Politecnico di Torino

III Facoltà di Ingegneria

Design and Implementation of a DigitalFilter Integrated Systems Architecture

Master's degree in Electronic Engineering

Authors: group 24

Campanella Andrea, Iacovelli Gianluca, Pala Stefano

Contents

CHAPTER 1

Lab 1: design and implementation of a digital filter

1.1 Reference model development

1.1.1 Design of a digital IIR filter direct form II

The delivery of the laboratory required the development of an IIR digital filter with direct form II architecture with the following features:

- Cut-off frequency $F_T = 2kHz$
- Number of bits $N_b = 10$
- Filter order $N = 1$
- Total Harmonic Distorsion $THD \leq -30dB$

Taking into account the equation that describes the behavior of the IIR filter:

$$y(n) = \sum_{i=0}^{N} a_i x[n-i] + \sum_{j=1}^{N} b_j y[n-j] \tag{1.1}$$

It is possible to evince the dependence of the filter output as a function of the coefficients a_i and b_j, these are necessary and define the global behavior of the IIR filter, so it was necessary to use tools for their computation. For this purpose Matlab was used, using the scripts made available on the *Portale della didattica*: *my_filter.m* and *my_iir_design.m.*These files made use of the function *butter*, which generates the coefficients of the digital filter with Butterworth response (maximum flat), with the desired filter order and the normalized cut-off frequency compared to the Nyquist frequency. The coefficients quantized at $10bits$ obtained are as follows:

- a =[512 -82]
- b =[215 215]

Github repository: https://github.com/campandrea/Lab_ISA/

Matlab's script also generated an implementation of the filter to show the response to two input signals, one in band and the other out of band; it also produced the graph of the frequency response of the filter, as showed in Figure 1.1:

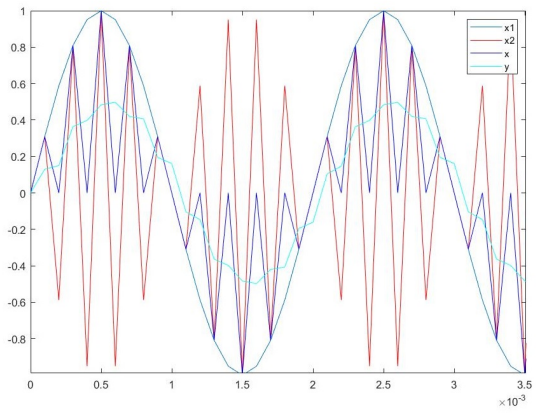

Figure 1.1: Filter Input and Output Signals

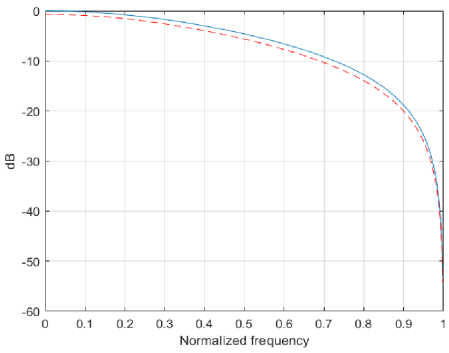

Figure 1.2: Butterworth response of the IIR filter

.

Signals x_1 and x_2 are the in-band signal and the out-of-band signal respectively, the x signal is the average between the two and is given as input to the Matlab program. In fact, as you can see from the output, the filter behaved exactly like a low pass, letting only the low frequency component to pass and attenuating all the others. Then, in Figure 1.2 the Butterworth response is represented as a function of the normalized frequency of the digital IIR filter.

1.1.2 Fixed point C models

After the development of the IIR filter in Matlab environment, a C language version was implemented to evaluate the performance of the filter in arithmetic *Fixed-point*. For this purpose it was used the C language program available on the teaching portal called *myfilterii.c*, to which were passed from the command line the sample file generated by Matlab and the file to save the output samples from the filter implemented in C language. The program produced an output file that was then analyzed in Matlab environment to evaluate the Total Harmonic Distortion(THD). Having chosen an initial representation of the 10 bits coefficients, the following THDs presented in Figure 1.3 have been obtained respectively with the output signal obtained on Matlab and then obtained in C fixed point arithmetic:

As can be seen from the graph in Figure 1.4, the Total Harmonic Distortion of the filter that made use of coefficients and signal represented on 10 bits is too high compared to the assigned specifications. Therefore the process described above has been iterated, using at each cycle one bit less in the representation, until the optimal value of bits has been obtained that allows to remain within the specifications of the *THD*, which corresponds to 5 bits. Then the THDs graphs have been reported, respectively calculated with the output signal obtained on Matlab and followed by the one evaluated with the fixed point arithmetic in C:
The final value of THD is therefore $-21.26dB$ which is perfectly in line with the assigned project specifications. From the graphs in **??**, it can be seen that in the first half of the observation window the signal has a high noise, generating a low stability of the oscillator, but over time it seems to weaken, increasing its frequency stability. From the difference of the two graphs instead we can notice that the frequency behavior of the two signals is very similar, because the difference of their variances is much below the order of magnitude of the unit.

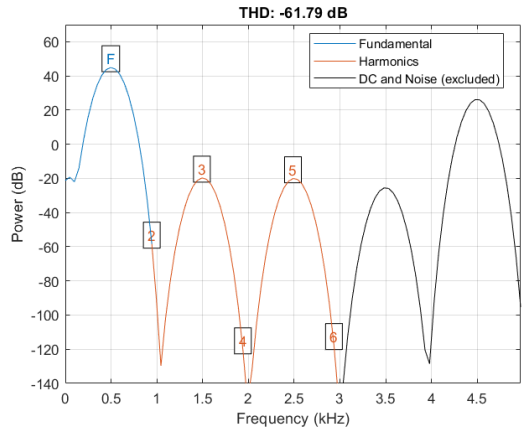

Figure 1.3: THD with Floating point arithmetic

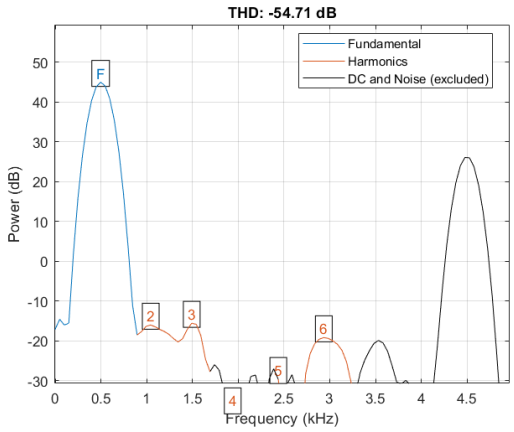

Figure 1.4: THD with 10 bit Fixed Point arithmetic

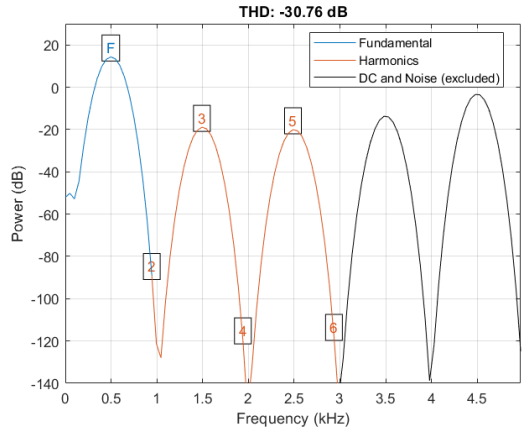

Figure 1.5: THD with Floating point arithmetic

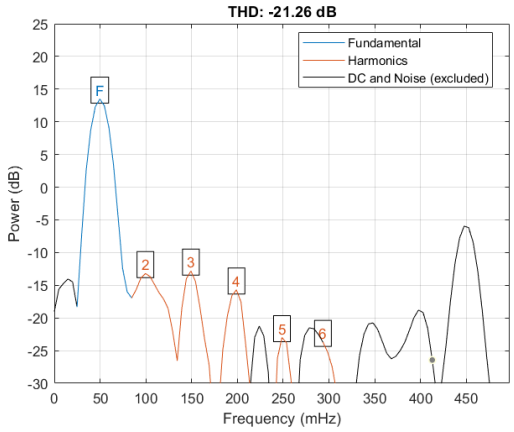

Figure 1.6: THD with 5 bit Fixed Point arithmetic

1.2 VLSI implementation

The purpose of this section is to present the implementation of the designed IIR filter, which in this case is an IIR filter with an accuracy of 10 bits and order of 1. In Figure 1.7 the filter interface is shown, where the parallelism n_b is set to 10 and since the order of the filter is 1, the necessary coefficients are a_0, a_1, b_0, b_1.

1.2.1 VHDL model

At the input and output of the filter there are registers to synchronize the data with the clock and two validation signals for the data, VIN and $VOUT$. In Figure 1.8 is shown the filter architecture, where the input and output registers have been neglected, the form used is the direct form II. The sampled input data enter from the $x[n]$ port, are processed and the results exit from the $y[n]$ port. The internal parallelism of the machine is different from the external one, to avoid the possibility of overflow with any combination of input data when summing, the precision has been extended to 12 bits by extending the sign to all input signals. The multiplier, on the other hand, has no problems of parallelism because it is possible to have sufficient precision by truncating the less significant bits at each multiplication and therefore having the same parallelism between input and output data. The

Github repository: here

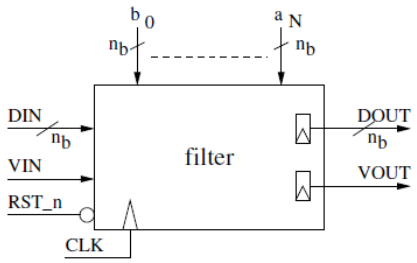

Figure 1.7: Filter pinout

internal register of the machine has an enable signal which is managed by the input data validation signal so that when invalid data is received, it does not sample keeping the correct output data.

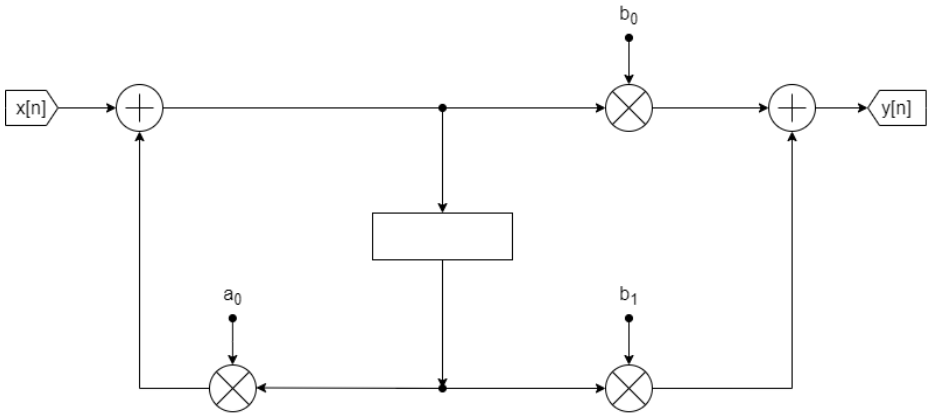

Figure 1.8: IIR filter architecture

1.2.2 Simulation

A solution with an input data generator and an output error checker has been adopted for the simulation of the circuit. A python script has been used to generate a sine waveform, which is processed in parallel by the designed circuit and the ideal model described in C. The output results are written on two files that are processed by an additional python script that computes the error as the difference between the output of the DUT and the C model. In Figure 1.9 is shown the wave with DUT signals. Note that when the VIN signal becomes valid the circuit starts to sample the input data and after a latency of 2 clock periods the results are ready to output, so the $VOUT$ signal is asserted.

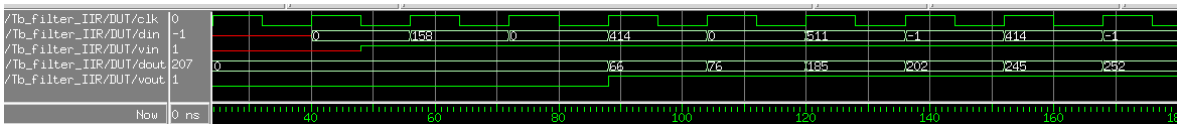

Figure 1.9: Start of the simulation

In Figure 1.10 the correct functioning can be noticed even when VIN is negated, in fact the data at the output of the filter, d_out, remain unchanged when this situation occurs because the circuit stops sampling the input data. When VIN returns to 1, after a latency of 1 clock period, the output data changes again.

Github repository: here

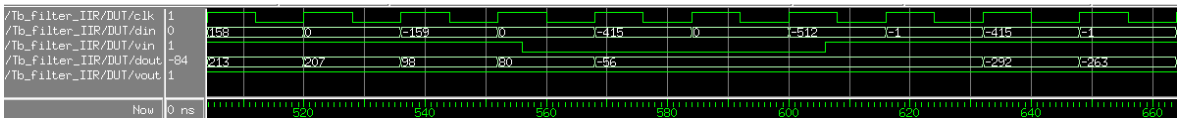

Figure 1.10: Simulation of the VIN signal transition

After verification of the correct timing of the circuit, the numerical results produced were checked. In Table 1.1 is shown an extract of the output data from the ideal model and the circuit designed with the same set of input data. It is noticeable that there is a perfect equality of results that confirms the correct operation of the designed filter.

Input	DUT output	C model output	Error
511	252	252	0
-1	213	252	0
414	207	207	0
-1	98	98	0
158	80	80	0
-1	-56	-56	0
-159	-77	-77	0
-1	-188	-188	0
-415	-206	-206	0
-1	-250	-250	0

Table 1.1: Results of the simulation

1.2.3 Logic Synthesis

The synthesis of the circuit was done with the Synopsys software. The first objective is to determine the maximum frequency that will allow the correct operation of the circuit and its area. The standard port libraries provided have been used to automatically synthesize even the complex logical blocks such as adders and multipliers. In a preliminary phase a clock with period $T_{clk} = 10\,\text{ns} \pm 0.07\,\text{ns}$ has been set. The result of the synthesis shows a slack of $+5.14\,\text{ns}$; the fact that the slack is positive implies that the clock constraints set have been met with the synthesis obtained. To evaluate the maximum operating frequency of the circuit, a clock period of 0 has been set, so theoretically the negative slack obtained should correspond to the minimum necessary clock period. However, this is only partially true, since setting the clock period to the new value will still result in a negative slack. This behavior is due to the fact that the synthesizer changes the structure of the internal slack according to the constraint provided on the clock as you can see from the data on the area. It was necessary to iterate this procedure several times to obtain a slack equal to 0 and therefore the maximum operating frequency of the circuit. In Table 1.2 a summary of the results obtained is shown.

T_{CLK} (ns)	slack (ns)	area $(\mu\text{m})^2$
10	5.14	1991
0	-3.13	2491
4.10	0	2187
16.4	11.54	1991

Table 1.2: Results of timing report

Github repository: here

The second goal is to find area and power consumption by setting $T_{CLK} = 4T_{min}$. For the area you can refer to Table 1.2. For the power computation, a record of the switching activity has been generated through a Modelsim simulation and saved on a vcd file, subsequently converted in saif file. The presence of these data serves during the calculation of the correct power consumption in the Synopsys environment. The results are shown in Figure 1.11.

Power Group	Internal Power	Switching Power	Leakage Power	Total Power
register (34.31%)	17.3502	1.3272	2.9308e+03	21.6082
sequential (0.00%)	0.0000	0.0000	0.0000	0.0000
combinational (65.69%)	1.2971	1.0033	3.9079e+04	41.3799
Total	18.6473 uW	2.3305 uW	4.2010e+04 nW	62.9881 uW

Figure 1.11: Power Report

It can be seen that the power is divided between the registers and the combinatorial part with a preponderance of this second contribution, 65%, due to the presence of 3 multipliers and 2 adders in the designed architecture.

1.2.4 Place & Route

The last operation is the place and route of the circuit using the Innovus software. The netlist generated by Synopsys with $T_{CLK} = 4T_{min}$ and the standard libraries have been used as a starting point. After the various required steps the final circuit shown in Figure 1.12 has been obtained.

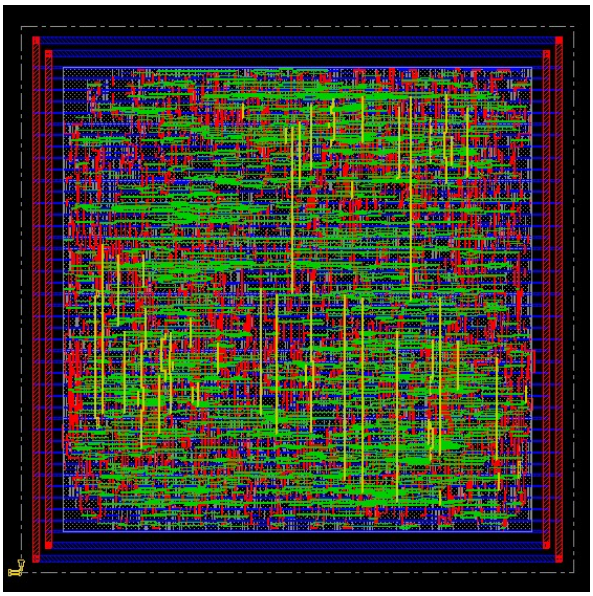

Figure 1.12: Resulting layout

The product layout has an area of 1959 μm^2, which is in agreement with the estimated Synopsys' estimate shown in Table 1.2, with a total of 986 cells and 2455 gates. Then the timing analysis was launched to verify that the timing constraints were correct, then the connectivity and geometry was

verified. Finally, using the switching activity calculated with Modelsim, the power consumption has been re-estimated. The results obtained are shown in Figure 1.13.

Power Units = 1mW

Group	Internal Power	Switching Power	Leakage Power	Total Power	Percentage (%)
Sequential	0.04046	0.01858	0.002993	0.06203	5.438
Macro	0	0	0	0	0
IO	0	0	0	0	0
Combinational	0.5293	0.5121	0.03724	1.079	94.56
Clock (Combinational)	0	0	0	0	0
Clock (Sequential)	0	0	0	0	0
Total	0.5698	0.5307	0.04024	1.141	100

Figure 1.13: Post place & route power report

The power values obtained are in agreement with those obtained by Synopsys in Figure 1.11. Moreover, since the calculation at this level is much more accurate as it takes into account the physical layout of the circuit, it can be seen that in fact the consumption of the combinatorial part is prevalent.

1.3 Advanced architecture development

In this section is analyzed the design of the same IIR filter seen previously but with an advanced architecture, developed with the J-look-ahead technique. The basic equation of the filter is the following:

$$y[n] = b_0x[n] + b_1x[n-1] + a_1y[n-1]$$

The J-look-ahead technique involves rewriting the equation by replacing the term $y[n-1]$ by evaluating it as a function of $y[n-2]$.

$$y[n-1] = b_0x[n-1] + b_1x[n-2] + a_1y[n-2]$$

Then it is possible to replace the result obtained in the starting equation:

$$y[n] = b_0x[n] + b_1x[n-1] + a_1(b_0x[n-1] + b_1x[n-2] + a_1y[n-2])$$

$$y[n] = b_0x[n] + (b_1 + a_1)x[n-1] + a_1b_1x[n-2] + {a_1}^2y[n-2]$$

While keeping the classical architecture of the IIR direct form II filter, the resulting equation can be implemented through the scheme of a second-order filter.

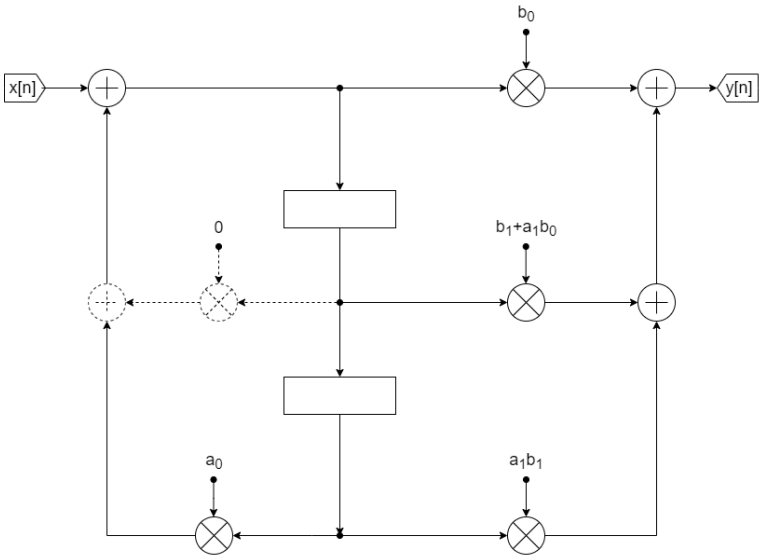

Figure 1.14: IIR filter with J-look-ahead technique

1.3.1 Design optimization

This structure can be, as opposed to the previous version, improved in terms of performance. The critical path of this scheme consists of 4 combinatorial blocks, two multipliers and two adders, though by means of the appropriate transformations it is possible to reduce the longest combinatorial path. In Figure 1.15 it has been identified the first cut-set that allows to move the register by changing the length of the critical path from 4 to 3.

In Figure 1.16 a feed-forward cut-set has been identified where it is possible to insert a pipeline stage on all three branches. In this way the combinatorial path including the b_0 multiplier is no longer part of the critical path. The longest combinatorial path includes the multiplier for the term $b_1 + a_1b_0$, the intermediate adder and the final adder. A further transformation allows to move the register from one side of the multiplier to the other without altering the behavior of the circuit, as in Figure 1.17.

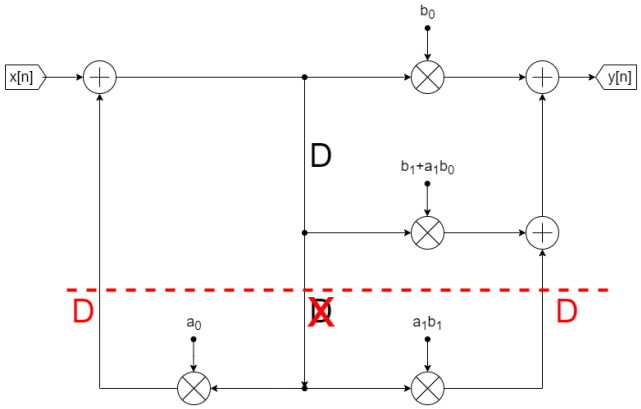

Figure 1.15: Advanced architecture with 1^{st} transformation

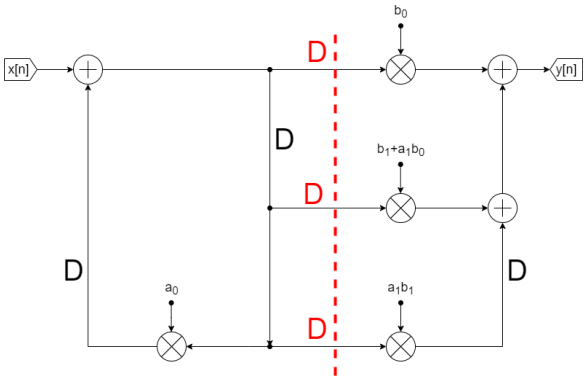

Figure 1.16: Advanced architecture with 2^{nd} transformation

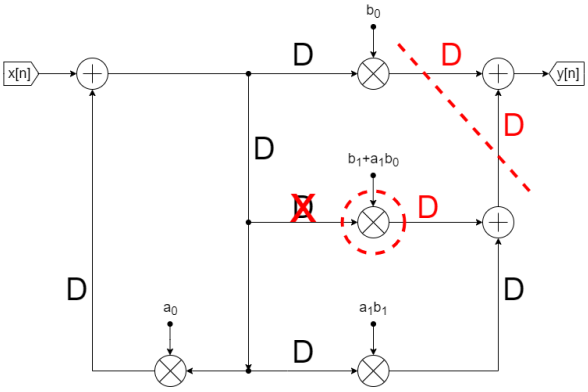

Figure 1.17: Advanced architecture with 3^{rd} transformation

The latter transformation in Figure 1.17 takes into account the feed-forward cut-set on the two input branches at the last adder and through the insertion of a pipeline stage bring the critical path to be equal to the delay of a single multiplier.

In Figure 1.18 the optimized architecture of the IIR filter can be observed, in which a pipeline stage has been inserted between each combinatorial block. These transformations allow the circuit to reach the maximum throughput at the expense of latency, which compared to the initial circuit has increased from $2\,T_{ck}$ to $4\,T_{ck}$.

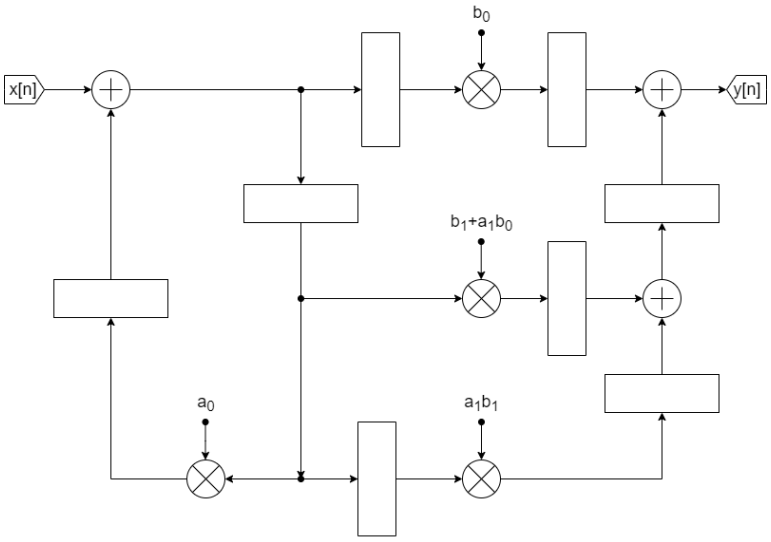

Figure 1.18: Optimized architecture of J-look-ahead IIR filter

1.3.2 C model

For the circuit, a C model was created for Fixed point analysis; for this model a new script written ad hoc was used, which uses a vector of 7 components to simulate the pipelined architecture of the component. The script name is *J-look-ahead.c* and was included in the report delivery.The output signal from the filter was then analyzed on Matlab to evaluate its effectiveness. Then it was evaluated the Total Harmonic Distortion that, as you can observe from the Figure 1.19, remained perfectly within the specifications.

1.3.3 Comparing Direct form II and J-Look-Ahead

What will be presented now is an analysis of the data obtained from the two filters, with the aim of demonstrating their equivalence in behavior despite the great difference between the two topologies. In particular they will be examined in Matlab environment:

- Initial signal values
- Average of the two signals
- Variance of the two signals

Notice that the IIR look-ahead filter topology in Figure 1.18 has two pipeline stages between the input and output signal, this implies that the signal has a higher latency than the Direct form II topology, which corresponds exactly to two clock cycles. The table in Figure 1.20 shows the results of the two implementations.

As the J-look-Ahead THD graph has anticipated, the two signals are not perfectly identical in the values and although the overall trend may be very similar, one may be interested in how different it actually is, so an analysis of the output signals from the two qualitative and quantitative filters is necessary at this point. Afterwards, the average and variance values of the two signals are represented:

Parameter	Direct Form II	J-Look-Ahead
Mean	−0.8308	−2.2537
Variance	23.6312	32.9303

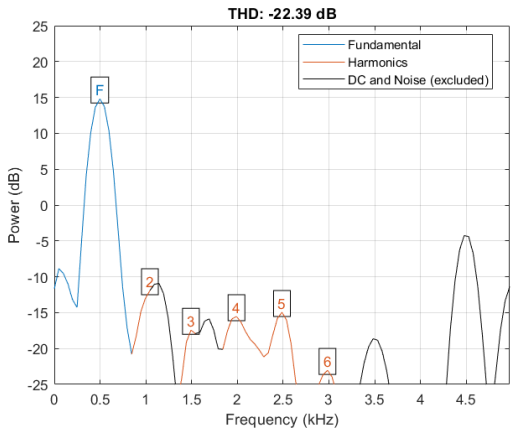

Figure 1.19: THD of IIR J-Look-Ahead filter

Direct Form II	J-Look-Ahead
0	0
1	0
1	0
4	1
5	1
7	5
6	5
4	6
4	6

Figure 1.20: Results of the two implementations compared

The two signals are not perfectly identical, and in fact they present different averages and variance, but they have the same order of magnitude and values consistent with the system under analysis. For a finer analysis it is necessary to use Allan's variance to measure how much the noise degrades the performance and how much the oscillatory behavior of the output is more or less stable. In Figure 1.21 the variance of the two signals is represented and in Figure 1.22 the difference between the two variance to assess how much the stability of Direct Form II differs from that of J-Look-Ahead, both graphs were generated using the Matlab function *allanvar()*:

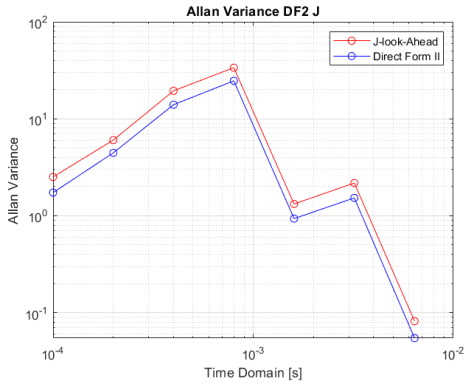

Figure 1.21: Allan variance of the Direct Form II and J-look-ahead IIR Filter

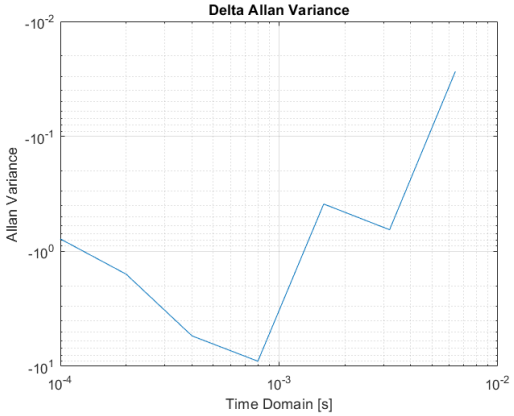

Figure 1.22: Difference of the 2 Allan variance

1.3.4 Simulation

For the simulation of the circuit has been used the same test-bench since the external entity of the circuit has not been modified with respect to the previous implementation. In Figure 1.23 the timing with the signals of the DUT is shown.In this case when the VIN signal becomes valid 4 clock periods are required until the first result is available at the output; the explanation for this longer latency is due to the pipeline stages introduced in the circuit.

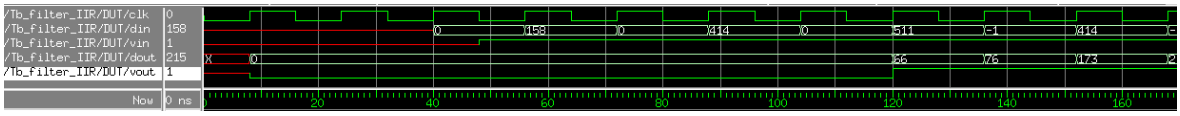

Figure 1.23: Start of the simulation

Finally in Figure 1.24 it is observed the correct operation of the circuit when VIN becomes low and then high. The internal registers do not sample when VIN is not asserted, so the output data does not change, keeping the output valid. When VIN returns to 1, only one clock period is required before the output changes value because the registers are already full.

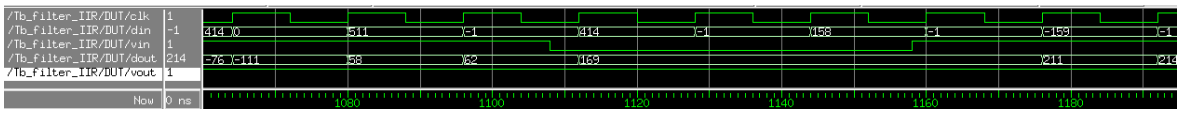

Figure 1.24: Simulation of VIN signal transition

Once the correct timing of the circuit has been verified, the output data has been compared with the corresponding C implementation of the realized model. With the same set of data used for the test-bench of the previous architecture, the same output results were obtained. You can refer to the data shown in Table 1.1 to have an outline of the results.

1.3.5 Logic Synthesis

Having verified the correct functioning of the filter with Modelsim, Synopsys has been used for synthesis. The objective is to calculate the maximum frequency of operation of the circuit and then setting $T_{CLK} = 4T_{min}$, compute the area and the dissipated power, using the data on switching activity produced by Modelsim. The results related to the clock period, the associated slack and the area are shown in Table 1.3.

T_{CLK} (ns)	slack (ns)	area $(\mu m)^2$
10	7.39	3200
0	-1.75	3834
2.15	0	3542
8.6	5.99	3200

Table 1.3: Results of timing report

Subsequently the net-list generated was used to determine the switching activity using Modelsim and obtain in this way an accurate estimation of the power consumption of the circuit. The results are shown in Figure 1.25.

Also in this case it is noted that the power consumption is mainly produced by the combinatorial part consisting of 3 adders and 4 multipliers. Comparing the results of the two architectures imple-

Power Group	Internal Power	Switching Power	Leakage Power	Total Power
register (28.36%)	58.3882	22.8767	1.0598e+04	91.8633
sequential (0.00%)	0.0000	0.0000	0.0000	0.0000
combinational (71.64%)	91.2519	84.2219	5.6595e+04	232.0690
Total	149.6401 uW	107.0986 uW	6.7194e+04 nW	323.9323 uW

Figure 1.25: Power Report

mented there are significant differences in terms of minimum period, in fact in the classic solution $T_{min} = 4.10\,\text{ns}$, while in the J-look-ahead solution $T_{min} = 2.15\,\text{ns}$, which corresponds to an increase in operating frequency of 48%. The performance enhancement is achieved thanks to the reduction of the critical path from 2 adders and 2 multipliers to only one multiplier. The improvement in performance, however, is paid for in terms of area, with an increase of 40%, and in terms of power. Comparing the data in Figure 1.11 and Figure 1.25 reveals an increase of 80% in total power; specifically, the power contributions that change significantly are those related to the switching power, 2.3 µW versus 107.1 µW, and to the internal power, 18.6 µW versus 149.6 µW. The increase of these contributions is given by the fact that in the second architecture there are more logical blocks.

1.3.6 Place & Route

The Place & Route with the innovus software has been carried out following the previous procedure. The generated circuit is the one shown in Figure 1.26, it has an area equal to $3186\,\mu\text{m}^2$ according to the one obtained from the Synopsys report, with a total of 1588 cells and 3993 gates. The values obtained are plausible if compared to those of the previous architecture because in this case there are more physical elements (registers, summers and multipliers) and consequently the need for more area to allocate all the necessary gates is expected.

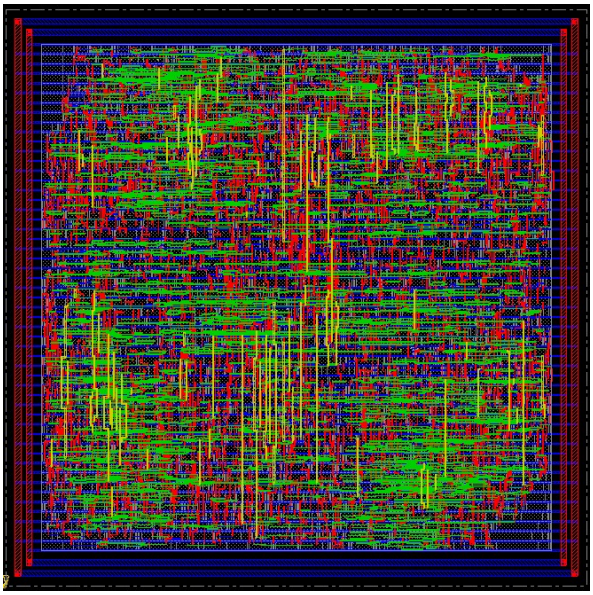

Figure 1.26: Resulting layout

Subsequently it was launched the timing analysis, the verification of connectivity and geometry, which showed the absence of any violation. Starting from the switching activity data obtained from Modelsim, an estimation of the power has been performed. The results are shown in Figure 1.27, also in this case the results obtained are in line with those obtained from the Synopsys power report.

Power Units = 1mW

Group	Internal Power	Switching Power	Leakage Power	Total Power	Percentage (%)
Sequential	0.1707	0.06718	0.01054	0.2484	21.6
Macro	0	0	0	0	0
IO	0	0	0	0	0
Combinational	0.467	0.381	0.05383	0.9018	78.4
Clock (Combinational)	0	0	0	0	0
Clock (Sequential)	0	0	0	0	0
Total	0.6377	0.4482	0.06437	1.15	100

Figure 1.27: Post place & route power report