Using keyboard buffer

Here is what I found out about using **c=XUartLite_RecvByte(uartRegAddr);**

XUartLite_RecvByte stores the data read into a buffer (FIFO), and only passes that data to char 'c' when a button is pressed. There a few problems that come with this. For a game, if I press a large string of characters in a row, the game will process each character each run cycle. So if I press "WASDSDSD." On the first loop it will read W, then A, then S, then D... and so on*. So in a game like snake, where you only care about the most recent button press, this would be problem, since the snake my move up, left, down, right... before the user's most recent key stroke is actually registered.
Another issue with using XUartLite_RecvByte is that it holds at this command until a stroke is pressed. So if you use this function in a game play loop, the game will only update when you press a key, and then hold until another key is pressed.

You can use the function **'XUartLite_IsReceiveEmpty'** to have the game run regardless of if a key is pressed. The function returns a 1 when the buffer is empty and a 0 if the buffer is not empty. You can use !XUartLite_IsReceiveEmpty to have the game loop until a character is added to the buffer.

*Note, the problem with typing in lots of characters still exists with isReceiveEmpty function.... the buffer would have to be cleared each time and I could not find a function after searching through forums that resolved this aspect for my code.

Hopefully this will help people in future classes,

Cadet Second Class Ryan D. Clendening