

Wave Digital Filtering for TI's Sensor Signal Processor MSP430

Abstract: Wave Digital Filters (WDF)^{1,2} have notable advantages: excellent stability properties even under nonlinear operating conditions resulting from overflow and roundoff effects, low coefficient word length requirements, inherently good dynamic range, stability under looped conditions, etc.

Compared with the often used averaging of measured sensor data, digital filtering has advantages such as low pass filtering with sharp cut-off region, notch filtering of noise, and so on.

For the design of WDF algorithms, specialized CAD programs have been designed to speed up the top-down design from filter specification to the machine program for the processor:^{3,4,5,6}

- LWDF_DESIGN allows the design of lattice-WDFs
- LWDF_COMP transforms a lattice-WDF structure into an assembler program for the MSP430
- DSP430 allows fast transient simulations of the filter algorithms on a model of the MSP430, analysis of frequency response, check of accuracy and proof of stability.

The programs enable MSP430 users to solve special measurement problems by means of robust digital filter algorithms.

Key words: digital filtering, wave digital filtering, WDF, sensor signal processor, MSP430, metering

Core competency: low-power metering systems

MEASUREMENT of physical quantities has its application in industrial and consumer products. Customers increasingly demand so-called "intelligent sensor" systems, systems that measure temperature, pressure, flow and other quantities with high accuracy. The products often are portable, so extremely low power consumption is a need, for example, a lifetime of 10 years with one battery. The new MSP430 sensor signal processor family is specially designed for such systems.

For measurement of analog quantities the MSP430 has an analog-to-digital converter (ADC) on chip. By taking several measured values (oversampling) and averaging them, the accuracy of the measuring process can be improved; this is done by software control. The improved accuracy is not free; the oversampling/averaging requires more time and more power.

The process of "continuous averaging" is demonstrated in *Figure 1* for eight samples. After each measurement the older samples are shifted through the registers T1 ... T8. The oldest value is lost, the new value enters T1 and the sum of the eight registers is calculated. The result is

$$Y = 0.125 \times (T1 + T2 + T3 + T4 + T5 + T6 + T7 + T8).$$

Figure 1 also represents the signal flow diagram of a finite-impulse-response (FIR) filter with a degree of seven and eight coefficients with the same value of 1.

The frequency response of this FIR filter is shown in *Figure 2*. Obviously, the attenuation is poor (only 12.8 dB). However, the filter is useful for the noise suppression of certain frequencies if the sample rate is chosen such that the attenuation notches appear at those noise frequencies (comb filter).

For applications needing more attenuation in the stopband (> 20 dB) eight multiplications with different coefficients (unequal 1.0) are required, but the multiplications consume additional CPU time.

The class of infinite-impulse-response (IIR) filters needs only few multiplications compared to FIR filters with regard to the same specification. Because of their recursive nature they may suffer from stability problems, especially when quantization of signals and coefficients is applied (finite word length effects).

Compared to classical IIR filters, the WDFs have excellent stability properties even under nonlinear operating conditions. The third section covers this in detail and explains how the multiplications can be

Ulrich Kaiser

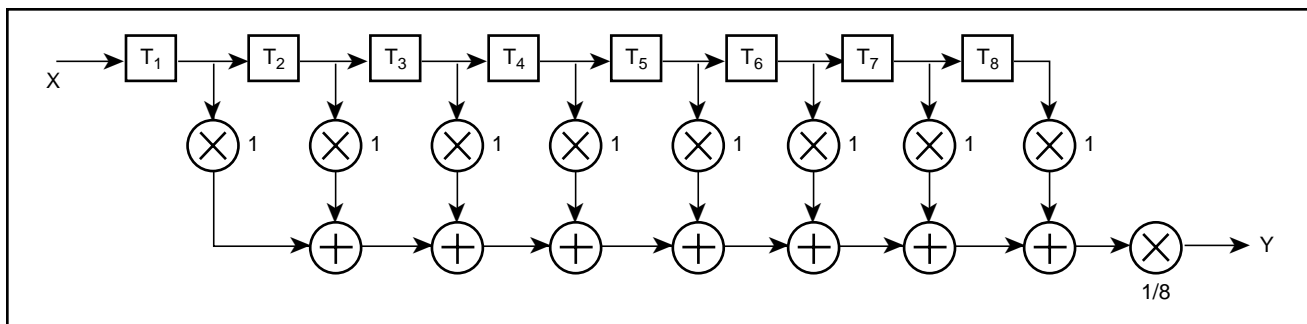


Figure 1. FIR Filter for Continuous Averaging.

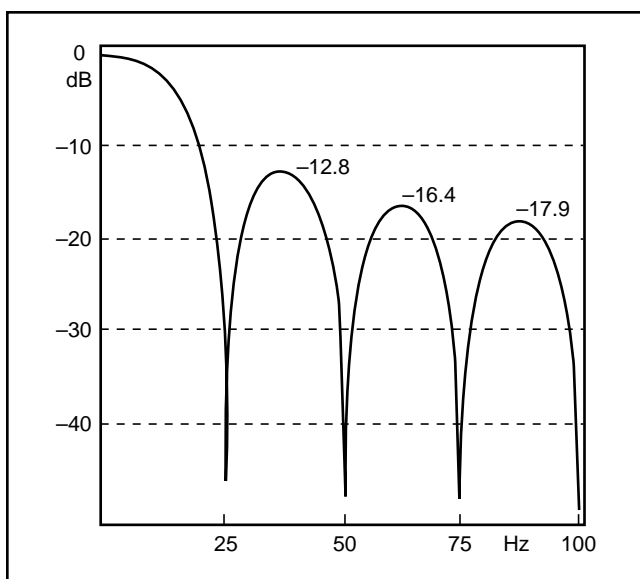


Figure 2. Frequency Response of the Filter of Figure 1.

performed most efficiently.

The fourth section presents the program system that allows the top-down design from filter specification to the running machine model of the MSP430. The different steps of the design process and the MSP430 model are explained.

In section five, use of the system is demonstrated with a lowpass filter of fifth order as an example. Design process, compilation, code generation and filter simulation are shown, the filter's function is tested and its stability properties are proven under finite wordlength conditions using the quantization process inherent in the MSP430 CPU model.

Architecture of the MSP430

The MSP430 family of controllers is built around a 16-bit CPU having a modular concept with special capabilities for analog processing (Figure 3).

The following list contains some features and capabilities of the MSP430 family. More information can be found in references 7, 8 and 9.

- Von-Neumann architecture
- Only three instruction formats, strong orthogonality
- One word per instruction (as far as possible)
- Memory-mapped I/O ports
- ADC (10 bits or more) with four inputs and current source
- Very low current consumption
- Full operation down to 2.5 V
- Modular design concept: modules are strictly memory-mapped.

The CPU contains a 16 bit ALU, 16 registers and instruction control logic. Registers R4, R5, ... R15 are general-purpose working registers. Four of the registers are reserved for special purposes:

- Program counter (PC)
- Stack pointer (SP)
- Status register (SR)
- Constant generator (CG2).

Some instructions are of special importance for implementation of digital filter algorithms:

- MOV load and store operations
- ADD and SUB arithmetic addition and subtraction
- RRA arithmetic right shift.

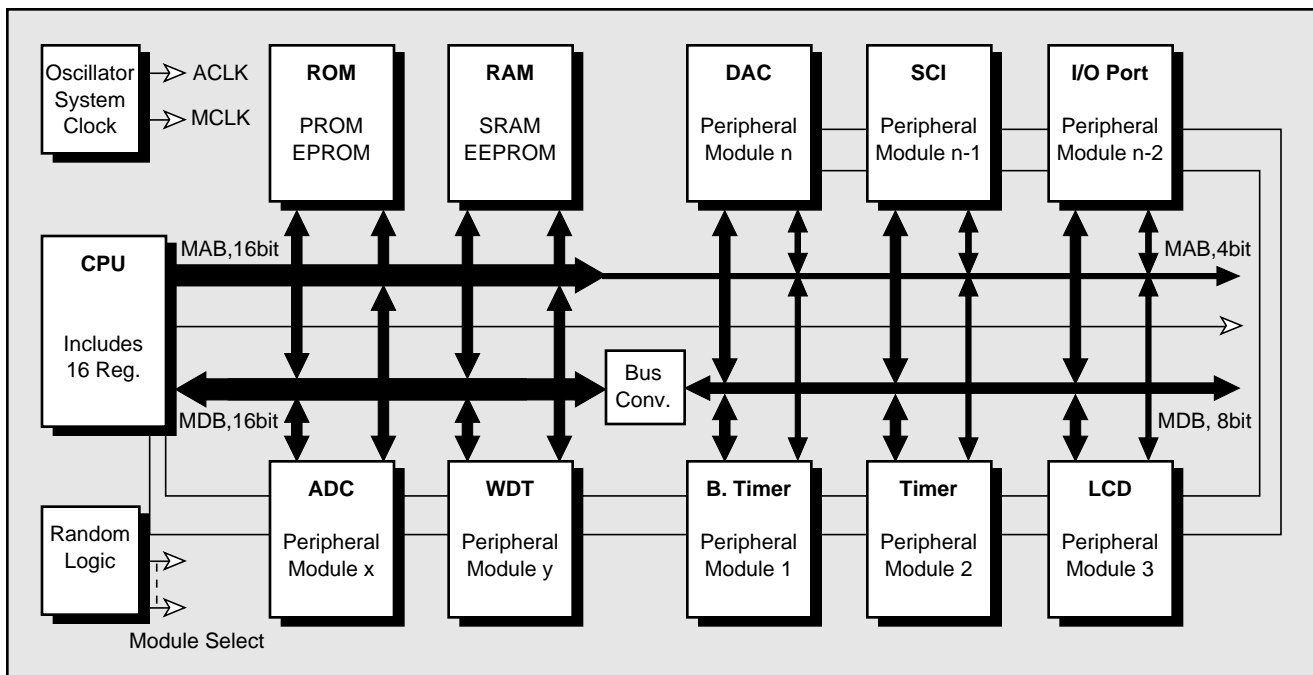


Figure 3. MSP430 System Configuration Principle.

Since there is no hardware multiplier on-chip, the multiplications are carried out by the Shift&Add multiplication algorithm. If Horner's Scheme is applied, the multiplication errors are minimized.² Such a Shift&Add multiplication is especially efficient if wave digital filters are chosen, because their signal flow diagrams show about three times more adders and subtractors than multipliers.

Wave Digital Filters

Wave digital filters are derived from time-continuous LC and microwave filters playing the role of the reference filter. The WDF concept, proposed in 1971 by Fettweis, provides digital filters having excellent properties:¹

- Stability under nonlinear operating conditions resulting from round off effects (small scale limit cycles)
- Response stability (overflow, large scale)¹⁰
- Passivity
- Stability under looped conditions
- Inherently good dynamic range
- Low coefficient wordlength requirements (low sensitivity)

- Shift&Add multiplication, no hardware multiplier
- Lattice WDFs allow easy sample rate change
- Top-down design from specification to operational filter.

A well-suited structure for top-down design is the lattice WDF^{3,4} (Figure 4). It consists of cascades of all-pass sections of 1st and 2nd order; i.e., two-port adaptors (arithmetic elements) and delay elements (state variables). The total filter order is always an odd number.

A large amount of hardware/software can be saved by specifying the 3dB-frequency at a quarter of the operating rate. Figure 5(a) shows such a bi-reciprocal WDF of 5th order containing two adaptors only — useful for band-split applications. Bi-reciprocal WDFs also allow sample rate decimation or interpolation: The input samples are fed alternately into the lattice branches, so that a decimation by two is performed (Figure 5(b)).^{2,5,11}

A two-port adaptor and two related delay elements are shown in Figure 6. Due to the WDFs low coefficient sensitivity, it is possible to choose the coefficients using canonical signed digit code (CSD, sum of powers-of-two), performing the multiplications with a

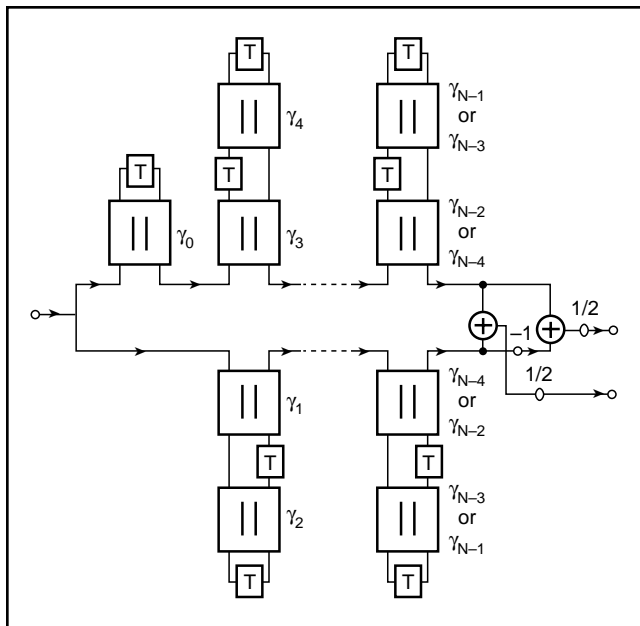


Figure 4. Lattice Wave Digital Filter of nth Order.

special Shift&Add algorithm (Horner's scheme). The preferable binary format is two's complement fixed point. The typical equation to calculate is

$$(x - y) \times \text{gamma} + z;$$

Figure 7 demonstrates; e.g., the calculation of

$$(T2 - T1) \times 0.3125 + T2$$

for node 29. The CSD representation of 0.3125 is 0.0101; i.e., $2^{-2} + 2^{-4}$ or $(2^{-2} + 1) \times 2^{-2}$. Hence, the calculation of node 29 and the related multiplication is performed as

$$N29 := [(T2 - T1) \times 2^{-2}] + (T2 - T1) \times 2^{-2} + T2;$$

The data representation chosen contains one sign bit, two guard bits and 13 rear bits (Figure 8) resulting in an $\text{LSB} = 122 \times 10^{-6}$ and a calculation region $R4 \{-4 \dots +4\text{-LSB}\}$. The state variables store data in the region $R1 \{-1 \dots +1\text{-LSB}\}$ (Figure 9). Converting $R1$ numbers to $R4$ is done by simple sign extension. The opposite conversion $R4 \rightarrow R1$ requires a non-linear operation; e.g., "saturation" (Figure 10).

The algorithm to be performed for the example in Figure 7 reads:

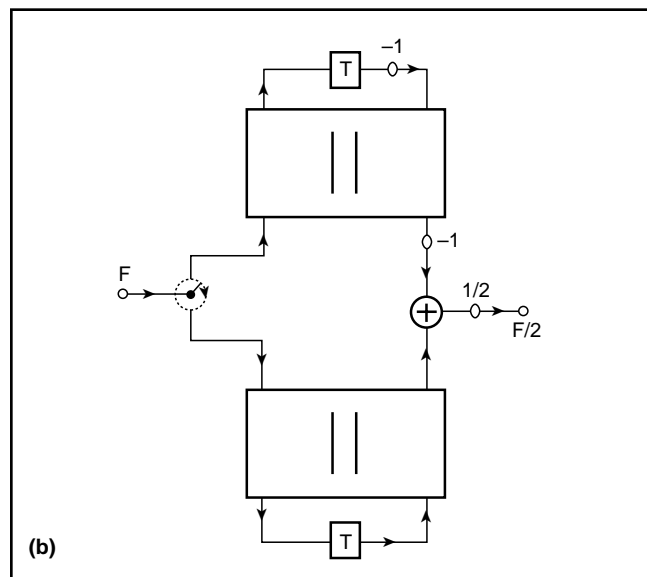
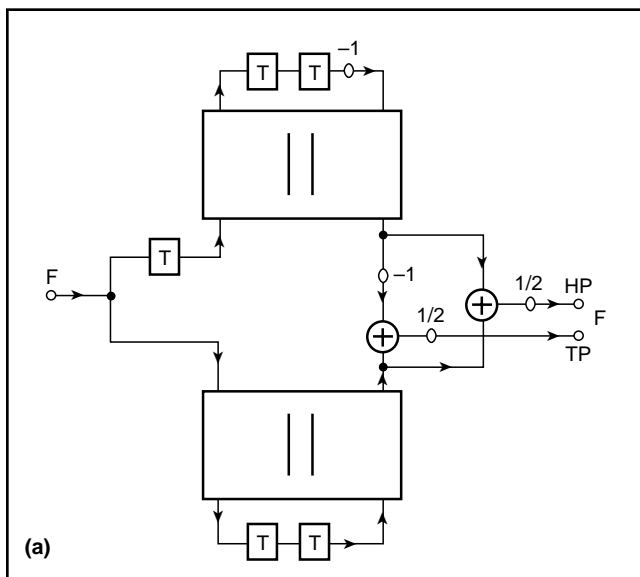


Figure 5. Bi-reciprocal 5th Order Lattice Wave Digital Filter (a) Bandsplit Filter (LP and HP) (b) Decimation Filter.

ter>.FSP, the routine LWDF_CALC calculates the necessary filter order (odd number) and the ideal coefficients. The result is stored in file <filter>.FIL and is used by subroutine LWDF_PLOT, which calculates the frequency response of the LWDF. With the specification limits, the frequency response graph can be viewed on a PC screen.

Furthermore, a file <filter>.SFD is generated which contains the signal flow diagram (SFD) of the LWDF structure in textual form. Figure 12 shows the elements of the structure description language WDFL, consisting of elements:^{4,5}

- Input
- Adaptor
- Delay

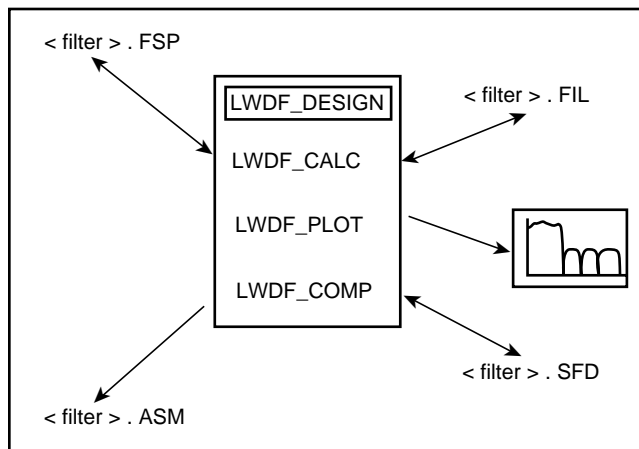


Figure 11. Development of Lattice Wave Digital Filters.

- Adder
- Shifter
- Output
- Some compiler directives.

Figure 13 shows how the structure of a fifth order LWDF is mapped onto the description by using WDFL elements:

```
.C ELP5 lowpass 5.order
input(1);
#
# upper branch
.B1
delay (2, 3, 1); T0
adaptor2 (1, 2, 3, 4, 0.2758713); gamma0
#
delay (7, 8, 1); T4
adaptor2 (6, 7, 8, 9, 0.1641284); gamma4
#
delay (5, 6, 1); T3
adaptor2 (4, 5, 9, 10, -0.7739261); gamma3
#
# lower branch
.B2
delay (13, 14, 1); T2
adaptor2 (12, 13, 14, 15, 0.3255008); gamma2
#
delay (11, 12, 1); T1
adaptor2 (1, 11, 15, 16, -0.3316880); gamma1
#
adder (10, 16, 17);
shifter (17, 18, -1);
output (18);
.END
```

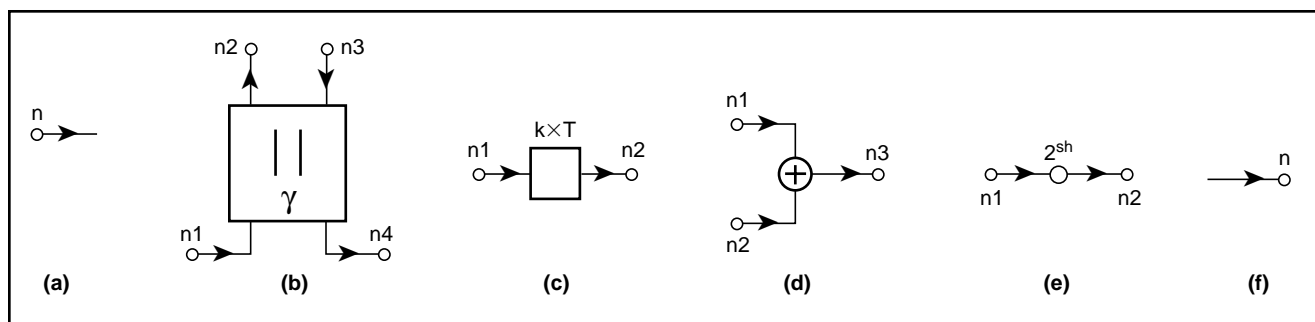


Figure 12. Structure Description Language Elements.

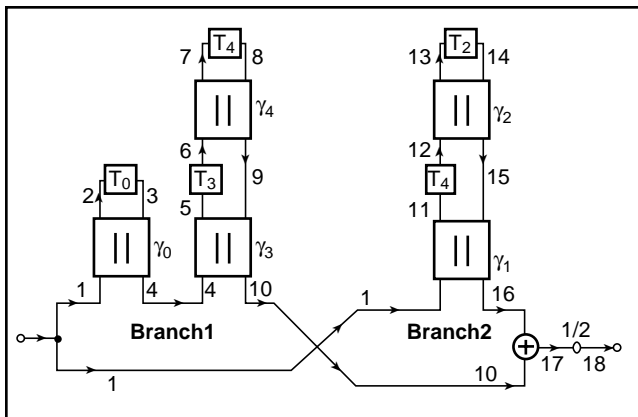


Figure 13. Structure of LP5.

Such a textual description has the advantage that it can be read by computer very easily. This is done by the compiler LWDF_COMP.^{4,5} The compiler reads the listing in file <filter>.SFD and generates an assembler language file <filter>.ASM for the MSP430.⁶

The compilation performs the following steps:

- Syntax check on the input file <filter>.SFD
- Allocation of registers for special variables
- Allocation of RAM space for the filter state variables (Tx)
- Allocation of one of four adaptor types (Figure 14)³ with regard to the region of gamma; mapping gamma (-1..+1-LSB) to alpha (0..0.5)
- Conversion of alpha from binary to CSD representation
- Expansion of delay elements
- Expansion of adaptors with regard to related delay elements
- Expansion of adder, shift and output elements
- Writing of file <filter>.ASM.

The expansion is a proper generation of MOV, ADD, SUB and RRA instructions so that the adaptor equations are formed and concurrently the CSD coefficients are transformed into Shift&Add instructions.

The file <filter>.ASM contains as usual a set of equations and groups of instructions. Figure 15 presents parts of the example file ELP5C.ASM.

After the filter description has been converted into an MSP430 assembler language file, it can be tested either by the MSP430-simulator or by a specially designed simulator/analyser DSP430, which is built around a model of the MSP430.⁶ The model is imple-

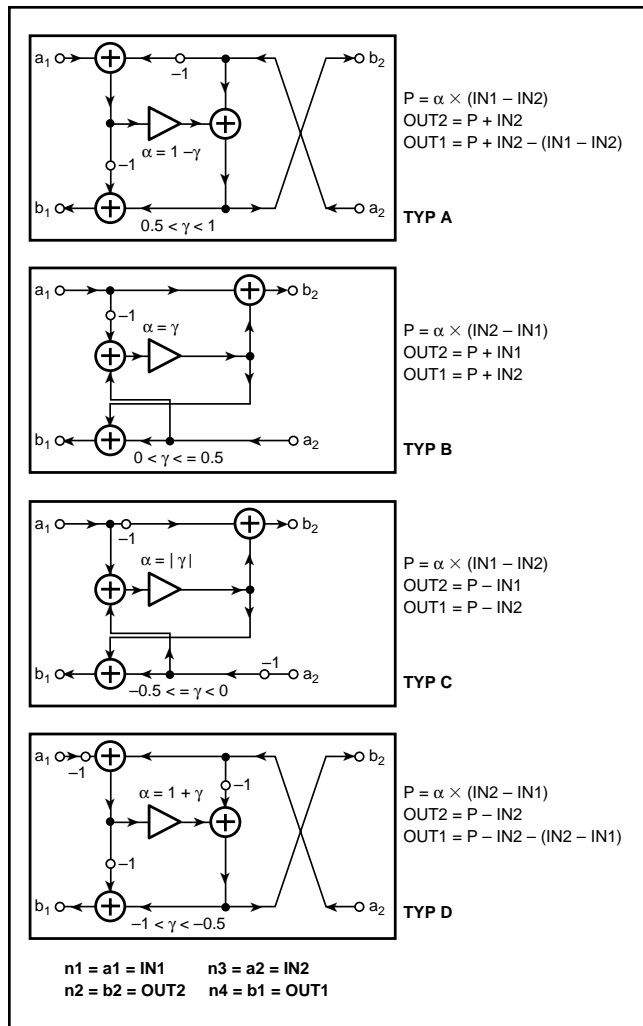


Figure 14. Four Different Types of Two-Port Adaptors.

mented with reduced functionality to speed-up the simulations.

DSP430 reads the file <filter>.ASM and converts the declarations and instructions into an internal data representation. To avoid an extra assembly step to machine code, the instructions are interpreted here.

To support the simulation, several arrays of real numbers (y0..Y7) are provided. They are used to store stimuli sequences, filter input and filter output sequences, output spectrum and so on. Figure 16 shows the simulator and its related input and output files. The simulator is controlled by more than 40 commands, which can be executed with the help of a command file <test>.EF. The following list contains some important commands.^{4,5,6}

```

; ELP5 for MSP430
INPUT .equ      00010h ; I/O address
OUTP  .equ      0002Ah ; I/O address
ADRT0 .equ      00200h ; RAM address for T0
ADRT1 .equ      00202h ; RAM address for T1
ADRT2 .equ      00204h ; RAM address for T2
ADRT3 .equ      00206h ; RAM address for T3
ADRT4 .equ      00208h ; RAM address for T4
MAXREG .equ      R10   ; for saturation
MINREG .equ      R9    ; for saturation
OUTREG .equ      R8    ; for scratch and output samples
INPREG .equ      R7    ; for input samples
SCRA2  .equ      R6    ; scratch
SCRA1  .equ      R5    ; scratch
ACCU   .equ      R4    ; accumulator
MAXSAT .equ      01FFFh ; for saturation, constant +1-LSB
MINSAT .equ      0E000h ; for saturation, constant -1

;
MOV #MAXSAT, MAXREG ; copy constant into register
MOV #MINSAT, MINREG ; copy constant into register

.....
.....
.....
; 2. adaptor of upper branch (see Figure 20 , nodes 6, 7, 8, 9)
MOV ADRT4, ACCU ; load T4, node 8
SUB ADRT3, ACCU ; subtract T3, node 6
MOV ACCU, SCRA2 ; save multiplicand := T4 - T3

;
RRA ACCU ; start with gamma 4 = ( 0.0010101 ) binary
RRA ACCU
ADD SCRA2, ACCU ; accu :=  $m \times 2^{-2} + m$ 
RRA ACCU
RRA ACCU
ADD SCRA2, ACCU ; accu :=  $(m \times 2^{-2} + m) \times 2^{-2} + m$ 
RRA ACCU
RRA ACCU
RRA ACCU ; p :=  $((m \times 2^{-2} + m) \times 2^{-2} + m) \times 2^{-3}$ 

;
ADD ADRT4, ACCU ; n9 := p + T4
MOV ACCU, OUTREG ; node 9 is stored

;
SUB ADRT4, ACCU ; recalculate p := n9 - T4
ADD ADRT3, ACCU ; n7 := p + T3
MOV ACCU, ADRT4 ; new T4

; do saturation, if necessary
CMP MAXREG, ACCU ; if ACCU > MAXREG then ADRT4 := MAXREG
JN MAX2
MOV MAXREG, ADRT4
JMP MIN2
MAX2 CMP ACCU, MINREG ; if ACCU < MINREG then ADRT4 := MINREG
JN MIN2
MOV MINREG, ADRT4
MIN2 ...
.....
.....
.....
RRA ACCU ; divide by 2
MOV ACCU, OUTREG ; save output sample
MOV OUTREG, &OUTP ; write to output port

; end LP5C
; most of the comments above have been added manually to improve explanation

```

Figure 15. Parts of Example File ELP5C.ASM.

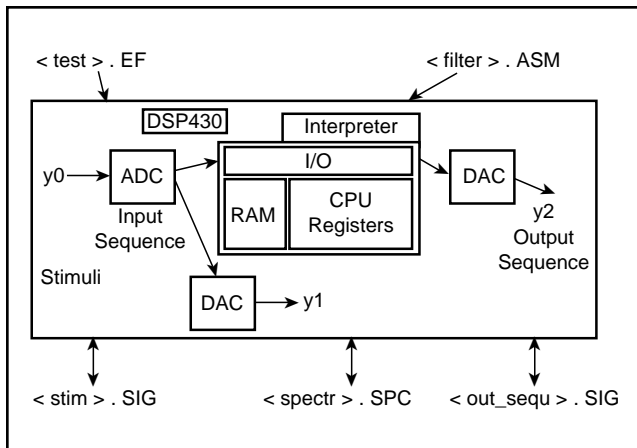


Figure 16. Simulator DSP430.

rosN	run simulator for N output samples
cnmax	change maximum number of samples used
ad	activate ADC
da	activate DAC
dd	display diagrams of y1 and y2 (filter input and output)
scale	change diagram scales
scaleY	change diagram Y-scale
sin	generate sine wave stimulus (fs, f, ampl., phase, offset)
square	generate square wave stimulus (fs, f, ampl., phase, offset)
saw	generate saw wave stimulus (fs, f, ampl., phase, offset)
tria	generate triangle wave stimulus (fs, f, ampl., phase, offset)

rect	generate rectangle stimulus (ampl., delay, offset)
random	generate pseudo random stimulus (sequence 2 ²⁰)
window	generate window function (n, window type)
ef	execute command file
showS	show array S (i.e. yS)
addSD	add arrays yD := yD + yS
subSD	subtract arrays yD := yD - yS
mulSD	multiply arrays yD := yD * yS
mufSD	multiply array by factor yD := yD * factor
copySD	copy yS to yD, yD := yS
saveS	store yS to file
loadD	read file and copy into yD
checklc	limit cycle checker
looptest	check filter under looped conditions
fou	Fourier transformation
x	exit simulator

For the generation of stimuli the commands “sin,” “square,” “saw,” “tria,” “rect,” “dirac,” “random” and “window” are implemented. Complex stimuli can be built, stored and restored from file <stim>.SIG. Figure 17 demonstrates the capability of generating a complex stimulus; the commands are:

sin 16000 300 0.5 0 0	{ sine wave 300Hz, fs=16kHz, in y0 }
copy03	{ copy signal to y3 }
sin 16000 5500 0.2 0 0	{ sine wave 5.5kHz in y0 }
add03	{ add y0 to y3 }
square 16000 66 0.5 -90 0.5	{ square wave 66Hz, ampl=0.5,

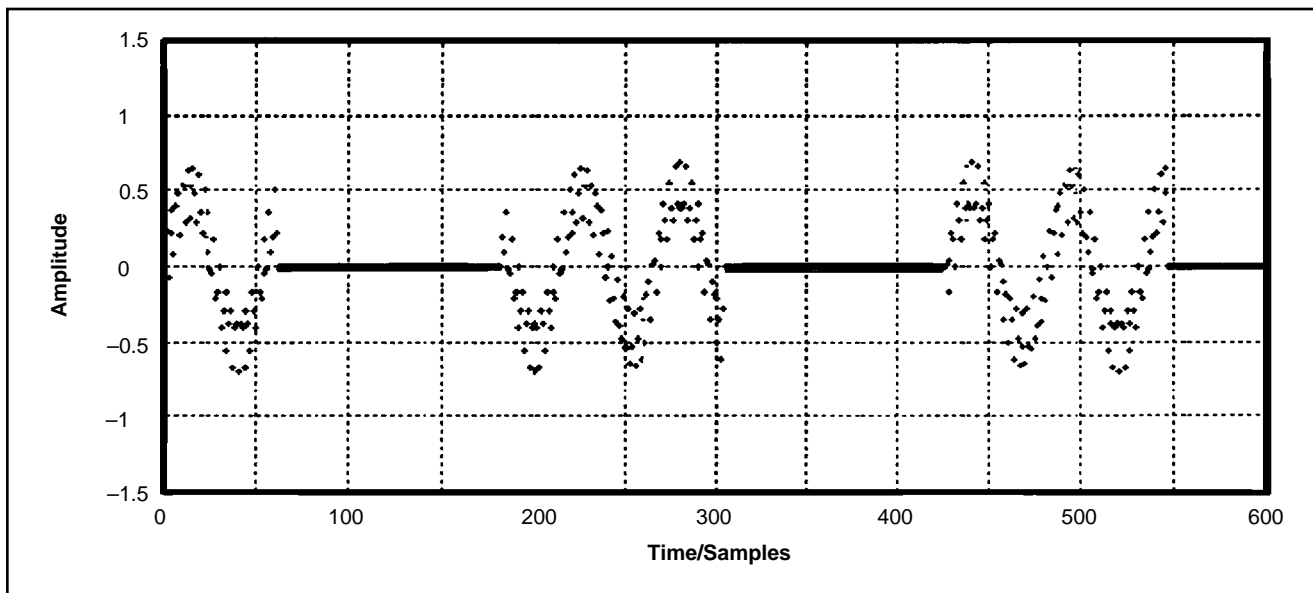


Figure 17. Complex Stimulus Generation.

```

mul03      phase=-90, offset=0.5 in y0 }
           { multiply y3 by y0 }
show3      { display sequence in y3 }

```

and the two superimposed sinewaves are amplitude-modulated by a square wave to produce a burst signal.

An ADC procedure converts a stimuli sequence, provided in real array y0, to an input sequence. This sequence is quantized to integer16 values which are fed into the MSP430 model. The quantized sequence also is converted back to real values stored in y1 for display purposes. The output sequence (integer16) is converted by a DAC procedure to a real sequence y2, too. The user can now view the input and output sequences in one diagram. So, the function of the filter can be tested easily.

If enough samples are available (e.g., 4200) an FFT over 4096 samples may be calculated (100 samples at the beginning of the output sequence should be omitted because the filter might not be in a steady state at the start). Different window functions (Hamming, Blackman, Blackman-Harris) may be applied to the sequences prior to the FFT. Spectra can be stored and restored; e.g., in file <outspec>.SPC to enable the user to compare different spectra from different simulation runs.

The simulator/analyzer DSP430 enables the user to perform a thorough evaluation of the filter's quality:^{5,10}

- Check gain
- Forced-response analysis
- Check dynamic range
- Check for harmonics
- Check for sub-harmonics
- Check for intermodulation
- Check zero-input granularity
- Check behaviour under looped conditions.

The next section also contains a demonstration of some of these checks.

Design of 5th Order Elliptic Lowpass ELP5

Using the program LWDFDES, the design of an elliptic lowpass filter shall be performed. The specification in file ELP5.SPC is as follows:

```

ELP5 lowpass 5.order  TITLE
5                      ELLIPTICAL
lp                      LOWPASS
16000.0                SAMPLE FREQUENCY
44.0                   AMIN DB (MIN. ATTENUATION, STOPBAND)

```

```

4600.0                FREQUENCY AT AMIN
0.045                 AMAX DB (MAX. ATTENUATION, PASSBAND)
3300.0                FREQUENCY AT AMAX
20.0                  PASSBAND DESIGN MARGIN

```

LWDFDES calculates the necessary filter order to 4.860; therefore the filter is 5th order (next odd number), - and the coefficients to

```

gamma 0 = 0.2758713
gamma 1 = -0.3316880
gamma 2 = 0.3255008
gamma 3 = -0.7739261
gamma 4 = 0.1641284

```

The user can now view the frequency response of stopband (*Figure 18*) and passband (*Figure 19*) in detail. One can see the margin between the filter curve and the specification. This margin can be used for the optimization of the coefficients; e.g., gamma 4 is changed to 0.1640625 = (0.0010101) binary. The complex task of coefficient optimization is covered in references 4 and 5.

LWDFDES finally generates the file ELP5.SFD, which contains the signal flow diagram of the low-pass filter (*Figure 20*) described by WDFL language:

```

.C ELP5 lowpass 5.order (optimized coefficients) input(1);
#
# upper branch
.B1
delay (2, 3, 1); T0
adaptor2 (1, 2, 3, 4, 0.25); gamma0
#
delay (7, 8, 1); T4
adaptor2 (6, 7, 8, 9, 0.1640625); gamma4
#
delay (5, 6, 1); T3
adaptor2 (4, 5, 9, 10, -0.765625); gamma3
#
# lower branch
.B2
delay (13, 14, 1); T2
adaptor2 (12, 13, 14, 15, 0.3125); gamma2
#
delay (11, 12, 1); T1
adaptor2 (1, 11, 15, 16, -0.3125); gamma1
#
adder (10, 16, 17);
shifter (17, 18, -1);
output (18);
.END

```

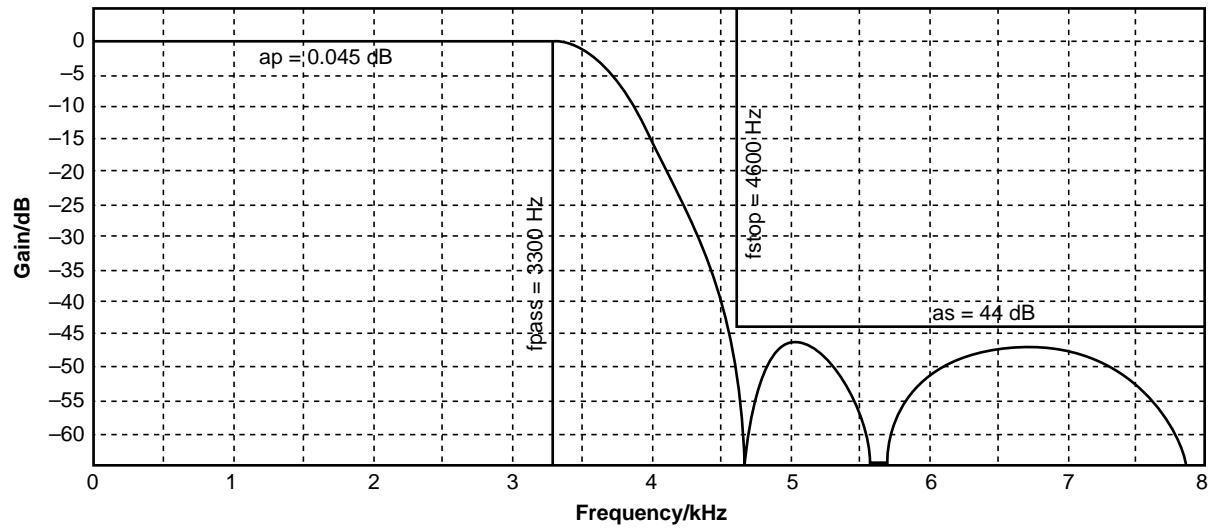


Figure 18. Elliptic 5th Order Low-Pass, Stopband.

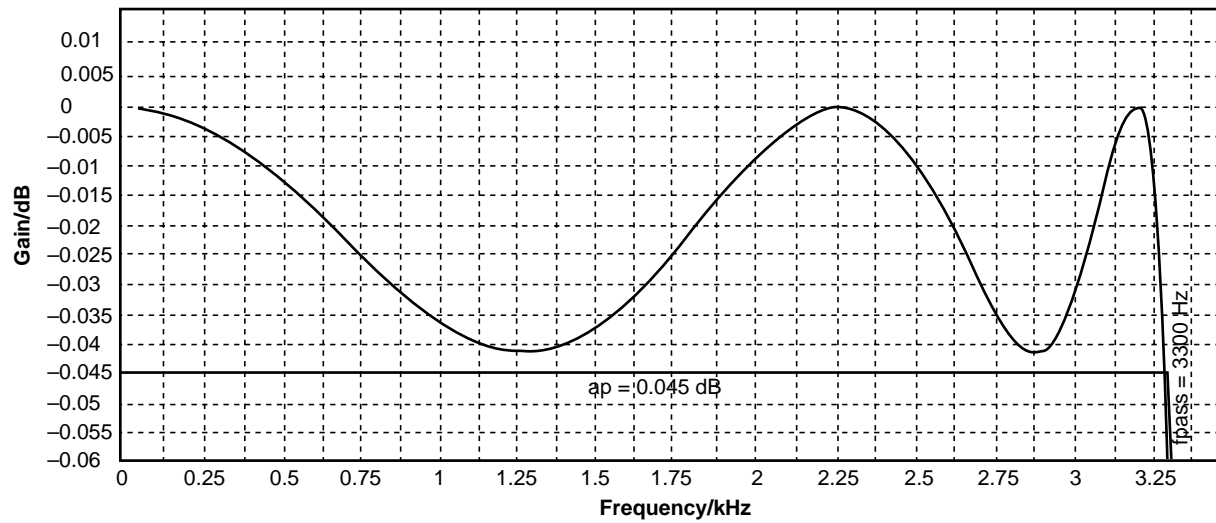


Figure 19. Elliptic 5th Order Low-Pass, Passband.

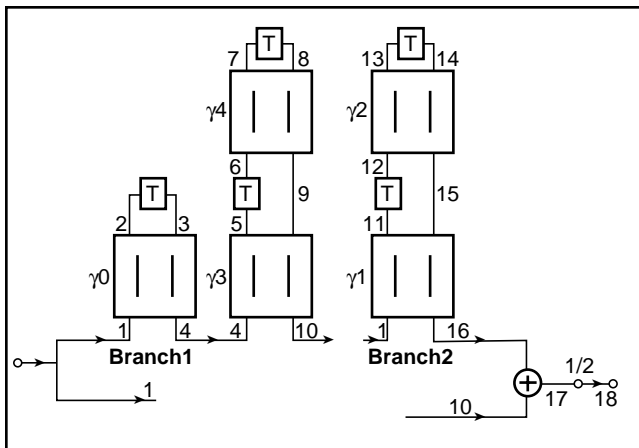


Figure 20. Elliptic 5th Order Low-Pass, Structure with Node Numbers.

The signal flow diagram is further input to the compiler LWDF_COMP. This specialized compiler expands the textual SFD into the final SFD of the LWDF (Figure 21). It produces the file ELP5.ASM containing the assembler program for the MSP430.

Now one can start the simulator program DSP430 and the file ELP5.ASM is read. The test and analysis of the lowpass ELP5 is performed.

At first, a fast and simple function test is done using the following commands:

```
sin 16000 300 0.999 0 0      { sine wave 300Hz }
ros100                       { run for 100 output samples }
sin 16000 5000 0.999 0 0     { sine wave 5kHz }
ros100
square 16000 500 0.999 0 0   { square wave 500Hz (y0 is overwritten) }
ros100
rect 0 0 0                    { zero input }
ros200
dd                             { display input and output }
```

The results are shown in Figure 22(a). After 20 samples the filter has reached a steady state and the output signal follows the input signal of 300 Hz. At the beginning the five state variables contained some random values, but that did not cause any problem. This “energy” stored in the variables is eliminated totally. The stopband behaviour is tested by the signal of 5 kHz. Figure 22(b) points out that the attenuation of 44 dB is obtained successfully (samples 175...200). With even higher resolution it is demon-

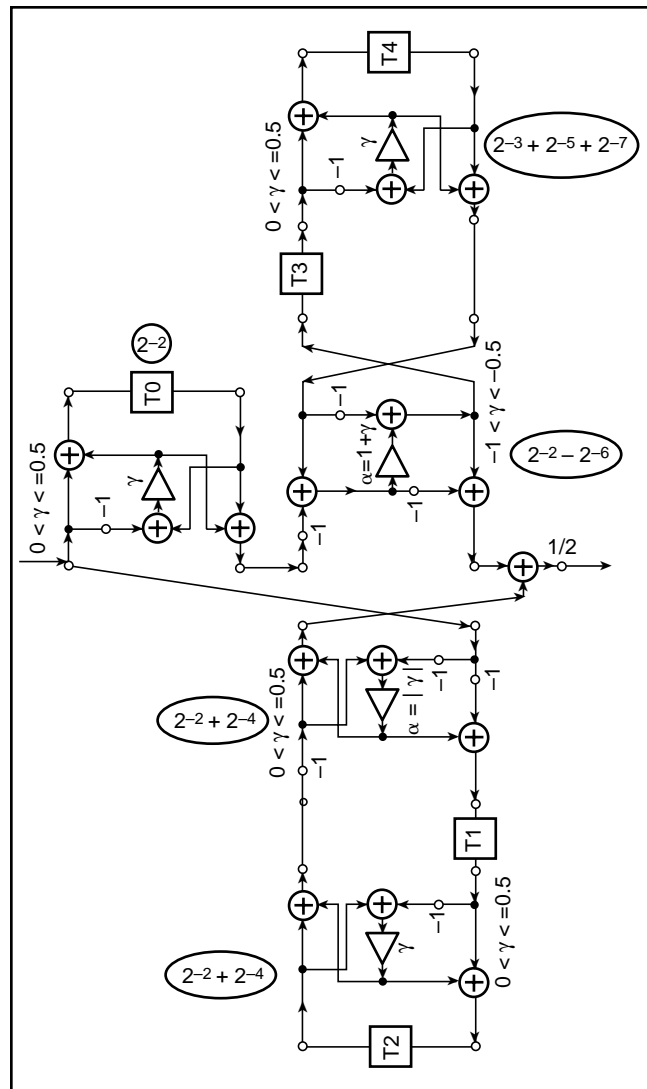


Figure 21. Elliptic 5th Order Low-Pass, Signal Flow Diagram.

strated (Figure 22(c)) that the LWDF needs only 60 cycles until the “energy” has dropped to the level of the input signal. It remains a constant amplitude of +1 LSB (= 0.122e-3), but no limit cycles can be observed. The forced-response stability property holds.

Another test for forced-response stability is shown in Figure 23(a). The filter, running in steady state at 300 Hz, is disturbed at the input with amplitude -1.0 for an interval of 10 samples. Also in this test, the filter recovers without problems. To calculate the error

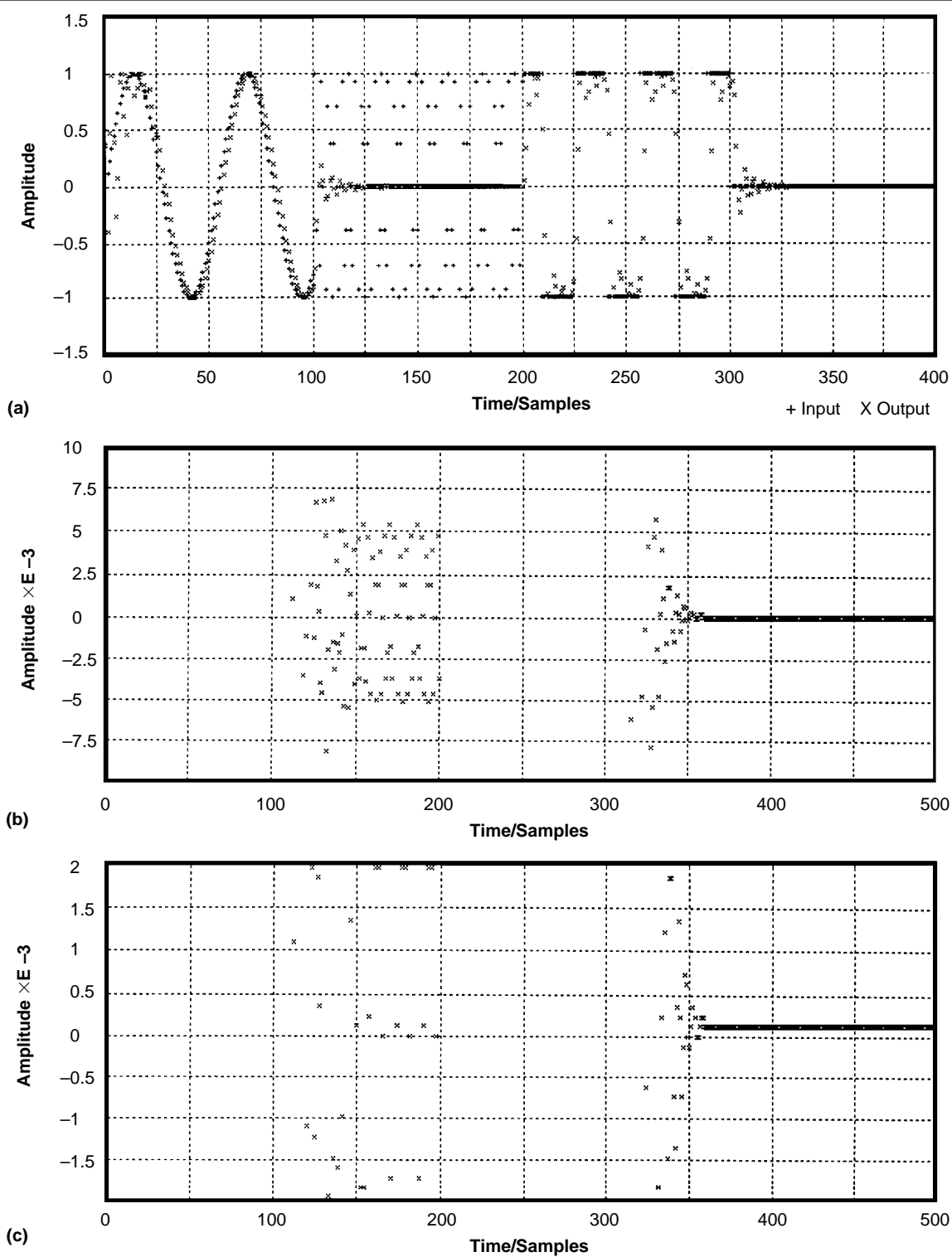
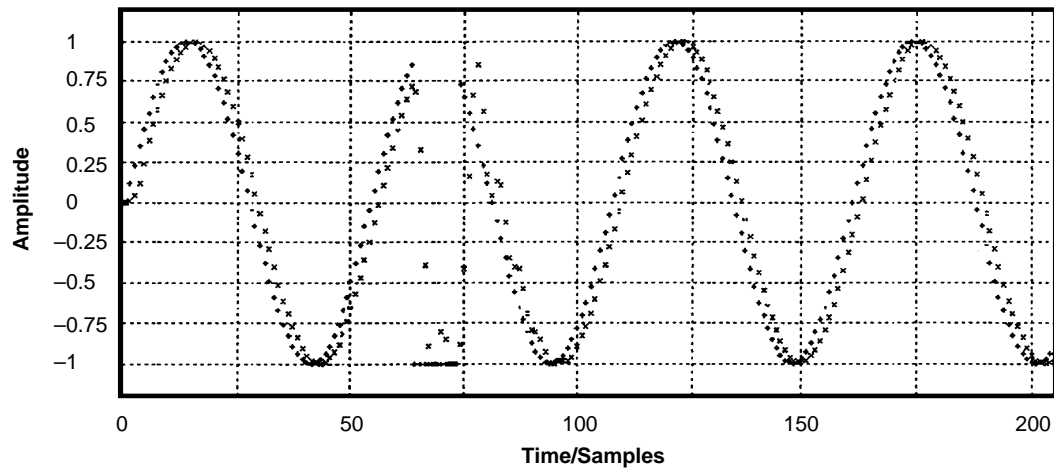
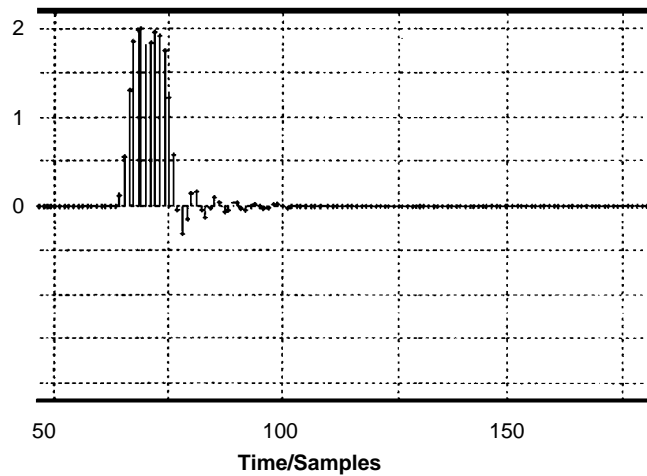


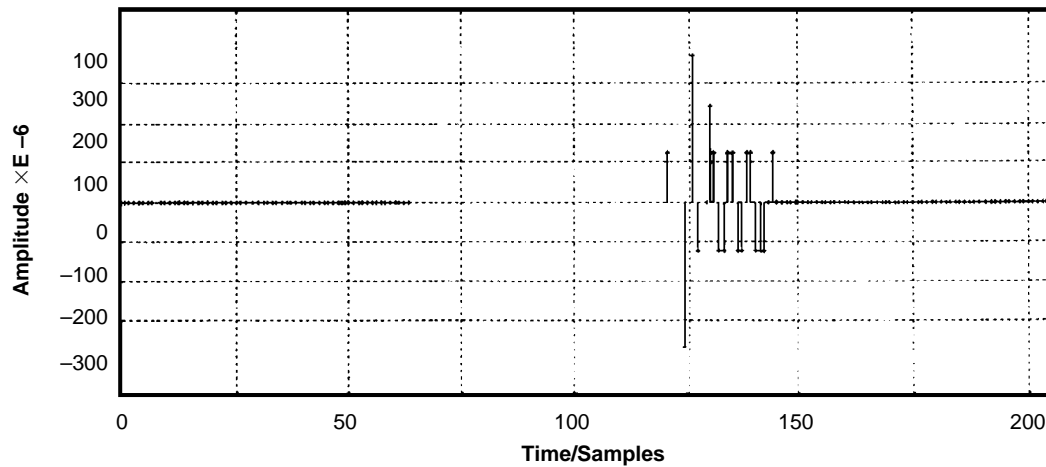
Figure 22. (a) Function Test of ELP5 with Different Stimuli; (b) Check Attenuation for 5kHz Signal; (c) Check Granularity for Input = 0.



(a)



(b)



(c)

Figure 23. Check for Forced Response Stability: (a) Disruption of Input Signal; (b) Difference Between Undisrupted and Disrupted Filter; (c) Same as Figure 23b, but y -unit = $0.1e-03$.

signal, the simulation runs without incident, and the output signals are subtracted. *Figure 23(b)* and *23(c)* present the error signal; the filter has “forgotten” the disruption after about 70 samples!

For the intermodulation test, two sine waves of 1934 Hz and 2860 Hz are superimposed; both have an amplitude of -10dB . After simulation of 4200 cycles an FFT with 4096 points and Blackman-Harris window is performed. The resulting spectrum (*Figure 24*) demonstrates that no intermodulation frequencies occur.

A special subroutine for zero-input limit cycle analysis is implemented in DSP430.^{4,5} The filter is fed with random values in the state variables; the number of samples necessary to reach the lowest stable output swing and the remaining granularity amplitude [reference 10, chapter 3.1.1] are measured.

Figure 25(a) demonstrates that ELP5 needs less than 110 samples to reduce the initial energy. *Figure 25(b)* and *(c)* show that the remaining granularity does not exceed the range of ± 2 LSB. This granularity is caused by the arithmetic used with value truncation during the Shift&Add multiplication algorithm, but it can be tolerated. (If magnitude truncation were applied, additional instructions must be added and the noise produced by the rounding is then correlated to the input signal.)

The next test evaluates the response of the filter

with regard to a sharp amplitude change (step signal). The input is changed from 0.0 to $0.984375 = 1 - 128 \times \text{LSB}$. The resulting output signal is presented in *Figure 26(a)*. The sudden, high input of “energy” to the filter results in overflow, but due to the saturation and inherent stability of the LWDF this effect is minimized. The output settles exactly to the value 0.984375. *Figure 26(b)* demonstrates this by means of finer resolution.

Another special subroutine checks the system behaviour under looped conditions.^{1,5} The filter is stimulated with certain initial values in the state variables. The output is connected to the input via a certain number of delays and the output is monitored (*Figure 27(a)* and *(b)*). The “energy” in the filter is decreasing; no positive feedback and no oscillation can be observed. Also, the filter can be excited by a stimulus (e.g., preference frequency) and after a certain number of samples it is switched into looped conditions and monitored.

Conclusions

The new sensor-signal processor MSP430 has a wordlength of 16 bits. Using its integrated ADC, it can measure physical quantities using external sensors. Usually, averaging is performed on the measured data. Averaging is a digital filtering process with low attenuation in the stopband. If a greater cal-

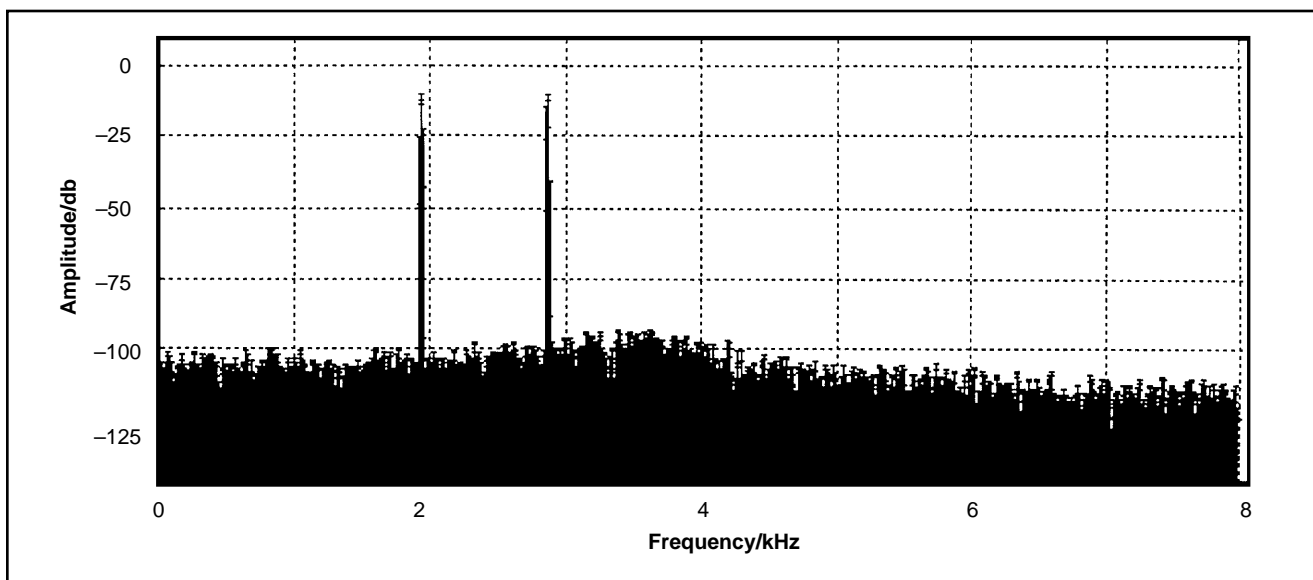


Figure 24. Intermodulation Test (-10dB @ 1934 Hz and 2860 Hz).

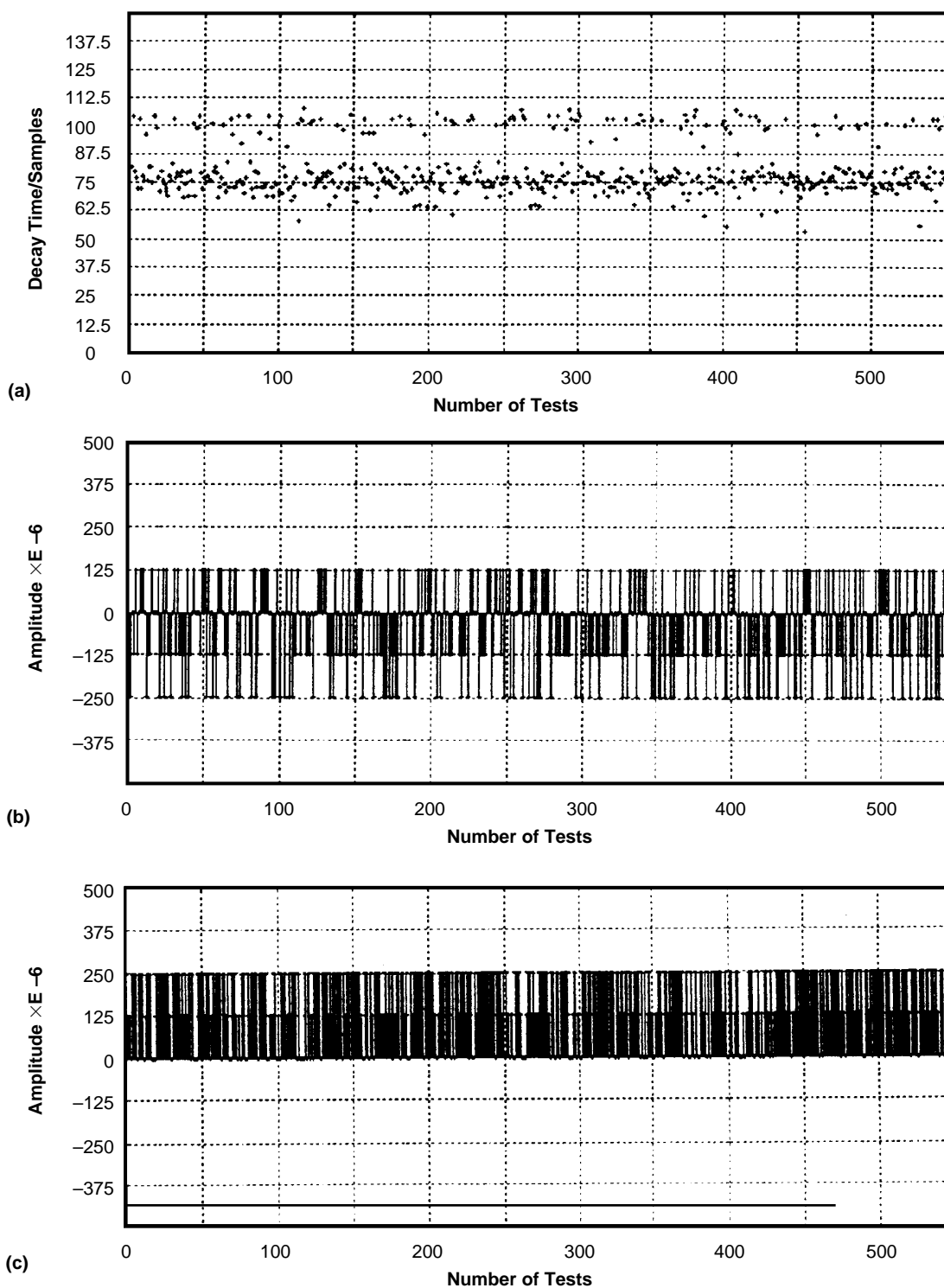


Figure 25. (a) Limit-Cycle Check (550 Tests); (b) Minimum Amplitudes; (c) Maximum Amplitudes.

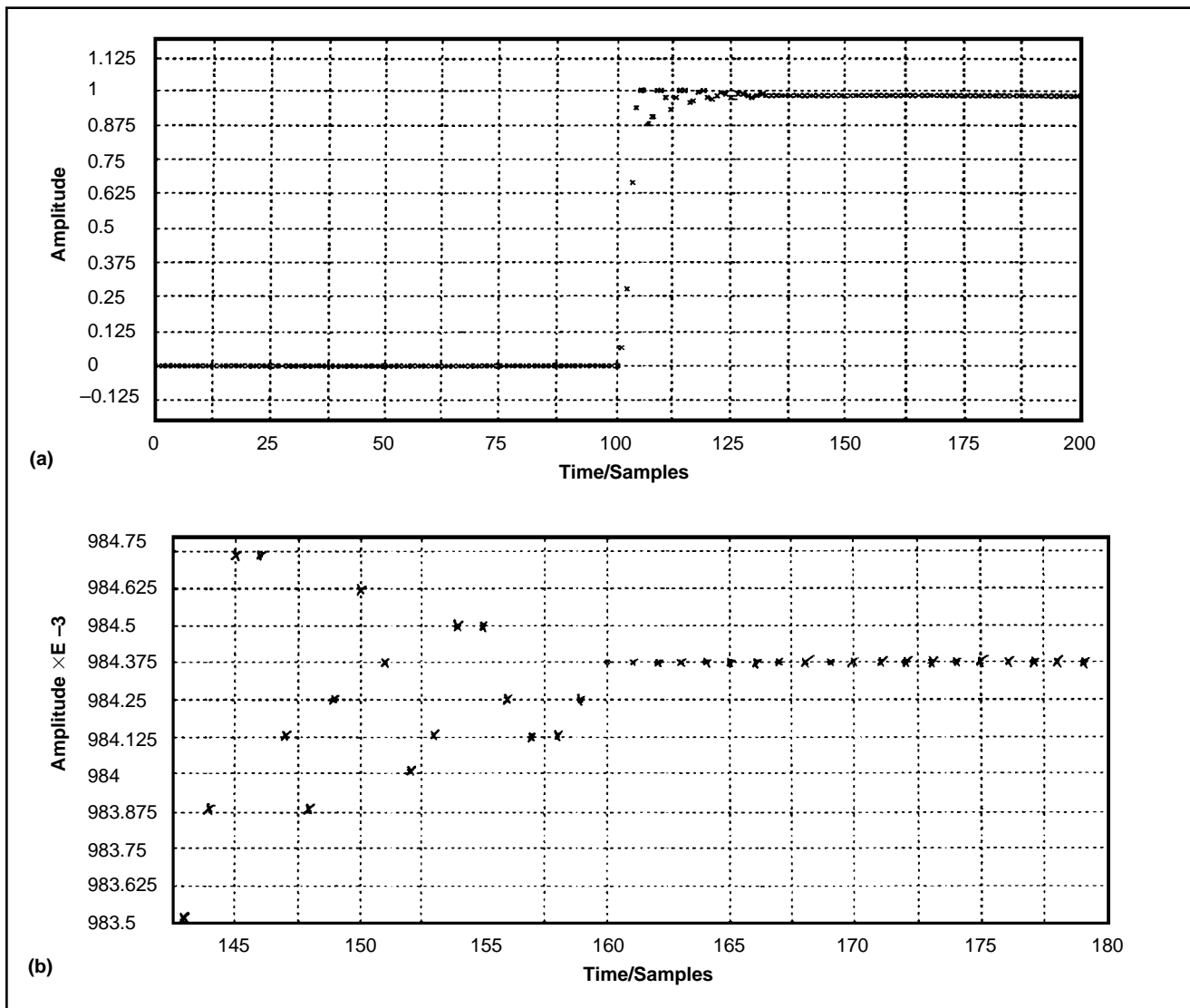


Figure 26. (a) Output Signal of ELP5 for Rectangular Input; (b) Same as Figure 26(a) but y-Unit = $0.125e-03$.

ulation effort can be tolerated, e.g., multiplications by constant coefficients, lattice wave digital filters are especially suited because:

- They can be designed by a set of simple formulas
- WDFs guarantee stability and robustness
- They provide low sensitivity in passband
- They support Shift&Add and Horner's scheme.

The MSP430 also supports Shift&Add multiplication with its instruction RRA for arithmetic right-shift.

An LWDF design and simulation system for the

MSP430 has been developed on the PC, based on the software for RISP.⁵

The filter design program LWDFDES calculates order and coefficients of a filter after specification of critical frequencies and attenuations. The obtained filter curve is displayed and the filter structure is described textually. This description is transformed into assembler language of the MSP430 to free the user from time-consuming and error-prone work.

The program DSP430 models the MSP430. To speed-up the simulations the model is reduced to:

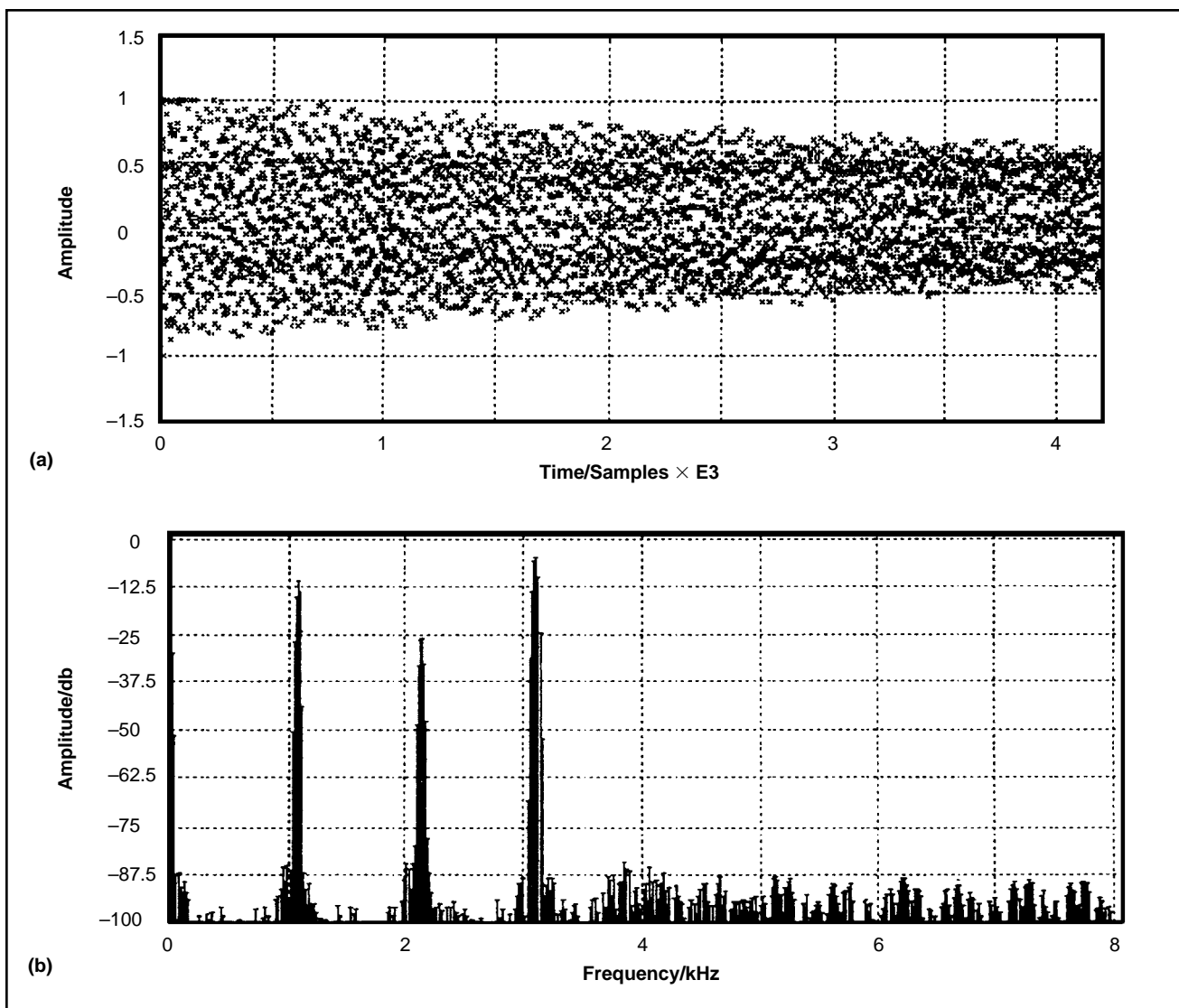


Figure 27. (a) Output Signal of ELP5 Under Looped Conditions; (b) Spectrum of the Signal in Figure 27a.

- CPU (16 bits)
- Set of registers (16 words of 16 bits)
- Subset of the MSP430 instruction set
- RAM
- I/O_Module.

During the simulation the typical DSP system ADC \rightarrow DSP \rightarrow DAC is modelled. First, the assembler file (generated by LWDF_COMP) is transformed into an internal data structure. This is interpreted during the simulation allowing the omission of MSP430 machine code.

The simulator offers many commands for the interactive control and the control by means of command files. These commands support, e.g.,

- Loading the filter program
- Generation of complex input sequences (stimuli)
- Simulation of N filter cycles
- Display of input and output sequences
- Calculation of spectra from sequences
- Analysis of filter stability and robustness
- Analysis of granularity
- Analysis of filter stability under looped conditions

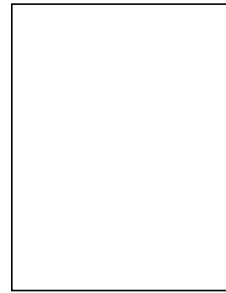
- Save/load of sequences or spectra
- Comparison of sequences or spectra.

The example of a 5th order elliptical lowpass filter demonstrates the performance of the design and simulation programs. The example shows the inherent stability properties of WDFs. In parallel, it is shown that wave digital filters can be run successfully on microprocessors without a hardware multiplier, i.e., the MSP430. About 18 instructions are needed per filter order.

The program system WDF430 allows the top-down development of lattice wave digital filters from filter specification to the tested assembler program.

References

1. A. Fettweis, "Wave Digital Filters: Theory and Practice," *IEEE Proceedings*, Vol. 74, No. 2, February 1986, pp. 270-327.
2. U. Kaiser, "Wave Digital Filters and their Significance for Customized Digital Signal Processing," *Texas Instruments Engineering Journal*, September-October 1985, Vol. 2, No. 5, pp. 29-44 and November-December 1985, Vol. 2, No. 6, pp. 12-33.
3. L. Gazsi, "Explicit Formulas for Lattice Wave Digital Filters," *IEEE Trans. Circuits and Systems*, Vol. CAS-32, No. 1, pp. 68-88, January 1985.
4. U. Kaiser, "A CAD System for the Design of Digital Filter Algorithms for the Reduced-Instruction Set Signal Processor RISP," *Proceedings of ISCAS '91*, June 1991, Singapore, pp. 45-48.
5. U. Kaiser, "RISP: Eine digitale Signalprozessor-Architektur mit reduziertem Befehlssatz fuer Wellendigitalfilter," Dissertation, Fakultät fuer Elektrotechnik, Ruhr-Universitaet Bochum, 1991.
6. W. Meixner, Entwurfs- und Simulationssystem fuer Bruecken-Wellen-digitalfilter fuer den Sensor-Signalprozessor MSP430, Fachhochschule Muenchen und Texas Instruments Deutschland, Diplomarbeit, October 1993.
7. *The MSP430Cxxx, A Mixed Signal Processor*, User's Guide Preliminary Specification, Rev. 0.43.
8. Horst Diewald, "System Design and Architecture of a Mixed Signal Processor," Euro Technical Conference (ETC), October 1993, Freising.
9. H-J Rothermel, New 16-bit RISC CPU for Low Power Industrial Applications, Euro Technical Conference (ETC), October 1993, Freising.
10. A. Fettweis, "On Assessing Robustness of Recursive Digital Filters," *European Transactions on Telecommunications*, Vol. 1, No. 2, March-April 1990, pp. 103-109.
11. E. Dijkstra, L. Cardoletti, O. Nys, C. Piguet, M. Degrauwe, "Wave Digital Decimation Filters in Oversampled A/D Converters," *Proceedings of ISCAS '88*, June 1988, pp. 2327-2330.



Ulrich Kaiser

Ulrich Kaiser is a CMOS design engineer in the mixed-signal design organization. He was elected Member Group Technical Staff in 1989.

After four years of teaching at vocational school in Soest, Ulrich joined TID in November 1980 as a software engineer for custom integrated circuits. He wrote several programs for the improvement of schematic capturing and layout generation programs for TID's sensor microcomputers. He also designed the digital filter parts for a digital codec.

In 1987 he changed to CMOS chip design, developing several transponder chips — from read/only through read/write to selectively addressable read/write multi-page transponders. He holds four patents with six pending.

Ulrich is a member of the IEEE. He is author or co-author of six publications in the field of wave digital filters and two about TIRIS transponder chip design. He supported four diploma theses in the field of wave digital filters.

After first experiments with soldering in 1963 and with TTL (TI's 7400 and 7490) in 1968, he received his B.Eng. from the Technische Hochschule Aachen in 1971. He received the M.Eng. from the Ruhr-Universitaet Bochum in 1974, his mastership for vocational school from the Federal State of Nordrhein-Westfalen in 1976, and a Ph.D. from the Ruhr-Universitaet Bochum in 1991. His Ph.D. dissertation was about a specialized RISC DSP architecture tailored to wave digital filters.

Besides CMOS chip design his interests are digital signal processing, CAD for MOS, programming languages and optimization.

Ulrich and his wife Regina enjoy reading and vacationing. They have a daughter, 17, a son, 20, and a foster-daughter, 29. □