1ST MAKER SPACE
Design.
Build.
Sustain.

# Frog Microcontroller Trainer (FMT): Tic Tac Toe

**Length of Time: 2 days x 55 Minutes**

*Recommended for Developmental*

**Contents**

**Overview:**

This lesson introduces students to how a computer program creates an interactive game using real hardware. Students use the Frog board to play Tic-Tac-Toe while exploring how inputs (potentiometer and buttons) control actions, how outputs (OLED display) communicate information, and how logic and data structures (arrays and conditionals) manage the game.

Through gameplay and guided analysis, students learn how the program tracks the board, checks for wins, and uses a simple AI strategy to make decisions. The lesson builds foundational understanding of algorithms, user input, and program flow, while giving students a hands-on way to test, analyze, and improve a working system.

**Objectives:**

*Students will be able to:*

Explain how user actions control a program using buttons and a potentiometer.

Describe how an array represents the game board.

Explain how the program uses logic to check for a win or draw.

Propose and implement (or pseudo-code) small modifications to the describe how a simple AI makes decisions in a game.

Test and evaluate how well a program works.

## Computer Science Teacher Association Standards:

- 2-AP-13 – Decompose problems and subproblems into parts to facilitate program design and implementation.
- 2-AP-17 – Systematically test and refine programs using a range of test cases.
- 3A-AP-15 – Justify the selection of specific control structures in terms of readability and performance.

## Materials:

- Frog Microcontroller Trainer from 1st Maker Space
- USB Power Cord
- Arduino IDE
- Program loaded onto each Frog
- PC or Mac
- Chromebooks if using Arduino Create for Education App
- Copies of Student Handout

## Preparation:

- Confirm the Frog boards, library, and Tic Tac Toe sketch compile and upload successfully.
- Run the program yourself and press each button to see the behavior.
- Decide which extensions are appropriate for your students' level.
- Print or distribute the student handouts.

## Background Information:

This Arduino program allows students to play Tic-Tac-Toe against the Frog board using the OLED screen, potentiometer, and buttons. Students control position selection with the potentiometer and confirm moves using Button 1. Button 2 resets the game. The program
also includes a simple AI that attempts to win, block the player, or make smart strategic choices.

What are the program functions?
Main control functions:
setup()

Initializes the program when the board powers on.

What it does:

* Starts serial communication
* Starts the OLED display
* Sets the button pins as inputs with pull-up resistors
* Clears the display

* Calls `resetGame()` to initialize the board
* Calls `showOpeningMenu()` to display instructions and wait for the player to start

loop()

This is the main control loop that runs over and over while the program is on.

What it does:

* Reads the potentiometer to choose a board position
* Maps that value to `selectedPos` from 0–8
* Checks whether the reset button was pressed
* Checks whether the confirm button was pressed
* Places the player's move if the chosen square is empty
* Calls `checkWinner()` and `isBoardFull()` after a move
* Runs the Arduino turn by calling `makeAIMove()`
* Calls `drawBoard()` to refresh the OLED display

showOpeningMenu()

Displays the title and start instructions before the game begins.

What it does:

* Prints the game title and controls on the OLED
* Waits until the player presses SW1 or SW2
* Starts the game when a button is pressed
* If SW2 is pressed, it also calls `resetGame()`

drawBoard()

Handles all OLED output during the game.

What it does:

* Clears the screen
* Displays the current game status:
  * `YOU WIN!`
  * `ARDUINO WINS!`
  * `DRAW!`
  * `Your Turn (X)`
  * `Arduino Turn...`
* Draws the 3×3 Tic-Tac-Toe grid

* Draws the X and O marks already stored in the board array
* Highlights the currently selected square
* Displays the selected position number at the bottom

resetGame()

Resets the game so a new round can begin.

What it does:

* Clears all 9 spaces in the `board[]` array
* Sets `playerTurn = true`
* Sets `gameOver = false`
* Clears the winner
* Resets `selectedPos` to 0

checkWinner()

Checks whether either player has won.

What it does:

* Checks all rows
* Checks all columns
* Checks both diagonals
* Returns `true` if three matching symbols are in a winning line
* Returns `false` if no winner is found

isBoardFull()

Checks whether the board is completely filled.

What it does:

* Looks through all 9 board spaces
* Returns `true` if there are no empty spaces left
* Returns `false` if at least one empty space remains

This function helps determine whether the game ends in a draw.

makeAIMove()

Controls the Arduino’s move.

What it does:

1. Tries to place `O` in a spot that wins the game
2. If it cannot win, it checks whether the player could win next and blocks that move
3. If no win or block is needed, it takes the center square if available
4. If the center is taken, it tries a corner
5. If no corner is open, it takes any remaining open space

This makes the AI **rule-based** and strategic, but not perfect.

Main Variables That Support the Control Functions

board[9]

Stores the 3×3 game board as a one-dimensional array.

playerTurn

Tracks whether it is the human player's turn or the Arduino's turn.

selectedPos

Stores which board space is currently selected by the potentiometer.

gameOver

Indicates whether the game has ended.

winner

Stores the result:

* `''X'` = player wins
* `''O'` = Arduino wins
* `''D'` = draw

| Lesson Elements | |
|---|---|
| **Introduction**<br>10-15 minutes | Show the Frog hardware and demonstrate the potentiometer and buttons.<br>- Introduce Tic-Tac-Toe as a computational problem. |

| | |
|---|---|
| **Explore Inputs and Outputs** – 15 minutes | Students observe how turning the potentiometer changes the selected grid cell.<br>- Discuss digital vs. analog input. |
| **Understanding Game Logic**– 20 minutes | Students examine how the board array stores X and O positions.<br>- Teacher walks through the win-checking function. |
| **Understanding the AI**– 15 minutes | The AI in this program follows a simple set of rules rather than "thinking" ahead like a human.<br><br>Each time it takes a turn, it checks the board and makes decisions in this order:<br><br>Try to win:<br>It looks for any move that would give it three in a row. If it finds one, it takes that spot.<br><br>Block the player:<br>If it can't win, it checks if the player is about to win on their next move. If so, it places its mark there to block.<br><br>Make a strategic move:<br>If neither of those apply, it chooses a good available space (often center, corners, or a random open spot).<br><br>So the AI is rule-based, not predictive—it reacts to the current board rather than planning multiple moves ahead. |
| **Reflection** – 10 Minutes | Students write a brief explanation of how user input affects program flow. |
| **Career Exploration:**<br>● A good resource is the IDOE Career Explorer database.<br>● Another good resource for career information is the Bureau of Labor Statistics | |
| **Practical Application:**<br>Game development, Decision-making systems, User interface design<br>AI in simple automation systems, Interactive kiosks or devices | |

## Additional Resources:

- 1st Maker Space Frog Microcontroller Library

- 1st Maker Space Learn Arduino with the Frog Microcontroller

## Performance Assessment/Check for Understanding :

- Was the student able to successfully load the sketch to the FMT?
- Check student handout for responses.

Additional Feedback: