Michael Tin (SID: 862321571)

CS120B Custom Laboratory Project Report

"Connect 4"
Name: Michael Tin
Submission Date: December 11, 2024

A. Introduction

This Custom Laboratory Project is a rendition of "Connect 4", which is a popular, turn-based board game played on a 6x7 vertical grid. The players take turns dropping colored checkers into the top of the grid, where they fall to the lowest available position in that column. The aim is to be the first player to form a horizontal, vertical, or diagonal line with four checkers.

The final rendition differs from the proposed rendition in some small ways that do not affect the basic functionality of the game at all. The differences are as follows:

- The Joystick Button is replaced by two separate buttons (Left and Right).
- The 4D7S Display is replaced by the 1D7S Display.
- The Passive Buzzer plays music, rather than making sound effects.
- There are two separate LEDs, instead of a single RGB LED.
    - The White LED turns on when the game is being played.
    - The Blue LED turns on to indicate USART communication is successful.

There are no major bugs or errors in the game that limit usability or basic functionality.

The complexities implemented were the ST7735 128x128 LCD, SN74HC595N 8-Bit Shift Register, Passive Buzzer, and USART Communication (described in next section).

Michael Tin (SID: 862321571)

B. Build-Upons

The following build-upons were SUCCESSFULLY implemented:
HiLetgo ST7735 128x128 LCD

- The Screen displays the current state of the game at all times.
- A typical Connect 4 Board has 6 rows and 7 columns, totalling 42 possible locations for the checkers. The Screen represents each of these locations with a 21 x 18 square.
- The square is blank (white) if there is no checker, red if Player 1 has dropped their checker in that location, and yellow if Player 2 has dropped their checker in that location
- The Screen refreshes each second (1000 ms, or 1 Hz).
- The Screen's functionality is implemented using SPI communication. The basic functionality and the C++ header file was provided by the TA.

Texas Instruments SN74HC595N 8-Bit Shift Register

- The Shift Register is responsible for sending the data to the 1D7S Display which serves as a timer. The Shift Register allows the 1D7S to be updated without needing to use an Arduino port for each segment.
- The three important Pins of the SN74HC595N are the Data, Clock, and Latch Pins.
- The values for the numbers displayed are stored in an array where each entry is made up of an 8-bit binary value. A custom shiftOut() function (which is modeled off the functionality of the Arduino library shiftOut() function) transmits the information stored in the array.
- The first bit of data is sent to the Data Pin, and then the Clock Pin is pulsed. This repeats for all 8 bits, after which the Latch Pin is pulsed.
- The entire process repeats each time a number is sent to the 1D7S Display.

Arduino Passive Buzzer

- The Passive Buzzer plays a song when the game is being played.
- The Passive Buzzer utilizes the Arduino's Timer1 PWM (PORTB1).
- The Prescaler of Timer1 is set to a value of 8. Then, different notes are generated by updating the ICR1 value (based on the formula from Lab #7). OCR1A is set to half of ICR1 to create a square wave that resembles a sine wave.

USART Communication

- A basic USART Communication is implemented to accommodate the Passive Buzzer.
- This was done primarily because the Passive Buzzer could not receive enough power from the first Arduino because of the other components wired to it.
- If the game is idle, Arduino #0 sends “1” to Arduino #1. In response, Arduino #1 turns off the Blue LED and turns off the Passive Buzzer.
- If the game is being played, Arduino #0 sends “0” to Arduino #1. In response, Arduino #1 turns on the Blue LED and turns on the Passive Buzzer. Here, the Blue LED indicates that USART Communication has successfully been established.

Michael Tin (SID: 862321571)

The following build-upons were NOT implemented:
EEPROM

- In theory, this would save a “win streak” for the game.

C. User Guide

Starting the Game:

- Ensure both Arduinos are ON. The LCD displays “Connect 4: Idle”.
- Pressing the Left Button starts the game.

Playing the Game:

- During gameplay, the LCD displays “Connect 4: Play”. Additionally, the Screen displays the current state of the board at all times, and the Passive Buzzer plays music.
- Player 1’s turn begins. The LCD displays “Player 1 Turn”.
- Player 1 uses the Right Button to select a column. Pressing the Right Button increments the selection forward and wraps back around after reaching 7. The selection is shown in the bottom right corner of the LCD.
- Player 1 has until the timer runs out to make their selection; the time is indicated on the 1D7S Display. After this time, the selection is made and the Screen updates accordingly.
- The checker is dropped into the lowest available space in the selected column. If the column is full, nothing happens.
- Player 2’s turn begins. The LCD displays “Player 2 Turn”.
- The gameplay repeats until somebody has won (a row, column, or diagonal of four identically colored checkers) or the board is full (top row is filled).
- The LCD displays “Connect 4: End” and one of the following (based on the result):
    - “Player 1 Wins!”
    - “Player 2 Wins!”
    - “Game Was Tied!”

Resetting the Game:

- Pressing the Left Button during gameplay resets the game.
- Pressing the Red “Reset” Button on the Arduino restarts the system after the game ends (a player wins, or the board is full).

Michael Tin (SID: 862321571)

D. Hardware Components

- Computing
    - ATmega328 Microcontroller (2x)
    - Texas Instruments SN74HC595N 8-Bit Shift Register (1x)
- Inputs
    - Button (2x)
- Outputs
    - HiLetgo ST7735 128x128 LCD
    - 1-Digit 7-Segment Display (1D7S)
    - LCD Screen (16x2)
    - White LED
    - Blue LED
    - Passive Buzzer

E. Software Libraries Used

No external C++ or Arduino libraries were used. However, the following external resources were utilized:

Provided by TA:

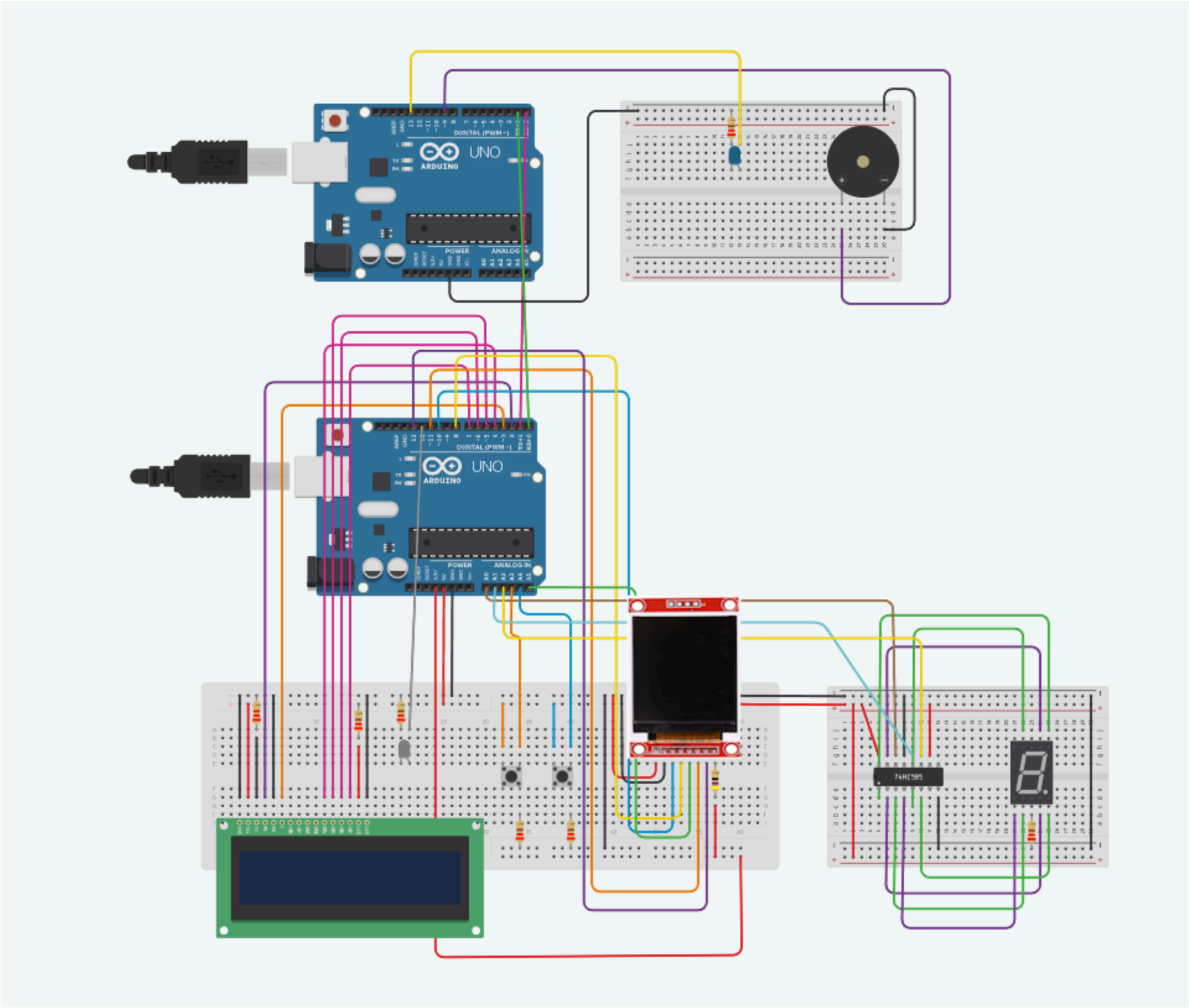
- helper.h
- LCD.h
- periph.h
- spiAVR.h
- timerISR.h
- usart_ATmega328p.h

External Resources:

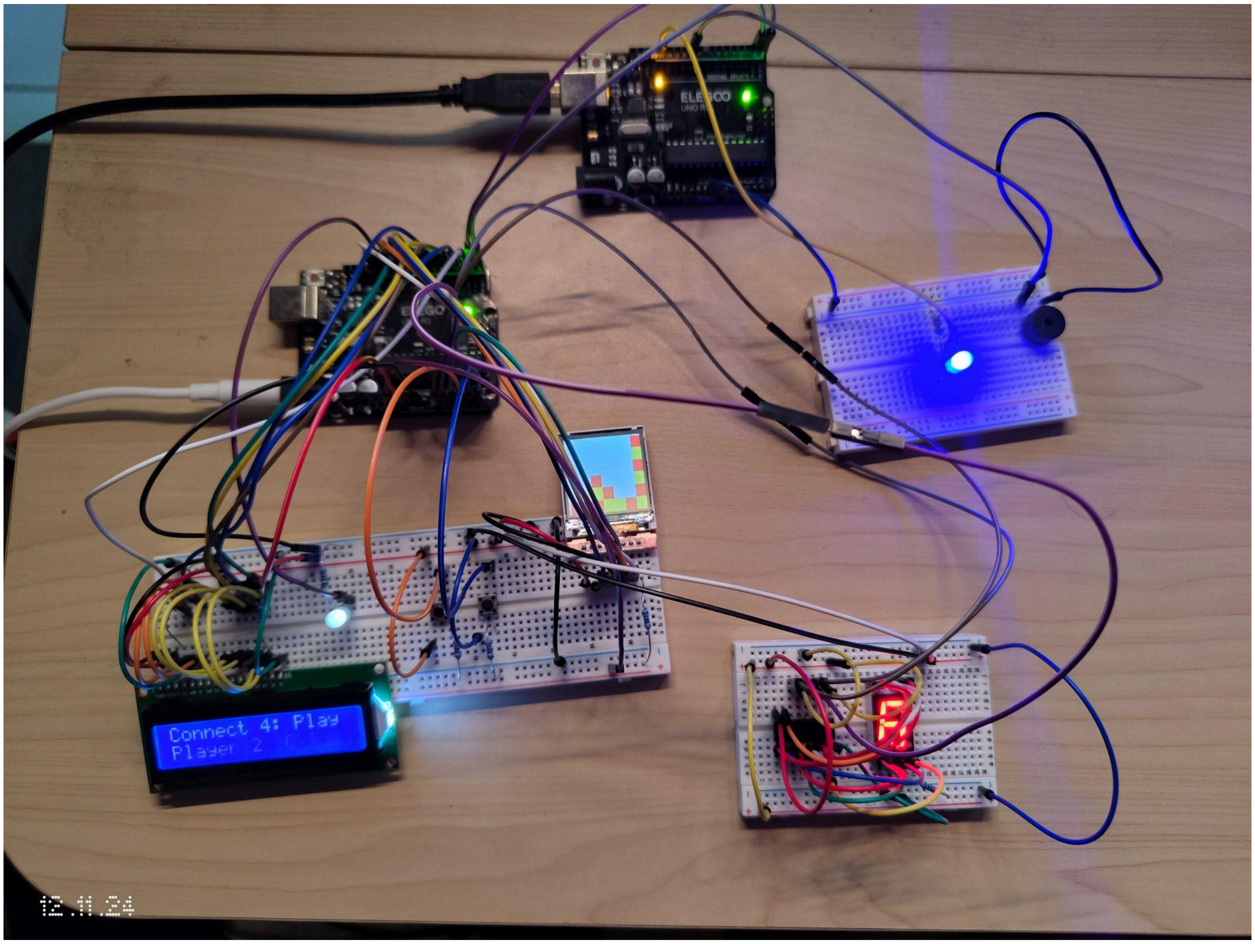
- https://stackoverflow.com/questions/39062111/java-how-to-check-diagonal-connect-four-win-in-2d-array
    - This information was used to implement a function to check if there is a winner.
- https://demmel.com/ilcd/help/16BitColorValues.htm
    - This information was used to reference 16-bit color values.

Michael Tin (SID: 862321571)

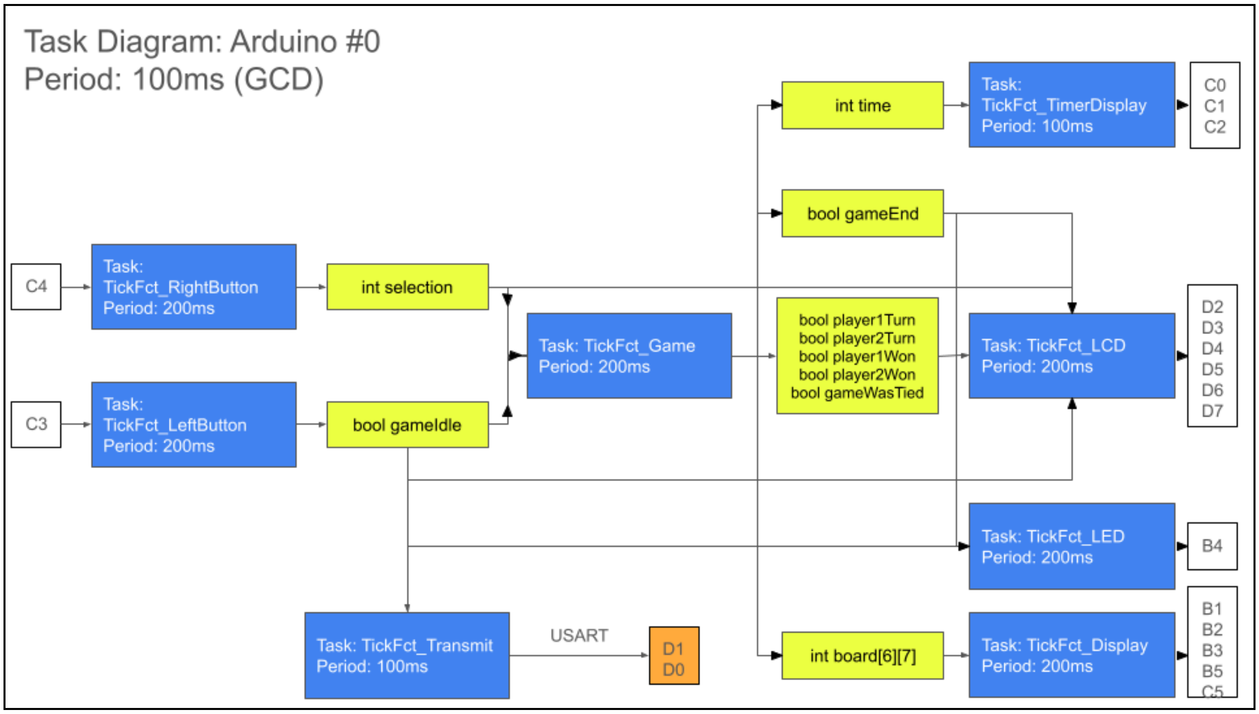
F. Wiring Diagram

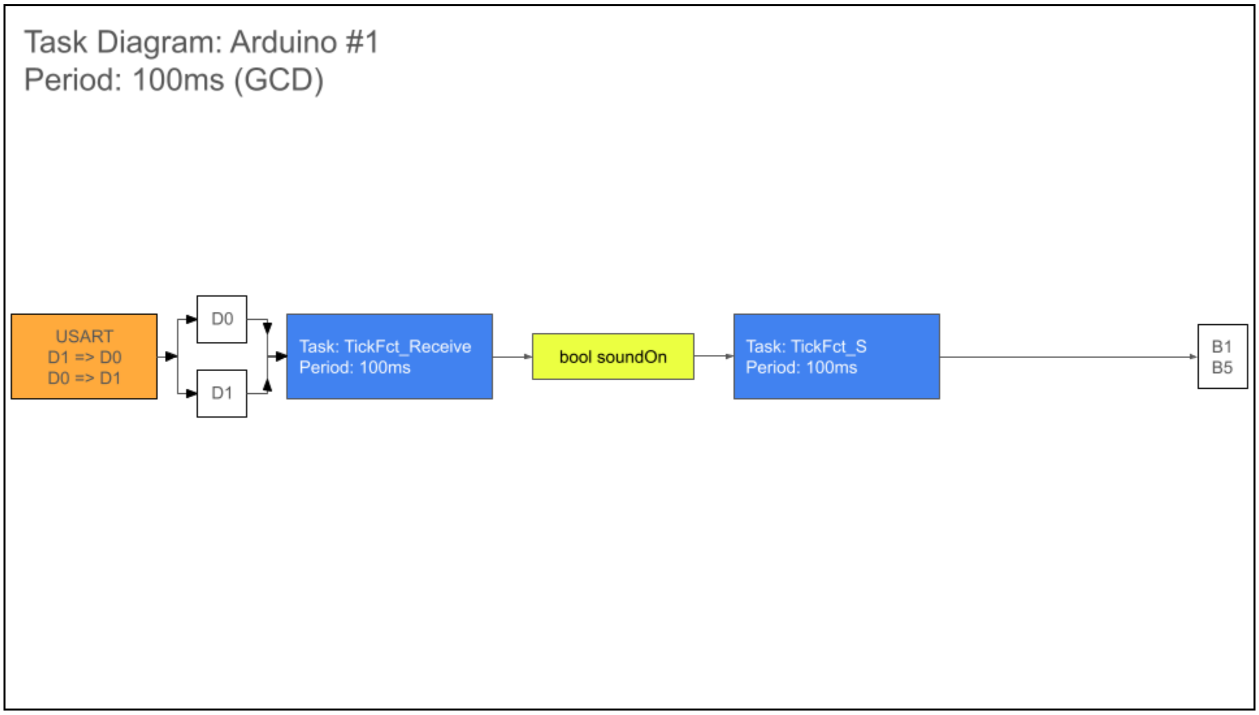


Michael Tin (SID: 862321571)



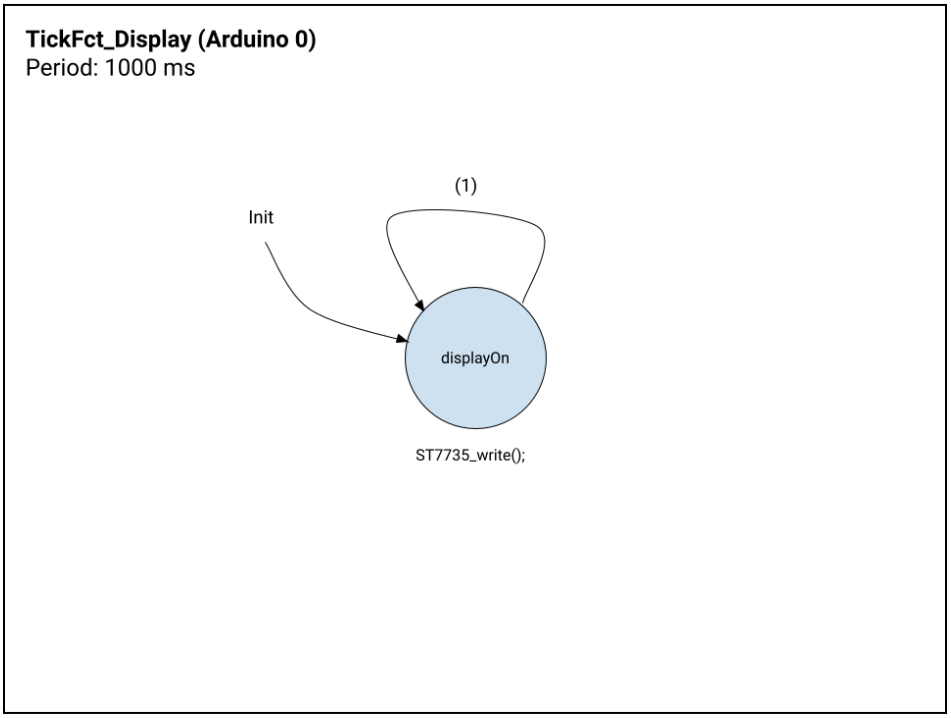
G. Task Diagram

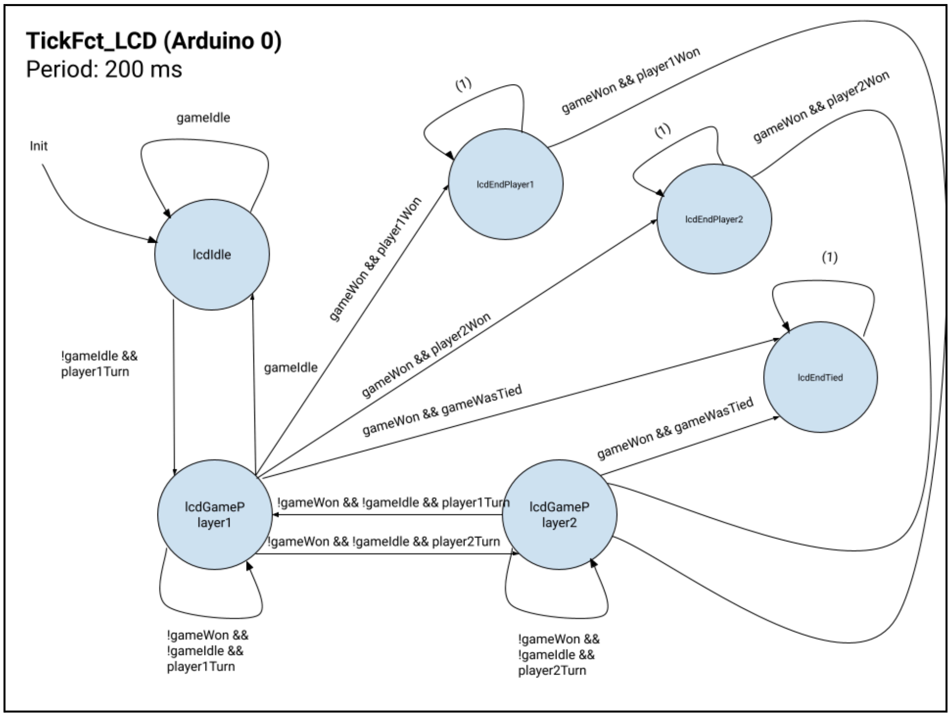




Michael Tin (SID: 862321571)

H. SynchSM Diagrams

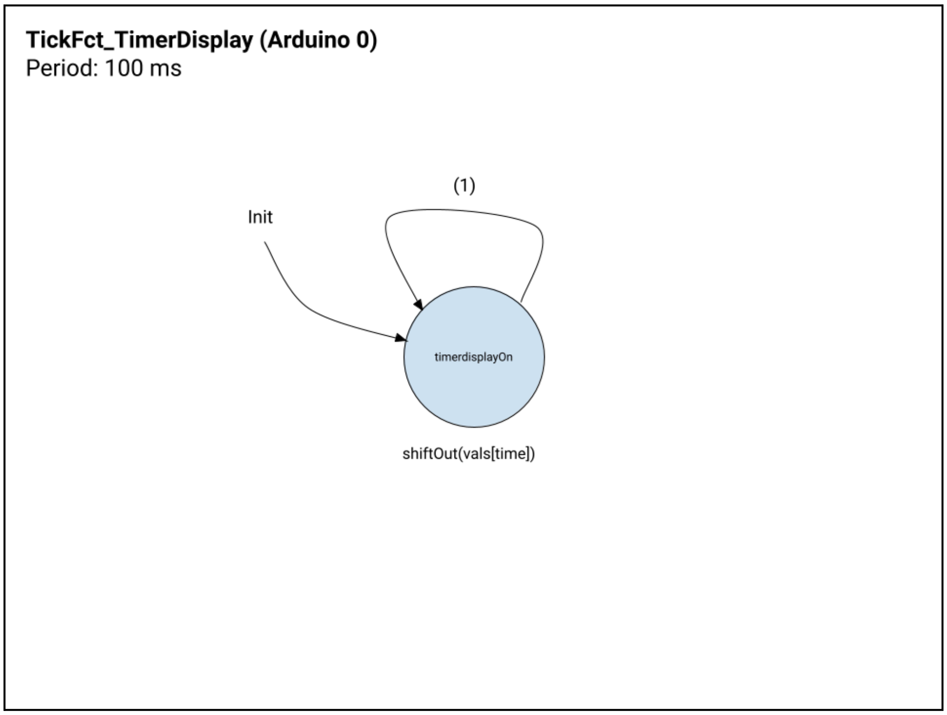




Michael Tin (SID: 862321571)

State Actions referenced here to reduce clutter on the SynchSM Diagram.

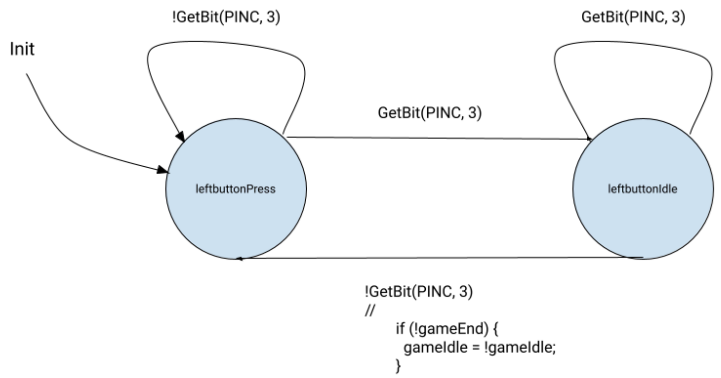
State	**Message Displayed**
lcdIdle	“Connect 4: Idle”
lcdGamePlayer1	“Connect 4: Play” “Player 1 Turn Col: (Selection)”
lcdGamePlayer2	“Connect 4: Play” “Player 2 Turn Col: (Selection)”
lcdEndPlayer1	“Connect 4: End” “Player 1 Wins!”
lcdEndPlayer2	“Connect 4: End” “Player 2 Wins!”
lcdEndTied	“Connect 4: End” “Game Was Tied!”



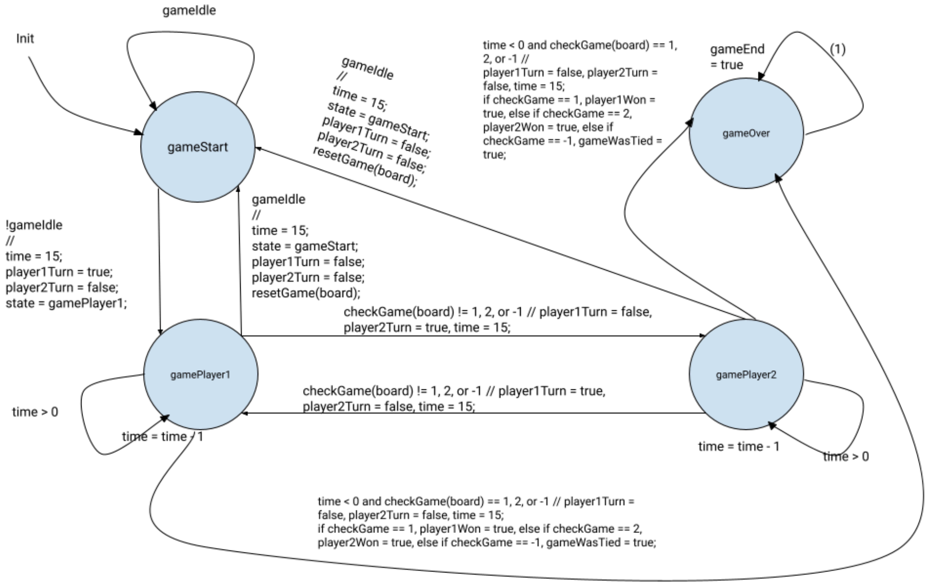
TickFct_TimerDisplay (Arduino 0)
Period: 100 ms

Michael Tin (SID: 862321571)

TickFct_LeftButton (Arduino 0)
Period: 200 ms

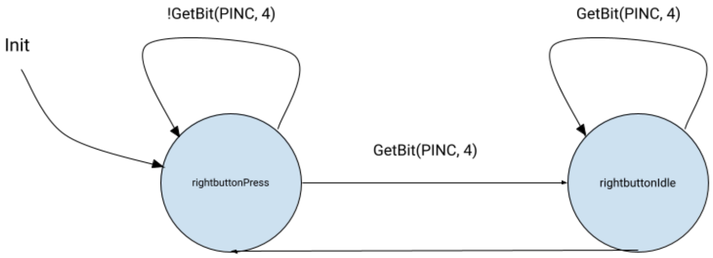


TickFct_Game (Arduino 0)
Period: 200 ms


gameIdle
Init
gameIdle
//
time = 15;
state = gameStart;
player1Turn = false;
player2Turn = false;
resetGame(board);
time < 0 and checkGame(board) == 1, 2, or -1 //
player1Turn = false, player2Turn = false, time = 15;
if checkGame == 1, player1Won = true, else if checkGame == 2, player2Won = true, else if checkGame == -1, gameWasTied = true;
gameEnd = true
(1)
gameStart
gameOver
!gameIdle
//
time = 15;
player1Turn = true;
player2Turn = false;
state = gamePlayer1;
gameIdle
//
time = 15;
state = gameStart;
player1Turn = false;
player2Turn = false;
resetGame(board);
checkGame(board) != 1, 2, or -1 // player1Turn = false, player2Turn = true, time = 15;
gamePlayer1
gamePlayer2
checkGame(board) != 1, 2, or -1 // player1Turn = true, player2Turn = false, time = 15;
time > 0
time = time - 1
time = time - 1
time > 0
time < 0 and checkGame(board) == 1, 2, or -1 // player1Turn = false, player2Turn = false, time = 15;
if checkGame == 1, player1Won = true, else if checkGame == 2, player2Won = true, else if checkGame == -1, gameWasTied = true;

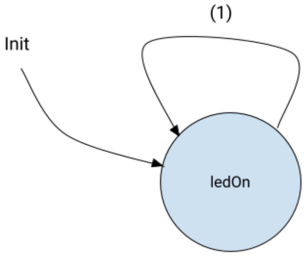
Michael Tin (SID: 862321571)

TickFct_RightButton (Arduino 0)
Period: 200 ms



```
!GetBit(PINC, 4)
//
        selection++;
        if (selection > 7) {
          selection = 1;
        }
```

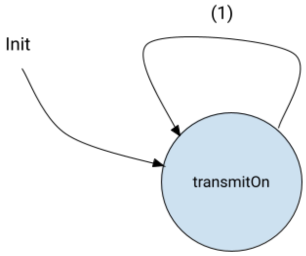
TickFct_LED (Arduino 0)
Period: 200 ms



```
if (gameEnd) {
  PORTB = SetBit(PORTB, 4, 0);
}
else if (!gameEnd) {
  if (!gameIdle) {
    PORTB = SetBit(PORTB, 4, 1);
  }
  else {
    PORTB = SetBit(PORTB, 4, 0);
  }
}
```

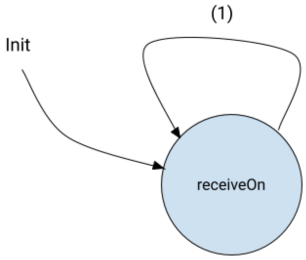
Michael Tin (SID: 862321571)

TickFct_Transmit (Arduino 0)
Period: 100 ms



```
if (USART_IsSendReady()) {
  USART_Send(gameIdle ? '1' : '0');
}
```

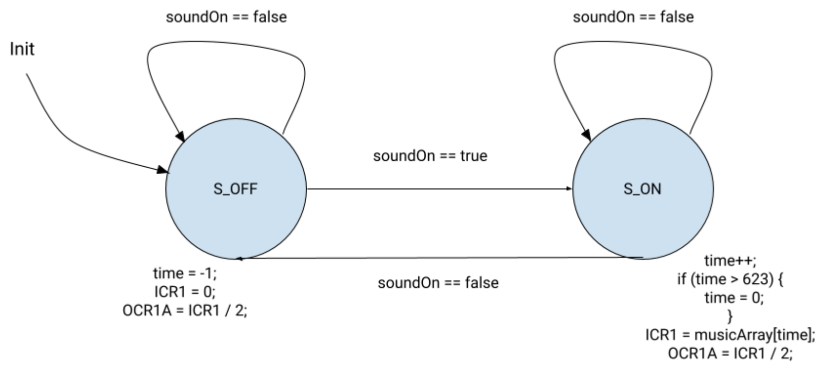
TickFct_Receive (Arduino 1)
Period: 100 ms



```
if (USART_HasReceived()) {
  receivedChar = USART_Receive();
  if (receivedChar == '1') {
    PORTB = SetBit(PORTB, 5, 0);
    soundOn = false;
  }
  else if (receivedChar == '0') {
    PORTB = SetBit(PORTB, 5, 1);
    soundOn = true;
  }
}
```

Michael Tin (SID: 862321571)

TickFct_S (Arduino 1)
Period: 100 ms



```
time = -1;
ICR1 = 0;
OCR1A = ICR1 / 2;
```

```
time++;
if (time > 623) {
time = 0;
}
ICR1 = musicArray[time];
OCR1A = ICR1 / 2;
```