LIGO-T2400089-v1 Advanced LIGO March 2024

IIR Filter Bank for FPGAs

Daniel Sigg

Table of Contents

1 Overview 3
2 The Filter Engine 3
3 The Filter Bank 5
3.1 Coefficient Memory 6
3.2 Microcode 7
3.3 Changing Coefficients 7
4 Pipelines 8
4.1 5x Configuration 8
4.2 2x Configuration 8
4.3 5x Configuration with SOS Switches Enabled 10

1 Overview

An IIR filter can be implemented as a series of second order sections (SOS). Here we look at an FPGA implementation written in VHDL code in Vivado and targeting the Xilinx 7-series and Ultrascale devices. The filter implementation uses a fixed-point number representation with 52-bits that is roughly equivalent to a double precision number.

Filter coefficients can be generated from a foton filter file using the CoeffGen52_2x program. The coefficient memory is a dual ported RAM that has a 64-bit wide read-only mode on the filter side, and a 32/64-bit wide read-write mode on the interface side. Typically, the interface side is accessible through the PCIe bus.

2 The Filter Engine

Following Reference[1] we write a filter as:

$$H(z) = g \prod_{k=1}^{N} c_{0k} \frac{1 + b_{1k}\, z^{-1} + b_{2k}\, z^{-2}}{1 - a_{1k}\, z^{-1} - a_{2k}\, z^{-2}} \tag{1}$$

In the above equation we separated out an overall gain factor, g, and individual gain factors, c_{0k}. The idea is to implement the multiplications c_{ok} as an efficient shift operation which reduces a second order section to 4 MAC (multiply-accumulate) operations that can be implemented efficiently using the DSP slices available in the FPGA. This shift operation for each SOS is important to keep the scaling near unity at a characteristic frequency and to take full advantage of all the digits in the fixed-point representation.

Restricting ourselves to a single second order section and denoting input values by x_i and output values with y_i the filter section can be written as:

$$y_i = c_0(x_i + b_1 x_{i-1} + b_2 x_{i-2}) + a_1 y_{i-1} + a_2 y_{i-2} \tag{2}$$

Since we have chosen the direct form I structure[2] rather than the more traditional direct form II, both input and output values serve as history values. And, since we set up the filter to be near unity gain, a suitable fixed-point representation is straight forward. Since the output value of the first second order section is the input value of the next section, the entire filter can be implemented using a single multiply-accumulate pipeline, where the very first input value must be scaled by the gain g (see Figure 1). History output values are shared with the history input values of the next section; therefore, we are only requiring two additional history values compared to the direct form II structure.

The implementation of the filter engine includes optional output registers, a separate output for the old input value, and optional switches that allow to turn off individual sections. Since the gain of a second order section isn't unity, a disabled SOS has to be replaced by a simple gain multiplication using a bypass path.

Additional logic is used to process overflow conditions. An overflow condition occurs if the result of the acculamator cannot fit into the bits of the output value. An overflow is also signaled if the input indicates an overflow from a previous processing stage.

[1] D. Sigg, "Implementing an IIR Filter in Hardware," LIGO-T050060.
[2] A.V. Oppenheim & R.W. Schafer, "Discrete-Time Signal Processing".

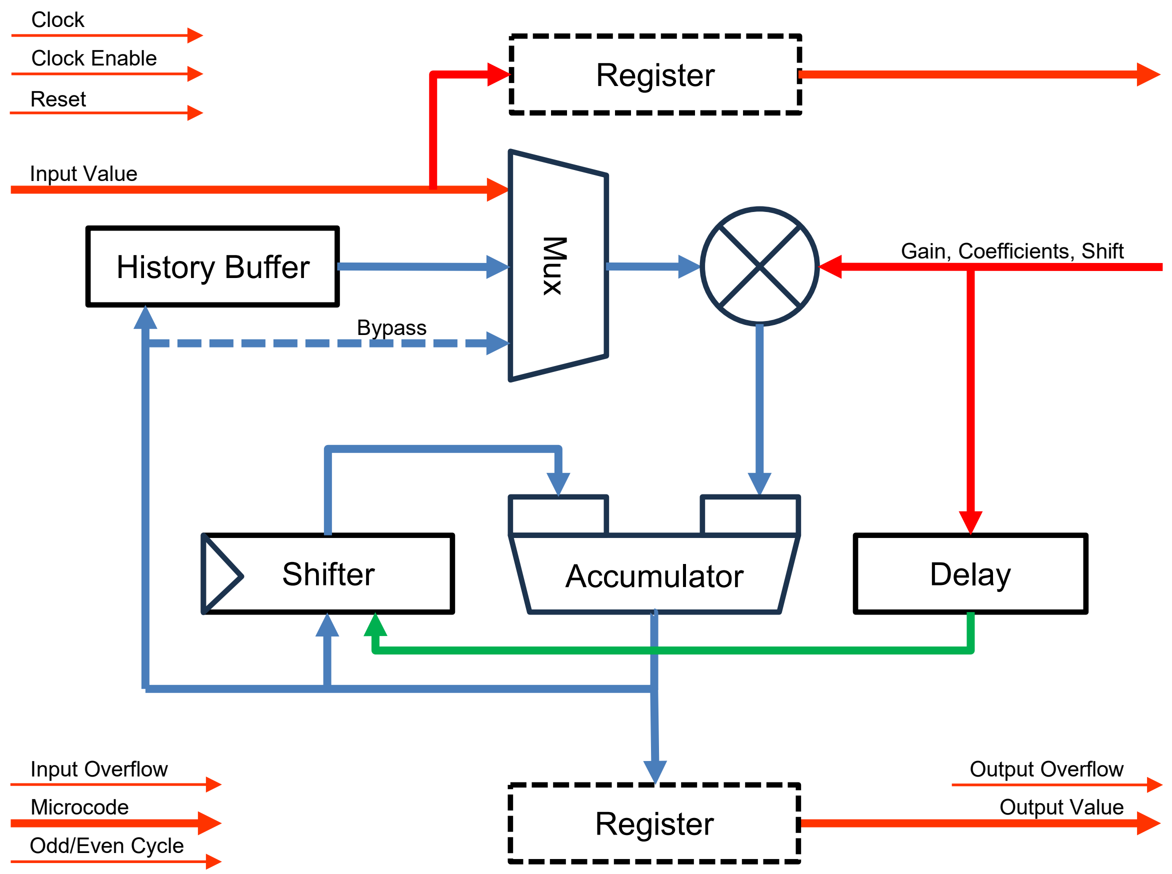

Figure 1: Filter Engine.

The parameters that are used to configure the filter engine:

Parameter	Type	Default	Description
InputWidth	Natural	32	Bit width of filter input and output values
OutputReg	Boolean	True	Determines if an output register is implemented
Cycles	Natural	6	Base 2 logarithm of number clock cycles within a filter cycle Can be 3 (8 cycles), 4 (16 cycles), 5, 6, 7, 8 or 9 (512 cycles)
ShiftBits	Natural	6	Base 2 logarithm of number bits used in the barrel shifter Shift values range from $2^{-(\text{ShiftBits}-1)}$ to $2^{\text{ShiftBits}-1}-1$
Multiplier	String	5x	Multiplier configuration: “2x” uses 2 multipliers (DSP slices) in parallel, “5x” uses 5 slices
DSP	String	DSP48E1	DSP slice configuration: “DSP48E1” for 7-series devices and “DSP48E2” for Ultrascale

If the “2x” configuration is selected, The OutputReg parameter should be set to True, since the availability of the filter output value depends on the Cycles count. With a register the filter value will always be available at the last cycle of the filter cycle or later. For a “5x” multiplier the output register can be omitted if the filter value is clocked out during the last cycle of the filter cycle. If a register is used, the filter value will be available on the first cycle of the next filter cycle. Also, for the “2x” configuration the minimum Cycles count is 5 (32).

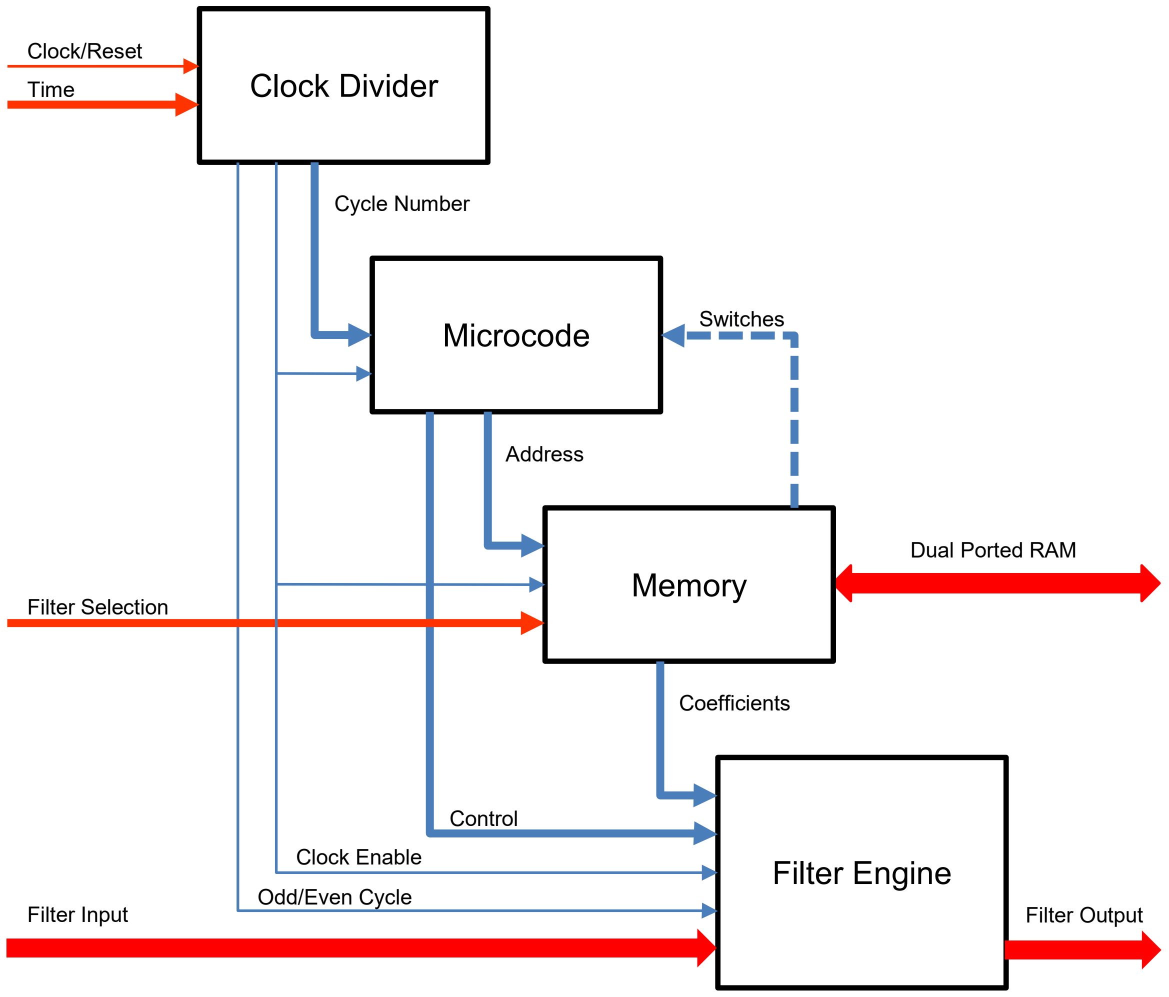

Figure 2: Filter Bank

3 The Filter Bank

The IIR_filter_bank module implements a bank of IIR filters, where each filter is using identical filter coefficients. A filter bank consists of a clock divider, a microcode generator, a memory block for the filter coefficients, and an IIR filter engine that can work on multiple input values in parallel (see Figure 1).

The clock divider generates a cycle count that enumerates through all filter steps and that is aligned to the 1 PPS of the GPS time. It generates a clock enable when the filter is intended to run slower than the clock rate. An odd/even cycle bit is used to switch the filter history values at the end of a cycle.

The microcode block generates both the control signals and the coefficient memory address based on the cycle count.

The memory block implements either a ROM or a dual ported RAM that contains the gain, filter coefficients and shift values. Typically, the memory block is larger than required for a filter. In this case the parameters of multiple filters can be loaded into the memory and selected using a filter selection input.

The parameters that are used to configure the filter bank:

Parameter	Type	Default	Description
RESOLUTION	Natural	26	Timing clock resolution or period is $2^{-\mathrm{RESOLUTION}}$ seconds
Cycles	Natural	6	Base 2 logarithm of number clock cycles within a filter cycle Can be 3 (8 cycles), 4 (16 cycles), 5, 6, 7, 8 or 9 (512 cycles)
LBD	Natural	0	Base 2 logarithm of clock divider ratio to run filter slower than clock rate; Must be lower or equal RESOLUTION
Filters	Positive	1	Number of filters in the filter bank
FilterWidth	Natural	32	Bit width of filter input and output values
FilterReg	Boolean	True	Determines if an output register is implemented
ShiftBits	Natural	6	Base 2 logarithm of number bits used in the barrel shifter Shift values range from $2^{-(\mathrm{ShiftBits}-1)}$ to $2^{\mathrm{ShiftBits}-1}$-1
Multiplier	String	5x	Multiplier configuration: “2x” uses 2 multipliers (DSP slices) in parallel, “5x” uses 5 slices
DSP	String	DSP48E1	DSP slice configuration: “DSP48E1” for 7-series devices and “DSP48E2” for Ultrascale
GainSwitch	Boolean	False	Enable setting the filter gain to 0
SosSwitch	Boolean	False	Enable individual switching of SOS stages
ResetType	String	Full	Reset type: “Full” - apply the reset for a full cycle and release at the end of a cycle, “Instant” - apply only as long as the reset line is active, “Goertzel” - apply every second during the last cycle
MemoryType	String	TPRAM	Memory type used: “SPROM” (ROM) or “TDPRAM” (RAM)
MemoryDepth	Natural	10	Memory depth in number of address bits: The minimum is 10 for TDPRAM, and 9 for SPROM
MemoryBank	Natural	7	Memory bank size in address bits: The minimum is 6, and up to the MemoryDepth Must be long enough to contain the required SOS sections
MemoryDelay	Natural	0	Additional delay in the output of the coefficient memory: 0 - data is available on the next clock cycle, 1 - uses an output register
MemoryFile	String	“none”	Memory initialization file
MemoryReg	Natural	1	Memory pipeline depth on the interface side of the dual port RAM: 1 - use of memory latch only; 2 - use of output register
DataBWidth	Natural	32	Data width on B side

3.1 Coefficient Memory

The coefficient memory is organized by filter sets. Each filter set covers an entire IIR filter. It is then possible to switch between filter sets using the filter selection input to load alternative filter coefficients. Typically this filter selection input is mapped into the PCIe parameter space and can be accessed through the host computer.

If the SosSwitch parameters is enabled, some space usually used for filter sets will be used for the alternate gain coefficients instead. This means that in this case the coefficient memory needs to be big enough to cover at least 2 filter sets. If big enough for just 2 filter sets, the space of the second filter set will be used for the alternate gain coefficients. If big enough for 4 or more filter sets, a quarter of the available space will be used for the alternate gain coefficients. Currently, SOS switches are only supported when the multiplier is set to “5x”.

The space of each filter set is sub divided into groups of 4 filter coefficients. The first group is used for the zero value, 2 64-bit values reserving up to 128 bits for the gain and SOS switches, and the overall gain coefficient, respectively. Each subsequent group represents a single SOS section with its b_1, b_2, a_1, and a_2 coefficients, respectively. The c_0 value is stored as part of the a_2 coefficient.

Gain and filter coefficients are represented by 53-bit fixed point values that are shifted left into a 64-bit quad word. The 53-bit coefficient values are two's complements values with the MSB being the sign bit and with the decimal point being located between the 2nd and 3rd bit. This allows for coefficients to represent any value between -2.0 (inclusive) and +2.0 (exclusive). The shift value c_0 is encoded in the unused lower 8-bits of the a_2 coefficient. This shift value is an 8-bit two's complements value ranging from -32 to +31. This allows for multiplications in the range of 2^{-32} to 2^{31} in multiples of 2.

The SOS switches are bit encoded in the 2 64-bit values in the first group of coefficients. The first value represents the gain switch at the LSB, then the SOS switches for the first SOS up to the 31st SOS. The second value represents the SOS switches for the 32nd SOS up to the 63rd SOS. A zero for the gain bit indicates that the normal gain is used, whereas a one indicates that the normal gain value is replaced by zero. For the SOS switches, a zero indicates a normal calculation of the section, whereas a one indicates that the section is off and the alternate gain value is used instead.

If filters are designed using the LIGO foton program, the CoeffGen52_2x program[3] can be used to generate the mem files needed by Vivado to preload the coefficient memory.

3.2 Microcode

The microcode block generates both the control signals and the coefficient memory address based on the cycle count. Examples for the "5x" multiplier configuration can be found in section 4.1, whereas section 4.2 shows the "2x" configuration. Section 4.3 shows an example where some of the SOS switches are set to off.

For the "5x" configuration 4 cycles are needed to calculate a single SOS, whereas for the "2x" configuration 12 cycles are needed. For the "5x" configuration 4 cycles are needed to apply the overall gain and flush the pipeline, whereas the "2x" configuration requires 7 cycles of overhead. The table below shows the available SOSs for filters with different cycle count:

Config.	8 cycles	16 cycles	32 cycles	64 cycles	128 cycles	256 cycles	>256 cyc.
2x	—	—	2	4	10	20	31
5x	1	3	7	15	31	63	127

3.3 Changing Coefficients

Changing coefficients in a filter memory section that is currently used for computing an IIR filter is not recommended. There is no guarantee that the coefficients will be updated at the start of the filter calculation and not half way through. If new filter coefficients are to be loaded for an online filter, one should load the new filter coefficients into an unused filter memory section and then use the filter selection bits to switch to this newly prepared filter. Changing the active filter is guaranteed to only take place at the beginning of a filter computation and is therefore save.

[3] https://svn.ligo.caltech.edu/svn/Altium-D2D/Software/FPGA-VHDL/PCIeTiming/IIRFilter/CoeffGen/CoeffGen52_2x

4 Pipelines

4.1 5x Configuration

$Y_{-mn} = X_{-m(n+1)}$ with m = 0 or 1 and n = 0, 1, … max sos.
Address sssssssh: sssssss is SOS numbering from 0 to 127; h=1 is −2 history, this bit is inverted every other cycle
Control: v is write result value, h is history write, r is accumulator reset, l is accumulator load for shift,
m is 1-save sum2 to history/0-save accumulator to history, and i is 0-input/1-history.

#	History		Input	Coefficients		Multiplier	Sum1	Sum2	Accumulator	Control
	Addr	Mem	Mux	Mem	Reg					vhrl_0m0i
-3	0b0000000_0	—	—	S_1	—	—	$a_{12}*Y_{-12}$	$a_{22}*Y_{-22}$	$c_{02}\ \Sigma X$	0b0000_0001
-2	0b0000000_1	X_{-20}	—	b_{20}	—	—	—	$a_{12}*Y_{-12}$	$c_{02}\ \Sigma X+a_{22}*Y_{-22}$	0b0000_0001
-1	0b0000000_0	X_{-10}	X_{-20}	b_{10}	b_{20}	—	—	—	$Y_{02} = c_{04}\ \Sigma X+ \Sigma Y+\ldots \rightarrow$ next Y_{-22}	0b1100_0001
0	0b0000001_1	—	X_{-10}	g	b_{10}	$b_{20}*X_{-20}$	—	—	—	0b0000_0000
1	0b0000001_0	Y_{-20}	X_{00}	a_{20}/c_{00}	g	$b_{10}*X_{-10}$	$b_{20}*X_{-20}$	—	—	0b0010_0001
2	0b0000001_1	Y_{-10}	Y_{-20}	a_{10}	a_{20}/c_{00}	$g*X_{00}$	$b_{10}*X_{-10}$	$b_{20}*X_{-20}$	0	0b0000_0001
3	0b0000001_0	X_{-21}	Y_{-10}	b_{21}	a_{10}	$a_{20}*Y_{-20}$	$g*X_{00}$	$b_{10}*X_{-10}$	$b_{20}*X_{-20}$	0b0000_0001
4	0b0000010_1	X_{-11}	X_{-21}	b_{11}	b_{21}	$a_{10}*Y_{-10}$	$a_{20}*Y_{-20}$	$g*X_{00}$	$b_{20}*X_{-20}+b_{10}*X_{-10}$	0b0100_0101
5	0b0000010_0	Y_{-21}	X_{-11}	a_{21}/c_{01}	b_{11}	$b_{21}*X_{-21}$	$a_{10}*Y_{-10}$	$a_{20}*Y_{-20}$	$b_{20}*X_{-20}+b_{10}*X_{-10}+ g*X_{00}$	0b0001_0001
6	0b0000010_1	Y_{-11}	Y_{-21}	a_{11}	a_{21}/c_{01}	$b_{11}*X_{-11}$	$b_{21}*X_{-21}$	$a_{10}*Y_{-10}$	$c_{00}\ \Sigma X+a_{20}*Y_{-20}$	0b0000_0001
7	0b0000010_0	X_{-22}	Y_{-11}	b_{22}	a_{11}	$a_{21}*Y_{-21}$	$b_{11}*X_{-11}$	$b_{21}*X_{-21}$	$Y_{00} = X_{01} = c_{00}\ \Sigma X+ \Sigma Y \rightarrow$ next X_{-21}	0b0100_0001
8	0b0000011_1	X_{-12}	X_{-22}	b_{12}	b_{22}	$a_{11}*Y_{-11}$	$a_{21}*Y_{-21}$	$b_{11}*X_{-11}$	$Y_{00}+ b_{21}*X_{-21}$	0b0000_0001
9	0b0000011_0	Y_{-22}	X_{-12}	a_{22}/c_{02}	b_{12}	$b_{22}*X_{-22}$	$a_{11}*Y_{-11}$	$a_{21}*Y_{-21}$	$Y_{00}+ b_{21}*X_{-21}+b_{10}*X_{-10}$	0b0001_0001
10	0b0000011_1	Y_{-12}	Y_{-22}	a_{12}	a_{22}/c_{02}	$b_{12}*X_{-12}$	$b_{22}*X_{-22}$	$a_{11}*Y_{-11}$	$c_{01}\ \Sigma X+a_{21}*Y_{-21}$	0b0000_0001
11	0b0000011_0		Y_{-12}	S_0	a_{12}	$a_{22}*Y_{-22}$	$b_{12}*X_{-12}$	$b_{22}*X_{-22}$	$Y_{01} = c_{01}\ \Sigma X+ \Sigma Y+\ldots \rightarrow$ next Y_{-21}	0b0100_0001
12	—	—	—	S_1	—	$a_{12}*Y_{-12}$	$a_{22}*Y_{-22}$	$b_{12}*X_{-12}$	$Y_{01}+b_{22}*X_{-22}$	0b0000_0001
13	0b0000000_0	—	—	0	—	—	$a_{12}*Y_{-12}$	$a_{22}*Y_{-22}$	$Y_{01}+b_{22}*X_{-22}+b_{12}*X_{-12}$	0b0001_0001
14	0b0000000_1	X_{-20}	—	b_{20}	—	—	—	$a_{12}*Y_{-12}$	$c_{02}\ \Sigma X+a_{22}*Y_{-22}$	0b0000_0001
15	—	X_{-10}	X_{-20}	b_{10}	b_{20}	—	—	—	$Y_{02} = c_{04}\ \Sigma X+ \Sigma Y+\ldots \rightarrow$ next Y_{-22}	0b1100_0001

4.2 2x Configuration

$Y_{-mn} = X_{-m(n+1)}$ with m = 0 or 1 and n = 0, 1, … 3.
Address sssssbbh: h=1 is −2 history, sssss is SOS numbering, bb are lsb(00), isb(01), msb(10).
Control: v is write result value, h is history write, r is accumulator reset, l is accumulator load,
mm is 00-no shift/01-17b shift/1X-34b shift, and ii is 00-input(lsb)/01-input(isb)/10-input(msb)/11-history.

#	History		Input	Coefficients		Multiplier	Shifter	Accumulator	Control
	Addr	Mem	Mux	Mem	Reg				vhrl_mmii
0	—	—	—	g	—	—	—	—	0b0000_0000
1	—	—	X_{00},l	g	g	—	—	—	0b0000_0001
2	0b00000_001	—	X_{00},i	g	g	$g*X_{00}$,l	—	—	0b0010_0010
3	0b00000_011	X_{-20},l	X_{00},m	b_{20}	g	$g*X_{00}$,i	$g*X_{00}$,l	0	0b0000_0111
4	0b00000_101	X_{-20},i	X_{-20},l	b_{20}	b_{20}	$g*X_{00}$,m	$g*X_{00}$,i	$g*X_{00}$,l	0b0000_1011
5	0b00000_000	X_{-20},m	X_{-20},i	b_{20}	b_{20}	$b_{20}*X_{-20}$,l	$g*X_{00}$,m	$g*X_{00}$,l+ $g*X_{00}$,i	0b0000_0011
6	0b00000_010	X_{-10},l	X_{-20},m	b_{10}	b_{20}	$b_{20}*X_{-20}$,i	$b_{20}*X_{-20}$,l	$g*X_{00}$ $\rightarrow$ next X_{-20}	0b0100_0111
7	0b00000_100	X_{-10},i	X_{-10},l	b_{10}	b_{10}	$b_{20}*X_{-20}$,m	$b_{20}*X_{-20}$,i	…	0b0000_1011
8	0b00001_001	X_{-10},m	X_{-10},i	b_{10}	b_{10}	$b_{10}*X_{-10}$,l	$b_{20}*X_{-20}$,m		0b0000_0011
9	0b00001_011	Y_{-20},l	X_{-10},m	a_{20}/c_{00}	b_{10}	$b_{10}*X_{-10}$,i	$b_{10}*X_{-10}$,l	$g*X_{00}+b_{20}*X_{-20}$	0b0000_0111
10	0b00001_101	Y_{-20},i	Y_{-20},l	a_{20}/c_{00}	a_{20}/c_{00}	$b_{10}*X_{-10}$,m	$b_{10}*X_{-10}$,i	…	0b0000_1011
11	0b00001_000	Y_{-20},m	Y_{-20},i	a_{20}/c_{00}	a_{20}/c_{00}	$a_{20}*Y_{-20}$,l	$b_{10}*X_{-10}$,m		0b0000_0011
12	0b00001_010	Y_{-10},l	Y_{-20},m	a_{10}	a_{20}/c_{00}	$a_{20}*Y_{-20}$,i	$a_{20}*Y_{-20}$,l	$g*X_{00}+b_{20}*X_{-20}+b_{10}*X_{-10}$	0b0001_0111
13	0b00001_100	Y_{-10},i	Y_{-10},l	a_{10}	a_{10}	$a_{20}*Y_{-20}$,m	$a_{20}*Y_{-20}$,i	c_{00} ΣX+ $a_{20}*Y_{-20}$,l	0b0000_1011
14	0b00001_001	Y_{-10},m	Y_{-10},i	a_{10}	a_{10}	$a_{10}*Y_{-10}$,l	$a_{20}*Y_{-20}$,m	…	0b0000_0011
15	0b00001_011	X_{-21},l	Y_{-10},m	b_{21}	a_{10}	$a_{10}*Y_{-10}$,i	$a_{10}*Y_{-10}$,l		0b0000_0111
16	0b00001_101	X_{-21},i	X_{-21},l	b_{21}	b_{21}	$a_{10}*Y_{-10}$,m	$a_{10}*Y_{-10}$,i		0b0000_1011
17	0b00001_000	X_{-21},m	X_{-21},i	b_{21}	b_{21}	$b_{21}*X_{-21}$,l	$a_{10}*Y_{-10}$,m		0b0000_0011
18	0b00001_010	X_{-11},l	X_{-21},m	b_{11}	b_{21}	$b_{21}*X_{-21}$,i	$b_{21}*X_{-21}$,l	$Y_{00} = X_{01} = c_{00}\ \Sigma X+ \Sigma Y$ $\rightarrow$ next X_{-21}	0b0100_0111
19	0b00001_100	X_{-11},i	X_{-11},l	b_{11}	b_{11}	$b_{21}*X_{-21}$,m	$b_{21}*X_{-21}$,i	…	0b0000_1011
20	0b00010_001	X_{-11},m	X_{-11},i	b_{11}	b_{11}	$b_{11}*X_{-11}$,l	$b_{21}*X_{-21}$,m		0b0000_0011
21	0b00010_011	Y_{-21},l	X_{-11},m	a_{21}/c_{01}	b_{11}	$b_{11}*X_{-11}$,i	$b_{11}*X_{-11}$,l		0b0000_0111
22	0b00010_101	Y_{-21},i	Y_{-21},l	a_{21}/c_{01}	a_{21}/c_{01}	$b_{11}*X_{-11}$,m	$b_{11}*X_{-11}$,i		0b0000_1011
…									
49	0b00100_100	Y_{-13},i	Y_{-13},l	a_{13}	a_{13}	$a_{23}*Y_{-23}$,m	$a_{23}*Y_{-23}$,i		0b0000_1011
50	—	Y_{-13},m	Y_{-13},i	a_{13}	a_{13}	$a_{13}*Y_{-13}$,l	$a_{23}*Y_{-23}$,m		0b0000_0011
51	—	—	Y_{-13},m	—	a_{13}	$a_{13}*Y_{-13}$,i	$a_{13}*Y_{-13}$,l		0b0000_0111
52	—	—	—	—	—	$a_{13}*Y_{-13}$,m	$a_{13}*Y_{-13}$,i		0b0000_1011
53	—	—	—	—	—	—	$a_{13}*Y_{-13}$,m		0b0000_0011
54	0b00100_010	—	—	—	—	—	—	$Y_{03} = c_{03}\ \Sigma X+ \Sigma Y+$… $\rightarrow$ next Y_{-23}	0b1100_0000
59	—	—	—	—	—	—	—		0b0000_0000
…									
62	—	—	—	—	—	—	—		0b0000_0000
63	—	—	—	—	—	—	—		0b0000_0000

4.3 5x Configuration with SOS Switches Enabled

The switches are off for the first SOS and third SOS:

- The overall gain will be replaced by 0, if the gain switch is set to off.
- For the first SOS an alternate gain, g_1, has to be supplied instead of a_{20}. This value must be the gain in the first SOS minus 1, times the overall gain.
- For the third SOS an alternate gain, g_2, has to be supplied instead of a_{12}. This value must or the gain in the third SOS minus 1. This is the same for any other SOS.

#	History		Input	Coefficients		Multiplier	Sum1	Sum2	Accumulator	Control
	Addr	Mem	Mux	Mem	Reg					vhrl_bm0i
-3	0b0000000_0	—	—	S_1	—	—	$a_{12}*Y_{-12}$	$a_{22}*Y_{-22}$	$c_{02}\ \Sigma X$	0b0000_0001
-2	0b0000000_1	X_{-20}	—	$b_{20} = 0$	—	—	—	$a_{12}*Y_{-12}$	$c_{02}\ \Sigma X+a_{22}*Y_{-22}$	0b0000_0001
-1	0b0000000_0	X_{-10}	X_{-20}	$b_{10} = 0$	0	—	—	—	$Y_{02} = c_{04}\ \Sigma X+\Sigma Y+\ldots \rightarrow$ next Y_{-22}	0b1100_0001
0	0b0000001_1	—	X_{-10}	g	0	$0*X_{-20}$	—	—	—	0b0000_0000
1	0b0000001_0	Y_{-20}	X_{00}	g_1	g	$0*X_{-10}$	0	—	—	0b0010_0000
2	0b0000001_1	Y_{-10}	X_{00}	$a_{10}= 0$	g_1	$g*X_{00}$	0	0	0	0b0000_0001
3	0b0000001_0	X_{-21}	Y_{-10}	b_{21}	0	g_1*X_{00}	$g*X_{00}$	0	0	0b0000_0001
4	0b0000010_1	X_{-11}	X_{-21}	b_{11}	b_{21}	$0*Y_{-10}$	g_1*X_{00}	$g*X_{00}$	$0\ /\ g*X_{00} \rightarrow$ next X_{-20}	0b0100_0101
5	0b0000010_0	Y_{-21}	X_{-11}	a_{21}/c_{01}	b_{11}	$b_{21}*X_{-21}$	0	g_1*X_{00}	$g*X_{00}$	0b0001_0001
6	0b0000010_1	Y_{-11}	Y_{-21}	a_{11}	a_{21}/c_{01}	$b_{11}*X_{-11}$	$b_{21}*X_{-21}$	0	$g*X_{00}+g_1*X_{00}$	0b0000_0001
7	0b0000010_0	X_{-22}	Y_{-11}	$b_{22} = 0$	a_{11}	$a_{21}*Y_{-21}$	$b_{11}*X_{-11}$	$b_{21}*X_{-21}$	$Y_{00} = X_{01} = (g+g_1)*X_{00} \rightarrow$ next X_{-21}	0b0100_0001
8	0b0000011_1	X_{-12}	X_{-22}	$b_{12} = 0$	0	$a_{11}*Y_{-11}$	$a_{21}*Y_{-21}$	$b_{11}*X_{-11}$	$Y_{00}+b_{21}*X_{-21}$	0b0000_0001
9	0b0000011_0	Y_{-22}	X_{-12}	a_{22}/c_{02}	0	$0*X_{-22}$	$a_{11}*Y_{-11}$	$a_{21}*Y_{-21}$	$Y_{00}+b_{21}*X_{-21}+b_{10}*X_{-10}$	0b0001_0001
10	0b0000011_1	Y_{-12}	Y_{-22}	g_2	0	$0*X_{-12}$	0	$a_{11}*Y_{-11}$	$c_{01}\ \Sigma X+a_{21}*Y_{-21}$	0b0000_0001
11	0b0000011_0		Y_{-12}	S_0	g_2	$0*Y_{-22}$	0	0	$Y_{01} = c_{01}\ \Sigma X+\Sigma Y+\ldots \rightarrow$ next Y_{-21}	0b0100_1001
12	—	—	—	S_1	—	g_2*Y_{01}	0	0	Y_{01}	0b0000_0001
13	0b0000000_0	—	—	0	—	—	g_2*Y_{01}	0	Y_{01}	0b0001_0001
14	0b0000000_1	X_{-20}	—	b_{20}	—	—	—	g_2*Y_{01}	Y_{01}	0b0000_0001
15	—	X_{-10}	X_{-20}	b_{10}	b_{20}	—	—	—	$Y_{02} = (1+g_2)Y_{01} \rightarrow$ next Y_{-22}	0b1100_0001