**The problem solved in this assignment**

We need figure on how to run an online game on a single severs. This program mainly use knowledge in term of socket programming, process, asynchronous communication and signal.

**This program is formed by three parts:**

**1. server.c**

This program is used to check the connections between each client. We will determine when we start a new game and how many players will join in each round by the protocol we defined in this code. After finished each round, we will recheck each connection until next round starts.

**2. dealer.c**

This program constructs a platform for multi-players. Dealer will game with each player simultaneously. According to the demand of each player, dealer will determine ‘hit’ or ‘stand’ the card. Finally, dealer will gain the results of all players.

**3. client.c**

This program needs to realize the function of communication. Each client will connect to servers and play with dealer by specific game protocol.

**The running process:**

**1. The choice of communication protocols**

Because the game among dealer and players is instant and consistent, TCP is the desired choice for keeping stability of communication.

**2. The process when servers accept all players**

Stack is used to save the current player. Server firstly entries into the service loop and wait the available connecting. When server receives the first confirmed information, the game will count down 30 seconds. During this period, any confirmed player will be allowed to join the game. If in this period, we gather all four players then game starts. I used timeout parameter to check whether client sent confirmed information in certain time.

**3 How dealer works**

After game starts, dealer is working in the server. First, dealer need fork() server processes. Each process is like a hand of dealer and connect to each player. In the parent process, we use signal (wait\_child) to block and wait all child processes signal. Then we signal to each process to continue game respectively. After that, parent process will wait all child processes end by waitpid function. When this loop is over, we begin to next round of game.

**4 How dealer\_hand works**

For each child process, they need to execute dealer\_hand function to play game with each player. Each player is independent. According to order of each player, each dealer\_hand will hand card to player until burst, stand or violate rules. After that, child process signals to their parent that player completes the game. Then, we enter to stage of “wait\_parent”, which parent process will continue to execute the following orders. If any player hasn’t burst or been kicked out, they need to keep executing “dealer” process until process exits

**5. Asynchronous and synchronous**

There are two execution stages for dealer process. First, dealer hands cards to different clients at same time and receive the order from clients. In this stage, all the hand processes are independent. This is asynchronous communication. Second, if all players stand or burst or violate rules, dealer must check them one by one. This is synchronous communication. Thus, by using signal, child process is able to block program and wake child processes, which realize the synchronous communication.

**6 Signal**

Through passing signal between parent and child process, we can easily achieve the function for synchronous and asynchronous. This mechanism fits for our multiplayer environment. However, when multi-signal arrive to parent process simultaneously, there is a small chance to cause signal overlap and this possibility do exists.