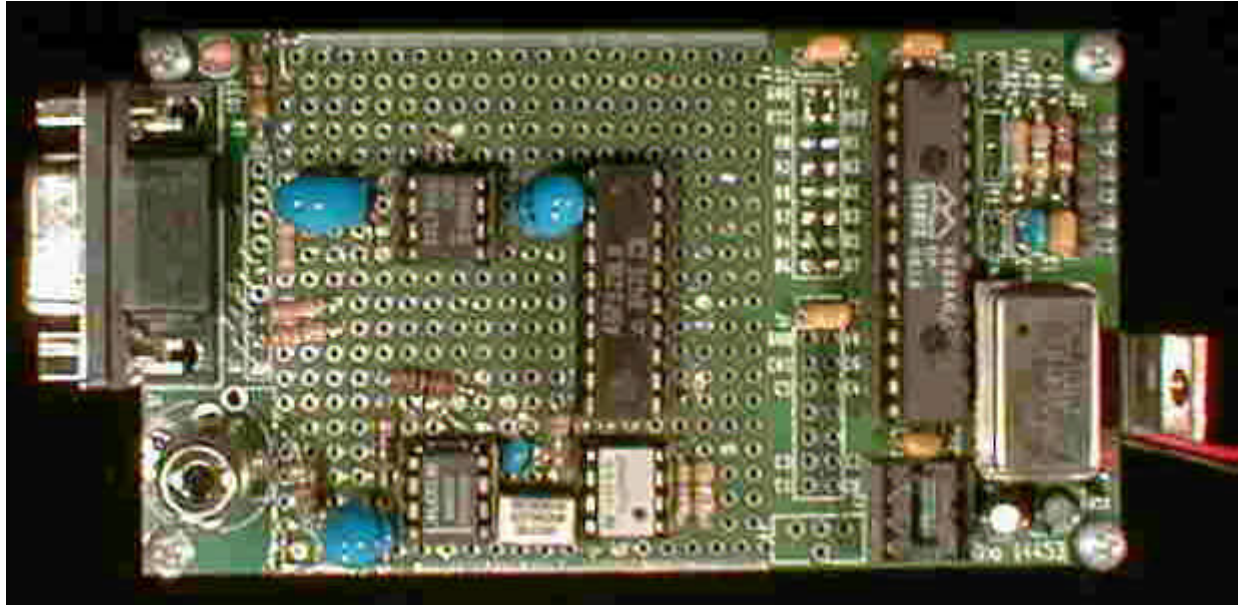
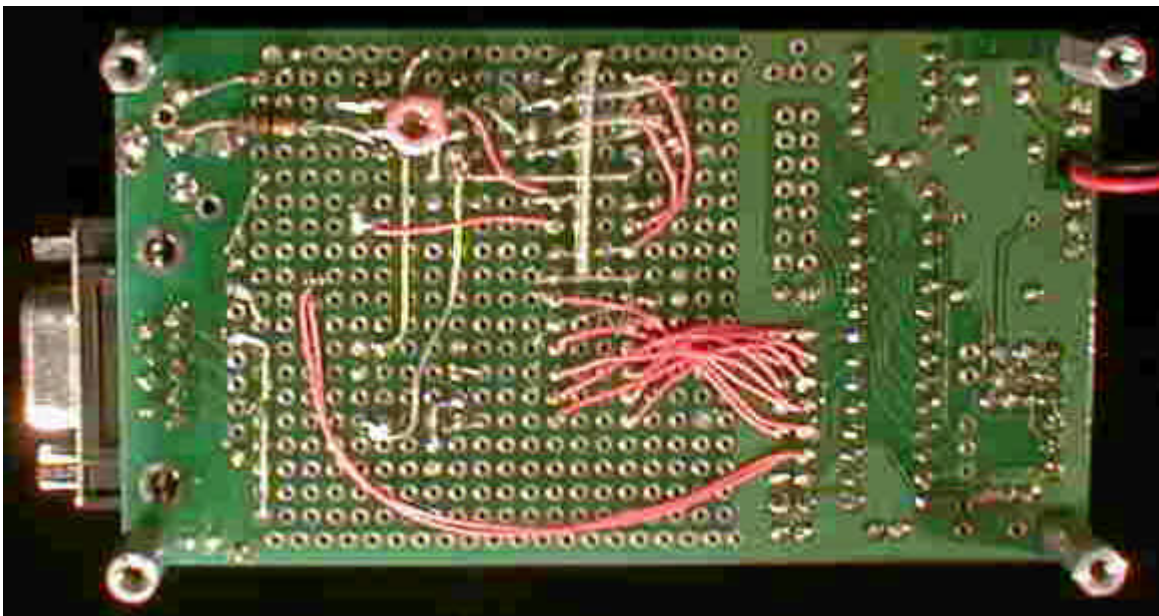


Rho Enterprises			
Box 33			
4100 W. Colfax Ave.			
Denver, CO 80204			
www.rhoent.com			
(720) 359-1467			
NCO by DDS			
Sheet 1	of 1	Rev 1	Size A
Drawn by Richard Ottosen		Date 6/23/99	
Drawing No. 14615			
Copyright 1999 Rho Enterprises			

Example using Rho Enterprises SX-28 Proto Board to build a Direct Digital Synthesis signal generator.



Component side of the DDS prototype



Circuit side of the DDS prototype

```

00001 ;SXDDS.asm      8/9/99      Richard Ottosen
00002 ;
00003 ;Example of a Numerically Controlled Oscillator by Direct Digital
Synthesis
00004 ; using a Scenix microcontroller
00005 ;
1998. 00006 ;Based on an article by Tom Napier in Circuit Cellar INK #99, October
00007 ;
00008
00009 ;The hardware:
00010 ; The D/A used is an Analog Devices AD7528 dual D/A converter.
allows 00011 ; The first D/A drives the reference input of the output D/A. This
00012 ; setting the output amplitude.
Technology LT1054 00013 ; The output amplifier is powered by -4.5 volts from a Linear
00014 ; charge pump. The output amplifier should be a high speed type with
rail to 00015 ; rail swing on the output. A Maxim MAX410 will sort of work here.
00016 ;
00017
00018
00019 ;This code is assembled by the Microchip Inc. cross-assembler MPASM.
00020
00021          RADIX          DEC
00022          ERRORLEVEL    -305
00023          INCLUDE       SXDefs.inc
00140          LIST
00141          PROCESSOR     16C57      ; "SX28AC"
00333          LIST
1010 06FA 00334          DATA    _FUSE          ;configuration bits (TURBO, SYNC,
OPTIONX, etc.)
1011 040E 00335          DATA    _FUSEX       ; (PINS, CARRYX, BOR40, BANKS, PAGES)
00339          LIST
00024
00025 ;***          EXPAND          ;Must expand for SXSim to work
is desired 00026          NOEXPAND         ;Should not expand if small printed listing
00027          LIST          ST=OFF, MM=OFF ;Keep listing small
00028
02FAF080 00029 MIPS      EQU      50000000      ;The SX instruction rate
00030
0047E903 00031 DEVICE    EQU      PINS28+PAGES4+BANKS8+OSCHS+BOR40+TURBO+STACKX+CARRYX
00032
0000000A 00033 Ver      EQU      10          ;Version # for outputing
ID location 00034 ID      'V','e','r',' ',Ver/10,'.',Ver-(Ver/10*10),' ' ;Version # in
00035
00036
00037 ;
00038 ;Where things are
00039 ;
00000008 00040 RAMBase EQU      08h          ;Start of RAM
00000020 00041 RAMTop  EQU      020h        ;Last RAM location +1
00000800 00042 MemSize EQU      2048        ;Program memory size in words
00043
00044
00045
00046 ;For instruction destination argument
00047 ;W      EQU      0
00048 ;F      EQU      1
00049

```

```

00000000      00050 Indir   EQU    00h      ;Used for indirects thru FSR
00000001      00051 RTCC   EQU    01h      ;Real Time Clock/Counter
00000002      00052 PC     EQU    02h      ;Program Counter Low
00000004      00053 FSR    EQU    04h      ;File Select Register (index
register)
00000005      00054 PortA  EQU    05h      ;I/O Port A
00000006      00055 PortB  EQU    06h      ;I/O Port B
00056
00000003      00057 Status EQU    03h      ;Status register:
00058 #DEFINE CF      Status,0      ; Bit 0 = Carry Flag
00059 #DEFINE ZF      Status,2      ; Bit 2 = Zero Flag
00060
00061 ;
00062 ;Some ASCII codes
00063 ;
0000000D      00064 CR     EQU    0Dh      ;Carriage Return
0000001B      00065 Esc    EQU    1Bh      ;Escape
00066
00067
00068 ;----- Constants -----
-----
00002580      00069
00070 Speed   EQU    9600      ;Set speed: 1200 to 19200 Baud
00071
00072
00073 ;-----
-----
00074 ;
00075 ;Start of RAM
00076 ;
0008          00077          ORG    RAMBase
00078
0008          00079 Zero    RES    1      ;Holds the value 0 for two's-
complementing W
0009          00080 Five    RES    1      ;Holds the value 5 for "ADDLW 5"
000A          00081 Fifteen RES    1      ;Holds the value 15 for "SUBLW 15"
00082
000B          00083 Count   RES    1      ;General purpose counter
000C          00084 DlyCnt  RES    1      ;General purpose delay counter
000D          00085 DelMCnt RES    1      ;Cycle counter for fast delays
000E          00086 BitCnt  RES    1      ;Bit counter for serial output
000F          00087 SerReg  RES    1      ;Data register for serial output
0010          00088 CmdPtr  RES    1      ;Pointer for command scanner table
00089
0011          00090 Freq    RES    3      ;Holds the increment value for the
desired
0014          00091 Phase   RES    3      ;Phase of the generated waveform
0017          00092 Waveform RES    1      ;Address of start of a waveform table
00093
0018          00094 Digit   RES    1      ;Holds digit for decimal input
0019          00095 RegA    RES    3      ;24 bit accumulator
00096
001C          00097 Temp    RES    3      ;Very temporary storage
00098
00099 ;-----
-----
00100
07FF          00101          ORG    MemSize-1      ;Reset vector
Message[306]: Crossing page boundary -- ensure page bits are set.
07FF 0A01     00102          GOTO   Reset      ;Start of program
00103
00104
0000          00105          ORG    0000h      ;Start of ROM
00106 ;
00107 ;Interrupt handler
00108 ;

```

```

0000          00109 IntHan
          00110          RETI          ;End of "IntHan
          00111
          00112
          00113 Reset   FGOTO   Start
          00114
          00115
0003 0022          00116 WaveTbl MOVWF   PC
          00117 ;
0004 0880 0883 0886 00118 SineTbl DT      080h,083h,086h,089h,08Ch,090h,093h,096h
      0889 088C 0890
      0893 0896
000C 0899 089C 089F 00119          DT      099h,09Ch,09Fh,0A2h,0A5h,0A8h,0ABh,0AEh
      08A2 08A5 08A8
      08AB 08AE
0014 08B1 08B3 08B6 00120          DT      0B1h,0B3h,0B6h,0B9h,0BCh,0BFh,0C1h,0C4h
      08B9 08BC 08BF
      08C1 08C4
001C 08C7 08C9 08CC 00121          DT      0C7h,0C9h,0CCh,0CEh,0D1h,0D3h,0D5h,0D8h
      08CE 08D1 08D3
      08D5 08D8
0024 08DA 08DC 08DE 00122          DT      0DAh,0DCh,0DEh,0E0h,0E2h,0E4h,0E6h,0E8h
      08E0 08E2 08E4
      08E6 08E8
002C 08EA 08EB 08ED 00123          DT      0EAh,0EBh,0EDh,0EFh,0F0h,0F1h,0F3h,0F4h
      08EF 08F0 08F1
      08F3 08F4
0034 08F5 08F6 08F8 00124          DT      0F5h,0F6h,0F8h,0F9h,0FAh,0FAh,0FBh,0FCh
      08F9 08FA 08FA
      08FB 08FC
003C 08FD 08FD 08FE 00125          DT      0FDh,0FDh,0FEh,0FEh,0FEh,0FFh,0FFh,0FFh
      08FE 08FE 08FF
      08FF 08FF
0044 08FF          00126          DT      0FFh
          00127 ;
0045 0880 0882 0884 00128 TritTbl DT      080h,082h,084h,086h,088h,08Ah,08Ch,08Eh
      0886 0888 088A
      088C 088E
004D 0890 0892 0894 00129          DT      090h,092h,094h,096h,098h,09Ah,09Ch,09Eh
      0896 0898 089A
      089C 089E
0055 08A0 08A2 08A4 00130          DT      0A0h,0A2h,0A4h,0A6h,0A8h,0AAh,0ACh,0AEh
      08A6 08A8 08AA
      08AC 08AE
005D 08B0 08B2 08B4 00131          DT      0B0h,0B2h,0B4h,0B6h,0B8h,0BAh,0BCh,0BEh
      08B6 08B8 08BA
      08BC 08BE
0065 08C0 08C2 08C4 00132          DT      0C0h,0C2h,0C4h,0C6h,0C8h,0CAh,0CCh,0CEh
      08C6 08C8 08CA
      08CC 08CE
006D 08D0 08D2 08D4 00133          DT      0D0h,0D2h,0D4h,0D6h,0D8h,0DAh,0DCh,0DEh
      08D6 08D8 08DA
      08DC 08DE
0075 08E0 08E2 08E4 00134          DT      0E0h,0E2h,0E4h,0E6h,0E8h,0EAh,0ECh,0EEh
      08E6 08E8 08EA
      08EC 08EE
007D 08F0 08F2 08F4 00135          DT      0F0h,0F2h,0F4h,0F6h,0F8h,0FAh,0FCh,0FEh
      08F6 08F8 08FA
      08FC 08FE
0085 08FF          00136          DT      0FFh
          00137 ;
0086 08FF 08FF 08FF 00138 SqrTbl  DT      0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
      08FF 08FF 08FF
      08FF 08FF
008E 08FF 08FF 08FF 00139          DT      0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
      08FF 08FF 08FF

```

```

0096 08FF 08FF 08FF 08FF 00140      DT      0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
08FF 08FF 08FF
009E 08FF 08FF 08FF 08FF 00141      DT      0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
08FF 08FF 08FF
00A6 08FF 08FF 08FF 08FF 00142      DT      0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
08FF 08FF 08FF
00AE 08FF 08FF 08FF 08FF 00143      DT      0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
08FF 08FF 08FF
00B6 08FF 08FF 08FF 08FF 00144      DT      0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
08FF 08FF 08FF
00BE 08FF 08FF 08FF 08FF 00145      DT      0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh,0FFh
08FF 08FF 08FF
00C6 08FF                                00146      DT      0FFh
                                00147
                                00148
                                00149 ;
                                00150 ;Generate the waveform
                                00151 ;
                                00152 #DEFINE RxD      PortA,0      ; Input: Receiver for serial data
                                00153 ;
00C7 0403                                00154 GenWave BCF      CF      ;
1 Cycles
00C8 0211                                00155      MOVF      Freq,W      ;Get the current frequency
1
00C9 01F4                                00156      ADDWF      Phase      ; and increment the phase
1
00CA 0212                                00157      MOVF      Freq+1,W      ;
1
00CB 01F5                                00158      ADDWF      Phase+1      ;
1
00CC 0213                                00159      MOVF      Freq+2,W      ;
1
00CD 01F6                                00160      ADDWF      Phase+2      ;
1
                                00161
00CE 0216                                00162      MOVF      Phase+2,W      ;
1
00CF 0403                                00163      BCF      CF      ;
1
00D0 06D6                                00164      BTFSC      Phase+2,6      ;Is it the 2nd half of a half cycle?
1
00D1 0088                                00165      SUBWF      Zero,W      ; then complement to effect a
1
                                00166      ; subtract from 64
00D2 0E7F                                00167      ANDLW      07Fh      ;Reduce to 6 bits for 90 degrees
1
00D3 01D7                                00168      ADDWF      Waveform,W      ;
1
00D4 0903                                00169      CALL      WaveTbl      ; and get amplitude
9
                                00170
00D5 0503                                00171      BSF      CF      ;
1
00D6 07F6                                00172      BTFSS      Phase+2,7      ;Is amplitude supposed to be
negative? 1
00D7 0088                                00173      SUBWF      Zero,W      ;Change polarity of sine amplitude
1
                                00174
00D8 0026                                00175      MOVWF      PortB      ;Output to DAC

```

```

1
00D9 0705      00176      BTFSS   RxD          ;Test for a start bit
1
00DA 0AC7      00177      GOTO    GenWave     ;
3
00178      RETP          ; and get new setting
0
00179          ;
=30 Cycles
00180
00181
00182 ;-----
-----
0200      00183      ORG     200h        ;Next Page
00184
00185 ;
00186 ;Table of commands
00187 ;
0200 0841      00188 CmdTbl  RETLW   'A'          ;Set the amplitude?
0201 0ACE      00189      GOTO    InpAmp
0202 0846      00190      RETLW   'F'          ;Set the frequency?
0203 0AE7      00191      GOTO    InpFreq
0204 0853      00192      RETLW   'S'          ;Set the sine waveform?
0205 0AF1      00193      GOTO    SetSine
0206 0854      00194      RETLW   'T'          ;Set the triangle waveform?
0207 0AF4      00195      GOTO    SetTri
0208 0850      00196      RETLW   'P'          ;Set the pulse (square) waveform?
0209 0AF7      00197      GOTO    SetSqr
020A 0861      00198      RETLW   'a'          ;Set the amplitude?
020B 0ACE      00199      GOTO    InpAmp
020C 0866      00200      RETLW   'f'          ;Set the frequency?
020D 0AE7      00201      GOTO    InpFreq
020E 0873      00202      RETLW   's'          ;Set the sine waveform?
020F 0AF1      00203      GOTO    SetSine
0210 0874      00204      RETLW   't'          ;Set the triangle waveform?
0211 0AF4      00205      GOTO    SetTri
0212 0870      00206      RETLW   'p'          ;Set the pulse (square) waveform?
0213 0AF7      00207      GOTO    SetSqr
0214 082A      00208      RETLW   '*'          ;Shift code for keypad command?
0215 0AFA      00209      GOTO    ShiftCmd
0216 081B      00210      RETLW   Esc          ;Shift code for keypad command?
0217 0AFA      00211      GOTO    ShiftCmd
0218 0800      00212      RETLW   0            ;Table terminator
0219 0800      00213      RETURN
00214
021A      00215 ShiftTbl
021A 0834      00216      RETLW   '4'          ;Set the amplitude?
021B 0ACE      00217      GOTO    InpAmp
021C 0836      00218      RETLW   '6'          ;Set the frequency?
021D 0AE7      00219      GOTO    InpFreq
021E 0831      00220      RETLW   '1'          ;Set the sine waveform?
021F 0AF1      00221      GOTO    SetSine
0220 0832      00222      RETLW   '2'          ;Set the triangle waveform?
0221 0AF4      00223      GOTO    SetTri
0222 0833      00224      RETLW   '3'          ;Set the pulse (square) waveform?
0223 0AF7      00225      GOTO    SetSqr
0224 0800      00226      RETLW   0            ;Table terminator
0225 0800      00227      RETURN
00228
00229
00230 ;
00231 ;General purpose command scanner.
00232 ; Enter with W-reg holding the command character and
00233 ; "CmdPtr" holding the address of the first byte of table to be
scanned.
00234 ; The table must be terminated with a null "command" =0 which has

```

its own

```
00235 ; address entry to handle the "command not found" condition.
00236 ;
0226 003C 00237 CmdScan MOVWF Temp ;Save the command
0227 0210 00238 _Cmd10 MOVF CmdPtr,W
0228 0933 00239 CALL _ScanGo
0229 0EFF 00240 ANDLW 0FFh ;Command table terminator?
022A 0643 00241 BTFSC ZF
022B 0A32 00242 GOTO _Cmd20 ; Yes, then do "no command found" and
return
022C 019C 00243 XORWF Temp,W ;Does table entry match command?
022D 0643 00244 BTFSC ZF
022E 0A32 00245 GOTO _Cmd20 ; Yes, then do the command found
022F 02B0 00246 INCF CmdPtr ;Point to next command in table
0230 02B0 00247 INCF CmdPtr
0231 0A27 00248 GOTO _Cmd10
00249
0232 0290 00250 _Cmd20 INCF CmdPtr,W ;Execute the command and return from
there
0233 0022 00251 _ScanGo MOVWF PC ;Get command from table
00252
00253
00254 include DELM.ISX ;Macro for cycle delays
00001 ;8/8/99 Richard Ottosen
00002
00003 ;
00004 ;This routine is for the SX parts in Turbo mode only. It does not
matter if the
00005 ; Carry fuse is set or clear.
00006 ;
00007 ;Macro to delay for M number of cycles from 0 through 65535.
00008 ; The macro includes paging for long calls.
00009 ;
00010 ;Uses the routine "DelW" to do the short delays and uses the variable
"DelMCnt"
00011 ; as well for long delays.
00012 ;
00013 DelM MACRO M
00014 IF LOW(M) ==0 ;No delay at all
00015 ENDIF
00016 IF LOW(M) ==1
00017 NOP ;Delay 1 cycle inline
00018 ENDIF
00019 IF LOW(M) ==2
00020 NOP ;Delay 2 cycles inline
00021 NOP
00022 ENDIF
00023 IF LOW(M) ==3
00024 GOTO $+1 ;Delay 3 cycles inline
00025 ENDIF
00026 IF LOW(M) ==4
00027 GOTO $+1 ;Delay 4 cycles inline
00028 NOP
00029 ENDIF
00030 IF LOW(M) ==5
00031 GOTO $+1 ;Delay 5 cycles inline
00032 NOP
00033 NOP
00034 ENDIF
00035 IF LOW(M) ==6
00036 GOTO $+1 ;Delay 6 cycles inline
00037 GOTO $+1
00038 ENDIF
00039 IF LOW(M) ==7
00040 ERRORLEVEL -306
00041 PAGEX Delay6>>9
```

```

00042      CALL Delay6      ;Delay 7 cycles
00043      ERRORLEVEL      +306
00044      ENDIF
00045      IF      LOW(M) ==8
00046      ERRORLEVEL      -306
00047      PAGEX Delay7>>9
00048      CALL Delay7      ;Delay 8 cycles
00049      ERRORLEVEL      +306
00050      ENDIF
00051      IF      LOW(M) ==9
00052      ERRORLEVEL      -306
00053      PAGEX Delay8>>9
00054      CALL Delay8      ;Delay 9 cycles
00055      ERRORLEVEL      +306
00056      ENDIF
00057      IF      LOW(M) ==10
00058      ERRORLEVEL      -306
00059      PAGEX Delay9>>9
00060      CALL Delay9      ;Delay 10 cycles
00061      ERRORLEVEL      +306
00062      ENDIF
00063      IF      LOW(M) ==11
00064      ERRORLEVEL      -306
00065      PAGEX Delay10>>9
00066      CALL Delay10     ;Delay 11 cycles
00067      ERRORLEVEL      +306
00068      ENDIF
00069      IF      LOW(M) ==12
00070      ERRORLEVEL      -306
00071      PAGEX Delay11>>9
00072      CALL Delay11     ;Delay 12 cycles
00073      ERRORLEVEL      +306
00074      ENDIF
00075      IF      LOW(M) ==13
00076      ERRORLEVEL      -306
00077      PAGEX Delay12>>9
00078      CALL Delay12     ;Delay 13 cycles
00079      ERRORLEVEL      +306
00080      ENDIF
00081      IF      LOW(M) ==14
00082      ERRORLEVEL      -306
00083      PAGEX Delay13>>9
00084      CALL Delay13     ;Delay 14 cycles
00085      ERRORLEVEL      +306
00086      ENDIF
00087      IF      LOW(M) ==15
00088      ERRORLEVEL      -306
00089      PAGEX Delay14>>9
00090      CALL Delay14     ;Delay 15 cycles
00091      ERRORLEVEL      +306
00092      ENDIF
00093      IF      LOW(M) ==16
00094      ERRORLEVEL      -306
00095      PAGEX Delay15>>9
00096      CALL Delay15     ;Delay 16 cycles
00097      ERRORLEVEL      +306
00098      ENDIF
00099
00100      IF      LOW(M) >16
00101      MOVLW LOW(M-1)
00102      ERRORLEVEL      -306
00103      PAGEX DelW>>9
00104      CALL DelW      ;Delay for 17 thru 255 cycles
00105      ERRORLEVEL      +306
00106      ENDIF
00107

```

```

00108
00109          IF      HIGH(M) !=0
00110          LOCAL Loop
00111          MOVLW HIGH(M)          ;Delay more for greater than 255
cycles
00112          MOVWF DelMCnt
00113 Loop      MOVLW 251
00114          ERRORLEVEL  -306
00115          PAGEX DelW>>9
00116          CALL DelW
00117          ERRORLEVEL  +306
00118          DECFSZ      DelMCnt
00119          GOTO Loop
00120          ENDIF
00121          ENDM
00122
00123
00124
00125 ; Eric Smith 7/8/96      Hacked by Richard Ottosen 8/8/99
00126
00127 ;-----
-----
00128 ;This version of DelW is for the Scenix parts in Turbo mode. It does
not matter
00129 ;if the Carry fuse is set or clear.
00130 ; DelW delays W cycles, including call, return, and one cycle for the
00131 ; MOVLW instruction to set up the count in W
00132 ; range is 16..255
00133 ;
00134 ; For example, the sequence:
00135 ;   MOVLW 17
00136 ;   CALL DelW
00137 ; will take 17 cycles to execute
00138 ;-----
-----
00139
00140 ; W value on entry:          16   17   18   19   20   21   22
23
00141 ; Caller's instructions:    ---   ---   ---   ---   ---   ---   ---
--  -----
00142 ;   MOVLW n                  0     0     0     0     0     0     0
0
00143 ;   CALL DelW                1     1     1     1     1     1     1
1
00144
0234 0503 00145 DelW   BSF      CF          ; 4     4     4     4     4     4     4
4
0235 008A 00146          SUBWF  Fifteen,W    ; 5     5     5     5     5     5     5
5
0236 0403 00147          BCF      CF          ; 6     6     6     6     6     6     6
6
0237 01C9 00148 DelWLp  ADDWF  Five,W      ; 7     7     7     7     7     7 13   7
12  7 12
0238 0703 00149          BTFSS  CF          ; 8     8     8     8     8     8 14
13  13
0239 0A37 00150          GOTO   DelWLp      ;          9     9
9
023A 01E2 00151          ADDWF  PC          ; 10    10    10    10    10    15
15  15
023B 0000 00152 Delay11 NOP          ;          13   14
023C 0000 00153 Delay10 NOP          ;          13   14
023D 0000 00154 Delay9  NOP          ;          13   14   15
023E 0000 00155 Delay8  NOP          ;          13   14   15   16
18
023F 0000 00156 Delay7  NOP          ;          13   14   15   16   17
19

```

```

19      20      00157 Delay6  RETP          ; 13  14  15  16  17  18
22      23      00158          ; 16  17  18  19  20  21
0241 0000      00159
0242 0000      00160 Delay15 NOP
0243 0000      00161 Delay14 NOP
0244 0A3D      00162 Delay13 NOP
00163 Delay12 GOTO    Delay9
00255
00256 ;
00257 ;Delay 250ns for DAC set up and hold times
00258 ;
0245      00259 Dly250ns
00260          DelM    MIPS/4000000
00261          RETP
00262
00263
00264 ;
00265 ;Initialize the hardware of the SX
00266 ;
0248 0004      00267 InitHW  CLRWDT          ;Reset watchdog timer and prescale
0249 0064      00268          CLRf    FSR
024A 0CFF      00269          MOVLW  11111111b
024B 0002      00270          OPTION
00271          ; bit 0,1,2 = Prescaler divider.
                ; WDT: /1 to /128, Timer0: /2 to
/256
00272          ; bit 3 = Prescaler assign: 1 = WDT
00273          ; bit 4 = RTCC edge: 1 = high to low
00274          ; bit 5 = RTCC source: 1 = external
00275          ; bit 6 = RTCC interrupt: 1 =
disabled
00276          ; bit 7 = Remap W-reg: 1 = Read RTCC
00277
0245 registers 00278          MODE    0Fh          ;Point the mode register to TRIS
00279
00280 ;PortA EQU    5          ;I/O PortA
00281 ;Note: "RxD" is defined elsewhere to prevent a forward reference
00282 ; #DEFINE    RxD    PortA,0          ; Input: Receiver for serial
data
00283 #DEFINE TxD    PortA,1          ; Output: Transmitter for serial data
00284 #DEFINE DACWrt PortA,2          ; Output: Low writes to the selected
DAC
00285 #DEFINE DACSel PortA,3          ; Output: Low selects the Amplitude
DAC
00286          ; High selects the output DAC
00287
00000001      00288 PADir  EQU    00000001b
024D 0C01      00289          MOVLW  PADir          ;Initialize Port A directions:
024E 0005      00290          TRIS   PortA
00291
00000000      00292 InitPA  EQU    00000000b          ;Initialize Port A outputs:
024F 0C00      00293          MOVLW  InitPA
0250 0025      00294          MOVWF  PortA
00295
00296 ;PortB EQU    6          ;I/O PortB
00297 #DEFINE nupb0  PortB,0          ; Bit 0 = Output:
00298 #DEFINE nupb1  PortB,1          ; Bit 1 = Output:
00299 #DEFINE nupb2  PortB,2          ; Bit 2 = Output:
00300 #DEFINE nupb3  PortB,3          ; Bit 3 = Output:
00301 #DEFINE nupb4  PortB,4          ; Bit 4 = Output:
00302 #DEFINE nupb5  PortB,5          ; Bit 5 = Output:
00303 #DEFINE nupb6  PortB,6          ; Bit 6 = Output:
00304 #DEFINE nupb7  PortB,7          ; Bit 7 = Output:
00305

```

```

00000000      00306 PBDir   EQU      00000000b
0251 0C00      00307          MOVLW   PBDir           ;Initialize Port b directions:
0252 0006      00308          TRIS    PortB
00309
00000000      00310 InitPB  EQU      00000000b           ;Initialize Port B outputs:
0253 0C00      00311          MOVLW   InitPB
0254 0026      00312          MOVWF   PortB
00313
00314          RETP                ;End of "InitHW"
00315
00316
00317 ;
00318 ;Receive a character as 8 bits, no parity, 1 stop and put in
"SerReg".
00319 ; This routine has the start, stop and data bits inverted because an
inverting
00320 ; RS-232 buffer is not used.
00321 ;
0256 0C08      00322 Rcv      MOVLW   8                ;Put # of data bits
0257 002E      00323          MOVWF   BitCnt           ; into counter
0258 0705      00324 _Rc05   BTFSS   RxD                ;Wait for the start bit
0259 0A58      00325          GOTO    _Rc05
00326
00327          DelM    MIPS/Speed/2-4 ;Delay 1/2 bit time
0264 0705      00328          BTFSS   RxD                ;Is start still active?
0265 0A56      00329          GOTO    Rcv                ; try again if not
00330
00331 _Rc10      DelM    MIPS/Speed-9       ;Delay 1 bit time
0270 0705      00332          BTFSS   RxD                ;If data is a zero
0271 0503      00333          BSF     CF                ; then set the carry
0272 0605      00334          BTFSC   RxD                ;If data is a one
0273 0403      00335          BCF     CF                ; then clear the carry
0274 032F      00336          RRF     SerReg           ;Get carry into data bit
0275 02EE      00337          DECFSZ  BitCnt           ;Count the data bits
0276 0A66      00338          GOTO    _Rc10           ; and exit when done
00339
00340          DelM    MIPS/Speed/2-4 ;Delay 1/2 bit time
0281 020F      00341          MOVF   SerReg,W
00342          RETP                ;End of "Rcv"
00343
00344
00345 ;
00346 ;Transmit character in W-REG as 8 bits, no parity, 2 stop.
00347 ; Speed is spec'ed in bits/sec (Baud).
00348 ; This routine has the start, stop and data bits inverted because an
inverting
00349 ; RS-232 buffer is not used.
00350 ;
0283 002F      00351 Xmit     MOVWF   SerReg           ;Put char into shift register
0284 0C09      00352          MOVLW   9                ;Put # of data bits + start bit
0285 002E      00353          MOVWF   BitCnt           ; into counter
0286 0525      00354          BSF     TxD                ;Send start bit
00355
00356 _Xmt10     DelM    MIPS/Speed-11      ;Delay 1 bit time
0291 00EE      00357          DECFSZ BitCnt           ;Count the data bits
0292 0643      00358          BTFSC   ZF                ;
0293 0A9A      00359          GOTO    _Xmt30           ; and exit when done
0294 032F      00360          RRF     SerReg           ;Get data bit into carry
0295 0603      00361          BTFSC   CF                ;If carry is set
0296 0425      00362          BCF     TxD                ; then Xmit a zero
0297 0703      00363          BTFSS   CF                ;If carry is clear
0298 0525      00364          BSF     TxD                ; then transmit a one
0299 0A87      00365          GOTO    _Xmt10
00366
029A 0425      00367 _Xmt30   BCF     TxD                ;Send stop bit
00368          DelM    MIPS/Speed           ;Delay for the first stop bit

```

```

00369          RETP          ;End of "Xmit"
00370
00371
00372 ;
00373 ;Subroutine to input a 24-bit integer in decimal ASCII FORMAT.
00374 ; The result is in RegA.
00375 ;
02A6 0079      00376 DecIn  CLRF    RegA          ;Initialize
02A7 007A      00377          CLRF    RegA+1
02A8 007B      00378          CLRF    RegA+2
00379
02A9 0956      00380 _DI10  CALL    Rcv          ;Read in an ASCII digit
02AA 0038      00381          MOVWF  Digit
02AB 0503      00382          BSF    CF
02AC 0C30      00383          MOVLW  '0'          ;Convert it to binary
02AD 00B8      00384          SUBWF  Digit
02AE 0703      00385          BTFSS  CF
02AF 0ACD      00386          GOTO   _NotDig      ;Branch if it wasn't a digit
02B0 0503      00387          BSF    CF
02B1 0C0A      00388          MOVLW  10
02B2 0098      00389          SUBWF  Digit,W
02B3 0603      00390          BTFSC  CF
02B4 0ACD      00391          GOTO   _NotDig
00392
00393 ;RegA:= RegA *10.
02B5 0219      00394          MOVF   RegA,W          ;Save starting value
02B6 003C      00395          MOVWF  Temp
02B7 021A      00396          MOVF   RegA+1,W
02B8 003D      00397          MOVWF  Temp+1
02B9 021B      00398          MOVF   RegA+2,W
02BA 003E      00399          MOVWF  Temp+2
00400
02BB 0C09      00401          MOVLW  9          ;10 times
02BC 002B      00402          MOVWF  Count
02BD 0403      00403 _DI20  BCF    CF          ;RegA:= RegA +Temp
02BE 021C      00404          MOVF   Temp,W
02BF 01F9      00405          ADDWF  RegA
02C0 021D      00406          MOVF   Temp+1,W
02C1 01FA      00407          ADDWF  RegA+1
02C2 021E      00408          MOVF   Temp+2,W
02C3 01FB      00409          ADDWF  RegA+2
02C4 02EB      00410          DECFSZ Count
02C5 0ABD      00411          GOTO   _DI20
00412
02C6 0403      00413          BCF    CF          ;RegA:= RegA +Digit
02C7 0218      00414          MOVF   Digit,W
02C8 01F9      00415          ADDWF  RegA
02C9 0040      00416          CLRW
02CA 01FA      00417          ADDWF  RegA+1
02CB 01FB      00418          ADDWF  RegA+2
02CC 0AA9      00419          GOTO   _DI10      ;Loop until not a digit
00420
00421 _NotDig  RETP          ;End of "DecIn"
00422
00423
00424 ;
00425 ;Input the amplitude setting
00426 ;
02CE 09A6      00427 InpAmp  CALL    DecIn        ;Get an 8 bit amplitude
02CF 0219      00428          MOVF   RegA,W
02D0 09D2      00429          CALL    SetAmp
00430          RETP          ;End of "InpAmp"
00431
00432
00433 ;
00434 ;Set the amplitude.

```

00435 ; W-reg holds the amplitude with 0FFh being full amplitude.  
00436 ; The value being presented to the output DAC (Port B) is saved and

restored.

```
00437 ;
02D2 003D 00438 SetAmp MOVWF Temp+1 ;Save amplitude
02D3 0206 00439 MOVF PortB,W ;Save present output value
02D4 003C 00440 MOVWF Temp
00441
02D5 0545 00442 BSF DACWrt ;Disable data latch
02D6 0945 00443 CALL Dly250ns ;Wait for Write set up time
02D7 0465 00444 BCF DACSel ;Select the amplitude DAC
02D8 0945 00445 CALL Dly250ns ;Wait for Select set up time
02D9 021D 00446 MOVF Temp+1,W ;Get the amplitude back
02DA 0026 00447 MOVWF PortB ; and output it
02DB 0945 00448 CALL Dly250ns ;Wait for data set up time
02DC 0445 00449 BCF DACWrt ;Enable data latch
02DD 0945 00450 CALL Dly250ns ;Delay 250ns
02DE 0545 00451 BSF DACWrt ;Disable data latch
02DF 0945 00452 CALL Dly250ns ;Delay 250ns
00453
02E0 021C 00454 MOVF Temp,W ;Get back the output value
02E1 0026 00455 MOVWF PortB
02E2 0945 00456 CALL Dly250ns ;Delay 250ns
02E3 0565 00457 BSF DACSel ;Select the output DAC
02E4 0945 00458 CALL Dly250ns ;Delay 250ns
02E5 0445 00459 BCF DACWrt ;Enable data latch
00460 RETP ;End of "SetAmp"
00461
00462
00463 ;
00464 ;Input the frequency setting
00465 ;
02E7 09A6 00466 InpFreq CALL DecIn ;Read in an integer frequency
02E8 09EA 00467 CALL SetFreq
00468 RETP ;End of "InpFreq"
00469
00470
00471 ;
00472 ;Set the frequency
00473 ;
02EA 00474 SetFreq
02EA 0219 00475 MOVF RegA,W ;Update the frequency setting
02EB 0031 00476 MOVWF Freq
02EC 021A 00477 MOVF RegA+1,W
02ED 0032 00478 MOVWF Freq+1
02EE 021B 00479 MOVF RegA+2,W
02EF 0033 00480 MOVWF Freq+2
00481 RETP ;End of "SetFreq"
00482
00483
00484 ;
00485 ;Set the Sine waveform
00486 ;
02F1 0C04 00487 SetSine MOVLW SineTbl ;Point to sine waveform table
02F2 0037 00488 MOVWF Waveform
00489 RETP ;End of "SetSine"
00490
00491
00492 ;
00493 ;Set the Triangle waveform
00494 ;
02F4 0C45 00495 SetTri MOVLW TriTbl ;Point to triangle waveform table
02F5 0037 00496 MOVWF Waveform
00497 RETP ;End of "SetTri"
00498
00499
```

```

00500 ;
00501 ;Set the Triangle waveform
00502 ;
02F7 0C86 00503 SetSqr   MOVLW   SqrTbl           ;Point to square waveform table
02F8 0037 00504           MOVWF   Waveform
00505           RETP             ;End of "SetSqr"
00506
02FA      00507 ShiftCmd
02FA 0C1A 00508           MOVLW   LOW (ShiftTbl)  ;Point to the shifted command table
02FB 0030 00509           MOVWF   CmdPtr
02FC 0956 00510           CALL    Rcv             ;Get the shifted command
02FD 0926 00511           CALL    CmdScan        ; and do it
00512           RETP             ;End of "ShiftCmd"
00513
00514
00515 ;
00516 ;===== Start of Program
=====
00517 ;
00518 ;
00519
0301 0068 00520 Start   FCALL   InitHW
can      00521           CLRFB   Zero           ; initialize the value of zero so we
using    00522           ; calculate the two's complement of W
00523           ; "SUBWF ZERO,W"
0302 0C05 00524           MOVLW   5             ;Holds the value 5 for "ADDLW 5"
0303 0029 00525           MOVWF   Five
0304 0C0F 00526           MOVLW   15          ;Holds the value 15 for "SUBLW 15"
0305 002A 00527           MOVWF   Fifteen
00528
00529
00530 ;           OscFreq/LoopLength      AccumulatorLength
00531 ;           ----- = -----
00532 ;           FrequencyOut            Increment
00533
00534 ;
00535 ;FreqOut = (MIPS /LoopCycles *N) / 2^^24
00536 ;N = FreqOut * 2^^24 /OscFreq *LoopCycles
00537 ;
00002710 00538 N           EQU      16777216 *30 /MIPS *1000 ;This gives about 10,000
for 1KHZ:
00539           ; 10,066,3296
0306 0C10 00540           MOVLW   LOW(N)       ;Initialize to 1 KHz
0307 0031 00541           MOVWF   Freq
0308 0C27 00542           MOVLW   HIGH(N)
0309 0032 00543           MOVWF   Freq+1
030A 0C00 00544           MOVLW   00h
030B 0033 00545           MOVWF   Freq+2
00546
00547
030C 0C04 00548           MOVLW   SineTbl      ;Initialize to sinewave
030D 0037 00549           MOVWF   Waveform
00550
030E 0C7F 00551           MOVLW   7Fh         ;Set at half amplitude
030F 09D2 00552           CALL    SetAmp
00553
00554
00555 ;
00556 ;===== MAIN LOOP =====
00557 ;
00558
0310 0C00 00559 Main   MOVLW   LOW (CmdTbl)      ;Point to the command table
0311 0030 00560           MOVWF   CmdPtr
00561           FCALL   GenWave        ;Generate the waveform and return if

```

a

```
00562 ; start bit is detected
0314 0956 00563 CALL Rcv ;Get the command
0315 0926 00564 CALL CmdScan ; and do it
00565
0316 0B10 00566 GOTO Main ;Restart at a new frequency or
amplitude
00567
00568 END
```

```
Errors : 0
Warnings : 0 reported, 19 suppressed
Messages : 1 reported, 25 suppressed
```