5位Fir高通滤波器

成员：*** *** *** *** *** *** B415

2017年11月

目录
CONTENTS

01.Fir滤波器简介

- 概念
- 特点
- 结构

Fir滤波器概念

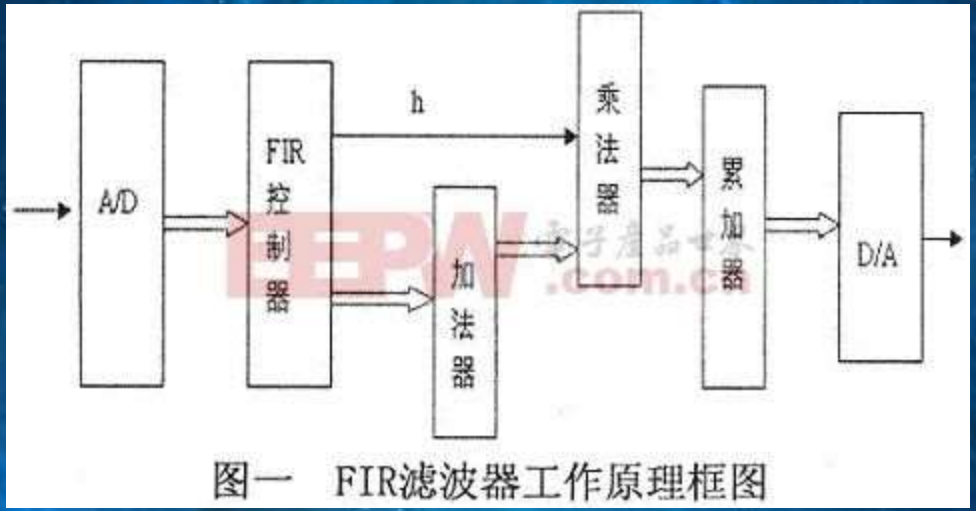
FIR(Finite Impulse Response)滤波器：有限长单位冲激响应滤波器，又称为非递归型滤波器，是数字信号处理系统中最基本的元件。

它可以在保证任意幅频特性的同时具有严格的线性相频特性，同时其单位抽样响应是有限长的，因而滤波器是稳定的系统。

因此，FIR滤波器在通信、图像处理、模式识别等领域都有着广泛的应用。

Fir滤波器特点

在信号进入FIR滤波器前，首先要将信号通过A/D器件进行模数转换，使之成为8bit的数字信号，一般可用速度较高的逐次逼进式A/D转换器。



图一　FIR滤波器工作原理框图

不论采用乘累加方法还是分布式算法设计FIR滤波器，滤波器输出的数据都是一串序列，要使它能直观地反应出来，还需经过数模转换，因此由FPGA构成的FIR滤波器的输出须外接D/A模块。

FPGA有着规整的内部逻辑阵列和丰富的连线资源，特别适合于数字信号处理任务，相对于串行运算为主导的通用DSP芯片来说，其并行性和可扩展性更好，利用FPGA乘累加的快速算法，可以设计出高速的FIR数字滤波器。

直接型Fir滤波器结构

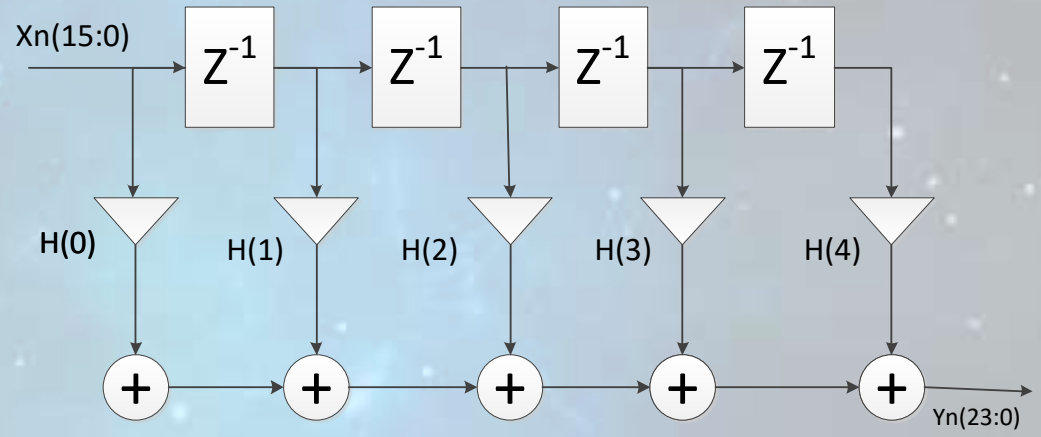
长度为M的因果有限冲激响应滤波器由传输函数H（z）描述：

$$H(z)=\sum_{k=0}^{M-1}h(k)z^{-k}$$

它是次数为M-1的z^{-1}的一个多项式。在时域中，上述有限冲激响应滤波器的输入输出关系为：

$$y(n)=\sum_{k=0}^{M-1}h(k)\,x(n-k)$$

其中y（n）和x（n）分别是输出和输入序列。



由Matlab得出五位Fir滤波器的Z域函数为：

$$H(z)=0.25\times(1+3.5z^{-1}+4z^{-2}+3.5z^{-3}+z^{-4})$$

02.各模块设计分析

- 模块种类
- 寄存器
- 乘法器
- 加法器
- 顶层模块

设计模块

寄存器用于寄
二值代码，只要
具有置1、置0的
可。

本设计中使用
复位reset_n端
器，当reset_n=
出信号q_out=d_
reset_n=0且上
到达时q_out=0。

从资源和速度
常系数乘法运算
位相加来实现。

本设计采用加
法器兼顾了资源
，将每个乘数例
个数组，然后移
得出乘积，这样
法运算可以一个
完成。

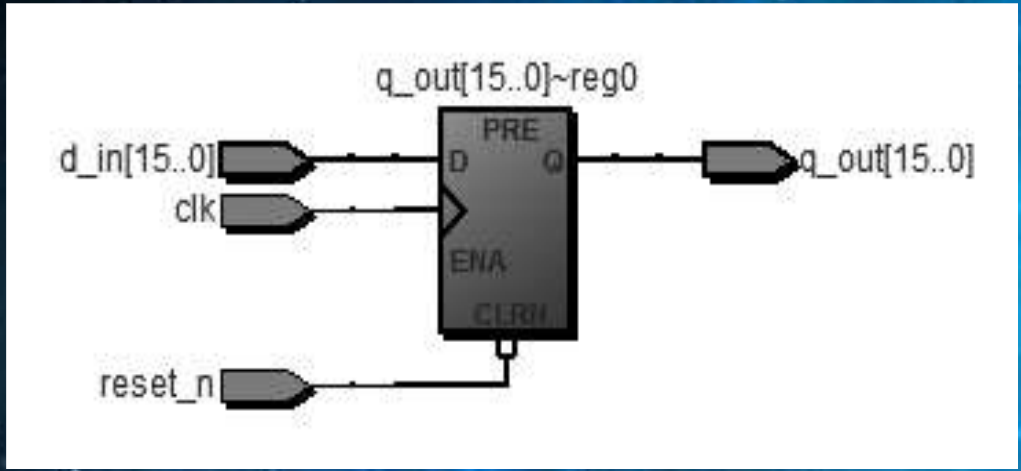
由于本设计只涉及到
相加，而没有减
以本加法器实现
位无符号数的相
。

即将输入的两
时钟脉冲到来时相加运
算，输出结果。

顶层模块

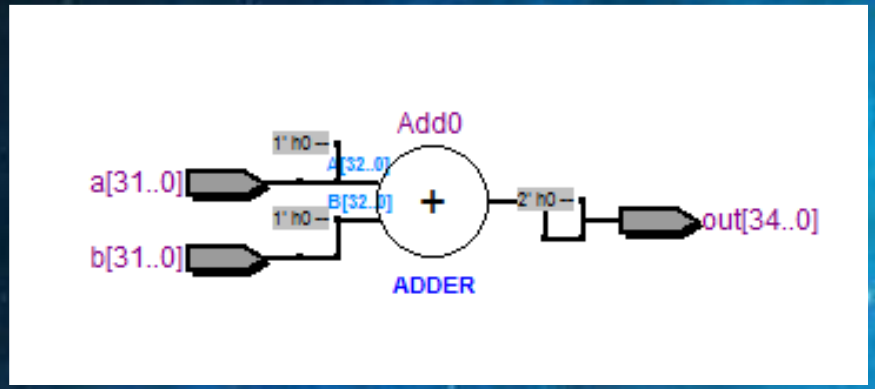
将3个16位寄存器，4个16位乘法器，3个32位加法器在顶层模块分别实例化。

寄存器



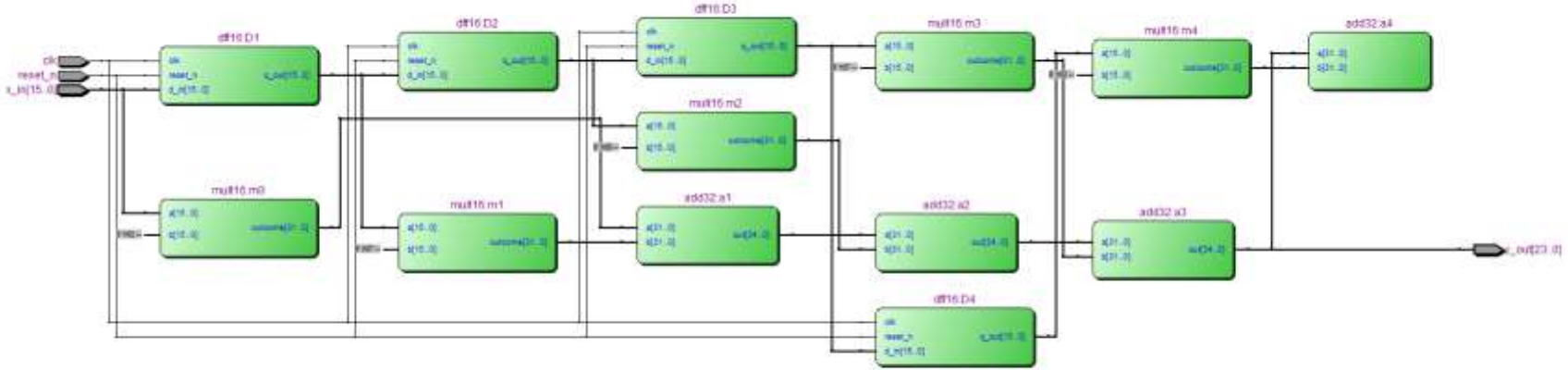

```
module Dff16(reset_n,clk,d_in,q_out);
input reset_n,clk;
input [15:0]d_in;
output reg [15:0]q_out;

always @(posedge clk or negedge reset_n)
  begin
    if(!reset_n)
       q_out<=16'h0;
      else
       q_out<=d_in;
  end
endmodule
```

加法器




```
module add32(a,b,out);
input [31:0]a,b;
output [34:0]out;

assign out=a+b;

endmodule
```

顶层模块


clk
reset_n
x_in[15..0]
dff16:D1
dff16:D2
dff16:D3
mult16:m3
mult16:m4
add32:a4
mult16:m0
mult16:m1
mult16:m2
add32:a1
add32:a2
add32:a3
dff16:D4
y_out[23..0]

顶层模块

此处乘以256，以便进行截尾的小数运算。

```
module Fir(reset_n,clk,x_in,y_out);

input reset_n,clk;
input [15:0]x_in;
output [23:0]y_out;

wire [15:0]q1,q2,q3,q4;
wire [31:0]mout0,mout1,mout2,mout3,mout4;
wire [34:0]aout1,aout2,aout3,aout4;

dff16
D1(.reset_n(reset_n),.clk(clk),.d_in(x_in),.q_out(q1)),
        D2(.reset_n(reset_n),.clk(clk),.d_in(q1),.q_out(q2)),
        D3(.reset_n(reset_n),.clk(clk),.d_in(q2),.q_out(q3)),
        D4(.reset_n(reset_n),.clk(clk),.d_in(q3),.q_out(q4));
```

```
mult16
    m0(.outcome(mout0),.a(x_in),.b(16'h004
0)),
    m1(.outcome(mout1),.a(q1),.b(16'h00e0)
),

m2(.outcome(mout2),.a(q2),.b(16'h0100)),
    m3(.outcome(mout3),.a(q3),.b(16'h00e0)
),
    m4(.outcome(mout4),.a(q4),.b(16'h0040)
);

add32  a1(.a(mout0),.b(mout1),.out(aout1)),
       a2(.a(aout1),.b(mout2),.out(aout2)),
       a3(.a(aout2),.b(mout3),.out(aout3)),
       a4(.a(aout3),.b(mout4),.out(aout4));

assign y_out=aout4[23:0];
```

乘法器

```
module
mult16(outcome,a,b);
input [15:0]a,b;
output
```


```
wire [3
wire [2
wire [2
wire [2
wire [2
wire [2
wire [2
wire [2
wire [2
wire [2
wire [1
wire [18:0] temp12;
wire [17:0] temp13;
wire [16:0] temp14;
wire [15:0] temp15;
```

```
assign temp15=mult16x1(a,b[0]);
assign temp14=((mult16x1(a,b[1]))<<1);
assign temp13=((mult16x1(a,b[2]))<<2);
assign temp12=((mult16x1(a,b[3]))<<3);
                (a,b[4]))<<4);
                (a,b[5]))<<5);
                a,b[6]))<<6);
                a,b[7]))<<7);
                a,b[8]))<<8);
                a,b[9]))<<9);
                a,b[10]))<<10);
                a,b[11]))<<11);
                a,b[12]))<<12);
                a,b[13]))<<13);
                a,b[14]))<<14);
                a,b[15]))<<15);
                temp2+temp3+temp4+temp5+
temp6+temp7+temp8+temp9+temp10+temp11+temp12+
temp13+temp14+temp15)/256;

endmodule
```

乘法器

顶层模块中

```
mult16  m0(.outcome(mout0),.a(x_in),.b(16'h0040)),
        m1(.outcome(mout1),.a(q1),.b(16'h00e0)),
        m2(.outcome(mout2),.a(q2),.b(16'h0100)),
        m3(.outcome(mout3),.a(q3),.b(16'h00e0)),
        m4(.outcome(mout4),.a(q4),.b(16'h0040));
```

此处为公式中系数乘以256进行8位的位扩展转变为整数。

$$H(z) = 0.25 \times (1 + 3.5z^{-1} + 4z^{-2} + 3.5z^{-3} + z^{-4})$$

即式中的0.25，0.875，1分别转换为64，224，256

乘法器中

```
assign
outcome=(temp0+temp1+tem
p2+temp3+temp4+temp5+te
mp6+temp7+temp8+temp9+t
emp10+temp11+temp12+tem
p13+temp14+temp15)/256;
```

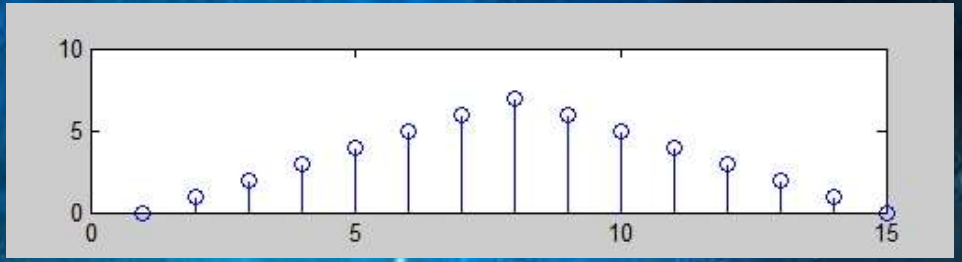
此处再除以256，截除后8位，在10进制中为去除小数，保留整数，从而实现利用乘法器进行小数运算。

03.构造仿真测试数据

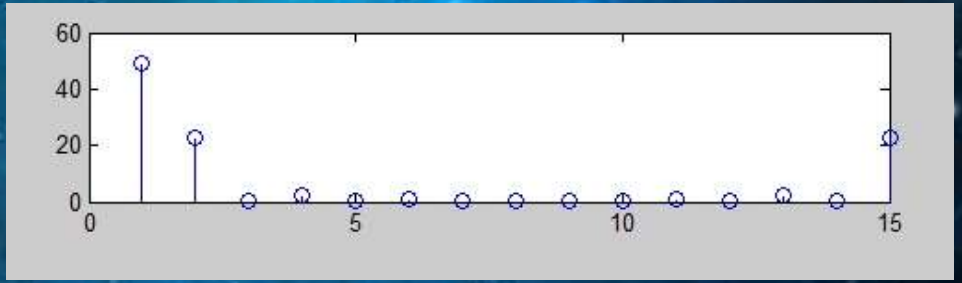
- 三角波数据
- 激励测试块

用Matlab产生三角波

```
n=1:7;
x1(n)=n-1;
n=8:15;
x1(n)=15-n;
x1=x1*100;
x2=uint16(x1);
fid=fopen('/tri.txt','wt+');
fprintf(fid,'%x\n',x2);
fclose(fid);

n=1:15;
subplot(2,1,1);
stem(n,x1);
subplot(2,1,2);
x1fft=fft(x1);
stem(n,abs(x1fft));
```



三角波时域图



三角波频域图

tri.txt

0
64
c8
12c
190
1f4
258
2bc
258
1f4
190
12c
c8
64
0

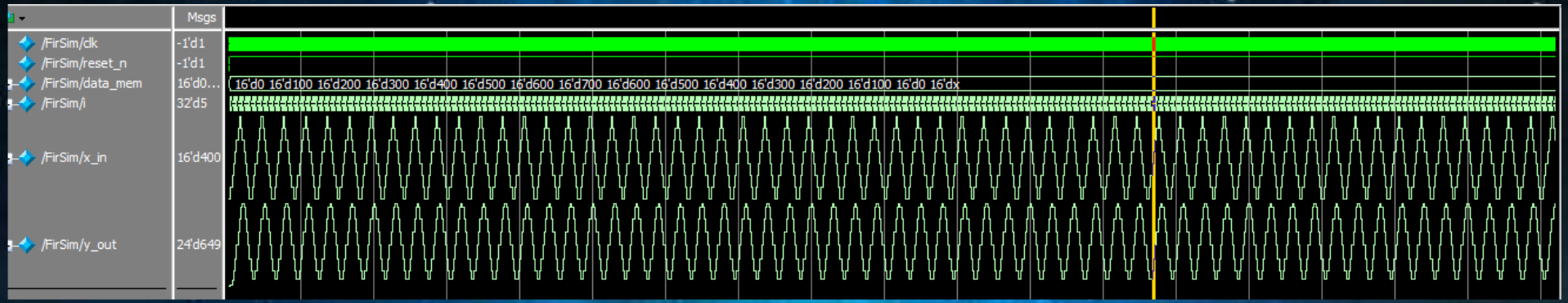
测试激励块

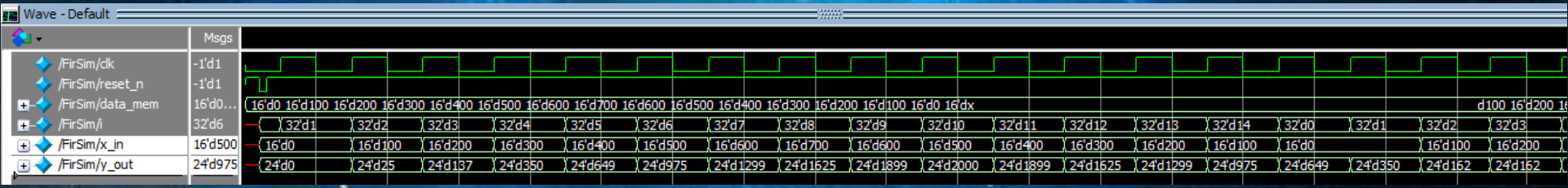
```
`timescale 1ns/1ns
`define clock 50

module test;

 reg clk,reset_n;
 reg [15:0]x_in;
 reg [15:0]data_mem[0:15];
 integer i;
 wire [23:0]y_out;

 always #`clock clk=~clk;

 initial
   begin
     clk=0;
     reset_n=1;
   end
 initial
   begin
     $readmemh("tri.txt",data_mem);
   end

   always @(posedge clk or negedge reset_n)
     begin
       if(!reset_n) begin
         x_in<=15'b0;
         i<=0;
         end
       else if(i<=13) begin
         x_in<=data_mem[i];
         i<=i+1;
         end
     end

   Fir fir(.reset_n(reset_n),.clk(clk),.x_in(x_in),.y_out(y_out));

endmodule
```

04.仿真结果及分析

- 仿真波形
- 数据验证

仿真波形


Msgs
/FirSim/clk
/FirSim/reset_n
/FirSim/data_mem
/FirSim/i
/FirSim/x_in
/FirSim/y_out
-1'd1
-1'd1
16'd0...
32'd5
16'd400
24'd649
16'd0 16'd100 16'd200 16'd300 16'd400 16'd500 16'd600 16'd700 16'd600 16'd500 16'd400 16'd300 16'd200 16'd100 16'd0 16'dx


Wave - Default
Msgs
/FirSim/clk
/FirSim/reset_n
/FirSim/data_mem
/FirSim/i
/FirSim/x_in
/FirSim/y_out
-1'd1
-1'd1
16'd0...
32'd6
16'd500
24'd975
16'd0 16'd100 16'd200 16'd300 16'd400 16'd500 16'd600 16'd700 16'd600 16'd500 16'd400 16'd300 16'd200 16'd100 16'd0 16'dx
32'd1 32'd2 32'd3 32'd4 32'd5 32'd6 32'd7 32'd8 32'd9 32'd10 32'd11 32'd12 32'd13 32'd14 32'd0 32'd1 32'd2 32'd3
16'd0 16'd100 16'd200 16'd300 16'd400 16'd500 16'd600 16'd700 16'd600 16'd500 16'd400 16'd300 16'd200 16'd100 16'd0 16'd100 16'd200
24'd0 24'd25 24'd137 24'd350 24'd649 24'd975 24'd1299 24'd1625 24'd1899 24'd2000 24'd1899 24'd1625 24'd1299 24'd975 24'd649 24'd350 24'd162 24'd162

数据验证

原始数据	0	100	200	300	400	500	600	700	600	500	400	300	200	100	0	0	100	200	300
0 0.25	0	25	50	75	100	125	150	175	150	125	100	75	50	25	0	0	25	50	75
1 0.875		0	87.5	175	262.5	350	437.5	525	612.5	525	437.5	350	262.5	175	87.5	0	0	87.5	175
2 1			0	100	200	300	400	500	600	700	600	500	400	300	200	100	0	0	100
3 0.875				0	87.5	175	262.5	350	437.5	525	612.5	525	437.5	350	262.5	175	87.5	0	0
4 0.25					0	25	50	75	100	125	150	175	150	125	100	75	50	25	0
计算结果	0	25	137.5	350	650	975	1300	1625	1900	2000	1900	1625	1300	975	650	350	162	162	350
仿真结果	0	25	137	350	649	975	1299	1625	1899	2000	1899	1625	1299	975	649	350	162	162	350

谢　谢　收　看