Hardware-optimal digital FIR filters: one ILP to rule them all and in faithfulness bind them

Anastasia Volkova*, Rémi Garcia†, Florent de Dinechin‡, and Martin Kumm§
*Univ Lyon, Inria, INSA Lyon, CITI, EA3720, 69621 Villeurbanne, France
†Nantes Université, CNRS, LS2N, Nantes, France
‡Univ Lyon, INSA Lyon, Inria, CITI, EA3720, 69621 Villeurbanne, France
§Fulda University of Applied Sciences, Germany
*anastasia.volkova@inria.fr †remi.garcia@univ-nantes.fr ‡florent.de-dinechin@insa-lyon.fr §martin.kumm@cs.hs-fulda.de

***Abstract*—This article addresses the implementation of Finite Impulse Response filters as digital hardware circuits. It formalizes, as a mathematical model, the problem of finding the optimal circuit for a given frequency specification and given input/output fixed-point formats. This model captures at the bit level a wide class of implementations (transposed-form circuits based on truncated shift-and-add adder graphs). It also captures formally the constraints due to the frequency specification, as well as those due to rounding to the output format. This model can be expressed as an Integer Linear Programming (ILP) problem, such that the optimal circuit (in terms of bit-level adders and registers) can be found by standard ILP solvers. This approach allows for a completely automatic tool from a frequency specification to a circuit with user-specified input and output formats. This tool is evaluated (with cost functions modeling FPGAs) on several benchmarks.**

I. Introduction

The hardware implementation of digital filters has received a lot of attention in the last half century. This article addresses the construction of circuits implementing Finite Impulse Response (FIR) filters for a given frequency specification (Fig. 1).

Classically, this process begins with a **filter design** (FD) step that determines the real-valued coefficients h_i of a polynomial transfer function $\mathcal{H}(z) = \sum_{i=0}^{N} h_i z^{-i}$ such that the frequency specification is strictly satisfied. A prerequisite for a hardware implementation is the **quantization** (Q) of the real coefficients into finite precision (fixed-point) data formats, in such a way that the frequency response is still respected. Using the quantized coefficients, a digital circuit C can be **implemented** (I) as one of a variety of filter structures (e.g., direct 1 or transposed form 2 in Fig. 1). For each filter structure, there are also multiple techniques to build the corresponding hardware multipliers and adders, and then each operator can be sized and rounded to optimize the resources. Two examples are shown in Fig. 1 as 3 and 4.

The purpose of this article is to address the construction of *optimal* circuits. To our knowledge, the state of the art, so far, only partially answers this problem: previous works (reviewed in Section II) either explore only a subset of the implementation space, or use heuristics, or combine local optimizations instead of a single global one, or generally do not address the issue of rounding. This last issue is particularly relevant to actual applications: typical circuits have the same input and output formats (for instance 16-bit fixed-point), thus making rounding mandatory in practice. But rounding is not linear, therefore rounded circuits are not linear time-invariant (LTI), and strictly speaking their frequency response is ill-defined.

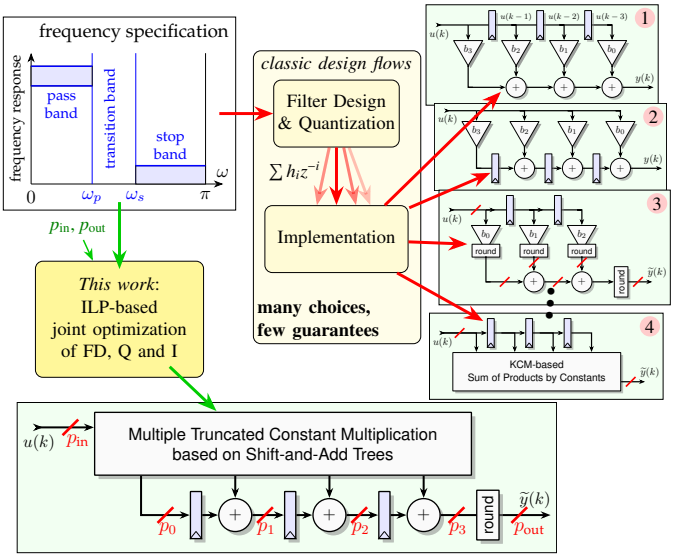


Fig. 1: From a frequency specification to an architecture.

In signal processing, rounding errors are typically modeled as noise, and the measure of the quality of the implementation step is a signal/quantification noise ratio (SQNR), in dB. This approach remains statistical, and founded on statistical hypotheses that are not always true in practice. The present work adopts a stricter point of view where the quality measure is the worst-case error. The format of the output word (e.g.,

8 bits) can be used as a specification of the accuracy (e.g., the absolute error on the output should not exceed 2^{-8}). This is called faithful rounding, and it can be extended to the frequency domain [1] as follows: "*A digital circuit C is faithful to a frequency specification $\mathcal{S}$ when there exists an LTI filter $\mathcal{L}$ satisfying $\mathcal{S}$ such that the output of C is a faithful rounding of the output of $\mathcal{L}$.*"

This definition was a missing brick in the construction of a model of practical circuits (i.e., including rounding) respecting frequency specification constraints. In the proposed model,

- the **implementation space** is the set of transposed-form circuits where the multiple constant multiplication (MCM) is implemented as a graph of (possibly truncated) add/subtract nodes; it is detailed in Section III.
- the **unknowns** are the filter coefficients and the numerous parameters of the implementation (the shift-and-add graph, the positions of truncations, etc.);
- the **cost function** is a function of the unknowns that models the cost of registers and adders at the bit level;
- the **quality constraints** express faithfulness in the worst case as stated above.

Finding the optimal circuit in this model can be expressed as a (mixed) Integer Linear Programming (ILP) problem. ILP is an efficient and versatile formalism to find optimal solutions of combinatorial problems over integer variables under linear constraints. Many powerful ILP solvers exist, so this approach can rely on them to actually perform the optimization. The detail of this ILP formulation is given in Section IV, and it is evaluated in practice in Section V.

II. Related work and positioning

This article completes a historical trend to replace a succession of local optimization steps FD, Q and I with a global one. The combination of the FD + Q steps has been studied since the 1960's [2], and can even be regarded as solved for certain practical instances of fixed-point FIR design [3]–[5].

The I step was historically treated separately, with a focus on multiplierless implementation of the MCM block using bit-shifts and additions. The construction of efficient MCM blocks can be based on the binary encoding of the coefficients [6]–[11], or can use graph-based heuristics [12]–[14]. Optimal MCM techniques were proposed relatively recently using dedicated Branch&Bound (B&B) algorithms [15], [16] or an ILP model [17], [18].

The combined FD+Q+I problem has been solved using dedicated heuristics and B&B algorithms but in a search space restricted to special encodings [19], [20]. Recent optimal B&B [21], [22] and ILP-based [23] approaches solve a complete FD+Q+I problem, but only in the space of (impractical) circuits that compute the filter output exactly. Besides, these works optimize the number of adders, while adders in an MCM may have different costs. The optimal truncated MCM problem was solved only later [18].

In this work, we use as basis the versatile ILP model from [23], as it is not as sensitive as the B&B [21], [22] to design-space size explosion. We improve it to provide practical implementations with internal data path truncations that are optimized but guarantee faithfulness. We achieve this by (1) using the truncated MCM model [18] counting resources at the bit-level (and not adder-level as in previous literature), and (2) linking the output accuracy requirement with truncation choices that are explored by the solver.

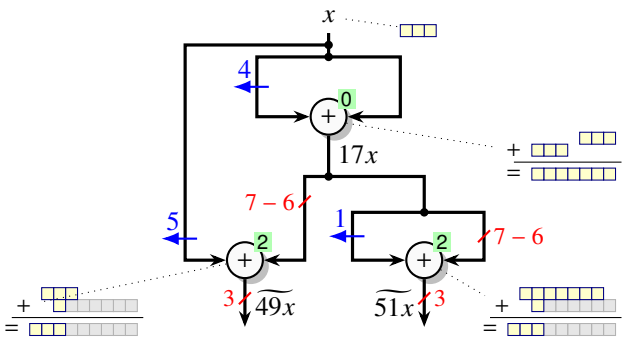


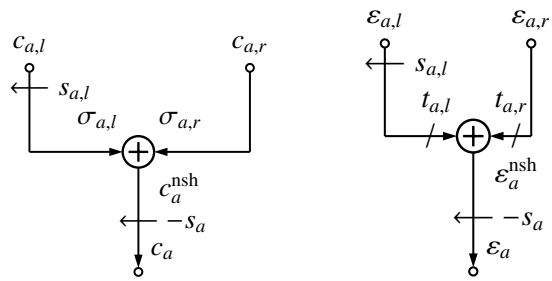
Fig. 2: An adder graph computing $49x$ and $51x$ in 3 additions, with its optimal truncations when the inputs are 3-bit numbers and a 3-bit output is needed. The cost of each adder is given in the green labels.

III. Truncated Multiple Constant Multipliers

In approaches based on shift-and-add circuits, an MCM may be represented as an adder-graphs, e.g., Fig. 2. The root of the adder graph is the input integer x. Each node represents the addition of two potentially negated and shifted inputs – a shift multiplies by a power of two. Each adder thus computes an intermediate factor, called its fundamental. For example, in Fig. 2 the first adder computes $17x = x \ll 4 + x = 2^4 \cdot x + x$.

For practical FIR filter implementations, the output of the MCM block will potentially be truncated to some intermediate format. To avoid computing unnecessary bits that will be rounded anyway, the truncations should be lifted into the adder graph while guaranteeing the faithfulness of the results. The Truncated MCM ILP model proposed in [18] solves this problem optimally: given a set of integer constants and associated input/output data word lengths, construct an adder graph describing a multiplierless solution together with the potential intermediate truncations that guarantee faithful rounding based on a worst-case error model.

At the bit level, an adder or subtracter is built out of full adders and half adders, and these may include inversions for subtractions. We do not distinguish these various cells in this work as they have the same cost on the FPGAs that we target for our experiments, and use the generic term *one-bit adder*. Fig. 2 also shows that for 3-bits inputs and outputs, the MCM requires 3 adders but these cost only 4 one-bit adders (green labels on each adder node). Indeed, since a 3-bit approximation to $51x$ is enough, removing the 6 lower bits from the 7-bit signal $17x$ (indicated as the label 7-6 on the wire) will save 5 one-bit adders while still ensuring the faithful rounding of both outputs. In total, truncations save 6 one-bit adders out of the 10 required for the exact MCM.



(a) Topology model (b) Error model

Fig. 3: Modeling adder graph topology and errors

In an ILP model [18], an adder node a compute the fundamental c_a (see Fig. 3a). The model aims at finding the best adder-graph topology such that the leaves of the graph compute every constant in the target set. The formal constraint linking together the left and right inputs of an adder node ($c_{a,l}$ and $c_{a,r}$) with the shift ($s_{a,l}$), the negations ($\sigma_{a,l}$ and $\sigma_{a,l}$) and a potential right-shift after the node (s_a) is the following:

$$c_a = 2^{-s_a}\left((-1)^{\sigma_{a,l}}\, 2^{s_{a,l}} c_{a,l} + (-1)^{\sigma_{a,r}}\, c_{a,r}\right). \tag{1}$$

The number of one-bit adders in each node, B_a, is in the worst case equal to its output word length and this can be propagated through the graph. However, certain topologies can be more advantageous (e.g., in Fig. 2, $17x$ is computed from non-overlapping numbers and uses no one-bit adders if x is unsigned). Similarly, truncating the data before additions can save one-bit adders. Numerous special cases were incorporated in the ILP model [18].

The objective is then to minimize the total number of one-bit adders,

$$\min \sum_a B_a. \tag{2}$$

Each truncation of t bits, however, induces an error $|\varepsilon_t| \leq 2^t$, which needs to be propagated. In the ILP model, an error ε_a is associated to each adder-node a and is influenced by the incoming shifted errors and a truncation-induced error (see Fig. 3). This is formalized by the following constraint:

$$\varepsilon_a = 2^{-s_a}\left(2^{s_{a,l}}\varepsilon_{a,l} + \varepsilon_{a,r} + 2^{t_{a,l}} + 2^{t_{a,r}}\right) \tag{3}$$

Finally, the error of each output adder c_{out} must be bounded by the corresponding $\overline{\varepsilon}_i$, deduced from the user specification.

Equation (3), together with the one-bit adder gain due to truncations t_a of each adder, permits the solver to automatically find the trade-off between the one-bit adder count B_a and the error bounds $\overline{\varepsilon}_i$. We refer the reader to [18] for the complete linearized model, which we integrated into our tool.

IV. One ILP to rule them all

Hardware-aware filter design problems usually start with a functional specification of the frequency domain behavior, together with the number of filter coefficients. An optimization procedure constraints filter coefficients to integer values and aims at minimizing the hardware resources required for the MCM block.This section incorporates the fine-grained Truncated MCM model [18] into the ILP formulation based on [23] and give a new model of all sources of errors to dispatch the acceptable error-budget between data paths and maximize truncations.

TABLE I: Relation between filter order N, number of coefficients M and function $c_m(\omega)$ for different filter types

Type	Sym.	N	M	$c_m(\omega)$
I	sym.	even	$\frac{N}{2}+1$	$c_m(\omega) = \begin{cases} 1 & \text{for } m = 0 \\ 2\cos(\omega m) & \text{for } m > 0 \end{cases}$
II	sym.	odd	$\frac{N+1}{2}$	$c_m(\omega) = 2\cos(\omega(m+1/2))$
III	asym.	even	$\frac{N}{2}$	$c_m(\omega) = 2\sin(\omega(m-1))$
IV	asym.	odd	$\frac{N+1}{2}$	$c_m(\omega) = 2\sin(\omega(m+1/2))$

A. Design of linear-phase FIR filters with fixed-point coefficients

An N-th order linear phase FIR filter can be described by its zero-phase frequency response which has the property that its magnitude is identical to that of the transfer function. Let $\underline{D}(\omega)$ and $\overline{D}(\omega)$ be the desired lower and upper bounds of the output frequency response $H_R(\omega)$. The associated frequency specification-based FIR filter design problem consists of finding real coefficients h_m, $m = 0, \ldots, M-1$ that fulfill

$$\underline{D}(\omega) \leqslant \sum_{m=0}^{M-1} h_m c_m(\omega) \leqslant \overline{D}(\omega), \qquad \forall \omega \in \Omega, \tag{4}$$

where $\Omega \subseteq [0, \pi]$ is a set of target frequency bands (usually pass and stopbands) and $c_m(\omega)$ terms are trigonometric functions. Relation between $c_m(\omega)$, number of coefficients M, the degree N and the filter type is given in Table I The FD problem (4) is a semi-infinite constraint, but it is usually discretized over $\Omega_{\text{d}} \subseteq \Omega$ [23].

Fixed-point FIR filter design problems further restrict the search space to integer variables h'_m with $|h'_m| < 2^B$, where the coefficients of $H_R(\omega)$ are

$$h_m = 2^{-B} h'_m \tag{5}$$

and B is the *maximum effective word length* of each coefficient (excluding sign bit).

Applying (5) to the discretized version of (4) we obtain a finite number of linear constraints over integer variables h'_m

$$2^B \underline{D}(\omega) \leqslant \sum_{m=0}^{M-1} h'_m c_m(\omega) \leqslant 2^B \overline{D}(\omega), \qquad \forall \omega \in \Omega_d. \tag{6}$$

B. Implementation space and cost function

MCM block: The filter coefficients h'_m should now be linked to the inputs of the truncated MCM problem. Similarly to [23], we exploit the versatility of ILP modeling and export the Truncated MCM model into the global ILP and provide the so-called glue constraints. First, we need to connect the integer

filter coefficients h'_m to the target constant set of Truncated MCM model:

$$h'_m = (-1)^{\phi} 2^s c_{a^M} \text{ if } o_{a,m,s,\phi} = 1 \tag{7}$$

where a^M denotes a multiplier-block adder, ϕ is the coefficient sign and binary variable $o_{a,m,s,\phi}$ encodes if h'_m, potentially shifted by s bits, is connected to the fundamental c_a.

Structural adder chain: Then, we need to connect the outputs of the Truncated MCM model to the structural adders. Denote a^S a set of at most $N-1$ structural adders, and B_{a^S} its one-bit adder cost. According to symmetries and even/odd degree of the filter (see Table I), most of the MCM-block coefficients h_m will appear twice in the adder chain. Since inputs of the structural adder are independent, we have to compute the number of one-bit adders based on worst-case data word length. The only way to gain one-bit adders is to introduce truncations t_{a^S-1} and t_{a^M} on the preceding structural adder and corresponding MCM-block output, respectively. Hence, the one-bit adder cost of a structural adder is, recursively,

$$B_{a^S} = \text{msb}_{a^S-1} + \text{msb}_{a^M} - \max(t_{a^S-1}, t_{a^M}), \tag{8}$$

where msb_{a^S-1} corresponds to the most significant bit position of the preceding structural adder, and msb_{a^M} corresponds to the correctly-computed multiplier-block output, respecting the filter type. We define msb_0 to the MSB position of $h_0\overline{x}$.

Cost function: In contrast to previous works, the use of Truncated MCM permits us to fine-tune the cost function to the gate-level. Our objective is to minimize the total number of one-bit adders in the MCM block and structural adder chain:

$$\min \sum_{a^M \cup a^S} (B_{a^M} + B_{a^S}) . \tag{9}$$

C. Data-path sizing constraints

As introduced in [1], faithfulness of a digital circuit to frequency specifications has two components: behavior of the linear part of the circuit is guaranteed by the FD constraints (4) and behavior of the nonlinear part due to internal rounding should be suppressed to the last bit of the computed output signal. Denote $\varepsilon_{\text{out}} = \widetilde{y}(k) - y(k)$ the output error, where $\widetilde{y}$ is the finite-precision counterpart of the real signal y. The faithfulness to output word length specification, e.g., in terms of the least significant bit position $2^{\ell_{\text{out}}}$ translates into the constraint $|\varepsilon_{\text{out}}| < 2^{\ell_{\text{out}}}$. Our task now is to define the link between the output error ε_{out}, the structural adder errors ε_{a^S} and the MCM-block errors ε_{a^M} that constrain the truncations.

First, we need to account for the final rounding (see Fig. 1 bottom), which induces an error that is bounded by $2^{\ell_{\text{out}}-1}$. Hence, the last structural adder must actually satisfy the following constraint: $\overline{\varepsilon} = |\varepsilon_{a^S}| < 2^{\ell_{\text{out}}-1}$, $a^S = N-1$.

We can recursively define the error-propagation rule through each potentially truncated structural adder:

$$\varepsilon_{a^S} = \varepsilon_{a^S-1} + \varepsilon_{a^M} + 2^{t_{a^S}} + 2^{t_{a^M}}, \tag{10}$$

where the the errors from the preceding registers and from the MCM block are added to the potential truncation errors. By counting the number of trailing zeros in inputs of each adder, as in [18], we significantly tighten the propagated error bound, since truncating those does not induce any error.

With (10) we provided the link to Truncated MCM block, and relate the truncations in structural adders with the overall rounding error, completing the global ILP model. The full list of constraints can be found in the tool's web page.

TABLE II: Specifications of the filters used in experiments

Name	Ω_p/π	Ω_s/π	δ_p	δ_s
T	$[0, 0.3]$	$[0.5, 1]$	0.01	0.01
S2	$[0, 0.042]$	$[0.14, 1]$	0.026	0.001

V. Experimental Results

To evaluate our method, we implemented the ILP model generation with Julia within the *Shift-And-Add circuits for digital FIR filters* (SAFIR) project[1]. To be able to perform hardware experiments, we used FloPoCo[2] and implemented a new operator `IntFIRTransposed`. The automatic test bench generation of FloPoCo was used to validate the obtained designs. All implemented tools are published as open-source.

We test our method on a reference specifications from the literature commonly referred to as $S2$ and used in many previous works [11], [20], [22]–[24]. We also introduce a small toy filter T for illustration purpose.

They are low-pass filters defined by

$$\begin{aligned} 1-\delta_p \leqslant H_R(\omega) \leqslant 1+\delta_p, \qquad & \omega \in \Omega_p \quad \text{(passband)}, \\ -\delta_s \leqslant H_R(\omega) \leqslant \delta_s, \qquad & \omega \in \Omega_s \quad \text{(stopband)}, \end{aligned}$$

where the values of $\delta_p, \delta_s, \Omega_p, \Omega_s$ for each specification are given in Table II.

For the ILP solving, we used Gurobi Optimizer [25] v10.0 with a timeout limit of 8 hours. The input word size was set to $w_{\text{in}} = 8$ bit. The results for different filters and different values of $\overline{\varepsilon}$ are given in Table III. To get a target output word size of w_{out}, $\overline{\varepsilon}$ has to be set to $\overline{\varepsilon} = 2^{\Delta w_{\text{out}}-1}$, where $\Delta w_{\text{out}} = w_{\text{out,full}} - w_{\text{out}}$ is the difference between the output word size of the full precision output and the target output word size. The table gives the number of adder/subtractors, and the number of one-bit adders for the MCM block (MCM), the structural adders (SA) and the total number (tot). For T, the ILP proves the optimality of solutions (one is shown in Fig. 4), but for S2 the model is too large and the optimality cannot be proven within the timeout. Still, the found solutions save one-bit adders thanks to truncation as expected.

Synthesis experiments have been performed for all designs using Vivado 2019.1 targeting an AMD Kintex (xc7k70tfbv484-3). The resulting LUT numbers and the critical path delay t_{cp} are shown in the last two columns of Table III. As each one-bit adder requires one LUT on current FPGAs (apart from unpredictable optimizations in synthesis),

[1] https://gitlab.com/filteropt/safir

[2] https://flopoco.org

TABLE III: Optimization and synthesis results

Filter	Target Error	#adders			#half/full adders			synthesis	
		MCM	SA	tot	MCM	SA	tot	LUTs	t_{cp} [ns]
T	$\bar{\varepsilon} = 0$	2	8	10	16	90	106	111	6.05
	$\bar{\varepsilon} < 2^4$	2	8	10	16	85	101	111	6.05
	$\bar{\varepsilon} < 2^8$	2	8	10	16	71	87	95	6.06
S2	$\bar{\varepsilon} = 0$	15	51	66	139	892	1031	1038	9.79
	$\bar{\varepsilon} < 2^4$	15	51	66	134	886	1020	1025	9.40
	$\bar{\varepsilon} < 2^8$	15	41	66	137	840	977	996	10.57

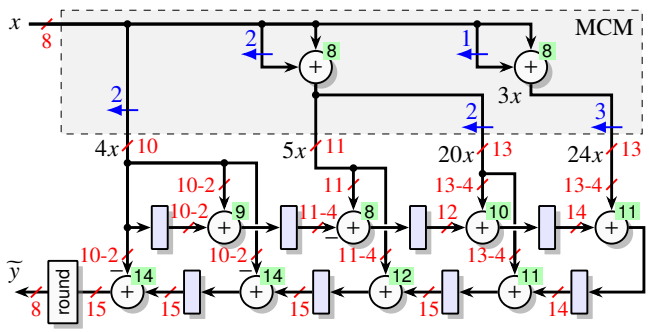


Fig. 4: The complete solution for the toy filter T faithful to 8 bits ($\bar{\varepsilon} < 2^6$) on signed inputs. Note that three of the structural adders actually compute subtractions, to implement the coefficients -4.

the obtained LUTs closely follow the number of one-bit adders from the optimization.

VI. Conclusion

Based on the recent improvements in the ILP-based approaches for the design, quantization and implementation of multiplierless FIR filters, we develop a last missing piece in the mathematical modeling and optimal implementation of hardware-efficient FIR filters. The versatility of ILP models ensures that changing the metric or updating the implementation method is as simple as adding new variables and constraints into the model. We present a specifications-to-VHDL push-button tool, SAFIR, which takes care of all operator design, sizing and rounding automatically, letting the digital filter designer to focus on more high-level implementations questions, e.g., input/output word lengths.

References

[1] F. de Dinechin, S.-I. Filip, M. Kumm, and A. Volkova, "Towards Arithmetic-Centered Filter Design," in *2021 IEEE 28th Symposium on Computer Arithmetic (ARITH)*. IEEE, Jun. 2021.

[2] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, 3rd ed. USA: Prentice Hall Press, 2009.

[3] D. M. Kodek, "LLL algorithm and the optimal finite wordlength FIR design," *IEEE Trans. on Sig. Proc.*, vol. 60, no. 3, pp. 1493–1498, 2012.

[4] N. Brisebarre, S.-I. Filip, and G. Hanrot, "A lattice basis reduction approach for the design of finite wordlength FIR filters," *IEEE Transactions on Signal Processing*, vol. 66, no. 10, pp. 2673–2684, 2018.

[5] D. M. Kodek, "An algorithm for the design of optimal finite wordlength FIR filters," *Digital Signal Processing*, vol. 144, pp. 1–7, 2024.

[6] H. Samueli, "An improved search algorithm for the design of multiplierless FIR filters with powers-of-two coefficients," *IEEE Transactions on Circuits and Systems*, vol. 36, no. 7, pp. 1044–1047, Jul. 1989.

[7] J. Yli-Kaakinen and T. Saramaki, "A systematic algorithm for the design of multiplierless FIR filters," *IEEE International Symposium on Circuits and Systems*, vol. 2, pp. 185–188, 2001.

[8] O. Gustafsson and L. Wanhammar, "Design of linear-phase FIR filters combining subexpression sharing with MILP," in *Midwest Symposium on Circuits and Systems*. IEEE, Aug. 2002, pp. III–9–III–12.

[9] A. P. Vinod and E.-K. Lai, "On the implementation of efficient channel filters for wideband receivers by optimizing common subexpression elimination methods," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 2, pp. 295–304, 2005.

[10] M. Aktan, A. Yurdakul, and G. Dundar, "An algorithm for the design of low-power hardware-efficient FIR filters," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 55, no. 6, pp. 1536–1545, 2008.

[11] A. Shahein, Q. Zhang, N. Lotze, and Y. Manoli, "A Novel Hybrid Monotonic Local Search Algorithm for FIR Filter Coefficients Optimization," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 59, no. 3, pp. 616–627, 2012.

[12] Y. Voronenko and M. Püschel, "Multiplierless Multiple Constant Multiplication," *ACM Trans. on Algorithms*, vol. 3, no. 2, pp. 1–38, 2007.

[13] O. Gustafsson, "A Difference Based Adder Graph Heuristic for Multiple Constant Multiplication Problems," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2007, pp. 1097–1100.

[14] ——, "Towards Optimal Multiple Constant Multiplication: A Hypergraph Approach," in *Asilomar Conference on Signals, Systems and Computers (ACSSC)*. IEEE, Oct. 2008, pp. 1805–1809.

[15] L. Aksoy, E. Da Costa, P. Flores, and J. Monteiro, "Exact and approximate algorithms for the optimization of area and delay in multiple constant multiplications," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 6, pp. 1013–1026, 2008.

[16] N. P. Lopes, L. Aksoy, V. Manquinho, and J. Monteiro, "Optimally solving the MCM problem using pseudo-boolean satisfiability," *arXiv preprint arXiv:1011.2685*, 2010.

[17] M. Kumm, "Optimal Constant Multiplication using Integer Linear Programming," in *IEEE Int. Symp. on Circ. and Systems (ISCAS)*, 2018.

[18] R. Garcia and A. Volkova, "Toward the Multiple Constant Multiplication at Minimal Hardware Cost," *IEEE Transactions on Circuits and Systems I: Regular Papers*, pp. 1–13, 2023.

[19] Y. Yu and Y. Lim, "Optimization of Linear Phase FIR Filters in Dynamically Expanding Subexpression Space," *Circuits, Systems, and Signal Processing*, vol. 29, no. 1, pp. 1–16, 2009.

[20] D. Shi and Y. J. Yu, "Design of Linear Phase FIR Filters With High Probability of Achieving Minimum Number of Adders," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 58, no. 1, pp. 126–136, 2011.

[21] L. Aksoy, P. Flores, and J. Monteiro, "SIREN: A depth-first search algorithm for the filter design optimization problem," in *Proceedings of the 23rd ACM International Conference on Great Lakes Symposium on VLSI*. ACM, 2013, p. 179–184.

[22] ——, "Exact and Approximate Algorithms for the Filter Design Optimization Problem," *Signal Processing, IEEE Transactions on*, vol. 63, no. 1, pp. 142–154, 2015.

[23] M. Kumm, A. Volkova, and S.-I. Filip, "Design of Optimal Multiplierless FIR Filters With Minimal Number of Adders," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, no. 2, pp. 658–671, Feb. 2023.

[24] Y. J. Yu and Y. C. Lim, "Design of Linear Phase FIR Filters in Subexpression Space Using Mixed Integer Linear Programming," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 54, no. 10, pp. 2330–2338, 2007.

[25] Gurobi Optimization Inc., "Gurobi Optimization Website." [Online]. Available: http://www.gurobi.com