

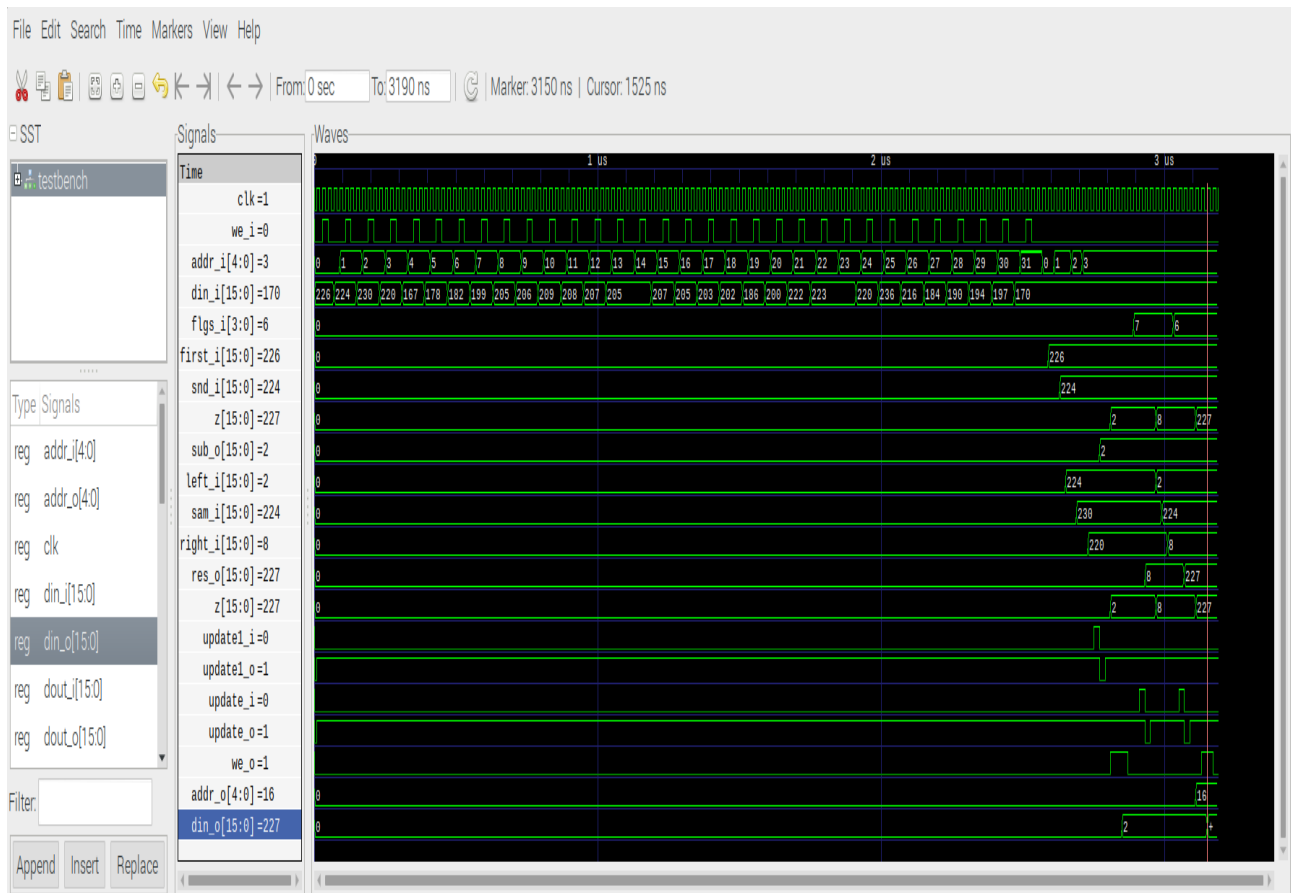
Signals `first_i` and `snd_i` are  $(\text{first\_i} - \text{snd\_i})$  with the result in signal `sub_o`.

This stored in the inst of ram used as the output.

The hi pass dwt uses signals from address 1 (`left_i`), 2 (`sam_i`), and 3 (`right_i`) of inst ram used as the input.

The signal `sub_o` in red is stored at `addr_o` 0. This signal is also the `left_i` of lo pass dwt filter (`flgs` = 6).

The signal `res_o` of the hi pass dwt filter (`flgs` = 7) is the `right_i` of lo pass dwt filter (`flgs` = 6).

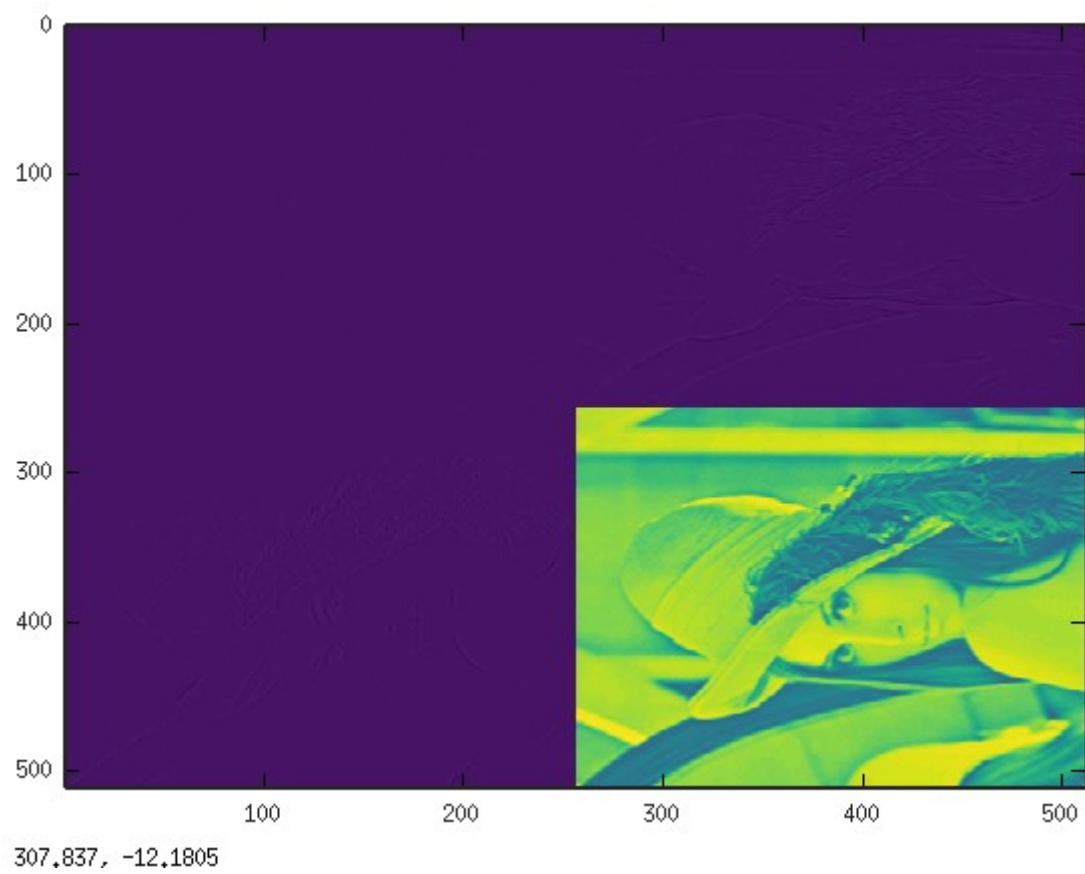


Hi & Lo pass dwt filters using the `lifting_step`.

This example is of 512 x 512 red.pgm



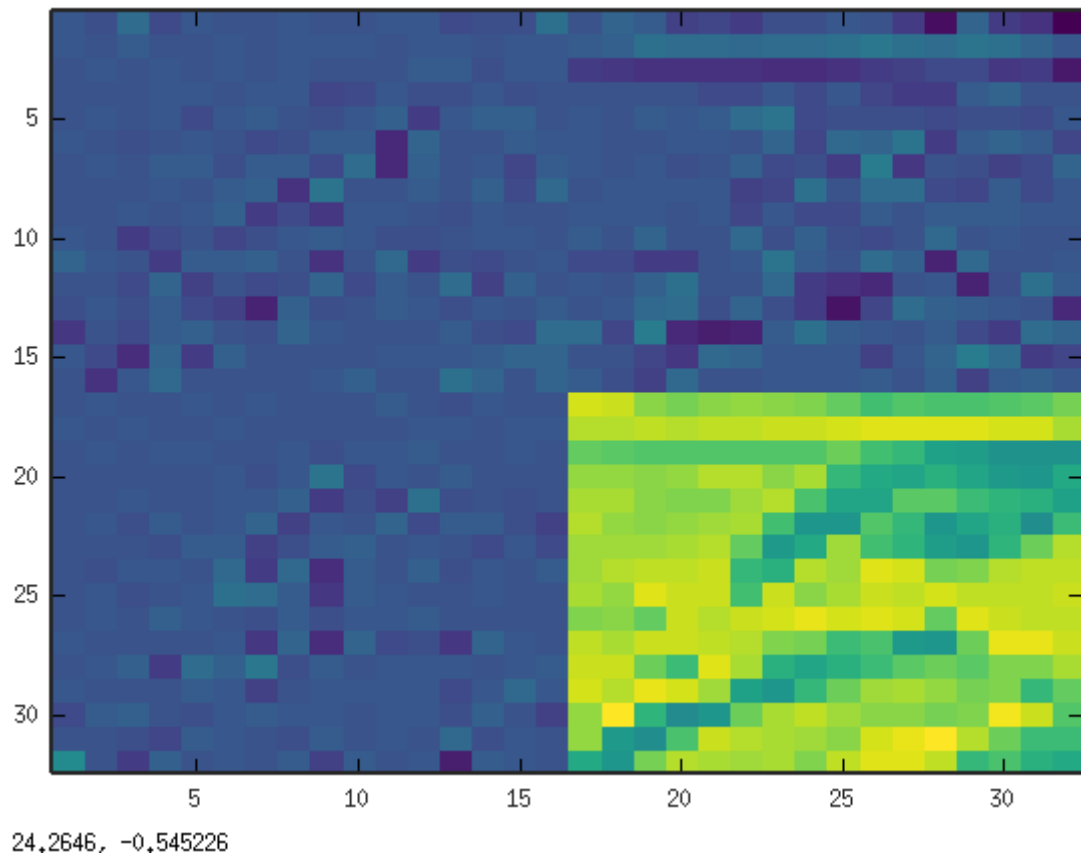
1 level DWT



This eample is of 32 x 32 red-32.pgm



1 level DWT



The first line of 32 x 32 image is read.

This is the first four dump of lifting.c which match the values obtained in VCD file below.

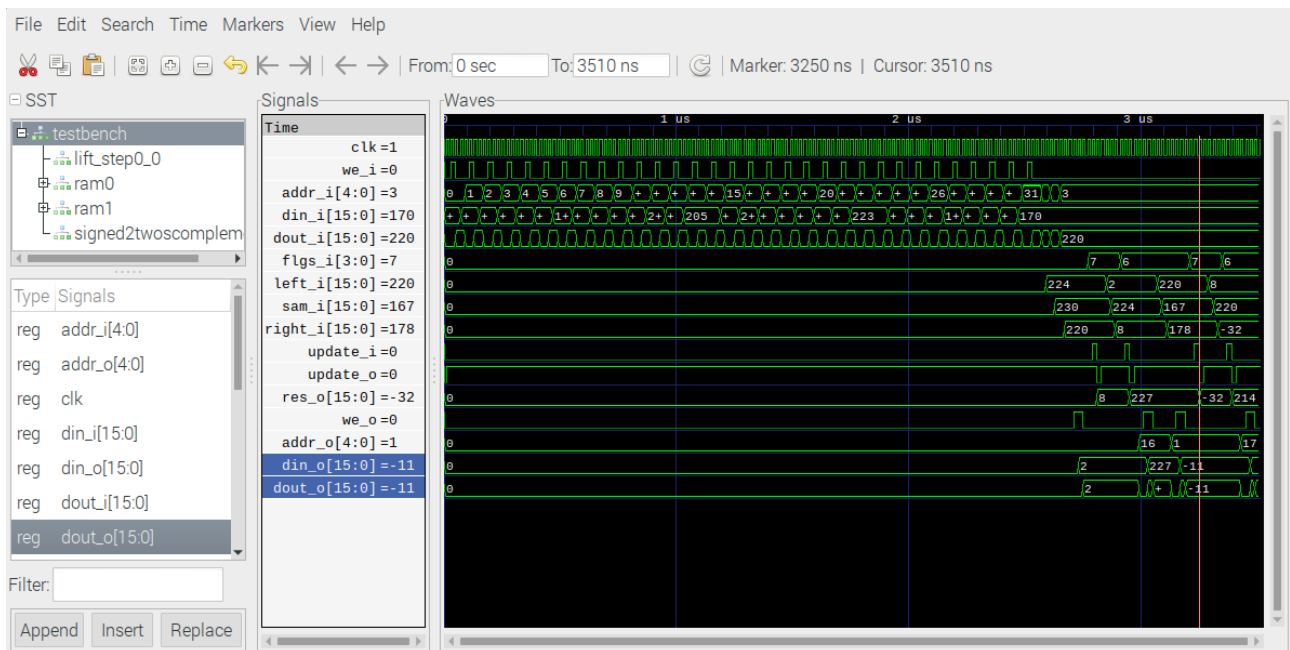
```

ip = 0xe196c8 tp = 0xe1a6c8 32
in singlelift row start loop
ip = 0xe196c8 op = 0xe1a6c8 opb = 0xe1aec8
start ap = 226 b = 224 cp = 230 d = 220
start HP filter *** l = 224 s = 230 r = 220 ss = 8
start HP filter ** ap = 2 b = 224 d = 220 cp = 8
start ****LP filter l = 2 s = 224 r = 8 ss = 227
start LP filter * opb = 227
HP filter *** l = 220 s = 167 r = 178 ss = -32
HP filter ** cp = -32 b = 220 d = 178 ap = 8 ip = 0xe196d8 op = 0xe1a748 opb =
0xe1af48 row = 0 col = 1
LP filter **** l = 8 s = 220 r = -32 ss = 214
LP filter * opb = 214

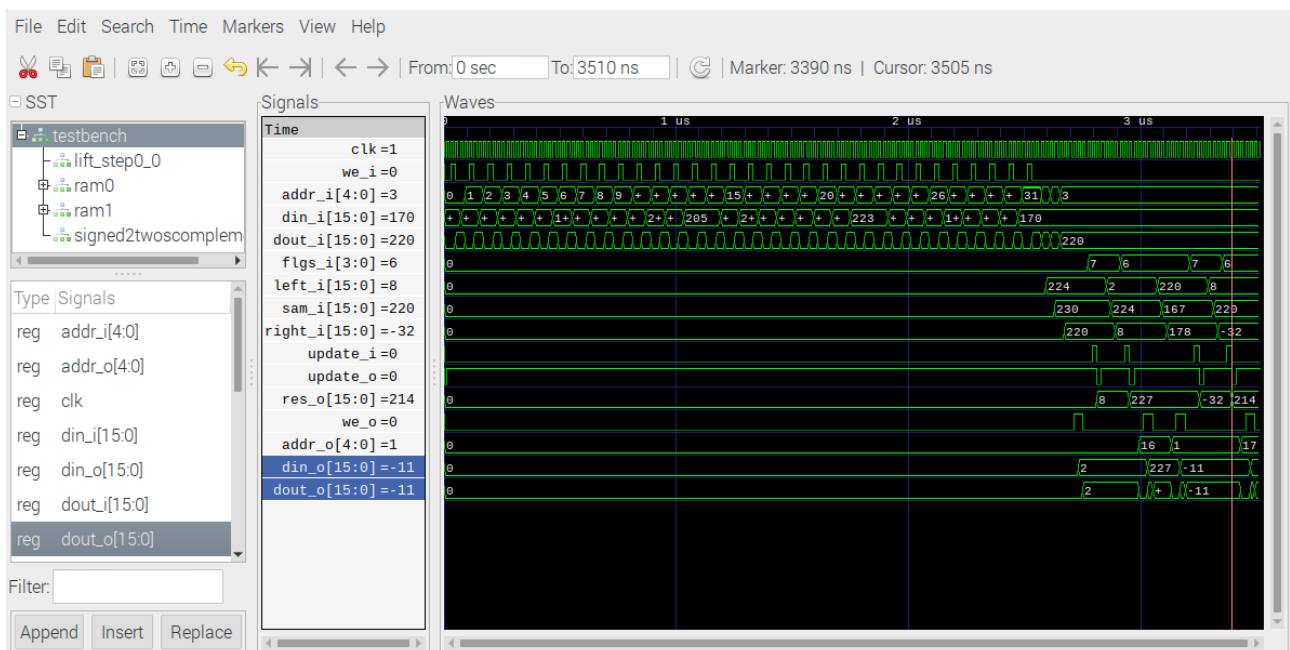
```







2nd-hi



The following verilog module performs the high pass flgs\_i = 7 and lo pass flgs\_i = 6 dwt filters when the update\_i is high and the samples left\_i, sam\_i, and right\_i. The output is res\_o. The are 16 bit signed values.

```
// File: lift_step.v
// Generated by MyHDL 0.11
// Date: Tue Dec 15 17:16:32 2020
```

```
`timescale 1ns/10ps
```

```
module lift_step (
```

```

left_i,
sam_i,
right_i,
flgs_i,
update_i,
clk,
res_o,
update_o
);

```

```

input signed [15:0] left_i;
input signed [15:0] sam_i;
input signed [15:0] right_i;
input [3:0] flgs_i;
input update_i;
input clk;
output signed [15:0] res_o;
reg signed [15:0] res_o;
output update_o;
reg update_o;

```

```

always @(posedge clk) begin: LIFT_STEP_RTL
    if ((update_i == 1)) begin
        update_o <= 0;
        case (flgs_i)
            'h7: begin
                res_o <= (sam_i - ($signed(left_i >>> 1) + $signed(right_i >>> 1)));
            end
            'h5: begin
                res_o <= (sam_i + ($signed(left_i >>> 1) + $signed(right_i >>> 1)));
            end
            'h6: begin
                res_o <= (sam_i + $signed(((left_i + right_i) + 2) >>> 2));
            end
            'h4: begin
                res_o <= (sam_i - $signed(((left_i + right_i) + 2) >>> 2));
            end
        endcase
    end
    else begin
        update_o <= 1;
    end
end

endmodule

```

The first line of 32 x 32 image is read into the ram module.



```
// File: ram.v
// Generated by MyHDL 0.11
// Date: Tue Dec 15 13:23:26 2020
```

```
`timescale 1ns/10ps
```

```
module ram (
    dout,
    din,
    addr,
    we,
    clk
);
// Ram model
```

```
output signed [15:0] dout;
wire signed [15:0] dout;
input signed [15:0] din;
input [4:0] addr;
input we;
input clk;

reg signed [15:0] mem [0:32-1];
```

```
always @(posedge clk) begin: RAM_WRITE
    if (we) begin
        mem[addr] <= din;
    end
end
```

```
assign dout = mem[addr];
```

endmodule

