Implementation approaches for 512-tap 60 GSa/s chromatic dispersion FIR filters

Anton Kovalev, Oscar Gustafsson and Mario Garrido

Implementation Approaches for 512-tap 60 GSa/s Chromatic Dispersion FIR Filters

Anton Kovalev, Oscar Gustafsson, and Mario Garrido
Department of Electrical Engineering, Linköping University,
SE-581 83 Linköping, Sweden. E-mail: oscar.gustafsson@liu.se

***Abstract*—In optical communication the non-ideal properties of the fibers lead to pulse widening from chromatic dispersion. One way to compensate for this is through digital signal processing. In this work, two architectures for compensation are compared. Both are designed for 60 GSa/s and 512 filter taps and implemented in the frequency domain using FFTs. It is shown that the high-speed requirements introduce constraints on possible architectural choices. Furthermore, the theoretical multiplication complexity estimates are not good predictors for the energy consumption. The results show that the implementation with 10% more multiplications per sample has half the power consumption and one third of the area consumption. The best architecture for this specification results in a power consumption of 3.12 W in a 65 nm technology, corresponding to an energy per complex filter tap of 0.10 mW/GHz.**

I. Introduction

Digital correction of physical fiber impairments has been an active topic for more than a decade [1]–[3]. In this work the focus is on chromatic dispersion. This can be modeled as

$$C\left(e^{j\omega T}\right) = e^{-jk(\omega T)^2}, \tag{1}$$

with

$$k = \frac{D\lambda^2 z}{4\pi c T^2}, \tag{2}$$

where D is the fiber dispersion parameter, λ is the wavelength, z is the propagation distance, c is the speed of light, and T is the sample period.

Chromatic dispersion leads to pulse widening, which for long fibers and/or high sample rates can spread over hundreds of adjacent pulses. To compensate for the chromatic dispersion a filter with transfer function

$$H\left(e^{j\omega T}\right) = \frac{1}{C\left(e^{j\omega T}\right)} = e^{jk(\omega T)^2} \tag{3}$$

is used. An estimate of the required number of filter taps is [4]

$$M \approx \lceil 2k\pi \rceil = \left\lceil \frac{D\lambda^2 z}{2cT^2} \right\rceil. \tag{4}$$

This means that the required filter order increases linearly with the distance and quadratically with the sample rate. With high sample rates and potentially long fibers, implementing these filters is a challenge.

Several methods to design chromatic dispersion compensation filters have been proposed [4]–[6], including optimal ones [5]. However, there are quite few optimized implementations able to handle the required sample rates, often tenths of GSa/s. Some previous works consider theoretical complexity computations [7], [8], but as will be seen in this work, the architecture mapping must be considered as well.

A few implementations have been reported for FPGAs [9], [10]. For example, in [9] a 128-tap filter operating at 20 GSa/s is implemented in a single FPGA by efficiently using the different FPGA resources. For ASIC implementation, Fougstedt et al [11] show some trade-offs when implementing the filters using fast FIR algorithms. The focus in [11] is on 56 GSa/s and filter lengths up to 64. In both [9] and [11], there is an inherent limitation in the fact that the filter length is the same as the number of samples processed per iteration. This is a natural limitation, as will be seen later, but will also constrain the maximum fiber length.

In this work the main focus is on the implementation of high-speed complex FIR filters where the filter length is longer than the number of samples processed in parallel. This allows longer fibers to be compensated, and/or, as the filter length increases quadractically with the sample rate, higher sample rates to be used compared to the earlier approaches. We discuss the impact of the architecture on the implementation and compare two different implementations.

In the next section, FFT-based FIR filtering is reviewed. Then, in Section III, the impact of high-speed implementation is discussed. In Section IV, implementation results are presented, before some concluding remarks are given.

II. FFT-based FIR Filtering

A. FFT+Mult Algorithm

It is since long established that FIR filtering can be performed in the frequency-domain and that the arithmetic complexity is reduced compared to time-domain implementation for long enough filters. An impulse response of length M can be convolved with a signal of length K using a DFT of length $N \geq M + K - 1$. The impulse response sequence and the input signal are both zero-padded to length N and transformed with the DFT. Then, the outputs are point-wise multiplied and the result inverse transformed with the IDFT. The result then forms the convolution of the two sequences. This is illustrated in Fig. 1.

For a continuous signal, either overlap-add or overlap-save can be used. For overlap-add, the results of two subsequent output sequences are added, while for overlap-save, N input samples are used instead of zero-padding. For both schemes,

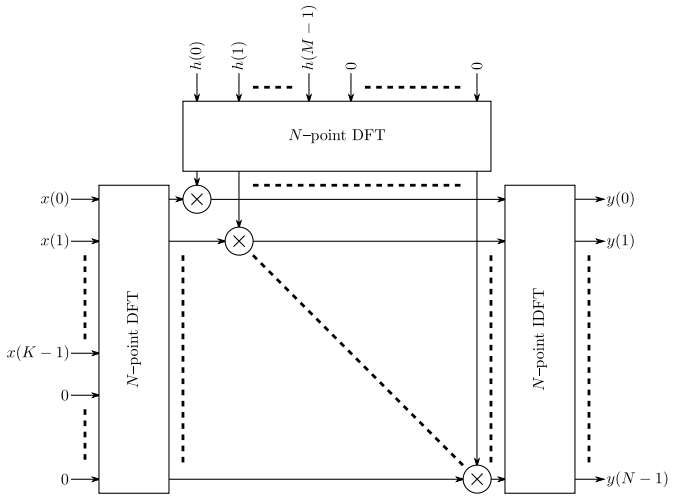

Fig. 1. DFT-based convolution.

K correct output samples are produced per DFT-IDFT iteration.

An approximation of the number complex multiplications for computing an N-point DFT using an FFT is $\frac{N}{2}\left(\log_2\left(N\right)-2\right)$, assuming that N is a power of two. The subtraction by two comes from the fact that there are only complex multiplications in between the stages and that one of the multiplication stages only consist of trivial multiplications, i.e., multiplications with 1 or $-j$. Hence, to compute K output samples using an impulse response of $M = N - K$ taps[1] an average of approximately

$$\frac{2 \times \frac{N}{2}\left(\log_2\left(N\right)-2\right)+N}{K} = \frac{N\left(\log_2\left(N\right)-1\right)}{N-M} \quad (5)$$

complex multiplications per sample are required. Here, it is assumed that the DFT of the impulse response is performed off-line and is not included in the expression. Based on (5), the optimal value of N can be determined. However, as seen later, there are more aspects to consider when implementing this using very high sample rates. This approach is referred to as FFT+Mult in the following.

B. FFT+FIR Algorithm

Another option is to separate the impulse response in polyphase components and perform the DFT on the polyphase components [12]. Typically, the number of samples processed in each iteration[2], L, is chosen to be the same as the polyphase factor, leading to that the DFT length is twice of that. Assuming that the impulse response length M is an integer multiple of L, this leads to N FIR filters of length $\frac{M}{L}$. The approximate average complexity to compute L samples with $N = 2L$ is

$$\frac{2 \times \frac{N}{2}\left(\log_2\left(N\right)-2\right)+N\frac{M}{L}}{L} = 2\log_2\left(L\right)-2+2\frac{M}{L} \quad (6)$$

[1]Here, we compute one sample less than the bound as this simplifies the architectures later.

[2]Figure 2 shows both the algorithm, where K should be used instead of L, and the architecture in the case of iso-morphic mapping.

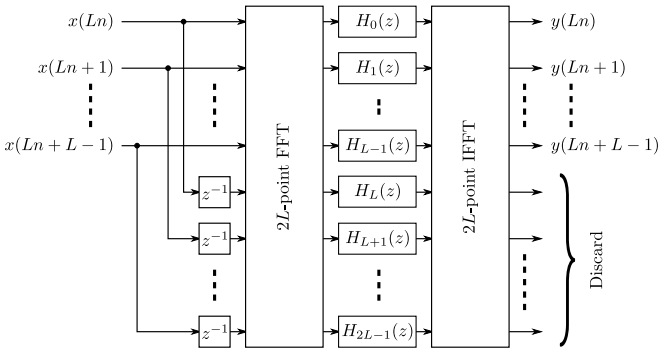

Fig. 2. FFT+FIR algorithm/architecture using polyphase decomposition and overlap-save with $N = 2L$ and $M = PL$ for integer P using fully parallel FFTs and IFFTs and P-tap FIR filters.

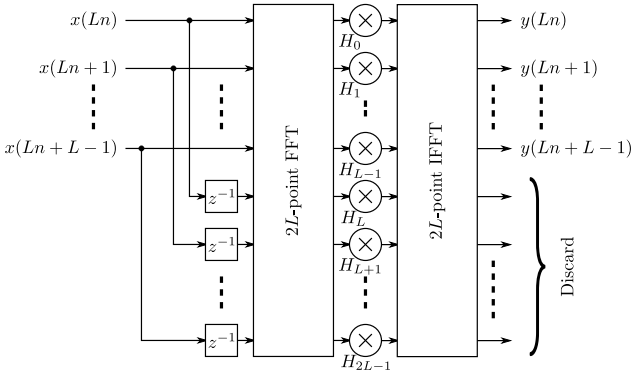

Fig. 3. Architecture for FFT-based filtering with $N = 2L$ and $L = K = M$ using fully parallel FFTs and IFFTs.

complex multiplications per sample. We will in the following refer to this implementation as FFT+FIR.

C. Complexity

The number of multiplications per sample for different N using the FFT+Mult approach and for $L = 128$ and the FFT+FIR approach[3], based on (5) and (6), respectively, is shown in Table I. In addition, results for direct time-domain implementation using polyphase FIR filters are also shown. It is clear that the frequency-domain implementations offer a multiplication complexity reduction of more than 25 times for these cases.

III. High-Speed Implementation Aspects

In previous work, the number of samples per iteration, L is selected to be equal to K and M [9], [11]. This leads to an iso-morphic mapping of the basic convolution in Fig. 1, which, with overlap-save processing, leads to the architecture in Fig. 3

[3]Strictly, K for the algorithm and L for the implementation must not be selected the same. However, to allow an iso-morphic mapping we select to do that here.

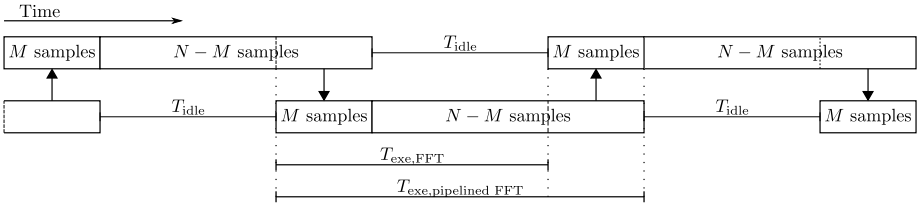

Fig. 4. Input scheduling of two pipelined FFTs with $N > 2M$.

TABLE I
MULTIPLICATIONS PER SAMPLE FOR THE DIFFERENT APPROACHES.

Approach	Mult. per sample
FFT+Mult, $N = 1024$	18.0
FFT+Mult, $N = 2048$	13.3
FFT+Mult, $N = 4096$	12.6
FFT+Mult, $N = 8192$	12.8
FFT+Mult, $N = 16384$	13.4
FFT+FIR, $L = 128$	20
FIR, $L = 128$	512

Consider the implementation of complex-valued FIR filters operating at about 60 GSa/s. Processing L samples per iteration, the clock frequency is

$$f_{\text{clk}} = \frac{f_{\text{sample}}}{L} = \frac{60 \times 10^9}{L}. \tag{7}$$

Ideally, the clock frequency should be selected based on the implementation technology and not based on the filter length, as in the case for the architecture in Fig. 3. In this work, we select $L = 128$, although $L = 64$, as in [11] may also be worth considering. This leads to a clock frequency of

$$f_{\text{clk}} = \frac{60 \times 10^9}{128} = 468.75 \text{ MHz}. \tag{8}$$

A. FFT+Mult Architecture

From Table I it can be seen that $N = 4096$ leads to the lowest complexity per sample. However, consider the operation of an FFT-based FIR filter processing 128 samples per clock cycle. We here consider overlap-save processing, so in each FFT M samples which were also processed in the previous FFT is processed again. Hence, every $T_{\text{exe,FFT}} = 4096 - 512 = 3584$ input sample, where $M = 512$ is the number of overlapping samples. This leads to that every $\frac{3584}{128} = 28$th clock cycle, a new FFT computation must be started. A pipelined FFT core such as the ones in [13] processing 128 samples per clock cycle requires $T_{\text{exe,pipelined FFT}} = \frac{4096}{128} = 32$ clock cycles to input all data. Then, in the next cycle, a new FFT can be started. This is illustrated in general in Fig. 4. For this case, two FFT cores are required to process the data. The resulting architecture is shown in Fig. 5. The FFT cores will idle for $T_{\text{idle}} = 24$ clock cycles before the next FFT computation on that core will start. The idle time will also cause additional control overhead, and, more importantly, will introduce a significant area overhead as the utilization will only be $\frac{T_{\text{exe,pipelined FFT}}}{T_{\text{exe,pipelined FFT}}+T_{\text{idle}}=24} \approx 57\%$. One way to solve this is to reduce the FFT length.

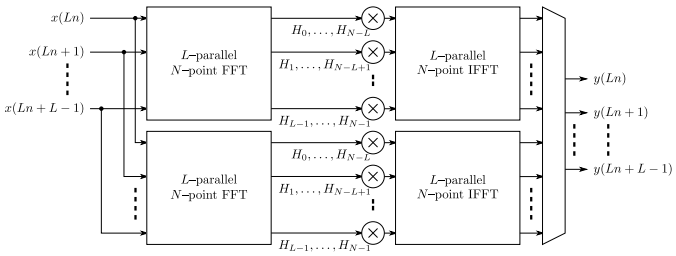

Fig. 5. FFT-Mult architecture for FFT-based filtering with $N = K + M$ and $K > L$, using two pipelined L-parallel FFTs and IFFTs. The processing for the branches starts at different times.

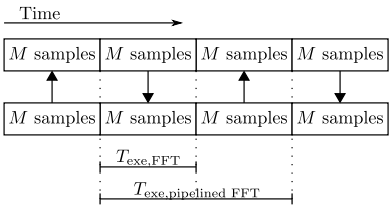

Fig. 6. Input scheduling of two pipelined FFTs with $N = 2M$.

For $N = 2048$, the same problem occurs. However, in this case, there are only $T_{\text{idle}} = 8$ clock cycles of idle time, leading to a utilization of $\frac{16}{24} \approx 67\%$. For $N = 1024$, the two FFT cores can work continuously and reach 100% utilization. Hence, this is the approach that we select. Note that a 1024-point FFT core will be physically smaller than a 2048-point or 4096-point. Hence, although the computational complexity is higher, the total chip area will be smaller as in all cases two FFT cores are required as well as two IFFT cores. For all these options, 128 multipliers are required for each FFT-IFFT pair, alternating between a number of different coefficients. For $N = 1024$, each multiplier has eight different coefficients. The number of multiplications per sample with $N = 2M$ is, based on (5), approximately

$$\frac{2M\left(\log_2\left(2M\right) - 1\right)}{2M - M} = 2\log_2\left(M\right). \tag{9}$$

B. FFT+FIR Architecture

For the FFT+FIR approach, one 256-point FFT and one 256-point IFFT are required. As a new computation is started each cycle, there is no need for two (or more) FFT cores. In between the FFT and the IFFT there are 256 4-tap FIR filters. Although the complexity is higher for this case, the

TABLE II
NUMBER OF REAL-VALUED BLOCKS WITH COMPLEXITY REDUCTION APPLIED AT THE MULTIPLIER OR FILTER LEVEL.

Structure	Multiplications	Add./sub.	Delays
Multiplier	12	15	6
Filter, Fig. 7	12	12	9

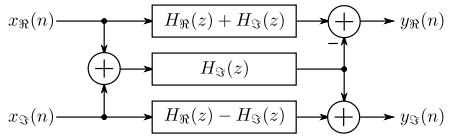

Fig. 7. Reduced complexity complex-valued FIR filter.

approach still has some potential benefits. First, the FFT and IFFTs are fully parallel, meaning that each twiddle factor multiplier is constant, and, hence, can be simplified. A final potential benefit is that of being able to reduce the number of multiplication similar to that of complex multiplication, but on the filter level. This is illustrated in Fig. 7 and the number of arithmetic operations and delay elements for a 4-tap FIR filter with either complexity reduction at the multiplier level or at the filter level is shown in Table II. Here, the coefficient additions are not included. It should be noted that due to pipelining, the number of registers will increase more rapidly when applied on the multiplier level, leading to a significant benefit of the reduced complexity complex-valued FIR filter.

If the filter length is reduced, the length of the FIR filters will be reduced for the FFT+FIR case. For example, with a filter length of 384, only three taps are needed, and, hence, if it is possible to disable one tap, the power consumption will be reduced. For the standard FFT approach, this can not be achieved as easily. One may in the latter case, conceptually, process 128 more samples and reduce the filter length by the same amount. However, the reconfiguration is far from trivial as it requires running the FFTs with different idle times, related to the discussion for $N = 2048$ and $N = 4096$.

The utilization of the different approaches assuming pipelined FFTs as discussed above is shown in Table III.

IV. RESULTS

The 512-tap complex FIR filter operating at about 60 GSa/s supports a fiber length of 2300 km. Assuming 16-QAM, simulations show that 6-bit ADCs can be used. The data

TABLE III
UTILIZATION FOR THE DIFFERENT APPROACHES.

Approach	Utilization
FFT+Mult, $N = 1024$	100%
FFT+Mult, $N = 2048$	66.7%
FFT+Mult, $N = 4096$	57.1%
FFT+Mult, $N = 8192$	53.3%
FFT+Mult, $N = 16384$	51.6%
FFT+FIR, $L = 128$	100%
FIR, $L = 128$	100%

TABLE IV
POWER AND AREA RESULTS FOR THE TWO FREQUENCY-DOMAIN APPROACHES AND BASELINE TIME-DOMAIN CASE.

Approach	Block	Area mm^2	Power W	Count
FFT+Mult	1024-point 128-parallel (I)FFT	4.9	1.59	4
	128 multiplications	1.0	0.48	2
	Total	**21.6**	**7.32**	
FFT+FIR	256-point fully parallel (I)FFT	2.0	0.80	2
	256 4-tap FIR filters	3.6	1.52	1
	Total	**7.6**	**3.12**	
FIR	**Estimated total**	**230**	**97**	

wordlength for both FFTs (and IFFTs) is $12 + 12$ bits and for multiplier/filter coefficients $8 + 8$ bits. The results presented here are for a commercial 65 nm standard cell library and $f_{\text{clk}} = 475$ MHz. This leads to $f_{\text{sample}} = 60.8$ GSa/s.

For the 1024-point (I)FFTs, the approach in [13] is used, while for the 256-point (I)FFTs a fully parallel architecture is used. For both cases, the radix-2^2 algorithm is used.

The power and area results for the different approaches are shown in Table IV. It is clear that the FFT+FIR filter is both smaller and, more importantly, consumes significantly less power. This is despite having about 10% more multiplications per sample as Table I shows. Hence, the constant multiplications in the FFT, the reduced complexity complex-valued FIR filter of Fig. 7, and the fact that the coefficients of the filters do not change together reduces the power consumption significantly. Also included in Table IV are values for a polyphase time-domain realization using FIR filters. These values are estimated based on the FIR filters used in the FFT+FIR approach. As expected both the area and power values are much higher than both filters in the frequency-domain.

The results also illustrate a potential drawback with time-multiplexed pipelined FFT architectures in this context. As stated above, the number of multiplications per sample is slightly lower for the FFT+Mult approach compared to the FFT+FIR approach. Although the FFT+FIR approach has some potential benefits when it comes to constant multiplications etc, a major part not yet mentioned is the amount of storage needed in the data shuffling in the pipelined architecture. Combined with the pipelining, this is a major contributor to the power consumption. Naturally, the fully parallel architecture is also pipelined, but there is no temporal data shuffling involved.

It should be stressed that the results depend on the specifications. Selecting, e.g., $L = 64$ will lead to 26 multiplications per sample for the FFT+FIR case, while the FFT+Mult case is unchanged. Hence, there is a larger difference in the number of multiplication that must be compensated for by the other benefits. Similarly, for $M = 1024$, the FFT+FIR with $L = 128$ will lead to 28 multiplications per sample, while the FFT+Mult approach requires 20 multiplications per sample using 2048-point FFTs. Most likely, the FFT+FIR-approach will still be advantageous in these cases, but the difference will be smaller.

TABLE V
ENERGY PER FILTER TAP.

Approach	Energy per tap, mW/GHz
FFT+Mult	0.20
FFT+FIR	0.10
FIR	3.1

TABLE VI
ESTIMATED POWER CONSUMPTION AND ENERGY PER FILTER TAP FOR THE FFT+FIR APPROACH WITH DIFFERENT FILTER LENGTHS.

Taps	Power, W	Energy per tap, mW/GHz
128	2.56	0.33
256	2.75	0.18
384	2.94	0.13
512	3.12	0.10

For a general case, both approaches should be considered, but it is clear from the results that just comparing the number of multiplications per sample is not enough to determine the best architecture.

It may also be interesting to consider the energy per filter tap, which is shown in Table V. For the FFT+FIR approach, the filter length can be scaled yielding a potential saving. The values are estimates based on interpolating the results for the complete filter and replacing the filter with multipliers. The results are shown in Table VI.

In [11], the 64-tap filter operating at 56 GSa/s consumed 5.7 W in a similar process technology. This corresponds to an energy per tap of 1.59 mW/GHz. Compared to the presented work, it is clear from Table VI that a longer filter have a relatively lower energy per tap. Furthermore, the higher clock frequency of the design in [11] leads to more pipelining registers. It is also likely that the FFT used in the proposed architecture is better optimized compared to the ones used in [11].

V. CONCLUSIONS

In this work two approaches for implementing long high-speed complex FIR filters was discussed. It was shown that for high-speed implementation the architecture choice is more important than the arithmetic complexity. In the example implementations, the FFT+FIR approach had about half the power consumption and one third of the area compared to the FFT+Mult approach, despite having about 10% more multiplications per sample. The best implemented 512-tap FIR filters required about 0.10 mW/GHz per complex filter tap.

REFERENCES

[1] E. Ip and J. M. Kahn, "Digital equalization of chromatic dispersion and polarization mode dispersion," *J. Lightw. Technol.*, vol. 25, no. 8, pp. 2033–2043, 2007.

[2] S. J. Savory, "Digital coherent optical receivers: Algorithms and subsystems," *IEEE J. Sel. Topics Quantum Electron.*, vol. 16, no. 5, pp. 1164–1179, 2010.

[3] D. Lavery, R. Maher, D. S. Millar, B. C. Thomsen, P. Bayvel, and S. J. Savory, "Digital coherent receivers for long-reach optical access networks," *J. Lightw. Technol.*, vol. 31, no. 4, pp. 609–620, Feb. 2013.

[4] S. J. Savory, "Digital filters for coherent optical receivers," *Optics Express*, vol. 16, no. 2, pp. 804–817, 2008.

[5] A. Eghbali, H. Johansson, O. Gustafsson, and S. J. Savory, "Optimal least-squares FIR digital filters for compensation of chromatic dispersion in digital coherent optical receivers," *J. Lightw. Technol.*, vol. 32, no. 8, pp. 1449–1456, Apr. 2014.

[6] A. Sheikh, C. Fougstedt, A. G. i Amat, P. Johannisson, P. Larsson-Edefors, and M. Karlsson, "Dispersion compensation FIR filter with improved robustness to coefficient quantization errors," *J. Lightw. Technol.*, vol. 34, no. 22, pp. 5110–5117, Nov. 2016.

[7] B. S. G. Pillai, B. Sedighi, W. Shieh, and R. S. Tucker, "Chromatic dispersion compensation – an energy consumption perspective," in *Optical Fiber Commun. Conf.* Optical Society of America, 2012, pp. OM3A–8.

[8] E. Serrano, P. Quiroga, A. Taddei, and M. del Barco, "Criteria to minimize power consumption of frequency domain equalizers in VLSI implementations," in *Proc. Workshop Inform. Process. Control*, Oct. 2015, pp. 1–4.

[9] F. de Dinechin, H. Takeugming, and J. M. Tanguy, "A 128-tap complex FIR filter processing 20 giga-samples/s in a single FPGA," in *Proc. Asilomar Conf. Signals Syst. Comput.*, Nov. 2010, pp. 841–844.

[10] S. B. Amado, F. P. Guiomar, and A. N. Pinto, "Digital equalization of chromatic dispersion in an FPGA," in *Proc. Conf. Telecommun.*, 2013.

[11] C. Fougstedt, A. Sheikh, P. Johannisson, A. G. i Amat, and P. Larsson-Edefors, "Power-efficient time-domain dispersion compensation using optimized FIR filter implementation," in *Signal Process. Photonic Commun.* Optical Society of America, 2015, pp. SpT3D–3.

[12] H. Kwan and M. Tsim, "High speed 1-D FIR digital filtering architectures using polynomial convolution," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, vol. 12, Apr. 1987, pp. 1863–1866.

[13] M. Garrido, J. Grajal, M. A. Sanchez, and O. Gustafsson, "Pipelined radix-2^k feedforward FFT architectures," *IEEE Trans. VLSI Syst.*, vol. 21, no. 1, pp. 23–32, Jan. 2013.