**SET09109 FUNDAMENTALS OF PARALLEL SYSTEMS  
CHALLENGE DESIGN DOCUMENT  
AUTHORED BY: ROSS MCLAREN (40174116)  
 JACK YORSTON (40174030)  
 BARTHOLOMEW PERSSON (40183743)**

**Documentation**

There are two major additions to the supplied implementation: enforcing turn taking for participating players and updating the board during gameplay for all players.

Each player has an ID called ‘myPlayerID’. For it to be a player’s turn this ID value must be equal to a variable, called ‘turnCounter’, created in the ControllerManager and passed to the PlayerManager via the GameDetails class. Whenever a player has selected a mismatching pair and presses the ‘NEXT TURN’ button on the UI an instance of the new NextTurn class (which stores the player’s ID) is written to the ControllerManager where it calculates the next value for ‘turnCounter’ so that the next player’s turn can start. Depending on whose turn was last it either increments the value of the ‘turnCounter’ by 1 or sets it back to 0 (starting back with the first player in the game) and sends it back to the PlayerManager.

In the supplied implementation the board was updated to the players when a matching pair was claimed, sending the ‘chosenPairs[]’ values, as well as the player and game ID, to the ControllerManager. Now those values are also sent to the ControllerManager through a new class called Update. An instance of Update is written to the ControllerManager whenever a single square is exposed, as well as when a mismatching pair is found. When this happens the ControllerManager writes those values back to the PlayerManager where a new case in the outerAlt switch (called CONTROLLER) updates the board for all players using the values received.

As there are no new communication channels, the process network diagram (shown below) is the same as the original implementation. To help visualise the discussed additions the channel interaction sequence can also be found below.

**Process Network Diagram**



**Channel Interaction Sequence**

