

**DIGITAL SYSTEM DESIGN APPLICATION**

**EHB436E CRN: 11280**

**Salih Ömer Ongün**

**040220780**

**Experiment 8**

**CONVOLUTION MODULE**

**Design Source**

**module** conv\_unit

**(**

**input** pixel\_clk**,**

**input** rst**,**

**input** enable**,**

**input** **signed** **[**4**:**0**]** pixel11**,**

**input** **signed** **[**4**:**0**]** pixel12**,**

**input** **signed** **[**4**:**0**]** pixel13**,**

**input** **signed** **[**4**:**0**]** pixel21**,**

**input** **signed** **[**4**:**0**]** pixel22**,**

**input** **signed** **[**4**:**0**]** pixel23**,**

**input** **signed** **[**4**:**0**]** pixel31**,**

**input** **signed** **[**4**:**0**]** pixel32**,**

**input** **signed** **[**4**:**0**]** pixel33**,**

**input** **signed** **[**3**:**0**]** kernel11**,**

**input** **signed** **[**3**:**0**]** kernel12**,**

**input** **signed** **[**3**:**0**]** kernel13**,**

**input** **signed** **[**3**:**0**]** kernel21**,**

**input** **signed** **[**3**:**0**]** kernel22**,**

**input** **signed** **[**3**:**0**]** kernel23**,**

**input** **signed** **[**3**:**0**]** kernel31**,**

**input** **signed** **[**3**:**0**]** kernel32**,**

**input** **signed** **[**3**:**0**]** kernel33**,**

**output** **[**3**:**0**]** pixel\_out

**);**

**reg** **[**3**:**0**]** pixel\_out\_reg**;**

**reg** **signed** **[**8**:**0**]** m1**,**m2**,**m3**,**m4**,**m5**,**m6**,**m7**,**m8**,**m9**;**

**wire** **signed** **[**9**:**0**]** sum1**,**sum2**,**sum3**,**sum4**;**

**wire** **signed** **[**10**:**0**]** sum5**,**sum6**;**

**wire** **signed** **[**11**:**0**]** sum7**;**

**wire** **signed** **[**12**:**0**]** sum8**;**

**always** **@(posedge** pixel\_clk**)** **begin**

**if** **(**rst **==** 1'b1**)** **begin**

pixel\_out\_reg **<=** 4'b0**;**

**end**

**else** **if** **(**enable **==** 1'b1**)** **begin**

m1 **<=** pixel11 **\*** kernel11**;**

m2 **<=** pixel12 **\*** kernel12**;**

m3 **<=** pixel13 **\*** kernel13**;**

m4 **<=** pixel21 **\*** kernel21**;**

m5 **<=** pixel22 **\*** kernel22**;**

m6 **<=** pixel23 **\*** kernel23**;**

m7 **<=** pixel31 **\*** kernel31**;**

m8 **<=** pixel32 **\*** kernel32**;**

m9 **<=** pixel33 **\*** kernel33**;**

**if(**sum8**>**15**)** **begin**

pixel\_out\_reg **<=** 4'd15**;**

**end**

**else** **if(**sum8**<**0**)** **begin**

pixel\_out\_reg **<=** 4'b0**;**

**end**

**else** **begin**

pixel\_out\_reg **<=** sum8**;**

**end**

**end**

**else** **begin**

pixel\_out\_reg **<=** 4'b0**;**

**end**

**end**

**assign** sum1 **=** m1 **+** m2**;**

**assign** sum2 **=** m3 **+** m4**;**

**assign** sum3 **=** m5 **+** m6**;**

**assign** sum4 **=** m7 **+** m8**;**

**assign** sum5 **=** sum1 **+** sum2**;**

**assign** sum6 **=** sum3 **+** sum4**;**

**assign** sum7 **=** sum5 **+** sum6**;**

**assign** sum8 **=** **-**1**\*(**sum7 **+** m9**);**

**assign** pixel\_out **=** pixel\_out\_reg**;**

**endmodule**

diyagram, metin, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

I set all multiplication calculations to be dependent on the clk signal. If the addition operation was in the always block, it would give incorrect results at certain intervals due to the clk signal. Because in the always block, which is dependent on the clk signal, all operations are performed simultaneously. That's why I did the addition operations with assign. I did the process in the code in the image above.

**Simulation Source**

**module** conv\_unit\_tb**();**

**reg** pixel\_clk **=** 1'b0**;**

**reg** rst **=** 1'b0**;**

**reg** enable **=** 1'b0**;**

**reg** **signed** **[**4**:**0**]** pixel11**,** pixel12**,** pixel13**;**

**reg** **signed** **[**4**:**0**]** pixel21**,** pixel22**,** pixel23**;**

**reg** **signed** **[**4**:**0**]** pixel31**,** pixel32**,** pixel33**;**

**reg** **signed** **[**3**:**0**]** kernel11**,** kernel12**,** kernel13**;**

**reg** **signed** **[**3**:**0**]** kernel21**,** kernel22**,** kernel23**;**

**reg** **signed** **[**3**:**0**]** kernel31**,** kernel32**,** kernel33**;**

**wire** **[**3**:**0**]** pixel\_out**;**

conv\_unit uut **(**

**.**pixel\_clk**(**pixel\_clk**),**

**.**rst**(**rst**),**

**.**enable**(**enable**),**

**.**pixel11**(**pixel11**),** **.**pixel12**(**pixel12**),** **.**pixel13**(**pixel13**),**

**.**pixel21**(**pixel21**),** **.**pixel22**(**pixel22**),** **.**pixel23**(**pixel23**),**

**.**pixel31**(**pixel31**),** **.**pixel32**(**pixel32**),** **.**pixel33**(**pixel33**),**

**.**kernel11**(**kernel11**),** **.**kernel12**(**kernel12**),** **.**kernel13**(**kernel13**),**

**.**kernel21**(**kernel21**),** **.**kernel22**(**kernel22**),** **.**kernel23**(**kernel23**),**

**.**kernel31**(**kernel31**),** **.**kernel32**(**kernel32**),** **.**kernel33**(**kernel33**),**

**.**pixel\_out**(**pixel\_out**)**

**);**

**always** **begin**

**#**20 pixel\_clk **=** **~**pixel\_clk**;**

**end**

**initial** **begin**

rst **=** 1**;**

enable **=** 0**;**

pixel11 **=** 0**;** pixel12 **=** 0**;** pixel13 **=** 0**;**

pixel21 **=** 0**;** pixel22 **=** 0**;** pixel23 **=** 0**;**

pixel31 **=** 0**;** pixel32 **=** 0**;** pixel33 **=** 0**;**

kernel11 **=** 1**;** kernel12 **=** 1**;** kernel13 **=** 1**;**

kernel21 **=** 1**;** kernel22 **=** **-**8**;** kernel23 **=** 1**;**

kernel31 **=** 1**;** kernel32 **=** 1**;** kernel33 **=** 1**;**

**#**50**;**

rst **=** 0**;**

enable **=** 1**;**

**#**100**;**

pixel11 **=** 5**;** pixel12 **=** 3**;** pixel13 **=** 4**;**

pixel21 **=** 7**;** pixel22 **=** 6**;** pixel23 **=** 2**;**

pixel31 **=** 1**;** pixel32 **=** 3**;** pixel33 **=** 9**;**

kernel11 **=** 1**;** kernel12 **=** 1**;** kernel13 **=** 1**;**

kernel21 **=** 1**;** kernel22 **=** **-**8**;** kernel23 **=** 1**;**

kernel31 **=** 1**;** kernel32 **=** 1**;** kernel33 **=** 1**;**

// İkinci test verisi

**#**200**;**

pixel11 **=** 2**;** pixel12 **=** 8**;** pixel13 **=** 4**;**

pixel21 **=** 5**;** pixel22 **=** 3**;** pixel23 **=** 7**;**

pixel31 **=** 9**;** pixel32 **=** 0**;** pixel33 **=** 6**;**

kernel11 **=** 1**;** kernel12 **=** 1**;** kernel13 **=** 1**;**

kernel21 **=** 1**;** kernel22 **=** **-**8**;** kernel23 **=** 1**;**

kernel31 **=** 1**;** kernel32 **=** 1**;** kernel33 **=** 1**;**

// Üçüncü test verisi

**#**200**;**

pixel11 **=** 4**;** pixel12 **=** 7**;** pixel13 **=** 5**;**

pixel21 **=** 3**;** pixel22 **=** 6**;** pixel23 **=** 1**;**

pixel31 **=** 8**;** pixel32 **=** 2**;** pixel33 **=** 0**;**

kernel11 **=** 1**;** kernel12 **=** 1**;** kernel13 **=** 1**;**

kernel21 **=** 1**;** kernel22 **=** **-**8**;** kernel23 **=** 1**;**

kernel31 **=** 1**;** kernel32 **=** 1**;** kernel33 **=** 1**;**

**#**200**;**

rst **=** 1**;**

enable **=** 0**;**

**#**50**;**

$finish**;**

**end**

**endmodule**

ekran görüntüsü, çizgi, devre içeren bir resim

Açıklama otomatik olarak oluşturulduSimulation Waveform

The upper values ​​show the pixels and the lower values ​​show the kernels. Since we expressed the kernel in a negative way, we took the negative of the result while calculating the result.

The code works correctly as seen in the waveform.

**BRAM MODULE**

Design Source

**module** b\_ram\_red

**(**

**input** pixel\_clk**,**

**input** wea**,**

**input** ena**,**

**input** **[**16**:**0**]** addra**,**

**input** **[**11**:**0**]** dina**,**

**output** **[**11**:**0**]** douta

**);**

blk\_mem\_gen\_0 red\_bram **(** // red

**.**clka**(**pixel\_clk**),** // input wire clka

**.**ena**(**ena**),** // input wire ena

**.**wea**(**wea**),** // input wire [0 : 0] wea

**.**addra**(**addra**),** // input wire [16 : 0] addra

**.**dina**(**dina**),** // input wire [11 : 0] dina

**.**douta**(**douta**)** // output wire [11 : 0] douta

**);**

**endmodule**

**Simulation Source**

**module** b\_ram\_red\_tb**();**

**reg** pixel\_clk **=** 1'b0**;**

**reg** wea **=** 1'b0**;**

**reg** ena **=** 1'b0**;**

**reg** **[**16**:**0**]** addra **=** 16'b0**;**

**reg** **[**11**:**0**]** dina **=** 12'b0**;**

**wire** **[**11**:**0**]** douta**;**

**integer** i**;**

b\_ram\_red uut

**(**

**.**pixel\_clk**(**pixel\_clk**),**

**.**wea**(**wea**),**

**.**ena**(**ena**),**

**.**addra**(**addra**),**

**.**dina**(**dina**),**

**.**douta**(**douta**)**

**);**

**always** **begin**

**#**20 pixel\_clk **<=** **~**pixel\_clk**;**

**end**

**initial** **begin**

**#**40

ena **=** 1**;**

$display**(**"we are reading"**);**

**#**300**;** // ilk baslangıcta LOCKED oluyor

**for(**i **=** 0**;** i**<**103148**;** i**=**i**+**1**)** **begin**

addra **=** i**;**

$display**(**"Address: %d, Data Read: %h"**,** addra**,** douta**);**

**#**80**;**

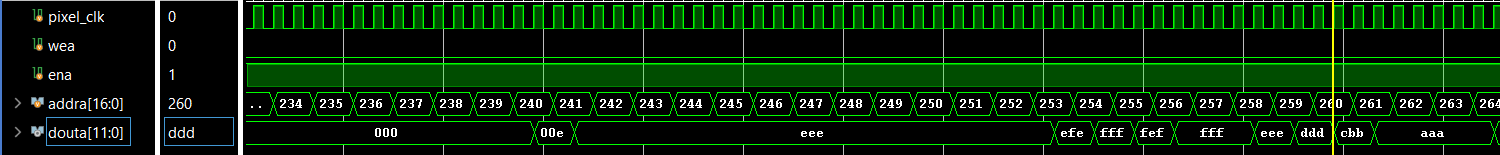
**end**

**#**50**;**

$finish**;**

**end**

**endmodule**

**Simulation Waveform**

I loaded the Bram code into a red input file and checked whether it was reading sequentially.

As can be seen in the graph, the code works correctly. Bram is serving the data correctly and in order.

**CONTROLLER MODULE**

**Design Source**

**module** controller

**(** //

**input** pixel\_clk**,**

**input** rst**,**

**input** enable**,**

**input** **[**11**:**0**]** data\_in**,**

**output** **reg** done**,**

**output** **reg** **[**16**:**0**]** address**,**

**output** **signed** **[**11**:**0**]** kernel1**,**

**output** **signed** **[**11**:**0**]** kernel2**,**

**output** **signed** **[**11**:**0**]** kernel3**,**

**output** **reg** **[**11**:**0**]** pixel1**,**

**output** **reg** **[**11**:**0**]** pixel2**,**

**output** **reg** **[**11**:**0**]** pixel3

**);**

**assign** kernel1 **=** 12'b0001\_0001\_0001**;**

**assign** kernel2 **=** 12'b0001\_1000\_0001**;**

**assign** kernel3 **=** 12'b0001\_0001\_0001**;**

**reg** **[**3**:**0**]** buffer1**[**641**:**0**];**

**reg** **[**3**:**0**]** buffer2**[**641**:**0**];**

**reg** **[**7**:**0**]** counter**;**

**reg** **[**9**:**0**]** sel\_column**;**

**reg** **[**7**:**0**]** column**;**

**reg** **[**8**:**0**]** row**;**

**reg** **[**11**:**0**]** prev\_data**;**

**localparam** FIRST\_LINE **=** 3'b000**;**

**localparam** SECOND\_LINE **=** 3'b001**;**

**localparam** SHIFT1 **=** 3'b010**;**

**localparam** SHIFT2 **=** 3'b011**;**

**localparam** SHIFT3 **=** 3'b100**;**

**localparam** DONE **=** 3'b101**;**

**reg** **[**2**:**0**]** state**;**

**always** **@(posedge** pixel\_clk**)** **begin**

**if** **(**rst **==** 1'b1**)** **begin**

state **<=** FIRST\_LINE**;**

counter **<=** 10'd0**;**

sel\_column **<=** 12'd0**;**

column **<=** 8'd0**;**

row **<=** 9'd2**;**

done **<=** 1'b0**;**

address **<=** 17'd0**;**

**end**

**else** **begin**

**if(**enable **==** 1'b1**)** **begin**

**case(**state**)**

FIRST\_LINE**:** **begin**

done **<=** 1'b0**;**

buffer1**[**counter **\*** 3**]** **<=** data\_in**[**11**:**8**];**

buffer1**[(**counter **\*** 3**)** **+** 1**]** **<=** data\_in**[**7**:**4**];**

buffer1**[(**counter **\*** 3**)** **+** 2**]** **<=** data\_in**[**3**:**0**];**

**if(**counter **==** 10'd213**)** **begin**

counter **<=** 10'd0**;**

state **<=** SECOND\_LINE**;**

address **<=** 17'd214**;**

**end**

**else** **begin**

counter **<=** counter **+** 1 **;**

address **<=** counter **+** 1**;**

**end**

**end**

SECOND\_LINE**:** **begin**

buffer2**[**counter **\*** 3**]** **<=** data\_in**[**11**:**8**];**

buffer2**[(**counter **\*** 3**)** **+** 1**]** **<=** data\_in**[**7**:**4**];**

buffer2**[(**counter **\*** 3**)** **+** 2**]** **<=** data\_in**[**3**:**0**];**

**if(**counter **==** 10'd213**)** **begin**

counter **<=** 10'd0**;**

state **<=** SHIFT1**;**

address **<=** **(**214 **\*** row**);**

**end**

**else** **begin**

counter **<=** counter **+** 1 **;**

address **<=** counter **+** 1**;**

**end**

**end**

SHIFT1**:** **begin**

pixel1 **<=** **{**buffer1**[**sel\_column**],**buffer1**[**sel\_column**+**1**],**buffer1**[**sel\_column **+** 2**]};**

pixel2 **<=** **{**buffer2**[**sel\_column**],**buffer2**[**sel\_column **+** 1**],**buffer2**[**sel\_column **+** 2**]};**

pixel3 **<=** data\_in**;**

**if(**sel\_column **==** 10'd639**)** **begin**

**if(**row **==** 9'd481**)** **begin**

state **<=**DONE**;**

**end**

**else** **begin**

row **<=** row **+** 1**;**

state **<=** SHIFT1**;**

sel\_column **<=** 10'd0**;**

column **<=** 8'd0**;**

address **<=** **(**214 **\*** row**);**

**end**

sel\_column **<=** 10'd0**;**

column **<=** 8'd0**;**

**end**

**else** **begin**

prev\_data **<=** data\_in**;**

column **<=** column **+** 1**;**

address **<=** **(**214 **\*** row**)** **+** column **+** 1**;**

sel\_column **<=** sel\_column **+** 1**;**

state **<=** SHIFT2**;**

**end**

**end**

SHIFT2**:** **begin**

//address <= (214 \* row) + column; // 3 0,

pixel1 **<=** **{**buffer1**[**sel\_column**],**buffer1**[**sel\_column **+** 1**],**buffer1**[**sel\_column **+** 2**]};**

pixel2 **<=** **{**buffer2**[**sel\_column**],**buffer2**[**sel\_column **+** 1**],**buffer2**[**sel\_column **+** 2**]};**

pixel3 **<=** **{**prev\_data**[**7**:**0**],**data\_in**[**11**:**8**]};**

state **<=** SHIFT3**;**

sel\_column **<=** sel\_column **+** 1**;**

**end**

SHIFT3**:** **begin**

pixel1 **<=** **{**buffer1**[**sel\_column**],**buffer1**[**sel\_column **+** 1**],**buffer1**[**sel\_column **+** 2**]};**

pixel2 **<=** **{**buffer2**[**sel\_column**],**buffer2**[**sel\_column **+** 1**],**buffer2**[**sel\_column **+** 2**]};**

pixel3 **<=** **{**prev\_data**[**3**:**0**],**data\_in**[**11**:**4**]};**

buffer2**[**sel\_column**]** **<=** prev\_data**[**3**:**0**];**

buffer2**[**sel\_column **-** 1**]** **<=** prev\_data**[**7**:**4**];**

buffer2**[**sel\_column **-** 1**]** **<=** prev\_data**[**7**:**4**];**

buffer2**[**sel\_column **-** 2**]** **<=** prev\_data**[**11**:**8**];**

buffer1**[**sel\_column**]** **<=** buffer2**[**sel\_column**];**

buffer1**[**sel\_column **-** 1**]** **<=** buffer2**[**sel\_column **-** 1**];**

buffer1**[**sel\_column **-** 2**]** **<=** buffer2**[**sel\_column **-** 2**];**

state **<=** SHIFT1**;**

sel\_column **<=** sel\_column **+** 1**;**

**end**

DONE**:** **begin**

state **<=** FIRST\_LINE**;**

counter **<=** 10'd0**;**

sel\_column **<=** 12'd0**;**

column **<=** 8'd0**;**

row **<=** 9'd2**;**

done **<=** 1'b1**;**

**end**

**default:** **begin**

state **<=** FIRST\_LINE**;**

counter **<=** 10'd0**;**

sel\_column **<=** 12'd0**;**

column **<=** 8'd0**;**

row **<=** 9'd2**;**

done **<=** 1'b0**;**

**end**

**endcase**

**end**

**end**

**end**

**endmodule**

In FIRST\_LINE and SECOND\_LINE states, I assigned the value of the first two lines to buffer1 and buffer2. Before going to SHIFT1 state, I gave the address variable the address of line 3, column 1 in BRAM. Thus, data\_in obtained the first 12 pixels of line 3.

In SHIFT1, I set the buffers to pixel outputs with the data coming from BRAM. With the If block, I checked whether the index came to the last column element on the line. If it did, I increased the line by one and gave the BRAM address of the bottom line to the address and made data\_in receive data from the bottom line. If the last line is where I am, I made it go to the DONE state. Before moving to SHIFT2, I shifted the BRAM address one step to the right and assigned the data to the prev\_data variable.

In SHIFT2, I received data from the buffers shifted one pixel to the right. For Pixel 3, I used the first 8 bits of prev\_data, which is the previous data, and the last 4 bits of the new data.

In SHIFT3, as in SHIFT2, I shifted one pixel to the right and assigned it to the outputs. Without starting a 3-pixel process again, I transferred the data to the next line and returned to SHIFT1.

When the whole picture convolves, it goes from SHIFT1 to DONE.

**data\_in**

**A B 9 C 1 3 7 2 D SHIFT1**

**Prev\_data data\_in**

**A B 9 C 1 3 7 2 D SHIFT2**

**Prev\_data data\_in**

**A B 9 C 1 3 7 2 D SHIFT3**

**Prev\_data Buffer2**

**data\_in Buffer2 Buffer1**

**A B 9 C 1 3 7 2 D SHIFT1**

**Simulation Source**

I simulate controller module in top module.

**TOP MODULE**

**module** top\_module

**(**

**input** clk**,**

**input** rst**,**

**output** VGA\_HS**,**

**output** VGA\_VS**,**

**output** data\_en**,**

**output** **[**3**:**0**]** VGA\_R**,**

**output** **[**3**:**0**]** VGA\_G**,**

**output** **[**3**:**0**]** VGA\_B

**);**

**wire** locked**;**

**wire** **[**16**:**0**]** address\_r**,**address\_b**,**address\_g**;**

**wire** **[**11**:**0**]** data\_in\_r**,**data\_in\_b**,**data\_in\_g**;**

**wire** dina**;**

**wire** **signed** **[**4**:**0**]** pixel11**,**pixel12**,**pixel13**,**pixel21**,**pixel22**,**pixel23**,**pixel31**,**pixel32**,**pixel33**;**

**wire** **signed** **[**3**:**0**]** kernel11**,**kernel12**,**kernel13**,**kernel21**,**kernel22**,**kernel23**,**kernel31**,**kernel32**,**kernel33**;**

**wire** **signed** **[**4**:**0**]** pixel11\_b**,**pixel12\_b**,**pixel13\_b**,**pixel21\_b**,**pixel22\_b**,**pixel23\_b**,**pixel31\_b**,**pixel32\_b**,**pixel33\_b**;**

**wire** **signed** **[**4**:**0**]** pixel11\_g**,**pixel12\_g**,**pixel13\_g**,**pixel21\_g**,**pixel22\_g**,**pixel23\_g**,**pixel31\_g**,**pixel32\_g**,**pixel33\_g**;**

**wire** wea**;**

**wire** done**;**

**wire** **[**11**:**0**]** kernel1**,**kernel2**,**kernel3**,**pixel1**,**pixel2**,**pixel3**,**pixel1\_b**,**pixel2\_b**,**pixel3\_b**,**pixel1\_g**,**pixel2\_g**,**pixel3\_g**;**

//wire [3:0] pixel\_out;

**assign** wea **=** 1'b0**;**

**assign** pixel11 **=** **{**1'b0**,**pixel1**[**11**:**8**]};**

**assign** pixel12 **=** **{**1'b0**,**pixel1**[**7**:**4**]};**

**assign** pixel13 **=** **{**1'b0**,**pixel1**[**3**:**0**]};**

**assign** pixel21 **=** **{**1'b0**,**pixel2**[**11**:**8**]};**

**assign** pixel22 **=** **{**1'b0**,**pixel2**[**7**:**4**]};**

**assign** pixel23 **=** **{**1'b0**,**pixel2**[**3**:**0**]};**

**assign** pixel31 **=** **{**1'b0**,**pixel3**[**11**:**8**]};**

**assign** pixel32 **=** **{**1'b0**,**pixel3**[**7**:**4**]};**

**assign** pixel33 **=** **{**1'b0**,**pixel3**[**3**:**0**]};**

**assign** pixel11\_b **=** **{**1'b0**,**pixel1\_b**[**11**:**8**]};**

**assign** pixel12\_b **=** **{**1'b0**,**pixel1\_b**[**7**:**4**]};**

**assign** pixel13\_b **=** **{**1'b0**,**pixel1\_b**[**3**:**0**]};**

**assign** pixel21\_b **=** **{**1'b0**,**pixel2\_b**[**11**:**8**]};**

**assign** pixel22\_b **=** **{**1'b0**,**pixel2\_b**[**7**:**4**]};**

**assign** pixel23\_b **=** **{**1'b0**,**pixel2\_b**[**3**:**0**]};**

**assign** pixel31\_b **=** **{**1'b0**,**pixel3\_b**[**11**:**8**]};**

**assign** pixel32\_b **=** **{**1'b0**,**pixel3\_b**[**7**:**4**]};**

**assign** pixel33\_b **=** **{**1'b0**,**pixel3\_b**[**3**:**0**]};**

**assign** pixel11\_g **=** **{**1'b0**,**pixel1\_g**[**11**:**8**]};**

**assign** pixel12\_g **=** **{**1'b0**,**pixel1\_g**[**7**:**4**]};**

**assign** pixel13\_g **=** **{**1'b0**,**pixel1\_g**[**3**:**0**]};**

**assign** pixel21\_g **=** **{**1'b0**,**pixel2\_g**[**11**:**8**]};**

**assign** pixel22\_g **=** **{**1'b0**,**pixel2\_g**[**7**:**4**]};**

**assign** pixel23\_g **=** **{**1'b0**,**pixel2\_g**[**3**:**0**]};**

**assign** pixel31\_g **=** **{**1'b0**,**pixel3\_g**[**11**:**8**]};**

**assign** pixel32\_g **=** **{**1'b0**,**pixel3\_g**[**7**:**4**]};**

**assign** pixel33\_g **=** **{**1'b0**,**pixel3\_g**[**3**:**0**]};**

**assign** kernel11 **=** kernel1**[**11**:**8**];**

**assign** kernel12 **=** kernel1**[**7**:**4**];**

**assign** kernel13 **=** kernel1**[**3**:**0**];**

**assign** kernel21 **=** kernel2**[**11**:**8**];**

**assign** kernel22 **=** kernel2**[**7**:**4**];**

**assign** kernel23 **=** kernel2**[**3**:**0**];**

**assign** kernel31 **=** kernel3**[**11**:**8**];**

**assign** kernel32 **=** kernel3**[**7**:**4**];**

**assign** kernel33 **=** kernel3**[**3**:**0**];**

**parameter** HPULSE **=** 96**;** // Hsync pulse

**parameter** HBP **=** 48**;** // Back Porch

**parameter** HACTIVE **=** 640**;** // Horizontal pixels

**parameter** HFP **=** 16**;** // Front Porch

**parameter** H1 **=** HPULSE**;**

**parameter** H2 **=** HPULSE**+**HBP**;**

**parameter** H3 **=** HPULSE**+**HBP**+**HACTIVE**;**

**parameter** H4 **=** HPULSE**+**HBP**+**HACTIVE**+**HFP**;**

// vertical timing parameters (default 640x480x60)

**parameter** VPULSE **=** 2**;** // Vsync pulse

**parameter** VBP **=** 33**;** // Back Porch

**parameter** VACTIVE **=** 480**;** // Vertical pixels

**parameter** VFP **=** 10**;** // Front Porch

**parameter** V1 **=** VPULSE**;**

**parameter** V2 **=** VPULSE**+**VBP**;**

**parameter** V3 **=** VPULSE**+**VBP**+**VACTIVE**;**

**parameter** V4 **=** VPULSE**+**VBP**+**VACTIVE**+**VFP**;**

clk\_wiz\_0 pixel\_clk\_gen

**(**

**.**pixel\_clk**(**pixel\_clk**),** // ok

**.**reset**(**rst**),** // ok

**.**locked**(**locked**),** // ok

**.**clk\_in1**(**clk**)** // ok

**);**

conv\_unit red\_conv

**(**

**.**pixel\_clk**(**pixel\_clk**),** // ok

**.**rst**(**rst**),** // ok

**.**enable**(**data\_en**),** // ok

**.**pixel11**(**pixel11**),** **.**pixel12**(**pixel12**),** **.**pixel13**(**pixel13**),** // ok

**.**pixel21**(**pixel21**),** **.**pixel22**(**pixel22**),** **.**pixel23**(**pixel23**),** // ok

**.**pixel31**(**pixel31**),** **.**pixel32**(**pixel32**),** **.**pixel33**(**pixel33**),** // ok

**.**kernel11**(**kernel11**),** **.**kernel12**(**kernel12**),** **.**kernel13**(**kernel13**),** //ok

**.**kernel21**(**kernel21**),** **.**kernel22**(**kernel22**),** **.**kernel23**(**kernel23**),** //ok

**.**kernel31**(**kernel31**),** **.**kernel32**(**kernel32**),** **.**kernel33**(**kernel33**),** //ok

**.**pixel\_out**(**VGA\_R**)** //ok

**);**

conv\_unit blue\_conv

**(**

**.**pixel\_clk**(**pixel\_clk**),** // ok

**.**rst**(**rst**),** // ok

**.**enable**(**data\_en**),** // ok

**.**pixel11**(**pixel11\_b**),** **.**pixel12**(**pixel12\_b**),** **.**pixel13**(**pixel13\_b**),** // ok

**.**pixel21**(**pixel21\_b**),** **.**pixel22**(**pixel22\_b**),** **.**pixel23**(**pixel23\_b**),** // ok

**.**pixel31**(**pixel31\_b**),** **.**pixel32**(**pixel32\_b**),** **.**pixel33**(**pixel33\_b**),** // ok

**.**kernel11**(**kernel11**),** **.**kernel12**(**kernel12**),** **.**kernel13**(**kernel13**),** //ok

**.**kernel21**(**kernel21**),** **.**kernel22**(**kernel22**),** **.**kernel23**(**kernel23**),** //ok

**.**kernel31**(**kernel31**),** **.**kernel32**(**kernel32**),** **.**kernel33**(**kernel33**),** //ok

**.**pixel\_out**(**VGA\_B**)** //ok

**);**

conv\_unit green\_conv

**(**

**.**pixel\_clk**(**pixel\_clk**),** // ok

**.**rst**(**rst**),** // ok

**.**enable**(**data\_en**),** // ok

**.**pixel11**(**pixel11\_g**),** **.**pixel12**(**pixel12\_g**),** **.**pixel13**(**pixel13\_g**),** // ok

**.**pixel21**(**pixel21\_g**),** **.**pixel22**(**pixel22\_g**),** **.**pixel23**(**pixel23\_g**),** // ok

**.**pixel31**(**pixel31\_g**),** **.**pixel32**(**pixel32\_g**),** **.**pixel33**(**pixel33\_g**),** // ok

**.**kernel11**(**kernel11**),** **.**kernel12**(**kernel12**),** **.**kernel13**(**kernel13**),** //ok

**.**kernel21**(**kernel21**),** **.**kernel22**(**kernel22**),** **.**kernel23**(**kernel23**),** //ok

**.**kernel31**(**kernel31**),** **.**kernel32**(**kernel32**),** **.**kernel33**(**kernel33**),** //ok

**.**pixel\_out**(**VGA\_G**)** //ok

**);**

b\_ram\_blue blue\_bram

**(**

**.**pixel\_clk**(**pixel\_clk**),** // ok

**.**wea**(**wea**),** // ok

**.**ena**(**data\_en**),** // ok

**.**addra**(**address\_b**),** // ok

**.**dina**(**dina**),** // ok

**.**douta**(**data\_in\_b**)** //ok

**);**

b\_ram\_red red\_bram

**(**

**.**pixel\_clk**(**pixel\_clk**),** // ok

**.**wea**(**wea**),** // ok

**.**ena**(**data\_en**),** // ok

**.**addra**(**address\_r**),** // ok

**.**dina**(**dina**),** // ok

**.**douta**(**data\_in\_r**)** //ok

**);**

b\_ram\_blue blue\_bram

**(**

**.**pixel\_clk**(**pixel\_clk**),** // ok

**.**wea**(**wea**),** // ok

**.**ena**(**data\_en**),** // ok

**.**addra**(**address\_b**),** // ok

**.**dina**(**dina**),** // ok

**.**douta**(**data\_in\_b**)** //ok

**);**

b\_ram\_green green\_bram

**(**

**.**pixel\_clk**(**pixel\_clk**),** // ok

**.**wea**(**wea**),** // ok

**.**ena**(**data\_en**),** // ok

**.**addra**(**address\_g**),** // ok

**.**dina**(**dina**),** // ok

**.**douta**(**data\_in\_g**)** //ok

**);**

controller red\_control

**(**

**.**pixel\_clk**(**pixel\_clk**),** // ok

**.**rst**(**rst**),** // ok

**.**enable**(**data\_en**),** // ok

**.**data\_in**(**data\_in\_r**),** // ok

**.**done**(**done**),** // ok

**.**address**(**address\_r**),** // ok

**.**kernel1**(**kernel1**),** // ok

**.**kernel2**(**kernel2**),** // ok

**.**kernel3**(**kernel3**),** // ok

**.**pixel1**(**pixel1**),** // ok

**.**pixel2**(**pixel2**),** // ok

**.**pixel3**(**pixel3**)** // ok

**);**

controller blue\_control

**(**

**.**pixel\_clk**(**pixel\_clk**),** // ok

**.**rst**(**rst**),** // ok

**.**enable**(**data\_en**),** // ok

**.**data\_in**(**data\_in\_b**),** // ok

**.**done**(**done**),** // ok

**.**address**(**address\_b**),** // ok

**.**kernel1**(**kernel1**),** // ok

**.**kernel2**(**kernel2**),** // ok

**.**kernel3**(**kernel3**),** // ok

**.**pixel1**(**pixel1\_b**),** // ok

**.**pixel2**(**pixel2\_b**),** // ok

**.**pixel3**(**pixel3\_b**)** // ok

**);**

controller green\_control

**(**

**.**pixel\_clk**(**pixel\_clk**),** // ok

**.**rst**(**rst**),** // ok

**.**enable**(**data\_en**),** // ok

**.**data\_in**(**data\_in\_g**),** // ok

**.**done**(**done**),** // ok

**.**address**(**address\_g**),** // ok

**.**kernel1**(**kernel1**),** // ok

**.**kernel2**(**kernel2**),** // ok

**.**kernel3**(**kernel3**),** // ok

**.**pixel1**(**pixel1\_g**),** // ok

**.**pixel2**(**pixel2\_g**),** // ok

**.**pixel3**(**pixel3\_g**)** // ok

**);**

**wire** **[**9**:**0**]** sel\_column **=** red\_control**.**sel\_column**;**

**wire** **[**9**:**0**]** counter **=** red\_control**.**counter**;**

**wire** **[**8**:**0**]** row **=** red\_control**.**row**;**

**wire** **[**7**:**0**]** column **=** red\_control**.**column**;**

**wire** **[**2**:**0**]** state **=** red\_control**.**state**;**

**wire** **[**11**:**0**]** prev\_data **=** red\_control**.**prev\_data**;**

vga\_driver **#(**

**.**HPULSE**(**HPULSE**),**

**.**HBP**(**HBP**),**

**.**HACTIVE**(**HACTIVE**),**

**.**HFP**(**HFP**),**

**.**VPULSE**(**VPULSE**),**

**.**VBP**(**VBP**),**

**.**VACTIVE**(**VACTIVE**),**

**.**VFP**(**VFP**)**

**)**

vga\_gen **(**

**.**pixel\_clk**(**pixel\_clk**),**

**.**rst**(**rst**),**

**.**VGA\_HS**(**VGA\_HS**),**

**.**VGA\_VS**(**VGA\_VS**),**

**.**data\_en**(**data\_en**)**

**);**

**endmodule**

I created 3 bram, 3 controller and 3 conv\_unit, 1 VGA modules. I adjusted required connection. I created sel\_column, counter, row, column, state, prev\_data to pursue change of controller.

**Simulation Source**

**module** top\_module\_tb**;**

**reg** clk**;**

**reg** rst**;**

**wire** data\_en**;**

**wire** VGA\_HS**;**

**wire** VGA\_VS**;**

**wire** **[**3**:**0**]** VGA\_R**;**

**wire** **[**3**:**0**]** VGA\_G**;**

**wire** **[**3**:**0**]** VGA\_B**;**

**integer** file\_r**,** file\_g**,** file\_b**;**

**integer** row\_count **=** 0**;**

**integer** col\_count **=** 0**;**

**initial** clk **=** 0**;**

**always** **#**5 clk **=** **~**clk**;**

top\_module uut **(**

**.**clk**(**clk**),**

**.**rst**(**rst**),**

**.**VGA\_HS**(**VGA\_HS**),**

**.**VGA\_VS**(**VGA\_VS**),**

**.**data\_en**(**data\_en**),**

**.**VGA\_R**(**VGA\_R**),**

**.**VGA\_G**(**VGA\_G**),**

**.**VGA\_B**(**VGA\_B**)**

**);**

**initial** **begin**

file\_r **=** $fopen**(**"vga\_r\_output.txt"**,** "w"**);**

file\_g **=** $fopen**(**"vga\_g\_output.txt"**,** "w"**);**

file\_b **=** $fopen**(**"vga\_b\_output.txt"**,** "w"**);**

**if** **(**file\_r **==** 0 **||** file\_g **==** 0 **||** file\_b **==** 0**)** **begin**

$display**(**"Error: Unable to open files for writing."**);**

$stop**;**

**end**

rst **=** 1**;**

**#**100**;**

rst **=** 0**;**

**#**5000000**;**

$fclose**(**file\_r**);**

$fclose**(**file\_g**);**

$fclose**(**file\_b**);**

$stop**;**

**end**

**always** **@(posedge** clk**)** **begin**

**if** **(!**rst **&&** data\_en**)** **begin**

$fwrite**(**file\_r**,** "%X"**,** VGA\_R**);**

$fwrite**(**file\_g**,** "%X"**,** VGA\_G**);**

$fwrite**(**file\_b**,** "%X"**,** VGA\_B**);**

col\_count **=** col\_count **+** 1**;**

**if** **(**col\_count **==** 642**)** **begin**

$fwrite**(**file\_r**,** "\n"**);**

$fwrite**(**file\_g**,** "\n"**);**

$fwrite**(**file\_b**,** "\n"**);**

col\_count **=** 0**;**

row\_count **=** row\_count **+** 1**;**

**if** **(**row\_count **==** 482**)** **begin**

$fclose**(**file\_r**);**

$fclose**(**file\_g**);**

$fclose**(**file\_b**);**

$stop**;**

**end**

**end**

**end**

**end**

**endmodule**

**Simulation Waveform**

**ekran görüntüsü, renklilik, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu**

I could not reach the same value with the output R,G,B pixels. I have checked my controller and other modules but I can't find where the problem is.

**Utilization Report**

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu