**RGB TO GRAYSCALE CONVERSION**

**MATLAB Code:**

function [x\_out, y\_out,r\_out,g\_out,b\_out]=rtg(x\_in, y\_in, r\_in, g\_in, b\_in)

persistent OrigImg

persistent x1 x2 y1 y2

D = [0.298 0 0 ;0 0.587 0;0 0 0.114];

RGB = [r\_in; g\_in; b\_in];

OrigImg\_1 = D\*RGB;

r\_out=OrigImg\_1(1);

g\_out=OrigImg\_1(2);

b\_out=OrigImg\_1(3);

% % r\_out=0.2989 \*r\_in ;

% % g\_out=0.5870 \* g\_in;

% % b\_out=0.1140 \* b\_in;

x\_out = x2;

x2 = x1;

x1 = x\_in;

y\_out = y2;

y2 = y1;

y1 = y\_in;

end

**Test Bench:**

WIDTH = 800;

HEIGHT = 450;

rout=zeros(450,800);

ImgData = double(imread("C:\Users\Public\lotus1.jpg"));

for y = 0:HEIGHT

for x = 0:WIDTH

if y >= 0 && y < HEIGHT && x >= 0 && x < WIDTH

b = ImgData(y+1,x+1,1);

g = ImgData(y+1,x+1,2);

r = ImgData(y+1,x+1,3);

else

b = 0;

g = 0;

r = 0;

end

[xOut, yOut,rout,gout,bout] =rtg(x, y, r, g, b);

Iout(y+1,x+1,1)=bout;

Iout(y+1,x+1,2)=gout;

Iout(y+1,x+1,3)=rout;

end

end

figure(1)

imshow(uint8(ImgData));

figure(2)

imshow(uint8(Iout),[]);

**VHDL Code:**

-- -------------------------------------------------------------

--

-- File Name: C:\Users\Public\codegen\rtg\hdlsrc\rtg\_fixpt.vhd

-- Created: 2024-05-05 23:30:56

--

-- Generated by MATLAB 9.12, MATLAB Coder 5.4 and HDL Coder 3.20

--

--

--

-- -------------------------------------------------------------

-- Rate and Clocking Details

-- -------------------------------------------------------------

-- Design base rate: 1

--

--

-- Clock Enable Sample Time

-- -------------------------------------------------------------

-- ce\_out 1

-- -------------------------------------------------------------

--

--

-- Output Signal Clock Enable Sample Time

-- -------------------------------------------------------------

-- x\_out ce\_out 1

-- y\_out ce\_out 1

-- r\_out ce\_out 1

-- g\_out ce\_out 1

-- b\_out ce\_out 1

-- -------------------------------------------------------------

--

-- -------------------------------------------------------------

-- -------------------------------------------------------------

--

-- Module: rtg\_fixpt

-- Source Path: rtg\_fixpt

-- Hierarchy Level: 0

--

-- -------------------------------------------------------------

LIBRARY IEEE;

USE IEEE.std\_logic\_1164.ALL;

USE IEEE.numeric\_std.ALL;

USE work.rtg\_fixpt\_pkg.ALL;

ENTITY rtg\_fixpt IS

PORT( clk : IN std\_logic;

reset : IN std\_logic;

clk\_enable : IN std\_logic;

x\_in : IN std\_logic\_vector(9 DOWNTO 0);

y\_in : IN std\_logic\_vector(8 DOWNTO 0);

r\_in : IN std\_logic\_vector(7 DOWNTO 0);

g\_in : IN std\_logic\_vector(7 DOWNTO 0);

b\_in : IN std\_logic\_vector(7 DOWNTO 0);

ce\_out : OUT std\_logic;

x\_out : OUT std\_logic\_vector(9 DOWNTO 0);

y\_out : OUT std\_logic\_vector(8 DOWNTO 0);

r\_out : OUT std\_logic\_vector(6 DOWNTO 0);

g\_out : OUT std\_logic\_vector(7 DOWNTO 0);

b\_out : OUT std\_logic\_vector(4 DOWNTO 0)

);

END rtg\_fixpt;

ARCHITECTURE rtl OF rtg\_fixpt IS

-- Constants

CONSTANT a0 : vector\_of\_unsigned14(0 TO 8) :=

(to\_unsigned(16#1312#, 14), to\_unsigned(16#0000#, 14), to\_unsigned(16#0000#, 14), to\_unsigned(16#0000#, 14),

to\_unsigned(16#2591#, 14), to\_unsigned(16#0000#, 14), to\_unsigned(16#0000#, 14), to\_unsigned(16#0000#, 14),

to\_unsigned(16#074B#, 14)); -- ufix14 [9]

-- Signals

SIGNAL enb : std\_logic;

SIGNAL x\_in\_unsigned : unsigned(9 DOWNTO 0); -- ufix10

SIGNAL x2\_reg\_reg : vector\_of\_unsigned10(0 TO 1); -- ufix10 [2]

SIGNAL x2 : unsigned(9 DOWNTO 0); -- ufix10

SIGNAL y\_in\_unsigned : unsigned(8 DOWNTO 0); -- ufix9

SIGNAL y2\_reg\_reg : vector\_of\_unsigned9(0 TO 1); -- ufix9 [2]

SIGNAL y2 : unsigned(8 DOWNTO 0); -- ufix9

SIGNAL r\_in\_unsigned : unsigned(7 DOWNTO 0); -- ufix8

SIGNAL g\_in\_unsigned : unsigned(7 DOWNTO 0); -- ufix8

SIGNAL b\_in\_unsigned : unsigned(7 DOWNTO 0); -- ufix8

SIGNAL RGB : vector\_of\_unsigned8(0 TO 2); -- ufix8 [3]

SIGNAL c : vector\_of\_unsigned24(0 TO 2); -- ufix24\_En14 [3]

SIGNAL c\_1 : vector\_of\_unsigned24(0 TO 2); -- ufix24\_En14 [3]

SIGNAL OrigImg\_1 : vector\_of\_unsigned14(0 TO 2); -- ufix14\_En6 [3]

SIGNAL r\_out\_tmp : unsigned(6 DOWNTO 0); -- ufix7

SIGNAL g\_out\_tmp : unsigned(7 DOWNTO 0); -- ufix8

SIGNAL b\_out\_tmp : unsigned(4 DOWNTO 0); -- ufix5

BEGIN

x\_in\_unsigned <= unsigned(x\_in);

enb <= clk\_enable;

-- % b\_out=0.1140 \* b\_in;

--

-- % g\_out=0.5870 \* g\_in;

--

-- % r\_out=0.2989 \*r\_in ;

x2\_reg\_process : PROCESS (clk, reset)

BEGIN

IF reset = '1' THEN

x2\_reg\_reg <= (OTHERS => to\_unsigned(16#000#, 10));

ELSIF clk'EVENT AND clk = '1' THEN

IF enb = '1' THEN

x2\_reg\_reg(0) <= x\_in\_unsigned;

x2\_reg\_reg(1) <= x2\_reg\_reg(0);

END IF;

END IF;

END PROCESS x2\_reg\_process;

x2 <= x2\_reg\_reg(1);

x\_out <= std\_logic\_vector(x2);

y\_in\_unsigned <= unsigned(y\_in);

y2\_reg\_process : PROCESS (clk, reset)

BEGIN

IF reset = '1' THEN

y2\_reg\_reg <= (OTHERS => to\_unsigned(16#000#, 9));

ELSIF clk'EVENT AND clk = '1' THEN

IF enb = '1' THEN

y2\_reg\_reg(0) <= y\_in\_unsigned;

y2\_reg\_reg(1) <= y2\_reg\_reg(0);

END IF;

END IF;

END PROCESS y2\_reg\_process;

y2 <= y2\_reg\_reg(1);

y\_out <= std\_logic\_vector(y2);

r\_in\_unsigned <= unsigned(r\_in);

g\_in\_unsigned <= unsigned(g\_in);

b\_in\_unsigned <= unsigned(b\_in);

-- HDL code generation from MATLAB function: rtg\_fixpt

--

-- b\_out = OrigImg\_1(3);

--

-- g\_out = OrigImg\_1(2);

--

-- r\_out = OrigImg\_1(1);

--

-- OrigImg\_1 = D' \* RGB; % Corrected matrix multiplication

--

-- RGB = [r\_in; g\_in; b\_in];

--

-- D = [0.2989; 0.5870; 0.1140]; % Corrected grayscale conversion matrix

--

-- %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

--

-- %

--

-- Generated by MATLAB 9.12 and Fixed-Point Designer 7.4 %

--

-- %

--

-- %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

RGB(0) <= r\_in\_unsigned;

RGB(1) <= g\_in\_unsigned;

RGB(2) <= b\_in\_unsigned;

c <= (OTHERS => to\_unsigned(16#000000#, 24));

p4\_output : PROCESS (RGB, c)

VARIABLE slice\_temp : unsigned(23 DOWNTO 0);

VARIABLE add\_cast : vector\_of\_unsigned25(0 TO 2);

VARIABLE mul\_temp : vector\_of\_unsigned22(0 TO 2);

VARIABLE add\_cast\_0 : vector\_of\_unsigned25(0 TO 2);

VARIABLE add\_temp : vector\_of\_unsigned25(0 TO 2);

BEGIN

slice\_temp := to\_unsigned(16#000000#, 24);

FOR l IN 0 TO 2 LOOP

slice\_temp := c(l);

FOR k IN 0 TO 2 LOOP

add\_cast(k) := resize(slice\_temp, 25);

mul\_temp(k) := a0(l + (3 \* k)) \* RGB(k);

add\_cast\_0(k) := resize(mul\_temp(k), 25);

add\_temp(k) := add\_cast(k) + add\_cast\_0(k);

slice\_temp := add\_temp(k)(23 DOWNTO 0);

END LOOP;

c\_1(l) <= slice\_temp;

END LOOP;

END PROCESS p4\_output;

OrigImg\_1\_gen: FOR t\_0 IN 0 TO 2 GENERATE

OrigImg\_1(t\_0) <= c\_1(t\_0)(21 DOWNTO 8);

END GENERATE OrigImg\_1\_gen;

r\_out\_tmp <= OrigImg\_1(0)(12 DOWNTO 6);

r\_out <= std\_logic\_vector(r\_out\_tmp);

g\_out\_tmp <= OrigImg\_1(1)(13 DOWNTO 6);

g\_out <= std\_logic\_vector(g\_out\_tmp);

b\_out\_tmp <= OrigImg\_1(2)(10 DOWNTO 6);

b\_out <= std\_logic\_vector(b\_out\_tmp);

ce\_out <= clk\_enable;

END rtl;

**Verilog Code:**

// -------------------------------------------------------------

//

// File Name: C:\Users\Public\codegen\rtg\hdlsrc\rtg\_fixpt.v

// Created: 2024-05-05 23:34:40

//

// Generated by MATLAB 9.12, MATLAB Coder 5.4 and HDL Coder 3.20

//

//

//

// -- -------------------------------------------------------------

// -- Rate and Clocking Details

// -- -------------------------------------------------------------

// Design base rate: 1

//

//

// Clock Enable Sample Time

// -- -------------------------------------------------------------

// ce\_out 1

// -- -------------------------------------------------------------

//

//

// Output Signal Clock Enable Sample Time

// -- -------------------------------------------------------------

// x\_out ce\_out 1

// y\_out ce\_out 1

// r\_out ce\_out 1

// g\_out ce\_out 1

// b\_out ce\_out 1

// -- -------------------------------------------------------------

//

// -------------------------------------------------------------

// -------------------------------------------------------------

//

// Module: rtg\_fixpt

// Source Path: rtg\_fixpt

// Hierarchy Level: 0

//

// -------------------------------------------------------------

`timescale 1 ns / 1 ns

module rtg\_fixpt

(clk,

reset,

clk\_enable,

x\_in,

y\_in,

r\_in,

g\_in,

b\_in,

ce\_out,

x\_out,

y\_out,

r\_out,

g\_out,

b\_out);

input clk;

input reset;

input clk\_enable;

input [9:0] x\_in;

input [8:0] y\_in;

input [7:0] r\_in;

input [7:0] g\_in;

input [7:0] b\_in;

output ce\_out;

output [9:0] x\_out;

output [8:0] y\_out;

output [6:0] r\_out;

output [7:0] g\_out;

output [4:0] b\_out;

wire enb;

reg [9:0] x2\_reg\_reg [0:1];

wire [9:0] x2\_reg\_reg\_next [0:1];

wire [9:0] x2;

reg [8:0] y2\_reg\_reg [0:1];

wire [8:0] y2\_reg\_reg\_next [0:1];

wire [8:0] y2;

wire [7:0] RGB [0:2];

wire [23:0] c [0:2];

reg [23:0] c\_1 [0:2];

wire [23:0] c\_2 [0:2];

wire [13:0] OrigImg\_1 [0:2];

reg signed [31:0] p4\_k;

reg signed [31:0] p4\_l;

reg [13:0] p4\_a0 [0:8];

reg [23:0] p4\_slice\_temp;

reg [24:0] p4\_add\_cast [0:2];

reg [21:0] p4\_mul\_temp [0:2];

reg [24:0] p4\_add\_cast\_0 [0:2];

reg [24:0] p4\_add\_temp [0:2];

assign enb = clk\_enable;

// % b\_out=0.1140 \* b\_in;

//

// % g\_out=0.5870 \* g\_in;

//

// % r\_out=0.2989 \*r\_in ;

always @(posedge clk or posedge reset)

begin : x2\_reg\_process

if (reset == 1'b1) begin

x2\_reg\_reg[0] <= 10'b0000000000;

x2\_reg\_reg[1] <= 10'b0000000000;

end

else begin

if (enb) begin

x2\_reg\_reg[0] <= x2\_reg\_reg\_next[0];

x2\_reg\_reg[1] <= x2\_reg\_reg\_next[1];

end

end

end

assign x2 = x2\_reg\_reg[1];

assign x2\_reg\_reg\_next[0] = x\_in;

assign x2\_reg\_reg\_next[1] = x2\_reg\_reg[0];

always @(posedge clk or posedge reset)

begin : y2\_reg\_process

if (reset == 1'b1) begin

y2\_reg\_reg[0] <= 9'b000000000;

y2\_reg\_reg[1] <= 9'b000000000;

end

else begin

if (enb) begin

y2\_reg\_reg[0] <= y2\_reg\_reg\_next[0];

y2\_reg\_reg[1] <= y2\_reg\_reg\_next[1];

end

end

end

assign y2 = y2\_reg\_reg[1];

assign y2\_reg\_reg\_next[0] = y\_in;

assign y2\_reg\_reg\_next[1] = y2\_reg\_reg[0];

// HDL code generation from MATLAB function: rtg\_fixpt

//

// b\_out = OrigImg\_1(3);

//

// g\_out = OrigImg\_1(2);

//

// r\_out = OrigImg\_1(1);

//

// OrigImg\_1 = D' \* RGB; % Corrected matrix multiplication

//

// RGB = [r\_in; g\_in; b\_in];

//

// D = [0.2989; 0.5870; 0.1140]; % Corrected grayscale conversion matrix

//

// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

//

// %

//

// Generated by MATLAB 9.12 and Fixed-Point Designer 7.4 %

//

// %

//

// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

assign RGB[0] = r\_in;

assign RGB[1] = g\_in;

assign RGB[2] = b\_in;

assign c[0] = 24'b000000000000000000000000;

assign c[1] = 24'b000000000000000000000000;

assign c[2] = 24'b000000000000000000000000;

always @\* begin

p4\_a0[0] = 14'b01001100010010;

p4\_a0[1] = 14'b00000000000000;

p4\_a0[2] = 14'b00000000000000;

p4\_a0[3] = 14'b00000000000000;

p4\_a0[4] = 14'b10010110010001;

p4\_a0[5] = 14'b00000000000000;

p4\_a0[6] = 14'b00000000000000;

p4\_a0[7] = 14'b00000000000000;

p4\_a0[8] = 14'b00011101001011;

p4\_slice\_temp = 24'b000000000000000000000000;

for(p4\_l = 32'sd0; p4\_l <= 32'sd2; p4\_l = p4\_l + 32'sd1) begin

p4\_slice\_temp = c[p4\_l];

for(p4\_k = 32'sd0; p4\_k <= 32'sd2; p4\_k = p4\_k + 32'sd1) begin

p4\_add\_cast[p4\_k] = {1'b0, p4\_slice\_temp};

p4\_mul\_temp[p4\_k] = p4\_a0[p4\_l + (32'sd3 \* p4\_k)] \* RGB[p4\_k];

p4\_add\_cast\_0[p4\_k] = {3'b0, p4\_mul\_temp[p4\_k]};

p4\_add\_temp[p4\_k] = p4\_add\_cast[p4\_k] + p4\_add\_cast\_0[p4\_k];

p4\_slice\_temp = p4\_add\_temp[p4\_k][23:0];

end

c\_1[p4\_l] = p4\_slice\_temp;

end

end

assign c\_2[0] = c\_1[0];

assign c\_2[1] = c\_1[1];

assign c\_2[2] = c\_1[2];

assign OrigImg\_1[0] = c\_2[0][21:8];

assign OrigImg\_1[1] = c\_2[1][21:8];

assign OrigImg\_1[2] = c\_2[2][21:8];

assign r\_out = OrigImg\_1[0][12:6];

assign g\_out = OrigImg\_1[1][13:6];

assign b\_out = OrigImg\_1[2][10:6];

assign ce\_out = clk\_enable;

assign x\_out = x2;

assign y\_out = y2;

endmodule // rtg\_fixpt

**Package File:**

-- -------------------------------------------------------------

--

-- File Name: C:\Users\Public\codegen\rtg\hdlsrc\rtg\_fixpt\_pkg.vhd

-- Created: 2024-05-05 23:30:56

--

-- Generated by MATLAB 9.12, MATLAB Coder 5.4 and HDL Coder 3.20

--

--

-- -------------------------------------------------------------

LIBRARY IEEE;

USE IEEE.std\_logic\_1164.ALL;

USE IEEE.numeric\_std.ALL;

PACKAGE rtg\_fixpt\_pkg IS

TYPE vector\_of\_unsigned10 IS ARRAY (NATURAL RANGE <>) OF unsigned(9 DOWNTO 0);

TYPE vector\_of\_unsigned9 IS ARRAY (NATURAL RANGE <>) OF unsigned(8 DOWNTO 0);

TYPE vector\_of\_unsigned8 IS ARRAY (NATURAL RANGE <>) OF unsigned(7 DOWNTO 0);

TYPE vector\_of\_unsigned24 IS ARRAY (NATURAL RANGE <>) OF unsigned(23 DOWNTO 0);

TYPE vector\_of\_unsigned25 IS ARRAY (NATURAL RANGE <>) OF unsigned(24 DOWNTO 0);

TYPE vector\_of\_unsigned22 IS ARRAY (NATURAL RANGE <>) OF unsigned(21 DOWNTO 0);

TYPE vector\_of\_unsigned14 IS ARRAY (NATURAL RANGE <>) OF unsigned(13 DOWNTO 0);

END rtg\_fixpt\_pkg;

**Image Input:**



**Image Output:**

****