                    DB              ThursdayCol     ; February
1515                    DB              FridayCol       ; March
1516                    DB              MondayCol       ; April
1517                    DB              WednesdayCol    ; May
1518                    DB              SaturdayCol     ; June
1519                    DB              MondayCol       ; July
1520                    DB              ThursdayCol     ; August
1521                    DB              SundayCol       ; September
1522                    DB              TuesdayCol      ; October
1523                    DB              FridayCol       ; November
1524                    DB              SundayCol       ; December
1525
1526 CalendarJ          DB              $00
1527                    DB              TuesdayCol      ; January
1528                    DB              FridayCol       ; February
1529                    DB              SaturdayCol     ; March
1530                    DB              TuesdayCol      ; April
1531                    DB              ThursdayCol     ; May
1532                    DB              SundayCol       ; June
1533                    DB              TuesdayCol      ; July
1534                    DB              FridayCol       ; August
1535                    DB              MondayCol       ; September
1536                    DB              WednesdayCol    ; October
1537                    DB              SaturdayCol     ; November
1538                    DB              MondayCol       ; December
1539
1540 CalendarK          DB              $00
1541                    DB              WednesdayCol    ; January
1542                    DB              SaturdayCol     ; February
1543                    DB              SundayCol       ; March
1544                    DB              WednesdayCol    ; April
1545                    DB              FridayCol       ; May
1546                    DB              MondayCol       ; June
1547                    DB              WednesdayCol    ; July
1548                    DB              SaturdayCol     ; August
1549                    DB              TuesdayCol      ; September
1550                    DB              ThursdayCol     ; October
1551                    DB              SundayCol       ; November
1552                    DB              TuesdayCol      ; December
1553
1554 CalendarL          DB              $00
1555                    DB              ThursdayCol     ; January
1556                    DB              SundayCol       ; February
1557                    DB              MondayCol       ; March
1558                    DB              ThursdayCol     ; April
1559                    DB              SaturdayCol     ; May
1560                    DB              TuesdayCol      ; June
1561                    DB              ThursdayCol     ; July
1562                    DB              SundayCol       ; August
1563                    DB              WednesdayCol    ; September
```

```
1564                DB              FridayCol       ; October
1565                DB              MondayCol       ; November
1566                DB              WednesdayCol    ; December
1567
1568 CalendarM       DB              $00
1569                DB              FridayCol       ; January
1570                DB              MondayCol       ; February
1571                DB              TuesdayCol      ; March
1572                DB              FridayCol       ; April
1573                DB              SundayCol       ; May
1574                DB              WednesdayCol    ; June
1575                DB              FridayCol       ; July
1576                DB              MondayCol       ; August
1577                DB              ThursdayCol     ; September
1578                DB              SaturdayCol     ; October
1579                DB              TuesdayCol      ; November
1580                DB              ThursdayCol     ; December
1581
1582 CalendarN       DB              $00
1583                DB              SaturdayCol     ; January
1584                DB              TuesdayCol      ; February
1585                DB              WednesdayCol    ; March
1586                DB              SaturdayCol     ; April
1587                DB              MondayCol       ; May
1588                DB              ThursdayCol     ; June
1589                DB              SaturdayCol     ; July
1590                DB              TuesdayCol      ; August
1591                DB              FridayCol       ; September
1592                DB              SundayCol       ; October
1593                DB              WednesdayCol    ; November
1594                DB              FridayCol       ; December
1595
1596
1597 **--------------------------------------------------------------
1598
1599 CalCodeEnd      EQU             *               ; ($5xxx+)
1600
1601 **--------------------------------------------------------------
1602
1603                 SAV             I.CALENDARMY    ;
1604                 LST             OFF
1605
1606                 END
1607
1608 *==============================================================
1609
```