Project - Bee Invaders

Tutorial 3: Moving The Bee & Display The Aliens On The Screen

This Tutorial Is Specifically For The Digilent Basys 3 Board

Proposed Game

Instructions

01 The "Top" module has changed to include the left and right buttons on the Basys 3 board. The pixel clock has also been linked to the "Top" module from the "vga640x480" module and used to ensure that all output to VGA / the screen are controlled by the 25MHz clock

Open the project "WIP" in Vivado

Double click on "Top (Top.v)" in the Sources (design) panel to open the module

Remove all the code in the "Top.v" box and copy & paste the code from either the "Top.v" file you downloaded or from below, into the "Top.v" code box

```
//----------------------------------------------------
// Top Module : Digilent Basys 3
// BeeInvaders Tutorial 3 : Onboard clock 100MHz
// VGA Resolution 640x480 @ 60Hz : Pixel Clock 25MHz
//----------------------------------------------------
`timescale 1ns / 1ps

module Top(
    input wire CLK, // Onboard clock 100MHz : INPUT Pin W5
    input wire RESET, // Reset button / Centre Button : INPUT Pin U18
    output wire HSYNC, // VGA horizontal sync : OUTPUT Pin P19
    output wire VSYNC, // VGA vertical sync : OUTPUT Pin R19
    output reg [3:0] RED, // 4-bit VGA Red : OUTPUT Pin G19, Pin H19, Pin J19, Pin N19
    output reg [3:0] GREEN, // 4-bit VGA Green : OUTPUT Pin J17, Pin H17, Pin G17, Pin D17
    output reg [3:0] BLUE, // 4-bit VGA Blue : OUTPUT Pin N18, Pin L18, Pin K18, Pin J18/ 4-bit VGA Blue : OUTPUT
Pin N18, Pin L18, Pin K18, Pin J18
    input btnR, // Right button : INPUT Pin T17
    input btnL // Left button : INPUT Pin W19
    );

    wire rst = RESET; // Setup Reset button
```

```
// instantiate vga640x480 code
wire [9:0] x; // pixel x position: 10-bit value: 0-1023 : only need 800
wire [9:0] y; // pixel y position: 10-bit value: 0-1023 : only need 525
wire active; // high during active pixel drawing
wire PixCLK; // 25MHz pixel clock
vga640x480 display (.i_clk(CLK),.i_rst(rst),.o_hsync(HSYNC),
                    .o_vsync(VSYNC),.o_x(x),.o_y(y),.o_active(active),
                    .pix_clk(PixCLK));

// instantiate BeeSprite code
wire BeeSpriteOn; // 1=on, 0=off
wire [7:0] dout; // pixel value from Bee.mem
BeeSprite BeeDisplay (.xx(x),.yy(y),.aactive(active),
                      .BSpriteOn(BeeSpriteOn),.dataout(dout),.BR(btnR),
                      .BL(btnL),.Pclk(PixCLK));

// instantiate AlienSprites code
wire Alien1SpriteOn; // 1=on, 0=off
wire Alien2SpriteOn; // 1=on, 0=off
wire Alien3SpriteOn; // 1=on, 0=off
wire [7:0] A1dout; // pixel value from Alien1.mem
wire [7:0] A2dout; // pixel value from Alien2.mem
wire [7:0] A3dout; // pixel value from Alien3.mem
AlienSprites ADisplay (.xx(x),.yy(y),.aactive(active),
                       .A1SpriteOn(Alien1SpriteOn),.A2SpriteOn(Alien2SpriteOn),
                       .A3SpriteOn(Alien3SpriteOn),.A1dataout(A1dout),
                       .A2dataout(A2dout),.A3dataout(A3dout),.Pclk(PixCLK));

// load colour palette
reg [7:0] palette [0:191]; // 8 bit values from the 192 hex entries in the colour palette
reg [7:0] COL = 0; // background colour palette value
initial begin
    $readmemh("pal24bit.mem", palette); // load 192 hex values into "palette"
end

// draw on the active area of the screen
always @ (posedge PixCLK)
begin
    if (active)
        begin
            if (BeeSpriteOn==1)
                begin
                    RED <= (palette[(dout*3)])>>4; // RED bits(7:4) from colour palette
```

```
                    GREEN <= (palette[(dout*3)+1])>>4; // GREEN bits(7:4) from colour palette
                    BLUE <= (palette[(dout*3)+2])>>4; // BLUE bits(7:4) from colour palette
                end
            else
            if (Alien1SpriteOn==1)
                begin
                    RED <= (palette[(A1dout*3)])>>4; // RED bits(7:4) from colour palette
                    GREEN <= (palette[(A1dout*3)+1])>>4; // GREEN bits(7:4) from colour palette
                    BLUE <= (palette[(A1dout*3)+2])>>4; // BLUE bits(7:4) from colour palette
                end
            else
            if (Alien2SpriteOn==1)
                begin
                    RED <= (palette[(A2dout*3)])>>4; // RED bits(7:4) from colour palette
                    GREEN <= (palette[(A2dout*3)+1])>>4; // GREEN bits(7:4) from colour palette
                    BLUE <= (palette[(A2dout*3)+2])>>4; // BLUE bits(7:4) from colour palette
                end
            else
            if (Alien3SpriteOn==1)
                begin
                    RED <= (palette[(A3dout*3)])>>4; // RED bits(7:4) from colour palette
                    GREEN <= (palette[(A3dout*3)+1])>>4; // GREEN bits(7:4) from colour palette
                    BLUE <= (palette[(A3dout*3)+2])>>4; // BLUE bits(7:4) from colour palette
                end
            else
                begin
                    RED <= (palette[(COL*3)])>>4; // RED bits(7:4) from colour palette
                    GREEN <= (palette[(COL*3)+1])>>4; // GREEN bits(7:4) from colour palette
                    BLUE <= (palette[(COL*3)+2])>>4; // BLUE bits(7:4) from colour palette
                end
        end
    else
            begin
                RED <= 0; // set RED, GREEN & BLUE
                GREEN <= 0; // to "0" when x,y outside of
                BLUE <= 0; // the active display area
            end
end
endmodule
```

02

Double click on "vga640x480 (vga640x480.v)" in the Sources (design) panel to open the module

Remove all the code in the "vga640x480.v" box and copy & paste the code from either the "vga640x480.v" file you downloaded or from below, into the "vga640x480.v" code box

```
//-------------------------------------------------
// vga640x480 Module : Digilent Basys 3
// BeeInvaders Tutorial 3 : Onboard clock 100MHz
// VGA Resolution 640x480 @ 60Hz : Pixel Clock 25MHz
//-------------------------------------------------
`timescale 1ns / 1ps

// Setup vga640x480 Module
module vga640x480(
    input wire i_clk, // 100MHz onboard clock
    input wire i_rst, // reset
    output wire o_hsync, // horizontal sync
    output wire o_vsync, // vertical sync
    output wire o_active, // high during active pixel drawing
    output wire [9:0] o_x, // current pixel x position
    output wire [9:0] o_y, // current pixel y position
    output reg pix_clk // 25MHz pixel clock
    );

    // setup VGA timings
    //----------------------------------
    // VGA 640x480 Horizontal Timing (line)
    localparam HSYNCSTART = 16; // horizontal sync start
    localparam HSYNCEND = 16 + 96; // horizontal sync end
    localparam HACTIVESTART = 16 + 96 + 48; // horizontal active start
    localparam HACTIVEEND = 16 + 96 + 48 + 640; // horizontal active end
    reg [9:0] H_SCAN; // horizontal line position

    // VGA 640x480 Vertical timing (frame)
    localparam VSYNCSTART = 10; // vertical sync start
    localparam VSYNCEND = 10 + 2; // vertical sync end
    localparam VACTIVESTART = 10 + 2 + 33; // vertical active start
    localparam VACTIVEEND = 10 + 2 + 33 + 480; // vertical active end
    reg [9:0] V_SCAN; // vertical screen position

    // set sync signals to low (active) or high (inactive)
    assign o_hsync = ~((H_SCAN >= HSYNCSTART) & (H_SCAN < HSYNCEND));
```

```
    assign o_vsync = ~((V_SCAN >= VSYNCSTART) & (V_SCAN < VSYNCEND));

    // set x and y values
    assign o_x = (H_SCAN < HACTIVESTART) ? 0 : (H_SCAN - HACTIVESTART);
    assign o_y = (V_SCAN < VACTIVESTART) ? 0 : (V_SCAN - VACTIVESTART);

    // set active high during active area
    assign o_active = ~((H_SCAN < HACTIVESTART) | (V_SCAN < VACTIVESTART));

    // generate 25MHz pixel clock using a "Fractional Clock Divider"
    reg [15:0] counter1;
    always @(posedge i_clk)
        {pix_clk, counter1} <= counter1 + 16'h4000; // divide 100MHz by 4 = 25MHz : (2^16)/4 = 16384 decimal or
4000 hex

    // check for reset / create frame loop
    always @ (posedge i_clk)
    begin
        // check for reset button pressed
        if (i_rst) // jump to start of a frame and reset registers
        begin
            H_SCAN <= 0;
            V_SCAN <= 0;
        end

        // loop through a full screen
        if (pix_clk)
        begin
            if (H_SCAN == HACTIVEEND)  // if at the end of a line update registers
            begin
                H_SCAN <= 0;
                V_SCAN <= V_SCAN + 1;
            end
            else
                H_SCAN <= H_SCAN + 1; // else increment horizontal counter

            if (V_SCAN == VACTIVEEND)  // if at the end of a screen reset vertical counter
                V_SCAN <= 0;
        end
    end
endmodule
```

03 Double click on "BeeSprite (BeeSprite.v)" in the Sources (design) panel to open the module

Remove all the code in the "BeeSprite.v" box and copy & paste the code from either the "BeeSprite.v" file you downloaded or from below, into the "BeeSprite.v" code box

```
//----------------------------------------------------
// BeeSprite Module : Digilent Basys 3
// BeeInvaders Tutorial 3 : Onboard clock 100MHz
// VGA Resolution 640x480 @ 60Hz : Pixel Clock 25MHz
//----------------------------------------------------
`timescale 1ns / 1ps

// Setup BeeSprite Module
module BeeSprite(
    input wire [9:0] xx, // current x position
    input wire [9:0] yy, // current y position
    input wire aactive, // high during active pixel drawing
    output reg BSpriteOn, // 1=on, 0=off
    output wire [7:0] dataout, // 8 bit pixel value from Bee.mem
    input wire BR, // right button
    input wire BL, // left button
    input wire Pclk // 25MHz pixel clock
    );

    // instantiate BeeRom code
    reg [9:0] address; // 2^10 or 1024, need 34 x 27 = 918
    BeeRom BeeVRom (.i_addr(address),.i_clk2(Pclk),.o_data(dataout));

    // setup character positions and sizes
    reg [9:0] BeeX = 297; // Bee X start position
    reg [8:0] BeeY = 433; // Bee Y start position
    localparam BeeWidth = 34; // Bee width in pixels
    localparam BeeHeight = 27; // Bee height in pixels

    always @ (posedge Pclk)
    begin
        if (xx==639 && yy==479)
            begin // check for left or right button pressed
                if (BR == 1 && BeeX<640-BeeWidth)
```

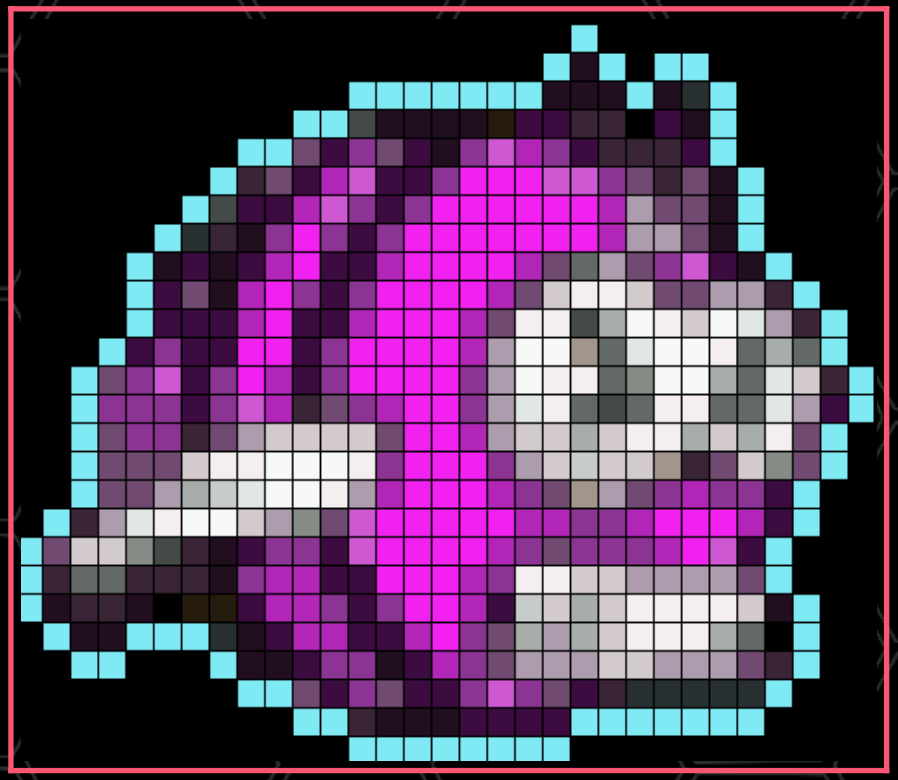
```
                    BeeX<=BeeX+1;
                if (BL == 1 && BeeX>1)
                    BeeX<=BeeX-1;
            end
        if (aactive)
            begin // check if xx,yy are within the confines of the Bee character
                if (xx==BeeX-1 && yy==BeeY)
                    begin
                        address <= 0;
                        BSpriteOn <=1;
                    end
                if ((xx>BeeX-1) && (xx<BeeX+BeeWidth) && (yy>BeeY-1) && (yy<BeeY+BeeHeight))
                    begin
                        address <= address + 1;
                        BSpriteOn <=1;
                    end
                else
                    BSpriteOn <=0;
            end
    end
endmodule
```

04

Follow the instructions from Tutorial 2 to create the 3 different Aliens in Gimp / HxD, calling each of them "Alien1.mem", "Alien2.mem" and "Alien3.mem" (remember to convert the "Sprite Sheet.xcf" to 64 colours

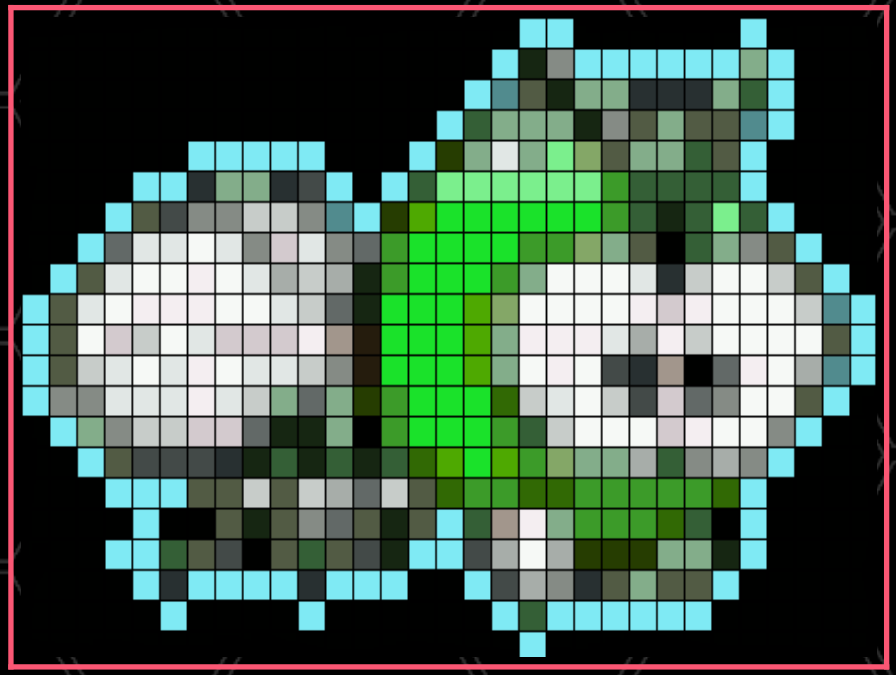
Alien1.mem : Sprite Size 31 x 26

```
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 01 39 00 39 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 39 39 39 39 39 39 39 01 01 01 39 01 09 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 39 0B 01 01 01 01 03 02 02 06 06 00 02 01 39 00 00 00 00 00 00 00 00 00 00 00 00 00 39 39 11 02 0D 11 02 01
0D 1B 0E 0D 02 06 06 06 02 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 06 11 02 0E 1B 02 02 0D 16 16 16 1B 1B 0D 11 06 11 01 39 00 00 00 00 00 00 00 00 00 00 00 39 0B 02 02 0E 1B 0D 02 0D 16 16 16 16 16 16 0E 29 11
11 01 39 00 00 00 00 00 00 00 00 00 00 00 39 09 06 01 0D 16 0D 02 0D 16 16 16 16 16 16 16 0E 29 29 11 01 39 00 00 00 00 00 00 00 00 00 00 00 39 01 02 01 02 0E 16 02 02 0E 16 16 16 16 0E 11 18 29 11 0D 1B 02 01 39 00 00 00 00 00
00 00 00 39 02 11 01 0E 16 0D 02 0D 16 16 16 16 0E 11 36 3E 3E 36 11 11 29 29 06 39 00 00 00 00 00 00 39 02 02 02 0E 16 02 02 0E 16 16 16 0E 11 3E 3E 0B 2C 3F 3E 36 3F 3C 29 06 39 00 00 00 00 39 02 0D 02 02 16
16 02 0D 16 16 16 16 0E 29 3F 3F 25 18 3C 3F 3F 3E 18 2C 18 39 00 00 00 39 11 0D 1B 02 0D 16 0E 02 0D 16 16 16 16 0D 29 3F 3E 3E 18 1F 3F 3F 2C 18 3C 36 06 39 00 00 39 0D 0D 0D 02 0D 1B 0E 06 11 0D 0E 16 16 0D
29 3C 3E 18 0B 18 3E 3E 18 18 3C 29 02 39 00 00 39 11 0D 0D 06 11 29 36 36 36 36 11 16 16 0E 29 36 36 2C 36 3E 3E 2C 36 2C 3E 11 39 00 00 00 39 11 11 11 36 3E 3E 3F 3F 3F 3E 0D 16 16 16 0D 29 36 34 36 36 25 06
11 36 1F 11 39 00 00 00 39 11 11 29 2C 34 3C 3F 3F 3E 29 0E 16 16 16 0E 0D 11 25 29 11 0D 0E 0D 0D 02 39 00 00 00 39 06 29 3C 3E 3F 3F 36 29 1F 11 1B 16 16 16 16 16 0E 0E 0D 0D 0E 16 16 16 0E 02 39 00 00 39 11
36 36 1F 0B 06 01 02 0D 0D 02 1B 16 16 16 16 0E 0D 11 0D 0D 0D 0E 16 1B 02 39 00 00 00 39 06 18 18 06 06 06 01 0D 0E 0E 02 02 16 16 16 0E 0D 3E 3E 36 36 29 29 29 11 39 00 00 00 39 01 06 06 01 00 03 03 02 0E
0E 0D 02 0D 16 16 0E 02 34 36 2C 36 3E 3E 3E 3E 36 01 39 00 00 00 39 01 01 39 39 39 09 01 02 0E 0E 02 02 0E 16 0D 11 2C 29 2C 36 3E 3E 3E 2C 18 00 39 00 00 00 00 00 39 39 00 00 00 39 01 01 02 0D 0D 01 02 0E 0D 11
29 29 29 36 36 29 29 29 11 06 39 00 00 00 00 00 00 00 00 00 00 00 39 39 11 02 0D 11 02 02 0D 1B 0D 11 02 06 09 09 09 09 09 39 00 00 00 00 00 00 00 00 00 00 00 00 00 39 39 06 01 01 01 02 02 02 02 39 39 39 39 39 39
39 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 39 39 39 39 39 39 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```



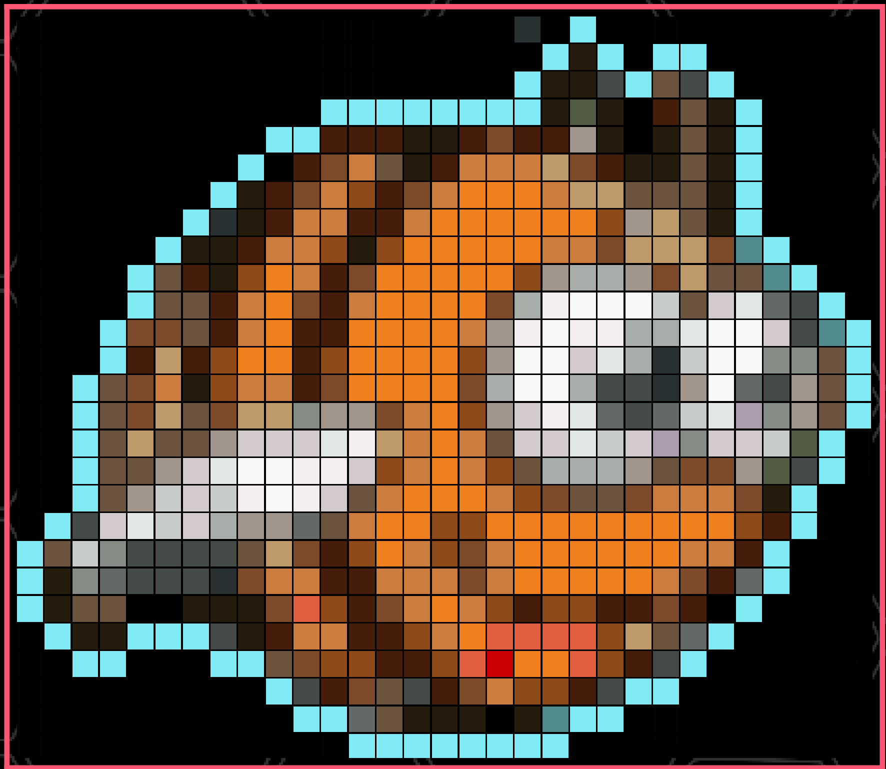
Alien2.mem : Sprite Size 31 x 21

```
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 39 00 00 00 00 00 00 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 04 1F 39 39 39 39 39 39 2A 39 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 39 1D 15 04 2A 2A 09 09 09 2A 0F 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 0F 2A 2A 2A 04 1F 15 2A 15 15 1D 39 00 00 00 00 00 00 00 00 00 39 39 39 39 39 00 00 00 39 0A
2A 3C 2A 37 26 15 2A 2A 0F 15 39 00 00 00 00 00 00 00 00 00 39 39 09 2A 2A 09 0B 39 00 39 0F 37 37 37 37 37 37 1E 0F 0F 0F 0F 39 00 00 00 00 00 00 00 39 15 0B 1F 1F 34 34 1F 1D 39 0A 20 2B 2B 2B 2B 2B 2B 1E 0F 04
0F 37 0F 39 00 00 00 00 00 00 39 18 3C 3C 3F 3C 1F 36 3C 1F 18 1E 2B 2B 2B 2B 1E 1E 26 2A 15 00 0F 2A 1F 15 39 00 00 00 39 15 3C 3F 3F 3E 3F 3C 2C 34 2C 04 1E 2B 2B 2B 1E 2A 3F 3F 3F 3C 09 34 3F 3F 34 15 39 00 39
15 3C 3F 3E 3E 3E 3F 3F 3C 34 18 04 2B 2B 2B 20 26 3F 3F 3F 3F 3E 36 3E 3F 3F 3F 34 1D 39 39 15 3F 36 34 3F 3C 36 36 36 3E 25 03 2B 2B 2B 20 2A 3E 3E 3E 3C 2C 3E 34 3F 3F 3F 3F 15 39 39 15 34 3C 3F 3C 3E 3F 3C
3C 34 1F 03 2B 2B 2B 20 2A 3F 3E 3F 0B 09 25 00 18 3E 3F 2C 1D 39 39 1F 1F 3C 3C 3C 3E 3C 34 2A 18 2A 0A 1E 2B 2B 2B 12 34 3C 3F 34 0B 36 18 1F 3F 3F 15 39 00 00 39 2A 1F 34 34 36 36 18 04 04 2A 00 1E 2B 2B 2B
1E 0F 34 3F 3F 3F 36 3E 3F 3F 1F 39 00 00 00 00 39 15 0B 0B 0B 09 04 0F 04 0F 04 0F 12 20 2B 20 1E 26 2A 2A 2C 0F 1F 2C 1F 39 00 00 00 00 00 00 39 39 39 15 15 34 15 34 2C 18 34 15 12 1E 1E 12 12 1E 1E 1E 1E 1E
12 39 00 00 00 00 00 00 00 00 39 00 00 15 04 15 1F 18 15 04 15 39 0F 25 3E 2A 1E 1E 1E 12 0F 00 39 00 00 00 00 00 00 00 39 39 0F 15 0B 00 15 0F 15 0B 04 39 39 0B 2C 3F 2C 0A 0A 0A 2A 2A 04 39 00 00 00 00 00 00
00 00 39 09 39 39 39 39 09 39 39 39 00 00 39 0B 2C 1F 09 15 2A 15 15 39 00 00 00 00 00 00 00 00 00 39 00 00 00 00 00 39 00 00 00 00 00 00 39 0F 39 39 39 39 39 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 39 00 00 00 00 00 00 00 00 00 00 00 00 00
```



Alien3.mem : Sprite Size 31 x 27

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 09 00 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 03 39 00 39 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 39 03 03 0B 39 14 0B 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 39 39 39 39 39 39 39 03 15 03 00 05 14 03 39 00 00 00 00 00 00 00 00 00 00 00 00 00 39 39 05 05 05 03 03
05 10 05 05 25 03 00 03 14 03 39 00 00 00 00 00 00 00 00 00 00 00 00 39 00 05 10 21 14 03 05 21 21 21 27 10 05 03 03 14 03 39 00 00 00 00 00 00 00 00 00 00 00 39 03 05 10 21 13 05 10 21 24 24 24 21 27 27 14 14
14 03 39 00 00 00 00 00 00 00 00 00 00 39 09 03 05 21 21 05 05 21 24 24 24 24 24 24 13 25 27 14 03 39 00 00 00 00 00 00 00 00 00 39 03 03 05 21 21 13 03 13 24 24 24 24 24 21 21 10 27 27 27 10 1D 39 00 00 00 00
00 00 00 39 14 05 03 13 24 21 05 10 24 24 24 24 24 13 25 2C 2C 25 10 27 14 14 1D 39 00 00 00 00 00 00 39 14 14 05 21 24 10 05 21 24 24 24 24 10 2C 3E 3F 3F 3F 34 14 36 3C 18 0B 39 00 00 00 00 39 10 10 14 05 21
24 05 05 24 24 24 24 21 25 3E 3F 3E 3E 2C 2C 3C 3F 3F 36 0B 1D 39 00 00 00 39 05 27 05 13 24 24 05 13 24 24 24 24 13 25 3F 3F 36 3C 2C 09 34 3F 3F 1F 1F 14 39 00 00 39 14 10 21 03 13 21 21 05 10 24 24 24 24 10
2C 3F 3F 2C 0B 0B 09 25 3F 18 0B 25 14 39 00 00 39 14 10 27 14 10 27 27 1F 25 25 10 21 24 13 25 36 3E 3C 18 0B 18 34 3C 29 1F 25 14 39 00 00 39 14 27 14 14 25 36 36 36 3C 3E 27 21 24 21 14 36 36 3C 34 36 29 1F
36 36 34 15 39 00 00 00 39 14 14 25 36 3C 3F 3F 3E 3E 36 13 21 24 21 13 14 2C 2C 2C 25 14 10 10 25 15 0B 39 00 00 00 39 14 25 34 36 34 3E 3F 3E 36 14 21 24 24 24 21 13 10 14 14 10 21 21 21 10 03 39 00 00 00 39
0B 36 3C 34 36 2C 25 25 18 14 21 24 24 13 13 24 24 24 24 24 24 24 24 24 13 05 39 00 00 39 14 34 1F 0B 0B 0B 0B 14 27 10 05 13 24 21 13 10 21 24 24 24 24 24 24 21 21 05 39 00 00 00 39 03 1F 18 0B 0B 0B 09 10 21
21 05 05 21 21 21 10 21 24 24 24 24 24 21 10 05 18 39 00 00 00 39 03 14 14 00 00 03 03 03 10 1A 13 05 10 21 24 21 13 05 13 13 05 05 10 05 00 39 00 00 00 00 00 39 03 03 39 39 39 0B 03 05 21 21 05 05 13 21 24 1A
1A 1A 1A 13 27 14 18 39 00 00 00 00 00 00 00 39 39 00 00 00 39 39 14 10 13 13 05 05 13 1A 08 24 24 1A 13 05 0B 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 0B 05 10 14 0B 05 10 21 13 13 05 0B 39 39 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 39 18 14 03 03 03 00 03 1D 39 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 39 39 39 39 39 39 39 00 00 00 00 00 00 00 00 00 00 00



05

Copy and paste the 3 files ("Alien1.mem", "Alien2.mem" and "Alien3.mem") into the project folder;

Path: BeeInvaders\Tutorials Basys 3\WIP\WIP.srcs\sources_1\new

In Vivado right click on "Design Sources" and left click on "Add Sources"

Select "Add or create design sources" and click "Next"

Select "Add Files" and navigate to the "Alien1.mem" which you copied to "BeeInvaders\Tutorials Basys 3\WIP\WIP.srcs\sources_1\new" folder. Select the file and click "OK" and then "Finish"

Do the same for "Alien2.mem" and "Alien3.mem"

06

Right click on "Design Sources" and left click on "Add Sources"

Select "Add or create design sources" and click "Next"

Select "+" and click on "Create File" or click on the "Create File" button

Make sure "Verilog" is the "File Type:", enter "Alien1Rom" in the box entitled "File name:", ensure "Local to Project" is the "File location:" and click "OK"

Select "Finish" at the next screen, "OK" at the following screen and "Yes" at the last screen

Double click on "Alien1Rom (Alien1Rom.v)" in the Sources (design) panel to open the module

Remove all the code in the "Alien1Rom.v" box and copy & paste the code from either the "Alien1Rom.v" file you downloaded or from below, into the "Alien1Rom.v" code box

```
//----------------------------------------------------
// Alien1Rom Module - Single Port ROM : Digilent Basys 3
// BeeInvaders Tutorial 3 : Onboard clock 100MHz
// VGA Resolution 640x480 @ 60Hz : Pixel Clock 25MHz
//----------------------------------------------------
`timescale 1ns / 1ps

// Setup Alien1Rom Module
module Alien1Rom(
    input wire [9:0] i_A1addr, // (9:0) or 2^10 or 1024, need 31 x 26 = 806
    input wire i_clk2,
    output reg [7:0] o_A1data // (7:0) 8 bit pixel value from Alien1.mem
    );
```

```
    (*ROM_STYLE="block"*) reg [7:0] A1memory_array [0:805]; // 8 bit values for 806 pixels of Alien1 (31
x 26)

    initial begin
            $readmemh("Alien1.mem", A1memory_array);
    end

    always @ (posedge i_clk2)
            o_A1data <= A1memory_array[i_A1addr];
endmodule
```

07 Do the same for Alien2Rom;

```
//-------------------------------------------------------
// Alien2Rom Module - Single Port ROM : Digilent Basys 3
// BeeInvaders Tutorial 3 : Onboard clock 100MHz
// VGA Resolution 640x480 @ 60Hz : Pixel Clock 25MHz
//-------------------------------------------------------
`timescale 1ns / 1ps

// Setup Alien1Rom Module
module Alien2Rom(
    input wire [9:0] i_A2addr, // (9:0) or 2^10 or 1024, need 31 x 21 = 651
    input wire i_clk2,
    output reg [7:0] o_A2data // (7:0) 8 bit pixel value from Alien2.mem
    );

    (*ROM_STYLE="block"*) reg [7:0] A2memory_array [0:650]; // 8 bit values for 651 pixels of Alien2 (31 x 21)

    initial begin
            $readmemh("Alien2.mem", A2memory_array);
    end

    always @ (posedge i_clk2)
            o_A2data <= A2memory_array[i_A2addr];
endmodule
```

08 Do the same for Alien3Rom;

```
//-------------------------------------------------------
// Alien3Rom Module - Single Port ROM : Digilent Basys 3
// BeeInvaders Tutorial 3 : Onboard clock 100MHz
// VGA Resolution 640x480 @ 60Hz : Pixel Clock 25MHz
//-------------------------------------------------------
`timescale 1ns / 1ps

// Setup Alien1Rom Module
module Alien3Rom(
    input wire [9:0] i_A3addr, // (9:0) or 2^10 or 1024, need 31 x 27 = 837
    input wire i_clk2,
    output reg [7:0] o_A3data // (7:0) 8 bit pixel value from Alien3.mem
    );

    (*ROM_STYLE="block"*) reg [7:0] A3memory_array [0:836]; // 8 bit values for 837 pixels of Alien3 (31 x 27)

    initial begin
            $readmemh("Alien3.mem", A3memory_array);
    end

    always @ (posedge i_clk2)
            o_A3data <= A3memory_array[i_A3addr];
endmodule
```

09 Right click on "Design Sources" and left click on "Add Sources"

Select "Add or create design sources" and click "Next"

Select "+" and click on "Create File" or click on the "Create File" button

Make sure "Verilog" is the "File Type:", enter "AlienSprites" in the box entitled "File name:", ensure "Local to Project" is the "File location:" and click "OK"

Select "Finish" at the next screen, "OK" at the following screen and "Yes" at the last screen

Double click on " AlienSprites (AlienSprites .v)" in the Sources (design) panel to open the module

Remove all the code in the "AlienSprites .v" box and copy & paste the code from either the "AlienSprites .v" file you downloaded or from below, into the "AlienSprites .v" code box

```
//----------------------------------------------------
// AlienSprites Module : Digilent Basys 3
// BeeInvaders Tutorial 3 : Onboard clock 100MHz
// VGA Resolution 640x480 @ 60Hz : Pixel Clock 25MHz
//----------------------------------------------------
`timescale 1ns / 1ps

// Setup AlienSprites Module
module AlienSprites(
    input wire [9:0] xx, // current x position
    input wire [9:0] yy, // current y position
    input wire aactive, // high during active pixel drawing
    output reg A1SpriteOn, // 1=on, 0=off
    output reg A2SpriteOn, // 1=on, 0=off
    output reg A3SpriteOn, // 1=on, 0=off
```

```
    output wire [7:0] A1dataout, // 8 bit pixel value from Alien1.mem
    output wire [7:0] A2dataout, // 8 bit pixel value from Alien2.mem
    output wire [7:0] A3dataout, // 8 bit pixel value from Alien3.mem
    input wire Pclk // 25MHz pixel clock
    );

// instantiate Alien1Rom code
    reg [9:0] A1address; // 2^10 or 1024, need 31 x 26 = 806
    Alien1Rom Alien1VRom (.i_A1addr(A1address),.i_clk2(Pclk),.o_A1data(A1dataout));

// instantiate Alien2Rom code
    reg [9:0] A2address; // 2^10 or 1024, need 31 x 21 = 651
    Alien2Rom Alien2VRom (.i_A2addr(A2address),.i_clk2(Pclk),.o_A2data(A2dataout));

// instantiate Alien3Rom code
    reg [9:0] A3address; // 2^10 or 1024, need 31 x 27 = 837
    Alien3Rom Alien3VRom (.i_A3addr(A3address),.i_clk2(Pclk),.o_A3data(A3dataout));

// setup character positions and sizes
    reg [9:0] A1X = 135; // Alien1 X start position
    reg [9:0] A1Y = 85; // Alien1 Y start position
    localparam A1Width = 31; // Alien1 width in pixels
    localparam A1Height = 26; // Alien1 height in pixels
    reg [9:0] A2X = 135; // Alien2 X start position
    reg [9:0] A2Y = 120; // Alien2 Y start position
    localparam A2Width = 31; // Alien2 width in pixels
    localparam A2Height = 21; // Alien2 height in pixels
    reg [9:0] A3X = 135; // Alien3 X start position
    reg [9:0] A3Y = 180; // Alien3 Y start position
    localparam A3Width = 31; // Alien3 width in pixels
    localparam A3Height = 27; // Alien3 height in pixels

    reg [9:0] AoX = 0; // Offset for X Position of next Alien in row
    reg [9:0] AoY = 0; // Offset for Y Position of next row of Aliens
    reg [9:0] AcounterW = 0; // Counter to check if Alien width reached
    reg [9:0] AcounterH = 0; // Counter to check if Alien height reached
    reg [3:0] AcolCount = 11; // Number of horizontal aliens in all columns

    always @ (posedge Pclk)
    begin
        if (aactive)
            begin
                // check if xx,yy are within the confines of the Alien characters
                // Alien1
```

```
if (xx==A1X+AoX-1 && yy==A1Y+AoY)
    begin
        A1address <= 0;
        A1SpriteOn <=1;
        AcounterW<=0;
    end
if ((xx>A1X+AoX-1) && (xx<A1X+A1Width+AoX) && (yy>A1Y+AoY-1) && (yy<A1Y+A1Height+AoY))
    begin
        A1address <= A1address + 1;
        AcounterW <= AcounterW + 1;
        A1SpriteOn <=1;
        if (AcounterW==A1Width-1)
            begin
                AcounterW <= 0;
                AoX <= AoX + 40;
                if(AoX<(AcolCount-1)*40)
     A1address <= A1address - (A1Width-1);
   else
   if(AoX==(AcolCount-1)*40)
     AoX<=0;
   end
    end
else
    A1SpriteOn <=0;

// Alien2
if (xx==A2X+AoX-1 && yy==A2Y+AoY)
    begin
        A2address <= 0;
        A2SpriteOn <=1;
        AcounterW<=0;
    end
if ((xx>A2X+AoX-1) && (xx<A2X+A2Width+AoX) && (yy>A2Y+AoY-1) && (yy<A2Y+AoY+A2Height))
    begin
        A2address <= A2address + 1;
        AcounterW <= AcounterW + 1;
        A2SpriteOn <=1;
        if (AcounterW==A2Width-1)
            begin
                AcounterW <= 0;
                AoX <= AoX + 40;
                if(AoX<(AcolCount-1)*40)
     A2address <= A2address - (A2Width-1);
   else
```

```
    if(AoX==(AcolCount-1)*40)
                    begin
            AoX<=0;
            AcounterH <= AcounterH + 1;
            if(AcounterH==A2Height-1)
                                begin
                        AcounterH<=0;
                        AoY <= AoY + 30;
                        if(AoY==30)
                            begin
                              AoY<=0;
                              AoX<=0;
                                 end
                    end
                       end
                end
     end
else
    A2SpriteOn <=0;

// Alien3
if (xx==A3X+AoX-1 && yy==A3Y+AoY)
    begin
        A3address <= 0;
        A3SpriteOn <=1;
        AcounterW<=0;
        AcounterH<=0;
    end
if ((xx>A3X+AoX-1) && (xx<A3X+AoX+A3Width) && (yy>A3Y+AoY-1) && (yy<A3Y+AoY+A3Height))
    begin
        A3address <= A3address + 1;
        AcounterW <= AcounterW + 1;
        A3SpriteOn <=1;
        if (AcounterW==A3Width-1)
            begin
                AcounterW <= 0;
                AoX <= AoX + 40;
                if(AoX<(AcolCount-1)*40)
      A3address <= A3address - (A3Width-1);
   else
   if(AoX==(AcolCount-1)*40)
                    begin
            AoX<=0;
            AcounterH <= AcounterH + 1;
```

```
                    if(AcounterH==A3Height-1)
                                    begin
                              AcounterH<=0;
                              AoY <= AoY + 36;
                              if(AoY==36)
                                  begin
                                    AoY<=0;
                                    AoX<=0;
                                        end
                                end
                            end
                        end
                      end
                  else
                      A3SpriteOn <=0;
              end
      end
endmodule
```

10 The constraints file has also been updated to accommodate for the left and right buttons on the Basys 3 board

Double click on "Basys3.xdc" in the Sources (Constraints) panel to open the module

Remove all the code in "Basys3.xdc" box and copy & paste the code from either the "Basys3.xdc" file you downloaded or from below, into the "Basys3.xdc" code box

```
##-------------------------------------------------
## Constraints Module : Digilent Basys 3
## BeeInvaders Tutorial 3 : Onboard clock 100MHz
## VGA Resolution 640x480 @ 60Hz : Pixel Clock 25MHz
##-------------------------------------------------

## Clock
set_property -dict {PACKAGE_PIN W5  IOSTANDARD LVCMOS33} [get_ports {CLK}];
create_clock -add -name sys_clk_pin -period 10.00 \
    -waveform {0 5} [get_ports {CLK}];

##Buttons : Use BTNC as Reset Button (active high)
set_property -dict {PACKAGE_PIN U18 IOSTANDARD LVCMOS33} [get_ports {RESET}];
set_property PACKAGE_PIN W19 [get_ports btnL]
  set_property IOSTANDARD LVCMOS33 [get_ports btnL]
set_property PACKAGE_PIN T17 [get_ports btnR]
  set_property IOSTANDARD LVCMOS33 [get_ports btnR]

## VGA Connector
set_property -dict {PACKAGE_PIN G19 IOSTANDARD LVCMOS33} [get_ports {RED[0]}];
set_property -dict {PACKAGE_PIN H19 IOSTANDARD LVCMOS33} [get_ports {RED[1]}];
set_property -dict {PACKAGE_PIN J19 IOSTANDARD LVCMOS33} [get_ports {RED[2]}];
set_property -dict {PACKAGE_PIN N19 IOSTANDARD LVCMOS33} [get_ports {RED[3]}];
set_property -dict {PACKAGE_PIN N18 IOSTANDARD LVCMOS33} [get_ports {BLUE[0]}];
set_property -dict {PACKAGE_PIN L18 IOSTANDARD LVCMOS33} [get_ports {BLUE[1]}];
set_property -dict {PACKAGE_PIN K18 IOSTANDARD LVCMOS33} [get_ports {BLUE[2]}];
set_property -dict {PACKAGE_PIN J18 IOSTANDARD LVCMOS33} [get_ports {BLUE[3]}];
set_property -dict {PACKAGE_PIN J17 IOSTANDARD LVCMOS33} [get_ports {GREEN[0]}];
set_property -dict {PACKAGE_PIN H17 IOSTANDARD LVCMOS33} [get_ports {GREEN[1]}];
```

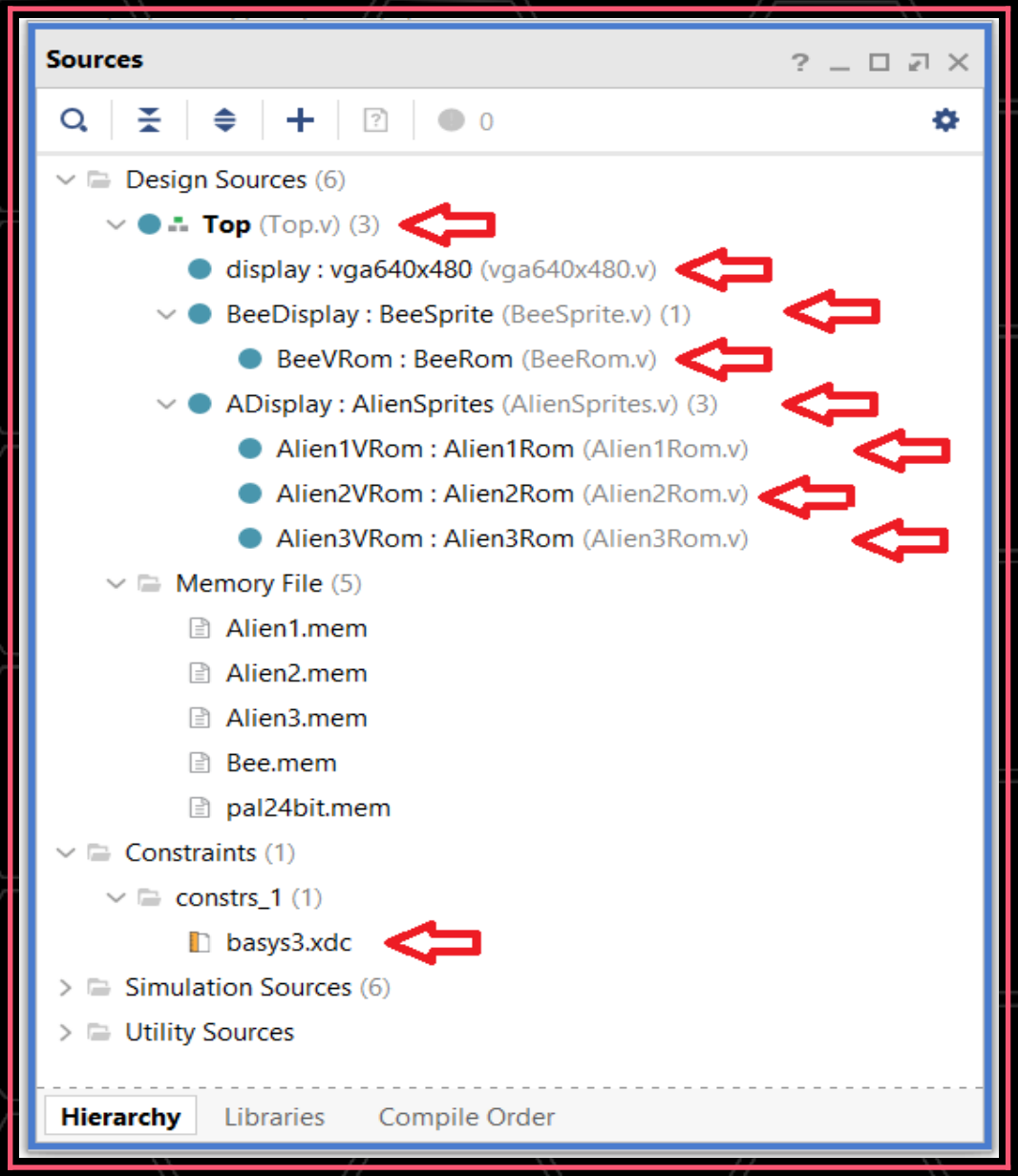
```
set_property -dict {PACKAGE_PIN G17 IOSTANDARD LVCMOS33} [get_ports {GREEN[2]}];
set_property -dict {PACKAGE_PIN D17 IOSTANDARD LVCMOS33} [get_ports {GREEN[3]}];
set_property -dict {PACKAGE_PIN P19 IOSTANDARD LVCMOS33} [get_ports {HSYNC}];
set_property -dict {PACKAGE_PIN R19 IOSTANDARD LVCMOS33} [get_ports {VSYNC}];

## Configuration options, can be used for all designs
set_property CONFIG_VOLTAGE 3.3 [current_design]
set_property CFGBVS VCCO [current_design]
```

11 "Run Synthesis" etc. and program the Basys 3 board

You should see a screen like below. Use the left / right buttons on the Basys 3 board to move the Bee

Explanation Of The Code



01 Top.v module

```
input btnR,                    // Right button : INPUT Pin T17
input btnL                     // Left button : INPUT Pin W19
```

This adds the left and right buttons as inputs to the "Top" module

```
// instantiate vga640x480 code
   wire [9:0] x;               // pixel x position: 10-bit value: 0-1023 : only need 800
   wire [9:0] y;               // pixel y position: 10-bit value: 0-1023 : only need 525
   wire active;                // high during active pixel drawing
   wire PixCLK;                // 25MHz pixel clock
   vga640x480 display (.i_clk(CLK),.i_rst(rst),.o_hsync(HSYNC),
                       .o_vsync(VSYNC),.o_x(x),.o_y(y),.o_active(active),
                       .pix_clk(PixCLK));
```

This adds the pixel clock from the "vga640x480" module to the "Top" module

The timing for updating the screen has since been found to be incorrect in Tutorial 2

Hopefully using the 25MHz pixel clock for VGA / screen updates will rectify the problem

This will also remove the related error/s produced in the "Messages" window

```
// instantiate BeeSprite code
    wire BeeSpriteOn;                   // 1=on, 0=off
    wire [7:0] dout;                    // pixel value from Bee.mem
    BeeSprite BeeDisplay (.xx(x),.yy(y),.aactive(active),
                          .BSpriteOn(BeeSpriteOn),.dataout(dout),.BR(btnR),
                          .BL(btnL),.Pclk(PixCLK));
```

This adds the pixel clock from the "Top" module to he "BeeSprite" module

```
    // instantiate AlienSprites code
    wire Alien1SpriteOn;                // 1=on, 0=off
    wire Alien2SpriteOn;                // 1=on, 0=off
    wire Alien3SpriteOn;                // 1=on, 0=off
    wire [7:0] A1dout;                  // pixel value from Alien1.mem
    wire [7:0] A2dout;                  // pixel value from Alien2.mem
    wire [7:0] A3dout;                  // pixel value from Alien3.mem
    AlienSprites ADisplay (.xx(x),.yy(y),.aactive(active),
                           .A1SpriteOn(Alien1SpriteOn),.A2SpriteOn(Alien2SpriteOn),
                           .A3SpriteOn(Alien3SpriteOn),.A1dataout(A1dout),
                           .A2dataout(A2dout),.A3dataout(A3dout),.Pclk(PixCLK));
```

This instantiates a new module called "AlienSprites"

There are 3 types of Aliens produced by the new module and their status (on/off) & 8 bit pixel values are passed to the "Top" module, all inline with the pixel clock

```
// draw on the active area of the screen
   always @ (posedge PixCLK)
   begin
      if (active)
         begin
            if (BeeSpriteOn==1)
               begin
                  RED <= (palette[(dout*3)])>>4;            // RED bits(7:4) from colour palette
                  GREEN <= (palette[(dout*3)+1])>>4;        // GREEN bits(7:4) from colour palette
                  BLUE <= (palette[(dout*3)+2])>>4;         // BLUE bits(7:4) from colour palette
               end
            else
            if (Alien1SpriteOn==1)
               begin
                  RED <= (palette[(A1dout*3)])>>4;          // RED bits(7:4) from colour palette
                  GREEN <= (palette[(A1dout*3)+1])>>4;      // GREEN bits(7:4) from colour palette
                  BLUE <= (palette[(A1dout*3)+2])>>4;       // BLUE bits(7:4) from colour palette
               end
            else
            if (Alien2SpriteOn==1)
               begin
                  RED <= (palette[(A2dout*3)])>>4;          // RED bits(7:4) from colour palette
                  GREEN <= (palette[(A2dout*3)+1])>>4;      // GREEN bits(7:4) from colour palette
                  BLUE <= (palette[(A2dout*3)+2])>>4;       // BLUE bits(7:4) from colour palette
               end
            else
            if (Alien3SpriteOn==1)
               begin
                  RED <= (palette[(A3dout*3)])>>4;          // RED bits(7:4) from colour palette
                  GREEN <= (palette[(A3dout*3)+1])>>4;      // GREEN bits(7:4) from colour palette
                  BLUE <= (palette[(A3dout*3)+2])>>4;       // BLUE bits(7:4) from colour palette
               end
            else
```

```
            begin
                RED <= (palette[(COL*3)])>>4;           // RED bits(7:4) from colour palette
                GREEN <= (palette[(COL*3)+1])>>4;       // GREEN bits(7:4) from colour palette
                BLUE <= (palette[(COL*3)+2])>>4;        // BLUE bits(7:4) from colour palette
            end
        end
    else
        begin
            RED <= 0;                                   // set RED, GREEN & BLUE
            GREEN <= 0;                                 // to "0" when x,y outside of
            BLUE <= 0;                                  // the active display area
        end
  end
endmodule
```

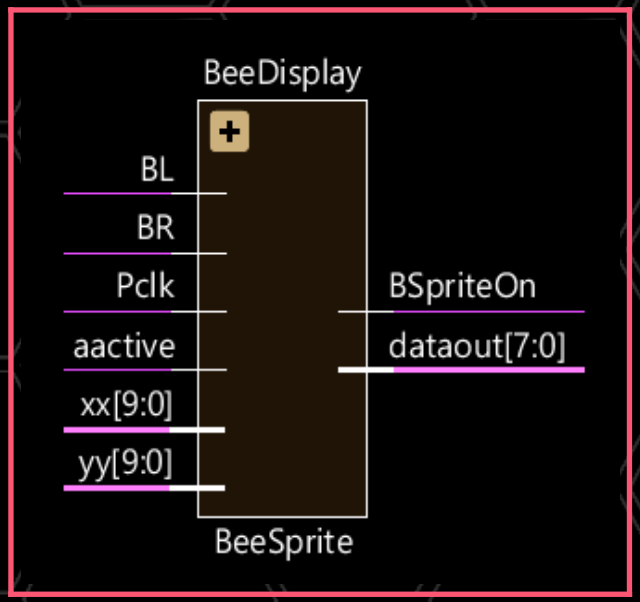
This now uses the positive edge of the pixel clock (always @ (posedge PixCLK)) instead of the 100MHz clock to draw the “Bee”, the new “Aliens” and the background

02 vga640x480.v module

```
// Setup vga640x480 Module
module vga640x480(
    input wire i_clk,                   // 100MHz onboard clock
    input wire i_rst,                   // reset
    output wire o_hsync,                // horizontal sync
    output wire o_vsync,                // vertical sync
    output wire o_active,               // high during active pixel drawing
    output wire [9:0] o_x,              // current pixel x position
    output wire [9:0] o_y,              // current pixel y position
    output reg pix_clk                  // 25MHz pixel clock
    );
```

The pixel clock has been added to the setup of the module in order that it can be passed to the "Top" module

03 BeeSprite.v module



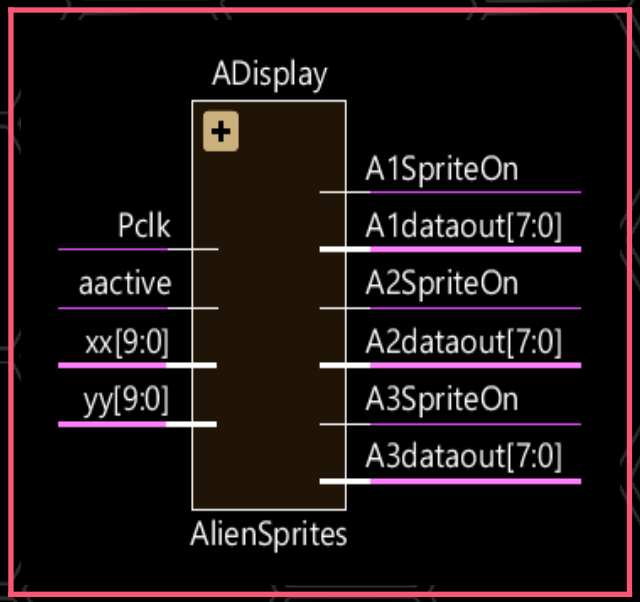

```
// Setup BeeSprite Module
module BeeSprite(
    input wire [9:0] xx,                    // current x position
    input wire [9:0] yy,                    // current y position
    input wire aactive,                     // high during active pixel drawing
    output reg BSpriteOn,                   // 1=on, 0=off
    output wire [7:0] dataout,              // 8 bit pixel value from Bee.mem
    input wire BR,                          // right button
    input wire BL,                          // left button
    input wire Pclk                         // 25MHz pixel clock
    );
```

This allows the "BeeSprite" module to process any left or right button presses and adds the pixel clock to the module

```
always @ (posedge Pclk)
   begin
      if (xx==639 && yy==479)
         begin                                          // check for left or right button pressed
            if (BR == 1 && BeeX<640-BeeWidth)
               BeeX<=BeeX+1;
            if (BL == 1 && BeeX>1)
               BeeX<=BeeX-1;
         end
      if (aactive)
```

If xx,yy are at the end of a screen update, increment / decrement accordingly the X position of Bee (Note, waiting for the xx,yy to be at the end of a screen update syncs the bee movement to the screen refresh rate)

04 AlienSprites.v module




```
`timescale 1ns / 1ps

// Setup AlienSprites Module
module AlienSprites(
    input wire [9:0] xx,                    // current x position
    input wire [9:0] yy,                    // current y position
    input wire aactive,                     // high during active pixel drawing
    output reg A1SpriteOn,                  // 1=on, 0=off
    output reg A2SpriteOn,                  // 1=on, 0=off
    output reg A3SpriteOn,                  // 1=on, 0=off
    output wire [7:0] A1dataout,            // 8 bit pixel value from Alien1.mem
    output wire [7:0] A2dataout,            // 8 bit pixel value from Alien2.mem
    output wire [7:0] A3dataout,            // 8 bit pixel value from Alien3.mem
    input wire Pclk                         // 25MHz pixel clock
    );
```

The above sets up the module and the 3 types of Aliens, their status (on/off) and 8 bit pixel values are passed to the "Top" module, all inline with the pixel clock

```
// instantiate Alien1Rom code
   reg [9:0] A1address;                                  // 2^10 or 1024, need 31 x 26 = 806
   Alien1Rom Alien1VRom (.i_A1addr(A1address),.i_clk2(Pclk),.o_A1data(A1dataout));

// instantiate Alien2Rom code
   reg [9:0] A2address;                                  // 2^10 or 1024, need 31 x 21 = 651
   Alien2Rom Alien2VRom (.i_A2addr(A2address),.i_clk2(Pclk),.o_A2data(A2dataout));

// instantiate Alien3Rom code
   reg [9:0] A3address;                                  // 2^10 or 1024, need 31 x 27 = 837
   Alien3Rom Alien3VRom (.i_A3addr(A3address),.i_clk2(Pclk),.o_A3data(A3dataout));

// setup character positions and sizes
   reg [9:0] A1X = 135;                                  // Alien1 X start position
   reg [9:0] A1Y = 85;                                   // Alien1 Y start position
   localparam A1Width = 31;                              // Alien1 width in pixels
   localparam A1Height = 26;                             // Alien1 height in pixels
   reg [9:0] A2X = 135;                                  // Alien2 X start position
   reg [9:0] A2Y = 120;                                  // Alien2 Y start position
   localparam A2Width = 31;                              // Alien2 width in pixels
   localparam A2Height = 21;                             // Alien2 height in pixels
   reg [9:0] A3X = 135;                                  // Alien3 X start position
   reg [9:0] A3Y = 180;                                  // Alien3 Y start position
   localparam A3Width = 31;                              // Alien3 width in pixels
   localparam A3Height = 27;                             // Alien3 height in pixels
```

This instantiates the 3 Alien roms and sets up the x,y coordinates for each character and their sizes

```
reg [9:0] AoX = 0;                  // Offset for X Position of next Alien in row
reg [9:0] AoY = 0;                  // Offset for Y Position of next row of Aliens
reg [9:0] AcounterW = 0;            // Counter to check if Alien width reached
reg [9:0] AcounterH = 0;            // Counter to check if Alien height reached
reg [3:0] AcolCount = 11;           // Number of horizontal aliens in all columns
```

AoX is used to draw the 11 columns of aliens at a set distance of pixels apart
AoY is used to draw the 2nd and 3rd type of aliens (each having 2 rows) at a set distance apart

AcounterW is used as a counter to check when the width of each alien has been drawn
AcounterH is used as a counter to check when the height of each alien has been drawn

AcolCount is used to determine how many aliens are in a row

```
always @ (posedge Pclk)
   begin
      if (aactive)
         begin
            // check if xx,yy are within the confines of the Alien characters
            // Alien1
            if (xx==A1X+AoX-1 && yy==A1Y+AoY)
               begin
                  A1address <= 0;
                  A1SpriteOn <=1;
                  AcounterW<=0;
               end
            if ((xx>A1X+AoX-1) && (xx<A1X+A1Width+AoX) && (yy>A1Y+AoY-1) && (yy<A1Y+A1Height+AoY))
               begin
                  A1address <= A1address + 1;
                  AcounterW <= AcounterW + 1;
```

```
            A1SpriteOn <=1;
            if (AcounterW==A1Width-1)
               begin
                  AcounterW <= 0;
                  AoX <= AoX + 40;
                  if(AoX<(AcolCount-1)*40)
                           A1address <= A1address - (A1Width-1);
                     else
                     if(AoX==(AcolCount-1)*40)
                        AoX<=0;
                  end
         end
      else
         A1SpriteOn <=0;
```

The first row of aliens (Alien1) consists of 1 row, unlike Alien2 and Alien3 which have 2 rows of each

This is very similar to the BeeSprite routine however, when AcounterW equals the width of Alien1 an offset (AoX) is incremented by 40 (the offset is added to the original aliens x coordinate), the starting point of the next alien in that row

At this point, if the 11 aliens have not been drawn the current address in the aliens memory data is decremented by the width of the alien, in order that the same data can be drawn for the next alien

If all aliens have been drawn the offset is reset to zero

This continues until xx,yy are outside the boundaries of the alien character / when the whole characters have been drawn

```
// Alien2
if (xx==A2X+AoX-1 && yy==A2Y+AoY)
    begin
       A2address <= 0;
       A2SpriteOn <=1;
       AcounterW<=0;
    end
 if ((xx>A2X+AoX-1) && (xx<A2X+A2Width+AoX) && (yy>A2Y+AoY-1) && (yy<A2Y+AoY+A2Height))
    begin
       A2address <= A2address + 1;
       AcounterW <= AcounterW + 1;
       A2SpriteOn <=1;
       if (AcounterW==A2Width-1)
          begin
             AcounterW <= 0;
             AoX <= AoX + 40;
             if(AoX<(AcolCount-1)*40)
                    A2address <= A2address - (A2Width-1);
              else
              if(AoX==(AcolCount-1)*40)
             begin
                    AoX<=0;
                    AcounterH <= AcounterH + 1;
                    if(AcounterH==A2Height-1)
                  begin
                        AcounterH<=0;
                        AoY <= AoY + 30;
                        if(AoY==30)
                           begin
                              AoY<=0;
                              AoX<=0;
                           end
                     end
             end
          end
    end
 else
    A2SpriteOn <=0;
```

Alien2 and Alien3 are drawn the same (2 rows of aliens each)

To draw the second row of aliens the counter AcounterH is used

When this equals the height of the alien an offset (AoY) is incremented (the offset is added to the original y coordinate), the starting point of the next alien in that column

05 Alien1Rom.v / Alien2Rom.v / Alien3Rom.v modules, Single Port ROMs

These modules are in the same format as the "BeeRom.v" module and load the character data from the associated memory file

Suggestions

1. Code improvements

Any improvements in the code used are most welcome. Please provide details of this for consideration in using in this tutorial

2. Errors or Mistakes

Any errors or mistakes spotted are most welcome, including incorrect explanations

3. Testbenches

I would like to include Testbenches in the tutorials. It would be most helpful to receive details / explanations of them

Tutorial 4

The next tutorial will include;

1. Moving the Alien characters from one side of the screen to the other continuously

2. How to display the 4 shields / hives (Single Port RAMs) and a 5th hive (Single Port ROM) used to hold the original data (used to restore the 4 hives to their original shape [without bullet holes] when they need to be redrawn)

3. As in previous tutorials, suggestions will be welcome / considered, including suggestions on the graphics used (have a go, it shouldn't be to difficult to produce better graphics than I have created)

Errors Found in Tutorial 2 / Amended In Tutorial 3

The timing was incorrect in respect of the output to the screen (the 100MHz clock was used instead of the 25MHz pixel clock)

To resolve this the pixel clock was extracted from the “vga640x480” module and used throughout with any code relating to the output to the screen

This also removed the timing error “messages” produced by Vivado