

Implementação em Hardware de um Filtro CIC

Um Filtro de Média Móvel

Valmir F. Silva

Universidade Federal de Campina Grande - UFCG

09/10/2024

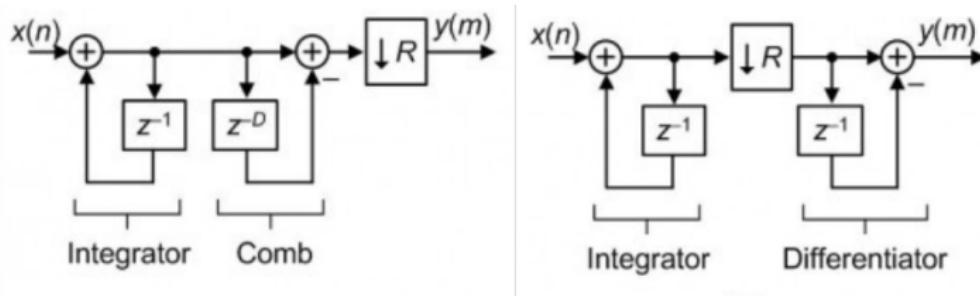


Sumário

- 1 Filtros Cascaded Integrator Comb - CIC
- 2 Descrição do Hardware em SystemVerilog
- 3 Instânciamento Automático de Módulos
- 4 Conversor DAC Modulado em PWM
- 5 Simulação - DCA PWM
- 6 Simulação - CIC
- 7 Teste em FPGA - CIC

Filtros Cascaded Integrator Comb - CIC

Figure: Filtro CIC com Decimação



Fonte: [DSP Related, 2024.](#)

Descrição do Hardware em SystemVerilog

Figure: Descrição em SystemVerilog das células base comb e integrador

```
comb.sv
module comb #(parameter WIDTH = 8)(
    input logic          clock ,
    input logic          ena   ,
    input logic          reset ,
    input logic [WIDTH-1:0] x_in ,
    output logic [WIDTH-1:0] y_out
);
    logic [WIDTH-1:0] regDelay;
    always_ff@(posedge clock, negedge reset)begin
        if(!reset)begin
            y_out <= 0;
            regDelay <= 0;
        end
        else begin
            regDelay <= x_in != 0 ?      x_in     : regDelay ;
            y_out  <= x_in != 0 ? x_in - regDelay : y_out ;
        end
    end
endmodule
```

```
integrator.sv
module integrator #(parameter WIDTH = 32)(
    input logic          clock ,
    input logic          ena   ,
    input logic          reset ,
    input logic signed [WIDTH-1:0] x_in ,
    output logic signed [WIDTH-1:0] y_out
);
    logic signed [WIDTH-1:0] regDelay;
    always@(posedge clock, negedge reset)begin
        if(!reset)begin
            y_out      <= 0;
            regDelay   <= 0;
        end
        else begin
            y_out      <= x_in + y_out;
        end
    end
endmodule
```

Fonte: própria, 2024.

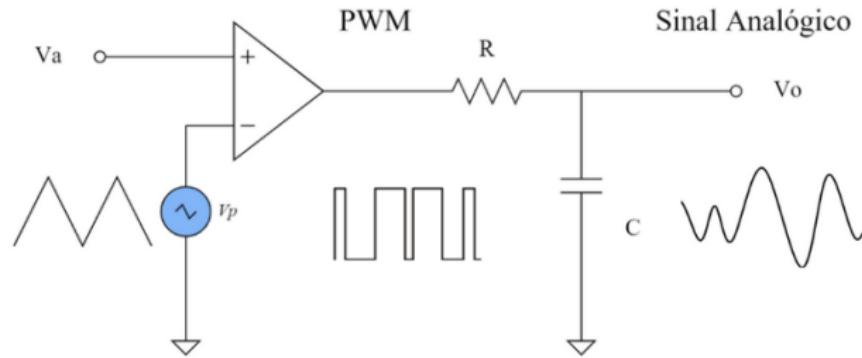
Instânciamento Automático de Módulos

Figure: Instânciador de Células

```
% filtro_em_cascadas.v
1  module filter#(parameter WIDTH = 8, SIZE = 16)(
2      input logic           clock      ,
3      input logic           reset      ,
4      input logic [WIDTH-1:0] x_in      ,
5      output logic [WIDTH-1:0] y_out     );
6
7      logic ena;
8      assign ena = 1;
9
10     logic [WIDTH-1:0]wire_out_integ[SIZE-1:0];
11     logic [WIDTH-1:0]wire_out_comb[SIZE-1:0];
12     logic [WIDTH-1:0]integ_wire_comb;
13     logic [4:0] counter;
14
15     always_ff@(posedge clock, negedge reset)begin
16         if(!reset)counter <= 0;
17         else if(counter == SIZE-1)counter <= 0;
18         else counter <- counter+1;
19     end
20
21     always_comb      integ_wire_comb <- !counter ? wire_out_integ[0]: 0;
22     assign          y_out = counter == SIZE-1 ? wire_out_comb[0]/SIZE : y_out;
23
24     genvar i; generate begin
25         for(i = 1; i < SIZE; i=i+1)begin:gen
26             integrator integ_n(
27                 .clock(clock),
28                 .reset(reset),
29                 .ena(ena),
30                 .x_in(wire_out_integ[i-1]),
31                 .y_out(wire_out_integ[i]));
32
33             comb comb_n(
34                 .clock(clock),
35                 .reset(reset),
36                 .ena(ena),
37                 .x_in(wire_out_comb[i-1]),
38                 .y_out(wire_out_comb[i]));
39
40         end end endgenerate
41         integrator integ_n0(
42             .clock(clock),
43             .reset(reset),
44             .ena(ena),
45             .x_in(x_in),
46             .y_out(wire_out_integ[0]));
47
48         comb comb_n0(
49             .clock(clock),
50             .reset(reset),
51             .ena(ena),
52             .x_in(x_in),
53             .y_out(wire_out_integ[0]));
54
55     end
56
57     assign y_out = wire_out_integ[0];
58
59 endmodule
```

Conversor DAC Modulado em PWM

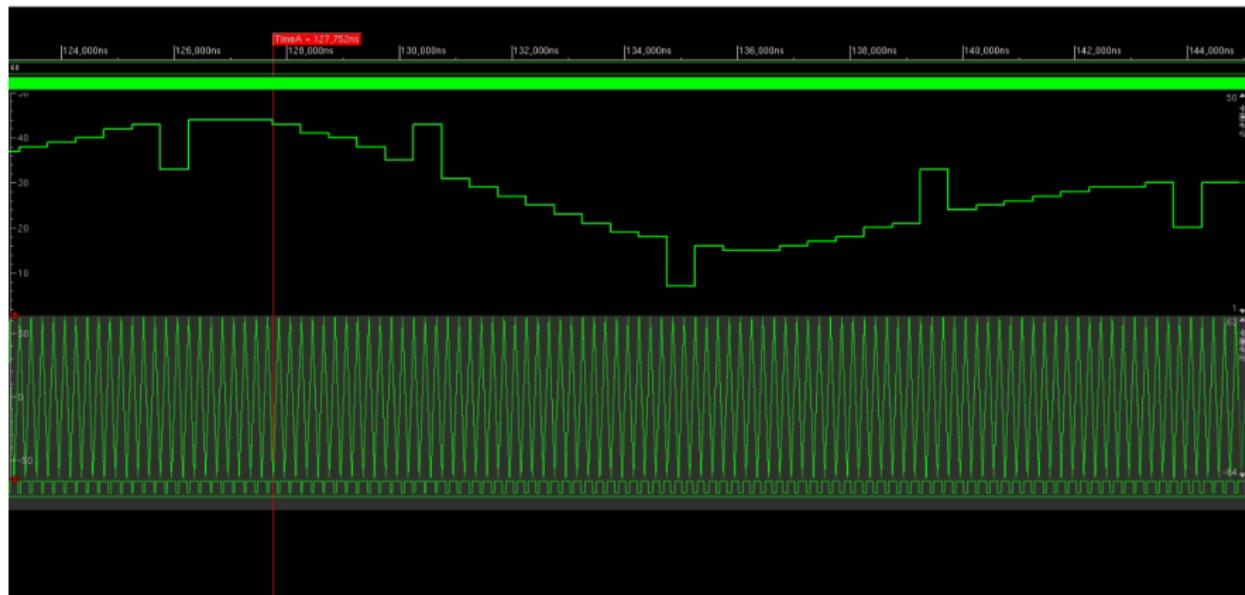
Figure: Ilustração do processo de Conversão digital Analógico



Fonte: própria, 2024.

Simulação - DCA PWM

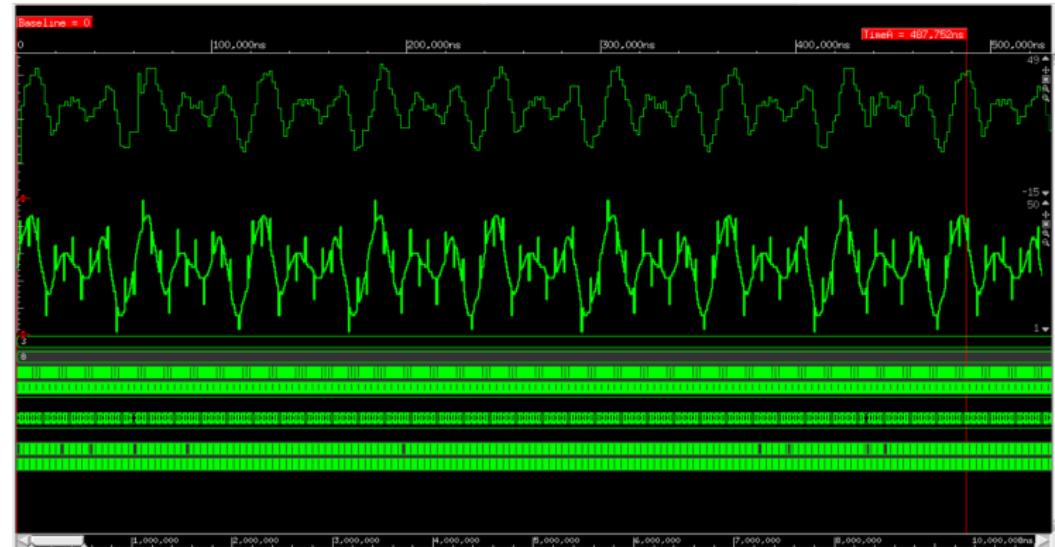
Figure: Simulação em Software do DCA



Fonte: própria, 2024.

Simulação - CIC

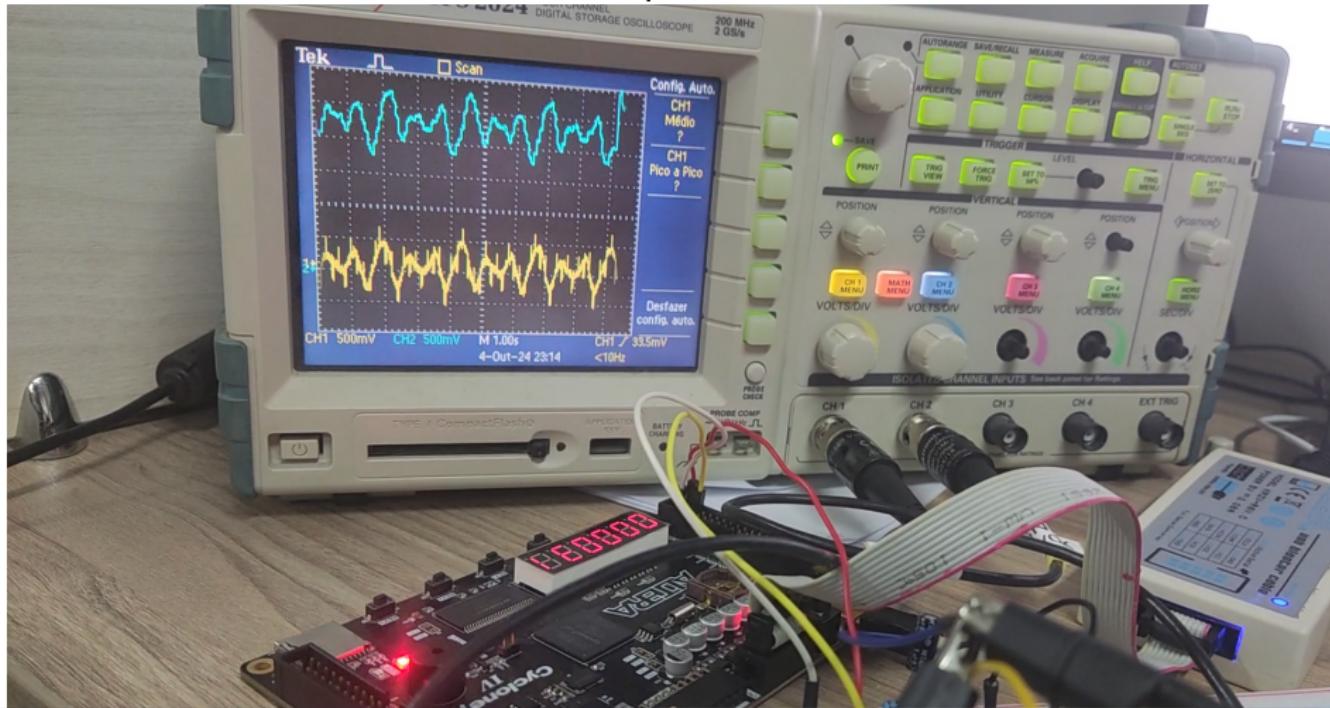
Figure: Simulação em Software do Filtro



Fonte: própria, 2024.

Teste em FPGA - CIC

Figure: Teste em FPGA



Obrigado!