6.111 Final Project Proposal
Amy Daitch
Elizabeth Murnane

Digital Interactive Poker

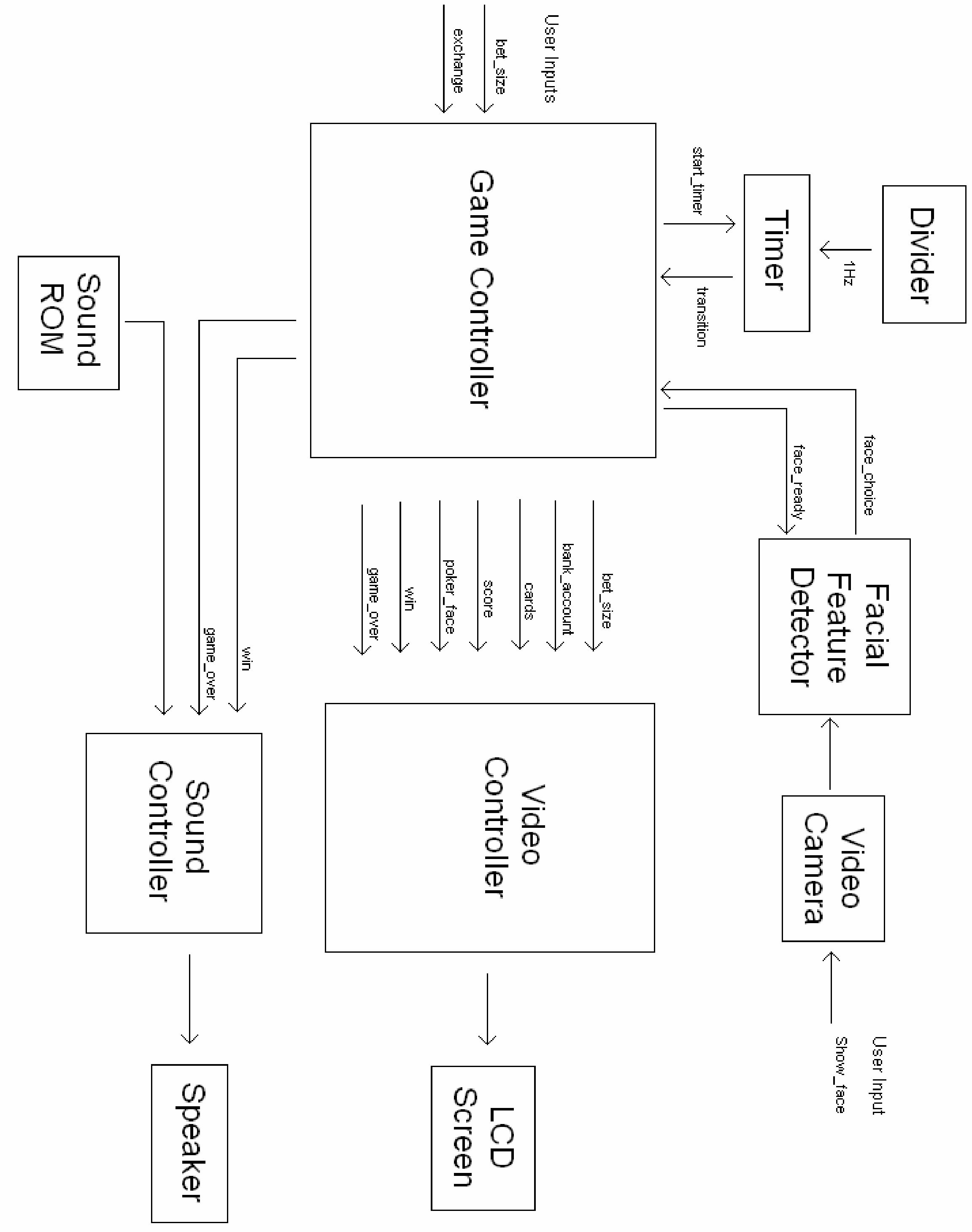
Overview:

We plan to implement a game of five card draw poker for a single player (against the computer- the dealer). To summarize the rules of the game, at the beginning of each round, each player is dealt five cards. A hand of five cards is ranked based on the standard poker point system (exact point system unimportant for now). After five cards are dealt to each player, the non-dealer makes his/her bet after which the dealer must either match the bet or fold (drop out of the round). Players can continue to increase bets, and a round of betting is over either when each player has bet the same amount and neither wishes to raise, or if a player decides to fold. Then, each player can trade up to three of his/her current cards. After the new cards are dealt, one more round of betting takes place, after which the players must reveal their cards. The player with the highest ranked hand wins the round and keeps all the chips in the pot. The game continues until one of the players is out of chips.

Since much of the fun of poker is in the psychology of the game (e.g. psyching out one's opponent, maintaining a "poker face", etc.) we decided to incorporate these aspects into the project, giving the computer some "human" qualities. The computer will be able to "see" the expression on the player's face (smile vs. straight face vs. frown) by analyzing the input from a video camera which is monitoring the player's facial expression (more likely, the player will simply chose one of a set of pre-made masks with easily detectible facial features) and may use this information to strategize its next move. Furthermore, the computer will generate a facial expression after each set of cards dealt (perhaps as an LED smiley face display or the like) which is somehow related to the computer's current hand. The player may then use this information to strategize his/her next move. To make the game even more interesting, the player may choose which of the computer's "personas" to play against. Each persona will have a different strategy which determines which cards it will trade, how much it will bet for each round, and what facial expression it displays, all as a function of its current hand of cards and its opponents facial expression. For example one persona might simply always bet high, never trade any cards, disregard the facial expression of its opponent, display a smiley face when it has a bad hand of cards, and a frown when it has a good hand of cards.

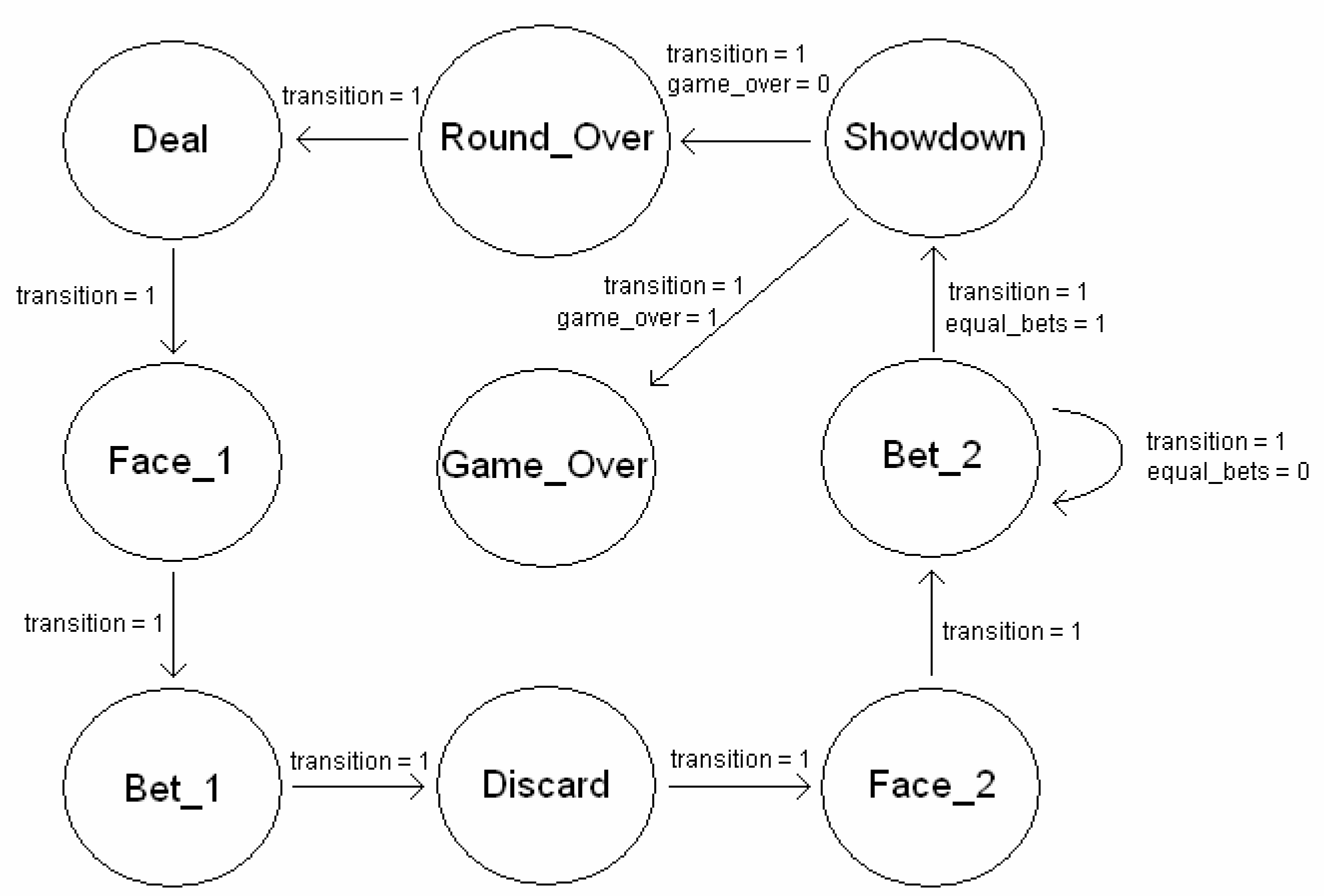
To simplify the design of this game, we will divide it into several subsystems which can be designed and tested individually before being integrated into the final product. We've divided this project into a Game Controller, Video Controller, Sound Controller, and Facial Feature Detector. These subsystems are described in more detail below:

Block Diagram



Game Controller:

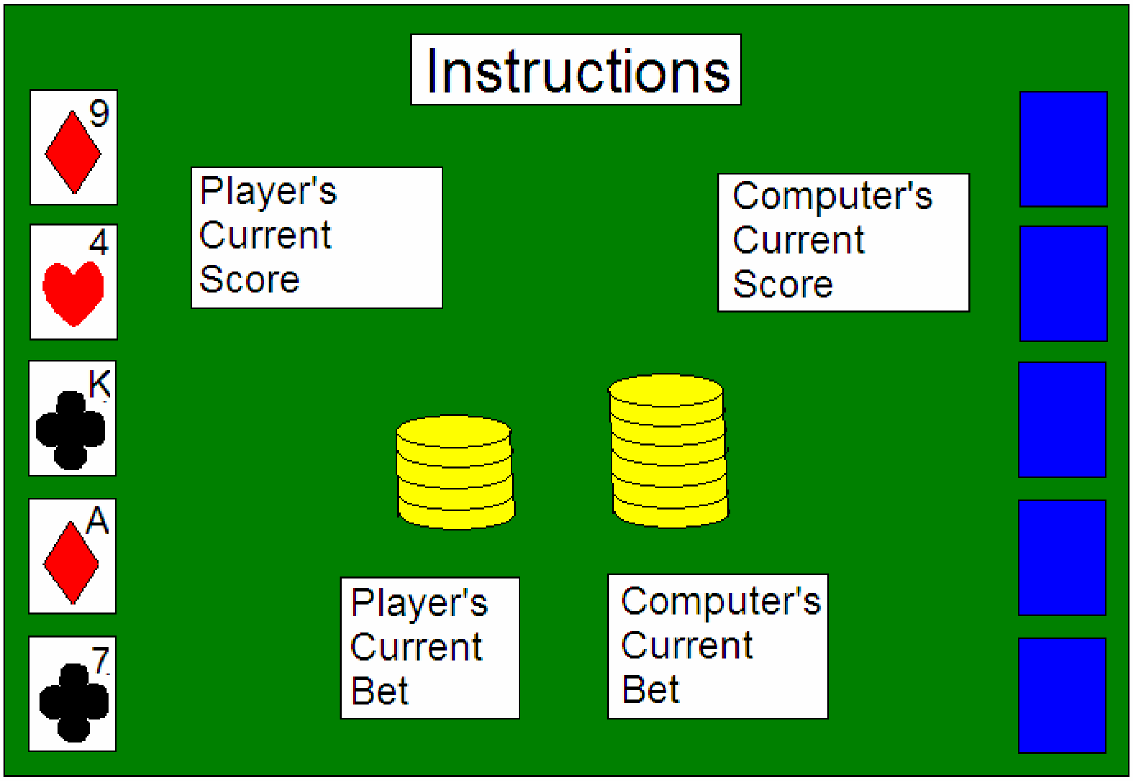
The poker game itself is controlled by a finite state machine. (State transition diagram on next page). We remain in a state until the user performs the action necessary to move to the next state (for example, we move out of the first betting state after the user has placed his bet). The first state is Deal (dealing the cards). Each hand is made up of 5 cards, and each card can be one of the 52 cards in a deck. We represent a card as a 6 bit binary number. The first 2 bits are the suit. 00 = Hearts, 01 = Diamonds, 10 = Clubs, and 11 = Spades. The next 4 bits are the value of the card which can range from 1 to 14. (Jack is given a value of 11, Queen 12, King 13, and Ace 14). So, five random 6 bit numbers are generated and assigned to be the hand of a player. After the cards have been randomly dealt and each player has a full hand, the players can put on their poker faces. So, the next state is the Face state, where the Game Controller outputs a face_ready signal to the Facial Feature Detector to signal that it is time for players to put on their poker faces. When playing, both computer and human players can look at opponents' faces in order to help them make betting or folding decisions. Both human and computer players choose a face. Human players get a face by choosing from a selection such as happy, sad, and angry. Their selection is detected with the Facial Feature Detector. Each computer player possesses a different personality, which determines the types of faces he makes. For example, the computer personality "Nasty" would always select an angry face, "Spaz" would randomly choose a face, and "Eye-for-Eye" would mimic the other player's face. After this, the state transitions to Bet_1, the first betting round. Here, players select the amount of money they want to bet. More is bet on a better hand, unless the player is bluffing. We keep track of how much money each player has in order to ensure that no player bids more money than he has. At the end of the first betting round, we transition to state Discard, where each player may discard and draw up to five cards. The computer determines which cards to discard based on a computer-move algorithm. This algorithm analyzes the hand and determines the best course of action. After discarding, the cards missing from a player's hand are randomly replaced, and the state transitions to a second Face state. Players can keep or change their poker faces before the state changes to Bet_2, the second betting round. After the final bets are placed, we transition to the Showdown state. Here, an algorithm determines the winner of the round. If the user loses and is out of money, the game_over signal goes high and we transition to state Game_Over. If the user still has money left, we go to the state Round_Over, where all the cards are removed from the table, faces are cleared, the pot is deposited in the winner's bank account, and the players' new scores are calculated. Finally, we transition back to the first state, Deal, and begin a new game. We can test the FSM one state at a time with LEDs and the FPGA display. This Game Controller needs to take in the following inputs: bet amount, poker face, and cards for exchange. We would like to allow the user to enter some of these inputs with a mouse. The Game Controller also has multiple outputs which are sent to the Video Controller and Sound Controller. The video controller receives and displays the players' hands, poker faces and bank account sizes. It is also fed bet sizes, player scores, and win/loss results. All of these things are displayed on an LCD screen. The Sound Controller also receives win/loss results. It accesses a Sound ROM and broadcasts the appropriate sound from a speaker.


Deal
Round_Over
Showdown
Face_1
Game_Over
Bet_2
Bet_1
Discard
Face_2
transition = 1
transition = 1
game_over = 0
transition = 1
transition = 1
game_over = 1
transition = 1
equal_bets = 1
transition = 1
equal_bets = 0
transition = 1
transition = 1
transition = 1
transition = 1

Video Controller:

The Video Controller takes as inputs various states of the game; the current hand of each player, each player's current score (chip count), current bet, current number of chips in the pot, and the current stage within a round (1st card dealing, 1st round of bets, card switching, 2nd round of bets, or end of round). It will output on the computer monitor an image of a poker table with the players' cards, scores, number of chips bet, and instructions for the player or a declaration of the winner at the end of the round. A possible layout of the screen is below. The monitor display will update at the start of every stage within a round according to the states of the inputs, so it can essentially be modeled as an FSM.

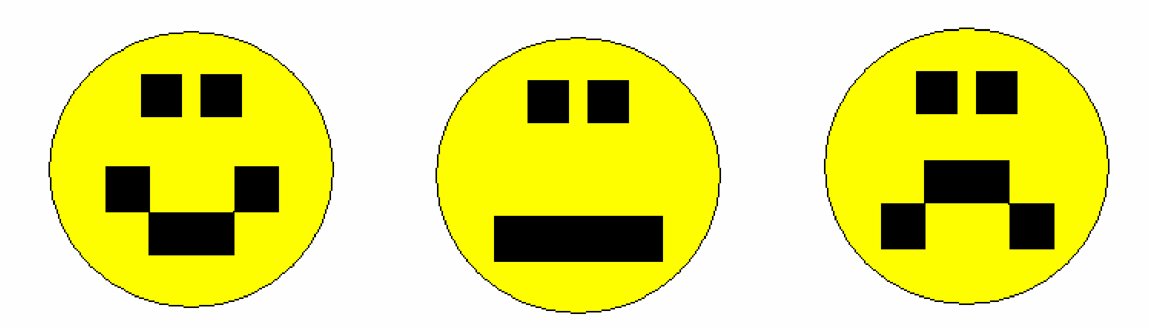
To design the Video Controller, I will divide it into a card module (which displays a card with a particular suit and value), chip module (creates a visual representation of a chip), and character string display. Each of these modules outputs some pixilated image. The complete video display will then be some combination of these images at various locations on the screen. I will use the xvga module of lab 4 as the basis for the Video Controller module, sending the appropriate signals to the DAC (hcount, vcount, pixel, etc.)



The submodules mentioned above can be easily tested by simply displaying an instantiation of each module (e.g. a single card) to the monitor. Once each submodule is functional, I will try positioning the different instantiations of each module on the screen to create a sample poker table setup. Then I will test out the FSM by taking inputs from the labkit components (switches, buttons, etc) to make sure the monitor display updates appropriately.

Facial Feature Detector

The Facial Feature Detector will detect the facial expression of the player (smile, straight face, or frown). Since it will be too difficult to detect such features on a human face, we will create three simple “masks” each with one of the above expressions easily detected (possible examples shown below) by dividing the image into a grid.



The input to this module will come from a video camera which monitors the player. Instead of interpreting a constant feed of video data, we will probably have the player press some button when he/she wants his/her new facial expression to be registered. This module will output one of three possible values (representing the three different facial expressions) to the game controller, which will modify the computer’s strategy accordingly.

Since this module only returns one of three possible outputs, it shouldn’t be too difficult to test. We could have three LEDs representing the three possible facial expressions. Then, by placing one of the masks in front of the video camera, we can see if the module is interpreting the image correctly by seeing which (if any) of the LEDs light up.