Mountain Hardware, Inc.

LEADERSHIP IN COMPUTER PERIPHERALS

APPLE CLOCK™

OPERATING MANUAL

APPLE CLOCK™

OPERATING MANUAL

(Revision 2, July 1979)

TABLE OF CONTENTS

INDEX OF TABLES

INTRODUCTION

WELCOME TO THE WORLD OF REAL-TIME!

Your Mountain Hardware Apple Clock^{TM} extends the reach of your Apple II* computer by adding the dimension of real time and date in intervals of from 1 millisecond to a little over one year.

On-board battery power keeps the clock running for periods of up to 4 days when your computer is turned off, either intentionally or due to a power outage. If you have down times longer than 4 days, refer to the section in this manual entitled "The Battery".

The Apple Clock is very easy to use because it contains an on-board ROM. This ROM contains software making it easy to obtain the time and date whether you are using Integer BASIC, Applesoft, or Assembly language.

Let your imagination guide you to the many ways in which your Apple Clock can be used.

*Apple II is a trademark of Apple Computer Company.

INSTALLATION

PLUG IN AND GO!

Installing the Apple Clock in your computer is very easy.

1. Turn your computer OFF.

2. Remove the top cover from the computer.

3. Take your Apple Clock and clip the battery connector onto the top of the battery.

4. Plug the Apple Clock into any empty slot on the back of the Apple computer board. You may use any slot EXCEPT SLOT #0. We recommend SLOT #4.

5. Be sure the Apple Clock is firmly seated in its socket.

6. Leave the cover off for the moment. You will need to have access to the switches on the Apple Clock when you set it.

7. Refer to the section in this manual called "Setting the Time".

8. Once the clock has been set, be sure you have changed the 'WRITE PROTECT' switch on the clock according to the instructions in "Setting the Time".

9. YOU ARE FINISHED! You may now replace the cover on your computer.

You are now ready to start reading the time. ONE THING TO REMEMBER - The on-board battery on the clock will take four (4) days to completely charge up. Therefore, leave your computer on continuously for at least four (4) days to completely charge your battery. If this is not done, you cannot be sure that the battery will keep the clock running when you shut your computer off. Once it is charged, however, the clock will keep running for up to 4 days when you shut your computer power off.

THE BATTERY

The Apple Clock is supplied with a rechargeable NiCad battery to keep the clock running when the computer is turned off, or when the power fails. This battery is attached to the Apple Clock on the back side of the board. The battery powers the clock circuitry on the board permitting the clock to keep correct time for periods up to 4 days, if it is fully charged.

> To fully charge the battery,
> the Apple computer must be left
> on for at least 4 days.

This initial charging time of 4 days is designed to maximize the life of the battery. As a general guideline, the battery should charge 2 hours for every 1 hour of use. The battery life is several years, but should be replaced if its performance drops significantly. You may obtain replacements anywhere batteries are sold.

If you anticipate that your computer is going to be turned off for periods longer than 4 days, you may incorporate a larger capacity battery. This can be done by clipping an additional battery clip to the one mounted on the Apple Clock. The two wires from this clip can be run outside the case of the Apple computer and connected to a larger battery. The size of the battery is unimportant, however you must use a battery with a voltage between 7-10V DC.

The battery is intended to support the clock if power fails, or if the computer is turned off for short periods of time. Your computer's lifetime will not be affected by leaving it on continually, and may even be increased. The power consumed by the Apple computer is less than an ordinary light bulb. Consequently we recommend that the Apple computer be left on continually. The clock's battery will keep the clock running if the power fails in your building, or if you turn the computer off for short periods of time (less than 4 days).

SETTING THE TIME

To set the time with the supplied cassette:

1. Load Applesoft into your system.

2. Load the "Set the Time" program supplied with your Apple Clock from cassette. (See Note.)

3. Change the 'WRITE PROTECT' switch to the 'WRITE' position. It is the top switch on the board. Press the switch down on the right side.

4. Set the Leap Year switch. If the current year is a leap year, press the switch on the LEFT. Press the switch down on the RIGHT if it is not a leap year. The leap year switch is the second switch down from the top.

5. Type 'RUN'.

6. Answer the questions that appear on the screen appropriately.

7. After the clock is set, 'WRITE PROTECT' it by pressing the 'WRITE PROTECT' switch down on the LEFT. This prevents the clock from being changed accidently.

The above procedure will need to be performed each January before the 20th of the month. If daylight-saving time is in effect in your area, update the clock as needed.

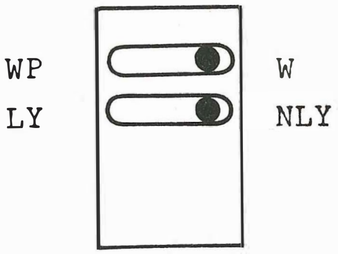
Store the "Set the Time" cassette in a secure spot for future use. A listing of the "Set the Time" cassette is provided here for your reference. Also listed is the assembly language program used by "Set the Time" (Lines 5ØØØ through 5Ø9Ø).

Note: If you are not using an Applesoft card but are using the Apple Disk, you must type 'CALL 3314' before running the program.

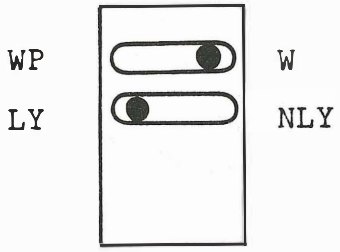
WRITE PROTECT SWITCH

The Apple Clock WRITE PROTECT switch must be in the PROTECT (WP) mode at all times EXCEPT when setting the clock. Otherwise the clock can change time when the computer power is turned off.

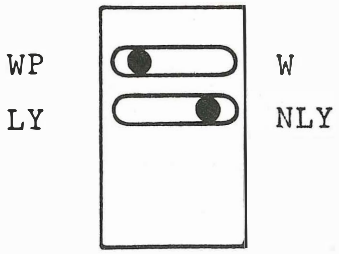
Put the switches in these positions when:



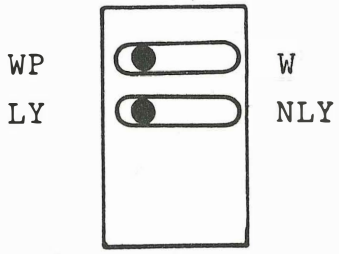
SETTING TIME AND IT IS NOT LEAP YEAR



SETTING TIME AND IT IS LEAP YEAR



THE CLOCK HAS BEEN SET AND IT IS NOT LEAP YEAR



THE CLOCK HAS BEEN SET AND IT IS LEAP YEAR

```
1   REM **** MOUNTAIN HARDWARE'S APPLE CLOCK
2   REM **** COPYRIGHT 1978
3   REM ***  SET THE TIME APPLESOFT  ***
4   REM
5   REM **** ADD OR CHANGE THESE LINES FOR DISK SYSTEM
6   REM
7   REM        20 D$=""  WHERE D$="CONTROL D"
8   REM        21 PRINT D$;"NOMON I,O,C"
9   REM      3020 PRINT D$;"PR#";SLOT
10   REM     3025 PRINT D$;"IN#";SLOT
18   REM     3040 PRINT D$;"IN#0"
20   REM     3045 PRINT D$;"PR#0"
24   REM
26   REM
28   REM
30   CALL  - 936
40   VTAB 10
50   PRINT "MOUNTAIN HARDWARE'S APPLE CLOCK"
60   VTAB 13
70   PRINT "DISPLAY OR SET THE TIME PROGRAM"
71  : PRINT : PRINT : PRINT "SEE LINES 5 THRU 10 FOR DI
      SK SYSTEM"
75   PRINT : PRINT
76   INPUT "INPUT THE CLOCK'S SLOT # ";SLOT
80   PRINT : PRINT
90   INPUT "DO YOU WANT TO SET THE TIME (Y ON N)";I$
95   IF I$ = "N" THEN  CALL  - 936: GOTO 2032
100   REM
110   REM  **** POKE IN THE ADVANCEROUTINE AT LOCATION
      $1000
120   REM
130   FOR I = 1 TO 68
140   READ J
150   POKE 767 + I,J
160   NEXT I
300   PRINT
301   PRINT "GIVE THE CURRENT TIME PLUS 30 SECONDS"
302   PRINT
310   INPUT "INPUT THE MONTH (1-12) ";MTH
320   INPUT "INPUT THE DAY (1-31) ";D
330   INPUT "INPUT THE HOUR (0-23) ";H
340   INPUT "INPUT THE MINUTE (0-59) ";M
350   INPUT "INPUT THE SECONDS (0-59) ";S
360   PRINT
365   PRINT "HIT RETURN WHEN YOU HAVE SET THE LEAP"
370   PRINT "SWITCH CORRECTLY, AND ARE SWITCHED FOR"
380   INPUT "WRITING TO THE CLOCK ";I$
390   PRINT : PRINT
```

```
500  REM
501  REM **** CHECK LEAP YEAR SWITCH
502  REM
505  REM   IF L=1 THEN IT'S A LEAP YEAR
510 L =  PEEK (49280 + 16 * SLOT)
511 L =  INT (L / 64)
512  IF L > 1 THEN L = L - 2
600  REM
601  REM **** FIND DAYS TO DATE -- DTD --
602  REM
605 DTD = 0
610  FOR I = 1 TO MTH
620  READ J
630 DTD = DTD + J
640  NEXT I
650 DTD = DTD + D - 1
660  IF L = 1 AND MTH > 2 THEN DTD = DTD + 1
700  REM
701  REM **** CALCULATE SECONDS TO DATE --STD --
702  REM
710 STD = DTD * 86400 + H * 3600 + M * 60 + S
800  REM
801  REM **** PREPARE SECONDS FOR CLOCK
802  REM
810 TEMP = 896: REM  RAM STORAGE AREA
820 S0 =  INT (STD / 2 ^ 20)
825  POKE TEMP,S0
830 STD = STD - S0 * 2 ^ 20
840 S1 =  INT (STD / 2 ^ 12)
850  POKE TEMP + 1,S1
860 STD = STD - S1 * 2 ^ 12
870 S2 =  INT (STD / 2 ^ 4)
880  POKE TEMP + 2,S2
890 STD = STD - S2 * 2 ^ 4
900  POKE TEMP + 3,STD * 16
910  REM
911  REM **** ALSO SAVE N2 AND N7
912  REM
920  POKE TEMP + 4,SLOT * 16 + 2
930  POKE TEMP + 5,SLOT * 16 + 7
1000  REM
1001  REM **** STOP CLOCK AND CALL ADVANCE ROUTINE
1002  REM
1005 SR = 49280 + SLOT * 16 + 5
1006 SP = 49280 + SLOT * 16 + 6
1010 I =  PEEK (SP)
1020  CALL 768: REM  CALL THE MACHINE LANG ADVANCE ROU
     TINE
2000  INPUT "HIT RETURN AT EXACT TIME ";I$
2010 I =  PEEK (SR): REM   START CLOCK
2020  CALL  - 936
2030  PRINT "DON'T FORGET TO WRITE PROTECT THE CLOCK"
2031  VTAB 24
2032  VTAB 24: PRINT "      HIT RESET TO STOP PROGRAM"
```

```
3000  REM
3001  REM **** DISPLAY THE TIME
3002  REM
3005  VTAB 6: PRINT "  MOUNTAIN HARDWARE'S APPLE CLOCK
     "
3006  PRINT : PRINT "            THE TIME IS"
3010  VTAB 22
3020  IN# SLOT
3025  PR# SLOT
3030  INPUT " ";I$
3040  IN# 0
3045  PR# 0
3050  VTAB 12: HTAB 8
3060  PRINT "  ";I$
3070  GOTO 3010
4999  REM **** ADVANCE SUBROUTINE DATA
5000  REM
5010  DATA 72,8,138,72,152,72,174,132
5020  DATA 3,172,133,3,189,128,192,205
5030  DATA 130,3,208,8,189,129,192,205
5040  DATA 131,3,240,6,185,128,192,76
5050  DATA 12,3,202,202,200,189,128,192
5060  DATA 41,31,205,128,3,208,8,189
5070  DATA 129,192,205,129,3,240,6,185
5080  DATA 128,192,76,37,3,104,168,104
5090  DATA 170,40,104,96
5999  REM **** MONTH DATA
6000  DATA  0,31,28,31,30,31,30,31,31,30,31,30
```

```
                  * MOUNTAIN HARDWARE'S
                  * APPLE CLOCK
                  * ADVANCE ROUTINE
                  * FOR SETTING THE CLOCK
                  * COPYRIGHT 1978
                  * GARY MUHONEN
                  *
                  * SYSTEM EQUATES
                  *
                  DS       EQU   $C080     ;DEVICE SELECT
                  DS1      EQU   $C081     ;DEV SEL +1
                  T0       EQU   $0380     ;TEMP STORAGE
                  T1       EQU   $0381     ;  FOR TIME
                  T2       EQU   $0382     ;  COMPARISON
                  T3       EQU   $0383
                  T4       EQU   $0384
                  T5       EQU   $0385
                  *
                  * PROGRAM STARTS AT $0300
                  *
                           ORG   $0300
                           OBJ   $5000
                  *
                  * SAVE REGISTERS
                  *
0300 48                    PHA
0301 08                    PHP
0302 8A                    TXA
0303 48                    PHA
0304 98                    TYA
0305 48                    PHA
                  *
                  *CHECK LOWEST TWO BYTES OF TIME
                  *AND INCREMENT AS NECESSARY
                  *
0306 AE 84 03     I2       LDX   T4       ;N2
0309 AC 85 03              LDY   T5       ;N7 FOR ADV1
030C BD 80 C0     C2       LDA   DS,X     ;LOAD TIME2 FROM CLOCK
030F CD 82 03              CMP   T2       ;DOES IT EQUAL DESIRED?
0312 D0 08                 BNE   A1       ;NO, GO ADVANCE
0314 BD 81 C0              LDA   DS1,X    ;YES, NOW LOAD TIME3
0317 CD 83 03              CMP   T3       ;DOES IT EQUAL DESIRED
031A F0 06                 BEQ   I0       ;YES, GO ON TO NEXT TIME
031C B9 80 C0     A1       LDA   DS,Y     ;NO, ADVANCE CLOCK (ADV1)
031F 4C 0C 03              JMP   C2       ;GO BACK TO CHECK TILL TIME DESIRED
```

```
                    *
                    *CHECK HIGHEST TWO BYTES OF TIME
                    *AND INCREMENT AS NECESSARY
                    *
0322 CA             IO      DEX    ;SET X TO TIMEO SPOT
0323 CA                     DEX
0324 C8                     INY    ;SET Y TO ADV2
0325 BD 80 CO       CO      LDA    DS,X     ;LOAD TIMEO
0328 29 1F                  AND    #$1F     ;STRIP OFF LEAP YEAR AND INTRPT BITS
032A CD 80 03               CMP    TO       ;DOES IT EQUAL DESIRED?
032D DO 08                  BNE    A2       ;NO, GO ADVANCE
032F BD 81 CO               LDA    DS1,X    ;YES, CHECK TIME1
0332 CD 81 03               CMP    T1       ;DOES IT EQUAL DESIRED?
0335 FO 06                  BEQ    END      ;YES, WE'RE DONE
0337 B9 80 CO       A2      LDA    DS,Y     ;NO, ADVANCE CLOCK (ADV2)
033A 4C 25 03               JMP    CO       ;GO BACK AND TRY AGAIN
                    *
                    *DONE,  SO RECOVER REGS AND RETURN
                    *
033D 68             END     PLA
033E A8                     TAY
033F 68                     PLA
0340 AA                     TAX
0341 28                     PLP
0342 68                     PLA
0343 60                     RTS

--- END ASSEMBLY ---
```

READING THE TIME

QUICK READS

The following tricks may be used to quickly read the time. They simply print the time on the screen. TRY THEM!

MONITOR	* n CTRL K
INTEGER	IN#n
APPLESOFT	IN#n

Hit RESET to stop.
n = the clock's slot number (1-7)

Table 1
QUICK READS

To get back into BASIC after hitting RESET, use one of the following procedures.

BASIC	COMMAND
Integer	CTRL C
Integer with DOS*	3DØG
Integer with Applesoft Card (Switch down)	CTRL C
Integer with Applesoft Card (Switch down) and DOS	3DØG
Applesoft - Cassette	ØG
Applesoft and DOS	3DØG
Applesoft Card (Switch up) and DOS	3DØG

Table 2
RE-ENTRY PROCEDURE

*DOS stands for Disk Operating System.

DISPLAYING THE TIME

The following four programs display the date and time as one line centered on the screen. Take the time to become familiar with these programs. They can easily be incorporated into programs you write later. We suggest you read the sections on Strings in the Applesoft BASIC Manual and the Apple II BASIC Programming Manual.

It is generally a good practice to set the SLOT number the clock is in at the very beginning of a program. Line 1Ø of the program demonstrates this. Elsewhere in the program, use SLOT instead of the number (2,3...). Later, if you move the clock to a different slot, you need change only one line, instead of searching for all the places where the number was specified.

When using Integer BASIC, it is necessary to dimension the strings. Applesoft does not require this.

Lines 4Ø through 9Ø of the Integer BASIC program without DOS should be used when reading the time from the clock. Leaving out Line 5Ø (PR#SLOT) will cause the time to be printed on the screen when an INPUT (Line 7Ø) is done. The time (T$) is passed to BASIC in the following format:

```
MONTH/DAY HOUR;MINUTE;SECOND.FRACTION
03/04 10;13;14.123
```

18 CHAR

When actually doing the input statement (INPUT " ", T$), note that a space is printed to the clock board. This is used so that the data returned from the INPUT statement is the same between Applesoft and Integer BASIC.

There is one difference between the INPUT statement in Applesoft and Integer BASIC.

With Integer BASIC, use

```
INPUT " ", T$
```

With Applesoft, use

```
INPUT " "; T$
```

```
  0 REM  *************  TIME  *************
  3 REM   *** INTEGER BASIC WITHOUT DOS ***
  5 REM
 10 SLOT=4: REM  SET THE SLOT#
 20 DIM T$(25): REM  DIMENSION THE TIME STRING
 30 CALL -936: REM  CLEAR THE SCREEN
 40 IN#SLOT: REM  SET INPUT TO CLOCK BOARD
 50 PR#SLOT: REM  SET OUTPUT TO CLOCK
 60 VTAB 23: REM  PUT CURSOR AT BOTTOM OF SCREEN
 70 INPUT " ",T$: REM   OBTAIN THE TIME
 80 IN#0: REM  RESTORE INPUT TO KEYBOARD
 90 PR#0: REM  RESTORE OUTPUT TO CRT
100 VTAB 12: TAB 10: REM  CENTER THE OUTPUT
110 PRINT T$: REM  OUTPUT TIME TO THE SCREEN
120 GOTO 40: REM  READ TIME AGAIN
```

```
0   REM **********  TIME  ***********
4   REM  *** APPLESOFT WITHOUT DOS ***
5   REM
10  SLOT = 4: REM  SET THE SLOT NUMBER
20   HOME : REM  CLEAR THE SCREEN
30   IN# SLOT: REM  SET INPUT TO CLOCK BOARD
40   PR# SLOT: REM  SET OUTPUT TO CLOCK BOARD
50   VTAB 23: REM  PUT CURSOR AT BOTTOM OF SCREEN
60   INPUT " ";T$: REM  OBTAIN THE TIME
70   IN# 0: REM  RESTORE INPUT TO KEYBOARD
80   PR# 0: REM  RESTORE OUTPUT TO CRT
90   VTAB 12: HTAB 10: REM  CENTER THE OUTPUT
100   PRINT T$: REM  OUTPUT THE TIME
110   GOTO 30: REM  READ TIME AGAIN
```

Things change when DOS (Disk Operating System) is active at the time you are running the program. Read your Disk Operating Manual, especially, "Use of the Disk Operating System From Within a Program".

A good programming technique is to set D$ equal to a CONTROL D, as in Line 2Ø. Then use D$ wherever a CONTROL D is required. Line 3Ø prevents the commands IN# and PR# from being printed to the screen when they are executed.

Compare the programs with DOS and without DOS as you will probably be using both versions. The main difference is that when DOS is active, the IN and PR statements are formatted differently.

```
  0 REM ***********  TIME  ***********
  4 REM *** INTEGER BASIC WITH DOS ***
  5 REM
 10 SLOT=4: REM  SET CLOCK SLOT#
 20 D$="": REM  D$="CONTROL D"
 30 PRINT D$;"NOMONI,O,C": REM  PREVENT DISK COMMAND FROM
     PRINTING ON SCREEN
 35 DIM T$(25): REM  DIMENSION TIME ARRAY
 40 CALL -936: REM  CLEAR THE SCREEN
 50 PRINT D$;"IN#";SLOT: REM  SET INPUT TO CLOCK BOARD
 60 PRINT D$;"PR#";SLOT: REM  SET OUTPUT TO CLOCK BOAR
 70 VTAB 23: REM  PUT CURSOR AT BOTTOM OF SCREEN
 80 INPUT " ",T$: REM   OBTAIN THE TIME
 90 PRINT D$;"IN#0": REM  RESTORE INPUT TO KEYBOARD
100 PRINT D$;"PR#0": REM  RESTORE OUTPUT TO CRT
110 VTAB 12: TAB 10: REM  CENTER OUPUT
120 PRINT T$: REM  OUTPUT TIME TO SCREEN
130 GOTO 50: REM  READ TIME AGAIN
```

```
0   REM  *********  TIME  *********
4   REM  *** APPLESOFT WITH DOS ***
5   REM
10  SLOT = 4: REM  SET THE SLOT NUMBER
20  D$ = "": REM  D$=" CONTROL D"
25   PRINT D$;"NOMONI,O,C": REM  KEEP DISK COMMANDS FRO
      M PRINTING
30   HOME : REM  CLEAR THE SCREEN
40   PRINT D$;"IN#";SLOT: REM  SET INPUT TO CLOCK
50   PRINT D$;"PR#";SLOT: REM   SET OUTPUT TO CLOCK
60   VTAB 23: REM  PUT CURSOR AT BOTTOM OF SCREEN
70   INPUT " ";T$: REM  OBTAIN THE TIME
80   PRINT D$;"IN#0": REM  RESTORE INPUT TO KEYBOARD
90   PRINT D$;"PR#0": REM  RESTORE OUTPUT TO KEYBOARD
100   VTAB 12: HTAB 10: REM  CENTER THE OUPUT
110   PRINT T$: REM  OUTPUT TIME TO SCREEN
120   GOTO 40: REM  READ TIME AGAIN
```

OTHER DISPLAY FORMATS

The previous programs simply print the time as it is given by the clock. At times, it will be desireable to use different formats or to only use part of the time. The DATE AND TIME programs here print the time as:

```
DATE: OCTOBER 31, 1978
TIME: 12:3Ø:45.923
```

The time is read (Line 5Ø to 13Ø) just as it was read in the previous section. Now, however, string manipulation is done to the time (T$). In the Integer BASIC program Lines 16Ø through 21Ø show how to find just the month, day, etc. They are repeated here for convenience.

```
MONTH$=T$(1,2)
DAY$=T$(4,5)
HOUR$ T$(7,8)
MINUTES=T$(10,11)
SECONDS=T$(13,14)
FRAC$=T$(16,18)
```

Table 3
INTEGER TIME STRING MANIPULATION

The various components of T$ can then be manipulated to obtain the desired results. Lines 23Ø to 37Ø test the month string to determine the name of the month. The date and time can then be printed out in whatever format is desired.

The Applesoft program is slightly different. The time (T$), however, is read in the same manner (Lines 5Ø through 13Ø). To obtain the elements of T$, the following string manipulations are done.

```
MTH$=LEFT$(T$,2)
DAY$=MID$(T$,4,2)
HOUR$=MID$(T$,7,2)
MINUTE$=MID$(T$,10,2)
SEC$=MID$(T$,13,1)
FRAC$=RIGHT$(T$,3)
```

Table 4
APPLESOFT TIME STRING MANIPULATION

```
  0 REM ********  DATE AND TIME  ********
  4 REM *** INTEGER BASIC WITHOUT DOS ***
  5 REM
 10 DIM T$(25),MONTH$(10),DAY$(2),HOUR$(2),MINUTE$(2
    ),SECOND$(2),FRAC$(3),YEAR$(4): REM  DIMENSION STRING
    S
 20 SLOT=4: REM  SET SLOT NUMBER
 25 YEAR$="1978": REM  SET THE YEAR
 30 CALL -936: REM  CLEAR THE SCREEN
 40 REM
 50 REM  READ THE TIME
 60 REM
 80 IN#SLOT: REM   SET INPUT TO CLOCK BOARD
 90 PR#SLOT: REM   SET OUTPUT TO CLOCK BOARD
100 VTAB 23: REM   PUT CURSOR AT BOTTOM OF SCREEN
110 INPUT " ",T$: REM   OBTAIN THE TIME
120 IN#0: REM   RESTORE INPUT TO KEYBOARD
130 PR#0: REM   RESTORE OUTPUT TO CRT
135 REM
140 REM  OBTAIN MONTH, DAY, HOUR,...ECT
150 REM
160 MONTH$=T$(1,2)
170 DAY$=T$(4,5)
180 HOUR$=T$(7,8)
190 MINUTE$=T$(10,11)
200 SECOND$=T$(13,14)
210 FRAC$=T$(16,18)
220 REM
230 REM  OBTAIN MONTH (JANUARY, FEBRUARY...)
240 REM
250 IF MONTH$="01" THEN MONTH$="JANUARY"
260 IF MONTH$="02" THEN MONTH$="FEBRUARY"
270 IF MONTH$="03" THEN MONTH$="MARCH"
280 IF MONTH$="04" THEN MONTH$="APRIL"
290 IF MONTH$="05" THEN MONTH$="MAY"
300 IF MONTH$="06" THEN MONTH$="JUNE"
310 IF MONTH$="07" THEN MONTH$="JULY"
320 IF MONTH$="08" THEN MONTH$="AUGUST"
330 IF MONTH$="09" THEN MONTH$="SEPTEMBER"
340 IF MONTH$="10" THEN MONTH$="OCTOBER"
350 IF MONTH$="11" THEN MONTH$="NOVEMBER"
360 IF MONTH$="12" THEN MONTH$="DECEMBER"
370 REM
380 REM  PRINT DATE AND TIME ON SCREEN
390 REM
400 VTAB 10: TAB 10: REM    CENTER OUTPUT
410 PRINT "DATE: ";MONTH$;" ";DAY$;", ";YEAR$
420 VTAB 12: TAB 10: REM  CENTER OUTPUT
430 PRINT "TIME: ";HOUR$;":";MINUTE$;":";SECOND$;"."
    ;FRAC$
440 GOTO 80: REM   READ TIME AGAIN
```

MSEC

MO |1,2| DAY |4/5| HR |7/8| MIN |10,11| |16,17,18|

FORMAT = MM/DD_HH:MM:SS.FFF

|13,14|

SEC

```
0  REM ******  DATE AND TIME  ******
4  REM *** APPLESOFT WITHOUT DOS ***
10  REM
20 SLOT = 4: REM  SET SLOT#
30 YEAR$ = "1979": REM  SET YEAR
40  HOME : REM  CLEAR SCREEN
50  REM
60  REM  READ THE TIME
70  REM
80  IN# SLOT: REM  SET INPUT TO CLOCK
90  PR# SLOT: REM  SET OUTPUT TO CLOCK
100  VTAB 23: REM  PUT CURSOR AT BOTTOM OF SCREEN
110  INPUT " ";T$: REM  OBTAIN TIME
120  IN# 0: REM  RESTORE INPUT TO KEYBOARD
130  PR# 0: REM  RESTORE OUTPUT TO CRT
200  REM
210  REM  OBTAIN MONTH, DAY, HOUR,...ECT
220  REM
230 MTH$ =  LEFT$ (T$,2)
240 DAY$ =  MID$ (T$,4,2)
250 HOUR$ =  MID$ (T$,7,2)
260 MINUTE$ =  MID$ (T$,10,2)
270 SEC$ =  MID$ (T$,13,2)
280 FRAC$ =  RIGHT$ (T$,3)
300  REM
310  REM   OBTAIN MONTH (JANUARY, FEBRUARY...)
320  REM
330 MTH =  VAL (MTH$): REM  FIND DECIMAL # FOR MONTH
340  RESTORE : REM  INITIALIZE DATA
350  FOR I = 1 TO MTH
360  READ MTH$: REM  FIND NAME OF MONTH
370  NEXT I
380  DATA  "JANUARY","FEBRUARY","MARCH","APRIL","MAY","JUNE"
390  DATA  "JULY","AUGUST","SEPTEMBER","OCTOBER","NOVEMBER","DECEMBER"
400  REM
410  REM  OUTPUT DATE AND TIME
420  REM
430  VTAB 10: HTAB 10: REM  CENTER OUTPUT
440  PRINT "DATE: ";MTH$;" ";DAY$;", ";YEAR$
450  VTAB 12: HTAB 10: REM  CENTER OUTPUT
460  PRINT "TIME: ";HOUR$;":";MINUTE$;":";SEC$;".";FRAC$
470  GOTO 80: REM  READ TIME AGAIN
```

Obtaining the name of the month in Applesoft is much easier since it is possible to change a string to its decimal value (Line 33Ø). Then data can be read until the correct month is found (Lines 34Ø through 39Ø). The time and date may then be printed in the desired format.

To print the time as AM or PM, add these lines to the Applesoft program.

```
290 HOUR =  VAL (HOUR$): REM CHANGE HOUR$ TO DECIMAL
291 HR = HOUR
292  IF HR = 0 THEN HOUR = 12
293  IF HR > 12 THEN HOUR = HR - 12
294 AMPM$ = "AM"
295  IF HR > 11 THEN AMPM$ = "PM"
460  PRINT "TIME: ";HOUR;":";MINUTE$;":";SEC$;".";FRAC$
     ;"  ";AMPM$;"     "
```

Table 5

APPLESOFT AM/PM

To print the time as AM or PM, add or change these lines in the Integer BASIC program.

```
 15 DIM HR10$(2),HR1$(2),AMPM$(2)
362 HR10$=HOUR$(1)
363 HR1$=HOUR$(2)
365 HR10= ASC(HR10$)-176
366 HR1= ASC(HR1$)-176
367 HOUR=(10*HR10)+HR1
368 HR=HOUR
369 AMPM$="AM"
370 IF HR=0 THEN HOUR=12
371 IF HR>12 THEN HOUR=HR-12
372 IF HR>11 THEN AMPM$="PM"
430 PRINT "TIME: ";HOUR;":";MINUTE$;":";SECOND$;".";
    FRAC$;"  ";AMPM$;"     "
```

Table 6

INTEGER AM/PM

If the previous programs are going to be run with DOS, change or add these lines.

```
 15 D$="": REM   D$="CONTROL D"
 16 PRINT D$;"NOMONI,O,C": REM   PREVENT DISK
     COMMAND FROM PRINTING ON SCREEN
 80 PRINT D$;"IN#";SLOT: REM   SET INPUT TO C
    LOCK BOARD
 90 PRINT D$;"PR#";SLOT: REM   SET OUTPUT TO
    CLOCK BOARD
110 INPUT " ",T$: REM    OBTAIN THE TIME
120 PRINT D$;"IN#0": REM    RESTORE INPUT TO
    KEYBOARD
130 PRINT D$;"PR#0": REM   RESTORE OUTPUT TO
    CRT
```

Table 7
PROGRAM CHANGES WITH DOS

YEAR

The Apple Clock does not keep track of the year. In order to print the year along with the time, a variable YEAR must be set up. The Integer Date and Time Program on page 17 of the manual demonstrates this. In this case, YEAR$ is a string. Line 25 sets the year to 1979. This line can be changed accordingly. On page 18, line 30 of the Applesoft Date and Time Program sets the year.

YEAR$ does not need to be a string in this case because its value is numeric (1979). However, if the year were to be printed as "Nineteen-Hundred-Seventy-Nine", a string must be used and dimensioned accordingly.

ELAPSED TIME

Using the Apple Clock, simple programs may be written to measure the elapsed time between two events.

At the initial event, read the clock and save as a string, maybe T1$. At the second event, read the time as T2$. Then, using the subroutine in the following program located at location 3ØØØ, the total seconds to date (STD) since January 1, can be found for each of the two times. If they are subtracted, the elapsed time in seconds is easily found. These seconds can be changed to Days, Hours, Minutes and Seconds by the subroutine at Location 4ØØØ.

The following program measures the time between two carriage returns typed.

```
0  REM ****  ELAPSED TIMER PROGRAM  ****
2  REM ****  APPLESOFT WITH DOS  ****
20  REM  *** SUBROUTINES MAY BE USED IN YOUR PROGRAMS
25  REM
30 D$ = "": REM  D$=" CONTROL D"
40  PRINT D$;"NOMON I,O,C": REM  DON'T PRINT DISK COMM
   ANDS
50  HOME
60 SLOT = 4: REM  SET CLOCK BOARD SLOT#
65  REM
70  REM  IF L=0 NOT A LEAP YEAR, IF L=1 IT IS A LEAP Y
   EAR
80 L =  PEEK (49280 + 16 * SLOT)
90 L =  INT (L / 64)
95  IF L > 1 THEN L = L - 2
96  REM
100  REM    T1$=THE INITIAL START TIME
110  REM    T2$=THE TIME AT A LATER TIME
120  INPUT "HIT RETURN TO START TIMER ";A$
125  PRINT
130  GOSUB 2000: REM  GET THE TIME
140 T1$ = T$: REM  T1$=INITIAL START TIME
150  INPUT "HIT RETURN AT DESIRED TEST TIME ";A$
155  PRINT
160  GOSUB 2000: REM  GET THE TIME NOW
170 T2$ = T$: REM  T2$=THE TEST TIME
200  REM
210  REM  FIND STD FOR T1$
220 T$ = T1$: GOSUB 3000:S1 = STD
230  REM  FIND STD FOR T2$
240 T$ = T2$: GOSUB 3000:S2 = STD
250  REM  FIND ELAPSED TIME  ET=S2-S1
260 ET = S2 - S1
290  VTAB 10
300  PRINT "THE ELAPSED TIME HAS BEEN "
310  PRINT ET;" SECONDS"
```

```
500  REM  CONVERT TO DAYS, HOURS MINUTES, SEDONDS
510  GOSUB 4000: REM  SUBR TO CALC THIS
515  VTAB 16
520  PRINT "DAYS=";D
530  PRINT "HOURS=";H
540  PRINT "MINUTES=";M
550  PRINT "SECONDS=";S
600  END
2000  REM
2005  REM  *** SUBR - GET THE TIME
2010  REM *** THESE NEED TO BE CHANGED IF DISK IS NOT
     USED
2030  PRINT D$;"IN#";SLOT
2040  PRINT D$;"PR#";SLOT
2050  INPUT " ";T$
2060  PRINT D$;"IN#0"
2070  PRINT D$;"PR#0"
2080  RETURN
3000  REM
3005  REM  SUBR - STD
3006  REM
3010  REM  CALCULATE SECONDS TO DATE FOR EACH TIME (ST
     D)
3020  REM  THIS IS THE NUMBER OF SECONDS SINCE JANUARY
      1
3030  REM  DO THIS FOR STRING TIME T$
3040  REM    RETURN A NUMBER - STD
3050  REM
3060  REM  FIND #'S FOR DATE AND TIME
3070 MT =  VAL ( MID$ (T$,1,2))
3080 D =  VAL ( MID$ (T$,4,2))
3090 H =  VAL ( MID$ (T$,7,2))
3100 M =  VAL ( MID$ (T$,10,2))
3110 S =  VAL ( MID$ (T$,13,6))
3130  REM  CALCULATE DAYS TO DATE - DTD
3135  RESTORE
3140 DTD = 0
3150  FOR I = 1 TO MT
3160  READ J
3170 DTD = DTD + J
3180  NEXT I
3200  DATA  0,31,28,31,30,31,30,31,31,30,31,30,31
3205  REM   ADD IN DAYS AND LEAP YEAR DAY
3210 DTD = DTD + D
3230  IF MT > 2 AND L = 1 THEN DTD = DTD + 1
3240  REM  FIND SECONDS TO DATE - STD
3250 STD = DTD * 86400 + H * 3600 + M * 60 + S
3300  RETURN
4000  REM
4010  REM  SUBR - PUT SECONDS INTO DAYS, HOURS, MINUTE
     S, SECONDS
4020  REM  GIVEN ET IN SECONDS
4040 D =  INT (ET / 86400)
4050 ET = ET - D * 86400
4060 H =  INT (ET / 3600)
4070 ET = ET - H * 3600
4080 M =  INT (ET / 60)
4090 S = ET - M * 60
4100  RETURN
```

INTERVAL TIMER

In some applications it may be necessary to perform a task at a particular time.

The method to do this is as follows:

1. Obtain the current time from the clock.
2. Convert to seconds to date (STD) using the subroutine in the previous program.
3. Add to it the desired wait time in seconds, and save this time.
4. At various points in your program, check the time and find the current STD.
5. If the current STD is equal or greater than the desired time, it's time to perform the desired task.

THEORY OF OPERATION

THE HARDWARE

The Apple Clock hardware design is composed of four main sections: The clock counters, the PROM circuitry, the supply regulator, and the interrupt hardware.

Clock Counters

The counters are placed across the top of the PC board. A 1 MHz crystal controls the frequency. Three dual BCD up counters (U1, U2, U3) are used to divide the frequency down to obtain the units of time less than a second. Therefore, 1's 1Ø's and 1ØØ's of milliseconds are available in BCD format.

Two 12-bit binary counters (U4, U5) provide the digits of 2^0 to 2^{23} seconds. A D-flipflop (U10) adds the last time digit of 2^{24} seconds.

A decoder (U13) is used to determine which digit is being read. It controls the enable lines on the Tri-State buffers (U14-U21).

The clock is set by first stopping the clock (reading from CØ8Ø+N6). Flipflop, U10, keeps the clock stopped until a START clock command is issued. Once the clock is stopped, the ADVANCE 1 command advances the first 12-bit binary counter. An ADVANCE 2 command advances the counters U4 and U10. Digits below a second cannot be set. They are automatically reset to zero when the clock is stopped. It is impossible to advance the clock when the clock is running. A 'WRITE PROTECT' switch on Pin 9 of U1Ø prevents the clock from being accidentally changed.

ROM Access

An on-board ROM, U27, provides easily accessible software for the user. The flipflop made of U8 and U23 is set by a READ from CNØØ-CNFF. This causes Pin 11 of U8 to go high and stay there until the flipflop (U8, U23) is reset. The ROM can then be read from C8ØØ-CF7F. The ROM is shut off when a $CFFF is addressed (U23), or when 'RESET' is hit.

The output buffers (U25, U26) are enabled by U6 which is dependent upon DEVICE SELECT, I/O SELECT, R/W or PROM ENABLE.

Regulator Circuit

There are two 5 volt supplies for the Apple Clock. One is derived from the 5 volts on the Apple bus. It is used to supply power to all the TTL circuits on the board. The other supply is derived from the +12 volts on the bus and regulated to +5 volts by U28. This regulator supplies the power for all the CMOS circuitry. When the Apple is turned off, the on-board battery supplies the CMOS circuitry, and keeps the clock running.

When the Apple is turned on, the battery is trickle-charged through R12. For faster charging time, R12 may be reduced at the expense of a shorter battery lifetime.

Interrupts

The Apple Clock is capable of generating interrupts on a regular basis. This means that the computer can be performing one task, and be interrupted to perform another task, then return back to the original program.

Interrupts may be enabled by writing a '1' to the set interrupt address, and disabled by writing an 'Ø' to the same address. Hitting 'RESET' will also disable interrupts.

The clock board will interrupt on one-second intervals when enabled. This one-second period is determined by which counter output drives Pin 11 of U11. The clock is factory set for 1-second intervals.

The procedure for handling interrupts is as follows:

1. Enable interrupts by writing a '1' to the SET INTERRUPT device select address.

2. Once a second, an interrupt will occur that will lower the IRQ line on the Apple bus if a higher priority peripheral (in a lower slot number) is not currently interrupting. The 'INT IN' line tells the clock board if a higher priority board is interrupting.

3. Along with the IRQ line going low, the INT OUT line will go low to tell lower priority boards that the clock is interrupting. This prevents them from interrupting.

4. If a higher priority board is interrupting, the clock will wait till it is done, and then the clock will interrupt.

5. If interrupts are enabled in software (CLI instruction), the Apple will perform a jump to an address contained in memory locations 3FE (low) and 3FF (high).

6. An interrupt automatically disables interrupts so that other interrupts may not occur immediately. Two forms of interrupt acknowledges must be performed to the Apple Clock to clear interrupts. One is 'CLEAR IRQ' which clears the IRQ line but not the 'INT-OUT' line. This procedure may be done to allow higher priority peripheral boards to interrupt, but not lower priority boards. To allow higher priority boards to interrupt, do a 'CLEAR IRQ' command early in the interrupt routine and do a 'CLEAR INT-OUT' command at the end of the interrupt routine to allow lower priority boards to interrupt. Before leaving the interrupt routine, both CLEARS should be performed so that other interrupting boards are not tied up and may perform their own interrupts.

7. **To prevent the Apple Clock from interrupting, a 'Ø' may be written to the Set Interrupt address, or 'RESET' may be pressed.**

THE SOFTWARE

The Apple II peripheral bus is memory mapped. Therefore, in order to talk to a particular device you must address its DEVICE SELECT address. The following table shows the relationship between the slot # the clock is in and the DEVICE SELECT address.

SLOT #	DEVICE SELECT HEX	ADDRESS DECIMAL
Ø	CØ8Ø-CØ8F	(-16256)-(-16241)
1	CØ9Ø-CØ9F	(-16240)-(-16225)
2	CØAØ-CØAF	(-16224)-(-16209)
3	CØBØ-CØBF	(-16208)-(-16193)
4	CØCØ-CØCF	(-16192)-(-16177)
5	CØDØ-CØDF	(-16176)-(-16161)
6	CØEØ-CØEF	(-16160)-(-16145)
7	CØFØ-CØFF	(-16144)-(-16129)

Table 8

DEVICE SELECT ADDRESSING

The following formula may also be used:

```
DEVICE SELECT ADDRESS = $CØ8Ø+$NX
                      = -16256+(16*N)+X
N = SLOT #
```

X may have any value from $Ø to $F(0-15). Please take note that a dollar sign ($) before a number means the number is in HEX. The value of X determines the action the clock will take. Following is a table which shows the command for the clock.

X	COMMAND
Ø	READ $2^{2Ø}-2^{24}$TIME BITS
1	READ $2^{12}-2^{19}$TIME BITS
2	READ $2^{4}-2^{11}$TIME BITS
3	READ 1ØØms-2^{3} TIME BITS
4	READ 1msecs-1Ømsecs
5	START CLOCK
6	STOP CLOCK
7	ADVANCE ONE OR CLEAR IRQ
8	ADVANCE TWO OR CLEAR INT-OUT
9	SET INTERRUPT

Table 9

CLOCK COMMANDS

Suppose your Apple Clock is in Slot #4. The following table lists the addresses and commands. All commands except SET INTERRUPT should be done with a PEEK when in BASIC, or a LOAD instruction from Assembly language.

ADDRESS		
HEX	DECIMAL	COMMAND
CØCØ	-16192	READS $2^{2Ø}$-2^{24} TIME BITS
CØC1	-16291	READS 2^{12}-2^{19} TIME BITS
CØC2	-1619Ø	READS 2^{4} -2^{11} TIME BITS
CØC3	-16189	READS 1ØØms-2^{3} TIME BITS
CØC4	-16188	READS 1ms-1Ømsec TIME BITS
CØC5	-16187	START CLOCK
CØC6	-16186	STOP CLOCK
CØC7	-16185	ADVANCE ONE OR CLEAR IRQ
CØC8	-16184	ADVANCE TWO OR CLEAR INT-OUT
CØC9	-16183	SET INTERRUPT

Table 10

SLOT #4 EXAMPLE

To start the clock simply do a

```
          PEEK(-16187)   in   BASIC

  or      LDA $CØC5   in Assembly Language
```

In order to stop the clock:

```
          PEEK(-16186)   in   BASIC

  or      LDA $CØC6   in Assembly Language
```

READING DIGITS OF TIME

There are five digits of time ranging from 1's of milliseconds to 2^{24} seconds. The digits less than a second are in BCD format. Digits of seconds and above are in binary.

To read a digit of time either PEEK a location when in BASIC or do a LOAD instruction from that address in assembly language.

When reading the top byte of time, only the bottom 5 bits of this byte are used for time. This table shows the bit configuration for the top byte (the lowest device select address).

X = Ø

	BIT	MEANING
MSB	7	(Ø) Clock Board Interrupting (1) Not Interrupting
	6	(1) Leap Year (Ø) Not Leap Year
	5	Not Used
	4	2^{24} Seconds
	3	2^{23} Seconds
	2	2^{22} Seconds
	1	2^{21} Seconds
LSB	Ø	$2^{2Ø}$ Seconds

Table 11
TOP BYTE OF TIME CONFIGURATION

ROM FIRMWARE

The Apple Clock has an on-board ROM that allows easy access to the time. It may be accessed from BASIC or assembly language, and provides date and time information.

When in BASIC, the input routine switches may be set using the IN#n command. Following this by an INPUT statement, the ROM will be activated and send back the time to BASIC.

To stop the printout of the time from an INPUT statement, the printout switch may be set to the clock board by using the PR#n command. After reading the time, both switches should be set back to Ø for normal Apple operation (PR#Ø:IN#Ø).

In order to access the time using the ROM from a machine language program you must put $#CN into location KSWH ($39) to make the ROM software work (where N is the slot # the clock is in). Then a JSR to CNØØ will put the time into location $28Ø and up as shown in Table 12.

ADDRESS	USE
Ø28Ø	Not Used
Ø281	Carriage Return
Ø282	1's milliseconds
Ø283	1Ø's milliseconds
Ø284	1ØØ's milliseconds
Ø285	.
Ø286	1's seconds
Ø287	1Ø's seconds
Ø288	;
Ø289	1's minutes
Ø28A	1Ø's minutes
Ø28B	;
Ø28C	1's hours
Ø28D	1Ø's hours
Ø28E	Space
Ø28F	1's days
Ø29Ø	1Ø's days
Ø291	/
Ø292	1's months
Ø293	1Ø's months
Ø294	Space
Ø2AØ	Counter
02A1	Temporary Storage

Table 12

ROM MEMORY USE

CHANGING ROMS

Your Apple Clock is shipped with a ROM which is a 2708 equivalent. At some later date you may wish to insert a 2716 PROM. This can easily be done by adding a few jumpers. On the PC board above the ROM notice there are numbers: 13, 11, 12, 8, 9 and 1Ø. Follow these instructions for a 2716 PROM.

Connect 8 to 9
Connect 11 to 12

2716 PROM
OR
ROM EQUIVALENT

Connect 8 to 10
Connect 11 to 13

2708 PROM
OR
ROM EQUIVALENT

INTERRUPTS

One of the main features of the Apple Clock is the ability to have interrupts occur at set intervals. Interrupts can add new dimensions to your computer. For instance, background and foreground programming is possible by letting the interrupt handler routine initiate the background program. Also, data can be sampled at precise intervals. A program example is presented here to demonstrate the use of interrupts and the writing of an interrupt handler program.

DISPLAY THE TIME: AN EXAMPLE

The following program demonstrates the use of interrupts and the Apple Clock. The program displays the date and time on the screen. Every time an interrupt occurs, the time is updated. Meanwhile the computer can be used as usual except:

```
*****************************************************
* Interrupts should not be used with the disk       *
* operating systems versions 3.1 or earlier.  DOS   *
* is not protected against interrupts (i.e.,        *
* interrupts are enabled during disk operation).    *
* An interrupt during a disk transfer could result  *
* in the destruction of information already on the  *
* disk.                                             *
*****************************************************
```

The program assumes the clock is in slot #4.

PROGRAM EXPLANATION

Interrupts must be initialized both in the computer and on the clock board. From BASIC a CALL 822 will initialize interrupts by jumping to the assembly initialization program shown here. This initialization program resides at $Ø336, and must be loaded in every time you power up or boot the disk. Line 60 of the assembly program saves the A register since it will be used. Line 61 disables interrupts. This is done to prevent an interrupt from occuring right after we enable the clock interrupts, but are still in the initialization program.

When an interrupt occurs, control jumps to the location pointed to by $Ø3FE (low byte) and $Ø3FF (high byte). Lines 62 through 65 set this to the interrupt handler starting address. The clock interrupts are enabled in 66 and 67. Line 68 clears the screen and homes the cursor.

```
:ASM
                *
                *         DISPLAY THE TIME
                *        INTERRUPT  PROGRAM
                *
                *              BY
                *           SHERI TALBOTT
                *
                *     MOUNTAIN HARDWARE, INC.
                *
                ***  ROUTINES ASSUMES THE CLOCK IS IN SLOT 4 ***
                *
                          OBJ   $300
                          ORG   $300
                TIME      EQU   $0286
                SCR       EQU   $766
                *
                ****DISPLAY THE TIME****
                ****INTERRUPT ROUTINE****
                *
                *THIS ROUTINE IS EXECUTED WHEN AN INTERRUPT
                *OCCURS AND IT PRINT THE TIME ON THE BOTTOM OF THE SCREEN
                *
0300 8A                   TXA   ;SAVE X REGISTER
0301 48                   PHA
0302 98                   TYA   ;SAVE Y REGISTER
0303 48                   PHA
0304 AD C7 C0             LDA   $C0C7 ;CLEAR IRQ ON CLOCK
0307 58                   CLI   ;ENABLE HIGHER PRIORITY INTERRUPTS
0308 A5 39                LDA   $39 ;SAVE $39
030A 48                   PHA
030B A9 C4                LDA   #$C4 ;SET UP FOR CLOCK (IN SLOT 4)
030D 85 39                STA   $39
030F 20 00 C4             JSR   $C400 ;READ CLOCK
0312 68                   PLA   ;RESTORE $39
0313 85 39                STA   $39
0315 A9 BA                LDA   #$BA ;PUT A COLON IN TIME INSTEAD OF SEIMICOLON
0317 8D 88 02             STA   $0288
031A 8D 8B 02             STA   $028B
031D A2 00                LDX   #$0 ;SET UP SCREEN POSITION COUNTER
031F A0 0E                LDY   #$0E ;SET POINTER TO OBTAIN TIME DIGITS
0321 B9 86 02   OUT       LDA   TIME,Y ;GET TIME DIGIT
0324 9D 66 07             STA   SCR,X ;STORE TO SCREEN
0327 E8                   INX
0328 88                   DEY
0329 10 F6                BPL   OUT  ;DONE ALL CHARACTERS?
032B 68                   PLA   ;RESTORE Y
032C A8                   TAY
032D 68                   PLA   ;RESTORE X
032E AA                   TAX
032F 78                   SEI   ;DISABLE INTERRUPTS
0330 AD C8 C0             LDA   $C0C8 ;CLEAR INT-OUT ON CLOCK BOARD
0333 A5 45                LDA   $45 ;RESTORE A
0335 40                   RTI
```

```
                    54 *
                    55 ****DISPLAY ON*****
56 *
57 *THIS ROUTINE INITIALIZES THE INTERRUPTS
                    58 *AND SET UP THE SCREEN WINDOW
                   59 *
336 48             60        PHA
0337 78             61        SEI
0338 A9 00          62        LDA   #$00 ;SET UP INT.  ADDR
033A 8D FE 03       63        STA   $3FE
033D A9 03          64        LDA   #$03
033F 8D FF 03       65        STA   $3FF
0342 A9 01          66        LDA   #$01 ;TURN ON CLOCK BOARD INTERRUPTS
0344 8D C9 C0       67        STA   $C0C9 ;(SET FOR SLOT 4)
0347 20 58 FC       68        JSR   $FC58 ;HOME CORSOR AND CLEAR SCREEN
034A A9 16          69        LDA   #$16 ;SET BOTTOM LINE FOR SCROLLING
034C 85 23          70        STA   $23
034E 58             71        CLI
034F 68             72        PLA   ;RESTORE A
0350 60             73        RTS
                    74 *
                    75 ****DISPLAY OFF****
                    76 *
                    77 *THIS ROUTINE TURNS OFF INTERRUPTS
                    78 *AND RESTORES THE SCREEN
                    79 *
0351 78             80        SEI
0352 48             81        PHA
0353 A9 00          82        LDA   #$0 ;TURN OFF CLOCK INTERRUPTS
0355 8D C9 C0       83        STA   $C0C9 ;(SET FOR SLOT 4)
0358 AD C7 C0       84        LDA   $C0C7 ;CLR IRQ (SET FOR SLOT 4)
035B AD C8 C0       85        LDA   $C0C8 ;CLR INT-OUT (SET FOR SLOT 4)
035E A9 18          86        LDA   #$18 ;RESTORE TO BOTTOM LINE
0360 85 23          87        STA   $23
0362 58             88        CLI
0363 68             89        PLA
0364 60             90        RTS
                    91 *
                    92 ****INT. DISABLE****
                    93 *
                    94 *THIS ROUTINE DISABLES INTERRUPTS
                    95 *
0365 78             96        SEI   ;DISABLE INTERRUPTS
0366 60             97        RTS
                    98 *
                    99 ****INT. ENABLE****
                   100 *
                   101 *THIS ROUTINE ENABLES INTERRUPTS
                   102 *
0367 58            103        CLI   ;ENABLE INTERRUPTS
0368 60            104        RTS
--- END ASSEMBLY ---
TOTAL ERRORS: 00
```

The bottom edge of the scrolling window is set in 69 and 70. This prevents the displayed time from being overwritten. Interrupts are enabled in 71 and the A register is restored in 72 before returning to the BASIC program.

WHEN AN INTERRUPT OCCURS

When an interrupt occurs, the computer jumps to the location pointed to by $Ø3FE and $Ø3FF. In this case it jumps to $Ø3ØØ where the interrupt routine resides.

First, all registers and memory which are used jointly by the main program and the interrupt routine, must be saved. They are restored before returning so that the system can continue where it left off before the interrupt. Lines 23 through 26 , 29 and 30 save the used registers and memory.

Line 27, clears the interrupt request line (IRQ) on the clock. This allows higher priority boards to interrupt the clock. The priority is determined by the slot number. Slot Ø has the highest priority, while slot 7 has the lowest priority. Line 28 enables interrupts.

Address $39 is saved in lines 29 and 30 because it needs to be changed in order to use the clock PROM to read the time. The clock PROM looks at location $39 to determine what command you are sending it. A $CN (N = clock slot number) must be in $39 before doing a JSR $CNØØ. Therefore, lines 31-33 read the clock and store the time in the locations shown on page 29 of the manual. Location $39 is restored in 34 and 35.

The time which is stored in locations $Ø28Ø to $Ø294, contains semi-colons instead of colons between the hour and minutes and seconds. This allows the clock to be read in Applesoft. Applesoft interprets a colon in a string to mean the end of the string thus the use of semicolons. Lines 36, 37 and 38 put colons where the semicolons are for a better visual appearance.

Line 39 sets the screen position counter while 40 sets a pointer for obtaining the time digits. In 41 the time digit is loaded, and stored to the screen in line 42. The counters are incremented and decremented respectively. When all digits are written to the screen (line 45), we are ready to return to the main program.

Lines 45 through 49 restore the X and Y registers.

Interrupts are disabled in 50 until we leave this interrupt routine. Line 51 clears the INT-out line on the clock. By clearing this, boards of lower priority than the clock may now interrupt after leaving the interrupt routine.

Therefore, there are 2 interrupts control lines on the clock which must be acknowledged. A CLEAR IRQ allows higher priority boards to interrupt. A CLEAR INT-OUT, allows lower priority boards to interrupt.

When an interrupt occurs, the Apple monitor saves the A register in location $45 before jumping to the interrupt routine. Therefore, the last step is to load the A register with location $45 before returning from the interrupt (line 52). An RTI enables interrupts.

Lines 91 through 104 were added to allow easy disabling and enabling of interrupts respectively. A BASIC call to the correct address will easily disable or enable interrupts.

CHANGING SLOTS

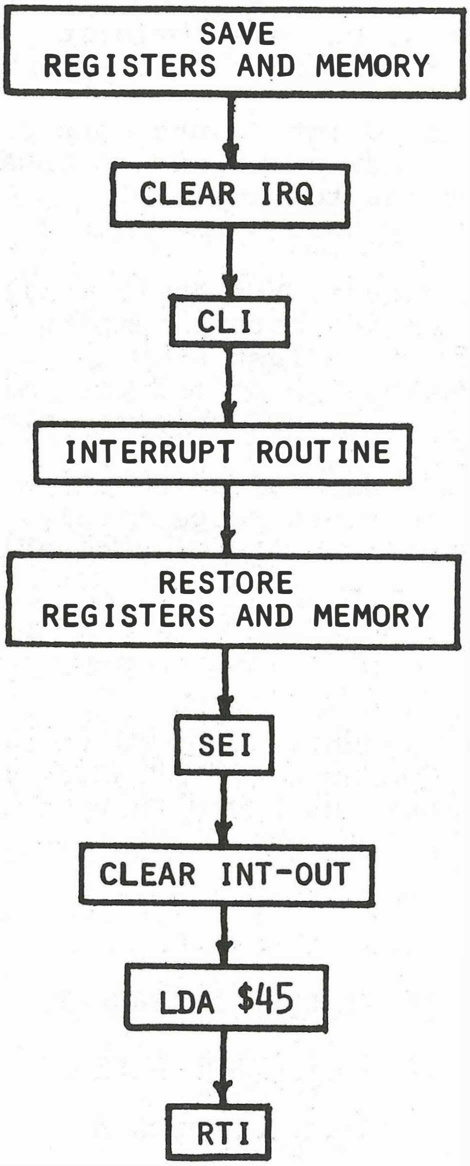
The previous program can be changed to use the clock in another slot. Change lines 27,31,33,51,67,83,84, and 85, according to instructions given on pages 26 through 28 of the clock manual.

BASIC CALLS

CALL 822	INITIALIZE INTERRUPTS
CALL 849	TURN OFF INTERRUPTS
CALL 869	DISABLE INTERRUPTS
CALL 871	ENABLE INTERRUPTS

These CALL's only work when the Display the Time program is in memory.

IN GENERAL



The above flow chart demonstrates the structure an interrupt routine should have.

CHANGING THE INTERRUPT FREQUENCY

The Apple Clock interrupt frequency is factory set for 1 second. It is possible to change this using an Xacto knife, wire and soldering iron.

First disconnect the battery and/or adapter from the clock.

In the upper left hand corner of the board above U11 and U12, there are two numbers 6 and 7. A trace connects the two points. Using an Xacto knife, cut the trace between 6 and 7. Be sure the trace is completely cut. If you have an Ohmmeter check the resistance between the two points to guarantee that they are disconnected.

Now using the table on page 38, connect point 7 to the appropriate IC to obtain the desired interrupt frequency.

The following table lists conversions which are useful.

1 Hour	=	3600 seconds
1 Day	=	86400 seconds
1 Week	=	604800 seconds
28 Days	=	2419200 seconds
30 Days	=	259200 seconds
31 Days	=	2678400 seconds

Interrupt Frequency	IC#	PIN #
.1mSec	U1	9
1mSec	U2	9
1ØmSec	U3	2
100mSec	U3	11
1 "	U3	14
2 "	U4	9
4 "	U4	7
8 "	U4	6
16 "	U4	5
32 "	U4	3
64 "	u4	2
128 "	U4	4
256 "	U4	13
512 "	U4	12
1Ø24 "	U4	14
2Ø48 "	U4	15
4Ø96 "	U4	1
8192 "	U5	9
16384 "	U5	7
32768 "	U5	6
65536 "	U5	5
131Ø72 "	U5	3
262144 "	U5	2
524288 "	U5	4
1Ø48576 "	U5	13
2Ø97152 "	U5	12
4194304 "	U5	14
8388608 "	U5	15
16777216 "	U5	1
33554432 "	U10	1

INTERRUPT FREQUENCY CONNECTION

SETTING THE FREQUENCY

Your Apple Clock has been factory assembled, burned in, and tested. The 1.0000MHz time base has been accurately set to within .001%. Vibrations or extreme temperatures can cause slight changes to the time base and may produce noticeable errors. If these errors are noticed, or if you desire to set this frequency more precisely for your environment, an accurate frequency counter and a small non-metallic screwdriver are required.

Connect the frequency counter with the ground lead to Pin 7 of U9, the positive lead to Pin 6 of U9. Adjust C1 for a frequency as close to 1.0000MHz as possible. Be sure the clock is at the same operating temperature as its normal environment.

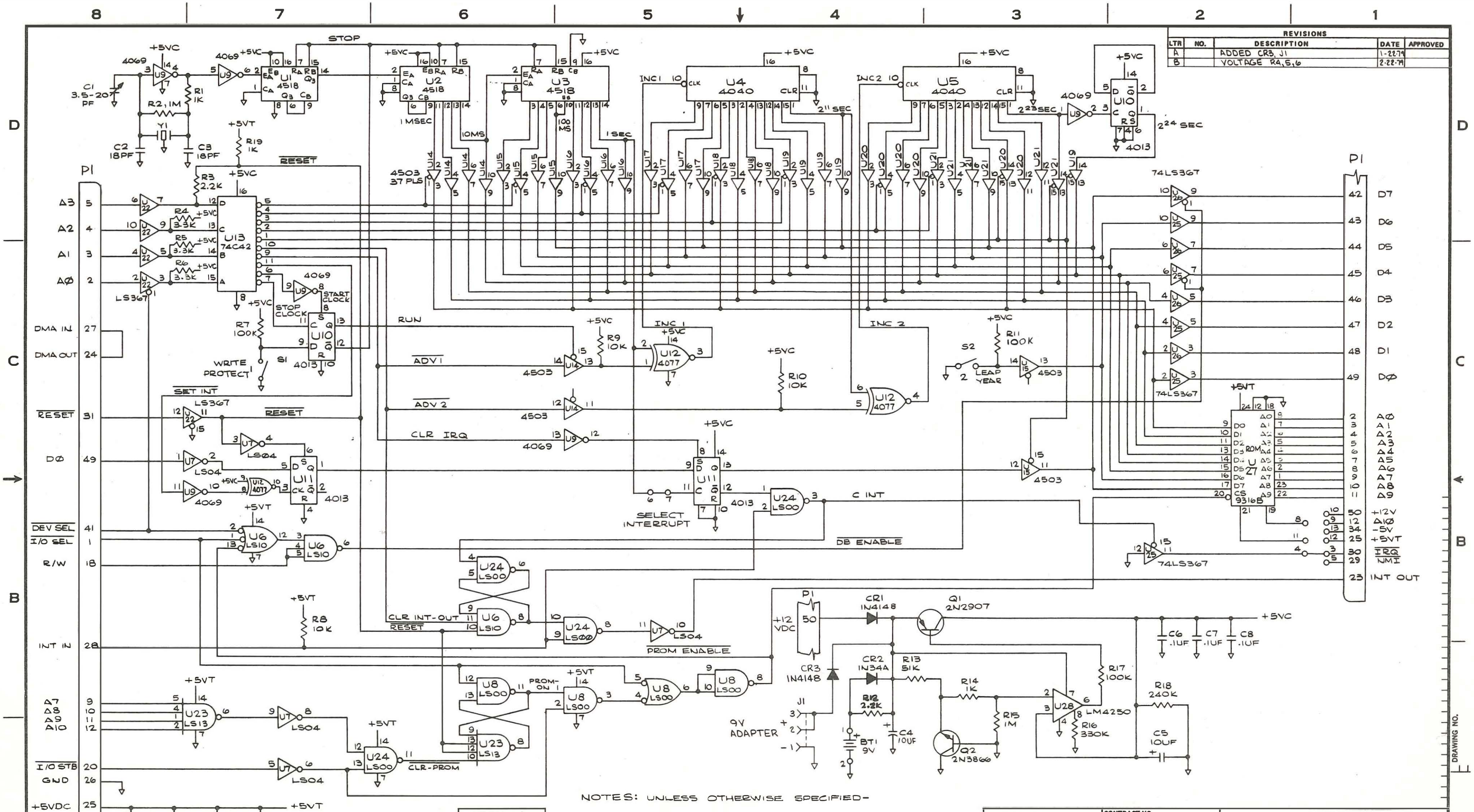
9 VOLT ADAPTER

A 9 volt adapter may be plugged into some Apple Clocks. Any standard 9 volt DC adapter with a microplug on it may be used. Most computer or electronic parts stores carry them. They are also available from Mountain Hardware.

After purchasing such an adapter, simply plug it into the jack on the bottom left side of the board. The adapter will keep the clock running indefinitely even though your computer is turned off. Leave the battery attached in case the power in your building fails. The adapter continuously charges the battery.

Caution! Be sure the adapter provides 9 volts DC minimum. Many adapters are sold as universal calculator adapters and may not provide a full 9 volts required by the clock.

REVISIONS				
LTR	NO.	DESCRIPTION	DATE	APPROVED
A		ADDED CR3, J1	1-22-79	
B		VOLTAGE R4,5,6	2-22-79	



NOTES: UNLESS OTHERWISE SPECIFIED-

1. ALL RESISTOR VALUES ARE IN OHMS, 1/4W, 5%.
2. U14 - U21 (4503) - PIN 16 = + 5VC, PIN 8=GND. U22, U25, U26 (74LS367) - PIN 16 = +5VT, PIN 8 = GND.

REF DES	
LAST USED	NOT USED
BT1 C12 CR3 P1 Q2 R19 S1 U28 Y1 13 J1	

14-000070	
NEXT ASSY	USED ON
APPLICATION	

UNLESS OTHERWISE SPECIFIED	CONTRACT NO.		Mountain Hardware, Inc.
XX DEC ± 010	DO NOT SCALE DRAWING		
XXX DEC ± .005			
FRACTIONS ± 1/64	DR LORETTA WILLE	09 28 78	SCHEMATIC DIAGRAM - APPLE CLOCK © 1978
ANGLES ± 0° 30′	CHK		
MATERIAL	PROJ ENGR	10-1-78	
TREATMENT			SIZE D / CODE IDENT NO. / DRAWING NO. 20-000071
FINISH RMS	MFG ENGR		SCALE NONE / WT / SHEET 1 OF 1

WARRANTY

Your factory-built Apple Clock is warranted against defects in materials and workmanship for a period of six (6) months from the date of delivery. We will repair or replace products that prove to be defective during the warranty period, provided they are returned to Mountain Hardware, Inc. No other warranty is expressed or implied. We are not liable for consequential damages. We reserve the right to refuse to repair any product that in our opinion has been subjected to abnormal electrical or mechanical abuse. Products out-of-warranty are subject to a minimal service fee.

Please feel free to contact us if you have any questions or problems.

Mountain Hardware

Located in the Santa Cruz Mountains of Northern California, Mountain Hardware, Inc. is a computer peripheral manufacturer dedicated to the production of use-oriented high technology products for the microcomputer. On-going research and development projects are geared to the continual supply of unique, innovative products that are easy to use and highly complementary in a broad variety of applications.

300 Harvey West Boulevard
Santa Cruz, CA 95060
(408) 429-8600