Electromechanical Timer Replacement

Solutions Cubed Real-Time Clock

Author: David Brobst
Solutions Cubed
Chico, CA
USA
email: solcubed@solutions

OVERVIEW

This design fragment is based upon converting an electromechanical timer idea to a PIC12CXXX 8-bit microcontroller.

DESIGN IDEA

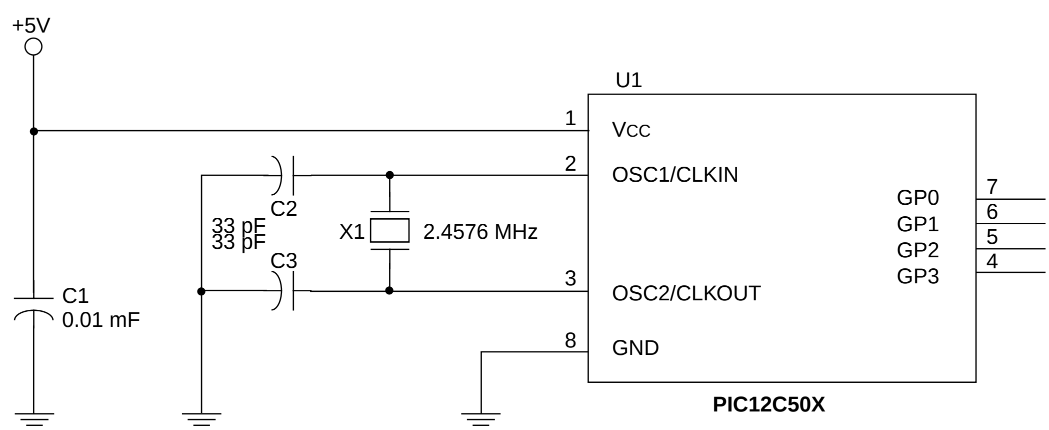
This design idea uses the PIC12C50X series of 8-pin microcontrollers to implement a medium accuracy real-time clock. There are two unique features to this design when compared to other real-time clock designs. The first is that common asynchronous communication baud rates can be easily implemented. The other is that leap year compensation is implemented in a straight forward and simple manner. Figure 1 shows the basic hardware for the design.

HARDWARE METHODOLOGY

The heart of the system is the 2.4576MHz (X1) crystal which can be found from any of the leading crystal manufacturers. The neat thing about this value is that it allows for an easy clock breakdown for asynchronous communication and allows for a fairly easy implementation of a real-time clock. As with any real-time clock, the accuracy of the crystal and the value of its load are the major factors in determining clock accuracy. In this case, X1 can be easily obtained with a 20 ppm accuracy and a 16 pF load.

By using the internal $\overline{MCLR}$ of the PIC12CXXX family, an extra input pin is made available. C1 is used for decoupling purposes.

FIGURE 1: BASIC HARDWARE FOR REAL-TIME CLOCK IMPLEMENTATION



SOFTWARE METHODOLOGY

Appendix A gives the code listing which will be discussed here.

As with almost all clock designs, a simple counter is used to keep track of the time. With a 2.4576 MHz crystal, the internal instruction cycle is 1.627604 μs, which at first glance does not seem very promising. However, there are exactly 61440 instruction cycles per 100 ms using this clock frequency. Using TMR0 with a 256 prescalar this breaks down to an overflow every 240 counts. Therefore TMR0 is preloaded with d’11’ every 100 ms so that TMR0 will overflow in exactly 100 ms.

The code knows that TMR0 has overflowed with a simple compare register TMR_OLD. If TMR0 is less than TMR_OLD then the timer has rolled over and it is time to update the real-time clock. After ten rollovers, the SECONDS register is incremented and so forth through the rest of the code.

In order to take into account the lag, when TMR0 starts counting after a write, and the prescaler being erased after a write, a timing loop is employed that implements an average error wait every time through the loop. This is the major source of error in the timekeeping process. The delay loop could be tailored to meet the individual cases of the code that the clock was implemented in, especially if the system was deterministic.

In order to not miss a rollover, the routine must be checked within 100 ms of the previous roll over.

This code keeps track all the way through years, with leap year compensation. The MONTH_TABLE routine is a simple computed `GOTO` look up table, with a special circumstance for February. Leap year occurs every four years, with the added bonus that the years it occurs on can be evenly divided by four. This means that if the YEARS register’s two least significant bits are zeros it is a leap year.

The last important bit of coding is that the YEARS register is merely a count up register, so that the year 2000 could be represented by d’100’, while 1900 would be d’00’. This is to help code get over the year 2000 hump. Before this type of counting would be a problem, it will be the year 2156. Hopefully, code and devices implemented now will not still be in service.

The code of interest is in the subroutine RTC. RTC calls MONTH_TABLE. This means that the PIC12C50X’S limited stack would be used up if RTC was used as a subroutine from the main program loop. However, it is relatively simple to put RTC into a straight line code, along with MONTH_TABLE. This way the whole thing could be in the main program loop and not impact precious stack depth, or subroutine space. It is presented in this manner to ease readability and understanding.

Further Expansion

Because of the clock frequency, common baud rates (2400, 9600, 19200) are easily obtainable and do not have the error associated with using off value clocks. Also, the speed of the clock allows for some fairly rigorous computational efforts to be realized along with an on-board time stamp.

Comparisons

This real-time clock is a good fit for applications where a moderate accuracy time, along with communication, is necessary while still meeting a low price and parts count. A 32.768 kHz solution is a better fit where accuracy is important (because of the prescalar and offset problems with the 2.4576 MHz version), or where low power consumption is vital.

Current Use

The ideas presented here have been incorporated into a product currently being offered by Solutions Cubed. It includes an alarm output along with serial communication capabilities.

RAM Used:8 bytes, 1 byte is a TEMP register

Subroutine Bytes:79

Program Bytes (as presented):110

Program Cycles (min, no roll over):9

Program Cycles (max, everything changes): 557

MICROCHIP TOOLS USED

Assembler/Compiler Version

MPLAB 3.22.02 and MPASM 1.50

APPENDIX A: SOURCE CODE

```
Appendix A:  Code Listing

MPASM 01.50 Released         MICROCLK.ASM   5-29-1997  13:01:23         PAGE  1


LOC  OBJECT CODE     LINE SOURCE TEXT
  VALUE

            00001 ;****************************************************************************
            00002 ;****************************************************************************
            00003 ;****                          SOLUTIONS CUBED                           ****
            00004 ;****               Frank Rossini, Lon Glazner, David Brobst             ****
            00005 ;****************************************************************************
            00006 ;****************************************************************************
            00007 ;
            00008 ;
            00009 ;****************************************************************************
            00010 ;****                   Solutions Cubed Real Time Clock                  ****
            00011 ;****************************************************************************
            00012 ;
            00013 ;       The purpose of this code is to develop a real time clock which
            00014 ;can interface directly and easily to a standard asynchronous communications
            00015 ;channel using the PIC12C50X chip.
            00016 ;
            00017 ;****************************************************************************
            00018 ;
            00019 ;
            00020 ;****************************************************************************
            00021 ;****************************************************************************
            00022 ;****   Define registers, constants, processor, and assembler directives ****
            00023 ;****************************************************************************
            00024 ;****************************************************************************
            00025 ;
            00026 ;Processor
            00027 ;
            00028         LIST    P=12C508                ;Processor used
            00029 ;
            00030 ;Processor defined registers and bits
            00031 ;
            00032         INCLUDE "C:\PIC\HEADERS\P12C508.INC"     ;Microchip include file
            00001         LIST
            00002 ; P12C508.INC  Standard Header File, Version 1.02    Microchip Technology, Inc.
            00105         LIST
            00033 ;
            00034 ;Program defined registers
            00035 ;
  00000007  00036 TEMP0           EQU     H'07'           ;Pseudo-WORKING registers
            00037 ;
  00000008  00038 TMR_OLD         EQU     H'08'
  00000009  00039 BIN1            EQU     H'09'           ;Time keeping registers
  0000000A  00040 SECONDS         EQU     H'0A'
  0000000B  00041 MINUTES         EQU     H'0B'
  0000000C  00042 HOURS           EQU     H'0C'
  0000000D  00043 DAYS            EQU     H'0D'
  0000000E  00044 MONTHS          EQU     H'0E'
  0000000F  00045 YEARS           EQU     H'0F'
            00046 ;
            00047 ;****************************************************************************
            00048 ;
            00049 ;
            00050 ;****************************************************************************
            00051 ;****************************************************************************
            00052 ;****                           Reset Vector                             ****
            00053 ;****************************************************************************
```

```
            ;****************************************************************************
0000                ORG     H'000'
0000 0A50           GOTO    MAIN
            ;****************************************************************************
            ;
            ;
            ;****************************************************************************
            ;****************************************************************************
            ;****                            Time Routines                          ****
            ;****************************************************************************
            ;****************************************************************************
            ;MONTH_TABLE -- Keeps track of number of days per month
            ;RTC -- Routine for real time clock
            ;****************************************************************************
            ;
            ;
            ;****************************************************************************
            ;MONTH_TABLE:  This table keeps track of the number of days in each month.
            ;It is not adjusted for leap year.  The NOP in the beginning of the table is
            ;because the first month, January, is denoted 1.  The MONTHS registers is
            ;assumed to be pre-loaded into W before this routine is called.
            ;       Called From:  TIME_INCREMENT
            ;       Modified Registers:  PCL, STATUS, TEMP0
            ;       Subroutines Called:  NONE
            ;
0001        MONTH_TABLE
0001 01E2           ADDWF   PCL,F
0002 0000           NOP
0003 0820           RETLW   H'20'                   ;d31 --- # of days in January
0004 0A0F           GOTO    CHECK_FEB               ;Leap year compensation
0005 0820           RETLW   H'20'                   ;d31 --- # of days in March
0006 081F           RETLW   H'1F'                   ;d30 --- # of days in April
0007 0820           RETLW   H'20'                   ;d31 --- # of days in May
0008 081F           RETLW   H'1F'                   ;d30 --- # of days in June
0009 0820           RETLW   H'20'                   ;d31 --- # of days in July
000A 0820           RETLW   H'20'                   ;d31 --- # of days in August
000B 081F           RETLW   H'1F'                   ;d30 --- # of days in September
000C 0820           RETLW   H'20'                   ;d31 --- # of days in October
000D 081F           RETLW   H'1F'                   ;d30 --- # of days in November
000E 0820           RETLW   H'20'                   ;d31 --- # of days in December
000F        CHECK_FEB
000F 020F           MOVF    YEARS,W                 ;Leap years are divisible by 4
0010 0027           MOVWF   TEMP0                   ;       therefore, two RRF should
0011 060F           BTFSC   YEARS,0                 ;       in the C bit
0012 081D           RETLW   H'1D'                   ;d28 --- Regular February
0013 062F           BTFSC   YEARS,1
0014 081D           RETLW   H'1D'                   ;d28 --- Regular February
0015 081E           RETLW   H'1E'                   ;d29 --- Leap year
            ;****************************************************************************
            ;
            ;
            ;****************************************************************************
            ;RTC:  This routine is used keep track of the real time of the program.
            ;       Called From:            MAIN_LOOP
            ;       Registers Used:         BIN1, DAYS, HOURS, MINUTES, MONTHS, SECONDS,
            ;                               STATUS, TEMP0, TMR_OLD, TMR0, YEARS
            ;       Subroutines Called:     MONTH_TABLE
            ;
0016        RTC
0016 0208           MOVF    TMR_OLD,W               ;Check to see if TMR0 rolled over
0017 0081           SUBWF   TMR0,W                  ;       during MORE_PROGRAM
0018 0603           BTFSC   STATUS,C                ;If C set then no roll over
0019 0A4F           GOTO    RTC_END
001A        TMR0_OFFSET
001A 0201           MOVF    TMR0,W                  ;Get offset correct
001B 0028           MOVWF   TMR_OLD
```



```
001C 0201   T0O_0   MOVF    TMR0,W                  ;Make sure TMR0 has incremented
001D 0088           SUBWF   TMR_OLD,W
001E 0643           BTFSC   STATUS,Z                ;If not equal then TMR0 has increment
001F 0A1C           GOTO    T0O_0
0020 0C52           MOVLW   H'52'                   ;Equalize TMR0 prescale error
0021 0027           MOVWF   TEMP0
0022 0000           NOP
0023 02E7   T0O_1   DECFSZ  TEMP0,F
0024 0A23           GOTO    T0O_1
0025 0C11           MOVLW   H'11'                   ;Put in offset
0026 01E1           ADDWF   TMR0,F
0027 0201           MOVF    TMR0,W                  ;Re-load so don't miss roll over
0028 0028           MOVWF   TMR_OLD
0029        TIME_INCREMENT
0029 02E9           DECFSZ  BIN1,F                  ;See if has been 1 second
002A 0A4F           GOTO    TI_END
002B 02AA           INCF    SECONDS,F               ;Increment SECONDS
002C 0C3C           MOVLW   H'3C'                   ;See if MINUTES should be incremented
002D 008A           SUBWF   SECONDS,W
002E 0743           BTFSS   STATUS,Z                ;If Z set then increment MINUTES
002F 0A4D           GOTO    TI_RESET
0030 006A           CLRF    SECONDS                 ;Reset SECONDS
0031 02AB           INCF    MINUTES,F               ;Increment MINUTES
0032 0C3C           MOVLW   H'3C'                   ;See if HOURS should be incremented
0033 008B           SUBWF   MINUTES,W
0034 0743           BTFSS   STATUS,Z                ;If Z set then increment HOURS
0035 0A4D           GOTO    TI_RESET
0036 006B           CLRF    MINUTES                 ;Reset MINUTES
0037 02AC           INCF    HOURS,F                 ;Increment HOURS
0038 0C18           MOVLW   H'18'                   ;See if DAYS should be incremented
0039 008C           SUBWF   HOURS,W
003A 0743           BTFSS   STATUS,Z                ;If Z set then increment DAYS
003B 0A4D           GOTO    TI_RESET
003C 006C           CLRF    HOURS                   ;Reset HOURS
003D 02AD           INCF    DAYS,F                  ;Increment Days
003E 020E           MOVF    MONTHS,W
003F 0901           CALL    MONTH_TABLE             ;Get number of days in month
0040 008D           SUBWF   DAYS,W
0041 0743           BTFSS   STATUS,Z                ;If Z set then month over
0042 0A4D           GOTO    TI_RESET
0043 0C01           MOVLW   H'01'                   ;Reset DAYS
0044 002D           MOVWF   DAYS
0045 02AE           INCF    MONTHS,F                ;Increment MONTHS
0046 0C0D           MOVLW   H'0D'                   ;See if at end of year
0047 008E           SUBWF   MONTHS,W
0048 0743           BTFSS   STATUS,Z                ;If Z set then at end of year
0049 0A4D           GOTO    TI_RESET
004A 0C01           MOVLW   H'01'                   ;Reset MONTHS
004B 002E           MOVWF   MONTHS
004C 02AF           INCF    YEARS,F
004D        TI_RESET
004D 0C0A           MOVLW   H'0A'                   ;Reset the number of times for 100mS
004E 0029           MOVWF   BIN1                    ;       overflow
004F        TI_END
004F 0800   RTC_END RETLW   H'00'
            ;***************************************************************************
            ;
            ;
            ;***************************************************************************
            ;***************************************************************************
            ;***************************************************************************
            ;****                          Main Program                             ****
            ;***************************************************************************
            ;***************************************************************************
            ;***************************************************************************
            ;
```



```
           00186 ;
           00187 ;****************************************************************************
0050       00188 MAIN
           00189 ;
0050       00190 CLEAR_REGISTERS
0050 0067  00191         CLRF    TEMP0                   ;Clear first RAM location for use
0051 0C18  00192         MOVLW   H'18'                   ;Number of registers to clear
0052 0027  00193         MOVWF   TEMP0
0053 0C08  00194         MOVLW   H'08'                   ;Start of RAM clearing
0054 0024  00195         MOVWF   FSR
0055       00196 CLEAR_LOOP
0055 0060  00197         CLRF    INDF                    ;Clear register pointed to
0056 02A4  00198         INCF    FSR,F                   ;Go to next RAM location to clear
0057 02E7  00199         DECFSZ  TEMP0,F                 ;Check to see if all clearing done
0058 0A55  00200         GOTO    CLEAR_LOOP
0059       00201 PORT_SETUP
0059 0C3B  00202         MOVLW   H'3B'                   ;0011 1011
005A 0026  00203         MOVWF   GPIO
005B 0C3B  00204         MOVLW   H'3B'                   ;0011 1011
005C 0006  00205         TRIS    GPIO
005D       00206 OPTION_SETUP
005D 0CC7  00207         MOVLW   H'C7'                   ;1100 0111 -- Wake up disabled, weak
005E 0002  00208         OPTION                          ;       PUs disabled, internal TMR0,
005F       00209 TIME_SETUP                              ;       1:256 prescalar to TMR0
005F 006A  00210         CLRF    SECONDS                 ;Set a beginning time:  12:00AM,
0060 006B  00211         CLRF    MINUTES                 ;       January, 1 1996
0061 006C  00212         CLRF    HOURS
0062 0C01  00213         MOVLW   H'01'
0063 002D  00214         MOVWF   DAYS
0064 002E  00215         MOVWF   MONTHS
0065 0C60  00216         MOVLW   H'60'                   ;d96
0066 002F  00217         MOVWF   YEARS
0067 0C0A  00218         MOVLW   H'0A'                   ;Overflow for 100mS register
0068 0029  00219         MOVWF   BIN1
0069 0C11  00220         MOVLW   H'11'                   ;Set up for 100mS overflow
006A 0021  00221         MOVWF   TMR0                    ;Set up for first find
006B 0028  00222         MOVWF   TMR_OLD
006C       00223 MAIN_LOOP
006C 0916  00224         CALL    RTC                     ;Time Routines
006D 0A6C  00225         GOTO    MAIN_LOOP
           00226 ;****************************************************************************
           00227 ;
           00228 ;End of code indicator
           00229 ;
           00230         END
MEMORY USAGE MAP ('X' = Used,  '-' = Unused)

0000 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0040 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXX-- ----------------

All other memory blocks unused.

Program Memory Words Used:   110
Program Memory Words Free:   402


Errors   :     0
Warnings :     0 reported,     0 suppressed
Messages :     0 reported,     0 suppressed
```

NOTES:

MICROCHIP

WORLDWIDE SALES & SERVICE

AMERICAS

Corporate Office
Microchip Technology Inc.
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 602-786-7200 Fax: 602-786-7277
Technical Support: 602 786-7627
Web: http://www.microchip.com

Atlanta
Microchip Technology Inc.
500 Sugar Mill Road, Suite 200B
Atlanta, GA 30350
Tel: 770-640-0034 Fax: 770-640-0307

Boston
Microchip Technology Inc.
5 Mount Royal Avenue
Marlborough, MA 01752
Tel: 508-480-9990 Fax: 508-480-8575

Chicago
Microchip Technology Inc.
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 630-285-0071 Fax: 630-285-0075

Dallas
Microchip Technology Inc.
14651 Dallas Parkway, Suite 816
Dallas, TX 75240-8809
Tel: 972-991-7177 Fax: 972-991-8588

Dayton
Microchip Technology Inc.
Two Prestige Place, Suite 150
Miamisburg, OH 45342
Tel: 937-291-1654 Fax: 937-291-9175

Los Angeles
Microchip Technology Inc.
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 714-263-1888 Fax: 714-263-1338

New York
Microchip Technology Inc.
150 Motor Parkway, Suite 416
Hauppauge, NY 11788
Tel: 516-273-5305 Fax: 516-273-5335

San Jose
Microchip Technology Inc.
2107 North First Street, Suite 590
San Jose, CA 95131
Tel: 408-436-7950 Fax: 408-436-7955

Toronto
Microchip Technology Inc.
5925 Airport Road, Suite 200
Mississauga, Ontario L4V 1W1, Canada
Tel: 905-405-6279 Fax: 905-405-6253

ASIA/PACIFIC

Hong Kong
Microchip Asia Pacific
RM 3801B, Tower Two
Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2-401-1200 Fax: 852-2-401-3431

India
Microchip Technology Inc.
India Liaison Office
No. 6, Legacy, Convent Road
Bangalore 560 025, India
Tel: 91-80-229-4036 Fax: 91-80-559-9840

Korea
Microchip Technology Korea
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea
Tel: 82-2-554-7200 Fax: 82-2-558-5934

Shanghai
Microchip Technology
RM 406 Shanghai Golden Bridge Bldg.
2077 Yan'an Road West, Hong Qiao District
Shanghai, PRC 200335
Tel: 86-21-6275-5700
Fax: 86 21-6275-5060

Singapore
Microchip Technology Taiwan
Singapore Branch
200 Middle Road
#10-03 Prime Centre
Singapore 188980
Tel: 65-334-8870 Fax: 65-334-8850

Taiwan, R.O.C
Microchip Technology Taiwan
10F-1C 207
Tung Hua North Road
Taipei, Taiwan, ROC
Tel: 886 2-717-7175 Fax: 886-2-545-0139

EUROPE

United Kingdom
Arizona Microchip Technology Ltd.
Unit 6, The Courtyard
Meadow Bank, Furlong Road
Bourne End, Buckinghamshire SL8 5AJ
Tel: 44-1628-851077 Fax: 44-1628-850259

France
Arizona Microchip Technology SARL
Zone Industrielle de la Bonde
2 Rue du Buisson aux Fraises
91300 Massy, France
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

Germany
Arizona Microchip Technology GmbH
Gustav-Heinemann-Ring 125
D-81739 Müchen, Germany
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

Italy
Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Taurus 1 V. Le Colleoni 1
20041 Agrate Brianza
Milan, Italy
Tel: 39-39-6899939 Fax: 39-39-6899883

JAPAN
Microchip Technology Intl. Inc.
Benex S-1 6F
3-18-20, Shinyokohama
Kohoku-Ku, Yokohama-shi
Kanagawa 222 Japan
Tel: 81-45-471- 6166 Fax: 81-45-471-6122

7/29/97