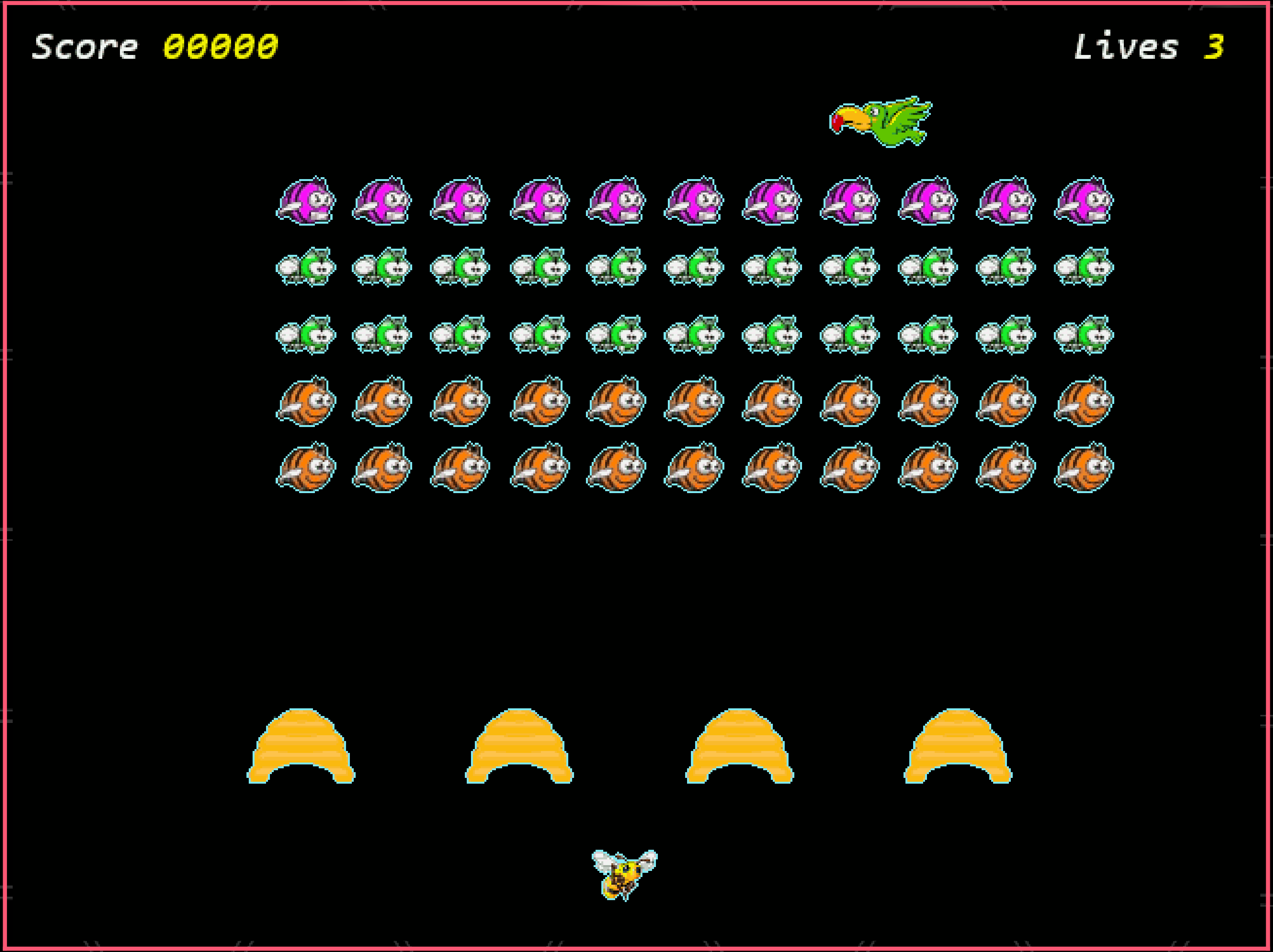
XILINX

# Project - Bee Invaders

## Tutorial 4: Moving The Aliens & Display The Hives 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 Shields / Hives

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 4 : 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));

// instantiate HiveSprites code
wire Hive1SpriteOn;     // 1=on, 0=off
wire Hive2SpriteOn;     // 1=on, 0=off
wire Hive3SpriteOn;     // 1=on, 0=off
wire Hive4SpriteOn;     // 1=on, 0=off
wire [7:0] H1dout;      // pixel value from Hive1
wire [7:0] H2dout;      // pixel value from Hive2
wire [7:0] H3dout;      // pixel value from Hive3
wire [7:0] H4dout;      // pixel value from Hive4
HiveSprites HDisplay (.xx(x),.yy(y),.aactive(active),
                      .H1SpriteOn(Hive1SpriteOn),.H2SpriteOn(Hive2SpriteOn),
                      .H3SpriteOn(Hive3SpriteOn),.H4SpriteOn(Hive4SpriteOn),
                      .H1dataout(H1dout),.H2dataout(H2dout),
                      .H3dataout(H3dout),.H4dataout(H4dout),
                      .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
            if (Hive1SpriteOn==1)
                begin
                    RED <= (palette[(H1dout*3)])>>4;            // RED bits(7:4) from colour palette
                    GREEN <= (palette[(H1dout*3)+1])>>4;        // GREEN bits(7:4) from colour palette
                    BLUE <= (palette[(H1dout*3)+2])>>4;         // BLUE bits(7:4) from colour palette
                end
                else
                if (Hive2SpriteOn==1)
                    begin
                        RED <= (palette[(H2dout*3)])>>4;        // RED bits(7:4) from colour palette
                        GREEN <= (palette[(H2dout*3)+1])>>4;    // GREEN bits(7:4) from colour palette
                        BLUE <= (palette[(H2dout*3)+2])>>4;     // BLUE bits(7:4) from colour palette
                    end
                else
                if (Hive3SpriteOn==1)
                    begin
                        RED <= (palette[(H3dout*3)])>>4;        // RED bits(7:4) from colour palette
                        GREEN <= (palette[(H3dout*3)+1])>>4;    // GREEN bits(7:4) from colour palette
                        BLUE <= (palette[(H3dout*3)+2])>>4;     // BLUE bits(7:4) from colour palette
                    end
                else
                if (Hive4SpriteOn==1)
                    begin
                        RED <= (palette[(H4dout*3)])>>4;        // RED bits(7:4) from colour palette
                        GREEN <= (palette[(H4dout*3)+1])>>4;    // GREEN bits(7:4) from colour palette
                        BLUE <= (palette[(H4dout*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 The “vga640x480” module has changed due to experiencing a left shift on the VGA screen

Double click on “vga640x480.v” in the Sources (design) panel, remove all the code in the module and copy & paste the code from either the downloaded file or from below, into the module code box

```
//--------------------------------------------------
// vga640x480 Module : Digilent Basys 3
// BeeInvaders Tutorial 4 : 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 [9:0] o_x,  // current pixel x position
    output wire [9:0] o_y,  // current pixel y position
    output wire o_active,   // high during active pixel drawing
    output reg pix_clk      // 25MHz pixel clock
    );

    // VGA 640x480 Horizontal Timing (line)
  localparam HACTIVE     = 640;                     // horizontal visible area
  localparam HBACKPORCH  = 48;                      // horizontal back porch
  localparam HFRONTPORCH = 16;                      // horizontal front porch
  localparam HSYNC       = 96;                      // horizontal sync pulse
  localparam HSYNCSTART  = 640 + 16;                // horizontal sync start
  localparam HSYNCEND    = 640 + 16 + 96 - 1;       // horizontal sync end
  localparam LINEEND     = 640 + 48 + 16 + 96 - 1;  // horizontal line end
  reg [9:0] H_SCAN;                                 // horizontal line position

  // VGA 640x480 Vertical timing (frame)
  localparam VACTIVE     = 480;                     // vertical visible area
  localparam VBACKPORCH  = 33;                      // vertical back porch
  localparam VFRONTPORCH = 10;                      // vertical front porch
  localparam VSYNC       = 2;                       // vertical sync pulse
    localparam VSYNCSTART  = 480 + 33;                // vertical sync start
```

```
        localparam VSYNCEND    = 480 + 33 + 2 - 1;         // vertical sync end
        localparam SCREENEND   = 480 + 10 + 33 + 2 - 1;    // vertical screen 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;
          assign o_y = V_SCAN;

          // set active high during active area
          assign o_active = ~(H_SCAN > HACTIVE) | (V_SCAN > VACTIVE);

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

        // check for reset / create frame loop
          always @(posedge i_clk)
          begin
            if (i_rst)
                  begin
                      H_SCAN <= 0;
                      V_SCAN <= 0;
                  end
                if (pix_clk)
                  begin
                    if (H_SCAN == LINEEND)
                        begin
                            H_SCAN <= 0;
                            V_SCAN <= V_SCAN + 1;
                        end
                    else
                        H_SCAN <= H_SCAN + 1;
                    if (V_SCAN == SCREENEND)
                        V_SCAN <= 0;
                  end
          end
      endmodule
```

**03** The “AlienSprites” module has changed to move the aliens from one side of the screen to the other

Double click on “AlienSprites.v” in the Sources (design) panel, remove all the code in the module and copy & paste the code from either the downloaded file or from below, into the module code box

```
//--------------------------------------------------
// AlienSprites Module : Digilent Basys 3
// BeeInvaders Tutorial 4 : 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
reg [1:0] Adir = 1;             // direction of aliens: 0=right, 1=left
reg [9:0] delaliens=0;          // counter to slow alien movement

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
```

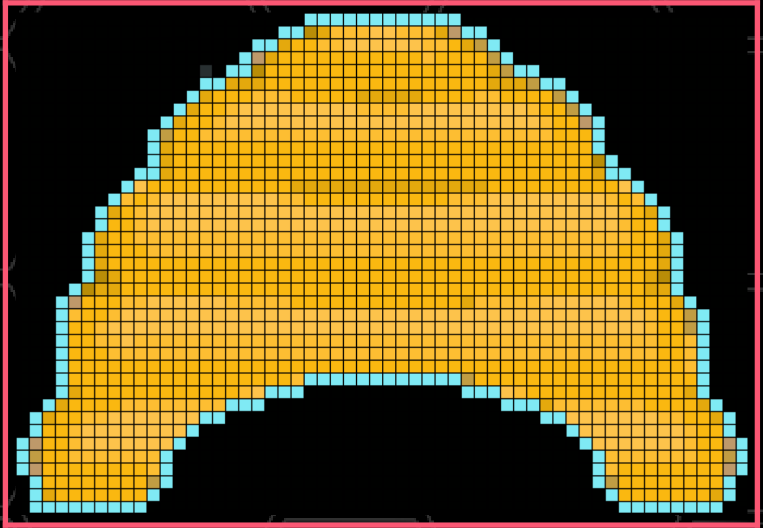
```
                else
                    A2SpriteOn <=0;

                // 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
```

```
    always @ (posedge Pclk)
    begin
        // slow down the alien movement / move aliens left or right
        if (xx==639 && yy==479)
            begin
                delaliens<=delaliens+1;
                if (delaliens>1)
                    begin
                        delaliens<=0;
                        if (Adir==1)
                            begin
                                A1X<=A1X-1;
                                A2X<=A2X-1;
                                A3X<=A3X-1;
                                if (A1X<3)
                                    Adir<=0;
                            end
                        if (Adir==0)
                            begin
                                A1X<=A1X+1;
                                A2X<=A2X+1;
                                A3X<=A3X+1;
                                if (A1X+A1Width+((AcolCount-1)*40)>636)
                                    Adir<=1;
                            end
                    end
            end
    end
endmodule
```

**04** Follow the instructions from Tutorial 2 to create the Hive in Gimp / HxD, calling it "Hive1.mem"

Hive1.mem : Sprite Size 56 x 39

```
00 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 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 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 39 39 22 2E 31 31 31 33 33 31 31 31 2E 27 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 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 39 2E 30 30 31 31 33 33 33
33 33 33 31 31 30 2E 28 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 00 00 00 00 00 00 00 39 27 2E 30 30 30 30 30 30 30 30 30 30 30 31 30 30 30 30 28 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 00 00 00 09 00 39 39 22 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 2E 28 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 00 00 00 00 00 00 39 39 2E 2E 2E 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 2E 2E 2E 28 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 00 00 00 39
2E 30 30 30 31 31 31 30 30 30 30 30 2E 2E 2E 2E 2E 30 30 30 30 31 31 30 30 30 30 28 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 39 2E 30 30 31 33 33 33 33 33 33 33 33 33 31 31
31 31 33 33 33 33 33 33 33 33 33 31 30 30 28 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 2E 30 30 31 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 31 30
30 27 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 28 30 30 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 30 30 30 39 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 39 2E 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 2E 30 30
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 22 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 39 2E 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 2E 39 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 33 30 30 30 30 30 30 30 30 30 30 30 2E 2E 2E 2E 2E 2E 2E 2E 2E 2E 2E 2E 2E 2E 2E 30 30 30 30 30 30
30 30 30 30 33 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 31 30 31 33 33 33 33 33 33 33 33 33 31 31 30 30 30 30 30 30 30 30 30 30 30 31 31 31 33 33 33 33 33 33 33 33 33 31 30 31 39 00 00 00 00 00 00 00
00 00 00 00 00 00 39 2E 30 31 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 31 30 2E 39 00 00 00 00 00 00 00 00 00 00 00 00 39 30 30 31 33 33 33 33
33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 31 30 30 39 00 00 00 00 00 00 00 00 00 00 00 39 2E 30 30 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31
31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 30 30 2E 39 00 00 00 00 00 00 00 00 00 39 2E 30 30 30 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31
31 31 31 31 30 30 30 2E 39 00 00 00 00 00 00 00 00 00 00 39 2E 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 2E 39 00 00 00 00 00
00 00 00 00 00 39 22 2E 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 2E 22 39 00 00 00 00 00 00 00 00 00 39 22 2E 2E 30 30 30 30 30 30
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 2E 2E 39 00 00 00 00 00 00 00 00 39 27 30 30 30 31 31 33 33 33 33 33 33 31 31 31 31 31 30 30 30 30 30 30 30
30 30 30 30 30 30 2E 31 31 31 31 31 33 33 33 33 33 33 31 30 30 30 2E 39 00 00 00 00 00 00 00 39 31 30 30 31 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33
33 33 33 33 33 33 31 30 30 28 39 00 00 00 00 00 00 39 30 30 31 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 31 30 30 28 39 00 00 00 00
00 00 00 39 30 30 31 31 33 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 30 30 31 39 00 00 00 00 00 00 39 30 30 30 31 31 31 31 31 31 31
31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 31 30 30 31 39 00 00 00 00 00 00 39 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 31 39 00 00 00 00 00 00 39 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 31 39 39 39 39 39 39 39 39 39 39 39 39 28 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30 30 30 39 00 00 00 00 00 00 39 2E 30 30 30 30 30 30 30 30 30 30 30 30 31 33 39 39 39 00 00 00 00 00 00 00 00 00 00 00 00 39 39 39 33 31 30 30 30 30 30 30 30 30 30 30 30 30 2E 39 00 00 00
00 00 39 2E 30 30 30 30 31 31 31 31 31 31 31 31 39 39 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 39 39 39 2E 31 31 31 31 31 31 31 30 30 30 30 2E 39 00 00 00 39 2E 30 30 31 31 33 33 33 33 33 33 33
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 39 33 33 33 33 33 33 33 31 31 30 30 2E 39 00 00 39 30 30 31 33 33 33 33 33 33 33 33 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 39 33 33 33 33 33 33 33 33 31 30 30 39 00 39 27 30 30 31 33 33 33 33 33 33 33 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
00 39 33 33 33 33 33 33 33 31 30 30 27 39 39 28 30 30 31 31 31 31 31 31 31 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 00 00 00 39 31 31 31 31 31 31 31 30 30 28 39
39 27 30 30 30 30 30 30 30 30 31 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 00 00 00 39 31 30 30 30 30 30 30 30 30 27 39 00 39 30 30 30 30 30 30 30 30 28 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 00 00 00 39 28 30 30 30 30 30 30 30 30 39 00 00 39 2E 30 30 30 30 30 30 30 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 00 00 00 00 00 39 30 30 30 30 30 30 30 2E 39 00 00 39 39 39 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 00 00 00 00 00 00 00 00 00
00 00 00 00 39 39 39 39 39 39 39 39 00 00
```



## 05

Copy and paste the “Hive1.mem” file 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 “Hive1.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”

## 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 “Hive1Ram” 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 “Hive1Ram.v” in the Sources (design) panel, remove all the code in the module and copy & paste the code from either the downloaded file or from below, into the module code box

```
//--------------------------------------------------
// Hive1Ram Module - Single Port RAM : Digilent Basys 3
// BeeInvaders Tutorial 4 : Onboard clock 100MHz
// VGA Resolution 640x480 @ 60Hz : Pixel Clock 25MHz
//--------------------------------------------------
`timescale 1ns / 1ps

// Setup Hive1Ram Module
module Hive1Ram(
    input wire [11:0] i_addr,   // (11:0) or 2^12 or 4096, need 56 x 39 = 2184
    input wire i_clk2,
    output reg [7:0] o_data,    // (7:0) 8 bit pixel value from Hive1
    input wire i_write,         // 1=write, 0=read data
    input wire [7:0] i_data     // (7:0) 8 bit pixel value to Hive1
    );
    (*ROM_STYLE="block"*) reg [7:0] H1memory_array [0:2183]; // 8 bit values for 2184 pixels of Hive1 (56
x 39)

    initial begin
            $readmemh("Hive1.mem", H1memory_array);
    end

    always @ (posedge i_clk2)
         if(i_write)
            H1memory_array[i_addr] <= i_data;
        else
            o_data <= H1memory_array[i_addr];
endmodule
```

## 07 Do the same for “Hive2Ram”;

```
//-----------------------------------------------------------
// Hive2Ram Module - Single Port RAM : Digilent Basys 3
// BeeInvaders Tutorial 4 : Onboard clock 100MHz
// VGA Resolution 640x480 @ 60Hz : Pixel Clock 25MHz
//-----------------------------------------------------------
`timescale 1ns / 1ps

// Setup Hive2Ram Module
module Hive2Ram(
    input wire [11:0] i_addr, // (11:0) or 2^12 or 4096, need 56 x 39 = 2184
    input wire i_clk2,
    output reg [7:0] o_data, // (7:0) 8 bit pixel value from Hive2
    input wire i_write, // 1=write, 0=read data
    input wire [7:0] i_data // (7:0) 8 bit pixel value to Hive2
    );

    (*ROM_STYLE="block"*) reg [7:0] H2memory_array [0:2183]; // 8 bit values for 2184 pixels of Hive2 (56 x 39)

    initial begin
            $readmemh("Hive1.mem", H2memory_array);
    end

    always @ (posedge i_clk2)
         if(i_write)
            H2memory_array[i_addr] <= i_data;
        else
            o_data <= H2memory_array[i_addr];
endmodule
```

## 08 Do the same for “Hive3Ram”;

```
//-----------------------------------------------------
// Hive3Ram Module - Single Port RAM : Digilent Basys 3
// BeeInvaders Tutorial 4 : Onboard clock 100MHz
// VGA Resolution 640x480 @ 60Hz : Pixel Clock 25MHz
//-----------------------------------------------------
`timescale 1ns / 1ps

// Setup Hive3Ram Module
module Hive3Ram(
    input wire [11:0] i_addr, // (11:0) or 2^12 or 4096, need 56 x 39 = 2184
    input wire i_clk2,
    output reg [7:0] o_data, // (7:0) 8 bit pixel value from Hive3
    input wire i_write, // 1=write, 0=read data
    input wire [7:0] i_data // (7:0) 8 bit pixel value to Hive3
    );

    (*ROM_STYLE="block"*) reg [7:0] H3memory_array [0:2183]; // 8 bit values for 2184 pixels of Hive3 (56 x 39)

    initial begin
            $readmemh("Hive1.mem", H3memory_array);
    end

    always @ (posedge i_clk2)
         if(i_write)
            H3memory_array[i_addr] <= i_data;
        else
            o_data <= H3memory_array[i_addr];
endmodule
```

## 09 Do the same for “Hive4Ram”;

```
//--------------------------------------------------------------------
// Hive4Ram Module - Single Port RAM : Digilent Basys 3
// BeeInvaders Tutorial 4 : Onboard clock 100MHz
// VGA Resolution 640x480 @ 60Hz : Pixel Clock 25MHz
//--------------------------------------------------------------------
`timescale 1ns / 1ps

// Setup Hive4Ram Module
module Hive4Ram(
    input wire [11:0] i_addr, // (11:0) or 2^12 or 4096, need 56 x 39 = 2184
    input wire i_clk2,
    output reg [7:0] o_data, // (7:0) 8 bit pixel value from Hive4
    input wire i_write, // 1=write, 0=read data
    input wire [7:0] i_data // (7:0) 8 bit pixel value to Hive4
    );

    (*ROM_STYLE="block"*) reg [7:0] H4memory_array [0:2183]; // 8 bit values for 2184 pixels of Hive4 (56 x 39)

    initial begin
            $readmemh("Hive1.mem", H4memory_array);
    end

    always @ (posedge i_clk2)
         if(i_write)
            H4memory_array[i_addr] <= i_data;
        else
            o_data <= H4memory_array[i_addr];
endmodule
```

## 10 Do the same for “Hive5Rom”, notice that this module is a ROM not a RAM

```
//-----------------------------------------------------
// Hive5Rom Module - Single Port ROM : Digilent Basys 3
// BeeInvaders Tutorial 4 : Onboard clock 100MHz
// VGA Resolution 640x480 @ 60Hz : Pixel Clock 25MHz
//-----------------------------------------------------
`timescale 1ns / 1ps

// Setup Hive5Rom Module
module Hive5Rom(
    input wire [11:0] i_addr, // (11:0) or 2^12 or 4096, need 56 x 39 = 2184
    input wire i_clk2,
    output reg [7:0] o_data // (7:0) 8 bit pixel value from Hive5
    );

    (*ROM_STYLE="block"*) reg [7:0] H5memory_array [0:2183]; // 8 bit values for 2184 pixels of Hive5 (56 x 39)

    initial begin
            $readmemh("Hive1.mem", H5memory_array);
    end

    always @ (posedge i_clk2)
            o_data <= H5memory_array[i_addr];
endmodule
```

11

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 "HiveSprites" 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

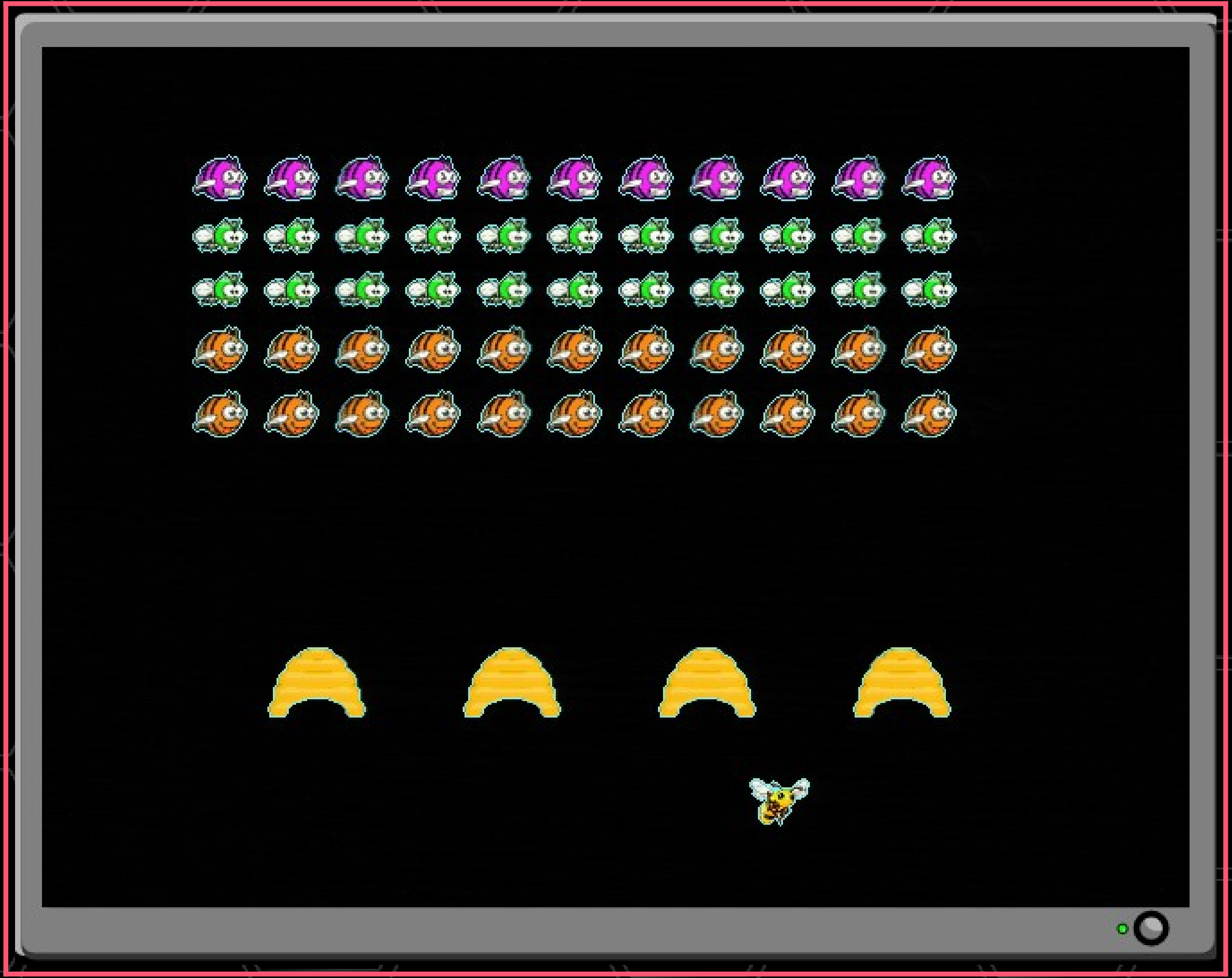
Double click on "HiveSprites.v" in the Sources (design) panel, remove all the code in the module and copy & paste the code from either the downloaded file or from below, into the module code box

```
//-----------------------------------------------------
// HiveSprites Module : Digilent Basys 3
// BeeInvaders Tutorial 4 : Onboard clock 100MHz
// VGA Resolution 640x480 @ 60Hz : Pixel Clock 25MHz
//-----------------------------------------------------
`timescale 1ns / 1ps

// Setup HiveSprites Module
module HiveSprites(
    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 H1SpriteOn,          // 1=on, 0=off
    output reg H2SpriteOn,          // 1=on, 0=off
    output reg H3SpriteOn,          // 1=on, 0=off
    output reg H4SpriteOn,          // 1=on, 0=off
    output wire [7:0] H1dataout,    // 8 bit pixel value from Hive1
    output wire [7:0] H2dataout,    // 8 bit pixel value from Hive2
    output wire [7:0] H3dataout,    // 8 bit pixel value from Hive3
    output wire [7:0] H4dataout,     // 8 bit pixel value from Hive4
    input wire Pclk                  // 25MHz pixel clock
    );

    // instantiate Hive1Ram code
    reg [11:0] H1address;            // 2^12 or 4096, need 56 x 39 = 2184
    Hive1Ram Hive1VRam (.i_addr(H1address),.i_clk2(Pclk),.o_data(H1dataout),//,
                        .i_write(0),.i_data(0));

    // instantiate Hive2Ram code
    reg [11:0] H2address;            // 2^12 or 4096, need 56 x 39 = 2184
    Hive2Ram Hive2VRam (.i_addr(H2address),.i_clk2(Pclk),.o_data(H2dataout),
                        .i_write(0),.i_data(0));

    // instantiate Hive3Ram code
    reg [11:0] H3address;            // 2^12 or 4096, need 56 x 39 = 2184
    Hive3Ram Hive3VRam (.i_addr(H3address),.i_clk2(Pclk),.o_data(H3dataout),
                       .i_write(0),.i_data(0));

    // instantiate Hive4Ram code
    reg [11:0] H4address;            // 2^12 or 4096, need 56 x 39 = 2184
    Hive4Ram Hive4VRam (.i_addr(H4address),.i_clk2(Pclk),.o_data(H4dataout),
                        .i_write(0),.i_data(0));

    // instantiate Hive5Rom code - Temporary disabled
//    reg [11:0] H5address;            // 2^12 or 4096, need 56 x 39 = 2184
//    Hive5Rom Hive5VRom (.i_addr(H5address),.i_clk2(Pclk),.o_data(H5dataout));

    // setup character positions and sizes
    reg [9:0] Hive1X = 127;          // Hive1 X start position
    reg [8:0] Hive1Y = 360;          // Hive1 Y start position
    reg [9:0] Hive2X = 237;          // Hive2 X start position
    reg [8:0] Hive2Y = 360;          // Hive2 Y start position
    reg [9:0] Hive3X = 347;          // Hive3 X start position
    reg [8:0] Hive3Y = 360;          // Hive3 Y start position
    reg [9:0] Hive4X = 457;          // Hive4 X start position
    reg [8:0] Hive4Y = 360;          // Hive4 Y start position
    localparam HiveWidth = 56;       // Hive width in pixels
    localparam HiveHeight = 39;      // Hive height in pixels

    always @ (posedge Pclk)
    begin
        if (aactive)
            // check if xx,yy are within the confines of the Hive characters
// hive1
begin
    if (xx==Hive1X-1 && yy==Hive1Y)
        begin
            H1address <= 0;
            H1SpriteOn <=1;
        end
    if ((xx>Hive1X-1) && (xx<Hive1X+HiveWidth) && (yy>Hive1Y-1) && (yy<Hive1Y+HiveHeight))
        begin
            H1address <= H1address + 1;
            H1SpriteOn <=1;
        end
    else
        H1SpriteOn <=0;

    // hive2
    if (xx==Hive2X-1 && yy==Hive2Y)
        begin
            H2address <= 0;
            H2SpriteOn <=1;
        end
    if ((xx>Hive2X-1) && (xx<Hive2X+HiveWidth) && (yy>Hive2Y-1) && (yy<Hive2Y+HiveHeight))
        begin
            H2address <= H2address + 1;
            H2SpriteOn <=1;
        end
    else
        H2SpriteOn <=0;

    // hive3
    if (xx==Hive3X-1 && yy==Hive3Y)
        begin
            H3address <= 0;
            H3SpriteOn <=1;
        end
    if ((xx>Hive3X-1) && (xx<Hive3X+HiveWidth) && (yy>Hive3Y-1) && (yy<Hive3Y+HiveHeight))
        begin
            H3address <= H3address + 1;
            H3SpriteOn <=1;
        end
    else
        H3SpriteOn <=0;

    // hive4
```

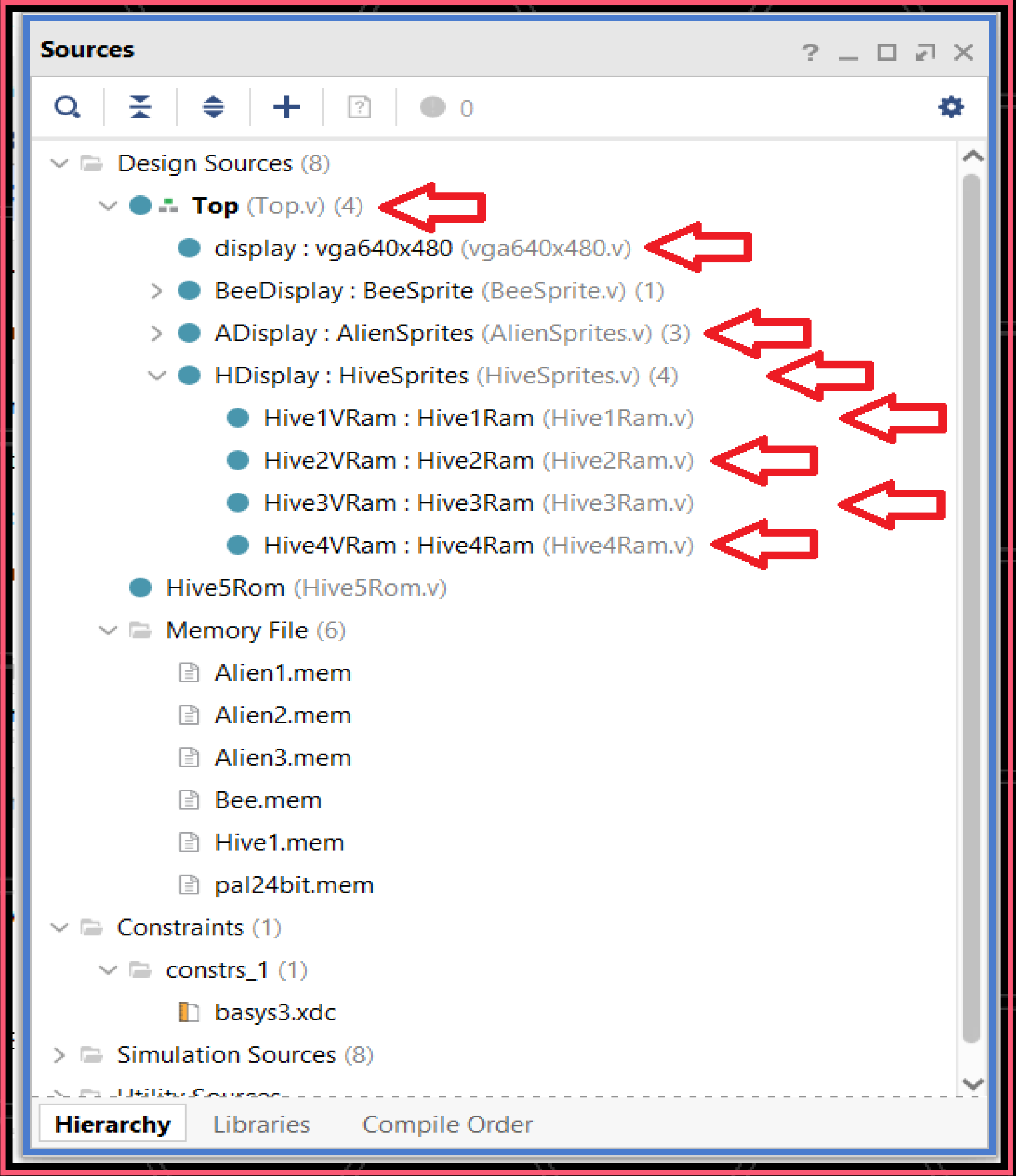
```
                if (xx==Hive4X-1 && yy==Hive4Y)
                    begin
                        H4address <= 0;
                        H4SpriteOn <=1;
                    end
                if ((xx>Hive4X-1) && (xx<Hive4X+HiveWidth) && (yy>Hive4Y-1) && (yy<Hive4Y+HiveHeight))
                    begin
                        H4address <= H4address + 1;
                        H4SpriteOn <=1;
                    end
                else
                    H4SpriteOn <=0;
            end
    end
endmodule
```

## 12 "Run Synthesis" etc. and program the Basys 3 board

You should see a screen like below



# Explanation Of The Code



Sources

Design Sources (8)

- Top (Top.v) (4)
  - display : vga640x480 (vga640x480.v)
  - BeeDisplay : BeeSprite (BeeSprite.v) (1)
  - ADisplay : AlienSprites (AlienSprites.v) (3)
  - HDisplay : HiveSprites (HiveSprites.v) (4)
    - Hive1VRam : Hive1Ram (Hive1Ram.v)
    - Hive2VRam : Hive2Ram (Hive2Ram.v)
    - Hive3VRam : Hive3Ram (Hive3Ram.v)
    - Hive4VRam : Hive4Ram (Hive4Ram.v)
- Hive5Rom (Hive5Rom.v)
- Memory File (6)
  - Alien1.mem
  - Alien2.mem
  - Alien3.mem
  - Bee.mem
  - Hive1.mem
  - pal24bit.mem

Constraints (1)

- constrs_1 (1)
  - basys3.xdc

Simulation Sources (8)

## 01 Top.v module

```
// instantiate HiveSprites code
    wire Hive1SpriteOn;                               // 1=on, 0=off
    wire Hive2SpriteOn;                               // 1=on, 0=off
    wire Hive3SpriteOn;                               // 1=on, 0=off
    wire Hive4SpriteOn;                               // 1=on, 0=off
    wire [7:0] H1dout;                                // pixel value from Hive1
    wire [7:0] H2dout;                                // pixel value from Hive2
    wire [7:0] H3dout;                                // pixel value from Hive3
    wire [7:0] H4dout;                                // pixel value from Hive4
    HiveSprites HDisplay (.xx(x),.yy(y),.aactive(active),
                .H1SpriteOn(Hive1SpriteOn),.H2SpriteOn(Hive2SpriteOn),
                .H3SpriteOn(Hive3SpriteOn),.H4SpriteOn(Hive4SpriteOn),
                .H1dataout(H1dout),.H2dataout(H2dout),
                .H3dataout(H3dout),.H4dataout(H4dout),
                .Pclk(PixCLK));
```

This instantiates a new module called;

"HiveSprites" which lets the "Top" module know when the 4 hives are "on" and require drawing on the screen using the pixel values from each of the 4 hives

```
else
if (Hive1SpriteOn==1)
    begin
        RED <= (palette[(H1dout*3)])>>4;            // RED bits(7:4) from colour palette
        GREEN <= (palette[(H1dout*3)+1])>>4;        // GREEN bits(7:4) from colour palette
        BLUE <= (palette[(H1dout*3)+2])>>4;         // BLUE bits(7:4) from colour palette
    end
else
if (Hive2SpriteOn==1)
    begin
        RED <= (palette[(H2dout*3)])>>4;            // RED bits(7:4) from colour palette
        GREEN <= (palette[(H2dout*3)+1])>>4;        // GREEN bits(7:4) from colour palette
        BLUE <= (palette[(H2dout*3)+2])>>4;         // BLUE bits(7:4) from colour palette
    end
else
if (Hive3SpriteOn==1)
    begin
        RED <= (palette[(H3dout*3)])>>4;            // RED bits(7:4) from colour palette
        GREEN <= (palette[(H3dout*3)+1])>>4;        // GREEN bits(7:4) from colour palette
        BLUE <= (palette[(H3dout*3)+2])>>4;         // BLUE bits(7:4) from colour palette
    end
else
if (Hive4SpriteOn==1)
    begin
        RED <= (palette[(H4dout*3)])>>4;            // RED bits(7:4) from colour palette
        GREEN <= (palette[(H4dout*3)+1])>>4;        // GREEN bits(7:4) from colour palette
        BLUE <= (palette[(H4dout*3)+2])>>4;         // BLUE bits(7:4) from colour palette
    end
```

This section of the code draws each hive on the screen if it is switched on

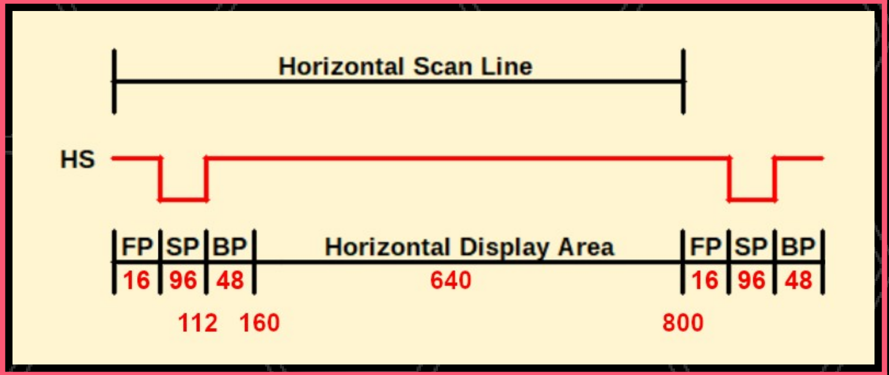
## 02 vga640x480.v module

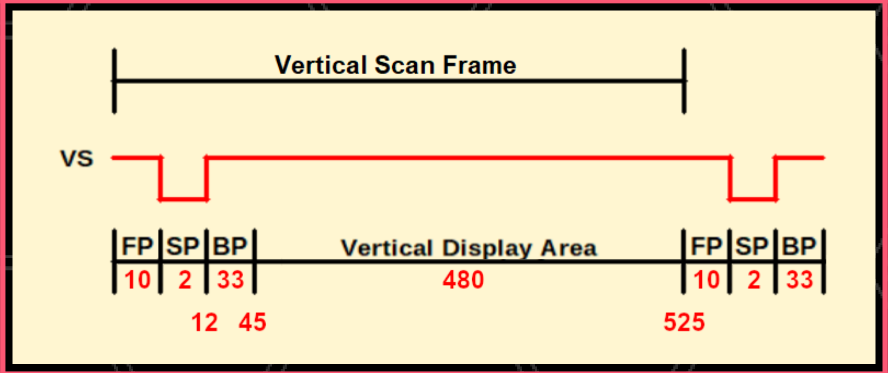
```
//----------------------------------------------------
// vga640x480 Module : Digilent Basys 3
// BeeInvaders Tutorial 4 : Onboard clock 100MHz
// VGA Resolution 640x480 @ 60Hz : Pixel Clock 25MHz
//----------------------------------------------------
`timescale 1ns / 1ps

// 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 [9:0] o_x,                   // current pixel x position
    output wire [9:0] o_y,                   // current pixel y position
    output wire o_active,                    // high during active pixel drawing
    output reg pix_clk                       // 25MHz pixel clock
    );

    // VGA 640x480 Horizontal Timing (line)
     localparam HACTIVE  = 640;              // horizontal visible area
     localparam HBACKPORCH  = 48;            // horizontal back porch
     localparam HFRONTPORCH = 16;            // horizontal front porch
     localparam HSYNC  = 96;                 // horizontal sync pulse
     localparam HSYNCSTART  = 640 + 16;      // horizontal sync start
     localparam HSYNCEND  = 640 + 16 + 96 - 1;   // horizontal sync end
     localparam LINEEND  = 640 + 48 + 16 + 96 - 1;  // horizontal line end
     reg [9:0] H_SCAN;                       // horizontal line position

     // VGA 640x480 Vertical timing (frame)
     localparam VACTIVE  = 480;              // vertical visible area
     localparam VBACKPORCH  = 33;            // vertical back porch
     localparam VFRONTPORCH = 10;            // vertical front porch
     localparam VSYNC  = 2;                  // vertical sync pulse
```

```
 localparam VSYNCSTART  = 480 + 33;                  // vertical sync start
 localparam VSYNCEND  = 480 + 33 + 2 – 1;            // vertical sync end
 localparam SCREENEND  = 480 + 10 + 33 + 2 - 1;      // vertical screen 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;
assign o_y = V_SCAN;

// set active high during active area
assign o_active = ~(H_SCAN > HACTIVE) | (V_SCAN > VACTIVE);

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

 // check for reset / create frame loop
always @(posedge i_clk)
    begin
      if (i_rst)
       begin
          H_SCAN <= 0;
          V_SCAN <= 0;
       end
```

```
        if (pix_clk)
            begin
                if (H_SCAN == LINEEND)
                  begin
                     H_SCAN <= 0;
                     V_SCAN <= V_SCAN + 1;
                  end
                else
                  H_SCAN <= H_SCAN + 1;
                if (V_SCAN == SCREENEND)
                  V_SCAN <= 0;
            end
     end
endmodule
```

| New VGA Controller | Original VGA Controller | New VGA Controller | Original VGA Controller |
|---|---|---|---|
| HSYNCSTART = 640 + 16; | HSYNCSTART = 16; | VSYNCSTART = 480 + 33; | VSYNCSTART = 10; |
| HSYNCEND = 640 + 16 + 96 - 1; | HSYNCEND = 16 + 96; | VSYNCEND = 480 + 33 + 2 - 1; | VSYNCEND = 10 + 2; |





I found that the VGA monitor I am using shifted the whole picture to the left when the code was added to move the aliens;

1. The monitor displayed a message stating “Auto Adjusting”
2. However, the screen could be moved to the right from the monitors menu controls

There are variations of the controller available on the internet and the ones I have seen either follow the technique used in my original controller;

Front Porch + Sync Pulse + Back Porch

with the horizontal or vertical active area added before or after the FP + SP + BP

Another variation was to have;

Front Porch + Horizontal or Vertical active area + Back Porch + Sync Pulse

Or even a combination of the above, which is how my latest controller appears to work best

I would be very interested to receive any information on the above or other techniques used

## 03 AlienSprites.v module

```
reg [1:0] Adir = 1;                                // direction of aliens: 0=right, 1=left
reg [9:0] delaliens=0;                             // counter to slow alien movement
```

This adds a register to control the movement direction of the aliens and a counter to add a delay period to slow down the aliens movement

```
always @ (posedge Pclk)
begin
    // slow down the alien movement / move aliens left or right
    if (xx==639 && yy==479)
        begin
            delaliens<=delaliens+1;
            if (delaliens>1)
                begin
                    delaliens<=0;
                    if (Adir==1)
                        begin
                            A1X<=A1X-1;
                            A2X<=A2X-1;
                            A3X<=A3X-1;
                            if (A1X<3)
                                Adir<=0;
                        end
                    if (Adir==0)
                        begin
                            A1X<=A1X+1;
                A2X<=A2X+1;
                A3X<=A3X+1;
                if (A1X+A1Width+((AcolCount-1)*40)>636)
                    Adir<=1;
            end
        end
    end
end
```

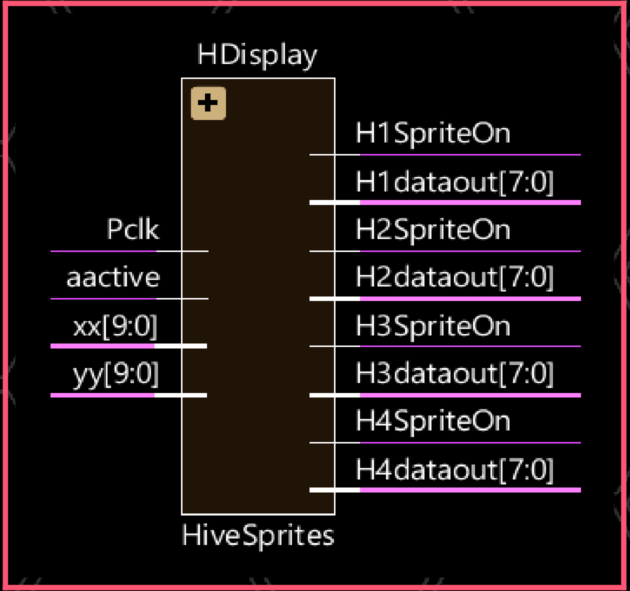
Once the counter delaliens is greater than 1 the current direction (left or right) of the aliens are checked

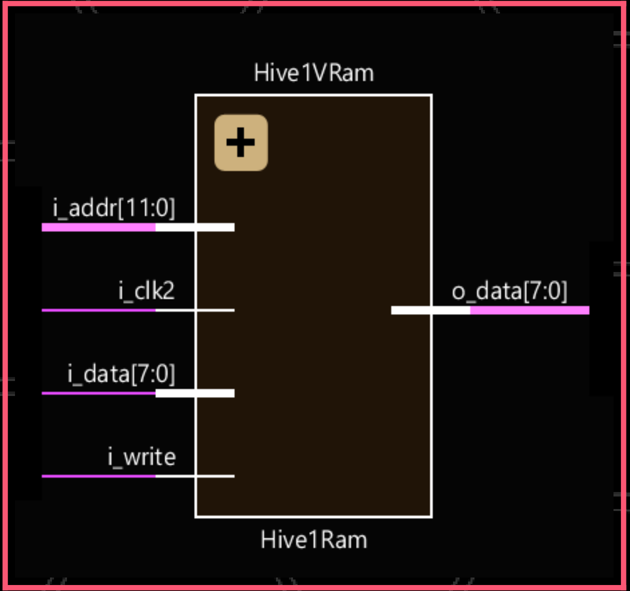
If the aliens have not reached the far left or right side of the screen the aliens position are decremented / incremented accordingly

If they have reached the edge of the screen the aliens direction is reversed

## 04 HiveSprites.v module

This displays four hives on the screen using the same technique as the “BeeSprite” however, they have been created as RAM in order that they can be reloaded with the original data when required





The Bee Rom was a Single Port ROM with two inputs (Clock & Address) and one output (Data out)

The Hive Rams are Single Port RAMs which have two additional inputs (Write Enable & Data in)

With these two extra inputs the memory array containing the Hive/s data can be modified

There is also a fifth hive created as a Single Port ROM: this will be used to restore the four on screen hives to their original format (without bullet holes)

## 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 5

The next tutorial will include;

1. Firing honey bullets from the Bee

2. Creating holes in the hives from the honey bullets

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