Speech Systems

38W255 DEERPATH ROAD
BATAVIA, ILLINOIS 60510

(312) 879-6880

PRECISION TIME MODULE

Operating Manual

Table of Contents

Archiver
: Pete Willard

Lettering
: This book was set in Ubuntu TrueType

Text Layout
: Ruby Gem - ASCIIDOCTOR-PDF

Precision Time Module

By Rich Parry

Speech Systems *Color Computer Specialists* 38 W 255 Deerpath Road Batavia, Il 50510

PRECISION TIME MODULE Operating Manual

This rare user manual was turning to dust in my shed. It has been recreated here for archival purposes and distributed under fair use due to its very limited effect on the market value of this SPEECH SYSTEMS product. Due to the deterioration of the printed paper, it has been re-typeset. *Some minor edits have been made for clarity* as some spots were difficult to follow and begged for some editing. Also, due to changes in Daylight Savings adjustment dates, the chip no longer updates at the correct times. In the 1980's, Motorola marked the `MC146818` "NOT RECOMMENDED FOR NEW DESIGNS" and afterward, Motorola made the `MC146818` chip obsolete. DALLAS/MAXIM do make a replacement for this part under numbers DS12885, DS12887, DS1685, or DS1687. With a small adjustment, these parts can be used in place of the `MC146818`. [2/28/2018 - PW]

Copyright Notice

Warranty

SPEECH SYSTEMS warranties the PRECISION TIME MODULE against defects in material and workmanship for a period of Ninety Days from the date of purchase to the original purchaser. The obligation of SPEECH SYSTEMS is limited to the repair or replacement of the product, free of all charges, which proves defective during this period. This warranty does not cover damage due to accident, negligence, abuse ot tampering.

Disclaimer

SPEECH SYSTEMS makes no other warranties or guarantees, express, statutory or implied, of any kind whatsoever with respect to the product purchased, and any other implied warranty of merchantability or fitness for a particular purpose is hereby disclaimed by SPEECH SYSTEMS and excluded from any agreement made by SPEECH SYSTEMS. The product is sold on an "as is" basis. SPEECH SYSTEMS will not be responsible for any damage of any kind not covered by the exclusive remedies set forth in this limited warranty. SPEECH SYSTEMS will not be responsible for any special, indirect or consequential damage caused by its products.

SPEECH SYSTEMS reserves the right to make changes to any products or specifications described in this manual without notification.

Moving Tape to Disk

All programs supplied with the PRECISION TIME MODULE are in **BASIC**. Therefore, transferring these programs to disk is very simple.

1. Insert the tape and press play on the tape player.
2. Type `CLOAD` followed by |ENTER| and the first program will load.
3. Now type `SAVE` "*PROGRAM*" and the program will be saved to disk.
4. Repeat until all programs are saved.

Disk Owners

The PRECISION TIME MODULE is a hardware device that normally fits into the slot reserved for the disk controller. Therefore disk owners will require some type of expansion device to allow simultaneous connection of the PRECISION TIME MODULE and a disk controller.

The PRECISION TIME MODULE will work in all expansion units that we know of. It will work perfectly well with SPEECH SYSTEMS Y-CABLE product. The Y-CABLE is a cable that contains a connector that plugs into the Color Computer and at the other end are two connectors just like the one that is inside the Color Computer. You insert your disk controller into one connector and the PRECISION TIME MODULE into the other.

If the Radio Shack Multi-Pak is used, the PRECISION TIME MODULE may be placed into any slot. If you use the Multi-Pak with the PRECISION TIME MODULE you do not need place the slot control switch on the front of the Multi-Pak in any particular position. In fact, you probably have your disk controller in slot 4 and the switch in position 4 and you can leave it that way. In other words, the PRECISION TIME MODULE is active at all times.

Owners who use a BT-1000 or the BT-2000 by BASIC TECHNOLOGY, will have to add an integrated circuit on the PRECISION TIME MODULE board. In the lower left hand corner of the PRECISION TIME MODULE is a place for a 7407 IC. You may insert this chip yourself, but we suggest you call SPEECH SYSTEMS first since opening the PRECISION TIME MODULE will void the warranty.

Getting Started

Insert the PRECISION TIME MODULE in the Color Computer directly or in an expansion unit if you are using a disk system. Load and RUN the program `SETTIME` to set the current time and date. Then load and RUN the `CLOCK` program. A description and explanation of these programs follows.

Program Descriptions

There are 3 **BASIC** programs supplied with your PRECISION TIME MODULE, they are: `SETTIME`, `CLOCK`, and `NUCLEUS`. The source listings are enclosed as a convenience. It is our hope that with an examination of these programs, together with this manual, you will have the necessary information to properly use the PRECISION TIME MODULE.

SETTIME

This program allows you to set the time and date on the module. When the PRECISION TIME MODULE leaves the factory, it is preset and we hope that you should not have to set it again. However, from time to time you may find this necessary. When the program is run, you are prompted to enter the time and date.

The program assumes that the hour value is entered using the 24 hour format, which is a value from 0 to 23.

```
For example, 2:00 in the afternoon is 14 hundred hours and 2:00 in the morning is 2 hundred hours.
```

The PRECISION TIME MODULE is initialized as a 24 hour clock rather than a 12 hour clock with AM and PM indications. If you wish to modify the program to initialize the PRECISION TIME MODULE as a 12 hour clock, we direct you to the data sheet sheet excerpts for the `MC146818` in this manual. It describes how to program the registers as a 12 hour clock.

The program will initialize the PRECISION TIME MODULE to use Daylight Savings Time. This means that on the last Sunday in April the PRECISION TIME MODULE increments from 1:59:59 AM to 3:00:00 AM. In addition, on the last Sunday in October when the time first reaches 1:59:59 AM it changes to 1:00:00 AM.

Daylight Savings periods in the United States were changed in 2007, making the DST adjustment dates in the `MC146818` no longer correct.

These special updates do not occur when the `DSE` bit is set to a `0`. Once again we direct you to the data sheet on the `MC146818` if you wish to modify the initialization steps that the `SETTIME` program uses.

```
100 '       SETTIME
110 '     BY RICH PARRY
120 '
130 '  COPYRIGHT (C) 1984
140 '    SPEECH SYSTEMS
150 '
160 ' THIS PROGRAM WILL ASK THE
170 ' USER  THE CURRENT TIME AND
180 ' DATE AND START THE CLOCK.
190 ' WHEN INSTRUCTED BY THE USER.
200 '
210 LET RTC=&HFF6E 'BASE ADDRESS OF RTC
220 DIM D(9)
230 '
240 ' REQUEST TIME AND DATE
250 '
260 CLS
270 PRINT "         SPEECH SYSTEMS"
280 PRINT "     CLOCK SETTIME PROGRAM"
290 PRINT@136,"MONTH (1-12)";:INPUT D(8)
300 IF D(8) < 1 OR D(8) > 12 THEN 290
310 PRINT@168,"DAY OF MONTH (1-31)";:INPUT D(7)
320 IF D(7) < 0 OR D(7) > 31 THEN 310
330 PRINT@200,"YEAR (0-99)";:INPUT D(9)
340 IF D(9) < 0 OR D(9) > 99 THEN 330
350 PRINT @232, "DAY OF WEEK (SUN=1)";:INPUT D(6)
360 IF D(6) < 1 OR D(6) > 7 THEN 350
370 PRINT@264,"HOUR (0-23)";:INPUT D(4)
380 IF D(4) < 0 OR D(4) > 23 THEN 370
390 PRINT@296,"MINUTE (1-59)";:INPUT D(2)
400 IF D(2) < 0 OR D(2) > 59 THEN 390
410 PRINT@328,"SECOND (1-59)";:INPUT D(0)
420 IF D(0) < 0 OR D(0) > 59 THEN 410
430 PRINT @384,"IS THIS CORRECT (Y/N)";
440 INPUT A$
450 IF A$ = "N" THEN 260
460 IF A$<>"Y" THEN 430
470 '
480 ' STORE DATE AND TIME
490 '
500 PRINT "HIT ANY KEY TO START"
510 IF INKEY$ = "" THEN 510
520 POKE RTC, 11    'CALL REG B
530 POKE RTC +1,&H80'STOP CLOCK
540 FOR I = 0 TO 9
550 POKE RTC, I
560 POKE RTC + 1, D(I)
570 NEXT I
580 POKE RTC, 10    'CALL REG A
590 POKE RTC +1,&H20'XTAL TYPE
600 POKE RTC, 11    'CALL REG B
610 POKE RTC +1,&H07'24 HOUR CLOCK
620 END
```

CLOCK

This program displays on the screen the time and date.

When the program starts, it first looks to see if the TRANSLATOR program (32K version) is loaded. The TRANSLATOR is a text to speech program that we supply with our speech synthesizers, the VOICE and SUPER VOICE. If this program is loaded, the time will be spoken each and every hour of the day. In addition, hitting any key on the keyboard will force the speech synthesizer to speak the time. You may wish to have the speech synthesizer speak the time more or less frequently (i.e. every 15 minutes).

We trust that an examination of the program and a little ingenuity will a1low you to perform this customizing.

If you do not have our speech synthesizer, the date and time will merely be displayed.

```
100 '           CLOCK
110 '       BY RICH PARRY
120 '
130 '     COPYRIGHT (C) 1984
140 '       SPEECH SYSTEMS
150 '
160 'A PROGRAM TO READ THE DATE/
170 'TIME OF THE RTC AND DISPLAY
180 'IT ON THE SCREEN.
190 CLS
200 '
210 CLEAR100,&H5EFF
220 ADR = &H5F00
230 BUF = ADR + 28
240 DEFUSR0 = ADR
250 IF PEEK(&H6000)=127 AND PEEK(&H6005)=32 THEN TALK=1 ELSE TALK=0
260 IF TALK=1 THEN DEFUSR1=&H6005 : GOTO 330
270 PRINT"THE TRNSLATE PROGRAM WHICH"
280 PRINT"COMES WITH OUR SPEECH"
290 PRINT"SYNTHESIZER IS NOT LOADED."
300 PRINT"THEREFORE, THIS PROGRAM"
310 PRINT"WILL NOT SPEAK THE TIME"
320 GOTO 350
330 PRINT"TRNSLATE IS LOADED, THIS"
340 PRINT"PROGRAM WILL SPEAK THE TIME"
350 PRINT:PRINT:PRINT
360 PRINT"HIT ANY KEY TO START"
370 IF INKEY$="" THEN 370
380 '
390 ' POKE ML PROGRAM IN DATA
400 ' DATA STATEMENTS IN MEMORY
410 '
420 FOR I=0 TO 27
430 READ DT
440 POKE ADR + I, DT
450 NEXT I
460 CLS
470 TREF=9999  'ANY NON REAL REFERENCE VALUE
480 SEC = 9999 ' ANY NON REAL SECOND VALUE
490 A=USR(0)
500 IF INKEY$<>"" THEN TREF=9999 : SEC=9999
510 IF PEEK(BUF+9) = SEC THEN 490
520 SEC = PEEK(BUF+9)  ' SAVE FOR FUTURE COMPARISON
530 YR = PEEK(BUF + 0)
540 MO = PEEK(BUF + 1)
550 DM = PEEK(BUF + 2)
560 DW = PEEK(BUF + 3)
570 HR = PEEK(BUF + 5)
```

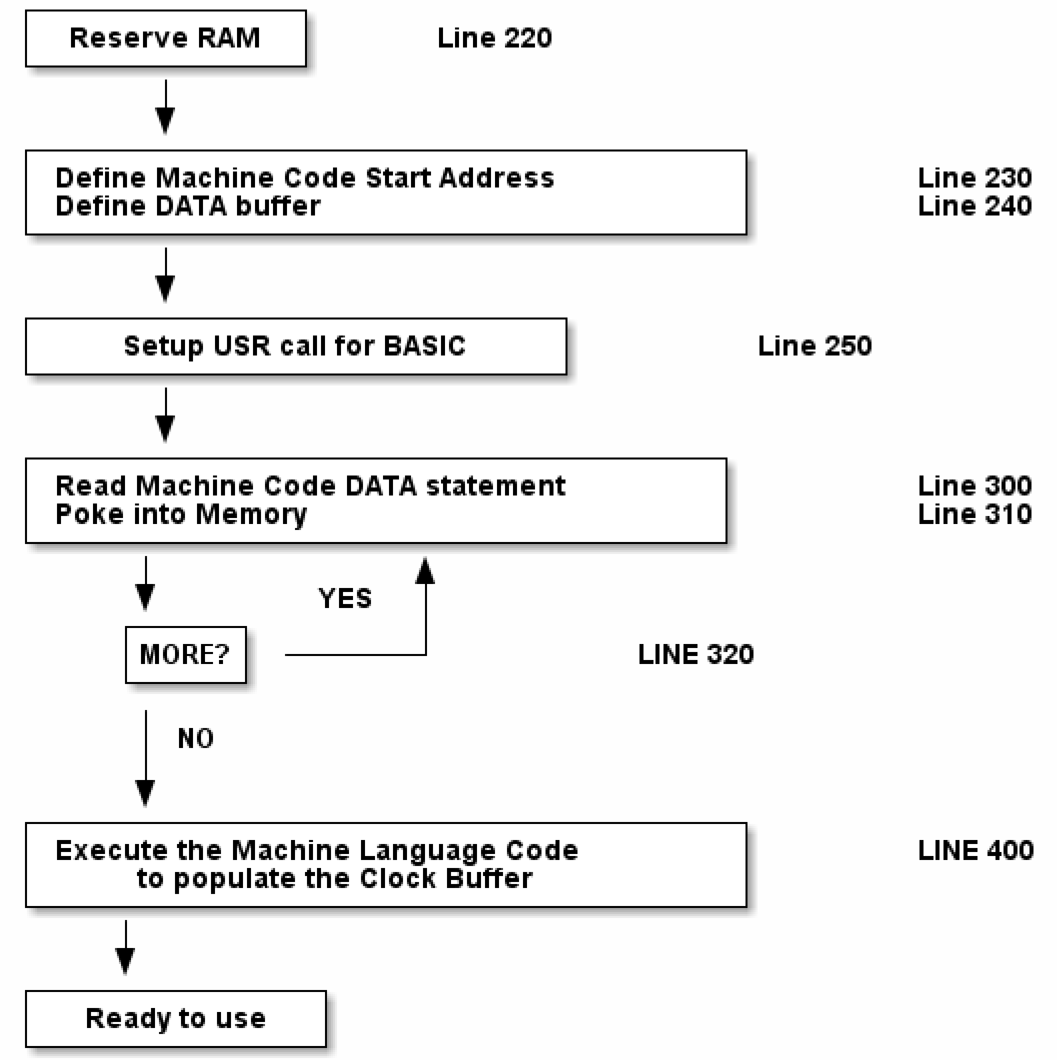
```
580 MI = PEEK(BUF + 7)
590 SE = PEEK(BUF + 9)
600 PRINT @300,"";
610 PRINT USING "##:##:##"; HR; MI; SE;
620 PRINT @196,"";
630 ON DW GOSUB 860,800,810,820,830,840,850
640 ON MO GOSUB 870,880,890,900,910,920,930,940,950,960,970,980
650 PRINT STR$(DM);
660 PRINT ", ";
670 PRINT YR;
680 IF TALK=0 OR TREF=HR THEN 750
690 TREF = HR : ' UPDATE REFERENCE
700 X$=USR1("THE TIME IS")
710 X$=USR1(STR$(HR))
720 X$=USR1("HOURS")
730 X$=USR1(STR$(MI))
740 X$=USR1("MINUTES")
750 GOTO 490
760 DATA &H30,&H8D,&H00,&H18,&HC6,&H09,&H86,&H0A
770 DATA &HB7,&HFF,&H6E,&HB6,&HFF,&H6F,&H2B,&HF6
780 DATA &HF7,&HFF,&H6E,&HB6,&HFF,&H6F,&HA7,&H80
790 DATA &H5A,&H2C,&HEB,&H39
800 PRINT "MONDAY   "; : RETURN
810 PRINT "TUESDAY  "; : RETURN
820 PRINT "WEDNESDAY"; : RETURN
830 PRINT "THURSDAY "; : RETURN
840 PRINT "FRIDAY   "; : RETURN
850 PRINT "SATURDAY "; : RETURN
860 PRINT "SUNDAY   "; : RETURN
870 PRINT "JANUARY  "; : RETURN
880 PRINT "FEBRUARY "; : RETURN
890 PRINT "MARCH    "; : RETURN
900 PRINT "APRIL    "; : RETURN
910 PRINT "MAY      "; : RETURN
920 PRINT "JUNE     "; : RETURN
930 PRINT "JULY     "; : RETURN
940 PRINT "AUGUST   "; : RETURN
950 PRINT "SEPTEMBER"; : RETURN
960 PRINT "OCTOBER  "; : RETURN
970 PRINT "NOVEMBER "; : RETURN
980 PRINT "DECEMBER "; : RETURN
```

NUCLEUS

This is not meant to be a complete **BASIC** program but rather it is the nucleus or starting point of a typical **BASIC** program that makes use of time and date. Any programs written in **BASIC** that need to access the time and date resources will need to have this code included since the clock machine code must be placed somewhere in memory.

Since the machine language program is completely position independent, it can handily be placed anywhere in memory. In the example shown, we have placed it at the top of memory for a 32K/64K Color Computer system. This is probably the best place for for it in most situations and as a result it should not be necessary to alter the program `NUCLEUS` very often.

Let's look at the program in detail to see just how the PRECISION TIME MODULE works.



LINE 220

```
220 CLEAR100,&H7EFF
```

In this line we are reserving memory for the machine language program that we are about to POKE into memory.

In the example, we are actually reserving 256 bytes when really only 38 bytes are necessary (call us wasteful if you like).

LINE 230 - 240

```
230 ADR = &H7F00
240 BUF = ADR + 28
```

In line 230 we are specifying the beginning of the machine language program and in line 240 we are specifying the beginning of the I0 byte buffer area where the time and date are placed by the machine language program.

The buffer area is 28 bytes after the starting location since the machine language program requires 28 bytes starting at location zero. An examination of the memory map diagram shown on a separate page should make this a bit clearer. The point to remember is that the machine language program requires 28 bytes.

Each time the machine language code is called it places the time and date at the very end of itself, a place we call the buffer area.

LINE 250

```
250 DEFUSR0 = ADR
```

Line 250 tells **BASIC** where we have placed the machine language subroutine.

LINES 290 - 360

```
290 FOR I = 0 TO 27
300 READ DT
310 POKE ADR + I, DT
320 NEXT I
330 DATA &H30,&H8D,&H00,&H18,&HC6,&H09,&H86,&H0A
340 DATA &HB7,&HFF,&H6E,&HB6,&HFF,&H6F,&H2B,&HF6
350 DATA &HF7,&HFF,&H6E,&HB6,&HFF,&H6F,&HA7,&H80
360 DATA &H5A,&H2C,&HEB,&H39
```

Lines 290 to 320 merely POKE’s into memory the machine language subroutines which are contained in the **BASIC** `DATA` statements on lines 330 to 360.

LINE 400

```
400 A=USR(0)
```

We are now really done, all we need do to find the current time and date is to call the subroutine (line 400) and then PEEK into the buffer memory for the time and date values (see lines 440 through 500).

We believe that a little study of the **BASIC** program `NUCLEUS` and the memory map diagram entitled "Format For Date & Time in Memory" should give you all you need to take advantage of the PRECISION TIME MODULE.

```
100 '
110 '                 NUCLEUS
120 '
130 ' THIS PROGRAM CONTAINS THE NEUCLEUS OF A BASIC
140 ' PROGRAM THAT USES THE PRECISION TIME MODULE
150 ' IT IS INTENDED AS A STARTING POINT FOR
160 ' WRITING YOUR OWN BASIC PROGRAM THAT REQUIRES
170 ' THE TIME AND DATE
180 '
190 '
200 ' SAVE SPACE FOR ML PROGRAM
210 '
220 CLEAR100,&H7EFF
230 ADR = &H7F00
240 BUF = ADR + 28
250 DEFUSR0 = ADR
260 '
270 ' POKE ML PROGRAM IN MEMORY
280 '
290 FOR I = 0 TO 27
300 READ DT
310 POKE ADR + I, DT
320 NEXT I
330 DATA &H30,&H8D,&H00,&H18,&HC6,&H09,&H86,&H0A
340 DATA &HB7,&HFF,&H6E,&HB6,&HFF,&H6F,&H2B,&HF6
350 DATA &HF7,&HFF,&H6E,&HB6,&HFF,&H6F,&HA7,&H80
360 DATA &H5A,&H2C,&HEB,&H39
370 '
380 ' CALL FOR DATE AND TIME
390 '
400 A=USR(0)
410 '
420 ' PEEK INTO MEMORY FOR TIME
430 '
440 YR = PEEK(BUF + 0) :' YEAR
450 MO = PEEK(BUF + 1) : 'MONTH
460 DM = PEEK(BUF + 2) : ' DAY OF THE MONTH
470 DW = PEEK(BUF + 3) : ' DAY OF THE WEEK
480 HR = PEEK(BUF + 5) : ' HOUR
490 MI = PEEK(BUF + 7) : ' MINUTES
500 SE = PEEK(BUF + 9) : ' SECOND
```

The Machine Language Program

One may ask why it is program that we must use a machine language to find out the time and date. The answer lies in the fact that BASIC is too slow. There is an instant in time lasting a few milliseconds in which the PRECISION TIME MODULE is rolling over. Rolling over means that once a second all the internal registers of the PRECISION TIME MODULE ic (MC146818) are changing. Although it is fast, it is not instantaneous. This means that there is a period in time when the data is invalid.

We must therefore examine the IC to find out if the data is valid. We do this by examining the `UPDATE IN PROGRESS` bit of one of the registers. An examination of the program will show that we loop until the data is valid and when it is ready we then read the time and date and store it in memory in the buffer area at the end of the program. The process of reading the time and date and storing must be done very rapidly, therefore a machine language program is used.

```
        NAM     NUCLEUS
*
* A POSITION INDEPENDENT PROGRAM TO PROCESS THE DATE AND TIME OF
* THE MC146818 AND STORE IT IN A BUFFER WHICH CAN BE
* USED IN A BASIC OR MACHINE LANGUAGE PROGRAM
*
* THE DALLAS DS1287 IS COMPATIBLE WITH THE MOTOROLA MC146818 RTC. IT
* CONTAINS AN INTEGRATED CRYSTAL CLOCK SOURCE AND LITHIUM BATTERY
*
* ADDRESS MAP & REGISTER DATA
* ===========================
*
*  RTC +ADDR    FUNCTION        HEX     BUFFER #
*       0       SECONDS         00      9
*       1       SECONDS ALARM   01      8       (NOT USED)
*       2       MINUTES         02      7
*       3       MINUTE ALARM    03      6       (NOT USED)
*       4       HOURS           04      5
*       5       HOUR ALARM      05      4       (NOT USED)
*       6       WEEK DAY        06      3
*       7       MONTH DAY       07      2
*       8       MONTH           08      1
*       9       YEAR            09      0
*      10       REGISTER A      0A      N/A
*      11       REGISTER B      0B      N/A
*      12       REGISTER C      0C      N/A
*      13       REGISTER D      0D      N/A
*
* REGISTER A FUNCTIONS
*
*  B7   B6   B5   B4   B3   B2   B1   B0
*  UIP  DV2  DV1  DV0  RS3  RS2  RS1  RS0
*
* BIT 7 - UIP UPDATE IN PROGRESS. READ ONLY BIT. LOGIC "1" = UPDATE IN
*         PROGRESS. IT MUST BE AT LOGIC "O" TO STORE A VALUE IN THE RTC
*
* BITS 6,5,4 - TIME BASE DIVIDER (ALLOWED VALUES 0,1,2)
*              1 = 4.194304 MHZ
*              2 = 1.048576 MHZ
*              3 = 32.768 KHZ
*              THESE BITS ARE NOT REQUIRED BY THE DS1287
*
* BITS 3,2,1,0 - RATE SELECTOR, USED TO SELECT OUTPUT SQUARE WAVE
*              HEX 00 = OUTPUTS DISABLED
*              HEX 01 = PIR 30.517uS SQW 32.768 KHZ
*              ..  ..
*              HEX FF = PIR 500MS SQW 2 HZ
*              THESE BITS ARE NOT REQUIRED BY THE DS1287
*
```

```
*
* REGISTER B FUNCTIONS
*
*  B7   B6   B5   B4   B3   B2   B1   B0
*  SET  PIE  AIE  UIE  SQWE DM  24/12 DSE
*
* SET  1 = ABORT THE UPDATE CYCLE AND ALLOW A WRITE TO OCCUR
*      0 = NORMAL OPERATION
*
* PIE  1 = PERIODIC INTERRUPT. IT DRIVES THE IRQ PIN LOW FOR A PERIOD DEFINED
*          BY REGISTER A BITS 3,2,1,0.
*      0 = BLOCKS THE PIE FROM CHANGING THE IRQ STATE.
*
* AIE  1 = ALLOWS AN ALARM SETTING TO GENERATE AN IRQ
*      0 = DISALLOWS AN ALARM SETTING TO GENERATE AN IRQ
*
* UIE  1 = UPDATE ENDED INTERRUPT ALLOWS THE UPDATE ENDED FLAG TO GENERATE
*          AN IRQ
*      0 = PREVENTS UIE FLAG FROM GENERATING AN IRQ
*
* SQWE 1 = ALLOWS THE SQUARE WAVE RATE SELECT TO TOGGLE THE SQW PIN AT THE
*          RATE SELECTED BY REGISTER A.
*      0 = HOLDS THE SQW PIN LOW.
*
* DM   1 = BINARY DATA MODE
*      0 = BCD DATA MODE
*
* 24/12 1 = 24 HOUR MODE
*       0 = 12 HOUR MODE
*
* DSE  1 = ENABLE DAYLIGHT SAVINGS MODE
*      0 = DISABLE DAYLIGHT SAVINGS MODE
*
*
* REGISTER C FUNCTIONS
*
*  B7   B6   B5   B4   B3   B2   B1   B0
*  IRQF PF   AF   UF   0    0    0    0
*
* REGISTER C IS READ ONLY AND CONTAINS INTERRUPT FLAG INFORMATION
*
* IRQF - IRQ FLAG ANYTIME THE IRQF FLAG IS "1", THE IRQ PIN IS DRIVEN LOW.
* PF   - PERIODIC INTERRUPT FLAG
* AF   - ALARM FLAG
* UF   - UPDATE FLAG
*
* REGISTER D ONLY CONTAINS 1 ACTIVE BIT. THE MSB OF THE D REGISTER
* CONTAINS THE "VRT" VALUE INDICATING THAT THE RTC HAS GOOD POWER
* AND THE THE IT IS NO LONGER IN POWER DOWN MODE. A "0" IS READ WHEN
* THE POWER SENSE PIN IS LOW.
*
*
       ORG     $7000
RTCAD  EQU     $FF6E   * RTC MUX ADDRESS
RTCDT  EQU     RTCAD+1 * RTC DATA ADDRESS
*
*
START  LEAX    BUFFER,PCR     * PREPARE THE BUFFER
       LDB     #9             * START LOADING THE BUFFER AT LOCATION 9
                              * DECREMENT TILL WE REACH 0
                              * IT WILL LOAD SECONDS (9) THOUGH YEAR(0)
CKUIP  LDA     #10            * PREPARE TO SELECT REGISTER A
       STA     RTCAD          * STORE REGISTER SELECTION IN THE RTC
       LDA     RTCDT          * GET REGISTER CONTENTS
       BMI     CKUIP          * IS THE UPDATE BIT SET? YES, LOOP UNTIL NOT
       STB     RTCAD          * NO, THEN UPDATE VALUE
       LDA     RTCDT          * READ SELECTED DATA FROM RTC
       STA     0,X+           * STORE IT IN THE LOCAL BUFFER
       DECB                   * MOVE TO NEXT REGISTER AND BUFFER LOCATION
       BGE     CKUIP          * IF THE BUFFER VALUE IS NOT 0, LOOP
       RTS                    * BUFFER VALUE = 0, WE ARE DONE
*
BUFFER RMB     10             * RESERVE 10 LOCATIONS FOR THE BUFFER
       END     START
```

Battery Backup

The PRECISION TIME MODULE contains a nickel cadmium battery to keep the clock running during power those times that the computer is off. The battery should last approximately 1 month without a charge. The battery is rechargeable which means you should not have to replace it. Should the battery need recharging, merely place it in the computer for at least 8 hours.

Where is the PRECISION TIME MODULE in Memory

The PRECISION TIME MODULE is a memory mapped hardware peripheral and as a result it must be placed somewhere in memory. It requires only 2 bytes of address space. We have decided to place the PRECISION TIME MODULE at $FF6E and $FF6F. We know of no other peripheral for the Color Computer located here and as a result the PRECISION TIME MODULE should not interfere with any other hardware device.

Powering Down the PRECISION TIME MODULE

You may notice sometimes that an un-powered PRECISION TIME MODULE will show lost time or perhaps even gives an invalid time when powered up again.

This undesired result does have a root cause. When shutting the Color Computer off and power is removed from MC6809 there is a brief moment where it can do almost anything… including writing to the memory addresses used by the PRECISION TIME MODULE. To help avoid this issue, the PRECISION TIME MODULE does have a circuit that helps detect when the computer is about to lose power but it does have its limitations and the results cannot be guaranteed to work every single time. There will be times when the time is incorrect the next time the computer is turned on.

Advanced Information for Advanced Users

The following pages contain information that will be of use to users with a good background in machine language programming and a modicum of knowledge in electronics. We at SPEECH SYSTEMS have developed a powerful PRECISION TIME MODULE that is capable of many applications. To describe all the features of the PRECISION TIME MODULE would require a much larger manual. We hope the unit is easy to use for everyone, and for those advanced users, we supply the basic technical specifications of the chip and a schematic in the hopes that they will supply the additional effort to customize the PRECISION TIME MODULE to their particular application.

Interrupts

The PRECISION TIME MODULE IC (MC146818) is capable of supplying interrupts to the computer (6809). When the PRECISION TIME MODULE leaves the factory, the jumper which connects the interrupt line, PIN 19 of the chip is left vacant so there is no connect to the `/NMI` line of 6809, PIN 4. Should you wish to connect the PRECISION TIME MODULE for an interrupt based application, you must open the PRECISION TIME MODULE case and solder a jumper wire connecting the two holes provided. The two plated through holes are located slightly down and to the right of pin 1 on the edge connector. Please note that only the advanced programmer should consider making use of this feature.

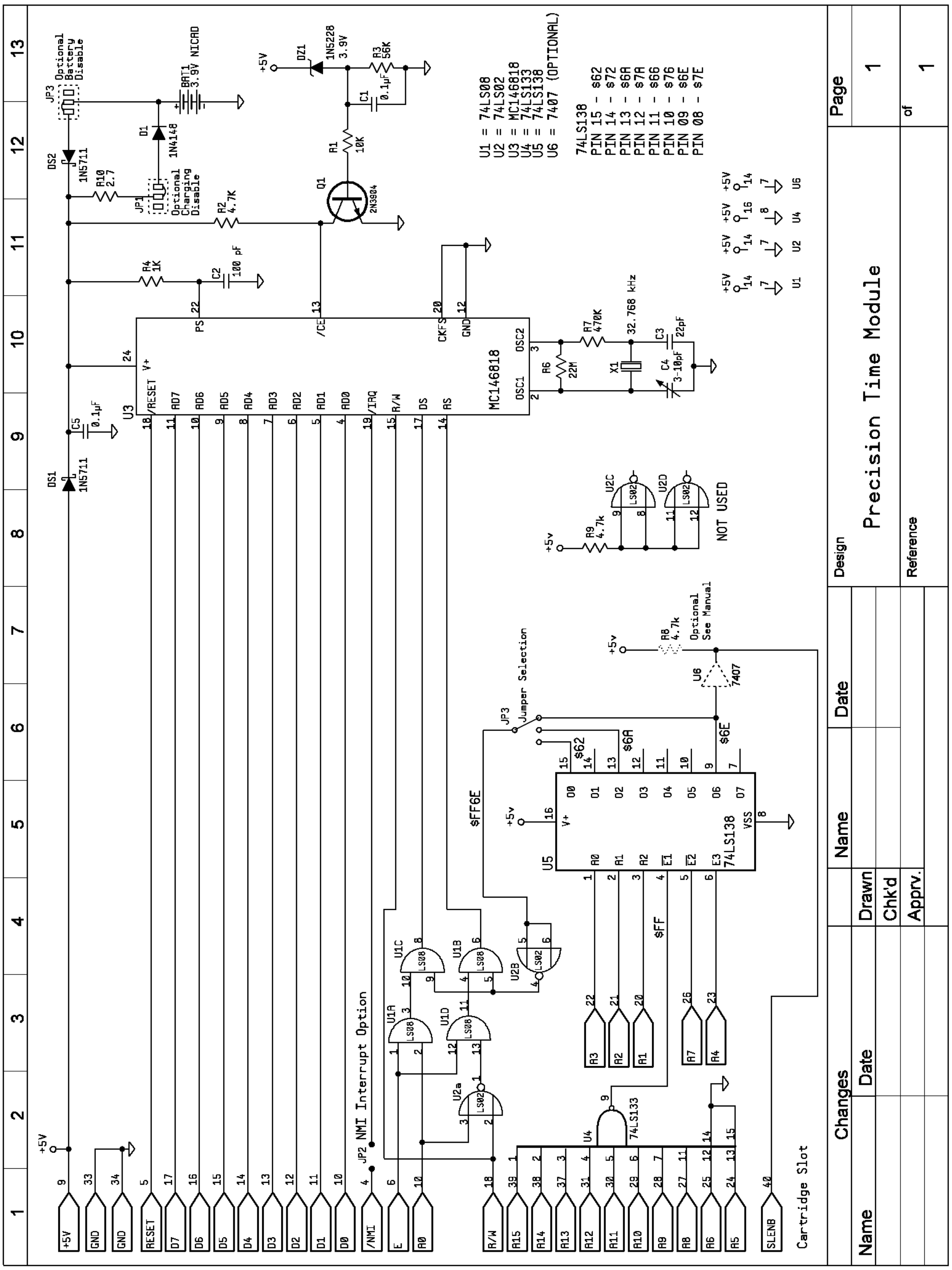
Proceed with caution. Opening the PRECISION TIME MODULE will void the warranty. You are expected to know what you are doing.

Adjusting The PRECISION TIME MODULE

The PRECISION TIME MODULE is calibrated at the factory. However, it may become necessary to fine tune the unit. Looking at the PRECISION TIME MODULE so the connector end that plugs into the color computer faces left, you will find an adjustable "trimmer capacitor" in the extreme upper left hand corner. Using a small screwdriver turn the capacitor dial slightly in one direction and record which way you turned it and how much. You will then need to wait a day or more to see if you turned it the correct amount and in the correct direction to get better crystal timing performance.

The adjustment dial "wraps around", therefore after turning it in one direction for more that 360 degrees you will be back to where you started. It also should be understood that opening the unit to make this adjustment voids the warranty.

Schematic Diagram



Hardware Details

The `MC146818` provided clock and calendar functions with 50 bytes of user RAM memory, and was designed to work with either the Motorola or Intel processor timing. These features could be made nonvolatile by providing a backup battery on the board to maintain the VDD supply. For timekeeping,the `MC146818` required a number of discrete components on the PCB along with a crystal to provide an input frequency. The input frequencies could be 32.768 kHz, 1.048576 MHz, or 4.194304 MHz.

MOTOROLA
SEMICONDUCTORS
3501 ED BLUESTEIN BLVD., AUSTIN, TEXAS 78721

MC146818A

Advance Information

CMOS

(HIGH-PERFORMANCE SILICON-GATE COMPLEMENTARY MOS)

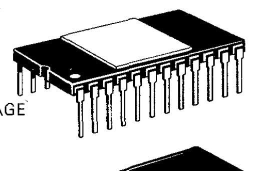
REAL-TIME CLOCK PLUS RAM

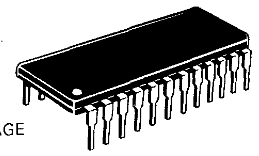
REAL-TIME CLOCK PLUS RAM (RTC)

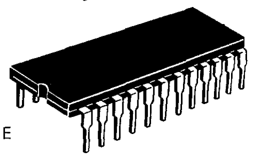
The MC146818A Real-Time Clock plus RAM is a peripheral device which includes the unique MOTEL concept for use with various microprocessors, microcomputers, and larger computers. This part combines three unique features: a complete time-of-day clock with alarm and one hundred year calendar, a programmable periodic interrupt and square-wave generator, and 50 bytes of low-power static RAM. The MC146818A uses high-speed CMOS technology to interface with 1 MHz processor buses, while consuming very little power.

The Real-Time Clock plus RAM has two distinct uses. First, it is designed as a battery powered CMOS part (in an otherwise NMOS/TTL system) including all the common battery backed-up functions such as RAM, time, and calendar. Secondly, the MC146818A may be used with a CMOS microprocessor to relieve the software of the timekeeping workload and to extend the available RAM of an MPU such as the MC146805E2.

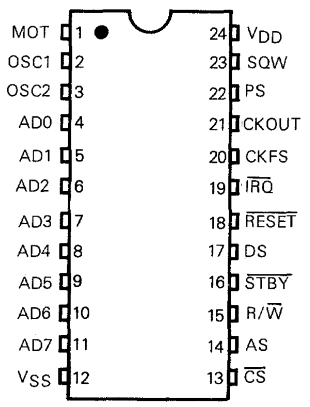
- Low-Power, High-Speed CMOS
- Internal Time Base and Oscillator
- Counts Seconds, Minutes, and Hours of the Day
- Counts Days of the Week, Date, Month, and Year
- 3 V to 6 V Operation
- Time Base Input Options: 4.194304 MHz, 1.048576 MHz, or 32.768 kHz
- Time Base Oscillator for Parallel Resonant Crystals
- 40 to 200 μW Typical Operating Power at Low Frequency Time Base
- 4.0 to 20 mW Typical Operating Power at High Frequency Time Base
- Binary or BCD Representation of Time, Calendar, and Alarm
- 12- or 24-Hour Clock with AM and PM in 12-Hour Mode
- Daylight Savings Time Option
- Automatic End of Month Recognition
- Automatic Leap Year Compensation
- Microprocessor Bus Compatible
- Selectable Between Motorola and Competitor Bus Timing
- Multiplexed Bus for Pin Efficiency
- Interfaced with Software as 64 RAM Locations
- 14 Bytes of Clock and Control Registers
- 50 Bytes of General Purpose RAM
- Status Bit indicates Data Integrity
- Bus Compatible Interrupt Signals ($\overline{IRQ}$)
- Three Interrupts are Separately Software Maskable and Testable
  Time-of-Day Alarm, Once-per-Second to Once-per-Day
  Periodic Rates from 30.5 μs to 500 ms
  End-of-Clock Update Cycle
- Programmable Square-Wave Output Signal
- Clock Output May Be Used as Microprocessor Clock Input
  At Time Base Frequency ÷ 1 or ÷ 4
- 24-Pin Dual-In-Line Package
- Quad Pack Also Available


L SUFFIX
CERAMIC PACKAGE
CASE 716


P SUFFIX
PLASTIC PACKAGE
CASE 709


S SUFFIX
CERDIP PACKAGE
CASE 623

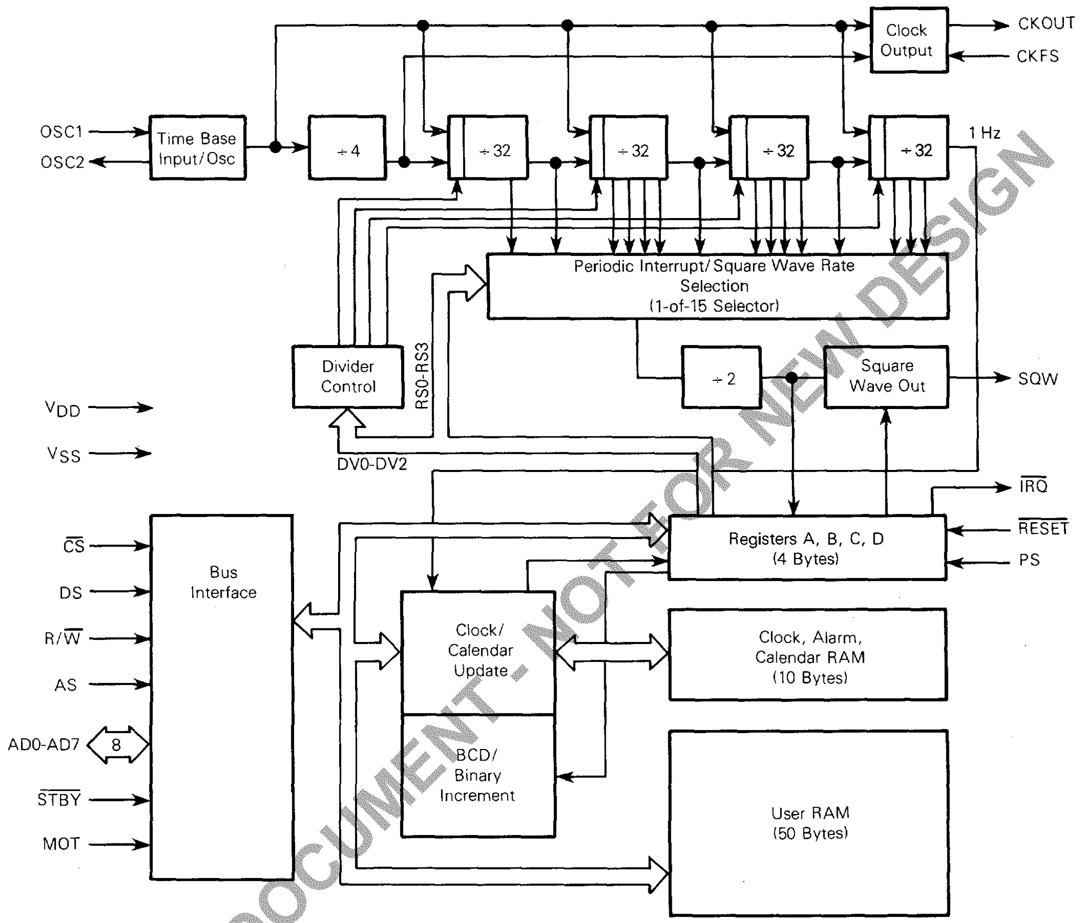
PIN ASSIGNMENT



This document contains information on a new product. Specifications and information herein are subject to change without notice.

ADI-1026

FIGURE 1 — BLOCK DIAGRAM



MAXIMUM RATINGS (Voltages referenced to V_{SS})

Ratings	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.3 to +8.0	V
All Input Voltages Except OSC1	V_{in}	$V_{SS}-0.5$ to $V_{DD}+0.5$	V
Current Drain per Pin Excluding V_{DD} and V_{SS}	I	10	mA
Operating Temperature Range MC146818A MC146818AC	T_A	T_L to T_H 0 to 70 −40 to 85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic Cerdip Ceramic	θ_{JA}	 120 65 50	°C/W

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high-impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{DD}$. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}).

SIGNAL DESCRIPTIONS

The block diagram in Figure 1, shows the pin connection with the major internal functions of the MC146818A Real-Time Clock plus RAM. The following paragraphs describe the function of each pin.

V_{DD}, V_{SS}

DC power is provided to the part on these two pins, V_{DD} being the more positive voltage. The minimum and maximum voltages are listed in the Electrical Characteristics tables.

MOT—MOTEL

The MOT pin offers flexibility when choosing bus type. When tied to V_{DD}, Motorola timing is used. When tied to V_{SS}, competitor timing is used. The MOT pin must be hardwired to the V_{DD} or V_{SS} supply and cannot be switched during operation of the MC146818A.

OSC1, OSC2 — TIME BASE, INPUTS

The time base for the time functions may be an external signal or the crystal oscillator. External square waves at 4.194304 MHz, 1.048576 MHz, or 32.768 kHz may be connected to OSC1 as shown in Figure 9. The internal time-base frequency to be used is chosen in Register A.

The on-chip oscillator is designed for a parallel resonant AT cut crystal at 4.194304 MHz, 1.048576 MHz or 32.768 kHz frequencies. The crystal connections are shown in Figure 10 and the crystal characteristics in Figure 11.

CKOUT — CLOCK OUT, OUTPUT

The CKOUT pin is an output at the time-base frequency divided by 1 or 4. A major use for CKOUT is as the input clock to the microprocessor; thereby saving the cost of a second crystal. The frequency of CKOUT depends upon the time-base frequency and the state of the CKFS pin as shown in Table 2.

CKFS — CLOCK OUT FREQUENCY SELECT, INPUT

When the CKFS pin is tied to V_{DD}, it causes CKOUT to be the same frequency as the time base at the OSC1 pin. When CKFS is tied to V_{SS}, CKOUT is the OSC1 time-base frequency divided by four. Table 2 summarizes the effect of CKFS.

TABLE 2 — CLOCK OUTPUT FREQUENCIES

Time Base (OSC1) Frequency	Clock Frequency Select Pin (CKFS)	Clock Frequency Output Pin (CKOUT)
4.194304 MHz	High	4.194304 MHz
4.194304 MHz	Low	1.048576 MHz
1.048576 MHz	High	1.048576 MHz
1.048576 MHz	Low	262.144 kHz
32.768 kHz	High	32.768 kHz
32.768 kHz	Low	8.192 kHz

SQW — SQUARE WAVE, OUTPUT

The SQW pin can output a signal from one of the 15 taps provided by the 22 internal-divider stages. The frequency of the SQW may be altered by programming Register A, as shown in Table 5. The SQW signal may be turned on and off using the SQWE bit in Register B.

AD0-AD7 — MULTIPLEXED BIDIRECTIONAL ADDRESS/DATA BUS

Multiplexed bus processors save pins by presenting the address during the first portion of the bus cycle and using the same pins during the second portion for data. Address-then-data multiplexing does not slow the access time of the MC146818A since the bus reversal from address to data is occurring during the internal RAM access time.

The address must be valid just prior to the fall of AS/ALE at which time the MC146818A latches the address from AD0 to AD5. Valid write data must be presented and held stable during the latter portion of the DS or $\overline{WR}$ pulses. In a read cycle, the MC146818A outputs eight bits of data during the latter portion of the DS or $\overline{RD}$ pulses, then ceases driving the bus (returns the output drivers to the high-impedance state) when DS falls in the Motorola case of MOTEL or $\overline{RD}$ rises in the other case.

AS — MULTIPLEXED ADDRESS STROBE, INPUT

A positive going multiplexed address strobe pulse serves to demultiplex the bus. The falling edge of AS or ALE causes the address to be latched within the MC146818A.

DS — DATA STROBE OR READ, INPUT

The DS pin has two interpretations via the MOTEL circuit. When emanating from a Motorola type processor, DS is a positive pulse during the latter portion of the bus cycle, and is variously called DS (data strobe), E (enable), and $\phi 2$ ($\phi 2$ clock). During read cycles, DS signifies the time that the RTC is to drive the bidirectional bus. In write cycles, the trailing edge of DS causes the Real-Time Clock plus RAM to latch the written data.

The second MOTEL interpretation of DS is that of $\overline{RD}$, $\overline{MEMR}$, or $\overline{I/OR}$ emanating from the competitor type processor. In this case, DS identifies the time period when the real-time clock plus RAM drives the bus with read data. This interpretation of DS is also the same as an output-enable signal on a typical memory.

R/$\overline{W}$ — READ/WRITE, INPUT

The MOTEL circuit treats the R/$\overline{W}$ pin in one of two ways. When a Motorola type processor is connected, R/$\overline{W}$ is a level which indicates whether the current cycle is a read or write. A read cycle is indicated with a high level on R/$\overline{W}$ while DS is high, whereas a write cycle is a low on R/$\overline{W}$ during DS.

The second interpretation of R/$\overline{W}$ is as a negative write pulse, $\overline{WR}$, $\overline{MEMW}$, and $\overline{I/OW}$ from competitor type processors. The MOTEL circuit in this mode gives R/$\overline{W}$ pin the same meaning as the write ($\overline{W}$) pulse on many generic RAMs.

$\overline{CS}$ — CHIP SELECT, INPUT

The chip-select ($\overline{CS}$) signal must be asserted (low) for a bus cycle in which the MC146818A is to be accessed. $\overline{CS}$ is not latched and must be stable during DS and AS (Motorola case of MOTEL) and during $\overline{RD}$ and $\overline{WR}$. Bus cycles which take place without asserting $\overline{CS}$ cause no actions to take place within the MC146818A. When $\overline{CS}$ is not used, it should be grounded. (See Figure 20).

$\overline{IRQ}$ — INTERRUPT REQUEST, OUTPUT

The $\overline{IRQ}$ pin is an active low output of the MC146818A that may be used as an interrupt input to a processor. The $\overline{IRQ}$ output remains low as long as the status bit causing the interrupt is present and the corresponding interrupt-enable bit is set. To clear the $\overline{IRQ}$ pin, the processor program normally reads Register C. The $\overline{RESET}$ pin also clears pending interrupts.

When no interrupt conditions are present, the $\overline{IRQ}$ level is in the high-impedance state. Multiple interrupting devices may thus be connected to an $\overline{IRQ}$ bus with one pullup at the processor.

$\overline{RESET}$ — RESET, INPUT

The $\overline{RESET}$ pin does not affect the clock, calendar, or RAM functions. On powerup, the $\overline{RESET}$ pin must be held low for the specified time, t_{RLH}, in order to allow the power supply to stabilize. Figure 12 shows a typical representation of the $\overline{RESET}$ pin circuit.

When $\overline{RESET}$ is low the following occurs:

a) Periodic Interrupt Enable (PIE) bit is cleared to zero,
b) Alarm Interrupt Enable (AIE) bit is cleared to zero,
c) Alarm Interrupt Enable (AIE) bit is cleared to zero,
d) Update ended Interrupt Flag (UF) bit is cleared to zero,
e) Interrupt Request status Flag (IRQF) bit is cleared to zero,
f) Periodic Interrupt Flag (PF) bit is cleared to zero,
g) The part is not accessible.
h) Alarm Interrupt Flag (AF) bit is cleared to zero,
i) $\overline{IRQ}$ pin is in high-impedance state, and
j) Square Wave output Enable (SQWE) bit is cleared to zero.

$\overline{STBY}$ — STAND—BY

The $\overline{STBY}$ pin, when active, prevents access to the MC146818A making it ideal for battery back-up applications. Stand-by operation incorporates a transparent latch. After data strobe (DS) goes low ($\overline{RD}$ or $\overline{WR}$ rises), $\overline{STBY}$ is recognized as a valid signal.

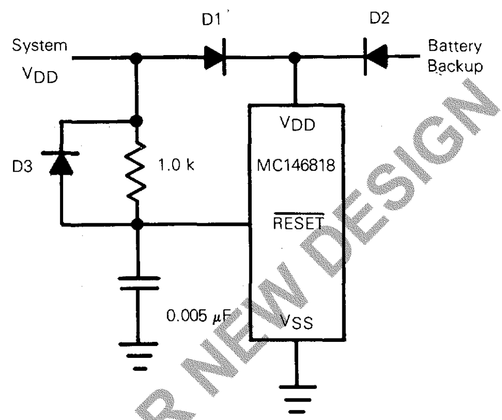
The $\overline{STBY}$ signal is totally asynchronous. Its transparent latch is opened by the falling edge of DS (rising edge of $\overline{RD}$ or $\overline{WR}$) and clocked by the rising edge of AS (ALE). Therefore, for $\overline{STBY}$ to be recognized, DS and AS should occur in pairs. When $\overline{STBY}$ goes low before the falling edge of DS (rising edge of $\overline{WR}$ or $\overline{RD}$), the current cycle is completed at that edge and the next cycle will not be executed.

PS — POWER SENSE, INPUT

The power-sense pin is used in the control of the valid RAM and time (VRT) bit in Register D. When the PS pin is low the VRT bit is cleared to zero.

When using the VRT feature during powerup, the PS pin must be externally held low for the specified t_{PLH} time. As power is applied, the VRT bit remains low indicating that the contents of the RAM, time registers, and calendar are not guaranteed. PS must go high after powerup to allow the VRT bit to be set by a read of register D.

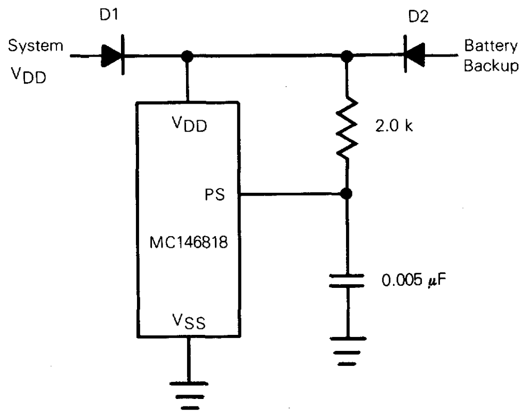
FIGURE 12—TYPICAL POWERUP DELAY CIRCUIT FOR RESET



D1 = MBD701 (Schottky) or Equivalent
D2 = D3 = 1N4148 or Equivalent

Note: If the RTC is isolated from the MPU or MCU power by a diode drop, care must be taken to meet V_{in} requirements.

FIGURE 13 — TYPICAL POWERUP DELAY CIRCUIT FOR POWER SENSE



D1 = MBD701 (Schottky) or Equivalent
D2 = 1N4148 or Equivalent

POWER-DOWN CONSIDERATIONS

In most systems, the MC146818A must continue to keep time when system power is removed. In such systems, a conversion from system power to an alternate power supply, usually a battery, must be made. During the transition from system to battery power, the designer of a battery backed-up RTC system must protect data integrity, minimize power consumption, and ensure hardware reliability.

The stand-by ($\overline{STBY}$) pin controls all bus inputs (R/$\overline{W}$, DS, AS, AD0-AD7) $\overline{STBY}$, when negated, disallows any unintended modification of the RTC data by the bus. $\overline{STBY}$ also reduces power consumption by reducing the number of transitions seen internally.

Power consumption may be further reduced by removing resistive and capacitive loads from the clock out (CKOUT) pin and the squarewave (SQW) pin.

During and after the power source conversion, the V_{IN} maximum specification must never be exceeded. Failure to meet the V_{IN} maximum specification can cause a virtual SCR to appear which may result in excessive current drain and destruction of the part.

ADDRESS MAP

Figure 14 shows the address map of the MC146818A. The memory consists of 50 general purpose RAM bytes, 10 RAM bytes which normally contain the time, calendar, and alarm data, and four control and status bytes. All 64 bytes are directly readable and writable by the processor program except for the following: 1) Registers C and D are read only, 2) bit 7 of Register A is read only, and 3) the high-order bit of the seconds byte is read only. The contents of four control and status registers (A, B, C, and D) are described in **REGISTERS.**

TIME, CALENDAR, AND ALARM LOCATIONS

The processor program obtains time and calendar information by reading the appropriate locations. The program may initialize the time, calendar, and alarm by writing to these RAM locations. The contents of the 10 time, calendar, and alarm bytes may be either binary or binary-coded decimal (BCD).

Before initializing the internal registers, the SET bit in Register B should be set to a "1" to prevent time/calendar updates from occurring. The program initializes the 10 locations in the selected format (binary or BCD), then indicates the format in the data mode (DM) bit of Register B. All 10 time, calendar, and alarm bytes must use the same data mode, either binary or BCD. The SET bit may now be cleared to allow updates. Once initialized the real-time clock makes all updates in the selected data mode. The data mode cannot be changed without reinitializing the 10 data bytes.

Table 3 shows the binary and BCD formats of the 10 time, calendar, and alarm locations. The 24/12 bit in Register B establishes whether the hour locations represent 1-to-12 or 0-to-23. The 24/12 bit cannot be changed without reinitializing the hour locations. When the 12-hour format is selected the high-order bit of the hours byte represents PM when it is a "1".

The time, calendar, and alarm bytes are not always accessible by the processor program. Once per second the 10 bytes are switched to the update logic to be advanced by one second and to check for an alarm condition. If any of the 10 bytes are read at this time, the data outputs are undefined. The update lockout time is 248 μs at the 4.194304 MHz and 1.048567 MHz time bases and 1948 μs for the 32.768 kHz time base. The Update Cycle section shows how to accommodate the update cycle in the processor program.

The three alarm bytes may be used in two ways. First, when the program inserts an alarm time in the appropriate hours, minutes, and seconds alarm locations, the alarm interrupt is initiated at the specified time each day if the alarm enable bit is high. The second usage is to insert a "don't care" state in one or more of three alarm bytes. The "don't care" code is any hexadecimal byte from C0 to FF. That is, the two most-significant bits of each byte, when set to "1", create a "don't care" situation. An alarm interrupt each hour is created with a "don't care" code in the hours alarm location. Similarly, an alarm is generated every minute with "don't care" codes in the hours and minutes alarm bytes. The "don't care" codes in all three alarm bytes create an interrupt every second.

FIGURE 14 — ADDRESS MAP

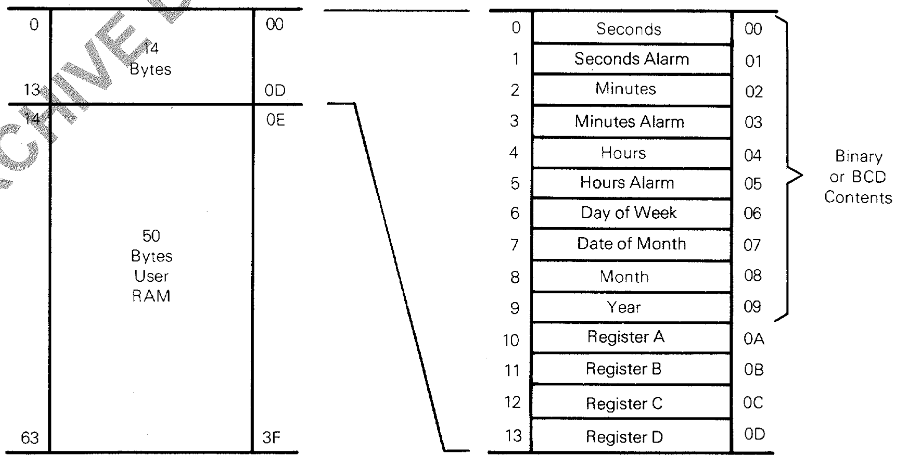


TABLE 3 — TIME, CALENDAR, AND ALARM DATA MODES

Address Location	Function	Decimal Range	Range		Example*	
			Binary Data Mode	BCD Data Mode	Binary Data Mode	BCD Data Mode
0	Seconds	0-59	$00-$3B	$00-$59	15	21
1	Seconds Alarm	0-59	$00-$3B	$00-$59	15	21
2	Minutes	0-59	$00-$3B	$00-$59	3A	58
3	Minutes Alarm	0-59	$00-$3B	$00-$59	3A	58
4	Hours (12 Hour Mode)	1-12	$01-$0C (AM) and $81-$8C (PM)	$01-$12 (AM) and $81-$92 (PM)	05	05
	Hours (24 Hour Mode)	0-23	$00-$17	$00-$23	05	05
5	Hours Alarm (12 Hour Mode)	1-12	$01-$0C (AM) and $81-$8C (PM)	$01-$12 (AM) and $81-$92 (PM)	05	05
	Hours Alarm (24 Hour Mode)	0-23	$00-$17	$00-23	05	05
6	Day of the Week Sunday = 1	1-7	$01-$07	$01-$07	05	05
7	Date of the Month	1-31	$01-$1F	$01-$31	0F	15
8	Month	1-12	$01-$0C	$01-$12	02	02
9	Year	0-99	$00-$63	$00-$99	4F	79

*Example: 5:58:21 Thursday 15 February 1979 (time is AM)

STATIC CMOS RAM

The 50 general purpose RAM bytes are not dedicated within the MC146818A. They can be used by the processor program, and are fully available during the update cycle.

When time and calendar information must use battery back-up, very frequently there is other non-volatile data that must be retained when main power is removed. The 50 user RAM bytes serve the need for low-power CMOS battery-backed storage, and extend the RAM available to the program.

When further CMOS RAM is needed, additional MC146818As may be included in the system. The time/calendar functions may be disabled by holding the DV0-DV2 dividers, in Register A, in the reset state by setting the SET bit in Register B or by removing the oscillator. Holding the dividers in reset prevents interrupts or SQW output from operating while setting the SET bit allows these functions to occur. With the dividers clear, the available user RAM is extended to 59 bytes. The high-order bit of the seconds byte, bit 7 of Register A, and all bits of Registers C and D cannot effectively be used as general purpose RAM.

INTERRUPTS

The RTC plus RAM includes three separate fully automatic sources of interrupts to the processor. The alarm interrupt may be programmed to occur at rates from once-per-second to one-a-day. The periodic interrupt may be selected for rates from half-a-second to 30.517 μs. The update-ended interrupt may be used to indicate to the program that an update cycle is completed. Each of these independent interrupt conditions are described in greater detail in other sections.

The processor program selects which interrupts, if any, it wishes to receive. Three bits in Register B enable the three interrupts. Writing a "1" to a interrupt-enable bit permits that interrupt to be initiated when the event occurs. A "0" in the interrupt-enable bit prohibits the IRQ pin from being asserted due to the interrupt cause.

If an interrupt flag is already set when the interrupt becomes enabled, the $\overline{\text{IRQ}}$ pin is immediately activiated, though the interrupt initiating the event may have occurred much earlier. Thus, there are cases where the program should clear such earlier initiated interrupts before first enabling new interrupts.

When an interrupt event occurs, a flag bit is set to a "1" in Register C. Each of the three interrupt sources have separate flag bits in Register C, which are set independent of the state of the corresponding enable bits in Register B. The flag bit may be used with or without enabling the corresponding enable bits.

In the software scanned case, the program does not enable the interrupt. The "interrupt" flag bit becomes a status bit, which the software interrogates, when it wishes. When the software detects that the flag is set, it is an indication to software that the "interrupt"event occurred since the bit was last read.

However, there is one precaution. The flag bits in Register C are cleared (record of the interrupt event is erased) when Register C is read. Double latching is included with Register C so the bits which are set are stable throughout the read cycle. All bits which are high when read by the program are cleared, and new interrupts (on any bits) are held after the read cycle. One, two or three flag bits may be found to be set when Register C is used. The program should inspect all utilized flag bits every time Register C is read to insure that no interrupts are lost.

The second flag bit usage method is with fully enabled interrupts. When an interrupt-flag bit is set and the corresponding interrupt-enable bit is also set, the $\overline{\text{IRQ}}$ pin is asserted low. $\overline{\text{IRQ}}$ is asserted as long as at least one of the three interrupt sources has its flag and enables bits both set. The IRQF bit in Register C is a "1" whenever the $\overline{\text{IRQ}}$ pin is being driven low.

The processor program can determine that the RTC initiated the interrupt by reading Register C. A "1" in bit 7

(IRQF bit) indicates that one or more interrupts have been initiated by the part. The act of reading Register C clears all the then-active flag bits, plus the IRQF bit. When the program finds IRQF set, it should look at each of the individual flag bits in the same byte which have the corresponding interrupt-mask bits set and service each interrupt which is set. Again, more than one interrupt-flag bit may be set.

DIVIDER STAGES

The MC146818A has 22 binary-divider stages following the time base as shown in Figure 1. The output of the dividers is a 1 Hz signal to the update-cycle logic. The divers are controlled by three divider bus (DV2, DV1, and DV0) in Register A.

DIVIDER CONTROL

The divider-control bits have three uses, as shown in Table 4. Three usable operating time bases may be selected (4.194304 MHz, 1.048576 MHz, or 32.768 kHz). The divider chain may be held at reset, which allows precision setting of the time, When the divider is changed from reset to an operating time base, the first update cycle is one-half second later. The divider-control bits are also used to facilitate testing the MC146818A.

SQUARE-WAVE OUTPUT SELECTION

Fifteen of the 22 divider taps are made available to a 1-of-15 selector as shown in Figure 1. The first purpose of selecting a divider tap is to generate a square-wave output signal at the SQW pin. The RS0-RS3 bits in Register A establish the square-wave frequency as listed in Table 5. The SQW frequency selection shares the 1-of-15 selector with periodic interrupts.

Once the frequency is selected, the output of the SQW pin may be turned on and off under program control with the square-wave output selection bits, or the SQWE output-enable bit may generate an asymmetrical waveform at the time of execution. The square-wave output pin has a number of potential uses. For example, it can serve as a frequency standard for external use, a frequency synthesizer, or could be used to generate one or more audio tones under program control.

TABLE 4 — DIVIDER CONFIGURATIONS

Time-Base Frequency	Divider Bits Register A			Operation Mode	Divider Reset	Bypass First N-Divider Bits
	DV2	DV1	DV0			
4.194304 MHz	0	0	0	Yes	–	N = 0
1.048576 MHz	0	0	1	Yes	–	N = 2
32.768 kHz	0	1	0	Yes	–	N = 7
Any	1	1	0	No	Yes	–
Any	1	1	1	No	Yes	–

Note: Other combinations of divider bits are used for test purposes only.

TABLE 5 — PERIODIC INTERRUPT RATE AND SQUARE WAVE OUTPUT FREQUENCY

Select Bits Register A				4.194304 or 1.048576 MHz Time Base		32.768 kHz Time Base	
RS3	RS2	RS1	RS0	Periodic Interrupt Rate t_{PI}	SQW Output Frequency	Periodic Interrupt Rate t_{PI}	SQW Output Frequency
0	0	0	0	None	None	None	None
0	0	0	1	30.517 μs	32.768 kHz	3.90625 ms	256 Hz
0	0	1	0	61.035 μs	16.384 kHz	7.8125 ms	128 Hz
0	0	1	1	122.070 μs	8.192 kHz	122.070 μs	8.192 kHz
0	1	0	0	244.141 μs	4.096 kHz	244.141 μs	4.096 kHz
0	1	0	1	488.281 μs	2.048 kHz	488.281 μs	2.048 kHz
0	1	1	0	976.562 μs	1.024 kHz	976.562 μs	1.024 kHz
0	1	1	1	1.953125 ms	512 Hz	1.953125 ms	512 Hz
1	0	0	0	3.90625 ms	256 Hz	3.90625 ms	256 Hz
1	0	0	1	7.8125 ms	128 Hz	7.8125 ms	128 Hz
1	0	1	0	15.625 ms	64 Hz	15.625 ms	64 Hz
1	0	1	1	31.25 ms	32 Hz	31.25 ms	32 Hz
1	1	0	0	62.5 ms	16 Hz	62.5 ms	16 Hz
1	1	0	1	125 ms	8 Hz	125 ms	8 Hz
1	1	1	0	250 ms	4 Hz	250 ms	4 Hz
1	1	1	1	500 ms	2 Hz	500 ms	2 Hz

PERIODIC INTERRUPT SELECTION

The periodic interrupt allows the $\overline{IRQ}$ pin to be triggered from once every 500 ms to once every 30.517 µs. The periodic interrupt is separate from the alarm interrupt which may be output from once per second to once per day.

Table 5 shows that the periodic interrupt rate is selected with the same Register A bits which select the square-wave frequency. Changing one also changes the other. But each function may be separately enabled so that a program could switch between the two features or use both. The SQW pin is enabled by the SQWE bit in Register B. Similarly the periodic interrupt is enabled by the PIE bit in Register B.

Periodic interrupt is usable by practically all real-time systems. It can be used to scan for all forms of inputs from contact closures to serial recieve bits or bytes. It can be used in multiplexing displays or with software counters to measure inputs, create output intervals, or await the next needed software function.

UPDATE CYCLE

The MC146818A executes an update cycle once per second, assuming one of the proper time bases is in place, the DV0-DV2 divider is not clear, and the SET bit in Register B is clear. The SET bit in the "1" state permits the program to initialize the time and calendar bytes by stopping an existing update and preventing a new one from occurring.

The primary function of the update cycle is to increment the seconds byte, check for overflow, increment the minutes byte when appropriate and so forth through to the year of the century byte. The update cycle also compares each alarm byte with the corresponding time byte and issues an alarm if a match or if a "don't care" code (11XXXXXX) is present in all three positions.

With a 4.194304 MHz or 1.048576 MHz time base the update cycle takes 248 µs while a 32.768 kHz time base update cycle takes 1984 µs. During the update cycle, the time, calendar, and alarm bytes are not accessible by the processor program. The MC146818A protects the program from reading transitional data. This protection is provided by switching the time, calendar, and alarm portion of the RAM off the microprocessor bus during the entire update cycle. If the processor reads these RAM locations before the update is complete, the output will be undefined. The update in progress (UIP) status bit is set during the interval.

A program which randomly accesses the time and date information finds data unavailable statistically once every 4032 attempts. Three methods of accommodating nonavailability during update are usable by the program. In discussing the three methods, it is assumed that at random points user programs are able to call a subroutine to obtain the time of day.

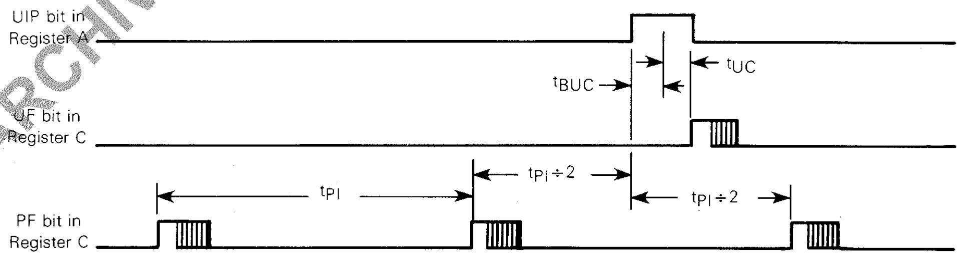
The first method of avoiding the update cycle uses the update-ended interrupt. If enabled, an interrupt occurs after every update cycle which indicates that over 999 ms are available to read valid time and date information. During this time a display could be updated or the information could be transferred to continuously available RAM. Before leaving the interrupt service routine, the IRQF bit in Register C should be cleared.

The second method uses the update-in-progress bit (UIP) in Register A to determine if the update cycle is in progress or not. The UIP bit will pulse once per second. Statistically, the UIP bit will indicate that time and date information is unavailable once every 2032 attempts. After the UIP bit goes high, the update cycle begins 244 µs later. Therefore, if a low is read on the UIP bit, the user has at least 244 µs before the time/calendar data will be changed. If a "1" is read in the UIP bit, the time/calendar data may not be valid. The user should avoid interrupt service routines that would cause the time needed to read valid time/calendar data to exceed 244 µs.

The third method uses a periodic interrupt to determine if an update cycle is in progress. The UIP bit in Register A is set high between the setting of the PF bit in Register C (see Figure 15). Periodic interrupts that occur at a rate of greater than $t_{BUC}+t_{UC}$ allow valid time and date information to be read at each occurrence of the periodic interrupt. The reads should be completed within $(T_{PI} \div 2)+t_{BUC}$ to ensure that data is not read during the update cycle.

To properly setup the internal counters for daylight savings time operation, the user must set the time at least two seconds before the rollover will occur. Likewise, the time must be set at least two seconds before the end of the 29th or 30th day of the month.

FIGURE 15 — UPDATE-ENDED AND PERIODIC INTERRUPT RELATIONSHIP



t_{PI} = Periodic Interrupt Time Interval (500 ms, 250 ms, 125 ms, 62.5 ms, etc. per Table 5)
t_{UC} = Update Cycle Time (248 µs or 1984 µs)
t_{BUC} = Delay Time Before Update Cycle (244 µs)

REGISTERS

The MC146818A has four registers which are accessible to the processor program. The four registers are also fully accessible during the update cycle.

REGISTER A ($0A)

MSB ... LSB — Read/Write Register except UIP

b7	b6	b5	b4	b3	b2	b1	b0
UIP	DV2	DV1	DV0	RS3	RS2	RS1	RS0

UIP — The update in progress (UIP) bit is a status flag that may be monitored by the program. When UIP is a "1", the update cycle is in progress or will soon begin. When UIP is a "0", the update cycle is not in progress and will not be for at least 244 μs (for all time bases). This is detailed in Table 6. The time, calendar, and alarm information in RAM is fully available to the program when the UIP bit is zero — it is not in transition. The UIP bit is a read-only bit, and is not affected by Reset. Writing the SET bit in Register B to a "1" inhibits any update cycle and then clears the UIP status bit.

TABLE 6 — UPDATE CYCLE TIMES

UIP Bit	Time Base (OSC1)	Update Cycle Time (t_{UC})	Minimum Time Before Update Cycle (t_{BUC})
1	4.194304 MHz	248 μs	—
1	1.048576 MHz	248 μs	—
1	32.768 kHz	1984 μs	—
0	4.194304 MHz	—	244 μs
0	1.048576 MHz	—	244 μs
0	32.768 kHz	—	244 μs

DV2, DV1, DV0 — Three bits are used to permit the program to select various conditions of the 22-stage divider chain. The divider selection bits identify which of the three time-base frequencies is in use. Table 4 shows that time bases of 4.194304 MHz, 1.048576 MHz, and 32.768 kHz may be used. The divider selection bits are also used to reset the divider chain. When the time/calendar is first initialized, the program may start the divider at the precise time stored in the RAM. When the divider reset is removed, the first update cycle begins one-half second later. These three read/write bits are not affected by RESET.

RS3, RS2, RS1, RS0 — The four rate selection bits select one of 15 tapes on the 22-stage divider, or disable the divider output. The tap selected may be used to generate an output square wave (SQW pin) and/or a periodic interrupt. The program may do one of the following: 1) enable the interrupt with the PIE bit, 2) enable the SQW output pin with the SQWE bit, 3) enable both at the same time at the same rate, or 4) enable neither. Table 5 lists the periodic interrupt rates and the square-wave frequencies that may be chosen with the RS bits. These four bits are read/write bits which are not affected by RESET.

REGISTER B ($0B)

MSB ... LSB — Read/Write Register

b7	b6	b5	b4	b3	b2	b1	b0
SET	PIE	AIE	UIE	SQWE	DM	24/12	DSE

SET — When the SET bit is a "0", the update cycle functions normally by advancing the counts once-per-second. When the SET bit is written to a "1", any update cycle in progress is aborted and the program may initialize the time and calendar bytes without an update occurring in the midst of initializing. SET is a read/write bit which is not modified by RESET or internal functions of the MC146818A.

PIE — The periodic interrupt enable (PIE) bit is a read/write bit which allows the periodic-interrupt flag (PF) bit in Register C to cause the IRQ pin to be driven low. A program writes a "1" to the PIE bit in order to receive periodic interrupts at the rate specified by the RS3, RS2, RS1, and RS0 bits in Register A. A zero in PIE blocks IRQ from being initiated by a periodic interrupt, but the periodic flag (PF) bit is still set at the periodic rate. PIE is not modified by any internal MC146818A functions, but is cleared to "0" by a RESET.

AIE — The alarm interrupt enable (AIE) bit is a read/write bit which when set to a "1" permits the alarm flag (AF) bit in Register C to assert IRQ. An alarm interrupt occurs for each second that the three time bytes equal the three alarm bytes (including a "don't care" alarm code by binary 11XXXXXX). When the AIE bit is a "0", the AF bit does not initiate an IRQ signal. The RESET pin clears AIE to "0". The internal functions do not affect the AIE bit.

UIE — The UIE (update-ended interrupt enable) bit is a read/write bit which enables the update-end flag (UF) bit in Register C to assert IRQ. The RESET pin going low or the SET bit going high clears the UIE bit.

SQWE — When the square-wave enable (SQWE) bit is set to a "1" by the program, a square-wave signal at the frequency specified in the rate selection bits (RS3 to RS0) appears on the SQW pin. When the SQWE bit is set to a zero the SQW pin is held low. The state of SQWE is cleared by the RESET pin. SQWE is a read/write bit.

DM — The data mode (DM) bit indicates whether time and calendar updates are to use binary or BCD formats. The DM bit is written by the processor program and may be read by the program, but is not modified by any internal functions or RESET. A "1" in DM signifies binary data, while a "0" in DM specifies binary-coded-decimal (BCD) data.

24/12 — The 24/12 control bit establishes the format of the hours bytes as either the 24-hour mode (a "1") or the 12-hour mode (a "0"). This is a read/write bit, which is affected only by software.

DSE — The daylight savings enable (DSE) bit is a read/write bit which allows the program to enable two special updates (when DSE is a "1"). On the last Sunday in April the time increments from 1:59:59 AM to 3:00:00 AM. On the last Sunday in October when the time first reaches 1:59:59 AM it changes to 1:00:00 AM. These special updates do not occur when the DSE bit is a "0". DSE is not changed by any internal operations or reset.

REGISTER C ($0C)

MSB ... LSB — Read-Only Register

b7	b6	b5	b4	b3	b	b1	b0
IRQF	PF	AF	UF	0	0	0	0

IRQF — The interrupt request flag (IRQF) is set to a "1" when one or more of the following are true:

PF = PIE = "1"
AF = AIE = "1"
UF = UIE = "1"
i.e., IRQF = PF•PIE + AF•AIE + UF•UIE

Any time the IRQF bit is a "1", the $\overline{\text{IRQ}}$ pin is driven low. All flag bits are cleared after Register C is read by the program or when the $\overline{\text{RESET}}$ pin is low.

PF — The periodic interrupt flag (PF) is a read-only bit which is set to a "1" when a particular edge is detected on the selected tap of the divider chain. The RS3 to RS0 bits establish the periodic rate. PF is set to a "1" independent of the state of the PIE bit. PF being a "1" initiates an $\overline{\text{IRQ}}$ signal and sets the IRQF bit when PIE is also a "1". The PF bit is cleared by a $\overline{\text{RESET}}$ or a software read of Register C.

AF — A "1" in the AF (alarm interrupt flag) bit indicates that the current time has matched the alarm time. A "1" in the AF causes the $\overline{\text{IRQ}}$ pin to go low, and a "1" to appear in the IRQF bit, when the AIE bit also is a "1." A $\overline{\text{RESET}}$ or a read of Register C clears AF.

UF — The update-ended interrupt flag (UF) bit is set after each update cycle. when the UIE bit is a "1", the "1" in UF causes the IRQF bit to be a "1", asserting $\overline{\text{IRQ}}$. UF is cleared by a Register C read or a $\overline{\text{RESET}}$.

b3 TO b0 — The unused bits of Status Register 1 are read as "0's". They can not be written.

REGISTER D ($0D)

MSB LSB

b7	b6	b5	b4	b3	b2	b1	b0
VRT	0	0	0	0	0	0	0

Read Only Register

VRT — The valid RAM and time (VRT) bit indicates the condition of the contents of the RAM, provided the power sense (PS) pin is satisfactorily connected. A "0" appears in the VRT bit when the power-sense pin is low. The processor program can set the VRT bit when the time and calendar are initialized to indicate that the RAM and time are valid. The VRT is a read only bit which is not modified by the $\overline{\text{RESET}}$ pin. The VRT bit can only be set by reading Register D.

b6 TO b0 — The remaining bits of Register D are unused. They cannot be written, but are always read as "0's."

TYPICAL INTERFACING

The MC146818A is best suited for use with microprocessors which generate an address-then-data multiplexed bus. Figures 16 and 17 show typical interfaces to bus-compatible processors. These interfaces assume that the address decoding can be done quickly. However, if standard metalgate CMOS gates are used, the $\overline{\text{CS}}$ setup time may be violated. Figure 18 illustrates an alternative method of chip selection which will accommodate such slower decoding.

The MC146818A can be interfaced to single-chip microcomputers (MCU) by using eleven port lines as shown in Figure 19. Non-multiplexed bus microprocessors can be interfaced with additional support.

There is one method of using the multiplexed bus MC146818A with non-multiplexed bus processors. The interface uses available bus control signals to multiplex the address and data bus together.

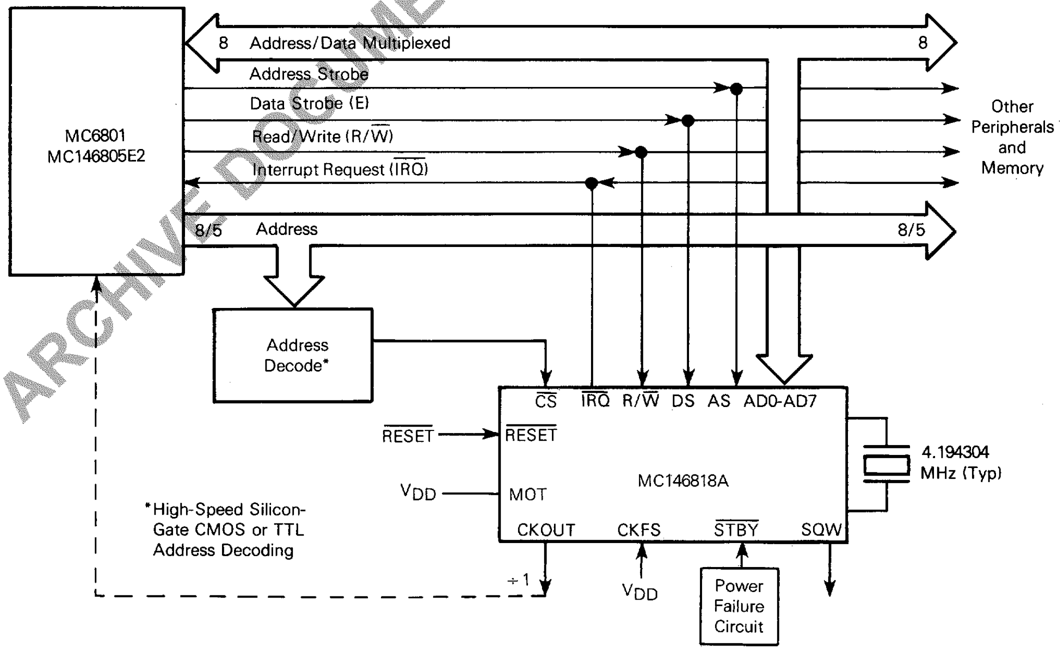
An example using either the Motorola MC6800, MC6802, MC6808, or MC6809 microprocessor is shown in Figure 20. When the MC146818A is I/O mapped as shown in Figures 19 and 20, the AS and DS inputs should be left in a low state when the part is not being accessed. Refer to the $\overline{\text{STBY}}$ pin description for the conditions which must be met before $\overline{\text{STBY}}$ can be recognized.

Figure 21 illustrates the subroutines which may be used for data transfers in a non-multiplexed system. The subroutines should be entered with the registers containing the following data:

Accumulator A: The address of the RTC to be accessed.
Accumulator B: Write: The data to be written.
Read: The data read from the RTC.

The RTC is mapped to two consecutive memory locations — RTC and RTC+1 as shown in Figure 20.

FIGURE 16 — MC146818A INTERFACED WITH MOTOROLA COMPATIBLE MULTIPLEXED BUS MICROPROCESSORS



MOTOROLA *Semiconductor Products Inc.*

FIGURE 17 — MC146818A INTERFACED WITH COMPETITOR COMPATIBLE MULTIPLEXED BUS MICROPROCESSORS

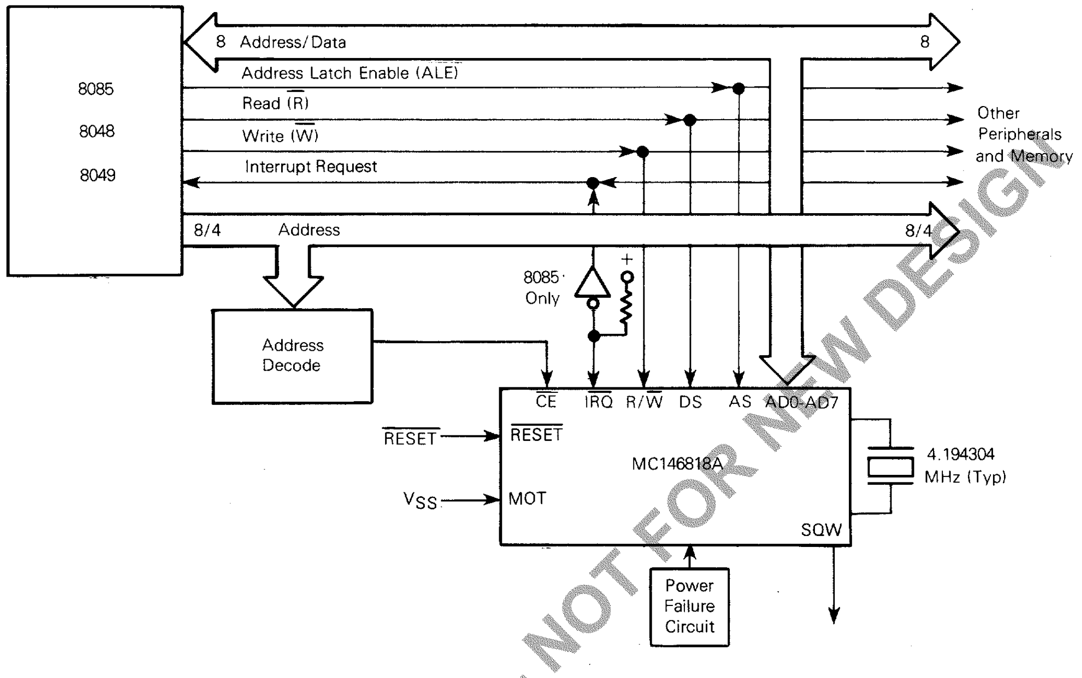
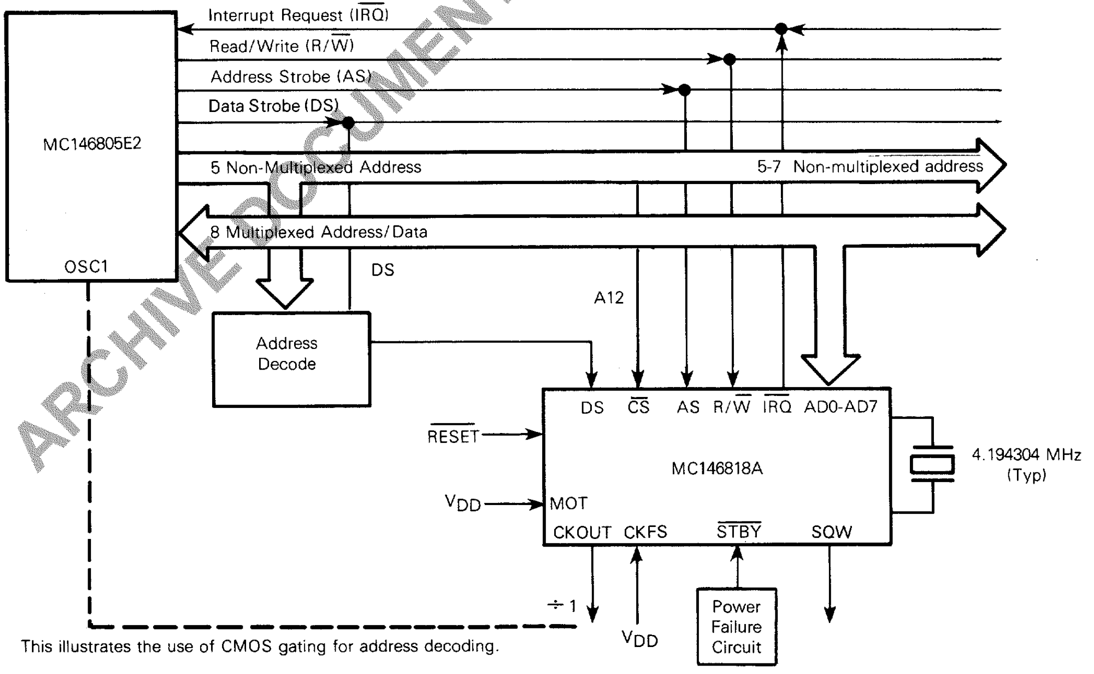
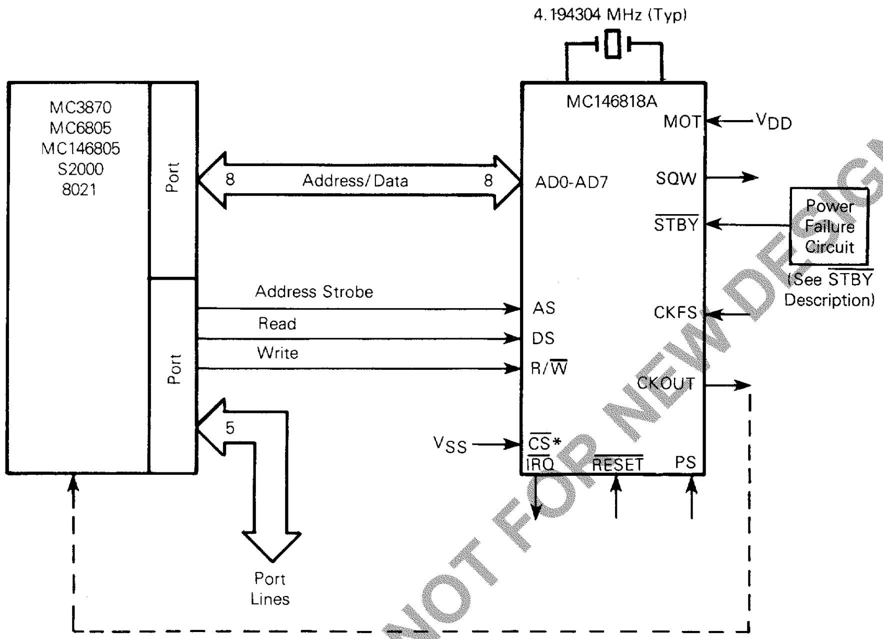


FIGURE 18 — MC146818A INTERFACE WITH MC146805E2 CMOS MULTIPLEXED MICROPROCESSOR WITH SLOW ADDRESSING DECODING



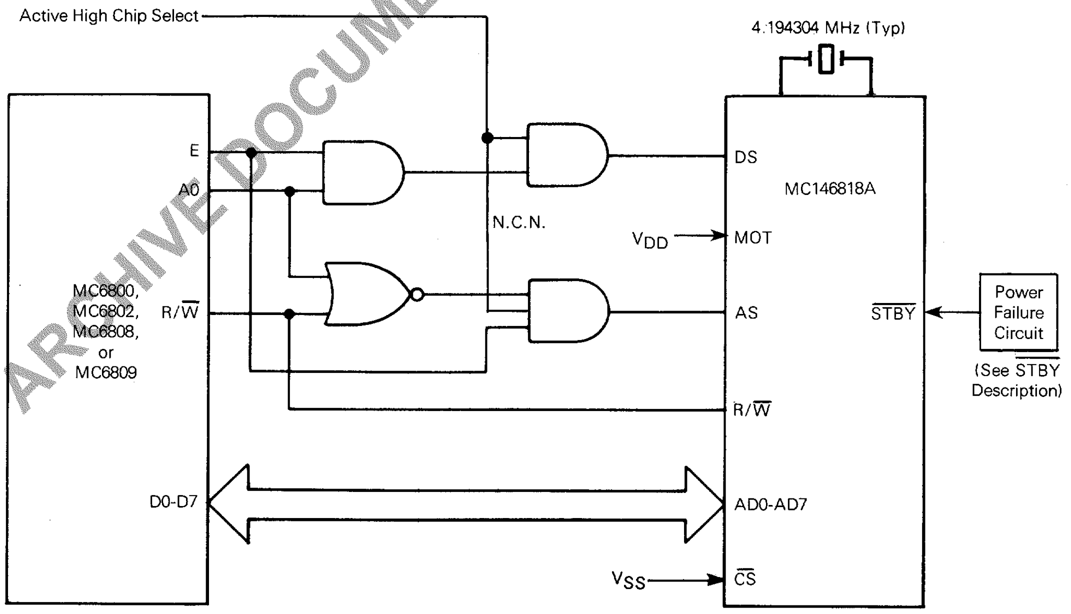
This illustrates the use of CMOS gating for address decoding.

FIGURE 19 — MC146818A INTERFACED WITH THE PORTS OF A TYPICAL SINGLE CHIP MICROCOMPUTER



*NOTE: $\overline{CS}$ can be controlled by a port pin (if available).

FIGURE 20 — MC146818A INTERFACED WITH MOTOROLA PROCESSORS

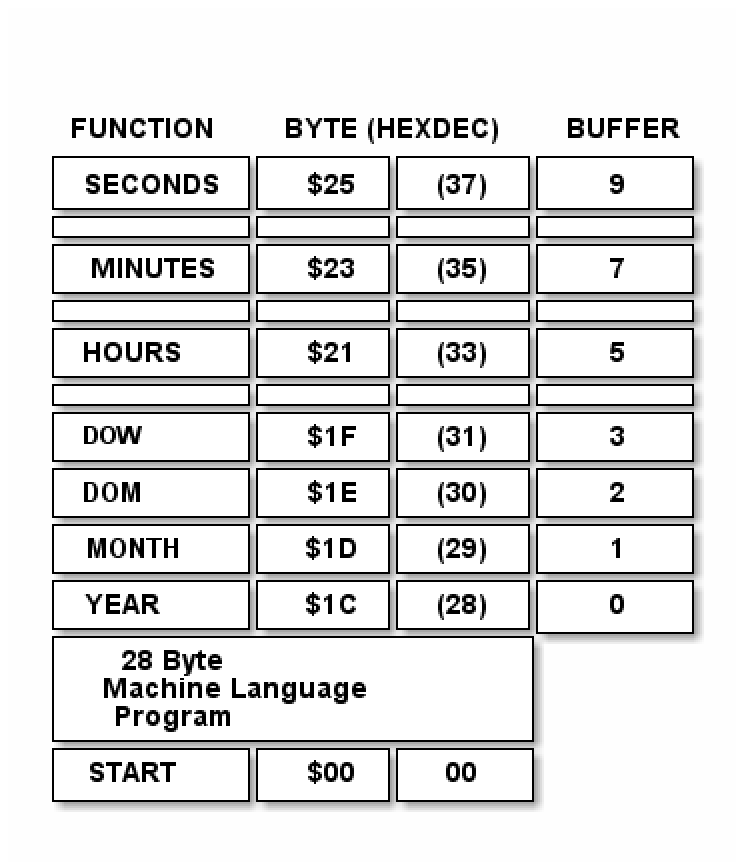


Time Module Register Map

Buffer Allocation

RTC +ADDR	FUNCTION	HEX	BUFFER #
0	SECONDS	00	9
1	SECONDS ALARM	01	8 (NOT USED)
2	MINUTES	02	7
3	MINUTE ALARM	03	6 (NOT USED)
4	HOURS	04	5
5	HOUR ALARM	05	4 (NOT USED)
6	WEEK DAY	06	3
7	MONTH DAY	07	2
8	MONTH	08	1
9	YEAR	09	0
10	REGISTER A	0A	N/A
11	REGISTER B	0B	N/A
12	REGISTER C	0C	N/A
13	REGISTER D	0D	N/A

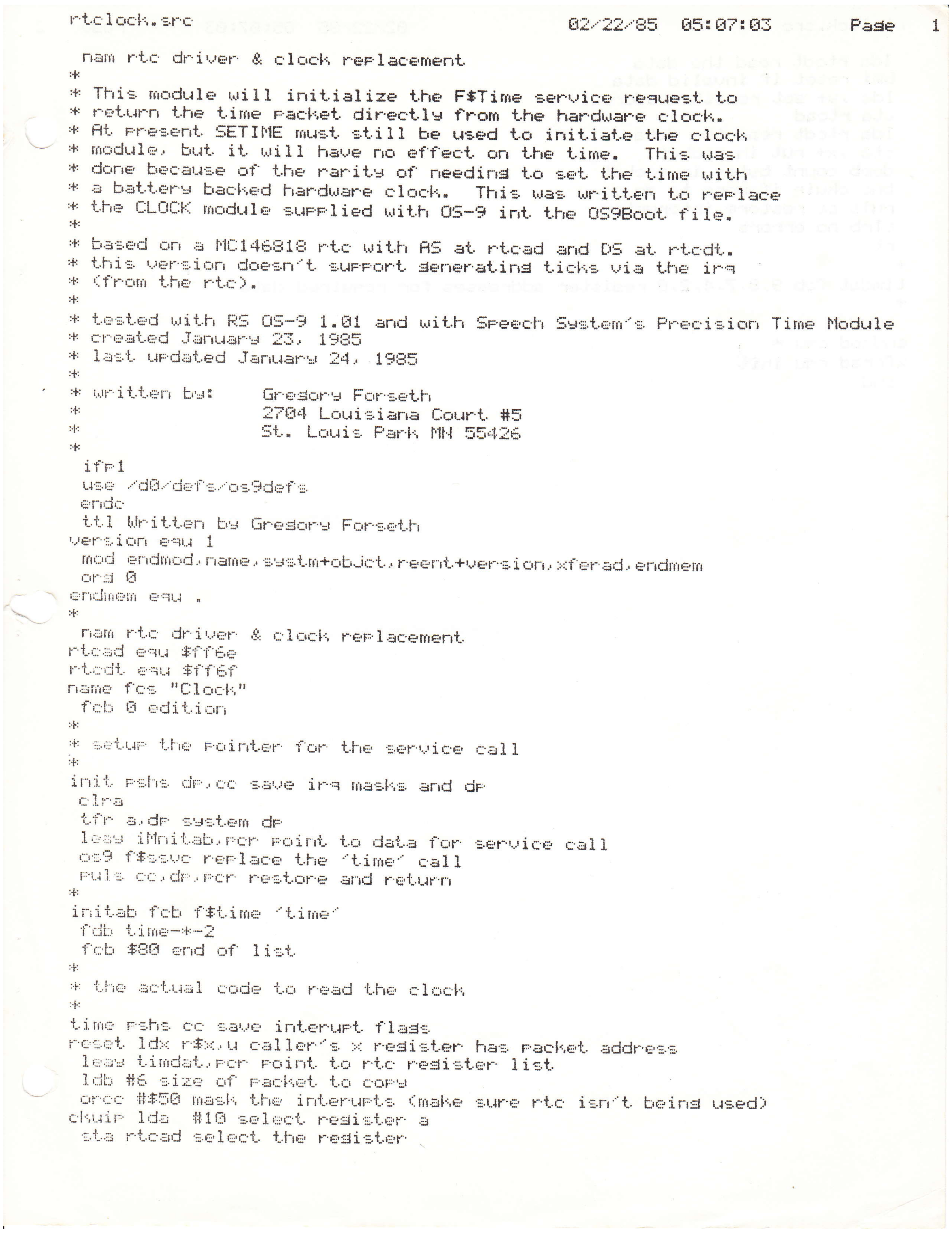
FORMAT FOR DATE/TIME IN MEMORY



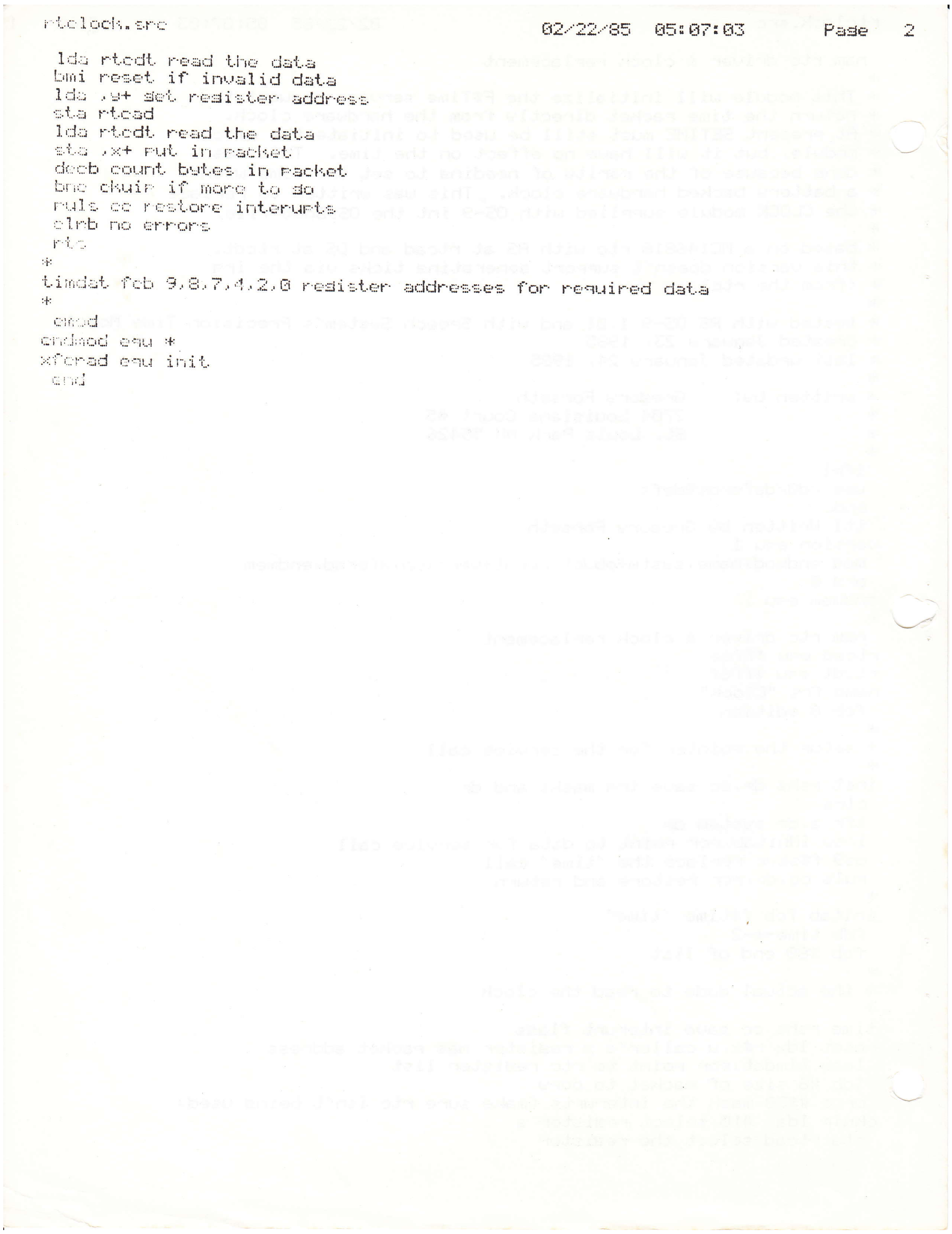
FUNCTION	BYTE (HEXDEC)		BUFFER
SECONDS	$25	(37)	9
MINUTES	$23	(35)	7
HOURS	$21	(33)	5
DOW	$1F	(31)	3
DOM	$1E	(30)	2
MONTH	$1D	(29)	1
YEAR	$1C	(28)	0
28 Byte Machine Language Program			
START	$00	00	

OS9 Support

Included on the following pages is a source listing that can be used to interface the Precision Time Module real time clock with the OS9 Operating System. SPEECH SYSTEMS supplies this listing on an AS-IS basis. SPEECH SYSTEMS assumes that you are familiar with OS9 and the related assembler programs to make use of this OS9 Specific code.



```
rtclock.src                                    02/22/85  05:07:03    Page    1
 nam rtc driver & clock replacement
*
* This module will initialize the F$Time service request to
* return the time packet directly from the hardware clock.
* At present SETIME must still be used to initiate the clock
* module, but it will have no effect on the time.  This was
* done because of the rarity of needing to set the time with
* a battery backed hardware clock.  This was written to replace
* the CLOCK module supplied with OS-9 int the OS9Boot file.
*
* based on a MC146818 rtc with AS at rtcad and DS at rtcdt.
* this version doesn't support generating ticks via the irq
* (from the rtc).
*
* tested with RS OS-9 1.01 and with Speech System's Precision Time Module
* created January 23, 1985
* last updated January 24, 1985
*
* written by:     Gregory Forseth
*                 2704 Louisiana Court #5
*                 St. Louis Park MN 55426
*
 ifp1
 use /d0/defs/os9defs
 endc
 ttl Written by Gregory Forseth
version equ 1
 mod endmod,name,systm+objct,reent+version,xferad,endmem
 org 0
endmem equ .
*
 nam rtc driver & clock replacement
rtcad equ $ff6e
rtcdt equ $ff6f
name fcs "Clock"
 fcb 0 edition
*
* setup the pointer for the service call
*
init pshs dp,cc save irq masks and dp
 clra
 tfr a,dp system dp
 leay iMnitab,pcr point to data for service call
 os9 f$ssvc replace the 'time' call
 puls cc,dp,pcr restore and return
*
initab fcb f$time 'time'
 fdb time-*-2
 fcb $80 end of list
*
* the actual code to read the clock
*
time pshs cc save interupt flags
reset ldx r$x,u caller's x register has packet address
 leay timdat,pcr point to rtc register list
 ldb #6 size of packet to copy
 orcc #$50 mask the interupts (make sure rtc isn't being used)
ckuip lda  #10 select register a
 sta rtcad select the register
```



```
rtclock.src                                          02/22/85  05:07:03      Page   2
  lda rtcdt read the data
  bmi reset if invalid data
  lda ,y+ set register address
  sta rtcad
  lda rtcdt read the data
  sta ,x+ put in packet
  decb count bytes in packet
  bne ckuir if more to do
  puls cc restore interupts
  clrb no errors
  rts
*
timdat fcb 9,8,7,4,2,0 register addresses for required data
*
 emod
endmod equ *
xferad equ init
 end
```