

Arcade Poker Deck

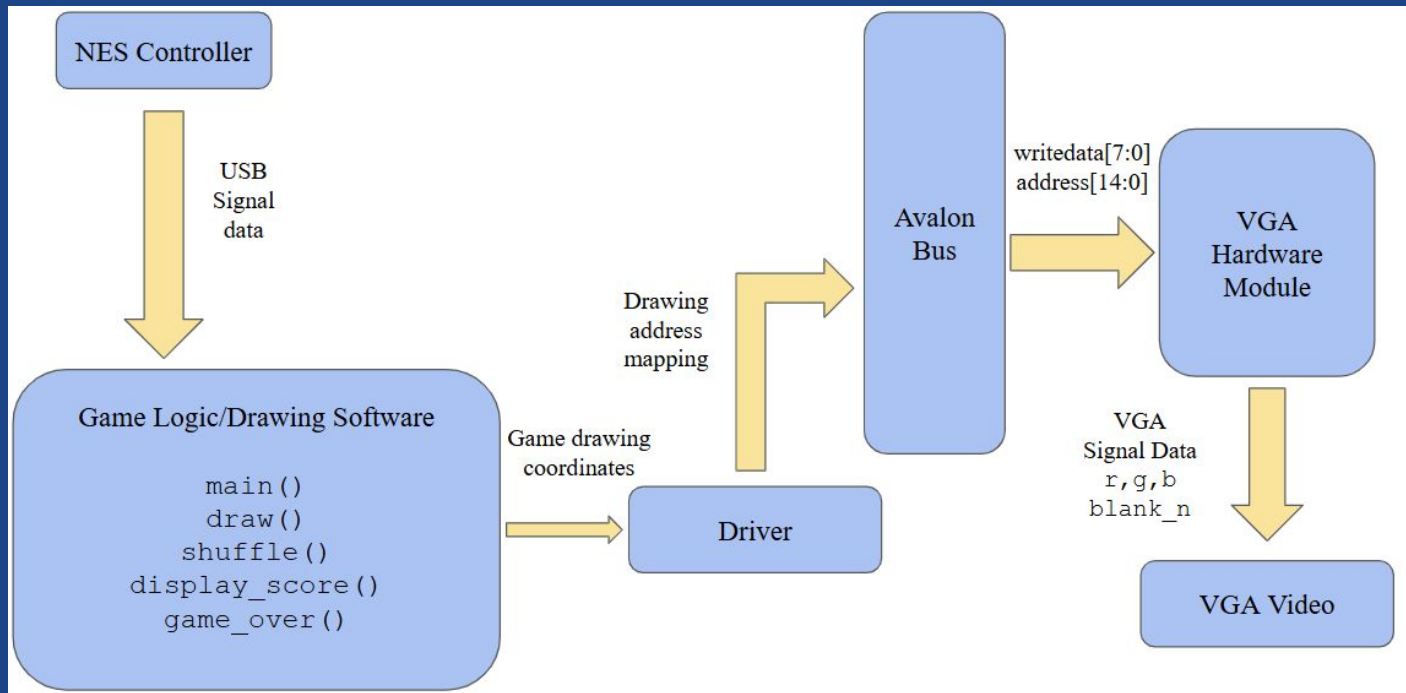
Builder Game: Balatro Minus

Mahdi Ali-Raihan, Timothy Melendez,
Mario Carrillo, Julio Ramirez

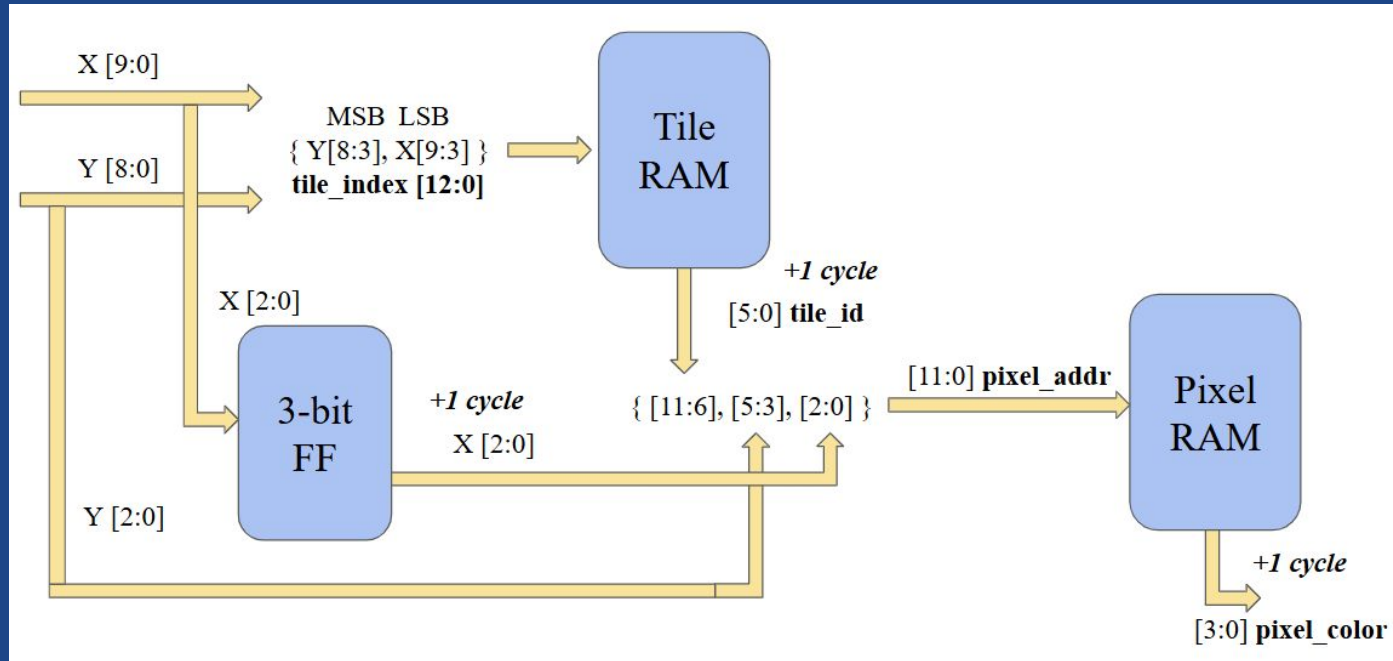
Game Overview

- Build poker hands using cards in your hand
- Reach a threshold score to beat the round
- Three successive rounds conclude an Ante, up to eight Antes
- You win if you beat the last Ante
- You lose if you fail to reach the threshold score in a round with the number of hands provided

System Overview

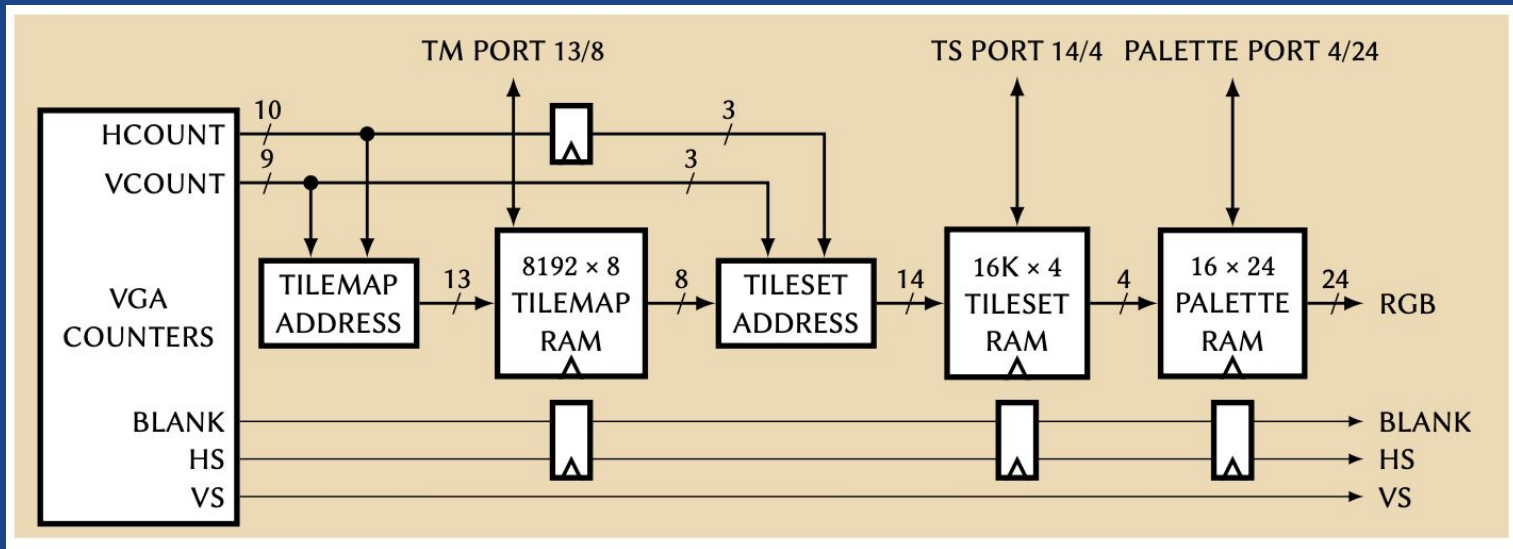


VGA Control: Original Design

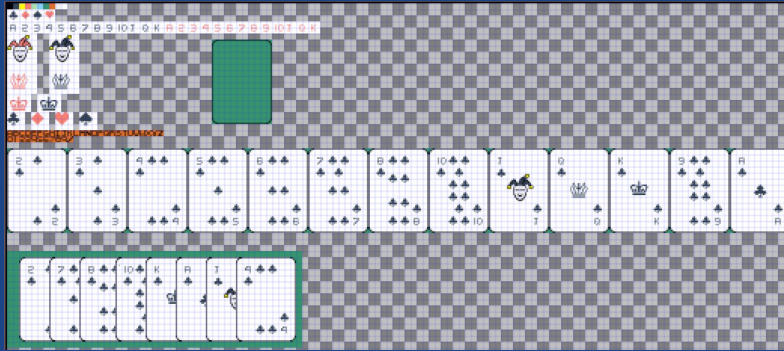


VGA Control: Actuality

Courtesy of Stephen A. Edwards



Tile Graphics



- Used 255 distinct tiles in our Poker game, which are all created in Aseprite

Convert Tileset png into hex file



↓ Tileset in indexed png form (tiles in one row)

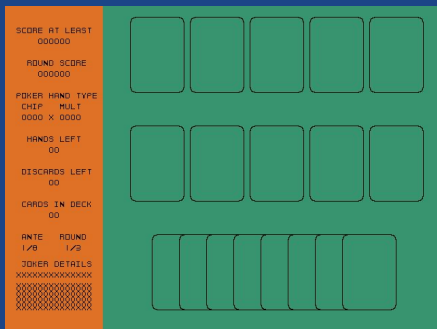
```
1 00000000
2 000000FF
3 36454FFF
4 FFFF00FF
5 FA8072FF
6 AFE1AFFF
7 89CFF0FF
8 F0F0F0FF
9 FFFFFFFF
10 37946EFF
11 DF7126FF
12 00000000
13 00000000
14 00000000
15 00000000
16 00000000
17
```

Palette.hex
(12 colors +
padding)

```
1 00 00 00 00 00 00 00 00
2 00 00 00 00 00 00 00 00
3 00 00 00 00 00 00 00 00
4 00 00 00 00 00 00 00 00
5 00 00 00 00 00 00 00 00
6 00 00 00 00 00 00 00 00
7 00 00 00 00 00 00 00 00
8 00 00 00 00 00 00 00 00
9 00 00 00 00 00 00 00 00
10 00 00 00 00 00 00 00 00
11 00 00 00 00 00 00 00 00
12 00 00 00 00 00 00 00 00
13 00 00 00 00 00 00 00 00
14 00 00 00 00 00 01 01
15 00 00 00 00 01 01 02 02
16 00 00 00 01 01 02 02 02
17 00 00 00 00 00 00 00 00
18 00 00 00 00 00 00 00 00
19 00 01 03 03 03 01 00 00
20 01 03 03 03 03 01 00 00
21 01 03 03 03 03 01 00 00
22 01 03 03 03 03 01 00 00
23 02 01 01 01 01 00 00 00
24 01 00 00 00 00 00 00 00
25 00 00 00 00 00 00 00 00
26 00 00 00 00 00 00 00 00
27 00 00 00 00 00 00 01 01
28 00 00 00 00 00 00 01 01
29 00 00 00 00 00 01 01 02
30 00 00 00 00 01 01 02 02
31 00 00 00 00 01 02 02 02
32 00 00 00 01 02 02 02 01
33 00 00 00 00 00 00 00 00
34 00 00 00 00 00 00 00 00
```

tileset.hex

Converting from .png to .hex



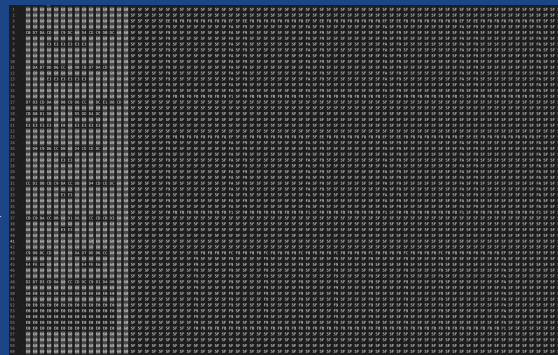
Game UI in indexed
PNG form

1	00000000
2	000000FF
3	36454FFF
4	FFFF00FF
5	FA8072FF
6	AFE1AFFF
7	89CFF0FF
8	F0F0F0FF
9	FFFFFFF
10	37946EFF
11	DF726FFF
12	00000000
13	00000000
14	00000000
15	00000000
16	00000000
17	

palette.hex

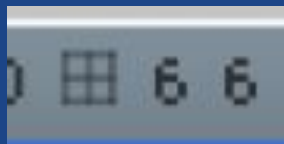
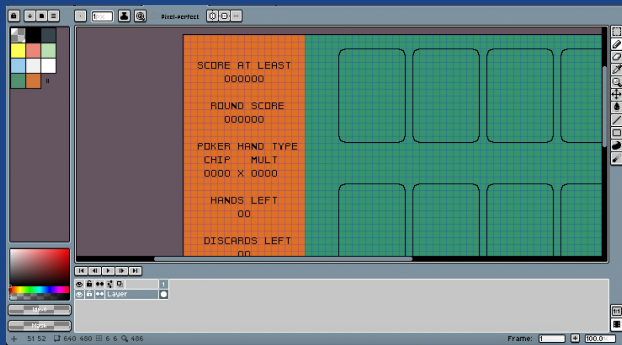
1	00 00 00 00 00 00 00 00
2	00 00 00 00 00 00 00 00
3	00 00 00 00 00 00 00 00
4	00 00 00 00 00 00 00 00
5	00 00 00 00 00 00 00 00
6	00 00 00 00 00 00 00 00
7	00 00 00 00 00 00 00 00
8	00 00 00 00 00 00 00 00
9	00 00 00 00 00 00 00 00
10	00 00 00 00 00 00 00 00
11	00 00 00 00 00 00 00 00
12	00 00 00 00 00 00 00 00
13	00 00 00 00 00 00 00 00
14	00 00 00 00 00 00 00 00
15	00 00 00 00 00 00 00 00
16	00 00 00 00 00 00 00 00
17	00 00 00 00 00 00 00 00
18	00 00 00 00 00 00 00 00
19	00 00 00 00 00 00 00 00
20	00 00 00 00 00 00 00 00
21	00 00 00 00 00 00 00 00
22	00 00 00 00 00 00 00 00
23	00 00 00 00 00 00 00 00
24	00 00 00 00 00 00 00 00
25	00 00 00 00 00 00 00 00
26	00 00 00 00 00 00 00 00
27	00 00 00 00 00 00 00 00
28	00 00 00 00 00 00 00 00
29	00 00 00 00 00 00 00 00
30	00 00 00 00 00 00 00 00
31	00 00 00 00 00 00 00 00
32	00 00 00 00 00 00 00 00
33	00 00 00 00 00 00 00 00

tileset.hex



Game UI indexed png is now
in .hex format

Retrieving Tile Row/Col to Modify from SW → HW



VGA HW-SW Interface

```
/* Device registers */
```

```
#define TILEMAP_BASE 0x0000 /* 8 KiB: 0x0000-0x1FFF */
```

```
#define PALETTE_BASE 0x2000 /* 64 B: 0x2000-0x203F */
```

```
#define TILESET_BASE 0x4000 /* 16 KiB: 0x4000-0x7FFF */
```

```
static void set_pixels(vga_poker_pixels_t *pixel_coors)
{
    int tile_id;
    int i;
    tile_id = pixel_coors->tile_id;
    for (i = 0; i < 64; i++) {
        int color;
        int new_addr;
        color = pixel_coors->pixels[i];
        new_addr = TILESET_BASE + (tile_id << 6) + i;
        pr_info("addr: %u\n", new_addr);
        iowrite8(color, dev.virtbase + new_addr);
    }
}
```

```
static void set_tile(vga_poker_tile_coords_t *tile_coords)
{
    int row;
    int col;
    int tile_id;
    unsigned short new_tile_addr;
    row = tile_coords->row;
    col = tile_coords->col;
    tile_id = tile_coords->tile_id;
    new_tile_addr = TILEMAP_BASE + (row << 7 | col);
    pr_info("addr: %u\n", new_tile_addr);
    iowrite8(tile_id, dev.virtbase + new_tile_addr);
}
```

```
static void set_palette(vga_poker_palette_t *palette)
{
    int base;

    if (palette->index >= 16) return;

    base = PALETTE_BASE + (palette->index << 2);
    pr_info("addr: %u\n", base);
    /* write into creg[7:0], [15:8], [23:16] */
    iowrite8(palette->red, dev.virtbase + base + 0);
    iowrite8(palette->green, dev.virtbase + base + 1);
    iowrite8(palette->blue, dev.virtbase + base + 2);
    /* commit into palette[pl->index] */
    iowrite8(0, dev.virtbase + base + 3);
}
```

NES Controller



```
/* Controller Input Functions */  
void initialize_debounce();  
int is_debounce_elapsed();  
int  
check_controller_input(unsigned  
char *report);  
void toggle_card_selection();  
void move_cursor(int direction);  
void  
process_controller_input(unsigned  
char *report);
```

Left: moves cursor left

Right: moves cursor right

A: select card

B: deselect card

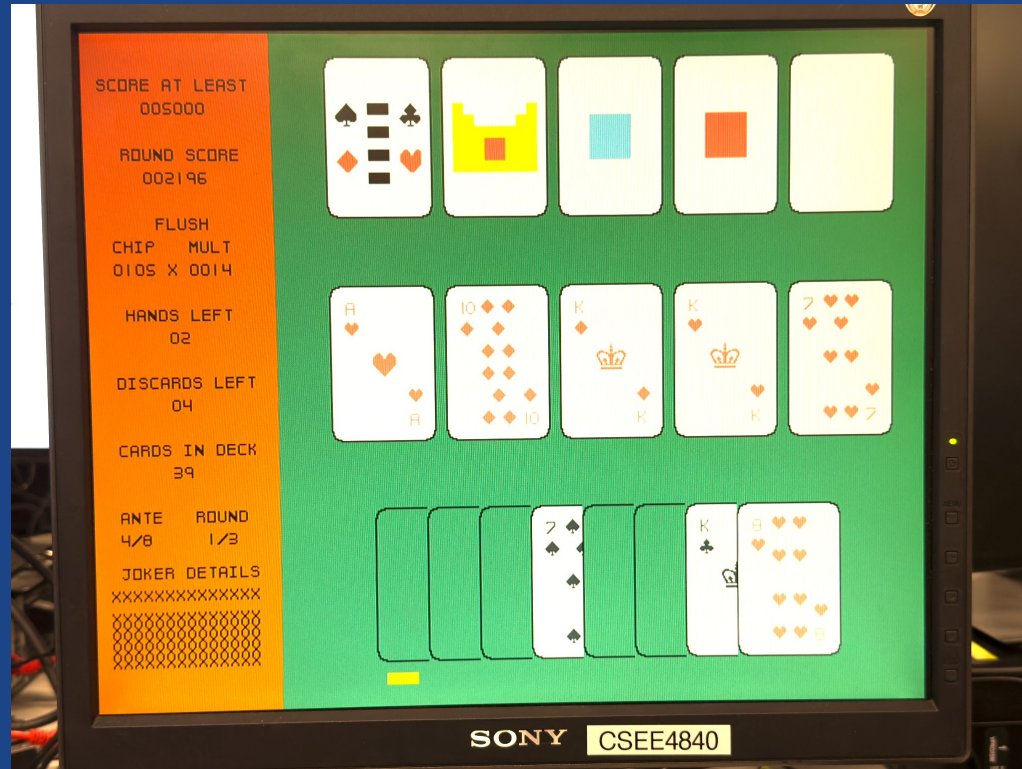
Select: discard selected cards

Start: play selected cards

Game Logic

- Init Deck, Shuffle, Hand Generation
- Select/Discard Cards
- Chip and Mult Values Based on Poker Hand Type (Pair, Two Pair, Flush, Straight, etc...)
- Current Score, Target Score, Discards, Hands,
- Game States (Win, Lose, Next Level, etc...)
- Joker Probability Drops

Image of Game



Demonstration