Keywords: MAXQ7667 Transducer

APPLICATION NOTE 4398

MAXQ7667 Transducer Diagnostics

Jun 02, 2009

Abstract: This application note explains how to use the MAXQ7667 smart SoC (system-on-a-chip) to diagnose the resonant frequency and damping characteristics of an ultrasonic transducer. The test setup is described and data presented. Software code is provided.

Introduction

This article shows how the MAXQ7667 smart SoC (system-on-a-chip) can quantify the resonant frequency and damping characteristics of an ultrasonic transducer. These capabilities are useful to diagnose transducer modules, to optimize system performance, and to assist in calibration during manufacturing. This application note is not a general "How to" document intended to work on all systems. Instead, the article assumes a detailed knowledge of a specific transducer model and how it performs in the conditions of interest. With that information, the user may be able to significantly streamline the tests shown here. These tests assume, finally, that any reflective targets in the transducer beam are more than a foot away from the transducer.

Test Setup

All the data in this article was taken by a MAXQ7667 in an EV (evaluation) kit. The data was transferred to a PC through the RS-232 port on the EV kit, and then plotted using an Excel spreadsheet. The transducer used was the 40kHz, 400EP250 that came with the EV kit. This transducer was mounted horizontally on the board, which simplified the task of adding foreign material to its surface in order to affect the transducer's performance. Three transducer conditions were used in each test: a clean dry transducer that represented normal operation; a transducer with water sitting on its face; and a transducer with a small piece of malleable putty on its face.

Figures 1 through **3** show that these different conditions cause significantly different transducer characteristics. Water on the face of the transducer lowers the resonant frequency and reduces damping. Putty has the opposite affect; it increases the resonant frequency and dramatically increases the damping.

Damping Test

The damping test measures how long the transducer resonates. This is a simple test that excites the transducer with one short pulse and then monitors the transducer's output to see how quickly the signal decays. Figures 1 through 3 show the damping with the echo receive path set to three different frequencies. All three figures easily show the severe damping caused by the putty. Finally, when looking for subtle changes in damping, the differences among the figures also reveal the importance of the receiver frequency.

Because this is a simple test, it can easily be run at more than one frequency. When testing at multiple frequencies, it is best if the width of the excitation pulse remains relatively constant. Using a single, narrow excitation pulse of consistent duration eliminates differences that would be caused by putting varying amounts of energy into the transducer, or by pumping the transducer at its resonant frequency.

In Figures 1, 2, and 3 the LPF (lowpass filter) was read at 25µs intervals. The transducer was excited with a single pulse approximately 6µs

wide. Note that this test can be run in less than 3ms, if only one frequency is used and if the PLL is already at the desired frequency. There is plenty of time between measurements to look for peak values and detect when the signal has decayed to a given percentage of the peak signal. If more than one frequency is used, then additional time will be needed for the additional measurements. There will also be additional time required for the PLL to settle at the new frequency.

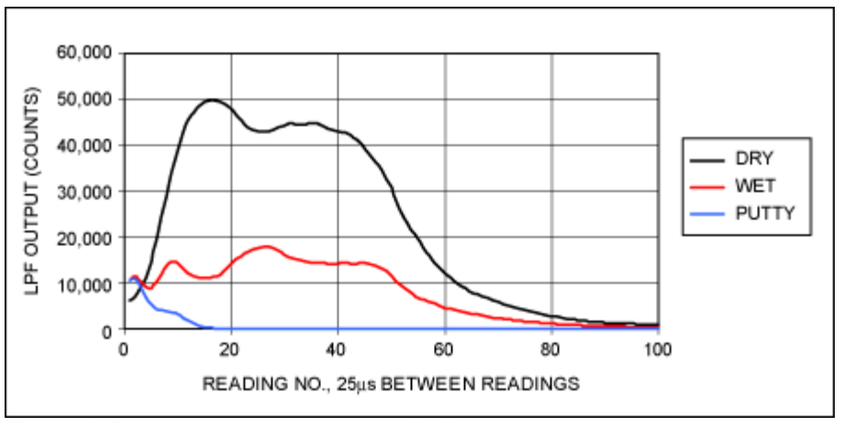


Figure 1. Damping measured with a 40kHz center-frequency filter.

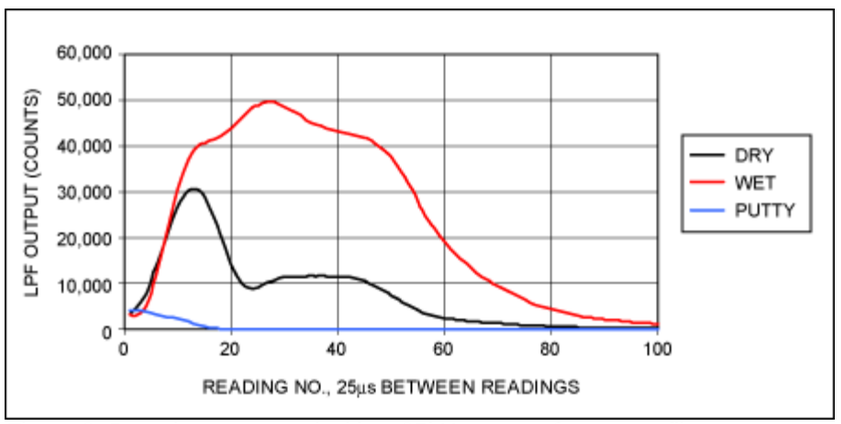


Figure 2. Damping measured with a 35kHz center-frequency filter.

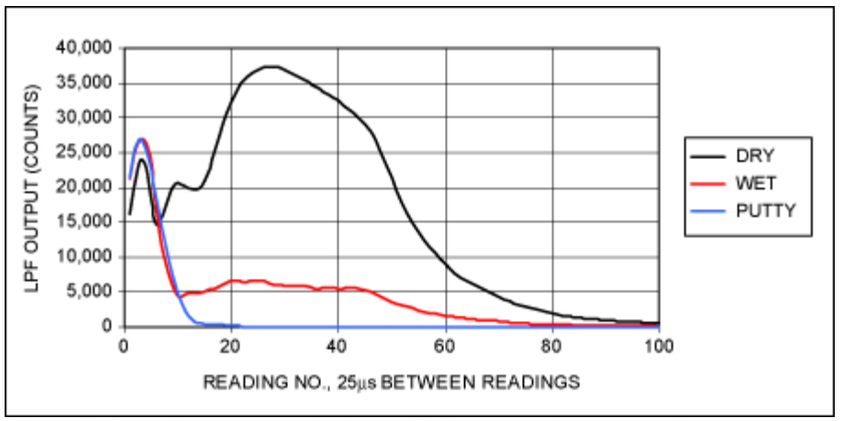


Figure 3. Damping measured with a 45kHz center-frequency filter.

Frequency Sweep

The frequency sweep will identify the transducer's resonant frequency. **Figure 4** clearly shows the effects of foreign materials (the three different test conditions) on the resonant frequency of the transducer. To gather the data for Figure 4, the transducer was pulsed 51 times as the receiver's center frequency was swept from 30kHz to 50kHz in steps of 400Hz. For each of the 51 frequencies, the same time interval was used between excitation and reading the LPF's output. Note that there is an important caveat with this diagnostic: you must use the correct interval between excitation and reading the LPF. This interval must be long enough to allow the echo receive path to come out of saturation, but short enough that the transducer is still producing a strong signal. It is likely that a single fixed interval will not work for all conditions. In Figure 2, for example, it is evident that if the transducer has putty on it, the signal will completely decay before a normal transducer would even come out of saturation. Because of this variation, the time from excitation to LPF measurement must be dynamic. Fortunately a damping test (see below) can easily determine a good time.

To collect the data for Figure 4 (i.e., LPF data counts), a damping test was first run at three different frequencies, and the time interval from excitation until the signal decayed to half of the peak value was saved. The longest time interval among the three tests was then used as the time interval in the frequency sweep test. This process determines a time interval that is appropriate for any transducer condition that still produces a good signal.

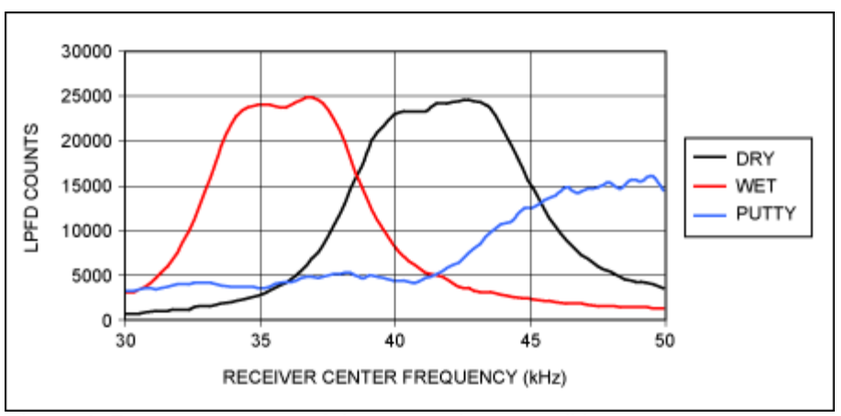


Figure 4. Frequency sweep of three transducer conditions.

A brief summary will be useful here. To determine the excitation-to-measurement interval, at least one damping test is needed before sweeping the frequency. Then the frequency sweep test takes much longer than the damping test. The frequency is stepped through many values, and the frequency must settle before exciting the transducer and taking the corresponding measurement. In this case the entire test

took about 650ms. The test time can be reduced by using a less conservative time for PLL frequency settling, by reducing the number of frequencies tested, and by reducing the number of damping tests. These changes may provide less precise information about the transducer, but the information could still be adequate for the application.

The data in Figure 4 shows a relatively wide resonance band for the transducer. The width of this band is more closely related to the bandwidth of the MAXQ7667's bandpass filter than to the performance of the transducer. Nonetheless, this test is still useful for identifying the center frequency of the transducer.

Simplifying the Test Conditions

The charts in this application note are easy for designers to interpret, but rather cumbersome for automated diagnostics. To simplify the process, one can reduce this data to a few numerical values for comparison with reference values stored in a table. The reference values can even be indexed by temperature. Temperature is easily obtained by using the SAR ADC and a thermistor.

The following code (**Appendix A**) was used to capture the data in Figure 4. This code also provides three pieces of data that can be used to define the condition of the transducer. These values are:

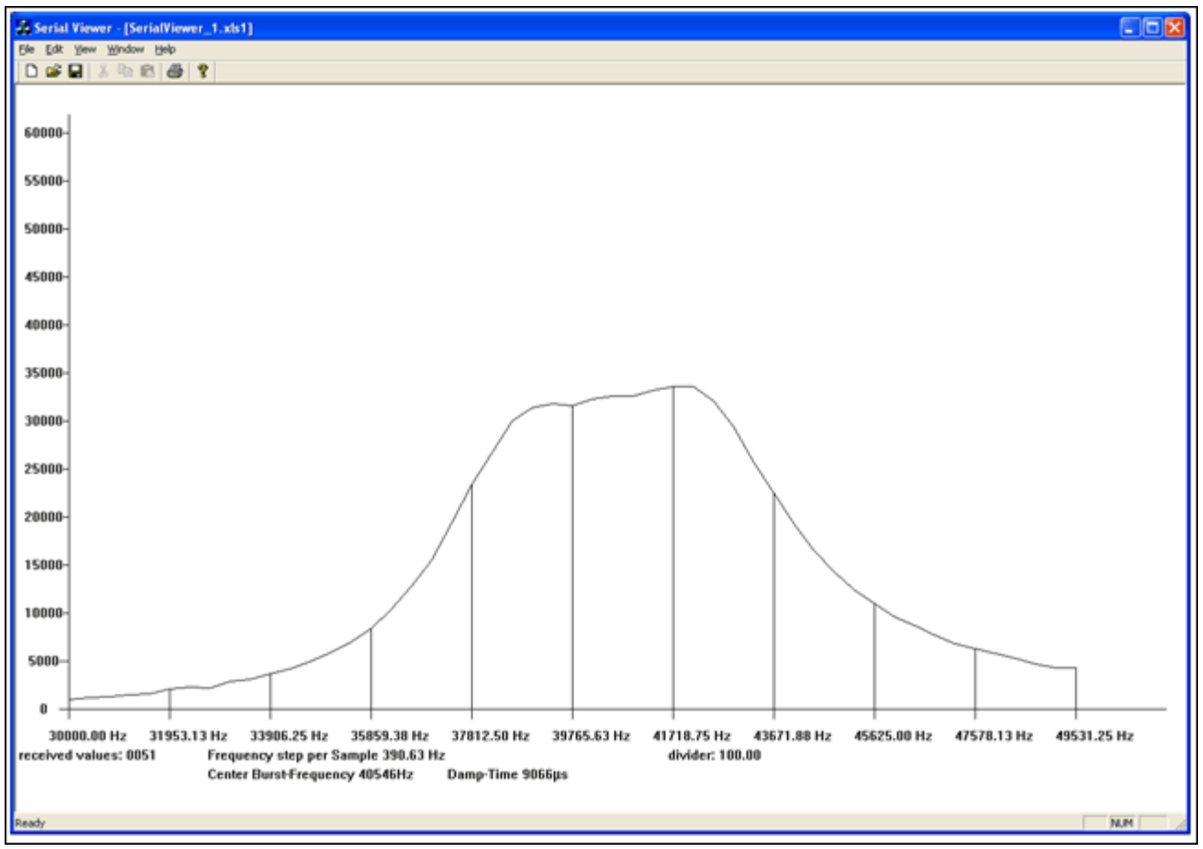
1. Peak value of the LPF during a damping test at the resonant frequency.
2. Damping time, or the time from excitation until the signal decays to half of the peak value.
3. Center frequency or resonant frequency. This value is expressed as the setting for the PLL and is the average of the lower PLL setting, where the signal is 70% of the peak signal, and the upper PLL setting, where the signal is 70% of the peak signal. The decision to use 70% was arbitrary.

Table 1 shows this data for each of the three transducer conditions. From this data a diagnostic program could easily distinguish among transducer conditions.

Table 1. Summary Data for Each Transducer Condition

	Peak LPF Reading (Counts)	Damping Time (µs)	Center Frequency (PLL)
Dry	24477	1394	290
Wet	24740	1435	150
Putty	15854	0323	430

You can download a complete software project for the IAR™ compiler 2.12A. If you connect COM1 of your PC to the RS-232 port of the EV kit, a PC program called, Transducer_Calibrate_115k2.exe, will show (**Figure 5**) the resonance curve over frequency and the evaluated resonance frequency.



More detailed image (PDF, 325kB)

Figure 5. Example output graph.

Appendix A. Software Example

MAXQ7667 Tranducer Calibration File

```
// This routine measures the transducer damping at the specified frequency (PLLfreq).
 // The system timer is used to measure when the LPF output (LPFD) has dropped to 1/2 the peak value.
 // This is the settling time that will be used later when doing the frequency sweep.

  unsigned damping_half_time (unsigned PLLfreq, unsigned pulse_width)
  {
          unsigned short i;
          unsigned short peak = 0;
          unsigned short half_peak = 0;
          unsigned short temp = 0;

          SCNT_bit.STIME = 0;           // Make sure system timer is off.
          STIM = 0;                     // Clear the system timer.
          SCNT_bit.STDIV = 4;           // Set system timer prescale divider to 16 (1µs per cycle).

          PLLF_bit.PLLF = PLLfreq;      // Set the PLL frequency.
          BPH = pulse_width;            // Pulse width = BPH/(receive frequency * 400) when BDIV = 0xC.
          usWaitTimer2(10000);          // Let the PLL settle for 10ms.
          SCNT_bit.STIME = 1;           // Start the system timer.

          BPH_bit .BSTT = 1;            // Send a burst.
          usWaitTimer2(50);             // Wait for LPF to partially settle.
```

```
          RunTimer0_us(20);             // Start timer # 1 with a reload time of every 20µs.

          for (i = 0; i < 200; i++)
          {
              temp = LPFD;                                          // Read the output of the lowpass
filter.
              if (temp > 0x2000 && temp > peak) {peak = temp;}    // Save the peak value if it is
greater than 2000.
              if (temp < peak/2 && half_peak == 0)    // If LPFD is less than half the peak value and
the half peak time
              {                                                           // has not been set, then
                half_peak = STIM;                                   // save the time it took to reach half
the peak value.
                i = 101;                                            // Half peak found so exit the loop.
              }
              while (T2CNB0_bit.TF2 == 0) {}                        // Wait for timer # 0.
              T2CNB0_bit.TF2 = 0;                                   // Clear flag.
          }
          T2CNA0_bit.TR2 = 0;                                       // Stop timer # 0.

          return half_peak;
   }

void main()
{
        unsigned short i = 0;
        unsigned short peak = 0;
        unsigned short first70 = 0;
        unsigned short second70 = 0;
        unsigned short center_pllf;

        unsigned short wait2measure;
        unsigned short halfpeak;


    init();


    //
*****************************************************************************************************
    //
*****************************************************************************************************
    // Configuration settings

      echo_receive_gain(0);           // Set receiver to minimum gain (allowed values 0-31)

      Burst_Clock_Divider = 400;      // for calculating the burst-frequency in PC-Application.
      burst_setup(BURST_CLK_PLL, BURST_PULSE_1, BURST_DIV_400, 0, PLL_CLOCK_16MHZ, 0);

      step_size=10;                   // Sets the step-size (steps go from 0 to 511).

    // END configuration settings.
    //
*****************************************************************************************************
    //
*****************************************************************************************************

    while(1)
    {
          // Use the "damping_half_time" routine to measure the time in µs that it takes for the
          // ringing to drop to half of the peak value.  Do this at more than one frequency so that
          // one of the frequencies will be within range of the transducer.
          wait2measure = damping_half_time(128, 88);   //Measure damping time at 35kHz.
          halfpeak = damping_half_time(256, 101);     //Measure damping time at 40kHz.
          if (wait2measure < halfpeak) {wait2measure = halfpeak;} // Save the longest time.
          halfpeak = damping_half_time(384, 113);     //Measure damping time at 45kHz.
          if (wait2measure < halfpeak) {wait2measure = halfpeak;} // Save the longest time.

          // Repeatedly pulse the transducer with a constant width pulse while sweeping the receiver
```

```
          // frequency.  Use the half-time value from the damping test (wait2measure) for the interval
          // between pulse transmission and reading LPFD (lowpass filter data).
          PLLF_bit.PLLF = 0;         // Start the sweep at 30kHz.
          BPH = 77;                  // Pulse ~6.3µs.  This value is easy to maintain over frequency.
          peak = 0;
          number_of_steps = 0;
          do    // Sweep from 30kHz to 50kHz with step_size * 39.062500kHz.
          {
              usWaitTimer2(5000);              // Wait 5ms for the frequency to settle.
              STIM = 0;                         // Reset the system timer.  It controls the Pulse to LPF
read time.
              BPH_bit .BSTT = 1;                // Send burst.
              while(STIM < wait2measure) {}    // Wait the specified amount of time for the ringing to
dampen.
              lpfdata[number_of_steps]= LPFD;  // Store the LPF reading.

              if(lpfdata[number_of_steps]> peak) {peak = lpfdata[number_of_steps];}   //Save the peak
value.

              number_of_steps++;
              PLLF_bit.PLLF = number_of_steps * step_size;          // Increase the frequency.
              BPH = 77 + number_of_steps;                           // Increase the duty cycle to
maintain pulse width.

          } while (PLLF < 512-step_size);

          // Find the center frequency based on the average of the two frequencies that have a
          // LPFD reading that is 70% of the peak reading.
          for (i = 0; i < number_of_steps; i++)
          {
                if (lpfdata[i] > peak*0.7)
                {
                    first70 = i;
                    i = number_of_steps;
                }
          }

          for (i = number_of_steps; i >0; i--)
          {
                if (lpfdata[i] > peak*0.7)
                {
                    second70 = i;
                    i= 1;
                }
          }

          i = (first70 + second70)/2;   // i = the loop value at the center frequency.
          center_pllf = i*step_size;         // Set PLLF_bit.PLLF to this value for the resonant
frequency.

          //Remeasure damping using the center frequency.
          damp_time = damping_half_time(center_pllf, 75);

         // At this point there are three valuable pieces of information about the transducer.
         // Peak = the peak value from the frequency sweep.
         // center_pllf = the PLLF setting at the resonant frequency.
         // damp_time = time for the resonance to decay to 1/2 the peak value.

          center_burst_frequency =   16000000/Burst_Clock_Divider*(center_pllf+768)/1024;

      SendData();

  } // While(1)
} // End Main
```

Related Parts		
MAXQ7667	16-Bit, RISC, Microcontroller-Based, Ultrasonic Distance-Measuring System	Free Samples

More Information
For Technical Support: http://www.maximintegrated.com/support
For Samples: http://www.maximintegrated.com/samples
Other Questions and Comments: http://www.maximintegrated.com/contact

Application Note 4398: http://www.maximintegrated.com/an4398
APPLICATION NOTE 4398, AN4398, AN 4398, APP4398, Appnote4398, Appnote 4398