Resonant Trimming Sequence

Maximilian Albrecht

ABSTRACT

This application report presents an efficient and precise method on how to achieve the desired resonant frequency of configuring the trim array with only a few iterations and measuring the resonant frequency. First, the minimum and maximum resonant frequency is measured; from this data a capacitance gradient is derived, which allows a good prediction of the required capacitance. The calculated capacitance is set, followed by a measurement step. Due to production tolerances, the resonant frequency could still be incorrect and require some further small iterations, which are described and discussed in this document.

Contents

List of Figures

1 Introduction

For optimized energy and data transfers, TI's analog front ends embed configurable capacitor arrays to adjust the resonant frequency of the external parallel resonant circuit consisting of a low frequency antenna and parallel resonant capacitor.

The capacitor array consists of several capacitors of different outputs that can be added to the parallel resonant circuit through a switch on each capacitor. This configuration is persistent also without supply voltage, assuring a trimmed resonant circuit even without battery.

To take advantage of the trim array, an efficient algorithm is required to adjust the capacitance to match the required target resonant frequency. The general process for this task is to enable and disable certain capacitors and subsequently measure the resulting resonant frequency. These steps are repeated until the desired result is achieved.

A variety of methods for this is possible. An easy but possibly time consuming approach could be to disable all capacitors and enable them one-by-one until the resonant frequency is in the desired range. Another method could be to start with half of the available capacitance enabled and then check if the current resonant frequency is below or above the desired frequency. Then, half of the remaining capacitance are enabled if it is below or half of the enabled capacitance are disabled if it is above the desired resonant frequency. This can be repeated until the desired capacitance is configured. A precise and efficient method is described in Section 3.

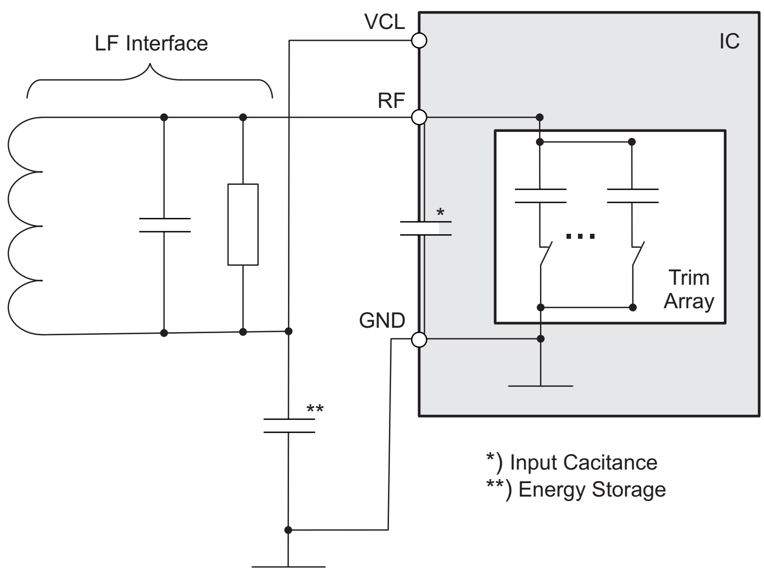
2 Hardware Description

2.1 The Capacitor Array

2.1.1 Capacitance Range and Dimensioning

The capacitor trimming array has a given capacitance range and consists, for example, of seven capacitors. The maximum capacitance of the array is equivalent to the sum of the capacitance of all capacitors. Each capacitor has half the size of the next larger capacitor.

That implies that the trim array has a capacitance resolution equivalent to the capacitance of the smallest capacitor.



2.1.2 Capacity Adjustment Using the Trim Byte

Those capacitors are controlled by the Trim Byte. The seven least significant bits determine which capacitors are enabled. They are ordered by capacitance and the least significant bit controls the smallest capacitor; therefore, the capacitance of the trimming array is proportional to the numeric value of the Trim Byte. For example, the Trim Byte = 40h adds twice the capacitance of the Trim Byte = 20h.

Examples

Examples	*Assuming a trimming array of seven capacitors with a maximum capacitance of 75 pF*		
Capacitance	*Trim Byte (Hex)*	*Trim Byte (Binary)*	
00.6 pF	01h	0000 0001b	Only the smallest capacitor is enabled
18.9 pF	20h	0010 0000b	Only the second largest capacitor is enabled
37.8 pF	40h	0100 0000b	Only the largest capacitor is enabled
75.0 pF	7Fh	0111 1111b	All capacitors are enabled

The trim byte that is equivalent to the desired capacitance can be calculated with this formula:

$$TrimByteValue = \frac{DesiredCapacitance}{RangeOfCapacitance} * TrimByteRange = \frac{C_{target}}{C_{max}} * 7\ Fh$$

3 Application

3.1 Description

This application report features a method of efficiently trimming the resonant frequency by calculating the “frequency shift per one capacitance unit”- gradient to determine a value for the Trim Byte, which is as close as possible to the optimized value. One “capacitance unit” is equivalent to the capacitance of the smallest capacitor in the array.

Due to non-ideal influences, it is necessary to repeatedly compare the current resonant frequency to the target frequency and slightly increase or decrease the capacitance until the best value is found. This is done in the Fine Trimming Sequence.

3.2 Flowcharts

3.2.1 Resonant Trimming Sequence

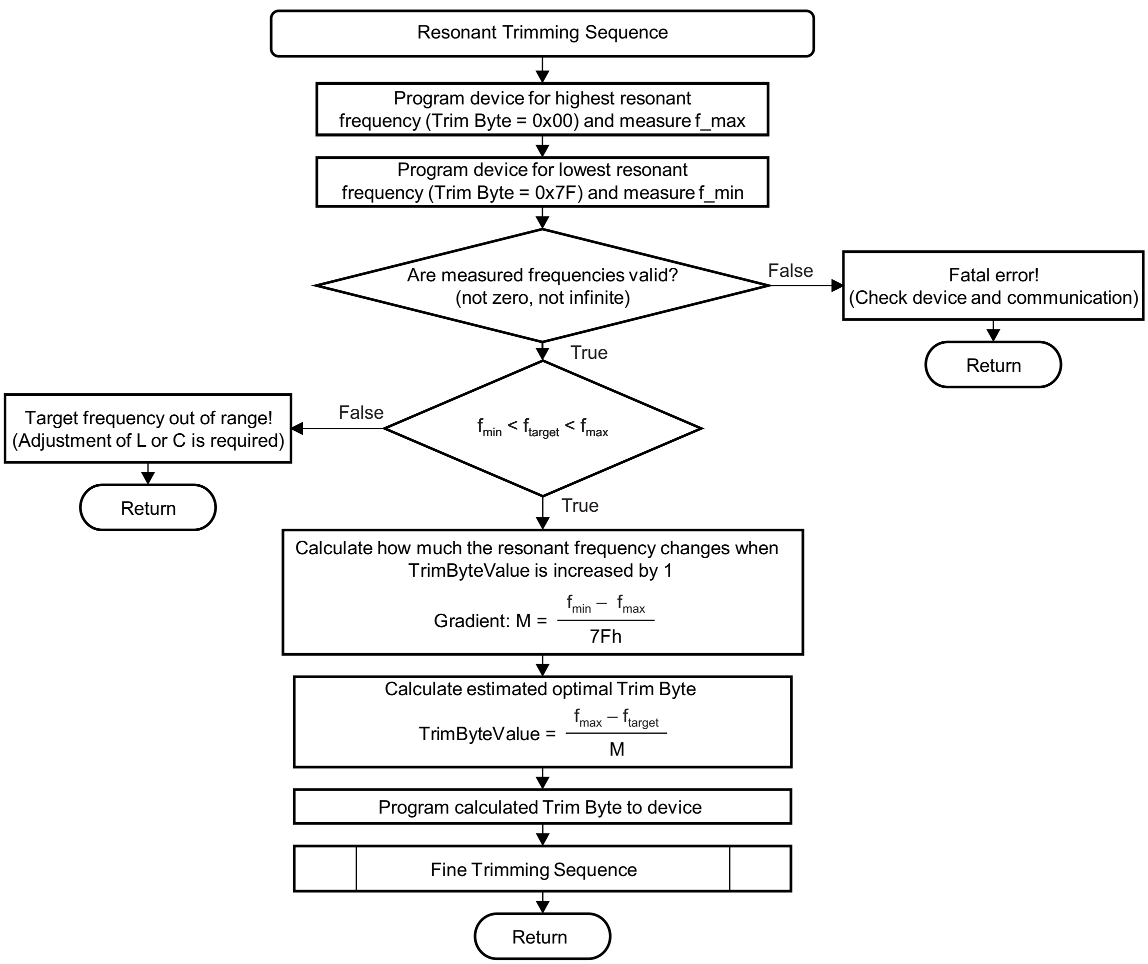


Figure 1. Resonant Trimming Flowchart

3.2.2 Fine Trimming Iteration

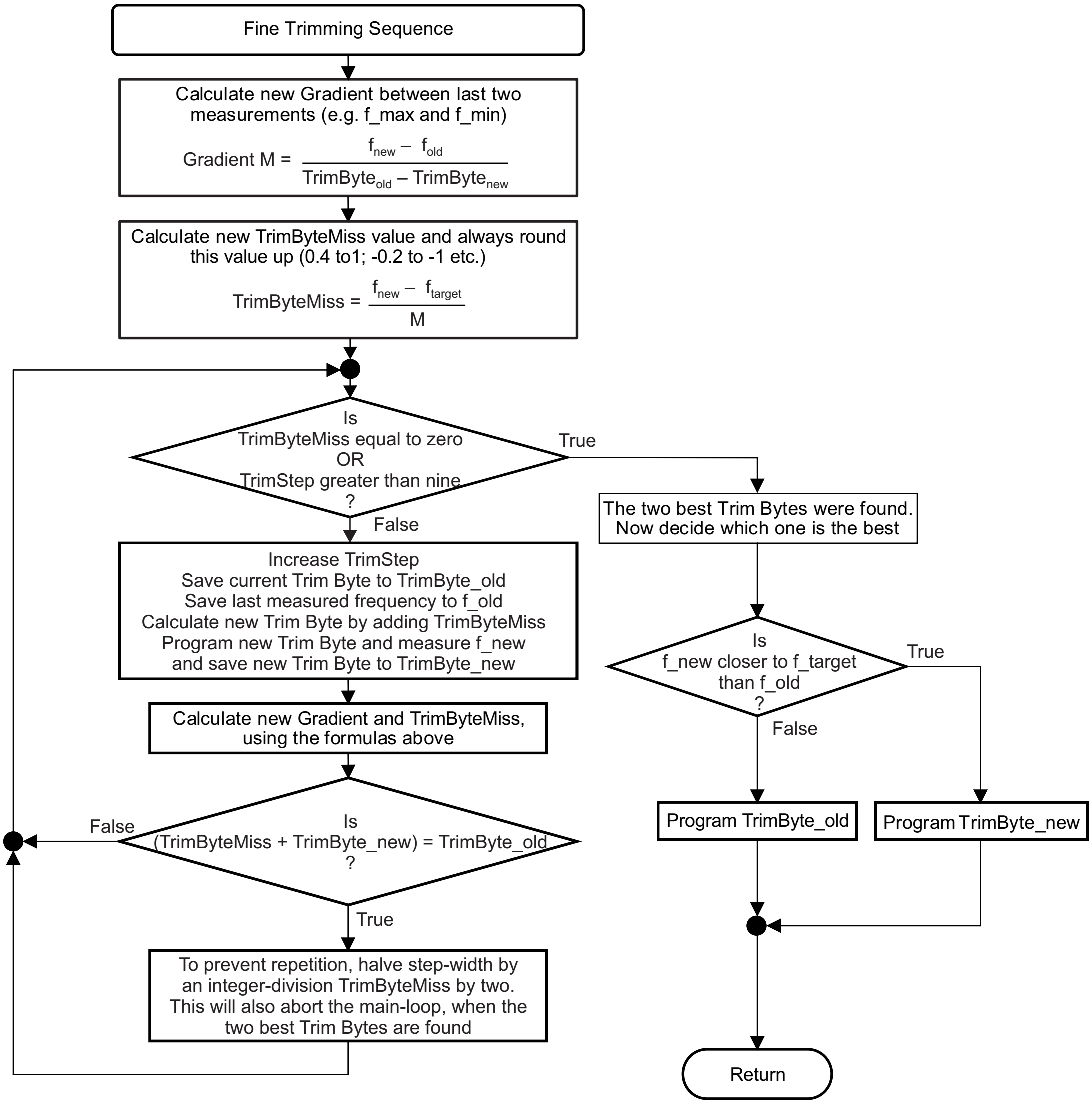


Figure 2. Fine Trimming Flowchart

3.3 Trimming a CRAID

The device used in the following example is a CRAID device with an LF antenna and a Trim Array with a maximum capacitance of 75 pF.

1. Define the desired target frequency.

   f_target = 134.2 kHz

2. Measure the maximum and minimum achievable frequency.

   f_max = 143.35 kHz (TrimByte=00h)
   f_min = 125.67 kHz (TrimByte=7Fh)

   Measured frequencies are neither zero, nor infinite. They are valid.
   f_target is in the available frequency range.
   Preconditions for successful trimming are met.

3. Calculate resolution of the Trim Byte (gradient) with the given formula:

$$M = \frac{f_{max} - f_{min}}{TrimByteRange} = \frac{143.35 - 125.67}{7Fh}\ kHz = 139\ \frac{Hz}{bit}$$

If the trim byte is decreased by 1, the resulting resonant frequency will increase by 139 Hz.

4. Calculate the computationally correct value for the Trim Byte, assuming an ideal linear characteristic.

$$TrimByteValue = \frac{f_{max} - f_{target}}{M} = \frac{143.35 - 134.20}{0.139} 66_{10} = 42h$$

5. Program *TrimByteValue* to the device and subsequently measure the resonant frequency.

   f_current = 132.45 kHz

   Now the *Fine Trimming Sequence* begins with the following parameters:

   - *f_old = f_max = 143.35 kHz*
   - *TrimByte_old = 00h*
   - *f_new = f_current = 132.45 kHz*
   - *TrimByte_new = 42h*

6. Calculate the new Gradient (this value slightly varies because of nonlinearities).

$$M = \frac{f_{new} - f_{old}}{TrimByte_{old} - TrimByte_{new}} = 165 \frac{Hz}{bit}$$

7. Calculate the new *TrimByteMiss* and round up the value.

$$TrimByteMiss = \frac{f_{new} - f_{target}}{M} = 10.6; after\ round\ up; -11$$

8. Count up *TrimStep*, save the old values and prepare the new values, since *TrimByteMiss* is not 0.

   - *TrimStep = 1*
   - *TrimByte_old = 42h*
   - *f_old = 132.45 kHz*
   - *TrimByte_new = TrimByte_old + TrimByteMiss = 66 – 11 = 55 = 37h*

9. Program *TrimByte_new* to the device and measure the frequency as *f_new*

   f_new = 134.13kHz

10. Calculate the new gradient *M* and *TrimByteMiss* like in steps 7 and 8.

    M = 153 [Hz / bit]
    TrimByteMiss = -0.45; after rounding up: -1

11. Start the next Trimming Step (similar to steps 9, 10, and 11), since (*TrimByte_new* + *TrimByteMiss*) is not *TrimByte_old* and *TrimByteMiss* is still greater than zero. After this is done, you have the following values:

    - *TrimStep = 2*
    - *TrimByte_old = 37h*
    - *f_old = 134.13 kHz*
    - *TrimByte_new = TrimByte_old + TrimByteMiss = 55 - 1 = 54 = 36h*
    - *f_new = 134.21 kHz*
    - *M = 80 [Hz / bit]*
    - *TrimByteMiss = 0.12; after rounding up : 1*

12. Know that the two best Trim Bytes (36h and 37h) are found, because (*TrimByte_new* + *TrimByteMiss*) is equal to *TrimByte_old*. *TrimByteMiss* is divided by two and due to the integer division is now zero and the main-loop will be aborted.

13. *f_new* is closer to the target frequency at 134.2 kHz than *f_old*, so *TrimByte_new* is recognized as optimal Trim Byte. The device is programmed to this value and is now trimmed as good as possible.