Application Report
SLAA331A–September 2006–Revised May 2018

Wave Digital Filtering Using an MSP430™ MCU

MSP430 Applications

ABSTRACT

Digital filtering is an integral part of many digital signal processing algorithms. Digital filters are characterized as either recursive [infinite impulse response (IIR)] or non-recursive [finite impulse response (FIR)] filters. IIR filters require a lower order for the same set of specifications compared to FIR filters, while FIR filters provide a linear phase property. However, IIR filters, if not designed properly, tend to be unstable and use coefficients that are difficult to implement. This application report describes the design of a certain type of IIR digital filter with excellent stability properties, known as a Lattice Wave Digital Filter (LWDF), given a set of analog filter specifications. An elementary knowledge of filters is sufficient to understand the design of LWDF as discussed in this report. The complete design of the basic filters (low-pass, high-pass, and band-pass) is given with examples. The code provided in this application report implements these LWDFs on the MSP430™ CPU. Project collateral discussed in this application report can be downloaded from the following URL: www.ti.com/lit/SLAA331.

Contents

1 Introduction 2
2 Lattice Wave Digital Filters 3
2.1 Structures for the Adaptors 4
2.2 Implementing the LWDF 5
2.3 Different Forms of LWDF 6
2.4 LWDF Tool 7
2.5 Implementation of the LWDF 7
3 Real-Time Filter Examples on MSP430 MCUs 10
3.1 Example 1: Low-Pass Filter 10
3.2 Example 2: High-Pass Filter 14
3.3 Example 3: Band-Pass Filter 18
3.4 Summary 19
4 Code Details 20
5 Filter Testing Methodology 21
5.1 Input Samples 21
5.2 Assembly Calls 21
5.3 Plots 21
5.4 Use of the Assembly Function 21
6 References 22

List of Figures

1 Signal Flow Diagram of a Wave Digital Filter 3
2 Type 1 Adaptor Interconnections, $0.5 < \gamma < 1$, $\alpha = 1 - \gamma$ 4
3 Type 2 Adaptor Interconnections, $0 < \gamma \leq 0.5$, $\alpha = \gamma$ 4
4 Type 3 Adaptor Interconnections, $-0.5 \leq \gamma < 0$, $\alpha = |\gamma|$ 5
5 Type 4 Adaptor Interconnections, $-1 < \gamma < -0.5$, $\alpha = 1 + \gamma$ 5
6 Cascade of Lattice Wave Digital Filters 6
7 Bireciprocal Lattice Wave Digital Filters 7
8 Signal Flow Diagram of 9th-Order Chebyshev Low-Pass LWDF 11
9 Interconnections for 9th-Order Chebyshev Low-Pass LWDF 12

10 Frequency Response of 9th-Order Chebyshev Low-Pass LWDF 13
11 Frequency Response of 19th-Order Chebyshev Low-Pass LWDF 13
12 Response at Complementary Output of 9th-Order Chebyshev Low-Pass LWDF 14
13 Signal Flow Diagram of 11th-Order Butterworth High-Pass LWDF 15
14 Interconnections for 11th-Order Butterworth High-Pass LWDF 16
15 Frequency Response of 11th-Order Butterworth High-Pass LWDF 17
16 Frequency Response of 14th-Order Elliptical Band-Pass LWDF 19

List of Tables

1 Low-Pass Filter Parameters 10
2 High-Pass Filter Parameters 15
3 Band-Pass Filter Parameters 18

Trademarks

1 Introduction

Digital signal processing (DSP) and digital filtering have been known for over 50 years.[7] A DSP system consists of an analog-to-digital converter (ADC), the processor for digital signals (that is, numbers), and the digital-to-analog converter (DAC). The input sampling of the analog signal is performed with a fixed frequency (sampling rate f_{s1}). A similar process is during reconstruction of the analog output signal with a fixed frequency (sampling rate f_{s2}, usually equal to f_{s1}).

Finite impulse response (FIR) filters are frequently chosen in digital signal processing applications, because of their inherent stability, their linear phase property, and the relative ease with which they are implemented. However, the order of such filters is usually much larger when compared to an infinite impulse response (IIR) filter for similar performance. This increase in order directly results in increased memory requirements and execution times. FIR filter implementation in an embedded system (with a microcontroller) under ultra-low power consumption is of concern as the FIR filter consume more power than an IIR filter. Under these conditions, it is worthwhile to investigate ways to implement IIR filters that are stable and scalable.

A class of IIR filters that fits this description is the lattice wave digital filter (LWDF). The concept of wave digital filters was first proposed by Professor Alfred Fettweis in 1971. This filter exhibits excellent stability properties under several nonlinear operating conditions.[1][2] The LWDF is completely characterized by a set of coefficients (gamma) that have excellent dynamic range and low word-length requirements. The LWDF (if designed correctly) is also free from round-off and overflow conditions, making it an excellent choice for low-end microcontrollers.

Among different WDF structures, the LWDFs can be designed advantageously by means of a set of simple equations and iterations.[4] To further aid in reducing CPU load, Horner's scheme, which performs multiplication by using only shift and add operations, can be utilized to implement the filtering.[3] An application of this on the MSP430 CPU shows good filtering performance and fast execution times.

2 Lattice Wave Digital Filters

The LWDF is a cascade of all-pass sections of first or second order. The all-pass sections consist of delay elements called adaptors. In this design, two-port adaptors that have two inputs and two outputs are considered. Each adaptor includes a multiplier and three adders. The multipliers are the filter coefficients gamma that characterize the LWDF. Methods to easily calculate these coefficients from the design specifications are described in reference [4]. These coefficients are always in the range –1 to 1. Figure 1 shows the signal flow diagram of a generic LWDF, with the block T representing a unit sample delay.

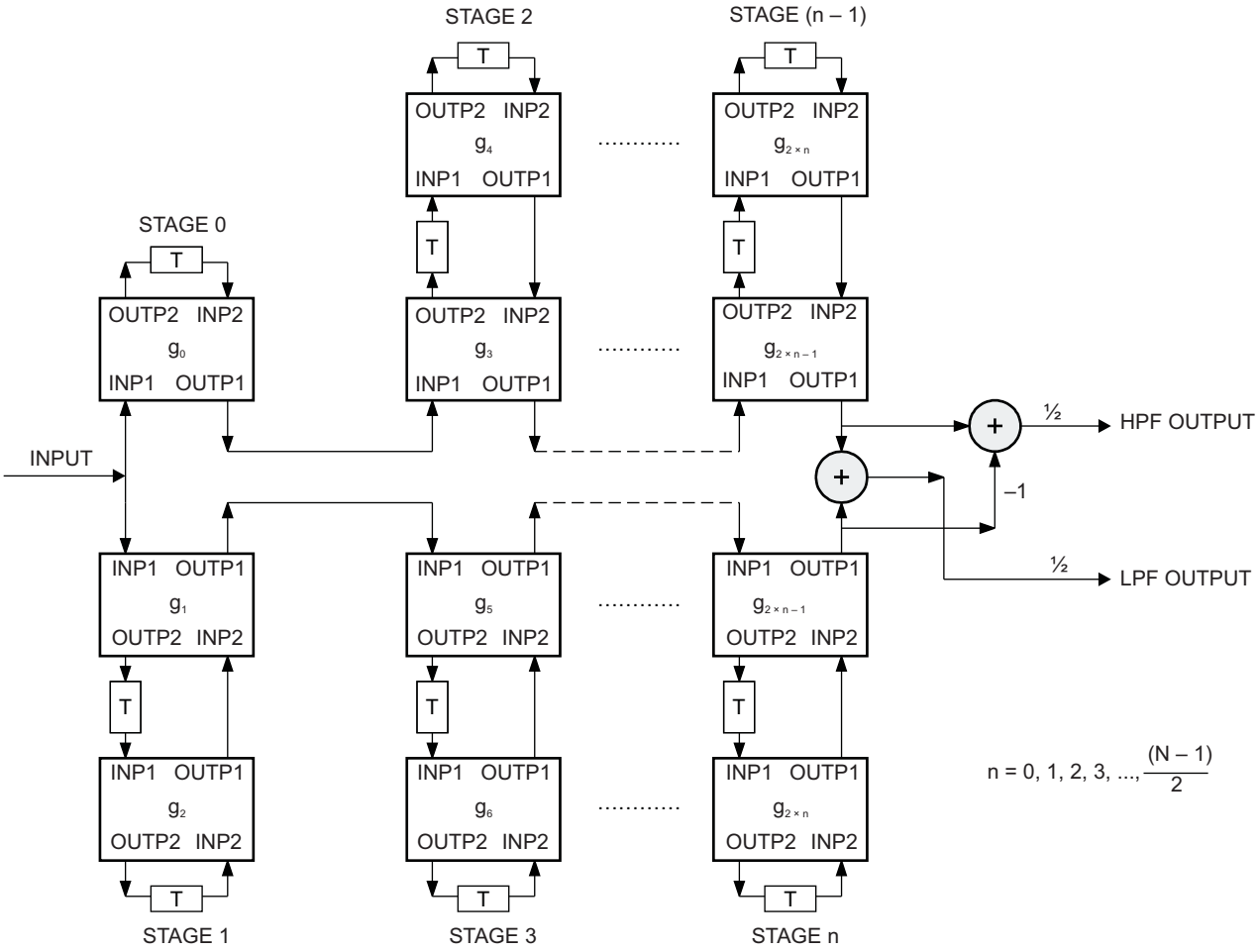


Figure 1. Signal Flow Diagram of a Wave Digital Filter

This diagram is common to all the low-pass and high-pass filters. The order for the low-pass (LP) and high-pass (HP) filters is always an odd number.[4] For any order N, there are (N+1)/2 stages and a maximum of N adaptors. The order N and the gamma coefficients (γ) in each adaptor vary with the filter specification, such as pass-band edge frequency, stop-band edge frequency, sampling frequency, pass-band ripple, and stop-band attenuation. In addition to the low-pass output, a complementary high-pass output is also available at no extra effort (see Figure 1). A band-pass filter (BPF) can be easily designed by the cascade of a high-pass and a low-pass filter. A band-reject filter (BRF) response is obtained at the complementary output of a band-pass filter structure.

2.1 Structures for the Adaptors

To further improve the amplitude scaling performance of the LWDF, different structure types for the adaptors are used, depending on the range of the gamma coefficient values. The entire coefficient range of –1 to 1 has been divided into four subranges, each of which is assigned a different structure. Each structure has a unique conversion of the gamma coefficient to a second coefficient, designated as the alpha coefficient. The alpha coefficient is always positive and in the range 0 to 0.5. Although there are numerous structures for these adaptors, for consistency, four widely used structures are described.[4] Figure 2 through Figure 5 show the structure and design equations for each of these four ranges.

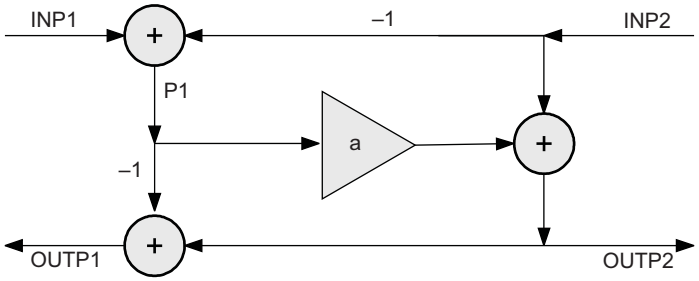


Figure 2. Type 1 Adaptor Interconnections, 0.5 < γ < 1, α = 1 – γ

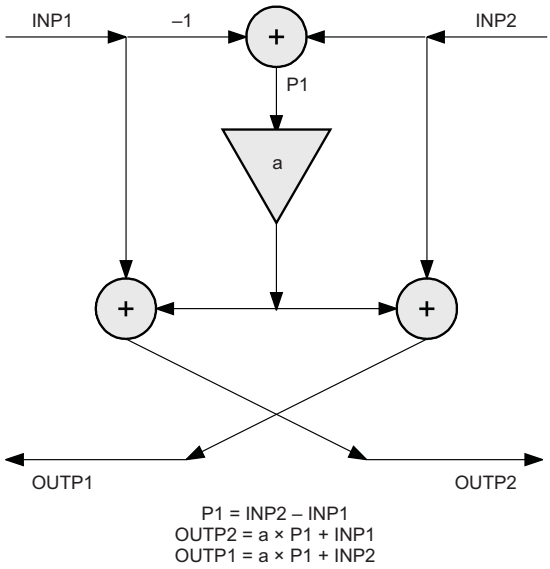


Figure 3. Type 2 Adaptor Interconnections, 0 < γ ≤ 0.5, α = γ

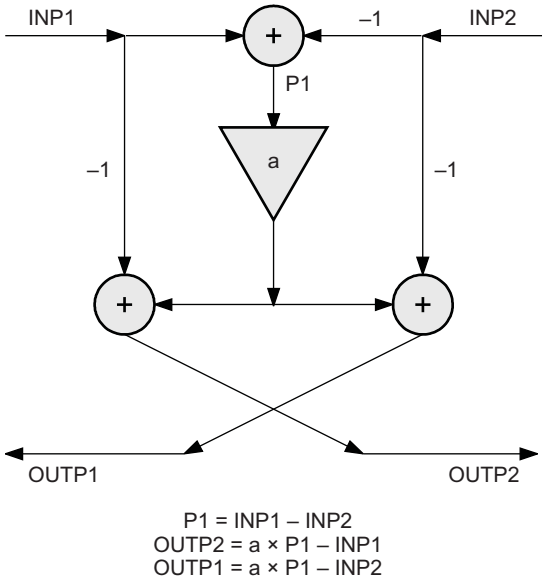


Figure 4. Type 3 Adaptor Interconnections, –0.5 ≤ γ < 0, α = |γ|

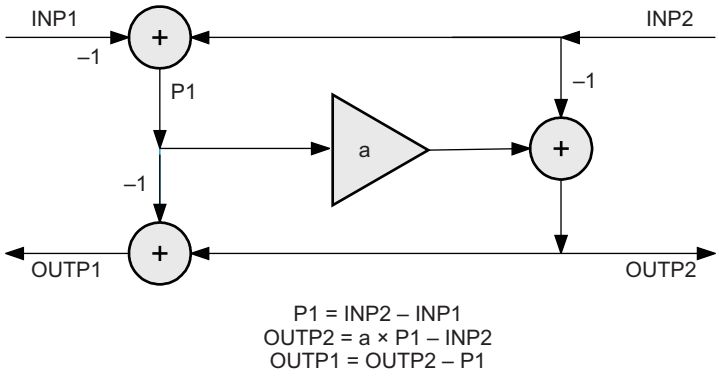


Figure 5. Type 4 Adaptor Interconnections, –1 < γ < –0.5, α = 1 + γ

Each structure consists of adders or subtractors and a multiplier. These structures form the signal flow diagram of each adaptor in the block diagram (see Figure 1).

2.2 Implementing the LWDF

Each adaptor structure is different, with only adders/subtractors and a multiplier. Even if a hardware multiplier is not available in the chosen MSP430 device, the multiply operations can be done efficiently using just shift and add operations. The MSP430 CPU can perform a register shift or an add operation in a single instruction cycle, making the entire implementation extremely efficient.[5] Section 3 gives examples of low-pass, high-pass, and band-pass filters on the MSP430 MCUs. Section 4 gives code details for each example. Section 5 gives the filter testing methodology. The MSP430 assembly and C codes are included in the zip file accompanying this report. Horner's algorithm can be made more efficient by using the canonical signed digit (CSD) representation, which reduces the number of add operations by grouping consecutive 1s in the alpha coefficients.[3]

Because the LWDF is completely characterized by its coefficients, the interconnections of the adaptors remain the same, irrespective of the of the design methodology used. The designs discussed in this application report are the Butterworth, Chebyshev, and Elliptic.[6] The Butterworth has the flattest response and is monotonic, both in the pass-band and stop-band. Among the two types of Chebyshev filters, the all-pole Type 1 has been used. This type exhibits ripple-like behavior in the pass-band and monotonic behavior in the stop-band. The Chebyshev filter usually gives a lower order in comparison to the Butterworth filter for the same set of filter specifications. Elliptic filters exhibit ripple characteristics in both the pass-band and the stop-band. Elliptic filters are hard to design but give the least order when compared to Butterworth and Chebyshev filters.

2.3 Different Forms of LWDF

The following sections describe a few different forms of the LWDF. These forms differ from a conventional LWDF only in the way the adaptors are interconnected.

2.3.1 Cascade of LWDFs

The implementation of the band-pass or the band-reject filter can be done by cascading the basic low-pass and high-pass filters. Although this can be accomplished easily by feeding the final output of the first filter into the input of the second filter, a small efficiency is achieved using the signal flow depicted in Figure 6. Again, complementary outputs are easily obtained with this approach. The signal flow diagram of Figure 6 shows the cascade of two 7th-order LWDFs. This is used to obtain a band-pass and a band-reject filter simultaneously. In this design, the first filter is an HPF, and the second is an LPF with the band-pass response at OUTPUT and the band-reject response at $\overline{\text{OUTPUT}}$.

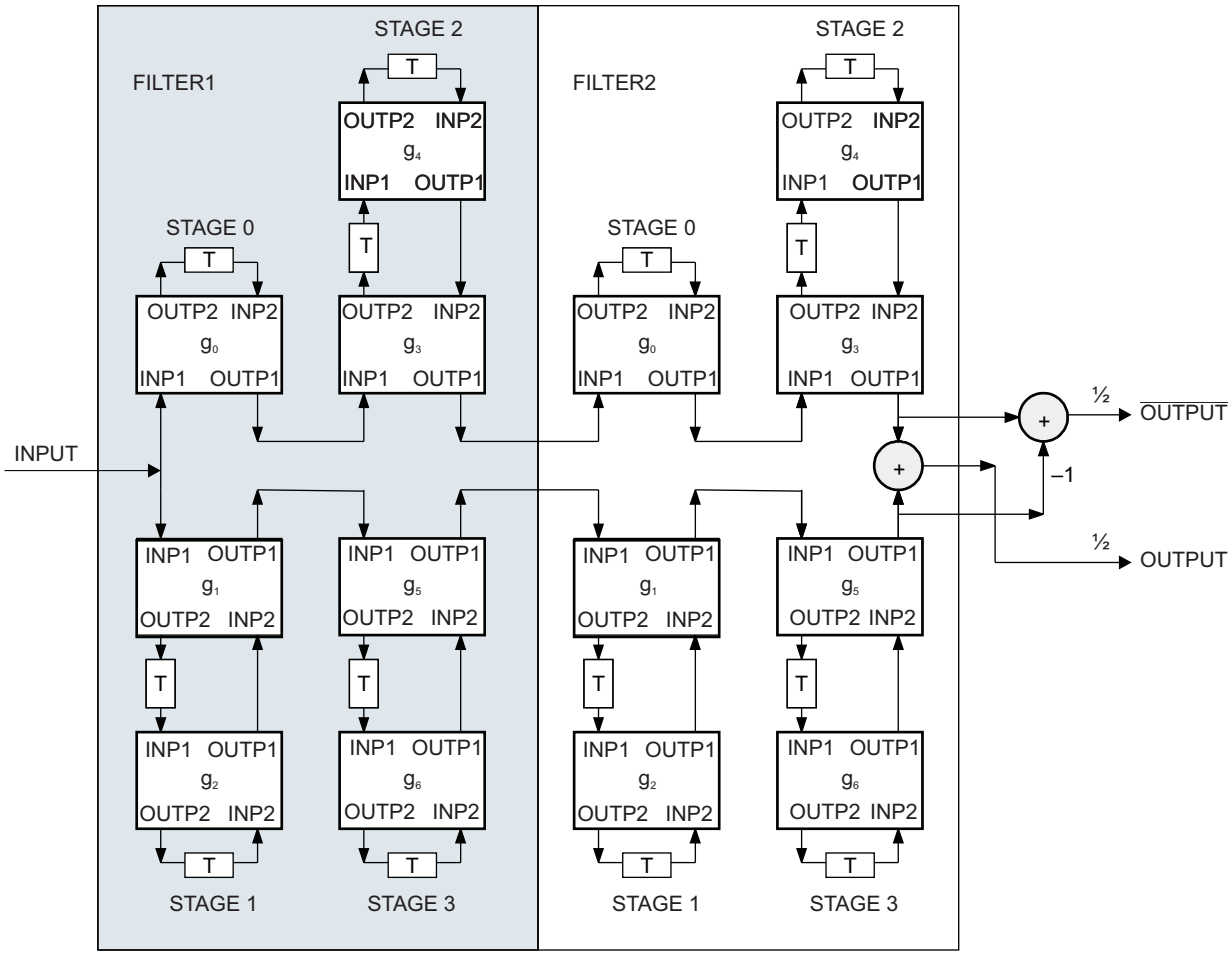
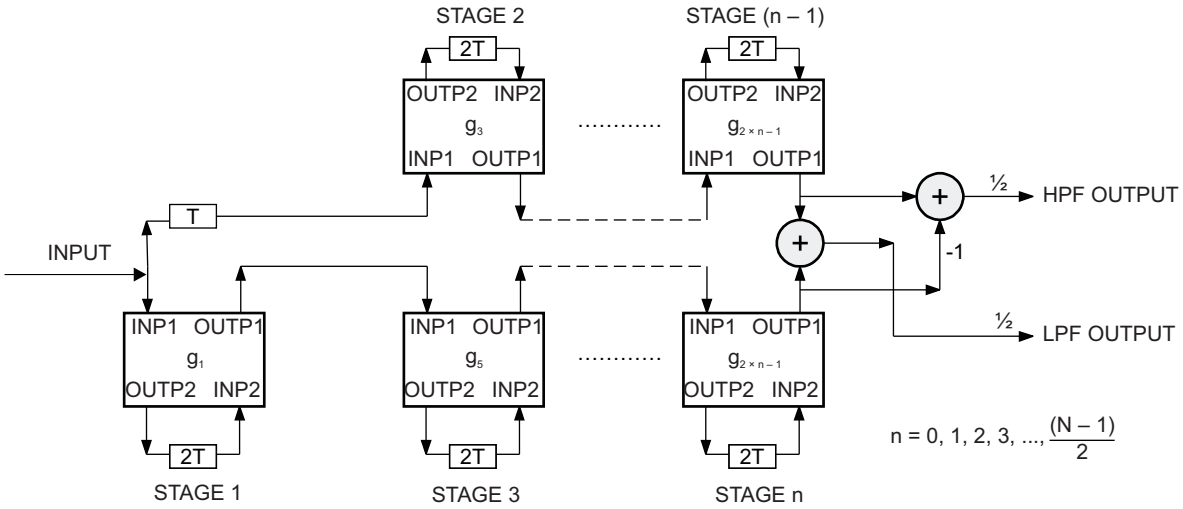


Figure 6. Cascade of Lattice Wave Digital Filters

2.3.2 Bireciprocal LWDF

The bireciprocal LWDF is a special form of LWDF that reduces the number of adaptors under certain conditions and can also be used as base for decimation and interpolation filters.[1][2] In a bireciprocal LWDF, all the even gamma coefficients, including γ_0, are zero, leaving only the odd-numbered adaptors to be implemented. This reduction in the number of adaptors reduces the total execution cycles and code size. However, this form of LWDF automatically results in a fixed cutoff frequency of one-fourth the sampling frequency. A generalized bireciprocal filter of order N is shown in Figure 7. The bireciprocal filter structure can only be used with the Butterworth and Elliptic filters; Chebyshev filters cannot be realized using this form.



NOTE: 2T represents a two-sample delay

Figure 7. Bireciprocal Lattice Wave Digital Filters

2.4 LWDF Tool

An LWDF, regardless of its response or design type, is completely characterized by its gamma coefficients and delay elements. These coefficients are explicitly found from the filter specifications. The executable file wdf_coeff.exe, provided by TI in conjunction with this application note, is a tool designed to obtain these coefficients for LWDF implementations on MSP430 MCUs. This interactive tool, when executed in the command window, prompts the user to enter the filter specifications. The tool then outputs the order, gamma coefficients, structure type for each adaptor, and the alpha coefficients in binary and CSD format. This information is written to a file named coeff.dat, which can be used to aid the design of the LWDF on the MSP430 CPU.

2.5 Implementation of the LWDF

This section provides the step-by-step procedure and a pseudo-code to implement a LWDF starting from the filter specifications.

Step 1: Obtaining the Coefficients for the LWDF

The filter specifications specify the response type (low-pass, high-pass, or band-pass) sampling frequency, pass-band edge frequency, stop-band edge frequency, pass-band ripple, stop-band ripple, the type of design (Butterworth, Chebyshev, or Elliptic), the filter form (normal or bireciprocal), and the design margin for the filter. The tool gives the gamma coefficients (γ_x) and the alpha coefficients (α_x) with the structure type for each adaptor. The alpha coefficients are also given in binary and CSD format to be used directly in the implementation of the LWDF.

Step 2 Implementation of the Filter on an MSP430 MCU

- **Initializations**

  For any order N, there are N adaptors, each with two inputs (INP1 and INP2) and two outputs (OUTP1 and OUTP2). Both inputs must be initialized or known before implementing the design equations in each adaptor. For order N, N delay elements are required. The initialization of the delay elements to zero is advantageous in order to reach a steady state earlier but is not absolutely necessary.

- **Implementation of Horner's Algorithm With CSD Representation**

  The design equations vary with the adaptor type according to Figure 2 through Figure 5, and the multiply in each adaptor should be efficiently done using the Horner's algorithm. Unique code is written for the multiply by the alpha coefficient in each adaptor, using shift and add operations only. CSD format should be used to reduce the execution time for this multiplication.

- **Use and Update of Delay Elements**

  The N delay elements (delay0, delay1…delay(N–1)) are updated during the filtering operation of each incoming sample. Each delay element value is updated with the OUTP2 value in the respective adaptor. They are subsequently used as inputs to some adaptors at the arrival of the next sample.

- **Filter Operation – Top Section**

  The top section of Figure 1 is always implemented first, starting with Adaptor 0, Adaptor 4, Adaptor 3, Adaptor 8, Adaptor 7, and so on.

  For Adaptor 0, the INP1 is equal to the incoming sample, and INP2 is equal to the delayed version of its own OUTP2 (stored at delay0 at the previous sample clock). The design equations give two outputs, OUTP1 and OUTP2. OUTP2 is stored in delay0, and OUTP1 becomes the INP1 for Adaptor 3.

  Adaptor 4 is implemented next with its inputs INP1 equal to the delayed version of OUTP2 (stored in delay3 at the previous sample clock) of Adaptor 3 and INP2 equal to the delayed version of its own OUTP2 (stored in delay4 at the previous sample clock). Of the two outputs obtained, OUTP2 is stored back in delay4, and OUTP1 is now INP2 for Adaptor 3. With both its inputs defined, Adaptor 3 is implemented next to give its two outputs. Of these, OUTP1 becomes INP1 for Adaptor 7, and OUTP2 is stored back in delay3. Repeating this procedure, all the adaptors in the top section are processed to finally give one output sample for the top section (see Figure 1).

- **Filter Operation – Bottom Section**

  The bottom section implementation is started with Adaptor 2, Adaptor 1, Adaptor 6, Adaptor 5, and so on. The design equations for Adaptor 4 and Adaptor 2 are implemented similar to the steps explained for the top section.

  For Adaptor 1, the INP1 is equal to the incoming sample, and INP2 is equal to the OUTP1 from Adaptor 2. Adaptor 6 and Adaptor 5 are implemented similarly. Repeating this procedure, all of the adaptors in the bottom section are processed to finally give one output sample for the bottom section (see Figure 1).

- **Output Sample**

  The final output sample is the sum (for LPF) or the difference (for HPF) of the top section and the bottom section output samples scaled by one-half. The LPF response is available at the output, and the HPF response is available at the complementary output (see Figure 1). If a BPF or BRF is desired, the cascaded LWDF is chosen with similar steps for each HPF followed by LPF.

2.5.1 Pseudo-Code

The filter in this example is a 9th-order Chebyshev low-pass filter. The type of each adaptor and the alpha coefficients in CSD format are available from the executable.

```
void main(void)
{
// Initializations
delay0=0, delay1=0, delay2=0, delay3=0, delay4=0, delay5=0, delay6=0, delay7=0, delay8=0

// Wave Digital Filter starts here
//Top section
INP1=Input sample, INP2=delay0
filter (alpha0, adaptor_type [0], INP1, INP2);
```

```
delay0=OUTP2
INP1=delay3, INP2=delay4
filter (alpha4,adaptor_type [4], INP1, INP2);
delay4=OUTP2
INP1=OUTP1 of Adaptor 0, INP2=OUTP1 of Adaptor 4
filter (alpha3, adaptor_type[3], INP1, INP2);
delay3=OUTP2
INP1=delay7, INP2=delay8
filter (alpha8, adaptor_type[8], INP1, INP2);
Delay8=OUTP2
INP1=OUTP1 of Adaptor 3, INP2=OUTP1 of Adaptor 8
filter (alpha7, adaptor_type[7], INP1, INP2);
Delay7=OUTP2
Output1=OUTP1

// The filter() implements the design equations for each adaptor based on its type.
// Bottom section
INP1=delay1, INP2=delay2
filter (alpha2, adaptor_type[2], INP1, INP2);
delay2=OUTP2
INP1= Input sample, INP2=OUTP1 of Adaptor 2
filter (alpha1, adaptor_type[1], INP1, INP2);
delay1=OUTP2
INP1=delay5, INP2=delay6
filter (alpha6, adaptor_type[6], INP1, INP2);
Delay6=OUTP2
INP1=OUTP1 of Adaptor 1, INP2=OUTP1 of Adaptor 6
filter (alpha5, adaptor_type[5], INP1, INP2);
Delay5=OUTP2
Output2=OUTP1

// The filter() implements the design equations for each adaptor based on its type.
// Final output
OutputLP = (Output1+Output2)/2 (For low-pass output)
OutputHP = (Output1-Output2)/2 (For high-pass output)
}
```

To obtain correct results, the code implementation of the adaptors should follow a particular sequence.

1. The top section is completely independent of the bottom section.
2. For the top section the sequence must be {0, 4, 3, 8, 7, 12, 11, ...}, and for the bottom section it must be {2, 1, 6, 5, 10, 9, ...}. If this is not maintained, incorrect results leading to instability are encountered.

3 Real-Time Filter Examples on MSP430 MCUs

This appendix contains numerical examples in the design of LWDF low-pass, high-pass, band-pass, and band-reject filters. The filter response is shown graphically as a performance analysis of the filter. The code to implement this LWDF on an MSP430 MCU is also is provided in the zip file. Section 5 has details on the filter testing methodology used to obtain these results.

3.1 Example 1: Low-Pass Filter

Filter specifications:

Sampling frequency	= 16000 Hz
Pass-band edge frequency	= 3400 Hz
Stop-band edge frequency	= 4500 Hz
Pass-band ripple	= 0.5 dB
Stop-band attenuation	= 50 dB
Filter type	= Chebyshev
Design margin(1)	= 0.5

Filter parameters:

Order = 9

(1) This factor indicates the amount of the design margin to be allocated to the pass band. It varies from 0 to 1.

Table 1 lists the gamma coefficients, adaptor type, alpha coefficients, and their representations in binary and CSD formats. Figure 8 shows the signal flow diagram for this WDF, and Figure 9 shows the interconnections with the adaptor structures. Figure 10 shows the frequency response for this filter. The plot shows the filter satisfying the specifications. The order, coefficients, and adaptor type change, even if only one of the specifications is modified. If the transition band is made narrower, with only the stop-band edge frequency changed to 3600 Hz, the order increases to 19. Figure 11 shows the frequency response of this filter.

Table 1. Low-Pass Filter Parameters

Stage	Gamma	Adaptor Type	Alpha	Alpha (Binary)	Alpha (CSD)(1)
0	0.667713527	1	0.33228647	0.0101010100010000	0.0101010100010000
1	–0.49630558	3	0.49630558	0.0111111100001101	0.1000000T00010T01
2	0.797917736	1	0.202082263	0.0011001110111011	0.010T01000T000T0T
3	–0.618835168	4	0.381164832	0.0110000110010100	0.10T00010T0010100
4	0.542641521	1	0.457358479	0.0111010100010101	0.100T010100010101
5	–0.766286584	4	0.233713416	0.0011101111010100	0.01000T000T010100
6	0.328193215	2	0.328193215	0.0101010000000100	0.0101010000000100
7	–0.919144204	4	0.080855796	0.0001010010110010	0.000101010T0T0010
8	0.217705053	2	0.217705053	0.0011011110111011	0.0100T0000T000T0T

(1) The T represents –1 in the ternary representation used in the CSD format.

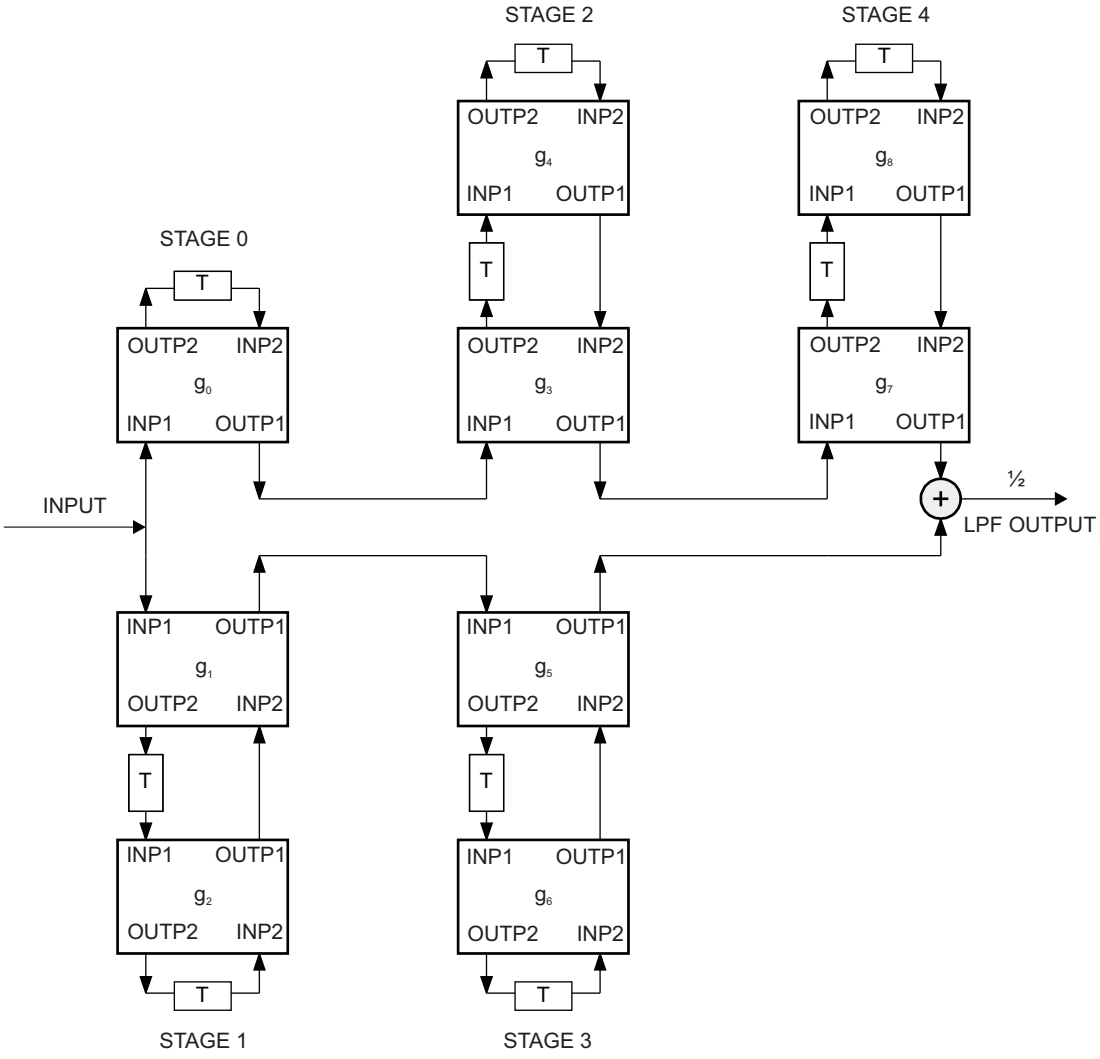


Figure 8. Signal Flow Diagram of 9th-Order Chebyshev Low-Pass LWDF

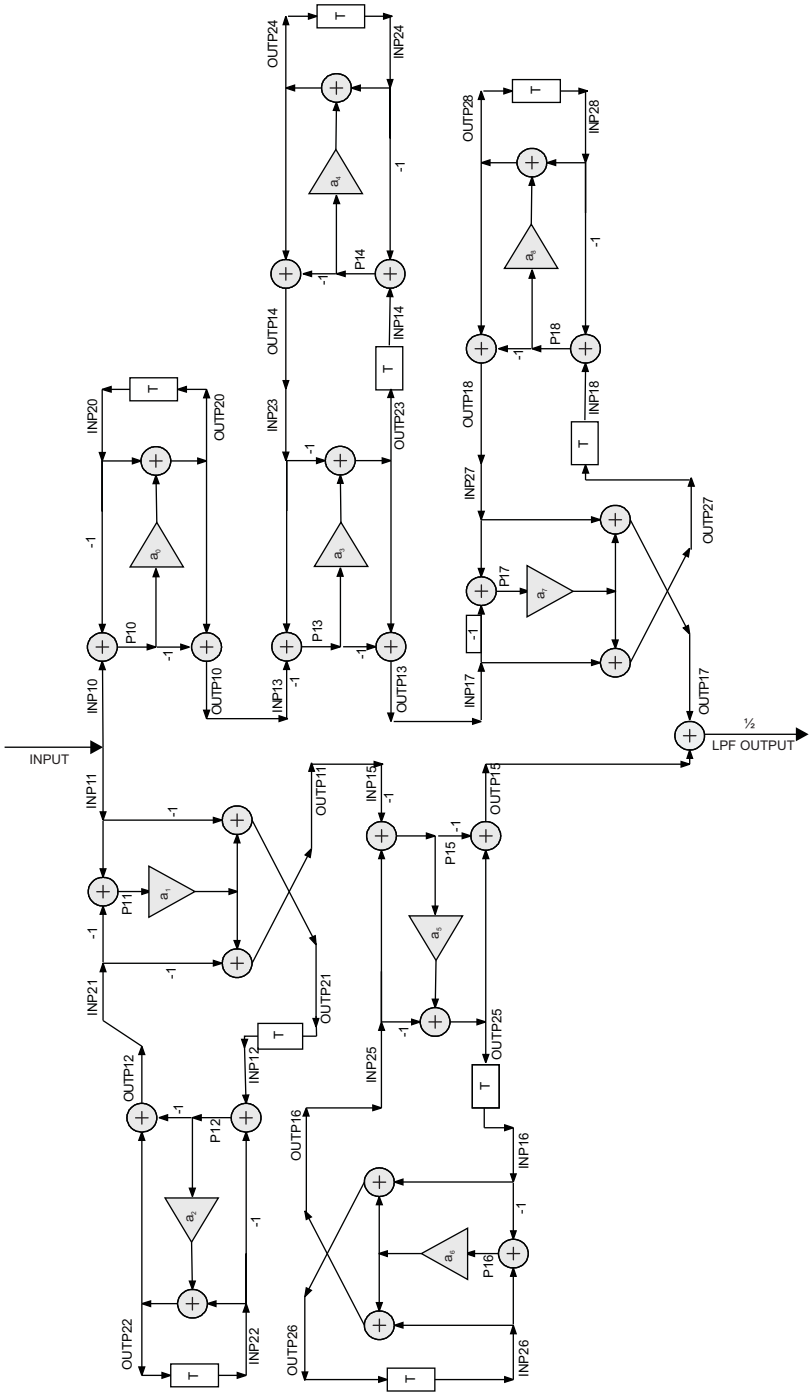


Figure 9. Interconnections for 9th-Order Chebyshev Low-Pass LWDF

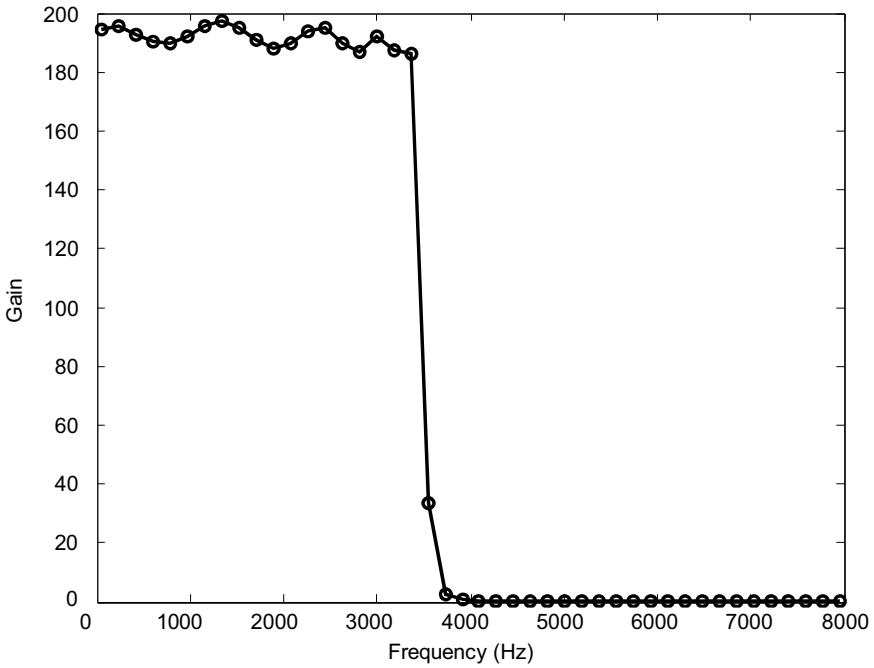


Figure 10. Frequency Response of 9th-Order Chebyshev Low-Pass LWDF

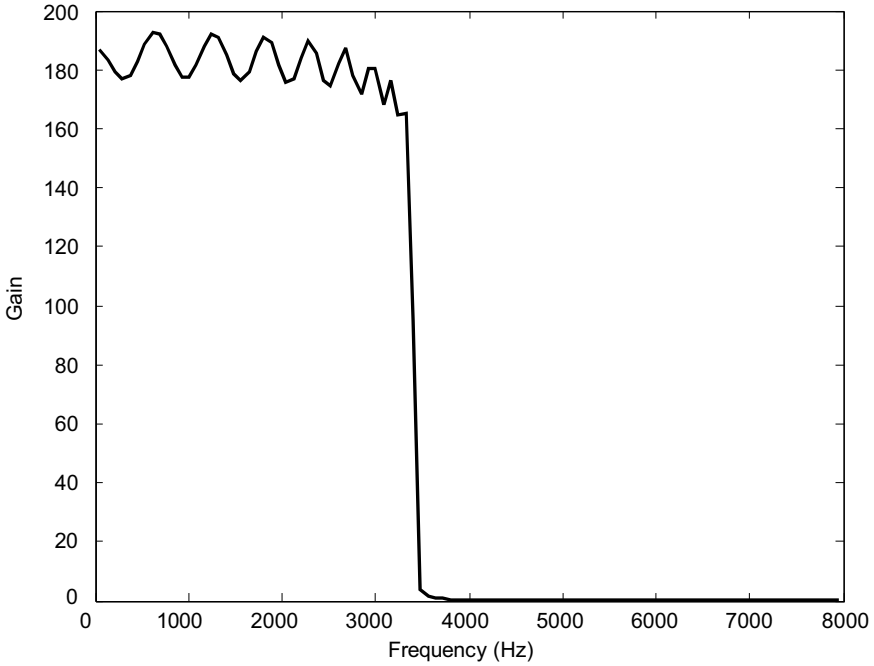


Figure 11. Frequency Response of 19th-Order Chebyshev Low-Pass LWDF

The entire LWDF can be implemented with single-cycle shift and add operations. For the same specifications, a complementary HPF can be obtained with no additional overhead in terms of design and just one additional instruction cycle. Figure 12 shows the response at the complementary output for the same filter.

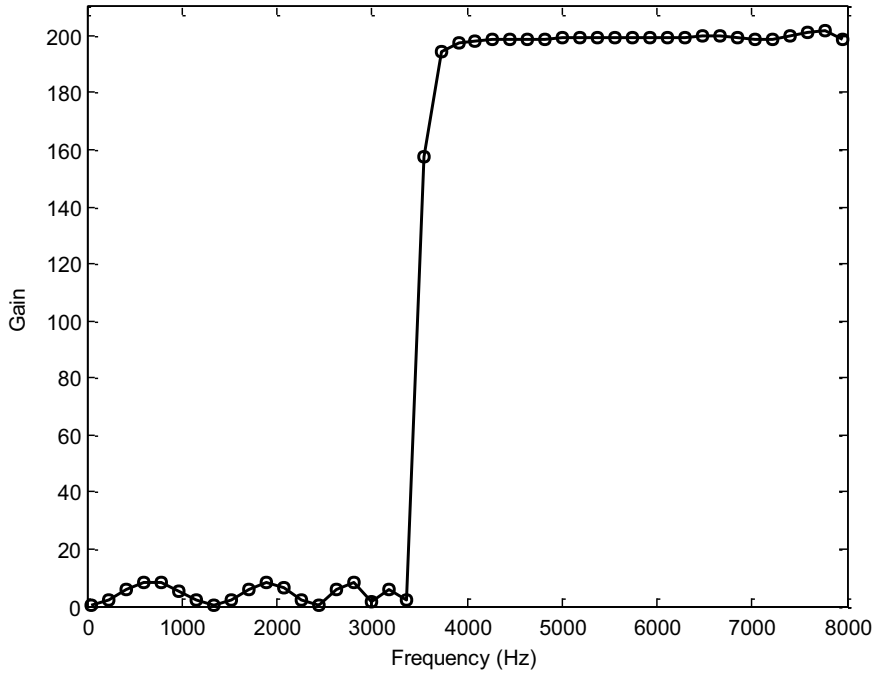


Figure 12. Response at Complementary Output of 9th-Order Chebyshev Low-Pass LWDF

The details of the implemented 9th-order filter on the MSP430FG439 MCU, in the absence of a hardware multiplier, are shown below.

CPU frequency	= 8 MHz
Cycles available between samples	= 500
Filter execution cycles	= 320
CPU utilization	= 64%

3.2 Example 2: High-Pass Filter

Filter specifications:

Sampling frequency	= 8000 Hz
Pass-band edge frequency	= 1100 Hz
Stop-band edge frequency	= 600 Hz
Pass-band ripple	= 1 dB
Stop-band attenuation	= 50 dB
Filter type	= Butterworth
Design margin	= 0.5

Filter parameters:

Order = 11

The Butterworth filter, also known as the maximally flat filter, gives a higher order when compared to the other types for the same set of specifications, but with an absence of ripples in the pass-band or the stop-band. Figure 13 shows the signal flow diagram for this 11th-order WDF, and Figure 14 shows the interconnections with the adaptor types. Figure 15 shows the frequency response for this filter. The plot shows the filter satisfying the specifications. Even in this case, a complementary LPF output can be obtained with no additional design overhead.

Table 2. High-Pass Filter Parameters

Stage	Gamma	Adaptor Type	Alpha	Alpha (Binary)	Alpha (CSD)[1]
0	0.581852484	1	0.418147516	0.0110101100001011	0.100T0T0T00010T0T
1	–0.356727908	3	0.356727908	0.0101101101010010	0.10T00T0T01010010
2	0.869375785	1	0.130624215	0.0010000101110000	0.00100010T00T0000
3	–0.412721971	3	0.412721971	0.0110100110101000	0.10T01010T0101000
4	0.869375785	1	0.130624215	0.0010000101110000	0.00100010T00T0000
5	–0.511030021	4	0.488969979	0.0111110100101101	0.10000T01010T0T01
6	0.869375785	1	0.130624215	0.0010000101110000	0.00100010T00T0000
7	–0.659368256	4	0.340631744	0.0101011100110011	0.10T0T00T010T010T
8	0.869375785	1	0.130624215	0.0010000101110000	0.00100010T00T0000
9	–0.868591148	4	0.131408852	0.0010000110100100	0.00100010T0100100
10	0.869375785	1	0.130624215	0.0010000101110000	0.00100010T00T0000

(1) The T represents –1 in the ternary representation used in the CSD format.

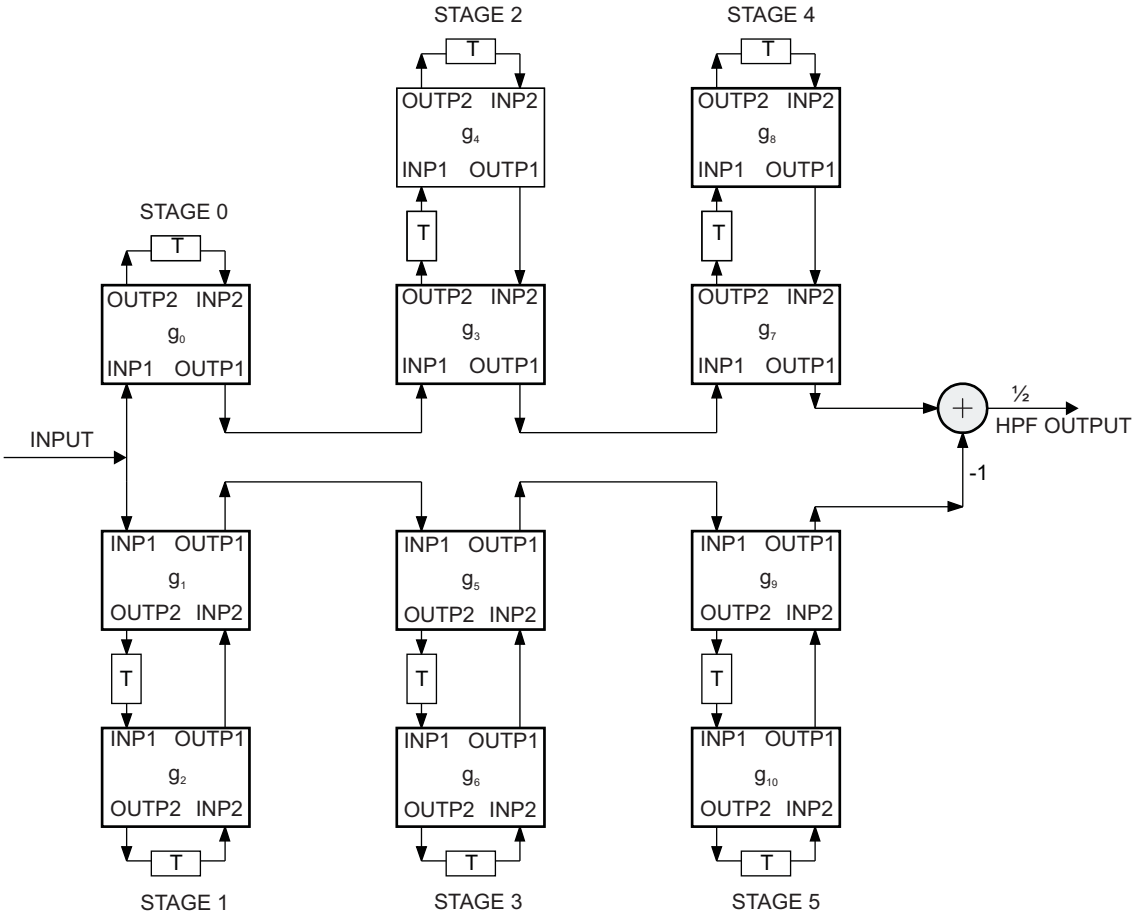


Figure 13. Signal Flow Diagram of 11th-Order Butterworth High-Pass LWDF

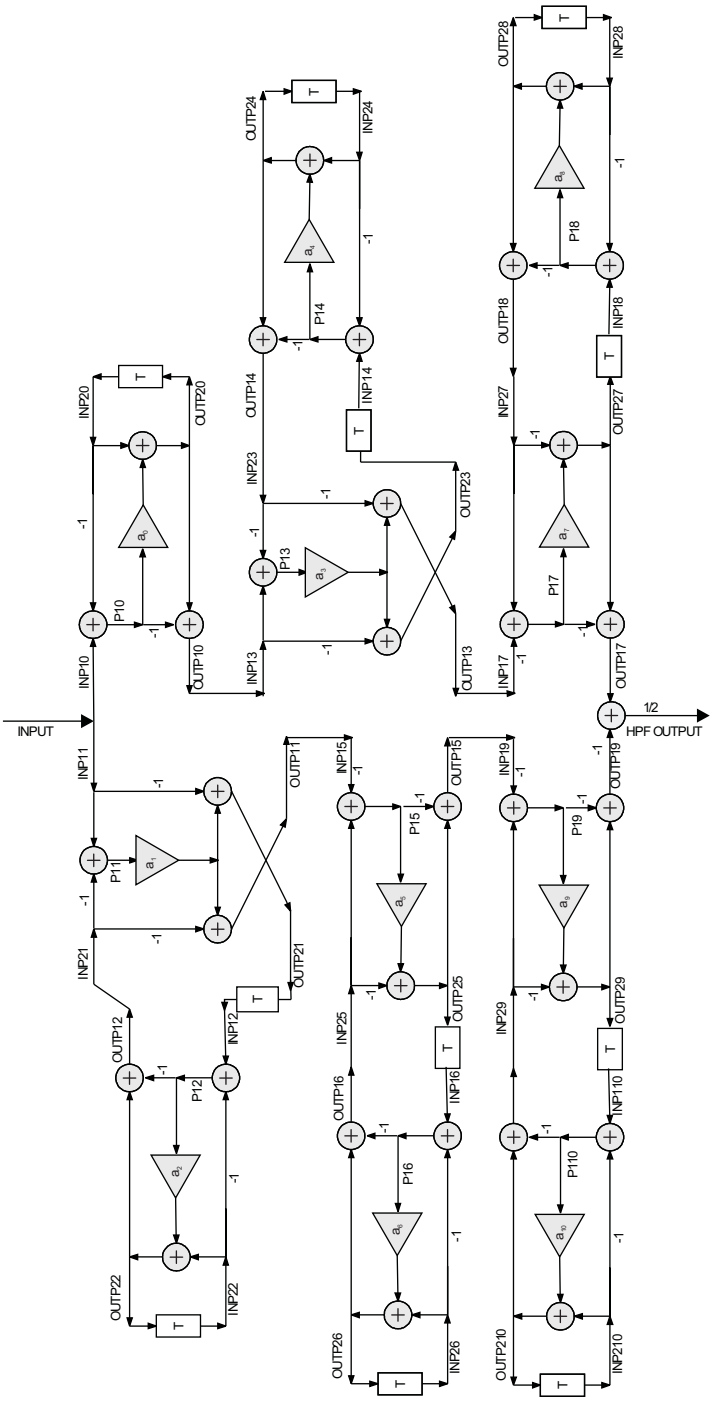


Figure 14. Interconnections for 11th-Order Butterworth High-Pass LWDF

Figure 15 shows the frequency response of this HPF. The Butterworth LWDF performs in accordance with the filter specifications.

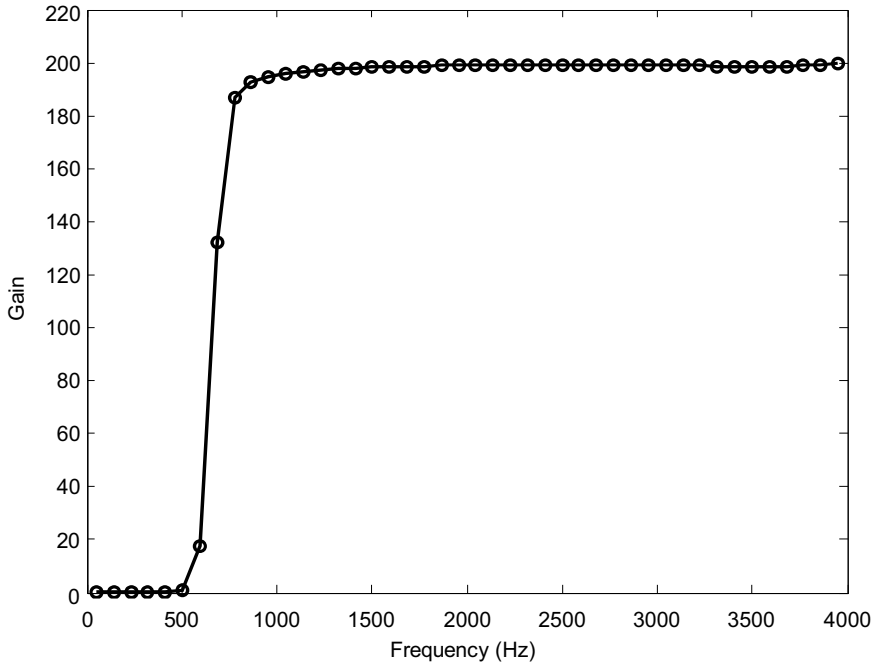


Figure 15. Frequency Response of 11th-Order Butterworth High-Pass LWDF

The details of the implemented filter on the MSP430FG439 CPU, in the absence of a hardware multiplier, are shown below:

CPU frequency	= 8 MHz
Cycles available between samples	= 1000
Filter execution cycles	= 378
CPU utilization	= 37.8%

The next example shows the design of a band-pass filter. The BPF is implemented as a cascade of an HPF and an LPF. Figure 6 shows the signal flow diagram for the cascade. The order of the band-pass filter is the sum of the orders of the HPF and LPF. To implement a realizable filter order for the same set of specifications, an elliptical filter is chosen for this design.

3.3 Example 3: Band-Pass Filter

Filter specifications:

Sampling frequency	= 8000 Hz
Lower stop-band edge frequency	= 700 Hz
Lower pass-band edge frequency	= 950 Hz
Lower pass-band ripple	= 0.5 dB
Lower stop-band attenuation	= 50 dB
Higher pass-band edge frequency	= 1500 Hz
Higher stop-band edge frequency	= 1850 Hz
Higher pass-band ripple	= 0.5 dB
Higher stop-band attenuation	= 50 dB
Filter type	= Elliptical
Design margin	= 0.5

Filter parameters:

Order = 14 (7 + 7)

This design involves cascading an HPF of lower-frequency design specifications with an LPF of higher-frequency design specifications. Table 3 lists the filter coefficients and adaptor type. Figure 16 shows the frequency response for the designed LWDF.

Table 3. Band-Pass Filter Parameters

Stage	Gamma	Adaptor Type	Alpha	Alpha (Binary)	Alpha (CSD)[1]
0	0.785463010271	1	0.214536989729	0.0011011011101011	0.0100T00T000T0T0T
1	–0.702979748323	4	0.297020251677	0.0100110000001001	0.01010T0000001001
2	0.932216495388	1	0.067783504612	0.0001000101011010	0.00010010T0T0T010
3	–0.851225771767	4	0.148774228233	0.0010011000010110	0.001010T00010T0T0
4	0.875354845789	1	0.124645154211	0.0001111111101000	0.0010000000T01000
5	–0.958244741485	4	0.041755258515	0.0000101010110000	0.00010T0T0T0T0000
6	0.847740514181	1	0.152259485819	0.0010011011111010	0.0010100T0000T010
7	0.55932087987	1	0.44067912013	0.0111000011010000	0.100T00010T010000
8	–0.483475352921	3	0.483475352921	0.0111101111000101	0.10000T000T000101
9	0.667972219066	1	0.332027780934	0.0101010011111111	0.0101010100000T0T
10	–0.755025929057	4	0.244974070943	0.0011111010110110	0.0100000T0T00T0T0
11	0.4551195777	2	0.4551195777	0.0111010010000010	0.100T010010000010
12	–0.93319529576	4	0.06680470424	0.0001000100011010	0.000100010010T010
13	0.369550526968	2	0.369550526968	0.0101111010011010	0.10T000T01010T010

(1) The T represents –1 in the ternary representation used in the CSD format.

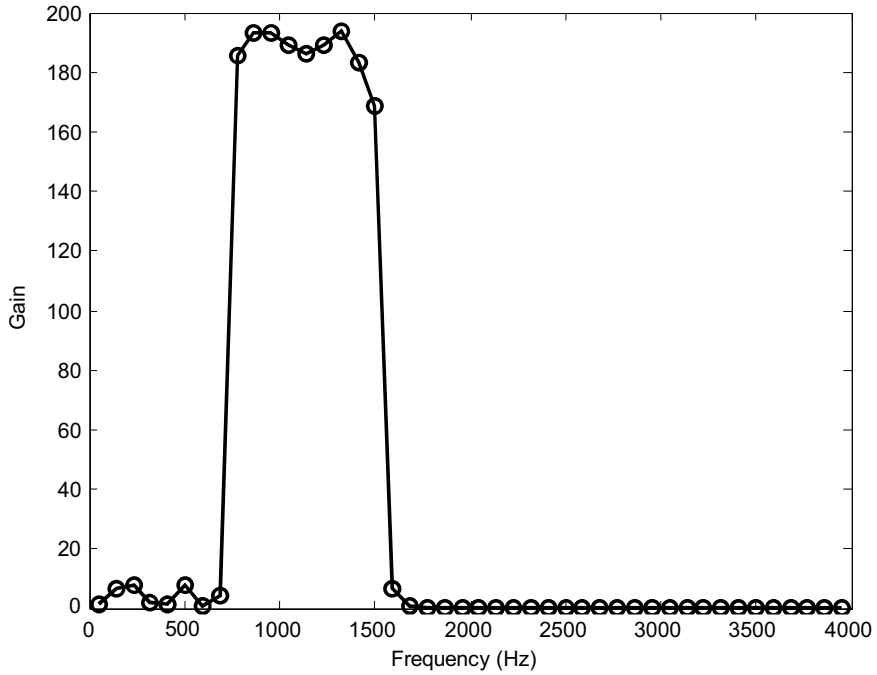


Figure 16. Frequency Response of 14th-Order Elliptical Band-Pass LWDF

The details of the implemented filter on the MSP430FG439 CPU, in the absence of a hardware multiplier, are:

CPU frequency	= 8 MHz
Cycles available between samples	= 1000
Filter execution cycles	= 501
CPU Utilization	= 50.1%

3.4 Summary

Looking at the execution cycles from Example 1 to Example 3, it can be concluded that the number of instruction cycles required for every increment in filter order is in the range of 30 to 35 cycles, with 16-bit word length for the coefficients. In terms of the design criterion, this range allows the estimation of the total number of instruction cycles required for the entire filter. The filter specifications can then be eventually relaxed to make it realizable using the MSP430. Another possibility is to shorten the width of coefficients slightly, so that the frequency-response specification is not violated.

4 Code Details

This appendix describes the code for MSP430 MCUs that provided for the WDF examples discussed in this report. The name of each piece of code and its functionality are:

- wdf_ex1.c

  C code that calls the MSP430 assembly wave digital filter code to perform the low-pass filtering discussed in Example 1
- wdf_ex1_asm.s43

  Assembly code written to perform the 9th-order Chebyshev low-pass filter of example 1
- sine_data_ex1.dat

  Data file that has the input samples for the frequency sweep for Example 1
- wdf_ex2.c

  C code that calls the MSP430 assembly wave digital filter code to perform the high-pass filtering discussed in Example 2
- wdf_ex2_asm.s43

  Assembly code written to perform the 11th-order Butterworth high-pass filter of Example 2
- sine_data_ex2.dat

  Data file that has the input samples for the frequency sweep for Example 2
- wdf_ex3.c

  C code that calls the MSP430 assembly wave digital filter code to perform the band-pass filtering discussed in Example 3
- wdf_ex3_asm.s43

  Assembly code written to perform the 18th-order elliptical band-pass filter of Example 3. This is done by the cascade of a 9th-order high-pass filter with a 9th-order low-pass filter.
- sine_data_ex3.dat

  Data file that has the input samples for the frequency sweep for Example 3

5 Filter Testing Methodology

This appendix describes the testing methodologies used for the frequency response of the examples in Section 4.

5.1 Input Samples

The input samples were generated offline using a simple C-program and the sine function. The range of frequencies is from 50 Hz to 50 Hz less than half the sampling frequency, equally divided into 44 steps, with 400 samples for each frequency. This totals to 17600 fixed-point values stored in the file "sine_data_ex(1,2, or 3).dat" provided with this application report. This file is included at the beginning of the wrapper C file for each example. This data pattern is different for Example 1, which has a different sampling frequency than Example 2 and Example 3. These assumptions have a direct impact on the appearance of the frequency response and the gain values. The gain value is just a number obtained during the verification of the filter in time domain. The LWDF, in general, produces a gain of 1 for all frequencies in the pass-band. Normalizing these gain values yields the true LWDF characteristics. LWDF have been shown to perform as per their specifications across all frequencies, exhibiting excellent performance overall.[1][2]

5.2 Assembly Calls

A function call of the assembly function that implements the LWDF is done for each of the 17600 samples to return one output sample for each call.

5.3 Plots

To accommodate the verification of the LWDF response across various frequencies on the MSP430, a simple square-and-accumulate operation of 400 consecutive output samples is done to calculate the gain at each frequency. These values are printed on the Terminal I/O of IAR Embedded Workbench when the program is executed. The frequency response plot is obtained when this gain is plotted against its corresponding frequency. The calculations described are performed only for the verification of the filter characteristics and are not part of the digital filtering operation or algorithm. This method does not conform to any industry standard or method to obtain a filter's frequency response. The reader is also advised to use this only as an approximation to a frequency response plot, rather than as an accurate measurement of performance. The performance of these filters under stringent resolutions can be verified using some of the resources cited in the reference section.

5.4 Use of the Assembly Function

The LWDF is unique for each example and is completely characterized by the assembly function. It can be included directly in a program with proper function calls and returns. Because the sampling frequency and the cutoff frequency are designed in relation to each other, the existing filter can be scaled; for example, divide the cutoff frequency by four and, consequently, also divide the sample frequency by four.

6 References

1. Fettweis, Alfred, "Wave Digital Filters: Theory and Practice", *Proceedings of the IEEE*, Vol. 74, pp. 276-327, February 1986.
2. Kaiser, Ulrich, "Wave Digital Filtering for TI's Sensor Signal Processor MSP430", *Texas Instruments Technical Journal*, Vol. 11, No. 6, 1994 S.65-83.
3. Venkat, Kripa, Efficient Multiplication and Division Using MSP430
4. Gazsi, Lajos, "Explicit Formulas for Lattice Wave Digital Filters", *IEEE Transactions on Circuits and Systems*, Vol. 32, No. 1, pp. 68-88, January 1985.
5. MSP430 MCU family user guides
6. Mitra, Sanjit K., *Digital Signal Processing: A Computer Based Approach*, Second Edition, McGraw Hill, 2001.
7. C. Marven, G. Evers, *A Simple Approach to Digital Signal Processing*, Texas Instruments, 1994, ISBN 0-904-047-00-8.

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from September 13, 2006 to May 16, 2018	Page
• Editorial updates throughout document	1

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265