8bitworkshop

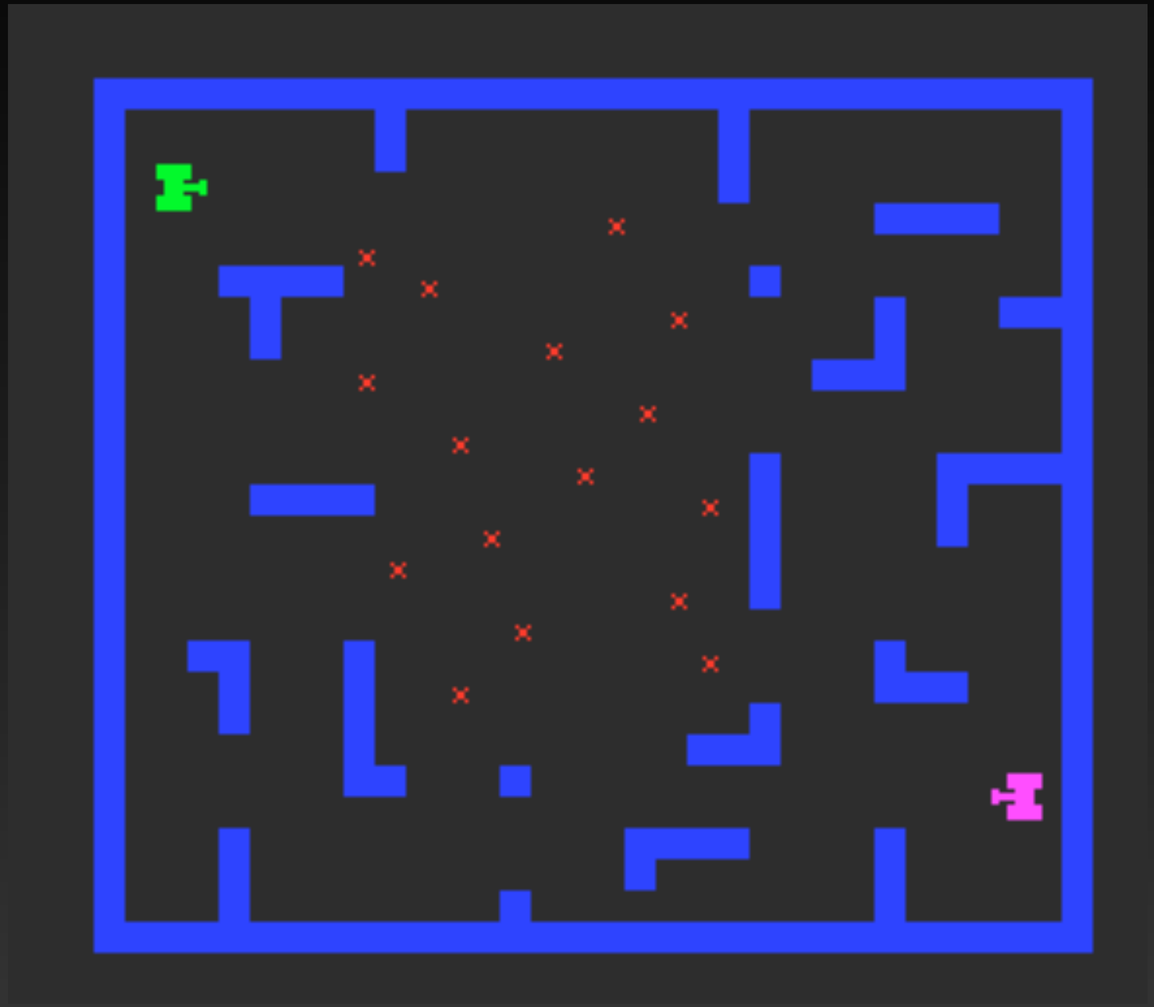
An Interactive Verilog Learning Tool

Steven Hugg, WOSET 2022

Introduction

8bitworkshop.com

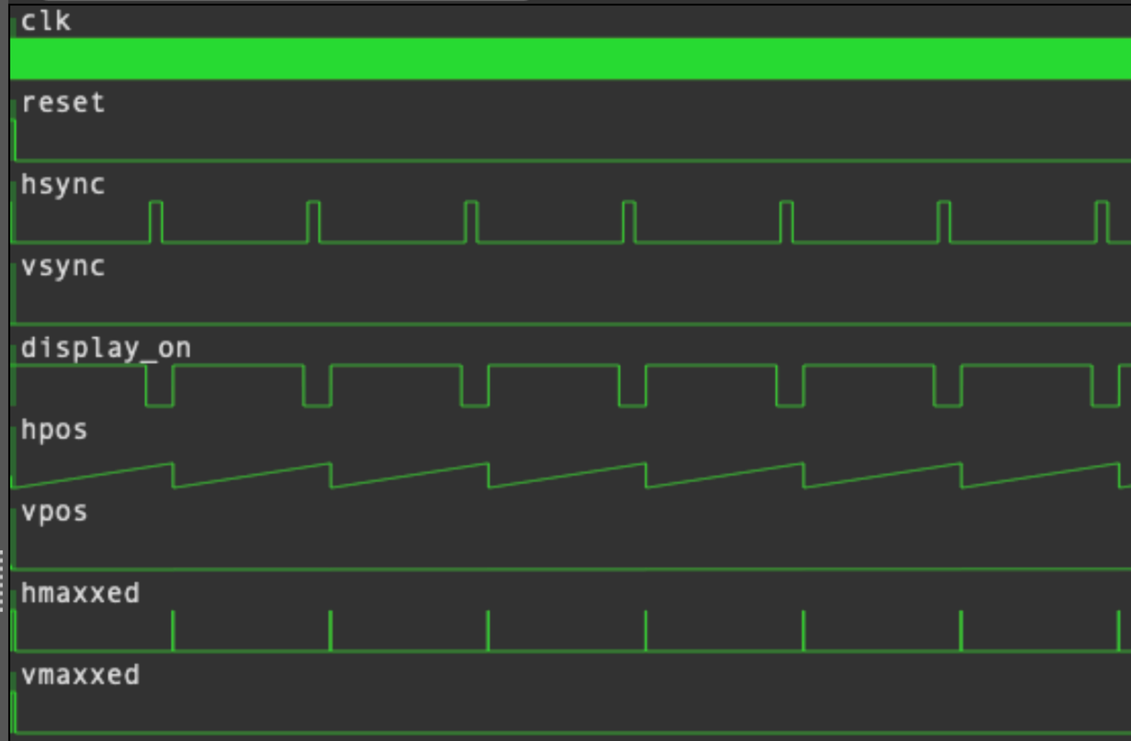
- 8bitworkshop is a web-based IDE for programming legacy 8-bit platforms
- Primarily assembler and C targeting 6502 and Z80 CPUs
- Focused on video games
- Educational and fun
- It also supports Verilog!



History of Arcade Games

- Arcade games in early 1970s were discrete logic designs
- Used binary counters to generate sync signals
- Maybe a small ROM for sequencing and bitmaps, but no code
- It’s educational and fun to reproduce them in Verilog

```
11
12  module hvsync_generator(clk, reset, hsync, vsync, display_on, hpos, vpos);
13
14    input clk;
15    input reset;
16    output reg hsync, vsync;
17    output display_on;
18    output reg [8:0] hpos;
19    output reg [8:0] vpos;
20
21    // declarations for TV-simulator sync parameters
22    // horizontal constants
23    parameter H_DISPLAY       = 256; // horizontal display width
24    parameter H_BACK          =  23; // horizontal left border (back porch)
25    parameter H_FRONT         =   7; // horizontal right border (front porch)
26    parameter H_SYNC          =  23; // horizontal sync width
27    // vertical constants
28    parameter V_DISPLAY       = 240; // vertical display height
29    parameter V_TOP           =   5; // vertical top border
30    parameter V_BOTTOM        =  14; // vertical bottom border
31    parameter V_SYNC          =   3; // vertical sync # lines
32    // derived constants
33    parameter H_SYNC_START    = H_DISPLAY + H_FRONT;
34    parameter H_SYNC_END      = H_DISPLAY + H_FRONT + H_SYNC - 1;
35    parameter H_MAX           = H_DISPLAY + H_BACK + H_FRONT + H_SYNC - 1;
36    parameter V_SYNC_START    = V_DISPLAY + V_BOTTOM;
37    parameter V_SYNC_END      = V_DISPLAY + V_BOTTOM + V_SYNC - 1;
38    parameter V_MAX           = V_DISPLAY + V_TOP + V_BOTTOM + V_SYNC - 1;
39
40    wire hmaxxed = (hpos == H_MAX) || reset;      // set when hpos is maximum
41    wire vmaxxed = (vpos == V_MAX) || reset;      // set when vpos is maximum
42
43    // horizontal position counter
44    always @(posedge clk)
45    begin
46      hsync <= (hpos>=H_SYNC_START && hpos<=H_SYNC_END);
47      if(hmaxxed)
48        hpos <= 0;
49      else
50        hpos <= hpos + 1;
51    end
```

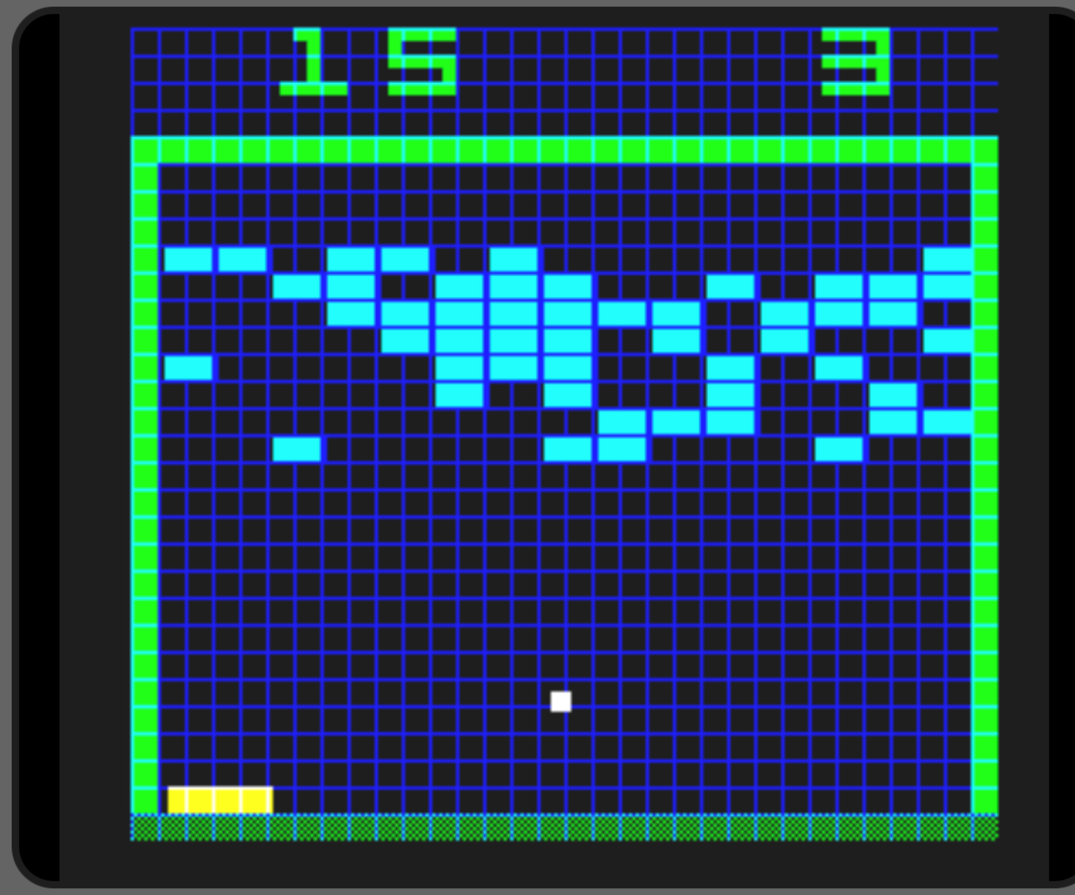


Verilog | Brick Smash Game | 60.00 fps | 8bitworkshop | Books

ball_paddle.v
hvsync_generator.v
digits10.v
scoreboard.v

Debug Tree
Asset Editor

```
1
2   `include "hvsync_generator.v"
3   `include "digits10.v"
4   `include "scoreboard.v"
5
6   /*
7   A brick-smashing ball-and-paddle game.
8   */
9
10  module ball_paddle_top(clk, reset, hpaddle, hsync, vsync, rgb
11
12    input clk;
13    input reset;
14    input hpaddle;
15    output hsync, vsync;
16    output [2:0] rgb;
17    wire display_on;
18    wire [8:0] hpos;
19    wire [8:0] vpos;
20
21    reg [8:0] paddle_pos; // paddle X position
22
23    reg [8:0] ball_x;     // ball X position
24    reg [8:0] ball_y;     // ball Y position
25    reg ball_dir_x;       // ball X direction (0=left, 1=right)
26    reg ball_speed_x;     // ball speed (0=1 pixel/frame, 1=2 p
27    reg ball_dir_y;       // ball Y direction (0=up, 1=down)
28
29    reg brick_array [0:BRICKS_H*BRICKS_V-1]; // 16*8 = 128 bits
30
31    wire [3:0] score0;    // score right digit
32    wire [3:0] score1;    // score left digit
33    wire [3:0] lives;     // # lives remaining
34    reg incscore;         // incscore signal
35    reg declives = 0; // TODO
36
37    localparam BRICKS_H = 16;     // # of bricks across
38    localparam BRICKS_V = 8;      // # of bricks down
39
40    localparam BALL_DIR_LEFT = 0;
41    localparam BALL_DIR_RIGHT = 1;
42    localparam BALL_DIR_DOWN = 1;
```



brick_index

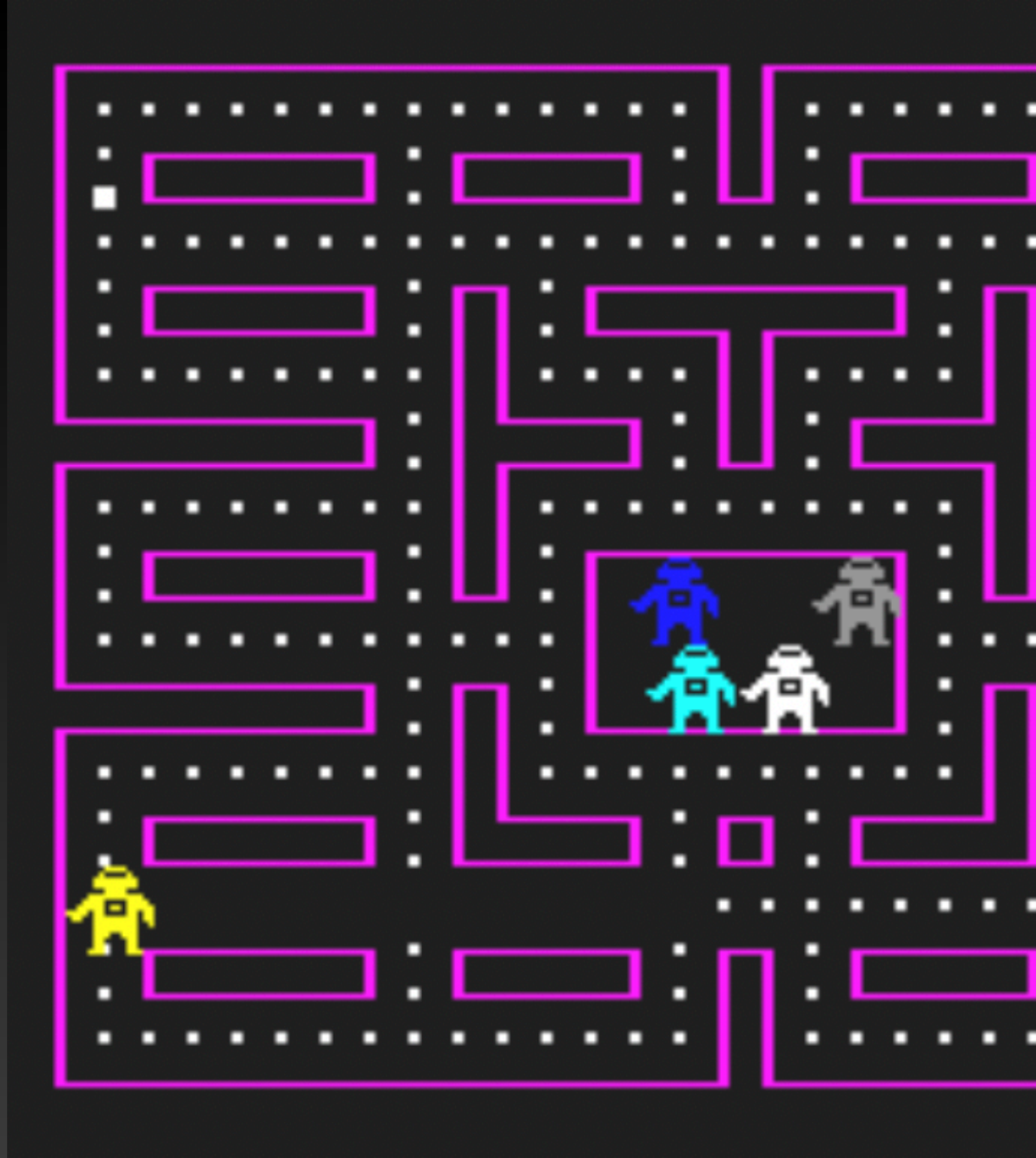
ball_pixel_collide

ball_collide_paddle

ball_collide_bits

8bitworkshop Example Modules

- Video sync generator
- Bitmap character display
- Switch and paddle input
- Sprites
- Sound
- ALU, CPU
- Multiple modules integrated into 16-bit computer design



Assembly Support

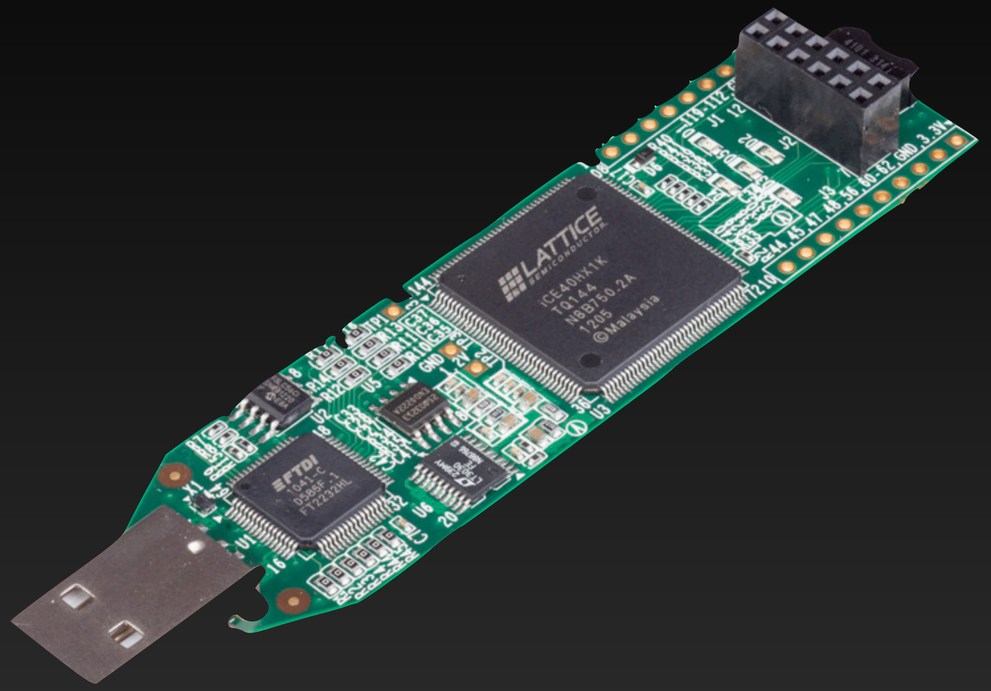
- IDE can compile custom assembly languages defined by JSON
- Used for custom CPU designs

```
{
  "name":"femto16",
  "width":16,
  "vars":{
    "reg":{"bits":3, "toks":["ax", "bx", "cx"
    "unop":{"bits":3, "toks":["zero","loada",
    "binop":{"bits":3, "toks":["or","and","xo
    "imm5":{"bits":5},
    "imm8":{"bits":8},
    "imm16":{"bits":16},
    "rel8":{"bits":8, "iprel":true, "ipofs":1
  },
  "rules":[
    {"fmt":"~binop ~reg,~reg",    "bits":["000
    {"fmt":"~unop ~reg",          "bits":["000
    {"fmt":"~binop ~reg,[~reg]", "bits":["000
    {"fmt":"mov [~reg],~reg",     "bits":["010

    {"fmt":"mov ~reg,[~imm8]",    "bits":["001
    {"fmt":"mov [~imm8],~reg",    "bits":["001
    {"fmt":"~binop ~reg,#~imm8", "bits":["11"
    {"fmt":"~binop ~reg,@~imm16","bits":["000
    {"fmt":"~binop ~reg,[~imm16]", "bits":["0
```

FPGAs

- Example code modified to run on Lattice FPGAs
- Icestorm toolchain and Yosys
- Drives a composite CRT, need a few resistors to get the levels needed



Links

- https://8bitworkshop.com/verilog
- https://github.com/sehugg/8bitworkshop
- https://github.com/sehugg/fpga-examples
- “Designing Video Game Hardware in Verilog”

Thanks!

Steven Hugg
@sehugg

info@8bitworkshop.com