ICCS 2023 & ICITES 2023

A Resource-efficient FIR Filter Design Based on an RAG Improved Algorithm

Mengwei Hu, Zhengxiong Li, Xianyang Jiang*
Wuhan University

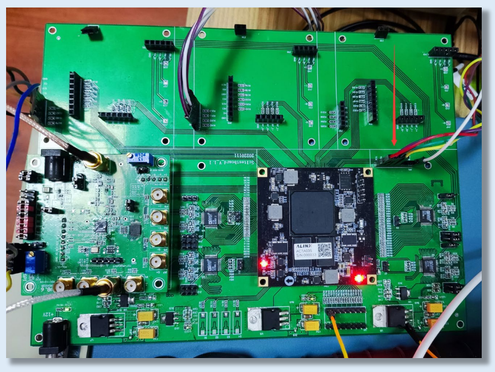
Abstract: In modern digital filter chip design, efficient resource utilization is a hot topic. Due to linear phase characteristics of FIR filters, a pulsed fully parallel structure can be applied to attack the problem. In order to further reduce hardware resource consumption especially caused by multiplication function, an improved RAG algorithm is proposed. Filters with different orders and for different algorithms are compared, and the experimental results show that the improved RAG algorithm is excellent in terms of logic resource utilization, resource allocation, running speed, and power consumption under different application scenarios. The proposed algorithm invokes a better circuit structure for FIR filter, it gives full play to resource allocation strategy and reduces logic resource consumption. The proposed circuit is faster and more stable, and suitable for a variety of complex application scenarios.

Motivation

Sources:
- Comes from a real-world project about RFIC calibration
- Used to calculate the leak energy

Demands:
- High-speed & High-performance
- Limited resources

Conflict:

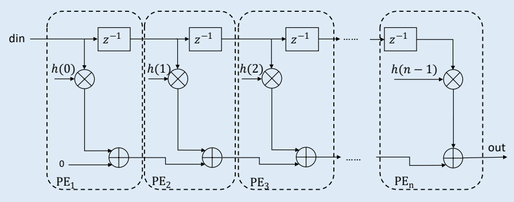


Demand 1: High-speed → Full parallel; Short critical path; Pre-add (Wallace tree)

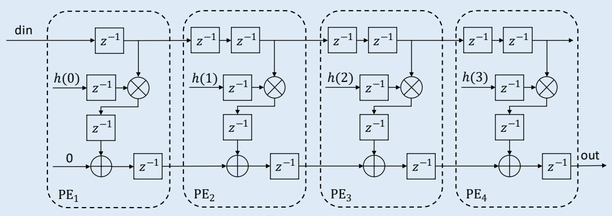
Demand 2: Resources-efficient ✕ A lot of resources required

Traditional Architecture

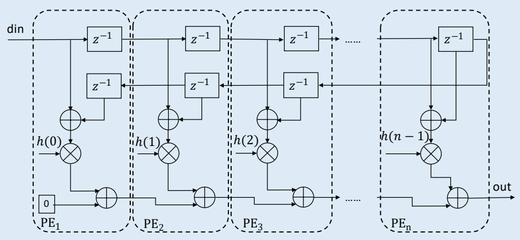
Some existing FIR filters' architecture



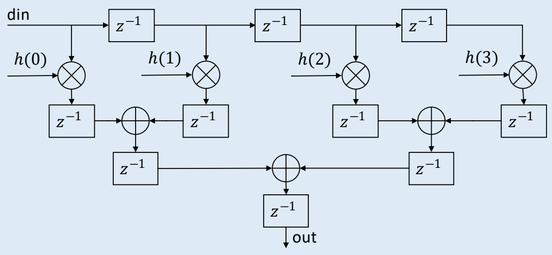
Basic



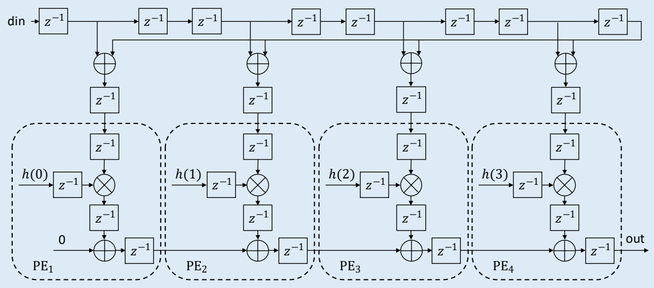
Systolic



Even Symmetry



Wallace Tree



Systolic + Even symmetric

Our Idea

Most of the resources are used to calculate addition and multiplication.

The highest clock frequency is restricted by critical path.

Reduced Adder Graph (RAG) algorithm

Multiplication → Shift & Add

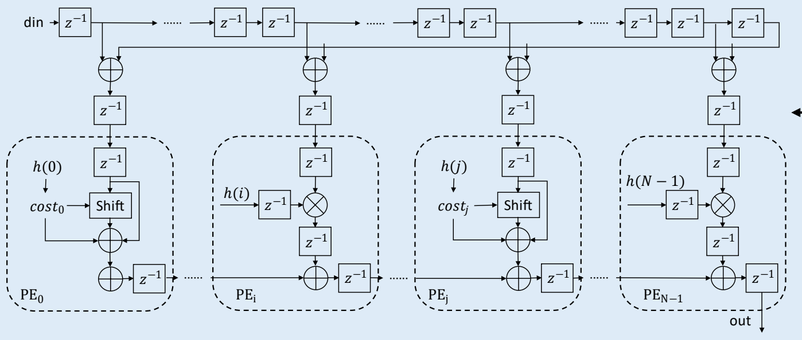
Concise summary:

For those smaller coefficients, shift and add.

For those bigger coefficients, multiply.

Methodology

Step 1: Calculate the absolute values of all coefficients and store them in the $coeff$.
Step 2: Remove duplicate coefficients and coefficients with value 2^n, resulting in a set denoted as N.
Step 3: Place the smaller coefficients into $coeff_r$, containing N/2 or (N-1)/2 coefficients.
Step 4: Store the remaining larger coefficients in $coeff_s$.
Step 5: Divide the even numbers in $coeff_r$ by 2^n to determine the base values.
Step 6: Use a table to find the corresponding adder depth for each base number; store these coefficients in $cost_n$, while coefficients not found in the table are stored in $cost_o$.
Step 7: Realize the coefficients in $cost_1$, calculate the sum/difference of coefficients in all realized cost sets, and use these to realize coefficients in higher cost sets, finally realizing the coefficients in $cost_o$.
Step 8: Implement the coefficients in $coeff_s$ according to the hardware structure of a systolic FIR filter with symmetric coefficients.



RAG Improved (Ours)

Merit: Balance the usage of DSP and LUT

Synthesis Results

Table 1: 64th order FIR filter

Resources /Performance	Algorithm Architecture		
	Pulsed Fully Parallel	RAG Algorithm	RAG Improved Algorithm
LUT	574	4956	934
FF	1286	528	904
DSP	4	0	2
Power(W)	32.8	234.7	38.6
Temp(℃)	70.8	125.0	79

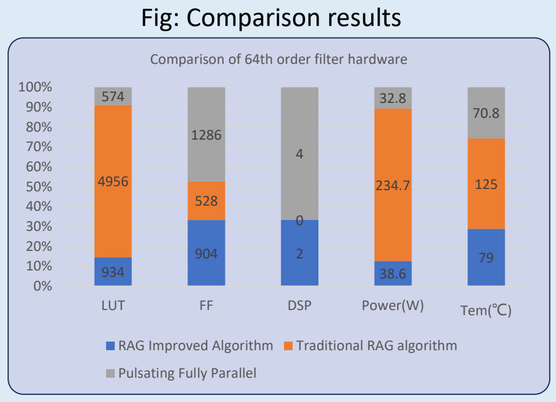
Table 2: 32th order FIR filter

Resources /Performance	Algorithm Architecture		
	Pulsed Fully Parallel	RAG Algorithm	RAG Improved Algorithm
LUT	358	695	555
FF	679	287	538
DSP	4	0	2
Power(W)	21.34	24.52	19.75
Temp(℃)	54.8	59.3	52.6

Table 3: 8th order FIR filter

Resources/ Performance	Algorithm Architecture		
	Pulsed Fully Parallel	RAG Algorithm	RAG Improved Algorithm
LUT	141	212	185
FF	203	120	222
DSP	4	0	2
Power(W)	41.762	33.673	36.75
Temp(℃)	83.4	72.1	76.4

Fig: Comparison results



Conclusion

- **Growing Demand for FIR Filters**: FIR filters are increasingly used in diverse applications with customized requirements.
- **Innovative Design Solution**: The paper proposes a new algorithmic structure suitable for most scenarios, especially when dealing with numerous filter orders and coefficients.
- **Resource Optimization**: The improved RAG algorithm optimizes resource allocation, using shift and add operations instead of direct multipliers.
- **Enhanced Speed and Efficiency**: Combining RAG with a fully parallel structure improves operation speed and resource utilization.
- **Resource Savings**: The design significantly reduces logic resource usage.
- **Low-Power Focus**: The design considers low-power requirements, a critical factor in modern applications.
- **Balanced Speed and Efficiency**: It addresses the challenge of balancing speed and resource consumption in FIR filter applications.