Winner Winner, Chicken Dinner!

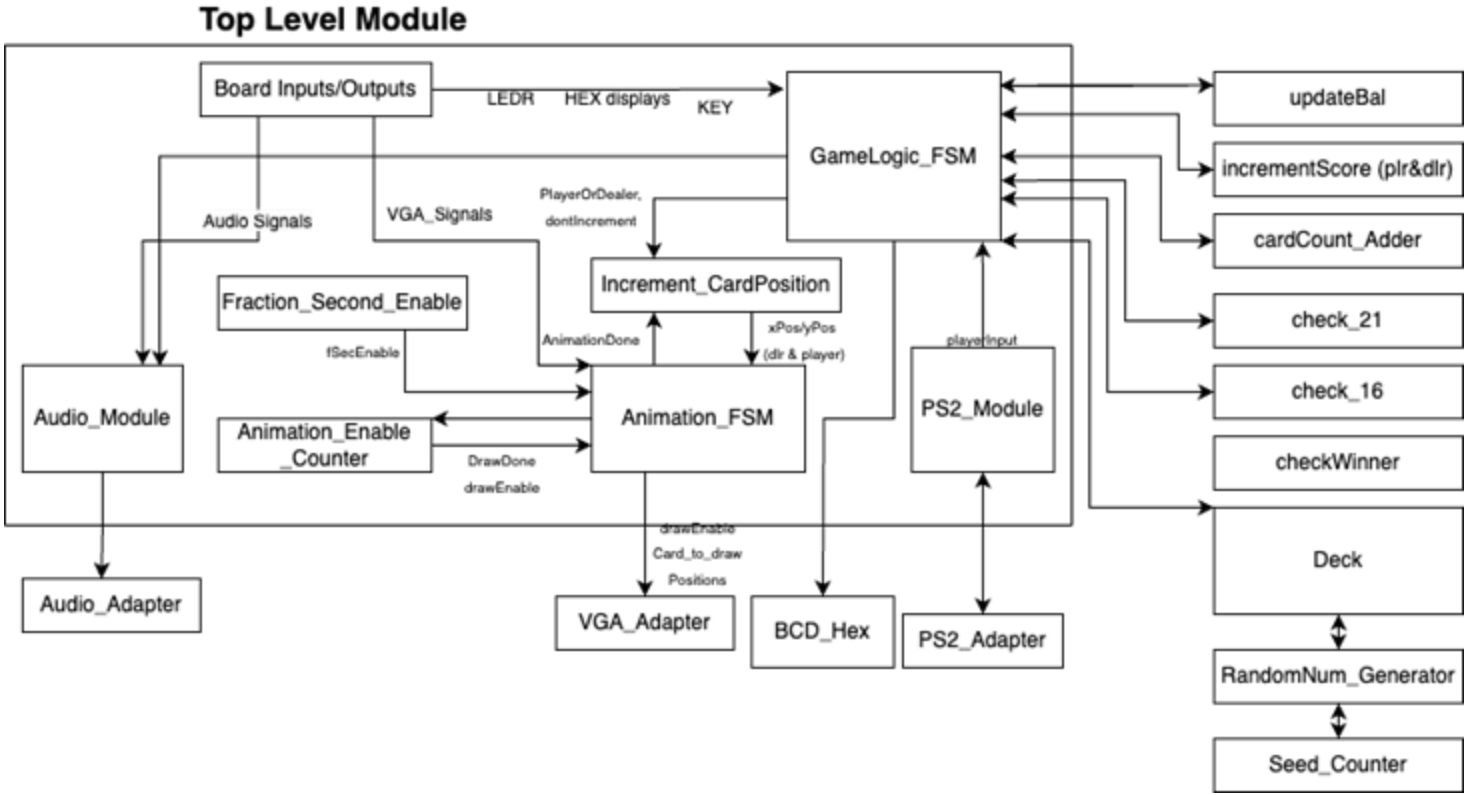
ECE241 Final Presentation

Arnav Patil, Jack Sloan

Project Description

Functions	Scope
● Simulate randomly drawing cards from 52-card deck. ● Handle a complete game of “regular” blackjack (win conditions and card scoring). ● Continuously display player and dealer score, as well as cards in both parties’ decks.	● Original scope: ○ Scoped out split and double down. ○ Scoped out ability to set bet amount. ● Scope creep: ○ No implementation of mouse in PS/2. ○ No voice/text prompts for player to make a move (stand or hit).

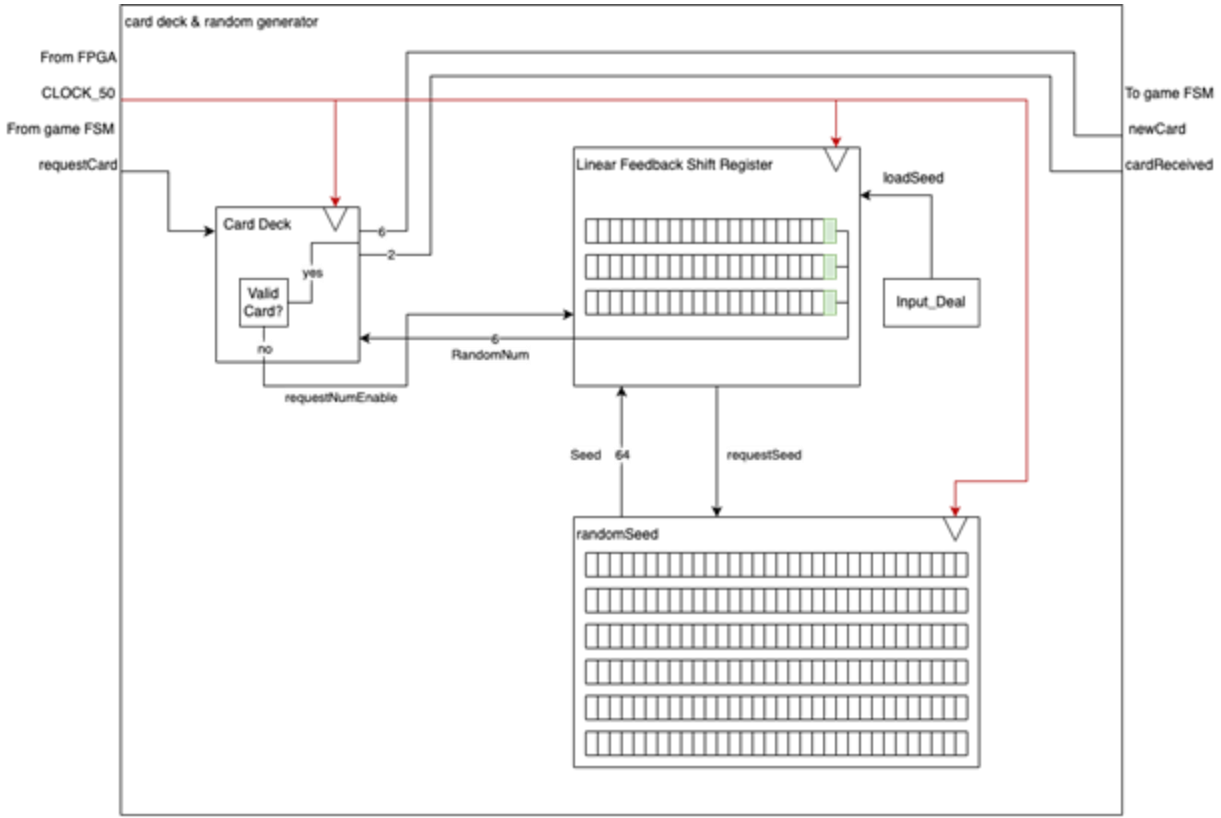
Block Diagrams – Top-Level Design Module



Random Number Generator & Deck Module

- `RandomNum` generator:
  - Uses 6 x 32 bit Linear Feedback Shift Registers
  - Seed changes on every `CLOCK_50`
- Deck Module:
  - When card request signal goes high, repeatedly probe `randomNum` generator until it generates a random number that 1. Is valid (52 of 64 values), and 2. Hasn't been chosen yet (stores number of cards pulled and)

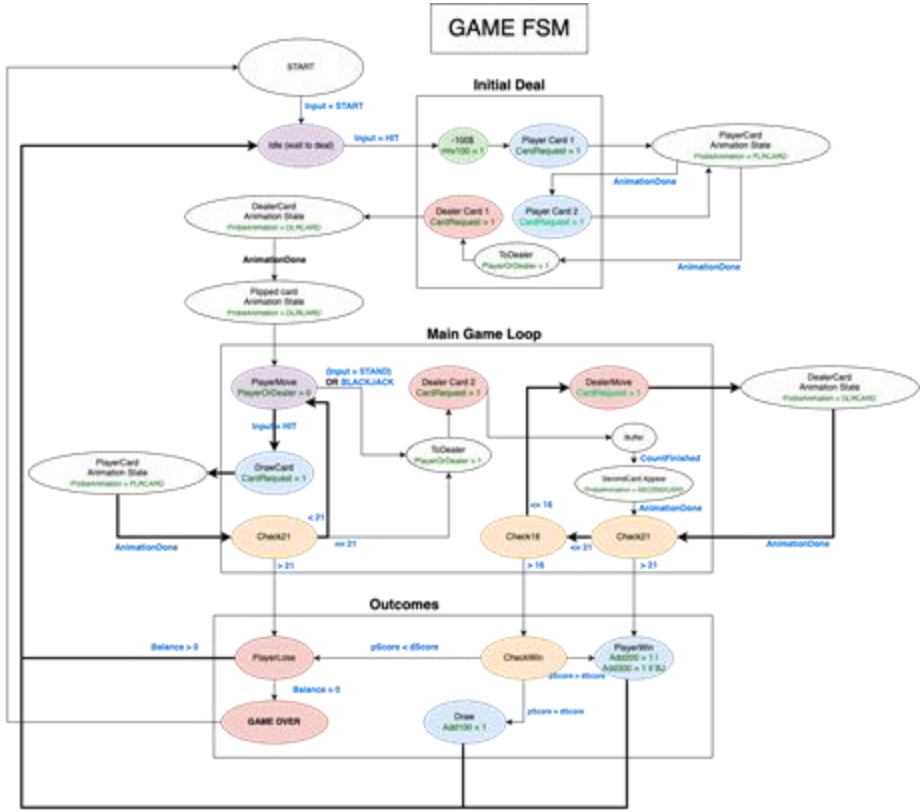
Random Number Generator & Deck Module



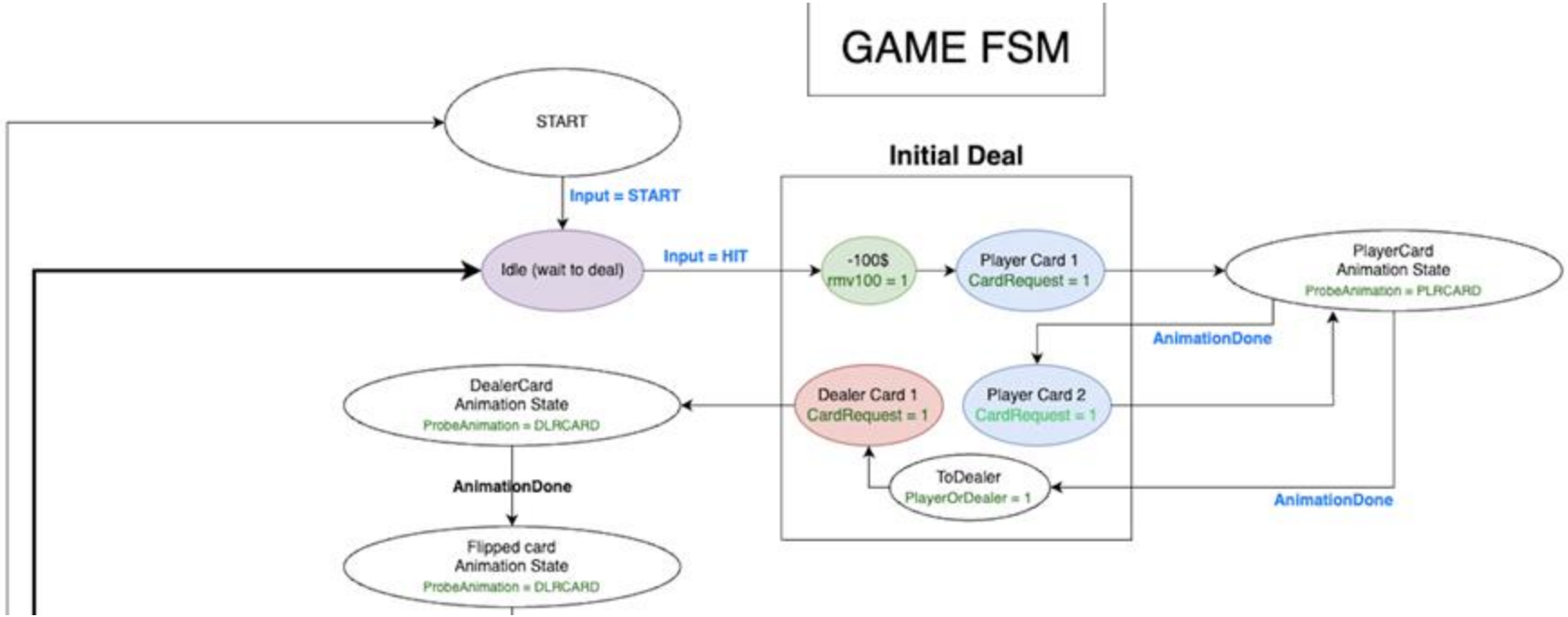
Block Diagram – Game Logic

Notes

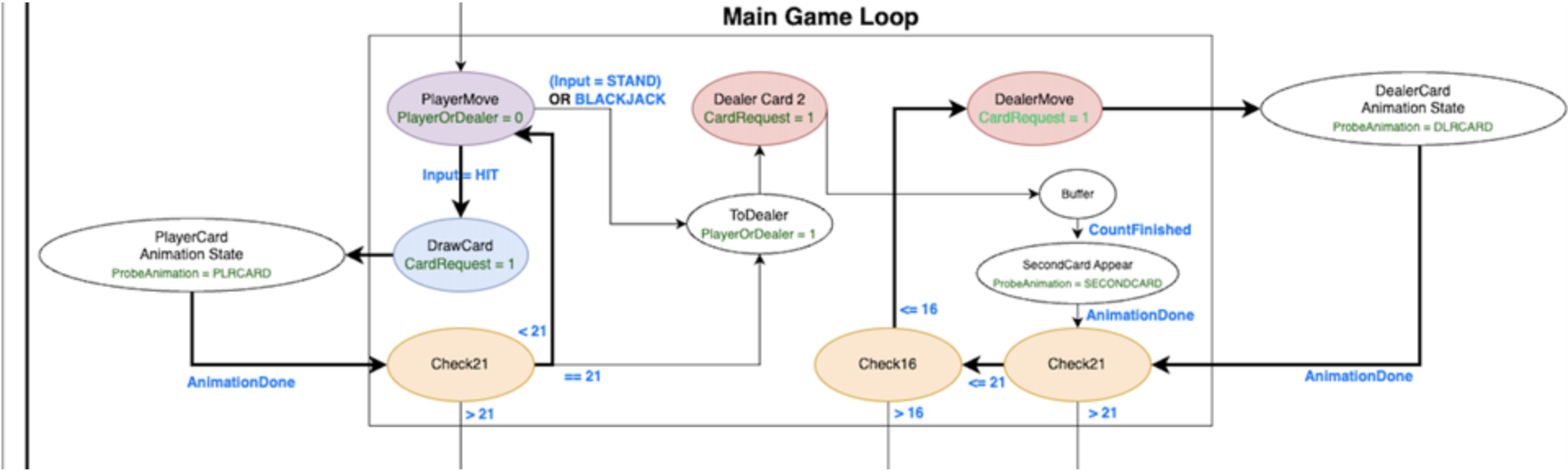
- All state changes are synched up to `CLOCK_50`
- Animation states send `probeAnimation` signal, and wait for Animation FSM to return “Animation finished” signal
- Not shown here but FSM instantiates necessary modules to update balance, card count, p&d scores, and random card



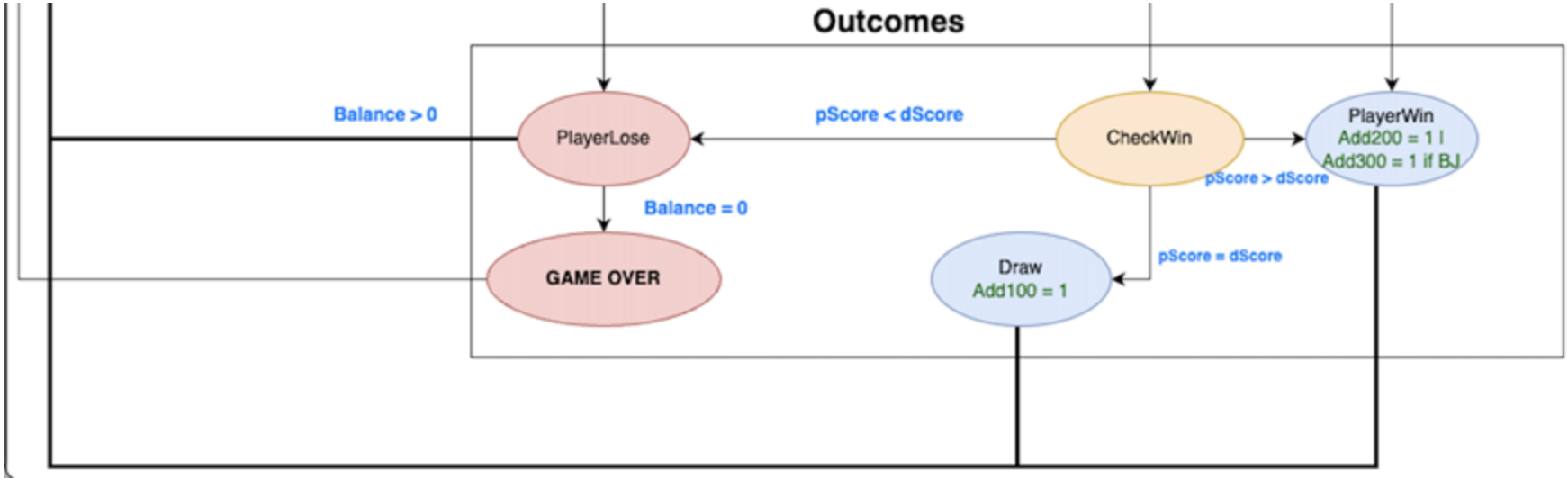
Initial Deal – FSM Part I



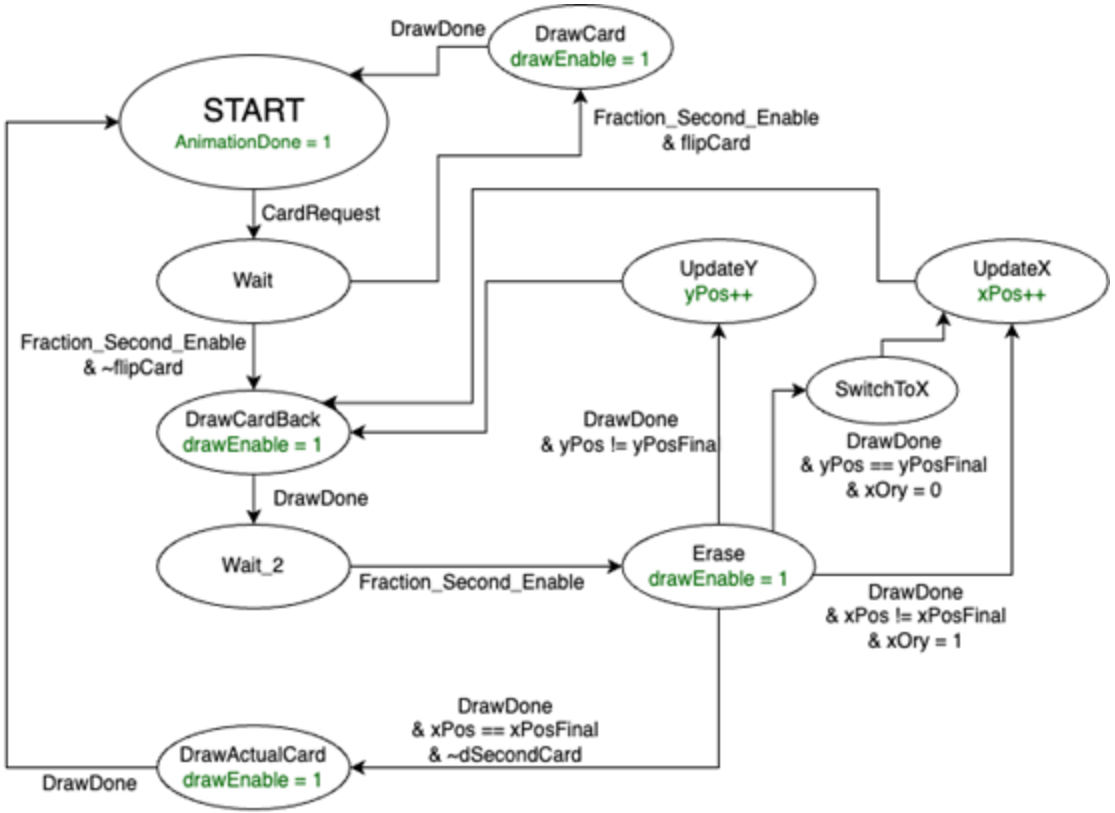
Main Game Loop – FSM Part II



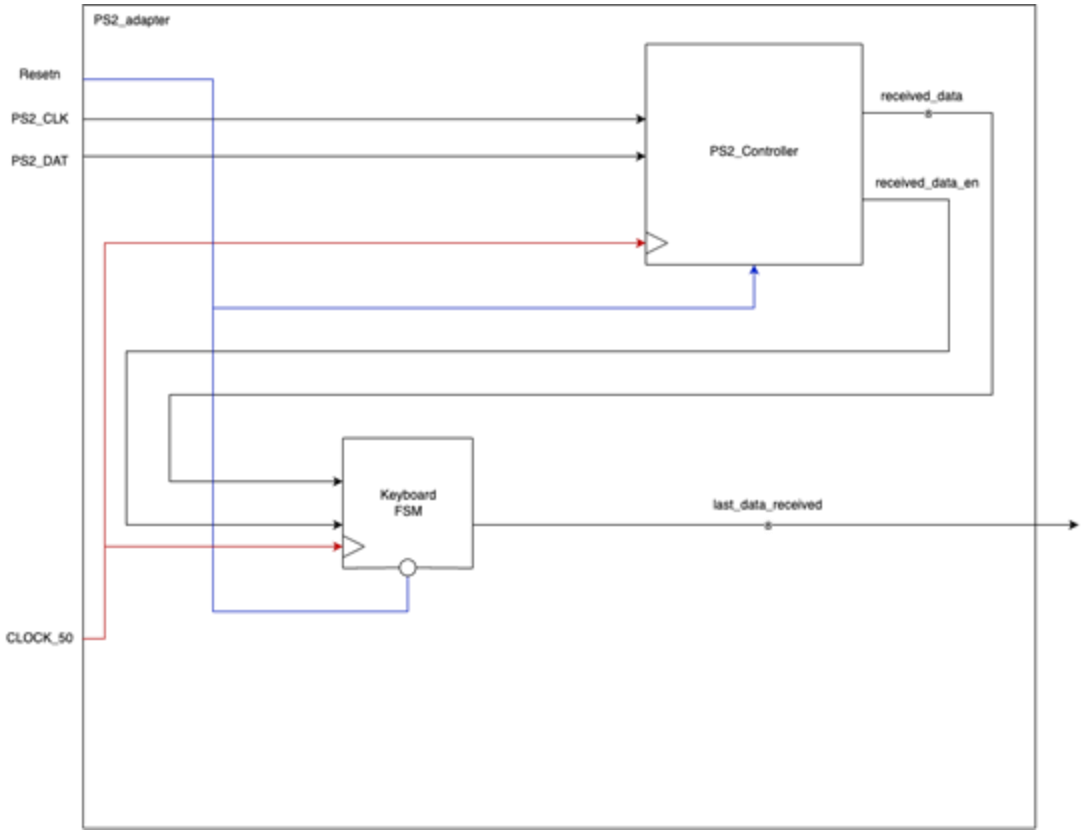
Outcome Condition – FSM Part III



Animations FSM



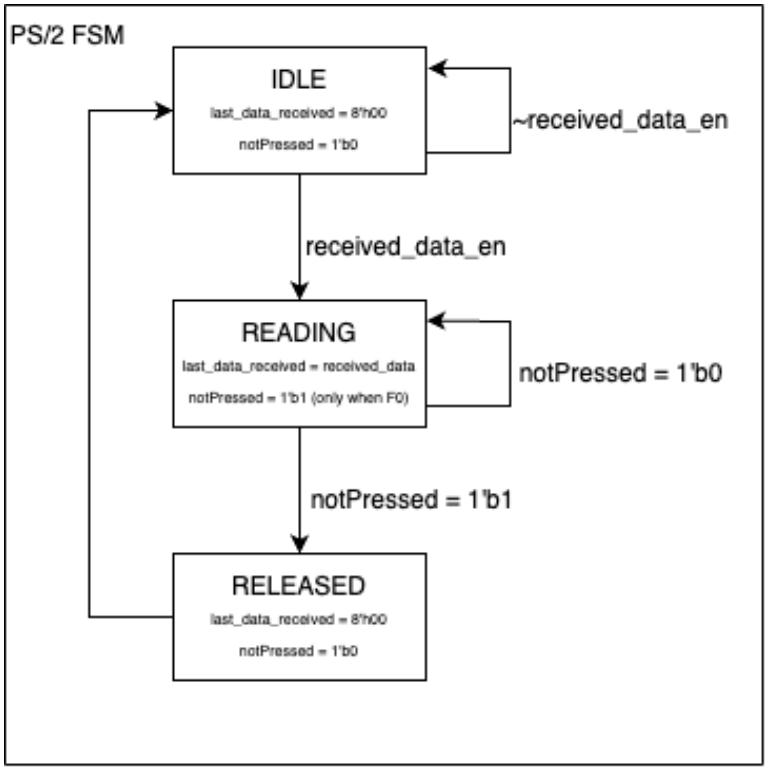
Block Diagrams – PS/2 Keyboard



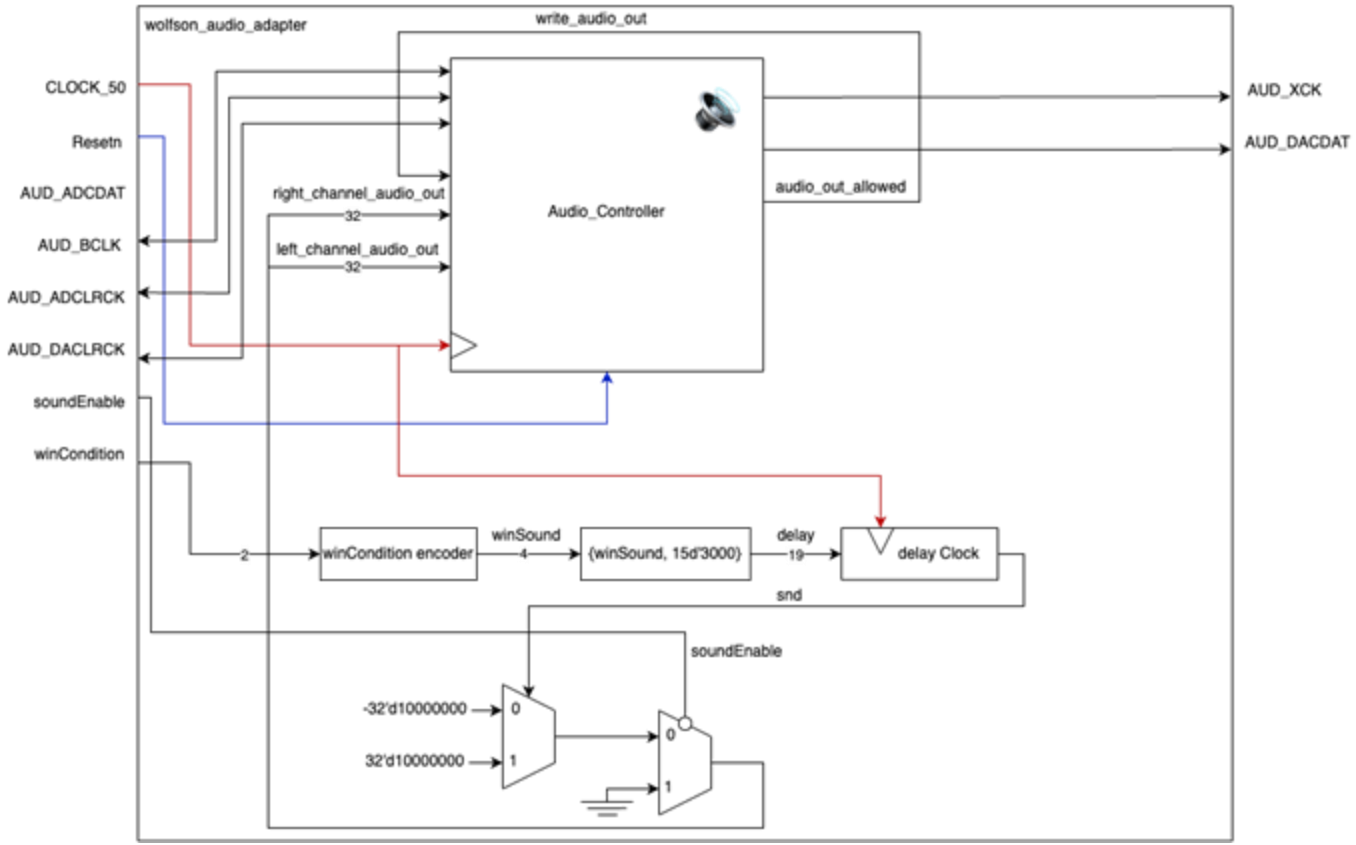
Block Diagrams – PS/2 Keyboard (FSM)

FSM Description

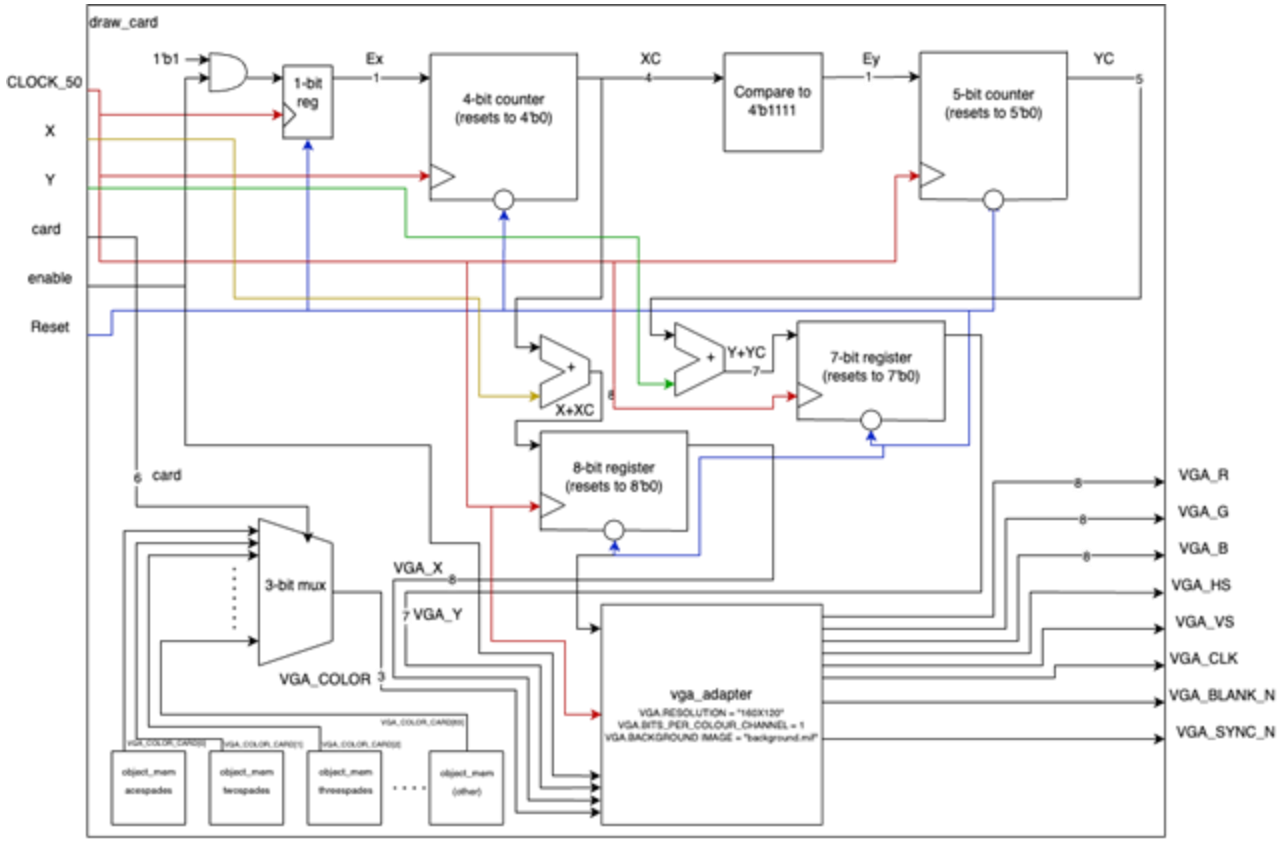
- Signal out of FSM (read by game loop) is `8'h00` by default.
- `received_data_en` asserted high when a **new** and **valid** data packet is read.
- At break code (F0), an internal flag goes high to transition to next state.
- State lowers flag, sets output to low, and automatically transitions to next state.



Block Diagrams – Wolfson Audio Codec



Block Diagrams – VGA Display



Obstacles

Generating sufficiently random numbers (avoiding the same seed each game)

- `RandomNum` generator:
  - Started by having initial seeds each staggered by 1
    - Randomized initial seeds (stagger values)
- Erasing:
  - Whenever I would change the animationFSM, it would mess up the dealer's second card animation
    - Realized that I forgot to
- Soft Ace
  - Store soft aces

Obstacles

No workaround to instantiating 52 (+) memory modules

- Design choice was to draw cards to the screen using memory module instead of pixel values (as done in Prof. Brown's "Object" exemplar).
- Name of the background file is a parameter to the altsyncram module instantiated in `object_mem`.
  - Cannot be defined from our VGA interface.
  - Cannot have a modifiable `init_file`.
- No choice but to instantiate 52+ times with a static parameter declaration.

Obstacles

Memory modules only accepted MIFs with serial input

- Instructor-provided bmp2mif converter produces vectorized MIF pixels, perfectly compatible with `vga_adapter` to draw VGA background.
- Vectorized MIF input would not work with object_mem, even after debugging.
  - Conclusion was that `object_mem` only accepts serialized input.
  - * Some other groups did not have this issue, but none of our implemented solutions worked.
- Developed a Python script to parse the file (had to learn to use `re` library)
  - Shared to GitHub to so others could use it too. [GitHub Repository]

Further Development

Game Logic/Functions

- Splitting – if a player draws two of the same card, they have the option to “split,” or play two games at once.
- Doubling down – if a player’s initial hand totals 9, 10, or 11, they may choose to place a new bet on one card which is not revealed till the bet is closed.

Revamping Audio

- Further developing audio integration into the project to allow for ambient music during gameplay and “jingle” noises on win screen, etc.

Further Development

Peripherals

- Scoped out mouse functionality from PS/2; future work could include buttons drawn to background and mouse click within pixel box parsed as button click.

Further Considerations

- Original idea stemmed from learning to “count” cards in a blackjack game.
- Much advanced project is using the FPGA as an acceleration testbench to implement an algorithm for counting cards.

Demo Video

Demo Video

- https://drive.google.com/file/d/1qDcLYdV6pRbxt4B5Hn6ZK8IF2aaEKnVO/view?usp=sharing
- Please note: this video was taken before some functionality was added to the project. After this video, we were able to implement a screen-erase function and cleaned up some aspects of the game logic.