Hindawi Publishing Corporation
International Journal of Reconfigurable Computing
Volume 2014, Article ID 546264, 5 pages
http://dx.doi.org/10.1155/2014/546264

Research Article

Efficient FPGA Hardware Reuse in a Multiplierless Decimation Chain

Guillermo A. Jaquenod,[1] Javier Valls,[2] and Javier Siman[3]

[1] *Facultad de Ingeniería, Universidad Nacional del Centro, 7400 Olavarría, Argentina*
[2] *Instituto de Telecomunicaciones y Aplicaciones Multimedia, Universidad Politécnica de Valencia, 46730 Gandia, Spain*
[3] *DTA S.A., 5000 Córdoba, Argentina*

Correspondence should be addressed to Guillermo A. Jaquenod; jaquenodg@gmail.com

Received 9 September 2013; Revised 31 October 2013; Accepted 13 November 2013; Published 5 January 2014

Academic Editor: Michael Hübner

In digital communications, an usual reception chain requires many stages of digital signal processing for filtering and sample rate reduction. For satellite on board applications, this need is hardly constrained by the very limited hardware resources available in space qualified FPGAs. This short paper focuses on the implementation of a dual chain of 14 stages of cascaded half band filters plus 2 : 1 decimators for complex signals (in-phase and quadrature) with minimal hardware resources, using a small portion of an UT6325 Aeroflex FPGA, as a part of a receiver designed for a low data rate command and telemetry channel.

1. Introduction

In digital receivers used in satellite applications [1], after conversion to an intermediate frequency (IF) and filtering, a common approach is to shift the signal to baseband [2]. Since the IF spectrum is not symmetrical, the translation to baseband creates a complex signal [3] composed of in-phase (I) and quadrature (Q) components; this complex signal is then band-limited by successive steps of filters and decimators before data detection. Many alternatives exist for filters and decimators, like cascaded integrator comb (CIC or Hogenauer) filters [4–6], cyclotomic polynomial filters [7–9], or polyphase decimators [10, 11].

These algorithms are difficult to be implemented using space qualified FPGAs. These devices are designed to support high levels of ionizing radiation, composed by alpha particles, protons, and heavy ions, so they use technologies three or more times bigger in size details than current commercial FPGA devices, and main datasheet parameters are total accumulated radiation (TID) dose in krads and tolerance to single event upsets (SEU) and single event latchup (SEL) in MeV·cm^2/mg; packages are also limited in size and weight to tolerate high accelerations and wide temperature operating ranges.

As a consequence of these constraints, space-qualified FPGAs are much slower and many times smaller in number of logical cells than commercial devices, and implementation efforts are directed to low complexity solutions [11, 12], trying to avoid multipliers or complex arithmetic operations.

The design described in this paper is focused on the implementation of a block of two chains of 14 filters and decimators used for the reception of the low data rate (4 kbps) spaceflight tracking and data network command channel [13], transmitted from ground to a low earth orbit satellite (LEO), and the paper details the hardware improvements used for efficient filtering processes.

2. The Communications Link

A LEO satellite with a circular orbit at 700 km over the ground moves at high speed in reference to the earth station completing one orbit in 100 minutes. Each time it gets over the horizon, during the 6 minutes of line of sight time, it establishes a communication link with the ground station. The RF signal transmitted from earth is created by linear phase modulation of a 2.025 to 2.110 GHz carrier with a 16 kHz subcarrier, which is in turn BPSK modulated by

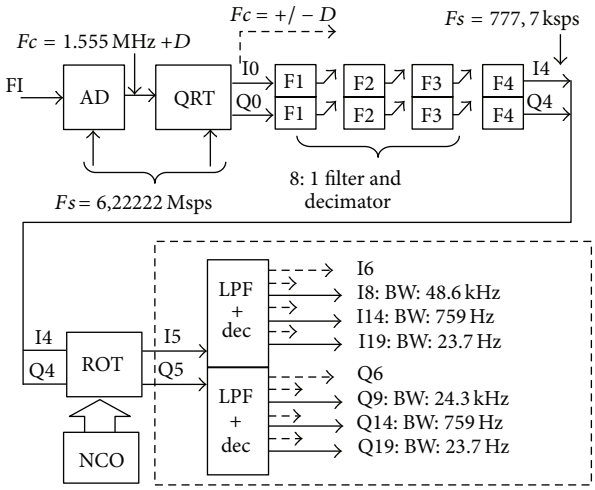

Figure 1: The complete processing chain.

the 4 kbps data signal. The resulting spectrum has a residual carrier (useful for tracking) with two main lobes at ±16 kHz (at −2 dBc) and two secondary lobes at ±32 kHz (at −13 dBc) and suffers a carrier Doppler drift of ±46.5 kHz and 11 dB of signal level variation, which must be tracked and compensated.

3. Building Blocks of Receiver

In the satellite receiver, the incoming RF signal is first filtered, amplified, and down-converted to a 70 MHz IF. Due to oscillators mismatch, thermal drifts, and Doppler, the carrier can be located with an offset of $D = \pm 140$ kHz from the central IF; therefore, the signal is filtered by a 500 kHz bandwidth SAW filter, much wider than the signal spectrum.

As shown in Figure 1, this IF signal is undersampled at $Fs = 6.222$ Msps (Fs is obtained by division of the FPGA 56 MHz oscillator XO, as $Fs = XO/9$). Fs is related to the IF by a factor 11.25, so Fc (the lowest replica of the IF signal) becomes located at $Fs/4 = 1.555$ MHz with offset D. Using a quarter rate translator (QRT) [14], this replica is translated to baseband, giving a complex signal (I0, Q0), yet shifted from 0 Hz by D.

At this point the (I0, Q0) signals are decimated 8 : 1 to filter "out of band" noise and reduce sample rate. This decimation is performed by cascading 4 half band filters (HBFs) for each I/Q component (HBF is a lowpass filter with cut frequency in $Fs/4$) with 2 : 1 decimators inserted in between; after F4 the sampling rate is 777.7 ksps, and noise over 194 kHz has been filtered. The signal is still affected by the D offset, so a CORDIC rotator (ROT) [15] driven by a numerically controlled oscillator (NCO) is used to shift it towards 0 Hz.

The rotator outputs I5/Q5 are fed to 14 stages of HBFs and decimators (referred as HBF6 to HBF19), with each filter running at half the rate of the previous one. The use of HBFs instead of CIC, polyphase, or other high-order filters is due to two facts:

(a) the need to have access to intermediate decimated results. Different stages of the filter chain are used for different purposes as follows:

(i) I8 (BW: 48.6 kHz), to recover 2x subcarrier frequency;

(ii) Q9 (BW: 24.3 kHz), is used to recover BPSK modulated subcarrier, and the modulation data;

(iii) I14/Q14 (BW: 759 Hz), are used by the frequency control loop, which in turn controls the NCO. This tap is selected because the frequency error after the sweep process is bounded to ±500 Hz;

(iv) I19/Q19 (BW: 23.7 Hz), are used to control the phase error, adjusting the NCO to track the Doppler drift.

(b) A common hardware structure simplifies hardware reuse.

This point is the innovation described in this paper: the implementation of the complete chain of 14 cascaded complex filters and decimators HBF6 to HBF19 (the circuit area bounded by a dashed rectangle in Figure 1) as a very compact common block, using only one dual port RAM, one simple arithmetic block, one 15 bit-counter, and one control state machine.

4. The Selected Half Band Filter

When a signal sampled at Fx rate is decimated 2 : 1, the spectrum at $Fx/2$ is folded over the origin. If the signals of interest (SOI) are located between 0 Hz and F_{SOI}, a filter with high attenuation between $(Fx/2 - F_{SOI})$ and $(Fx/2)$ must be applied before decimation. The architecture of the filter used, described in [16], is shown in Figure 2.

The transfer function of this filter in Z domain is

$$H(z) = \frac{0.5 \times \left(A + z^{-1} + z^{-2} + A \times z^{-3}\right)}{1 + A \times z^{-2}}. \quad (1)$$

The Aeroflex UT6325 FPGA [17] has no multipliers, and the two products that seem to be needed (the A gain and the 0.5x multiplier) are replaced by shift and add operations. The filter behavior was evaluated for simple A values: for F1 to F3 and HBF6 to HBF19 the chosen gain A is $3/8 = 0.375 = 1/2 - 1/8$, obtained by two fixed shifts and one subtraction; for F4 two cascaded HBFs (F4a/F4b) with gains $A = 0.5$ and $A = 0.375$ are used. The response for $A = 0.375$ (thin line) and the combined effect of F4a/F4b (thick line) are shown in Figure 3.

For gain 0.375 the filter provides a convenient behavior:

(i) flat response within −0.1 dB from the origin to $0.15Fx$;

(ii) attenuation greater than −39 dB from $0.4Fx$ to $Fx/2$.

Besides it good filtering performance, the main reason to select this filter is that it can be easily synthesized as a sequential machine, using two temporary registers T and IX, and five RAM positions to store the input/output value FIO, and the four delay-line values B, C, D, and E.

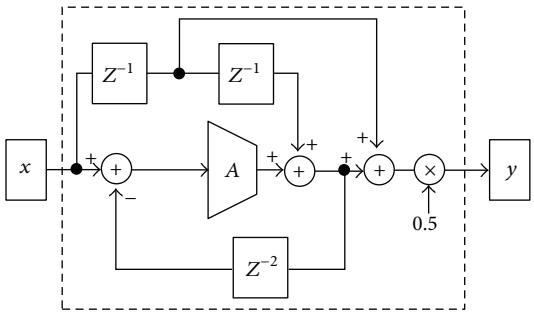

Figure 2: Half Band filter architecture.

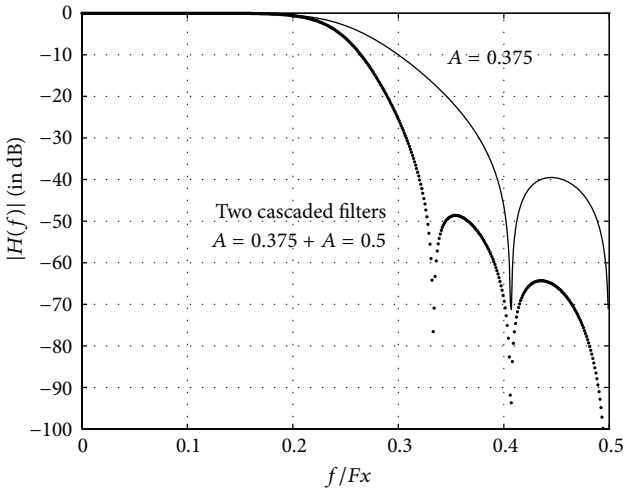

Figure 3: Half Band filter transfer function.

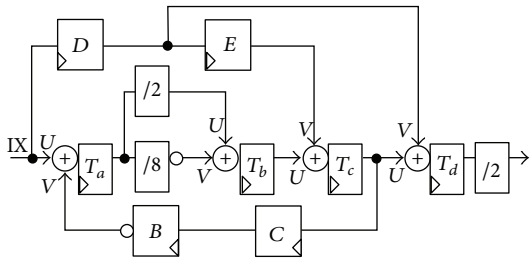

Figure 4: Half band filter data flow.

Figure 4 shows a flow diagram of the filter, and the subscript letter on the T-register indicates the storage function fulfilled by T in successive steps of the sequential evaluation. The complete filter can be evaluated in 12 XO clock cycles, reserving two clock cycles for every addition, which leaves enough time for ripple carry propagation in the adder circuit.

5. Time Requirements to Implement Each Decimated Filter

In Section 3, it was written that each filter from HBF6 to HBF19 runs at half the rate of the previous one, and although each filter must be computed at its own rate to upgrade the delay-line values B, C, D, and E, only one half of its output values is used by the next filter, due to decimation. Since samples at I5/Q5 come at $Fs/8$ = XO/72 rate, there are 72 XO clock cycles available to compute the whole chain.

If N cycles are needed for HBF6, the summation of time requirements for the dual chain of filters is $2 * N * (1 + 1/2 + 1/4 + 1/8 + 1/16 + \cdots)$ which *is always less than* $4 * N$ (this is the key feature), so if the available 72 cycles are divided by 4, it results that $N = 18$ cycles to solve each filter, a 50% margin over the time needed for the sequential solution as shown in Section 4.

In the common multiplexed circuit designed to evaluate the $2 \times 14 = 28$ filters, a 15-bit counter updated at XO/N rate is used to decide when to compute each filter. Calling "s" the LSB of the counter and using "x" to denote a "do not care" value, the time schedule is a function of the counter value:

xxxxxxxxxxxxx1s = HBF6 stage: I(s = 0) and Q(s = 1)
xxxxxxxxxxxx10s = HBF7 stage: I(s = 0) and Q(s = 1)
xxxxxxxxxxx100s = HBF8 stage: I(s = 0) and Q(s = 1)
xxxxxxxxxx1000s = HBF9 stage: I(s = 0) and Q(s = 1)
xxxxxxxxx10000s = HBF10 stage: I(s = 0) and Q(s = 1)
xxxxxxxx100000s = HBF11 stage: I(s=0) and Q(s = 1)
xxxxxxx1000000s = HBF12 stage: I(s=0) and Q(s = 1)
xxxxxx10000000s = HBF13 stage: I(s=0) and Q(s = 1)
xxxxx100000000s = HBF14 stage: I(s=0) and Q(s = 1)
xxxx1000000000s = HBF15 stage: I(s=0) and Q(s = 1)
xxx10000000000s = HBF16 stage: I(s=0) and Q(s = 1)
xx100000000000s = HBF17 stage: I(s=0) and Q(s = 1)
x1000000000000s = HBF18 stage: I(s=0) and Q(s = 1)
10000000000000s = HBF19 stage: I(s=0) and Q(s = 1).

Table 1 shows how the different stages are computed as the counter increases; the grey shaded are those whose values are used in the next stage, and those not shaded are computed, but their output is ignored (decimated). It must be noted that the value 00000000000000s is unused, giving place to add more filters and decimators, if desired.

6. Hardware Proposal

The hardware used for solving the 28 filters is shown in Figure 5 and is composed of the following blocks.

Arithmetic Block (AB). Together with the RAM, it includes the IX- and T-registers, one parallel adder, and some multiplexers. Although the inputs I5/Q5 to filter HBF6 are 14-bit wide (2 integer + 12 fractional), the arithmetic operations are solved with 17-bit precision, adding 3 guard bits to consider the filter gain and to avoid overflow problems. The block checks these 3 upper bits and saturates the output when it is over 12 bits.

Sequential Control Machine (SCM). A simple circuit, based on a 5-bit counter, generates the RAM address for reading and writing within each page, the write strobe for the RAM,

Table 1: Time slot allocation for the first 4 stages.

HBF6		I	Q			I	Q			I	Q			I	Q			I	Q			I	Q			I	Q			I	Q
HBF7				I	Q								I	Q							I	Q							I	Q	
HBF8								I	Q																	I	Q				
HBF9																				I	Q										

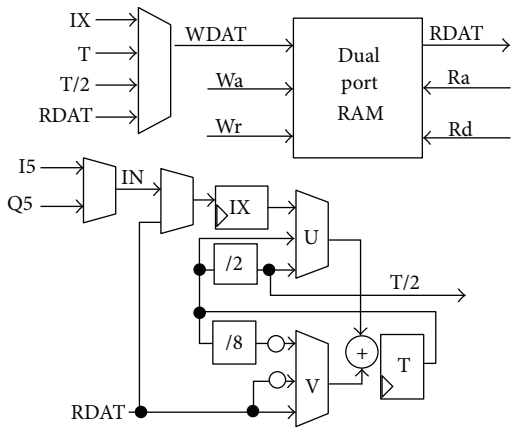

Figure 5: Hardware resources for the complete filter.

the enable for the registers, and the selection signals for the multiplexers. It is synchronized by a pulse coming from the rotator block.

RAM. RAM is a 256 × 16 dual port RAM built using two UT6325 RAM blocks, with asynchronous read.

Slot Time Counter (STC). STC a 15-bit binary counter updated at XO/18 rate, as described in Section 5.

First One Position (FOP). It detects the position of the first "1" from right to left in the upper 14 bits of STC. Together with the LSB of the STC, the FOP output is used to address 28 pages of RAM, one for each filter; eight RAM positions are assigned to each page, although only five are used. Combined with the SCM it generates the strobes I8, Q9, I14/Q14, and I19/Q19 to identify when the filter computation is ready for different filters. FOP and STC have 18 XO cycles for update, so both circuits are implemented with serial arithmetic to minimize the use of hardware resources.

7. Compilation Results

The benefits of this design are evident when compilation reports are evaluated. The hardware proposed, compiled for an UT6325PQ208-5 Aeroflex FPGA, consumes 2 of the 24 RAM cells (8.3% of the FPGA RAM available), only 174 logic cells (11.3% of the scarce 1536 FPGA cells), 99 registers (3.2% of the FPGA flipflops) and runs at 65 MHz; if the saturation block is omitted, the report is 159 cells (10.4%) and 83 registers (2.7%).

8. Comparison with Other Solutions

On [18], it is described a similar approach for a 3-tap filter, where different filters are multiplexed in time, but each filter is realized using distributed logic: 14 registers, 4 multipliers, 4 adders, and 11 multiplexers; this solution requires the redesign and growth of all multiplexers' size as the number of taps increases. No reports are given about effective implementation of [18] using an ASIC or FPGA. Instead, for the smaller regular structure shown in Figure 5 (5 MUX, 2 registers, one adder, no multiplier, and 16 bytes of RAM memory), the addition of a new decimation step only requires 16 new bytes of RAM, one new bit on STC, and—sometimes—another bit on FOP. In fact, since each filter runs at one half the rate of the previous one, this same hardware can be used for a chain of half band filters plus 2 : 1 decimators as large as desired, provided that enough RAM is available.

The solution showed on [19] for the filter is similar to the implementation proposed on this paper in what [19] calls combinatorial-sequential implementation; its synthesis results report a higher number of registers (246 instead of 99), for a single filter. However, since no filters are multiplexed on time, it cannot be compared in relation to RAM usage or control and scheduling complexity.

Automated solutions, like those obtained with DSP Builder [20] and Matlab/Silulink tools only look for performance and speed, and they make no efforts on hardware multiplexing and processing dead time usage as the decimation chain reduces the sampling rate. The number of registers, RAM and multipliers used by these solutions is huge, orders of magnitude higher than the solution proposed in this paper.

The main emphasis of this paper is on hardware reuse: at a comfortable speed, a complete chain of 28 second-order filters and decimators is implemented using only one parallel adder for the arithmetic operations, no multipliers, and a common RAM to store the state of the full set of filters; if the total hardware is divided by the 28 filters, FPGA requirements are less than 7 combinatorial cells and a mean of 3.5 flipflops to compute each second-order filter with 16 bits of precision.

9. Conclusion

Most DSP application reports put emphasis on new algorithms or levels of parallelism, with abstraction of real hardware requirements; it is a reasonable approach since modern commercial devices seem to have unlimited resources. However, this constraint is mandatory for space applications.

The key improvement of this solution is described in Section 5: instead of wasting processing speed capabilities running hardware for slower filters at slow speeds, this hardware is always running at the full speed to support all filters,

and time slots assigned by the scheduler to slower filters are more distanced than those assigned to faster ones.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] F. Daneshgaran and M. Laddomada, "Transceiver front-end technology for software radio implementation of wideband satellite communication systems," *Wireless Personal Communications*, vol. 24, no. 2, pp. 99–121, 2003.

[2] A. A. Abidi, "The path to the software-defined radio receiver," *IEEE Journal of Solid-State Circuits*, vol. 42, no. 5, pp. 954–966, 2007.

[3] V. Considine, "Digital complex sampling," *Electronics Letters*, vol. 19, no. 16, pp. 608–609, 1983.

[4] H.-K. Yang and W. M. Snelgrove, "High speed polyphase CIC decimation filters," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS '96)*, pp. 229–232, Atlanta, Ga, USA, May 1996.

[5] R. A. Losada and R. Lyons, "Reducing CIC filter complexity," *IEEE Signal Processing Magazine*, vol. 23, no. 4, pp. 124–126, 2006.

[6] F. J. A. de Aquino, C. A. F. da Rocha, and L. S. Resende, "Design of CIC filters for software radio system," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '06)*, Toulouse, France, May 2006.

[7] M. Laddomada, "Design of multistage decimation filters using cyclotomic polynomials: optimization and design issues," *IEEE Transactions on Circuits and Systems I*, vol. 55, no. 7, pp. 1977–1987, 2008.

[8] R. J. Hartnett and G. F. Boudreaux-Bartels, "On the use of cyclotomic polynomial prefilters for efficient FIR filter design," *IEEE Transactions on Signal Processing*, vol. 41, no. 5, pp. 1766–1779, 1993.

[9] K. Supramaniam and Y. Lian, "Complexity reduction for frequency-response masking filters using cyclotomic polynomial prefilters," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS '06)*, pp. 3297–3300, Island of Kos, Greece, May 2006.

[10] J. Rothweiler, "Polyphase quadrature filters-a new subband coding technique," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '83)*, vol. 8, pp. 1280–1283, Boston, Mass, USA, 1983.

[11] M. W. Coffey, "Optimizing multistage decimation and interpolation processing," *IEEE Signal Processing Letters*, vol. 10, no. 4, pp. 107–110, 2003.

[12] M. W. Coffey, "Optimizing multistage decimation and interpolation processing—part II," *IEEE Signal Processing Letters*, vol. 14, no. 1, pp. 24–26, 2007.

[13] NASA, "STDN user's manual," Goddard Space Flight Center, (NASA-TM-X-72932) STDN User's Manual N75-78163, 124 pages, UNCLAS 00/98 03939, 1998.

[14] R. E. Reedy and M. C. Comparini, "Perspective of RF CMOS/mixed signal integration in next generation satellite systems," in *Proceedings of the 11th Gallium Arsenide Applications Symposium (GAAS '03)*, Munich, Germany, October 2003.

[15] http://www.andraka.com/files/crdcsrvy.pdf.

[16] B. W. Maples and K. A. Fix, "An IF sampling digital receiver implementation for space-based command and telemetry applications," CMC Electronics Cincinnati, Mason, Ohio, USA.

[17] http://aeroflex.com/ams/pagesproduct/datasheets/RadTolEclipseFPGA.pdf.

[18] T. C. Denk and K. K. Parhi, "Synthesis of folded pipelined architectures for multirate DSP algorithms," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 6, no. 4, pp. 595–607, 1998.

[19] N. Sedaghati-Mokhtari, S. Rahmanian, and S. M. Fakhraie, "Hardware implementation analysis for digital filters," in *Proceedings of the 14th Iranian Conference on Electrical Engineering (ICEE '06)*, Tehran, Iran, May 2006.

[20] http://www.altera.com/literature/ug/ug_fir_compiler_ii.pdf.