Totally Terrific Talking Thermometer

Author: *David Brobst*
Solutions Cubed
Chico, CA

OVERVIEW

This design is being used as an entry into the Microchip De$igning for Dollar$ design contest for the month of July. It is based upon monitoring and communicating environmental conditions with a PIC12CXXX 8-bit microcontroller.

DESIGN IDEA

Since man first crawled out of the primordial swamp a thought has constantly flitted across his mind: "Damn it's hot -- I wonder how hot?" Old men in barber shops still greet each other with the time honored, "Hot one, ain't it?" Wouldn't the originator of this harmless social pleasantry be amazed if the barber was able to give the proper response, along with the EXACT temperature, all without looking up from the bald head he is desperately trying to stretch into a 20 minute haircut?

Everyone is currently aware of the massive difficulties being experienced by the space station Mir and its exhausted crew. It is plastered all over the TV, radio, and print media. It all started with an aborted satellite docking, which punctured a hole into the space station. We'll never know if the reason for this mishap was because the controller who was trying to bring the satellite in wondered how hot it was and momentarily glanced up from the controls to catch a glimpse at the thermometer, thereby dooming he and his crew. But we do know this is a viable situation, or could be, at least on paper, sort of.

How about the blind? How are they supposed to tell temperature? Mercury is a liquid at room temperature. Liquid substances are VERY hard to make Braille characters from. Maybe something *audible* would be in order. Mmmmmm. . . .

How about the person that already has everything? What could be more pleasing and heartwarming than a talking thermometer? I wonder how one of those would work? Is there such a thing? Boy, if there was, I bet its super high tech and very expensive.

Not to fear the 4T-X2000 is here. The 4T- is short for Totally Terrific Talking Thermometer. X2000 signifies how rad, nifty, cool, out of control, and space age the 4T-X2000 is.

The 4T-X2000 uses the PIC12C508 8 pin microcontroller to control a thermometer and sound controller. The sound is supplied by the low cost and easily acquired Information Storage Devices ISD2590. There were numerous temperature measurement possibilities which could have been implemented. However the Dallas DS1820 is a spiffy device using a 1-wire communication/power bus so it was chosen.

Figure 1 gives the block diagram of the system. Figure 2 gives a flow chart of the main program flow. The system works fairly simply. It sits in an idle state until the push button is pressed and held down. This applies power to the system. The microcontroller then queries the temperature sensor to find the ambient temperature. After the temperature is found, it is converted into two values: 10s and 1s. These are then cross referenced to the sounds available. These two sounds are then played back to back which gives the temperature in degrees F. The program then sits in a dummy loop taking no further action. The only way to get the temperature again is to release the button and repeat the procedure.

FIGURE 1: BLOCK DIAGRAM OF 4T-X2000

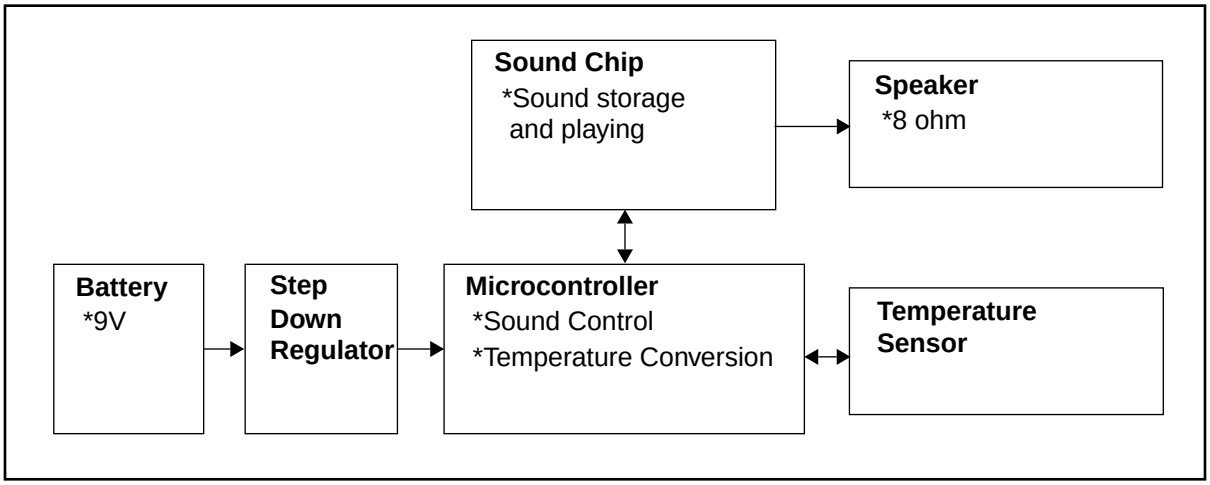
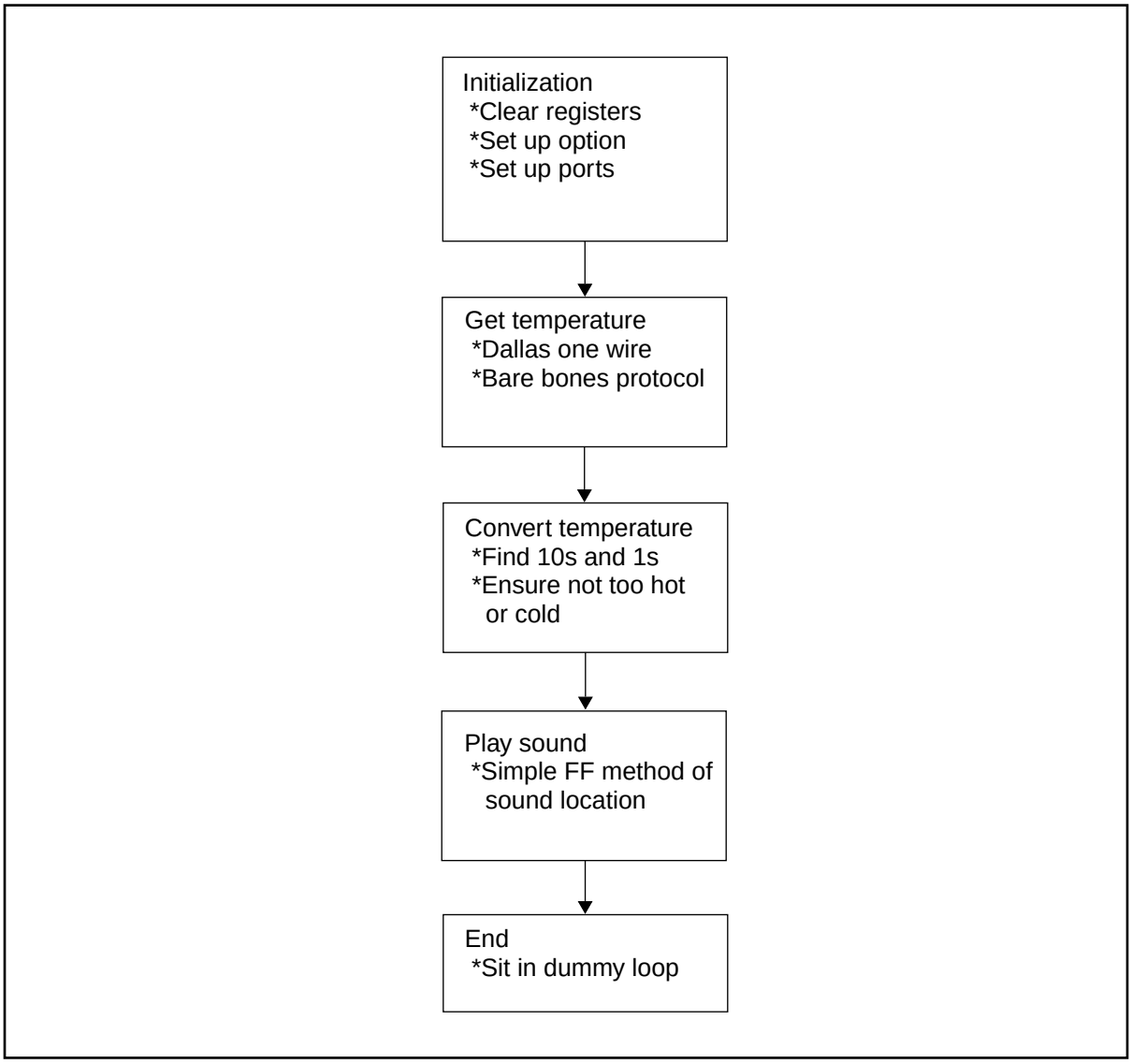


FIGURE 2: FLOW CHART OF 4T-X2000 OPERATION



HARDWARE METHODOLOGY

Appendix A gives the schematic of the 4T-X2000. The heart of the system is the PIC12C508. It is run with off of its handy and versatile built-in internal 4MHz RC oscillator. This frees up two bi-directional I/O pins. The internal /MCLR was used just to show that it COULD BE DONE. The envelope is no place for the 4T-X2000. It pushes it all over the place.

With the plethora of I/O pins available, a whole slew of nifty things were attached to the 4T-X2000. The temperature measurement is accomplished with the Dallas DS1820. The DS1820 uses a neat 1-wire communication protocol/parasitic power bus. This means that digital communication can be accomplished using only two wires. So for the same connector issues of a thermistor, a noise-immune digital communication interface can be implemented. The pull up resistor(R5) allows for the pseudo-open collector communication channel to be implemented. More information on the DS1820 can be found in the Dallas System Extension Data book (1995/1996).

The rest of the I/Os were used to interface to the ISD2590 sound chip. This chip is able to directly drive a speaker, thereby alleviating the need for drive circuitry. R4 is used to keep the chip in constant play mode and stay out of record. R1, R2, and R3 are used to put the chip in the special message cueing mode. All of the other address lines are not used and tied to ground to avoid entering spurious modes. The /OVF pin is left floating because it is an output.

SOFTWARE METHODOLOGY

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

The software can be broken up into three main sections: Setup, Temperature, and Sound. They will be discussed in that order.

Setup

The setup is fairly simple for this program and chip. First all of the user registers are cleared so they are of a known state going into the program. A simple indirect addressing routine is used for this. Then the OPTION register is set up. After this the GPIO pins are set up for direction and initial values. The most important thing here is that the PD pin is high, thereby powering the ISD chip down. This ensures it will not spuriously play during the power up sequence.

Temperature

All of the actual temperature measurement is done in the DS1820, therefore the PIC merely has to read in the data and then it is ready to roll. As mentioned before, the DS1820 uses a 1 wire interface which relies on pulse widths in a data window to signify ones and zeros. It also needs to be an open collector type system. The GPIO TRIS command allows for a fairly easy implementation of this.

The DS1820 reads temperature in Celsius to the nearest half degree. In order to be used easily in America, this temperature is converted over to Fahrenheit using a simple look up table. Appendix B shows the conversion table. The temperature is limited at 40F and 99F to show how temperature out of range could be handled. The DS1820 actually measures from -55C to +125C.

Sound

The ISD2590 allows for a very sophisticated addressing system in 150mS increments. This was not used. (hah!)

Because of the I/O constraints one of the built in “operational modes” was used: Message Cueing. This allows for rapid fast forwarding to relative positions of messages. Therefore, to play the “70” sound, all that is necessary is knowing that the 70 degree message is in position 3. Appendix C shows the positions of the messages in the chip. After getting to the message that needs to be played and simple chip enable strobe is performed and the message plays and the PIC is in “hands off” mode. In this case the PIC waits till the message is over to ensure that the second message does not start playing over the first.

The ISD chip is reset between the playing of the first and second message to ensure that the message pointer is at the beginning of the address space.

RAM Used:	12 bytes, 3 bytes are TEMP registers
Program Bytes (as presented):	283 bytes
OSC:	Internal RC
WDT:	Off
CP:	Off
$\overline{\text{MCLR}}$	Internal
Check Sum:	E95A

WHERE TO GO FROM HERE

There are numerous places to go with the 4T-X200. First is utilizing the GP3 input pin. A humidity sensor with a frequency output could be added here or something along those lines. With some thought and judicious signaling some of the I/Os to the ISD2590 could be multiplexed (in particular the A0 and the PD line).

The sound could be dramatically improved by spending more time recording the words in the memory buffer. This would allow for smoother message concatenation. Using the top of the line ISD development system, that we do not have access to, would allow computer editing of the messages. Also the chips that have less space for sound use a faster sampling rate (Nyquist strikes again), which increases the sound quality. And finally, ISD provides a serially controlled sound chip. . . .

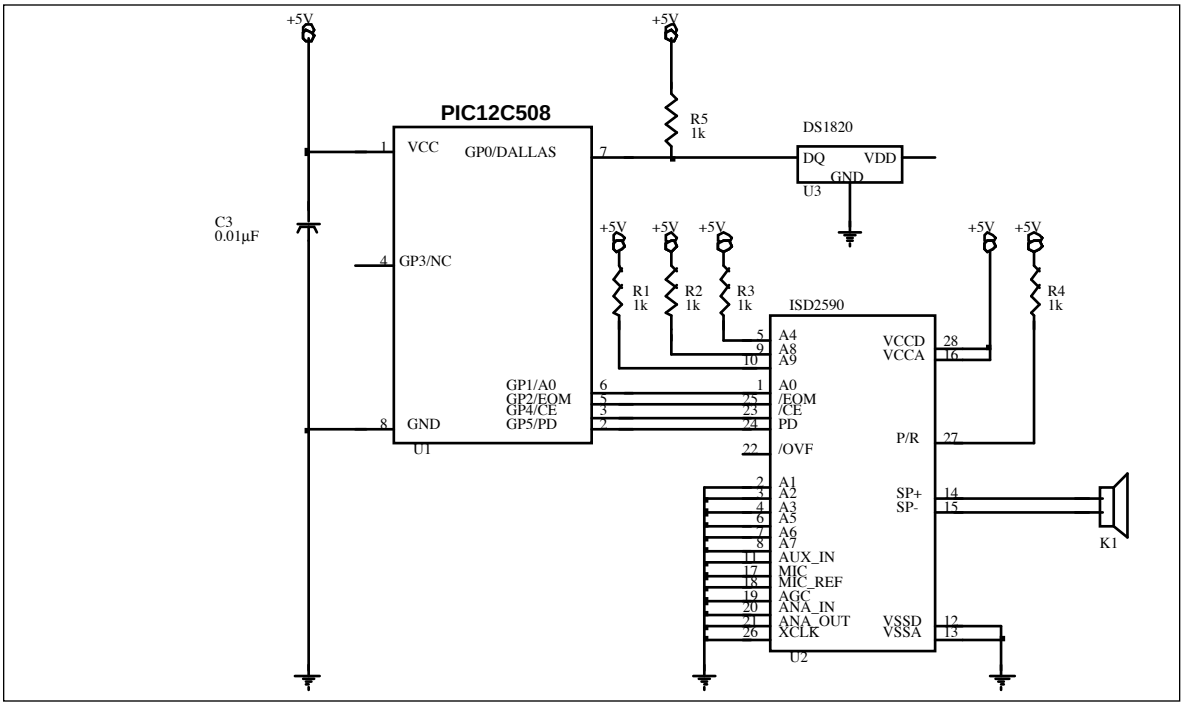
The implementation of the DS1820 communication is bare-bones. Multiple DS1820s can be put on one bus which requires greater communication overhead than was implemented. However multiple temperature sensors would allow the 4T-X2000 to be a serious temperature measuring and talking machine. Also the DS1820 has temperature alarms, scratch-pad RAM, and CRC protected communication. None of this was implemented. However, because the communication is working, utilizing the rest of this should be like putting gravy on a Thanksgiving turkey.

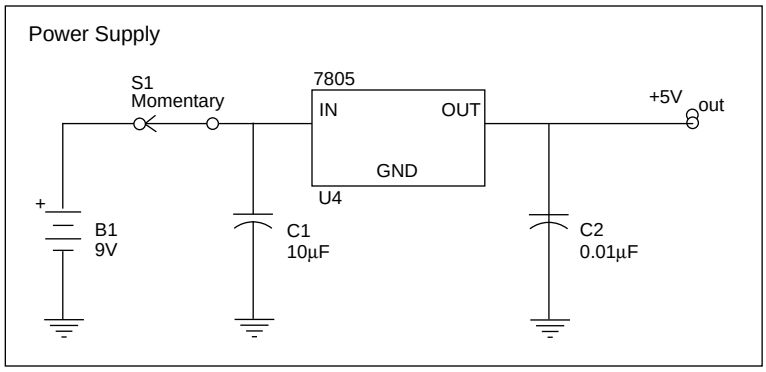
4T-X2000 DIRECTIONS FOR USE

1. Place the temperature sensor (at the end of the blue and black wire) whre you want the temperature measured.
2. Press and Hold Down the black button.
3. After the message has stopped, repeat above for different temperatures.

Note: Try temperature ranges of 30°F to 110 °F to get the range of messages.

APPENDIX A: SCHEMATIC





APPENDIX B: TEMPERATURE RANGES

Degrees C	Degrees F	Converted Value		Degrees C	Degrees F	Converted Value
4.5	40.1	40		28.5	83.3	83
5	41	41		29	84.2	84
5.5	41.9	42		29.5	85.1	85
6	42.8	43		30	86	86
6.5	43.7	44		30.5	86.9	87
7	44.6	45		31	87.8	88
7.5	45.5	46		31.5	88.7	89
8	46.4	46		32	89.6	90
8.5	47.3	47		32.5	90.5	91
9	48.2	48		33	91.4	91
9.5	49.1	49		33.5	92.3	92
10	50	50		34	93.2	93
10.5	50.9	51		34.5	94.1	94
11	51.8	52		35	95	95
11.5	52.7	53		35.5	95.9	96
12	53.6	54		36	96.8	97
12.5	54.5	55		36.5	97.7	98
13	55.4	55		37	98.6	99
13.5	56.3	56		37.5	99.5	100
14	57.2	57		38	100.4	100
14.5	58.1	58				
15	59	59				
15.5	59.9	60	Note: Formula Used to Convert from Celsius to Fahrenheit: Degrees F = (1.8*Degrees C) + 32			
16	60.8	61				
16.5	61.7	62				
17	62.6	63				
17.5	63.5	64				
18	64.4	64				
18.5	65.3	65				
19	66.2	66				
19.5	67.1	67				
20	68	68				
20.5	68.9	69				
21	69.8	70				
21.5	70.7	71				
22	71.6	72				
22.5	72.5	73				
23	73.4	73				
23.5	74.3	74				
24	75.2	75				
24.5	76.1	76				
25	77	77				
25.5	77.9	78				
26	78.8	79				

Degrees C	Degrees F	Converted Value		Degrees C	Degrees F	Converted Value
26.5	79.7	80				
27	80.6	81				
27.5	81.5	82				
28	82.4	82				

APPENDIX C:

Message Number	Message
0	It's 40 . . .
1	It's 50 . . .
2	It's 60 . . .
3	It's 70 . . .
4	It's 80 . . .
5	It's 90 . . .
6	1 degrees.
7	2 degrees.
8	3 degrees.
9	4 degrees.
10	5 degrees.
11	6 degrees.
12	7 degrees.
13	8 degrees.
14	9 degrees.
15	degrees.
16	It's cold. . .
17	It's hot. . .

APPENDIX D: SOURCE CODE

```
MPASM 01.50 Released           TTEMP3.ASM   7-22-1997  10:06:19         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 Talking Temperature Meter                ****
            00011 ;*******************************************************************************
            00012 ;
            00013 ;       The purpose of this code is to develop an entry for the Microchip
            00014 ;De$igning for Dollar$ 12CXXX design contest, session 3.  This session's
            00015 ;objective is to design a system which monitors the environment.  The basic
            00016 ;design for Solutions Cubed's entry is a talking temperature meter.  A
            00017 ;PIC12C508 is used as the brains; the Dallas DS1820 is used as the
            00018 ;temperature measuring device; the Information Storage Devices ISD2590 is
            00019 ;used to provide the sound capability.
            00020 ;
            00021 ;TTEMP1.ASM:  Implements all sound functions.  (7-20-97)
            00022 ;TTEMP2.ASM:  Implements reading the DS1820.  (7-21-97)
            00023 ;TTEMP3.ASM:  Implements code on 12C508.  (7-22-97)
            00024 ;*******************************************************************************
            00025 ;
            00026 ;
            00027 ;*******************************************************************************
            00028 ;*******************************************************************************
            00029 ;****   Define registers, constants, processor, and assembler directives ****
            00030 ;*******************************************************************************
            00031 ;*******************************************************************************
            00032 ;
            00033 ;Processor
            00034 ;
            00035         LIST    P=12C508                        ;Processor used
            00036 ;
            00037 ;Processor defined registers and bits
            00038 ;
            00039         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
            00040 ;
            00041 ;Program defined registers
            00042 ;
  00000007  00043 TEMP0                   EQU     H'07'   ;Pseudo-WORKING registers
  00000008  00044 TEMP1                   EQU     H'08'
  00000009  00045 TEMP2                   EQU     H'09'
  0000000A  00046 FLAG0                   EQU     H'0A'   ;Pseudo-STATUS registers
            00047 ;
  0000000B  00048 TEMP_HI                 EQU     H'0B'   ;High byte of temperature
  0000000C  00049 TEMP_LO                 EQU     H'0C'   ;Low byte of temperature
  0000000D  00050 FAREN_TEMP              EQU     H'0D'   ;Farenheit temperature
            00051 ;
  0000000E  00052 COM_REG                 EQU     H'0E'   ;Used for sending and receiving
            00053 ;
  0000001C  00054 MESSAGE1                EQU     H'1C'   ;First part of temperature message
  0000001D  00055 MESSAGE2                EQU     H'1D'   ;Second part of temperature message
```

```
  0000001E  00056 LSD                     EQU     H'1E'   ;Used in binary to BCD conversion
  0000001F  00057 MSD                     EQU     H'1F'
            00058 ;
            00059 ;Program defined bits
            00060 ;
            00061 ;FLAG0 bits
  00000000  00062 COLD            EQU     H'00'           ;Set if temperature is less than 40F
  00000001  00063 HOT             EQU     H'01'           ;Set if temperature is more than 99F
            00064 ;
            00065 ;I/O definitions
            00066 #DEFINE         DALLAS  GPIO,0          ;Dallas temperature sensor I/O line
            00067 #DEFINE         A0      GPIO,1          ;A0 bit of ISD device
            00068 #DEFINE         EOM     GPIO,2          ;EOM indicator of ISD
            00069 #DEFINE         BUTTON  GPIO,3          ;Button to start temperature
            00070 #DEFINE         CE      GPIO,4          ;Chip Enable of ISD
            00071 #DEFINE         PD      GPIO,5          ;Power Down of ISD
            00072 ;
            00073 ;*****************************************************************************
            00074 ;
            00075 ;
            00076 ;*****************************************************************************
            00077 ;*****************************************************************************
            00078 ;****                            Reset Vector                             ****
            00079 ;*****************************************************************************
            00080 ;*****************************************************************************
0000        00081         ORG     H'000'
0000 0A99   00082         GOTO    MAIN
            00083 ;*****************************************************************************
            00084 ;
            00085 ;
            00086 ;*****************************************************************************
            00087 ;*****************************************************************************
            00088 ;****                            Audio Routines                           ****
            00089 ;*****************************************************************************
            00090 ;*****************************************************************************
            00091 ;TEMP_LOOK_UP -- Look up table for CONVERT_TEMP
            00092 ;BIN2BCD_SMALL -- Binary to BCD (0-99)
            00093 ;CONVERT_TEMP -- Converts from degrees C to degrees F
            00094 ;ISD_RESET -- Reset the ISD device
            00095 ;*****************************************************************************
            00096 ;
            00097 ;
            00098 ;*****************************************************************************
            00099 ;TEMP_LOOK_UP:  This routine is a look up table which converts the value in
            00100 ;W to degrees F.  It is placed out of alphebetical order, to ensure the page
            00101 ;bits are correct.
            00102 ;       Called From:            CONVERT_TEMP
            00103 ;       Registers Used:         PCL
            000104 ;       Subroutines Called:     NONE
            00105 ;
0001        00106 TEMP_LOOK_UP
0001 01E2   00107         ADDWF   PCL,F
0002 0828   00108         RETLW   H'28'                   ;40F
0003 0829   00109         RETLW   H'29'                   ;41F
0004 082A   00110         RETLW   H'2A'                   ;42F
0005 082B   00111         RETLW   H'2B'                   ;43F
0006 082C   00112         RETLW   H'2C'                   ;44F
0007 082D   00113         RETLW   H'2D'                   ;45F
0008 082E   00114         RETLW   H'2E'                   ;46F
0009 082E   00115         RETLW   H'2E'                   ;46F
000A 082F   00116         RETLW   H'2F'                   ;47F
000B 0830   00117         RETLW   H'30'                   ;48F
000C 0831   00118         RETLW   H'31'                   ;49F
000D 0832   00119         RETLW   H'32'                   ;50F
000E 0833   00120         RETLW   H'33'                   ;51F
000F 0834   00121         RETLW   H'34'                   ;52F
```

```
0010 0835   00122         RETLW   H'35'                   ;53F
0011 0836   00123         RETLW   H'36'                   ;54F
0012 0837   00124         RETLW   H'37'                   ;55F
0013 0837   00125         RETLW   H'37'                   ;55F
0014 0838   00126         RETLW   H'38'                   ;56F
0015 0839   00127         RETLW   H'39'                   ;57F
0016 083A   00128         RETLW   H'3A'                   ;58F
0017 083B   00129         RETLW   H'3B'                   ;59F
0018 083C   00130         RETLW   H'3C'                   ;60F
0019 083D   00131         RETLW   H'3D'                   ;61F
001A 083E   00132         RETLW   H'3E'                   ;62F
001B 083F   00133         RETLW   H'3F'                   ;63F
001C 0840   00134         RETLW   H'40'                   ;64F
001D 0840   00135         RETLW   H'40'                   ;64F
001E 0841   00136         RETLW   H'41'                   ;65F
001F 0842   00137         RETLW   H'42'                   ;66F
0020 0843   00138         RETLW   H'43'                   ;67F
0021 0844   00139         RETLW   H'44'                   ;68F
0022 0845   00140         RETLW   H'45'                   ;69F
0023 0846   00141         RETLW   H'46'                   ;70F
0024 0847   00142         RETLW   H'47'                   ;71F
0025 0848   00143         RETLW   H'48'                   ;72F
0026 0849   00144         RETLW   H'49'                   ;73F
0027 0849   00145         RETLW   H'49'                   ;73F
0028 084A   00146         RETLW   H'4A'                   ;74F
0029 084B   00147         RETLW   H'4B'                   ;75F
002A 084C   00148         RETLW   H'4C'                   ;76F
002B 084D   00149         RETLW   H'4D'                   ;77F
002C 084E   00150         RETLW   H'4E'                   ;78F
002D 084F   00151         RETLW   H'4F'                   ;79F
002E 0850   00152         RETLW   H'50'                   ;80F
002F 0851   00153         RETLW   H'51'                   ;81F
0030 0852   00154         RETLW   H'52'                   ;82F
0031 0852   00155         RETLW   H'52'                   ;82F
0032 0853   00156         RETLW   H'53'                   ;83F
0033 0854   00157         RETLW   H'54'                   ;84F
0034 0855   00158         RETLW   H'55'                   ;85F
0035 0856   00159         RETLW   H'56'                   ;86F
0036 0857   00160         RETLW   H'57'                   ;87F
0037 0858   00161         RETLW   H'58'                   ;88F
0038 0859   00162         RETLW   H'59'                   ;89F
0039 085A   00163         RETLW   H'5A'                   ;90F
003A 085B   00164         RETLW   H'5B'                   ;91F
003B 085B   00165         RETLW   H'5B'                   ;91F
003C 085C   00166         RETLW   H'5C'                   ;92F
003D 085D   00167         RETLW   H'5D'                   ;93F
003E 085E   00168         RETLW   H'5E'                   ;94F
003F 085F   00169         RETLW   H'5F'                   ;95F
0040 0860   00170         RETLW   H'60'                   ;96F
0041 0861   00171         RETLW   H'61'                   ;97F
0042 0862   00172         RETLW   H'62'                   ;98F
0043 0863   00173         RETLW   H'63'                   ;99F
            00174 ;*******************************************************************************
            00175 ;
            00176 ;
            00177 ;*******************************************************************************
            00178 ;BIN2BCD_SMALL:  This routine converts the FAREN_TEMP number into a two byte
            00179 ;BCD number.  The tens digit is stored in MSD and the ones digit in LSD.
            00180 ;This routine only converts  numbers between 00 and 99.  This is based on the
            00181 ;routine found in AN582 in the Embedded Control Handbook.
            00182 ;       Called From:            MAIN
            00183 ;       Registers Used:         FAREN_TEMP, LSD, MSD, STATUS
            00184 ;       Subroutines Called:     NONE
            00185 ;
0044        00186 BIN2BCD_SMALL
0044 020D   00187         MOVF    FAREN_TEMP,W
```

```
0045 003E   00188         MOVWF   LSD
0046 007F   00189         CLRF    MSD
0047 0C0A   00190 CHECK10 MOVLW   H'0A'                   ;See if LSD less than 10
0048 009E   00191         SUBWF   LSD,W
0049 0703   00192         BTFSS   STATUS,C                ;If C set then greater than 10
004A 0A4E   00193         GOTO    BIN2BCD_SMALL_END
004B 003E   00194         MOVWF   LSD                     ;Get ready for next try
004C 02BF   00195         INCF    MSD,F
004D 0A47   00196         GOTO    CHECK10
004E        00197 BIN2BCD_SMALL_END
004E 0800   00198         RETLW   H'00'
            00199 ;*******************************************************************************
            00200 ;
            00201 ;
            00202 ;*******************************************************************************
            00203 ;CONVERT_TEMP:  This routine converts the DS1820 temperature which comes in
            00204 ;in degrees C to degrees F.  Temperatures greater than 99F are deemed too hot
            00205 ;while those lower than 40F are too cold.  Therefore, the entire range of the
            00206 ;sensor is definitely not used.  Because the degrees are given to the nearest
            00207 ;1/2 degree C, all of the measurements are "double" what they should be.
            00208 ;       Called From:            MAIN
            00209 ;       Registers Used:         FAREN_TEMP, FLAG0, STATUS, TEMP_HI, TEMP_LO
            000210 ;       Subroutines Called:     TEMP_LOOK_UP
            00211 ;
004F        00212 CONVERT_TEMP
004F 020B   00213         MOVF    TEMP_HI,W               ;See if below freezing
0050 0743   00214         BTFSS   STATUS,Z                ;If Z set then a positive temperature
0051 0A61   00215         GOTO    SET_COLD
0052 0C09   00216         MOVLW   H'09'                   ;See if too cold (<40 F)
0053 008C   00217         SUBWF   TEMP_LO,W
0054 0703   00218         BTFSS   STATUS,C                ;If C set then temp > 40 F
0055 0A61   00219         GOTO    SET_COLD
0056 0C4D   00220         MOVLW   H'4D'                   ;See if too hot (>99F)
0057 008C   00221         SUBWF   TEMP_LO,W
0058 0603   00222         BTFSC   STATUS,C                ;If C clear then temp < 100 F
0059 0A5F   00223         GOTO    SET_HOT
005A 0C09   00224         MOVLW   H'09'                   ;Set up for look up table
005B 008C   00225         SUBWF   TEMP_LO,W
005C 0901   00226         CALL    TEMP_LOOK_UP            ;Do conversion
005D 002D   00227         MOVWF   FAREN_TEMP              ;Buffer temperature
005E 0A62   00228         GOTO    CONVERT_TEMP_END
005F 052A   00229 SET_HOT BSF     FLAG0,HOT
0060 0A62   00230         GOTO    CONVERT_TEMP_END
0061        00231 SET_COLD
0061 050A   00232         BSF     FLAG0,COLD              ;Temperature too cold
0062        00233 CONVERT_TEMP_END
0062 0800   00234         RETLW   H'00'
            00235 ;*******************************************************************************
            00236 ;
            00237 ;
            00238 ;*******************************************************************************
            00239 ;ISD_RESET:  This routine resets the ISD chip to ensure that it works
            00240 ;properly.
            00241 ;       Called From:            MAIN
            00242 ;       Registers Used:         GPIO, TEMP0, TEMP1
            00243 ;       Subroutines Called:     NONE
            00244 ;
0063        00245 ISD_RESET
0063 05A6   00246         BSF     PD                      ;Ensure address pointer reset
0064 0067   00247         CLRF    TEMP0                   ;Delay to ensure powered down
0065 0C40   00248         MOVLW   H'40'
0066 0028   00249         MOVWF   TEMP1
0067 02E7   00250 IR_1    DECFSZ  TEMP0,F
0068 0A67   00251         GOTO    IR_1
0069 02E8   00252         DECFSZ  TEMP1,F
006A 0A67   00253         GOTO    IR_1
```

```
006B 04A6   00254         BCF     PD                      ;Turn ISD device back on
006C        00255 ISD_RESET_END
006C 0800   00256         RETLW   H'00'
            00257 ;*******************************************************************************
            00258 ;
            00259 ;
            00260 ;*******************************************************************************
            00261 ;*******************************************************************************
            00262 ;****                          Dallas Routines                              ****
            000263 ;*******************************************************************************
            00264 ;*******************************************************************************
            00265 ;DLONG_DELAY -- About 770uS delay
            00266 ;DOUT_HIGH -- Allows pullup to pull line high
            00267 ;DOUT_LOW -- Pulls line low
            00268 ;DRECEIVE -- Receives data from 1820
            00269 ;DSEND -- Sends data to 1820
            00270 ;DSHORT_DELAY -- About a 60uS delay
            00271 ;*******************************************************************************
            00272 ;
            00273 ;
            00274 ;*******************************************************************************
            00275 ;DLONG_DELAY -- This routine is a simple delay loop of about 770uS (4MHz
            00276 ;XTAL).  It is used after the reset pulse to ensure that the DS1820 is
            00277 ;powered up.
            00278 ;       Called From:            MAIN
            00279 ;       Registers Used:         TEMP0
            00280 ;       Subroutines Called:     NONE
            00281 ;
006D        00282 DLONG_DELAY
006D 0067   00283         CLRF    TEMP0
006E 02E7   00284 DLD_1   DECFSZ  TEMP0,F
006F 0A6E   00285         GOTO    DLD_1
0070        00286 DLONG_DELAY_END
0070 0800   00287         RETLW   H'00'
            00288 ;*******************************************************************************
            00289 ;
            00290 ;
            00291 ;*******************************************************************************
            00292 ;DOUT_HIGH:  This routine sets the DALLAS line as an input which allows the
            00293 ;external pull up resistor to make the line high.
            00294 ;       Called From:            MAIN
            00295 ;       Registers Used:         TRISGPIO
            00296 ;       Subroutines Called:     NONE
            00297 ;
0071        00298 DOUT_HIGH
0071 0C05   00299         MOVLW   H'05'                   ;0000 0101 -- Make RA0 an input
0072 0006   00300         TRIS    GPIO
0073        00301 DOUT_HIGH_END
0073 0800   00302         RETLW   H'00'
            00303 ;*******************************************************************************
            00304 ;
            00305 ;
            00306 ;*******************************************************************************
            00307 ;DOUT_LOW:  This routine sets the DALLAS line as an output and then actively
            00308 ;pulls the line low.
            00309 ;       Called From:            MAIN
            00310 ;       Registers Used:         GPIO, TRISGPIO
            00311 ;       Subroutines Called:     NONE
            00312 ;
0074        00313 DOUT_LOW
0074 0C04   00314         MOVLW   H'04'                   ;0000 0100 -- Make RA0 an output
0075 0006   00315         TRIS    GPIO
0076 0406   00316         BCF     DALLAS
0077        00317 DOUT_LOW_END
0077 0800   00318         RETLW   H'00'
            00319 ;*******************************************************************************
```

```
                00320 ;
                00321 ;
                00322 ;*******************************************************************************
                00323 ;DRECEIVE:  This routine receives data from the DALLAS chip.
                00324 ;       Called From:            MAIN
                00325 ;       Registers Used:         COM_REG, GPIO, STATUS, TEMP1
                00326 ;       Subroutines Called:     DOUT_HIGH, DOUT_LONG, DSHORT_DELAY
                00327 ;
0078            00328 DRECEIVE
0078 0C08       00329         MOVLW   H'08'                   ;Number of bits to receive
0079 0028       00330         MOVWF   TEMP1
007A 0974       00331 DR_1    CALL    DOUT_LOW                ;Start read time slot
007B 0971       00332         CALL    DOUT_HIGH               ;Allow to read
007C 0706       00333         BTFSS   DALLAS                  ;See if should be 1
007D 0403       00334         BCF     STATUS,C
007E 0606       00335         BTFSC   DALLAS                  ;See if should be 0
007F 0503       00336         BSF     STATUS,C
0080 032E       00337         RRF     COM_REG,F               ;LSB first
0081 0994       00338         CALL    DSHORT_DELAY            ;Ensure slot is long enough
0082 02E8       00339         DECFSZ  TEMP1,F
0083 0A7A       00340         GOTO    DR_1
0084            00341 DRECEIVE_END
0084 0800       00342         RETLW   H'00'
                00343 ;*******************************************************************************
                00344 ;
                00345 ;
                00346 ;*******************************************************************************
                00347 ;DSEND: This routine sends commands from the controller to the DS1820.  Data
                00348 ;is sent LSB first with no parity, stop, or start bits
                00349 ;       Called From:            MAIN
                00350 ;       Registers Used:         COM_REG, GPIO, STATUS, TEMP1
                00351 ;       Subroutines Called:     DOUT_HIGH, DOUT_LOW, DSHORT_DELAY
                00352 ;
0085            00353 DSEND
0085 002E       00354         MOVWF   COM_REG                 ;Buffer command to send
0086 0C08       00355         MOVLW   H'08'                   ;Number of bits to send
0087 0028       00356         MOVWF   TEMP1
0088 0974       00357 DS_0    CALL    DOUT_LOW                ;Start the write slot
0089 032E       00358         RRF     COM_REG,F               ;LSB first, see what to send
008A 0703       00359         BTFSS   STATUS,C                ;See if send 1
008B 0A8E       00360         GOTO    SEND0
008C 0971       00361         CALL    DOUT_HIGH               ;Make DALLAS high
008D 0A8F       00362         GOTO    DS_1
008E 0974       00363 SEND0   CALL    DOUT_LOW
008F 0994       00364 DS_1    CALL    DSHORT_DELAY            ;Wait 60uS to allow DS1820 to sample
0090 0971       00365         CALL    DOUT_HIGH               ;Make bus back high
0091 02E8       00366         DECFSZ  TEMP1,F                 ;See if all bits sent
0092 0A88       00367         GOTO    DS_0
0093            00368 DSEND_END
0093 0800       00369         RETLW   H'00'
                00370 ;*******************************************************************************
                00371 ;
                00372 ;
                00373 ;*******************************************************************************
                00374 ;DSHORT_DELAY:  This routine is used during the communication routines to
                00375 ;delay for the 60uS that the DS1820 needs between the bit slots for reading
                00376 ;and writing.
                00377 ;       Called From:            DSEND
                00378 ;       Registers Used:         TEMP0
                00379 ;       Subroutines Called:     NONE
                00380 ;
0094            00381 DSHORT_DELAY
0094 0C14       00382         MOVLW   H'14'
0095 0027       00383         MOVWF   TEMP0
0096 02E7       00384 DSD_1   DECFSZ  TEMP0,F
0097 0A96       00385         GOTO    DSD_1
```

```
0098        00386 DSHORT_DELAY_END
0098 0800   00387         RETLW   H'00'
            00388 ;*******************************************************************************
            00389 ;
            00390 ;
            00391 ;*******************************************************************************
            00392 ;*******************************************************************************
            00393 ;*******************************************************************************
            00394 ;****                              Main Code                                ****
            00395 ;*******************************************************************************
            00396 ;*******************************************************************************
            00397 ;*******************************************************************************
            00398 ;
            00399 ;
0099        00400 MAIN
0099 0025   00401         MOVWF   OSCCAL                  ;Get calibration value for RC
009A        00402 CLEAR_REGISTERS
009A 0067   00403         CLRF    TEMP0                   ;Clear first RAM location for use
009B 0C18   00404         MOVLW   H'18'                   ;Number of registers to clear
009C 0027   00405         MOVWF   TEMP0
009D 0C08   00406         MOVLW   H'08'                   ;Start of RAM clearing
009E 0024   00407         MOVWF   FSR
009F        00408 CLEAR_LOOP
009F 0060   00409         CLRF    INDF                    ;Clear register pointed to
00A0 02A4   00410         INCF    FSR,F                   ;Go to next RAM location to clear
00A1 02E7   00411         DECFSZ  TEMP0,F                 ;Check to see if all clearing done
00A2 0A9F   00412         GOTO    CLEAR_LOOP
00A3        00413 OPTION_SETUP
00A3 0C09   00414         MOVLW   H'09'                   ;1100 1111
00A4 0002   00415         OPTION
00A5        00416 PORT_SETUP
00A5 0C0D   00417         MOVLW   H'0D'                   ;0000 1101
00A6 0006   00418         TRIS    GPIO
00A7 0C3D   00419         MOVLW   H'3D'                   ;0011 1101 -- Initial GPIO values
00A8 0026   00420         MOVWF   GPIO
00A9        00421 GET_TEMPERATURE
00A9 0974   00422         CALL    DOUT_LOW                ;Issue reset pulse for 480-960uS
00AA 096D   00423         CALL    DLONG_DELAY             ;Wait the required time
00AB 0971   00424         CALL    DOUT_HIGH               ;Set line high
00AC 0606   00425 GT_1    BTFSC   DALLAS                  ;Wait for presense pulse
00AD 0AAC   00426         GOTO    GT_1
00AE 0706   00427 GT_2    BTFSS   DALLAS                  ;Wait for presense over
00AF 0AAE   00428         GOTO    GT_2
00B0 096D   00429         CALL    DLONG_DELAY             ;Allow for DS1820 to recover
00B1 0CCC   00430         MOVLW   H'CC'                   ;Get 'skip ROM' command ready
00B2 0985   00431         CALL    DSEND
00B3 0C44   00432         MOVLW   H'44'                   ;Get 'convert T' command ready
00B4 0985   00433         CALL    DSEND
00B5 0067   00434         CLRF    TEMP0                   ;Set up for 500mS delay to allow for
00B6 0068   00435         CLRF    TEMP1                   ;       temperature conversion
00B7 0C03   00436         MOVLW   H'03'
00B8 0029   00437         MOVWF   TEMP2
00B9 02E7   00438 GT_3    DECFSZ  TEMP0,F
00BA 0AB9   00439         GOTO    GT_3
00BB 02E8   00440         DECFSZ  TEMP1,F
00BC 0AB9   00441         GOTO    GT_3
00BD 02E9   00442         DECFSZ  TEMP2,F
00BE 0AB9   00443         GOTO    GT_3
00BF 0974   00444         CALL    DOUT_LOW                ;Issue reset pulse for 480-960uS
00C0 096D   00445         CALL    DLONG_DELAY             ;Wait the required time
00C1 0971   00446         CALL    DOUT_HIGH               ;Set line high
00C2 0606   00447 GT_4    BTFSC   DALLAS                  ;Wait for presense pulse
00C3 0AC2   00448         GOTO    GT_4
00C4 0706   00449 GT_5    BTFSS   DALLAS                  ;Wait for presense over
00C5 0AC4   00450         GOTO    GT_5
00C6 096D   00451         CALL    DLONG_DELAY             ;Allow for DS1820 to recover
```

```
00C7 0CCC   00452         MOVLW   H'CC'                ;Get 'skip ROM' command ready
00C8 0985   00453         CALL    DSEND
00C9 0CBE   00454         MOVLW   H'BE'                ;Get 'read scratchpad' command ready
00CA 0985   00455         CALL    DSEND
00CB 0978   00456         CALL    DRECEIVE             ;Get 1 byte
00CC 020E   00457         MOVF    COM_REG,W
00CD 002C   00458         MOVWF   TEMP_LO
00CE 0978   00459         CALL    DRECEIVE             ;Get 2nd byte
00CF 020E   00460         MOVF    COM_REG,W
00D0 002B   00461         MOVWF   TEMP_HI
00D1 0974   00462         CALL    DOUT_LOW             ;Stop communications -- issue reset
00D2 096D   00463         CALL    DLONG_DELAY          ;Wait the required time
00D3 0971   00464         CALL    DOUT_HIGH            ;Set line high
00D4 0606   00465 GT_6    BTFSC   DALLAS               ;Wait for presense pulse
00D5 0AD4   00466         GOTO    GT_6
00D6 0706   00467 GT_7    BTFSS   DALLAS               ;Wait for presense over
00D7 0AD6   00468         GOTO    GT_7
00D8 096D   00469         CALL    DLONG_DELAY          ;Allow for DS1820 to recover
00D9        00470 PLAY_SOUND
00D9 094F   00471         CALL    CONVERT_TEMP         ;Convert to Farenheit
00DA 060A   00472         BTFSC   FLAG0,COLD           ;See if play too cold sound
00DB 0AEB   00473         GOTO    TOO_COLD
00DC 062A   00474         BTFSC   FLAG0,HOT            ;See if play too hot sound
00DD 0AEE   00475         GOTO    TOO_HOT
00DE 0944   00476         CALL    BIN2BCD_SMALL        ;Convert to BCD so know what sounds
00DF 0C04   00477         MOVLW   H'04'                ;Get message position for tens
00E0 009F   00478         SUBWF   MSD,W
00E1 003C   00479         MOVWF   MESSAGE1
00E2 021E   00480         MOVF    LSD,W                ;See if a 0 in 1s position
00E3 0743   00481         BTFSS   STATUS,Z             ;If Z set then a 0
00E4 0AE7   00482         GOTO    PLSO1
00E5 0C0A   00483         MOVLW   H'0A'                ;Set up to play "degrees'
00E6 003E   00484         MOVWF   LSD
00E7 0C05   00485 PLSO1   MOVLW   H'05'                ;Get message position for ones
00E8 01DE   00486         ADDWF   LSD,W
00E9 003D   00487         MOVWF   MESSAGE2
00EA 0AF1   00488         GOTO    SOUND_LOOP1
00EB        00489 TOO_COLD
00EB 0C10   00490         MOVLW   H'10'                ;Set up to play too cold messege
00EC 003D   00491         MOVWF   MESSAGE2
00ED 0B08   00492         GOTO    SOUND_LOOP2
00EE 0C11   00493 TOO_HOT MOVLW   H'11'                ;Set up to play too hot messege
00EF 003D   00494         MOVWF   MESSAGE2
00F0 0B08   00495         GOTO    SOUND_LOOP2
00F1        00496 SOUND_LOOP1
00F1 0963   00497         CALL    ISD_RESET
00F2 0526   00498         BSF     A0                   ;Set up to scan through messages
00F3 0000   00499         NOP
00F4 021C   00500 SL1_1   MOVF    MESSAGE1,W           ;See if at message position
00F5 0643   00501         BTFSC   STATUS,Z             ;If Z set then at position
00F6 0B01   00502         GOTO    SL1_4
00F7 0486   00503         BCF     CE                   ;Go to first message
00F8 0000   00504         NOP
00F9 0586   00505         BSF     CE
00FA 0000   00506         NOP
00FB 0646   00507 SL1_2   BTFSC   EOM                  ;Wait for EOM to go low
00FC 0AFB   00508         GOTO    SL1_2
00FD 0746   00509 SL1_3   BTFSS   EOM                  ;Wait for EOM to go back high
00FE 0AFD   00510         GOTO    SL1_3
00FF 00FC   00511         DECF    MESSAGE1,F           ;See if at message yet
0100 0AF4   00512         GOTO    SL1_1                ;Not at message yet
0101 0426   00513 SL1_4   BCF     A0                   ;Turn on current message
0102 0000   00514         NOP
0103 0486   00515         BCF     CE                   ;Enable message to play
0104 0000   00516         NOP
0105 0586   00517         BSF     CE                   ;Turn off play
```

```
0106 0646   00518 SL1_5   BTFSC   EOM                     ;Ensure messege over
0107 0B06   00519         GOTO    SL1_5
0108        00520 SOUND_LOOP2
0108 0963   00521         CALL    ISD_RESET
0109 0526   00522         BSF     A0                      ;Set up to scan through messages
010A 0000   00523         NOP
010B 0486   00524 SL2_1   BCF     CE                      ;Go to first message
010C 0000   00525         NOP
010D 0586   00526         BSF     CE
010E 0000   00527         NOP
010F 0646   00528 SL2_2   BTFSC   EOM                     ;Wait for EOM to go low
0110 0B0F   00529         GOTO    SL2_2
0111 0746   00530 SL2_3   BTFSS   EOM                     ;Wait for EOM to go back high
0112 0B11   00531         GOTO    SL2_3
0113 02FD   00532         DECFSZ  MESSAGE2,F              ;See if at message yet
0114 0B0B   00533         GOTO    SL2_1                   ;Not at message yet
0115 0426   00534         BCF     A0                      ;Turn on current message
0116 0000   00535         NOP
0117 0486   00536         BCF     CE                      ;Enable message to play
0118 0000   00537         NOP
0119 0586   00538         BSF     CE                      ;Turn off play
011A        00539 MAIN_END
011A 0B1A   00540         GOTO    MAIN_END                ;See if should do some more
            00541 ;*******************************************************************************
            00542 ;
            00543 ;
            00544 ;*******************************************************************************
            00545 ;End of code indicator
            00546         END

MEMORY USAGE MAP ('X' = Used,  '-' = Unused)

0000 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0040 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0080 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
00C0 : XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
0100 : XXXXXXXXXXXXXXXX XXXXXXXXXX----- ---------------- ----------------

All other memory blocks unused.

Program Memory Words Used:   283
Program Memory Words Free:   229


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