Univerza *v Ljubljani*
Fakulteta *za računalništvo*
in informatiko

Poročilo o projektu tic-tac-toe
VHODNO IZHODNE NAPRAVE

Avtor: Nikolas Filip Žnidaršič

Profesor: viš. pred. dr. Robert Rozman

Contents

PARTS LIST:

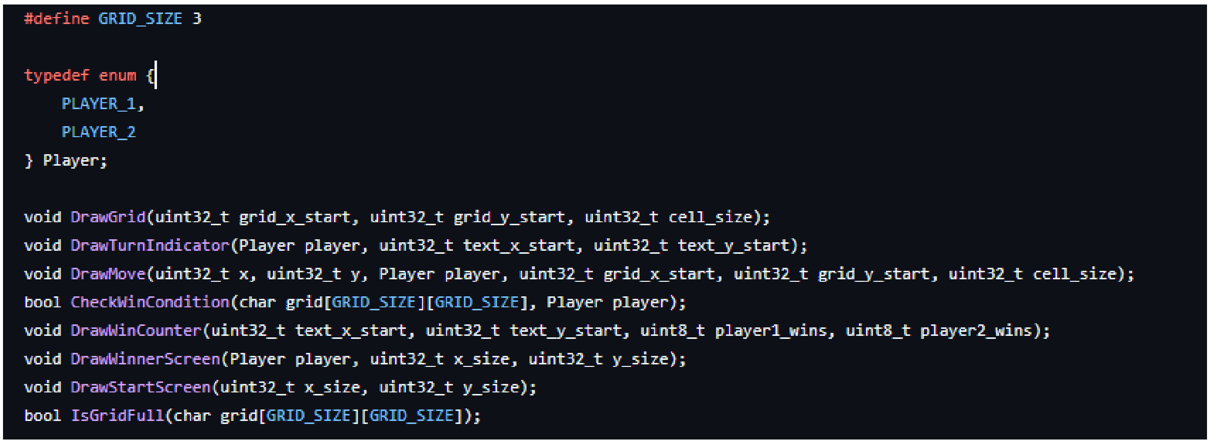
- STM32H7 discovery kit
- Laptop

Zasnova

Kot zasnovo za moj projekt sem vedel, da morm uproabljati touch zaslon za H7, za to sem najprej kloniral sledeči repozitorij od kolega Aljaža Jusa (https://github.com/AljazJus/DisplayTutorial_STM32H750B-DK/tree/main). Potem pa dodamo kodo v main.c. Delovanje repozitorija lahko preverimo z uporabo touch screen sample aplikacije, ki pride z repozitorijem.

Programska koda

Definicije v kodi



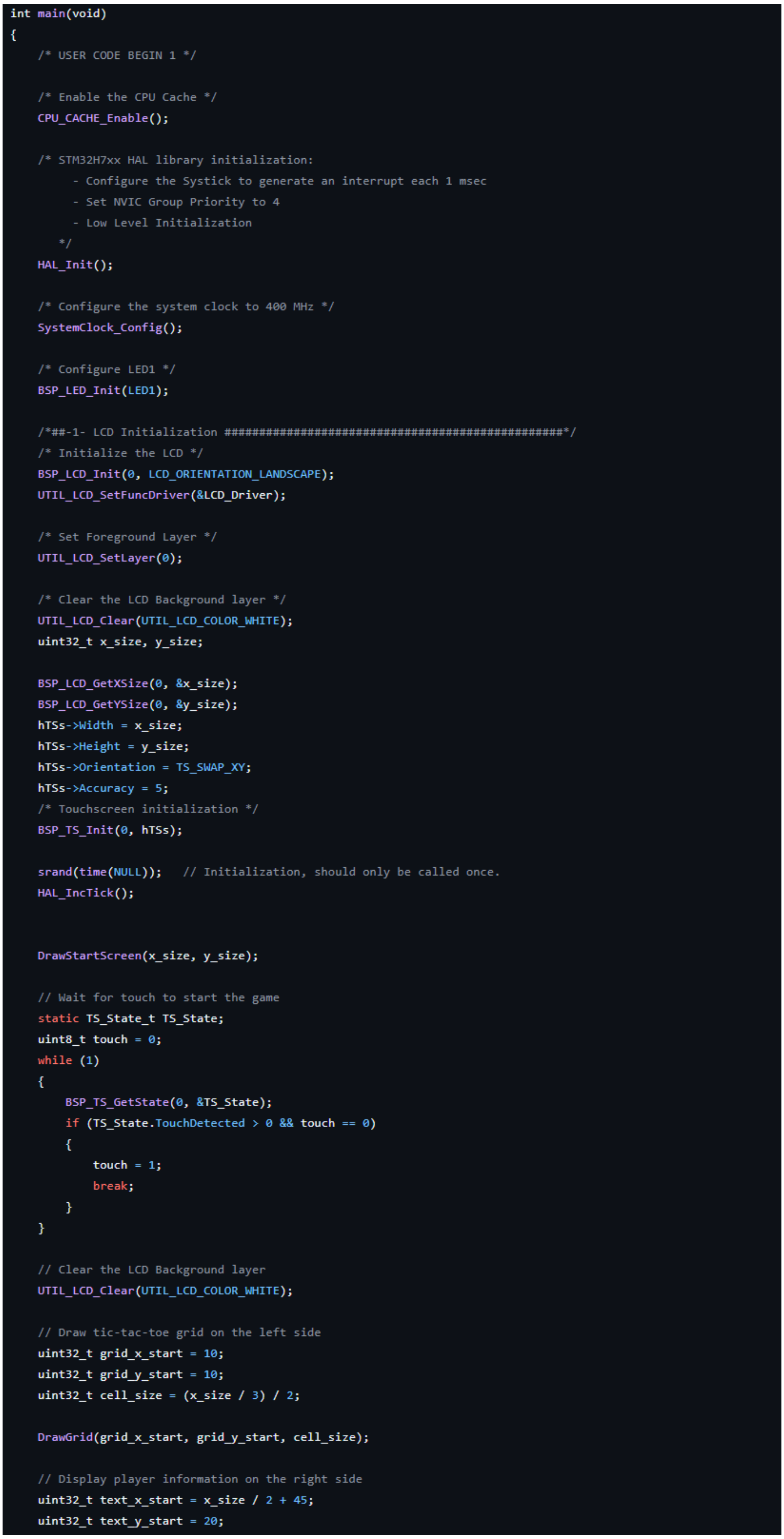
```
#define GRID_SIZE 3

typedef enum {
    PLAYER_1,
    PLAYER_2
} Player;

void DrawGrid(uint32_t grid_x_start, uint32_t grid_y_start, uint32_t cell_size);
void DrawTurnIndicator(Player player, uint32_t text_x_start, uint32_t text_y_start);
void DrawMove(uint32_t x, uint32_t y, Player player, uint32_t grid_x_start, uint32_t grid_y_start, uint32_t cell_size);
bool CheckWinCondition(char grid[GRID_SIZE][GRID_SIZE], Player player);
void DrawWinCounter(uint32_t text_x_start, uint32_t text_y_start, uint8_t player1_wins, uint8_t player2_wins);
void DrawWinnerScreen(Player player, uint32_t x_size, uint32_t y_size);
void DrawStartScreen(uint32_t x_size, uint32_t y_size);
bool IsGridFull(char grid[GRID_SIZE][GRID_SIZE]);
```

Kot je razvidno najprej definiramo grid_size da je 3x3 potem definiramo player enume, da lahko kasneje v igri ločimo med križci in krožci.

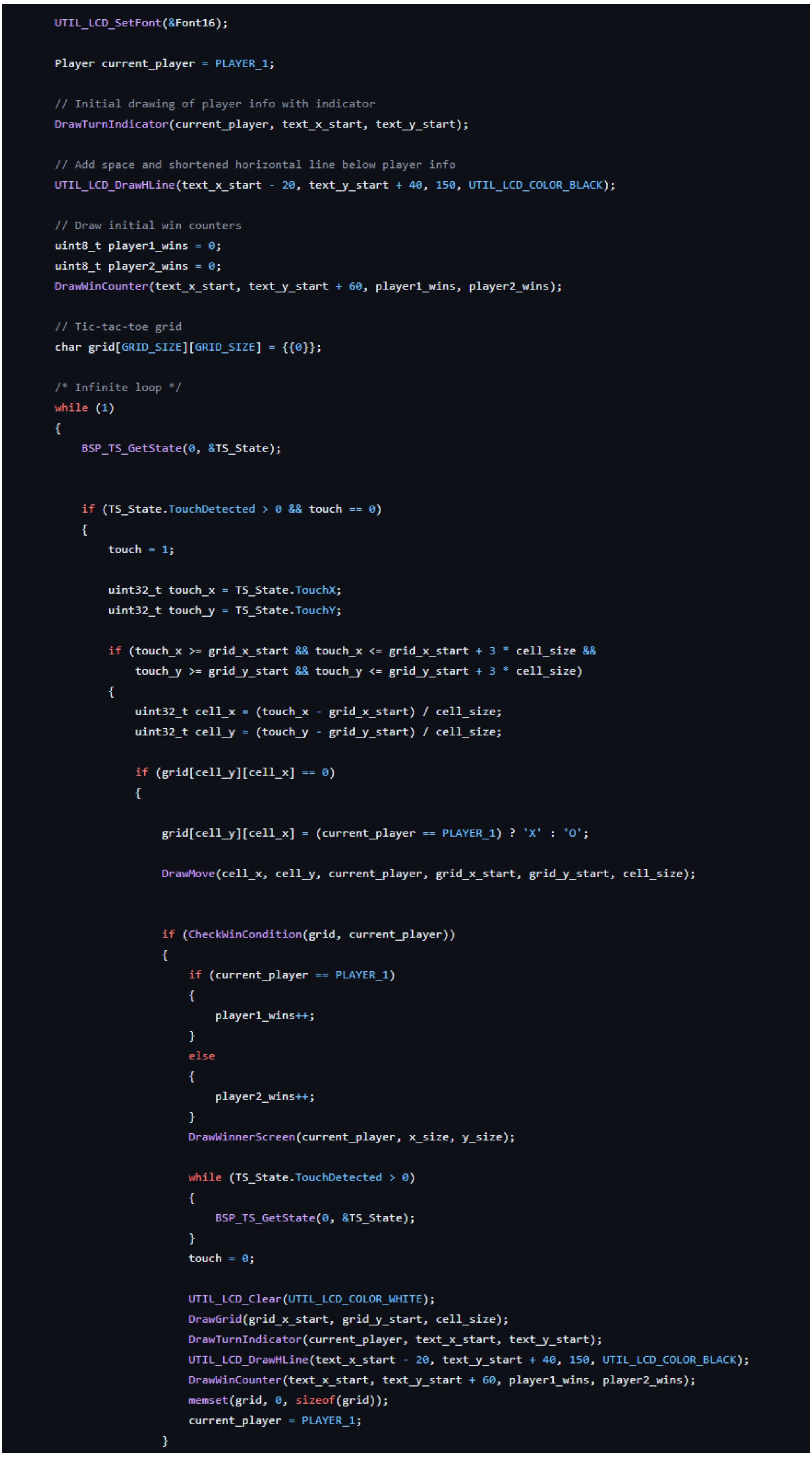
Glavne funkcije

- DrawGrid: Funkcija nariše mrežo za križce in krožce s pomočjo navpičnih in vodoravnih črt.
- DrawTurnIndicator: Prikaže, kateri igralec je na vrsti (X za igralca 1 ali O za igralca 2).
- DrawMove: Prikaže igralčev znak (X ali O) v ustrezni celici glede na dotik.
- CheckWinCondition: Preveri, ali je trenutni igralec zmagal z zasedbo vrstice, stolpca ali diagonale.
- IsGridFull: Preveri, ali je mreža popolnoma zapolnjena, kar bi pomenilo izenačenje.
- DrawWinCounter: Prikaže število zmag za vsakega igralca.
- DrawWinnerScreen: Prikaže zaslon z zmago trenutnega igralca ali zaslon za neodločen izid.
- DrawStartScreen: Prikaže začetni zaslon z navodilom za začetek igre.

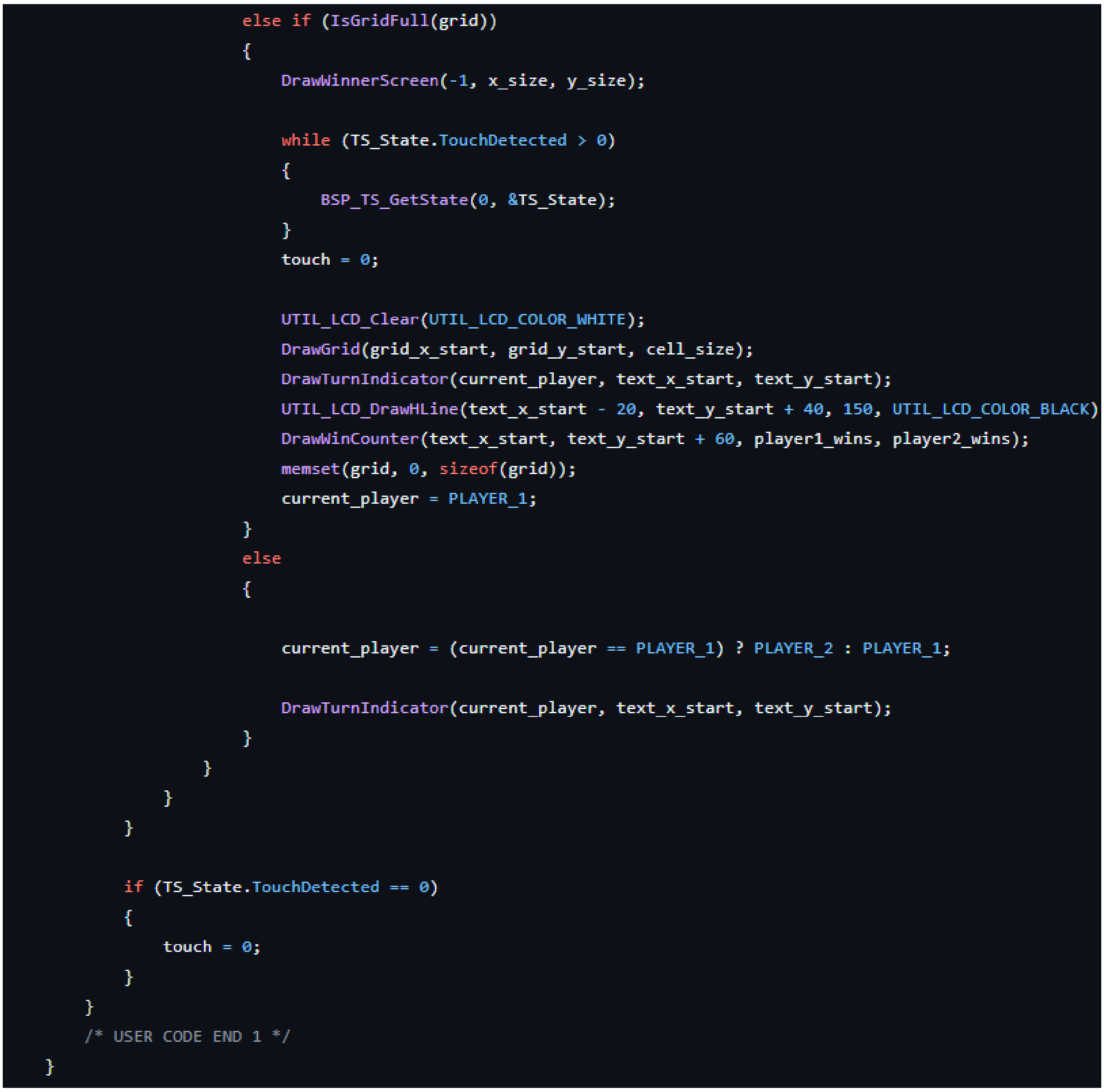
Glavna zanka igre



```c
int main(void)
{
    /* USER CODE BEGIN 1 */

    /* Enable the CPU Cache */
    CPU_CACHE_Enable();

    /* STM32H7xx HAL library initialization:
         - Configure the Systick to generate an interrupt each 1 msec
         - Set NVIC Group Priority to 4
         - Low Level Initialization
       */
    HAL_Init();

    /* Configure the system clock to 400 MHz */
    SystemClock_Config();

    /* Configure LED1 */
    BSP_LED_Init(LED1);

    /*##-1- LCD Initialization #################################################*/
    /* Initialize the LCD */
    BSP_LCD_Init(0, LCD_ORIENTATION_LANDSCAPE);
    UTIL_LCD_SetFuncDriver(&LCD_Driver);

    /* Set Foreground Layer */
    UTIL_LCD_SetLayer(0);

    /* Clear the LCD Background layer */
    UTIL_LCD_Clear(UTIL_LCD_COLOR_WHITE);
    uint32_t x_size, y_size;

    BSP_LCD_GetXSize(0, &x_size);
    BSP_LCD_GetYSize(0, &y_size);
    hTSs->Width = x_size;
    hTSs->Height = y_size;
    hTSs->Orientation = TS_SWAP_XY;
    hTSs->Accuracy = 5;
    /* Touchscreen initialization */
    BSP_TS_Init(0, hTSs);

    srand(time(NULL));   // Initialization, should only be called once.
    HAL_IncTick();


    DrawStartScreen(x_size, y_size);

    // Wait for touch to start the game
    static TS_State_t TS_State;
    uint8_t touch = 0;
    while (1)
    {
        BSP_TS_GetState(0, &TS_State);
        if (TS_State.TouchDetected > 0 && touch == 0)
        {
            touch = 1;
            break;
        }
    }

    // Clear the LCD Background layer
    UTIL_LCD_Clear(UTIL_LCD_COLOR_WHITE);

    // Draw tic-tac-toe grid on the left side
    uint32_t grid_x_start = 10;
    uint32_t grid_y_start = 10;
    uint32_t cell_size = (x_size / 3) / 2;

    DrawGrid(grid_x_start, grid_y_start, cell_size);

    // Display player information on the right side
    uint32_t text_x_start = x_size / 2 + 45;
    uint32_t text_y_start = 20;
```



```
    UTIL_LCD_SetFont(&Font16);

    Player current_player = PLAYER_1;

    // Initial drawing of player info with indicator
    DrawTurnIndicator(current_player, text_x_start, text_y_start);

    // Add space and shortened horizontal line below player info
    UTIL_LCD_DrawHLine(text_x_start - 20, text_y_start + 40, 150, UTIL_LCD_COLOR_BLACK);

    // Draw initial win counters
    uint8_t player1_wins = 0;
    uint8_t player2_wins = 0;
    DrawWinCounter(text_x_start, text_y_start + 60, player1_wins, player2_wins);

    // Tic-tac-toe grid
    char grid[GRID_SIZE][GRID_SIZE] = {{0}};

    /* Infinite loop */
    while (1)
    {
        BSP_TS_GetState(0, &TS_State);


        if (TS_State.TouchDetected > 0 && touch == 0)
        {
            touch = 1;

            uint32_t touch_x = TS_State.TouchX;
            uint32_t touch_y = TS_State.TouchY;

            if (touch_x >= grid_x_start && touch_x <= grid_x_start + 3 * cell_size &&
                touch_y >= grid_y_start && touch_y <= grid_y_start + 3 * cell_size)
            {
                uint32_t cell_x = (touch_x - grid_x_start) / cell_size;
                uint32_t cell_y = (touch_y - grid_y_start) / cell_size;

                if (grid[cell_y][cell_x] == 0)
                {

                    grid[cell_y][cell_x] = (current_player == PLAYER_1) ? 'X' : 'O';

                    DrawMove(cell_x, cell_y, current_player, grid_x_start, grid_y_start, cell_size);


                    if (CheckWinCondition(grid, current_player))
                    {
                        if (current_player == PLAYER_1)
                        {
                            player1_wins++;
                        }
                        else
                        {
                            player2_wins++;
                        }
                        DrawWinnerScreen(current_player, x_size, y_size);

                        while (TS_State.TouchDetected > 0)
                        {
                            BSP_TS_GetState(0, &TS_State);
                        }
                        touch = 0;

                        UTIL_LCD_Clear(UTIL_LCD_COLOR_WHITE);
                        DrawGrid(grid_x_start, grid_y_start, cell_size);
                        DrawTurnIndicator(current_player, text_x_start, text_y_start);
                        UTIL_LCD_DrawHLine(text_x_start - 20, text_y_start + 40, 150, UTIL_LCD_COLOR_BLACK);
                        DrawWinCounter(text_x_start, text_y_start + 60, player1_wins, player2_wins);
                        memset(grid, 0, sizeof(grid));
                        current_player = PLAYER_1;
                    }
```



```
                    else if (IsGridFull(grid))
                    {
                        DrawWinnerScreen(-1, x_size, y_size);

                        while (TS_State.TouchDetected > 0)
                        {
                            BSP_TS_GetState(0, &TS_State);
                        }
                        touch = 0;

                        UTIL_LCD_Clear(UTIL_LCD_COLOR_WHITE);
                        DrawGrid(grid_x_start, grid_y_start, cell_size);
                        DrawTurnIndicator(current_player, text_x_start, text_y_start);
                        UTIL_LCD_DrawHLine(text_x_start - 20, text_y_start + 40, 150, UTIL_LCD_COLOR_BLACK)
                        DrawWinCounter(text_x_start, text_y_start + 60, player1_wins, player2_wins);
                        memset(grid, 0, sizeof(grid));
                        current_player = PLAYER_1;
                    }
                    else
                    {

                        current_player = (current_player == PLAYER_1) ? PLAYER_2 : PLAYER_1;

                        DrawTurnIndicator(current_player, text_x_start, text_y_start);
                    }
                }
            }
        }

        if (TS_State.TouchDetected == 0)
        {
            touch = 0;
        }
    }
    /* USER CODE END 1 */
}
```

Glavna zanka igre v funkciji main se začne po inicializaciji sistemskih komponent, vključno z LCD zaslonom in sistemom za upravljanje na dotik. Med igro se prikaže mreža križcev in krožcev, kjer igralci izmenično opravljajo poteze. Po vsaki potezi se ustrezno prikaže simbol igralca na mreži, nato pa se preverijo pogoji za zmago ali izenačenje. Ko eden od igralcev zmaga ali pride do neodločenega izida, se prikaže zaslon s končnim rezultatom. Po tem se igra ponastavi in lahko začne znova.

Potek igre

Igra križcev in krožcev se začne z začetnim zaslonom, ki čaka na dotik za začetek igre. Igralca izmenično izbirata prazne celice na mreži, kjer eden postavlja križce (X), drugi pa krožce (O). Po vsaki potezi se preveri, ali je kateri igralec zasedel celotno vrstico, stolpec ali diagonalo, kar predstavlja pogoje za zmago. Če kateri igralec izpolni te pogoje, igra prikaže zaslon z razglasitvijo zmagovalca. Če je mreža zapolnjena in nihče ni zmagal, se igra konča z neodločenim izidom.