and there is not some special move to make, so the only thing left to do is to claim a square. But, wait, not all squares are equal strategically. So the order of evaluation (state order) is critical.

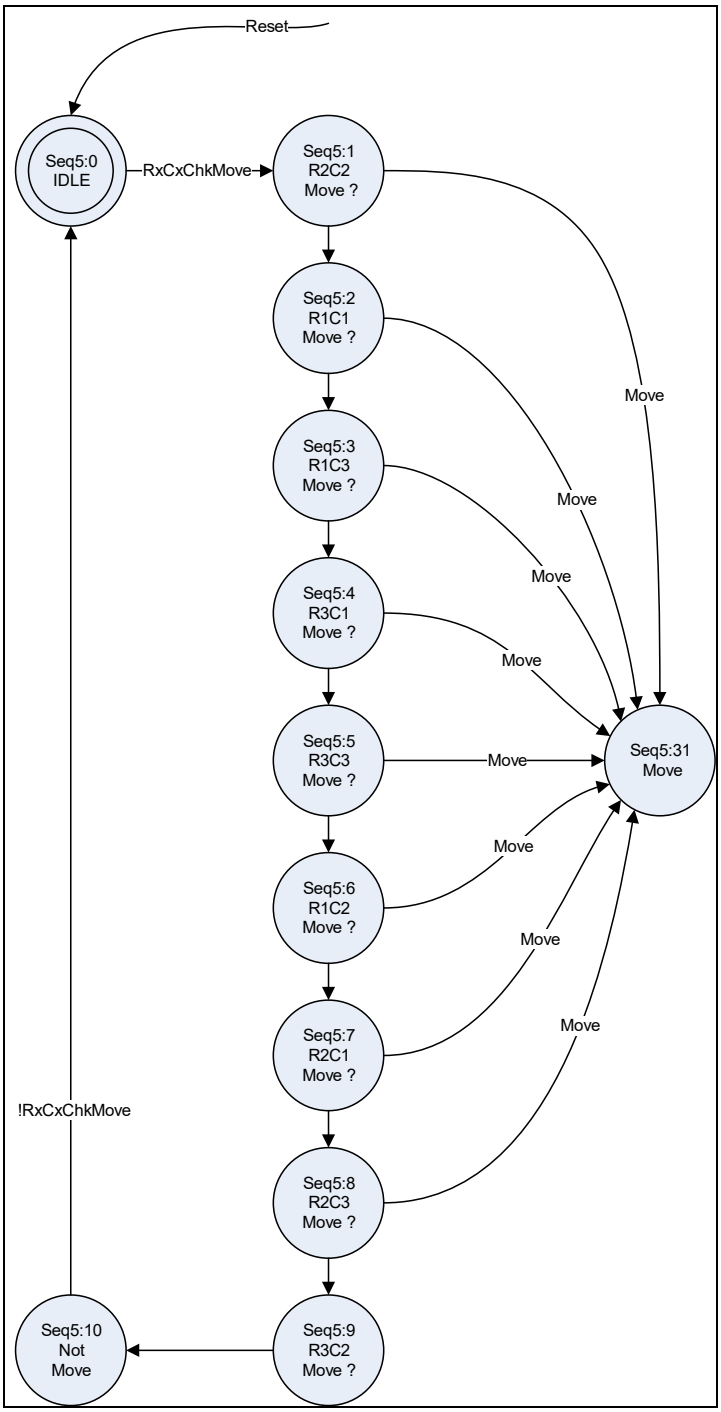


Figure 12 Move State Machine Diagram

7 HMI / GAME PLAY INTERFACE

I have written a simple program that will run on the Weintek family of touch screen HMIs. I am running the code on the MT8971iE HMI that has a 400 x 800 pixel resolution and an Ethernet connection. I am using Modbus TCP/IP as the interface protocol between the HMI and the TRI PLC. The following is a screen shot of HMI during game play:

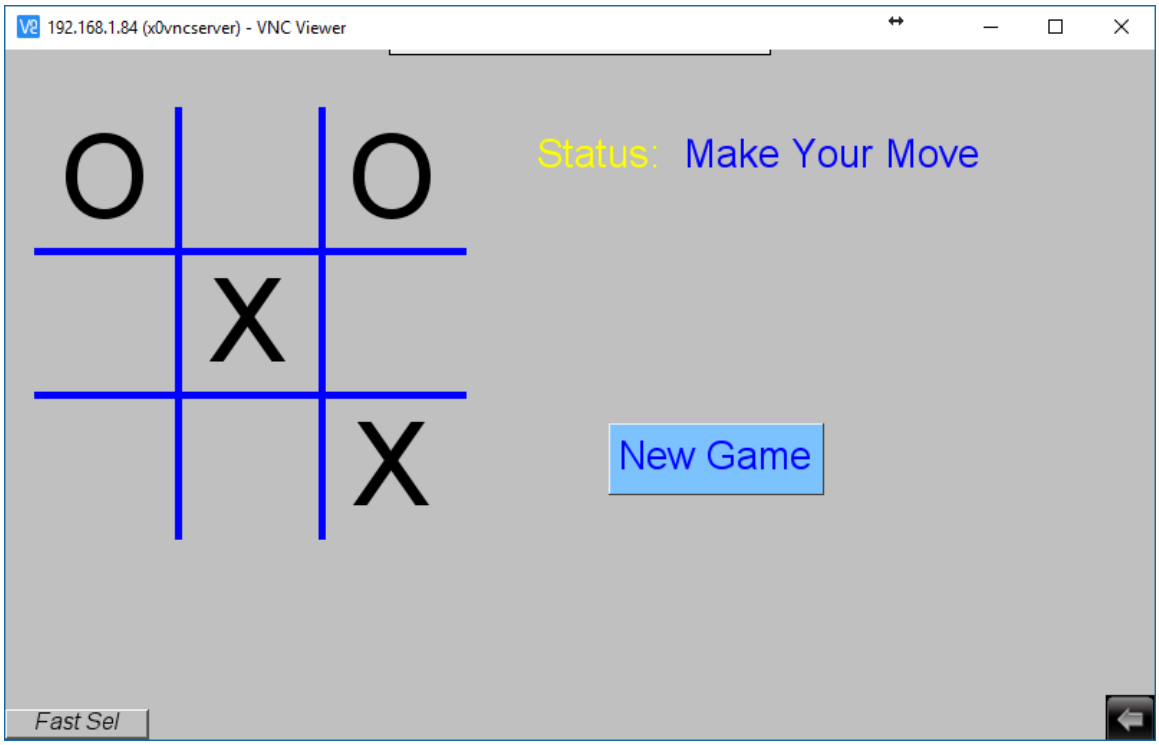


Figure 13HMI Screen during Game Play

In the following subsections I will describe a bit of how the game paly interface works

7.1 HMI IMPLEMENTATION

The 3x3 game board is used both display the current status of the game and to allow the human to play by touching an empty square. Game status is shown in the upper right side of the screen. The "New Game" button allows the human to start a new game.

7.1.1 Game Board on HMI

The 3x3 game board is implemented using a 3x3 array of HMI Toggle Switch objects. The Toggle switch objects are used to set corresponding RELAYS in the PLC via Modbus TCP/IP. This is much as you might have guessed. However the game board needs to display 3 possible states for each of the 9 squares on the game board:

1. An Empty, available, square
2. A square occupied by an "X"
3. A square occupied by an "O"

By my count, this is 3 different states. The HMI's toggle switch object an only display 2 states on off. So I had to use another HMI object, the Word lamp. Word lamps can be used to display multiple states, so they can handle the display of " ", "X" or "O" easily. But the Word Lamp can't change the state of a RELAY in the PLC.

So for every square on the game board, there are 2 HMI objects stacked on top of each other:

1. A toggle switch object that can set a single RELAY in the PLC. This object has no graphic so it is invisible or transparent.
2. A Word lamp that displays the status of the square by reading a 16-bit value for the PLC. The word lamp is programed to display either " ", "X" or "0". Three states!

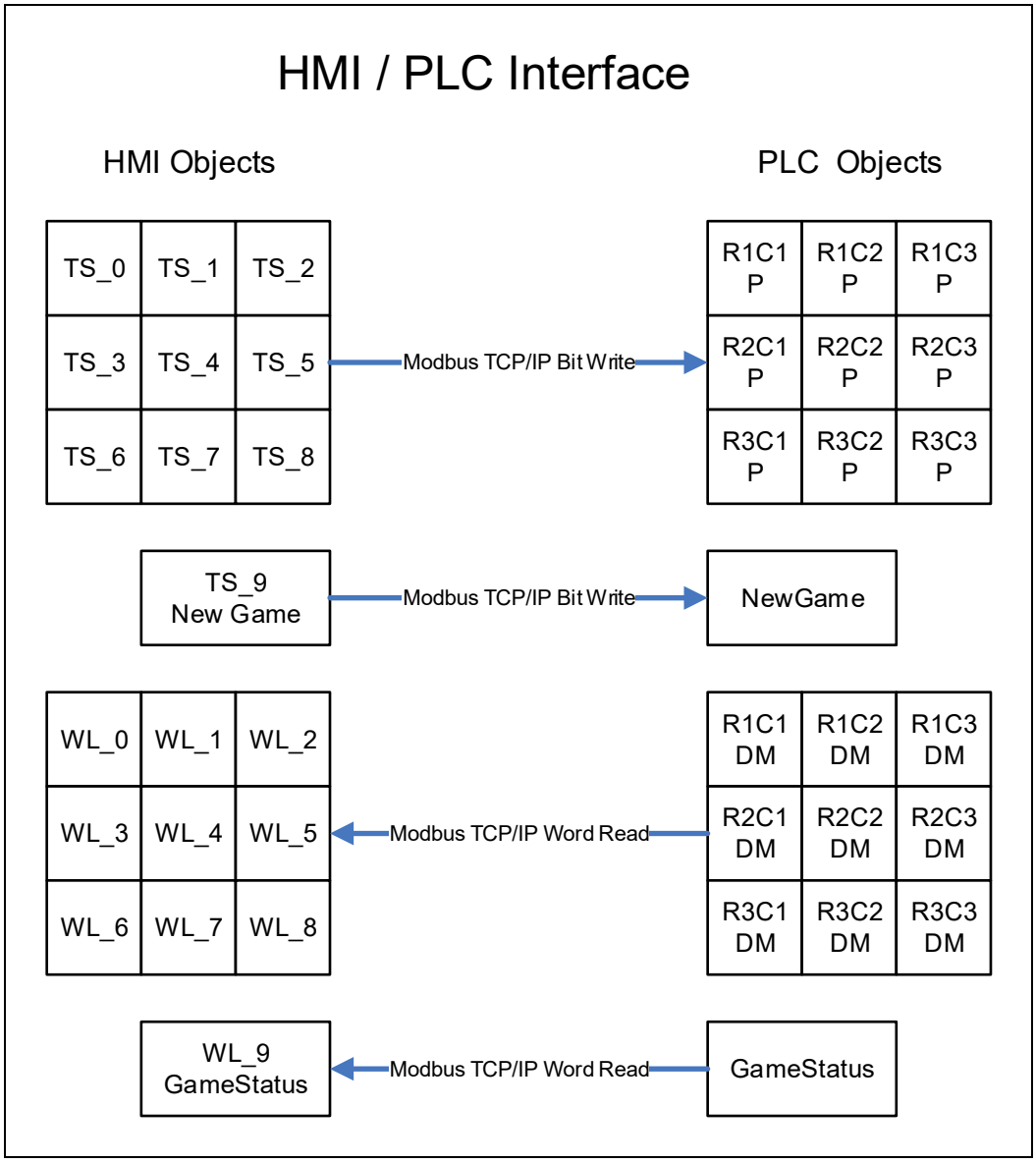


Figure 14 HMI/PLC Interface

7.1.2 Game Status on HMI

The PLC, also, manages a number of RELAYS to indicate the game status. The HMI reads the 16 bit PLC register as GameStatus, RELAY[2]. The PLC manages the individual RELAYS Play, Draw and Win as shown in the following table:

Table 3 GameStatus Word / RELAYs

GameStatus, RELAY[2]	
I/O #	Name
17	Play
18	Draw
19	Win

The HMI uses a Word Lamp object that is using the "LSB" mode to translate the 16-bit Game Status value into the following text :

Table 4 Game Status Encoding

Play	Draw	Win	State	Status Message
0	0	0	0	Computer Working
1	0	0	0	Make Your Move
0	1	0	0	Cat's Game
0	0	1	0	I Won!

7.1.3 Game Play on HMI

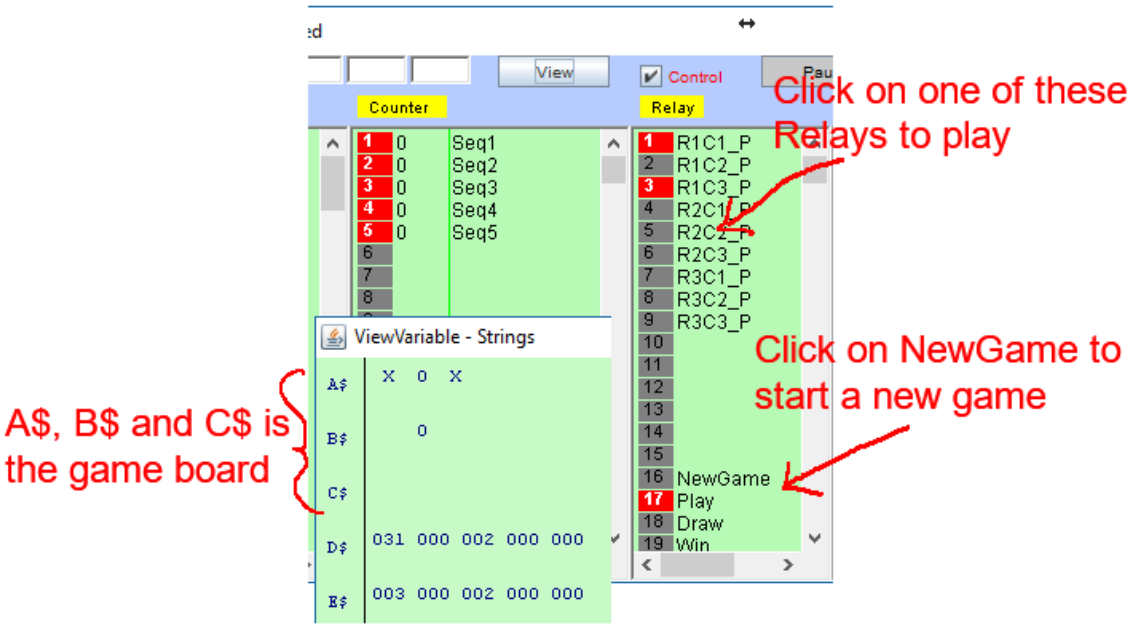
The HMI sets one of 9 RELAYs in the PLC indicate which square that the human wants to play. These relays represent the 3x3 game board and are named R1C1_P .. R3C3_P. Ladder logic in the PLC detects when a square has been played by the human and this starts the main state machine running to figure out what the PLC will play in response. The PLC is responsible for clearing these RELAYs.

7.1.4 3x3 Game Board on HMI

The HMI "reads" 9 consecutive words in the DM[] memory array to display the current game board status. These DM[] locations are named, via the #Define mechanism, as R1C1_DM through R3C3_DM. The PLC is responsible for maintain these DM[] locations.

7.2 How to Play the Game without an HMI

You can simply run the PLC program using the i-TRiLOGI simulator or On-line monitoring if the program is running on real hardware. Click on the "View" menu button and scroll to the String Variables. The following is a slightly edited view of what you need to see and what you need to do:



8 LAST WORD

Have fun. Remember that I wrote this stuff to finish something that I started in another life.

If you have questions about this stuff, you can always ask them on the forum.

Gary Dickinson

gary.s.dickinson@gmail.com