*TMS320 DSP*
*DESIGNER'S NOTEBOOK*

# *Extending Fixed-Point Dynamic Ranges*

*APPLICATION BRIEF: SPRA249*

*Alex Tessarolo*
*Digital Signal Processing Products*
*Semiconductor Group*

*Texas Instruments*
*January 1995*

TEXAS INSTRUMENTS

## IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain application using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

## TRADEMARKS

**CONTACT INFORMATION**


| | |
|---|---|
| US TMS320 HOTLINE | (281) 274-2320 |
| US TMS320 FAX | (281) 274-2324 |
| US TMS320 BBS | (281) 274-2323 |
| US TMS320 email | dsph@ti.com |

## Contents



## Examples

# Extending Fixed-Point Dynamic Ranges

## Abstract

In many advanced control problems such as state estimators, Kalman filters and some high Q filters, the dynamic range/accuracy of the coefficient can sometimes be beyond the range of a Q15 number while the data value can be typically represented as a Q15 number.

This document discusses how you can extend the fixed-point math dynamic range beyond the range of a Q15 number with a minimum of instructions.

# Design Problem

How can you extend the fixed-point math dynamic range beyond the range of a Q15 number with a minimum of instructions?

# Solution

In many advanced control problems such as state estimators, Kalman filters and some high Q filters, the dynamic range/accuracy of the coefficient can sometimes be beyond the range of a Q15 number while the data value can be typically represented as a Q15 number.

Aside from trying to dynamically scale the coefficients (to extract as much accuracy as possible) or trying to use floating point math, there is a technique that can perform 32-bit × 16-bit math at an effective 4 cycles per Tap and potentially 2 cycles per Tap for larger then 6th order systems (plus some fixed overhead of about 8-13 cycles).

The trick is to re-scale the numbers and represent the problem as an integer value + a fractional value. For example:

$$Y = 2.391456*X0 - 0.0235045*X1 + 0.000329758*X2 - 34.3392345*X3$$

In the above equation, the filter coefficients have a dynamic range exceeding a 16-bit Q15 number. If we re-scale the problem as follows:

$$Y = [1224.425472*X0 - 12.034304*X1 + 0.168836096*X2 - 17581.68806*X3]/512$$

And then allocate the following coefficient values:

$$Y = [(A0i+A0f)*X0 + (A1i+A1f)*X1 + (A2i+A2f)*X2 + (A3i+A3f)*X3]/512$$

Where:

A0i = 1224 = 04C8h
A0f = 0.425472 = 3676h (= 0.425476074)
A1i = –12 = FFF4h
A1f = –0.034304 = FB9Ch (= –0.034301758)
A2i = 0 = 0000h
A2f = 0.168836096 = 159Ch (= 0.168823242)
A3i = –17581 = BB53h
A3f = –0.68806 = A7EEh (= –0.688049316)

The problem then reduces to calculating the following:

$$Y = (A0i*X0 + A1i*X1 + A2i*X2 + A3i*X3) + (A0f*X0 + A1f*X1 + A2f*X2 + A3f*X3)$$

This is like calculating two filter banks. The above problem is coded in the example below:

*Example 1. Code Example*

```
; Assume:        X0,X1,X2,X3 = Q15 (-1 range 0.999053955)
;                Y = Q10 (-32 range +31.99902344)
;       Ymin-max = 2.391456 + 0.0235045 + 0.000329758 + 34.3392345
;                = +/- 36.75452476
;                Sat = 06000h
;                Round = 08000h
     SETC        OVM            ; Enable saturation.
     SETC        SXM            ; Enable sign extension.
     SPM         3              ; Set shift mode = -6
     LT          A0f
     MPY         X0             ; P = A0f*X0
     LTP         A1f            ; ACC = A0f*X0
     MPY         X1             ; P = A1f*X1
     LTA         A2f            ; ACC = ACC + A1F*X1
     MPY         X2             ; P = A2f*X2
     LTA         A3f            ; ACC = ACC + A2f*X2
     MPY         X3             ; P = A3f*X3
     LTA         A0i            ; ACC = ACC + A3f*X3
     SPM         0
     SACH        Temp,6         ; On C5X replace by BSAR 9
     LAC         Temp,1         ; ACC = ACC/512
     ; instruction.
     MPY         X0             ; P = A0i*X0
     LTA         A1i            ; ACC = ACC + A0i*X0
     MPY         X1             ; P = A1i*X1
     LTA         A2i            ; ACC = ACC + A1i*X1
     MPY         X2             ; P = A2i*X2
     LTA         A3i            ; ACC = ACC + A2i*X2
     MPY         X3             ; P = A3i*X3
     APAC                       ; ACC = ACC + A3i*X3
     ADDS        Round          ; Round result.
     ADDH        Sat            ; Saturate Y to Q10 value
     SUBH        Sat
     SUBH        Sat
     ADDH        Sat
     SACH        Y,1            ; Y = Q10 number.

; Cycles = 13 + 4n cycles (n = number of taps).

; Note: If saturation is not required, Cycles = 8 + 4n cycles
```

If the number of taps is greater then 6, then a RPT loop can be used for each bank and the effective cycles/tap can be approximately 2. The above technique is almost equivalent to a floating-point notation with a 4-bit exponent and a 16-bit mantissa.