lab4_UART_GAME

Generated by Doxygen 1.8.13

Contents

Chapter 1

README

In this laboratory work, a program was developed in the form of the game "Guess a Number". The UART interface is used to transfer data between the program and the board. To communicate with the program, the terminal emulator on the UART PC is used on the chipKIT Wi-FIRE board. It is necessary to guess the number in the range from 0 to 10, which each time is defined by the function rand(). Only 3 attempts are allowed. For each attempt, the program will respond: YES (if the number is guessed right), too high (if the number is greater), too low (if the number is less). If the number was correct, the winning count is increased by 1. If the number is guessed incorrectly or not guessed 3 times, the counter of the losses is increased by 1. If the win count reaches 5, a message is displayed that the jackpot has been won. If the counter of the losses has reached the value 5, messages are displayed that the jackpot is not won. When you enter 11, the win and loss counters are reset to zero.

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

File Documentation

3.1 D:/GIT/TheConnectedMCU_Labs/bkarachok/lab4_UART_GAME/configuration_bits.c File Reference

3.2 D:/GIT/TheConnectedMCU_Labs/bkarachok/lab4_UART_GAME/game.c File Reference

```
#include "game.h"
#include "UART.h"
```

Functions

- int Get_Guess (void)
- void GuessingGame (void)

3.2.1 Function Documentation

3.2.1.1 Get_Guess()

```
int Get_Guess (
              void  )

// processing of received data from the user
    char c;
    int n=0, done=0;

    while (1) {
        c = UART4_getc();
        UART4_putc(c); // echo back character
        if ((c >= '0') && (c <= '9')) {
            n *= 10;
            n += c-'0';
        } else {
            if (c == '\r')
                return n;
            else
                return ILLEGAL_CHAR;
        }
    }
```

3.2.1.2 GuessingGame()

```
void GuessingGame (
              void  )
```

```
int guess, number, guesses_left, won=0, num_won=0, num_lost=0, reset_flag=0;
char buffer[80];

while (1) { // new game
    UART4_puts("\r\nGuessed integer number in the range from 0 to 10.\r\n");
    number = rand() % 10; // random value of number from 0 to 10;
    won = 0;
    for (guesses_left=NUM_GUESSES; (guesses_left>0) && (!won); guesses_left--) {
        sprintf(buffer, "You have %d guesses left.\r\n", guesses_left);
        UART4_puts(buffer);
        UART4_puts("What is this number?? Please type it and press enter.\r\n");
        UART4_puts("If you want to reset counters of wins and losses, type 11.\r\n");
        guess = Get_Guess();
        UART4_puts("\r\n");
        if (guess == ILLEGAL_CHAR) { //cheking correct data
            UART4_puts("That's not a number! You lost your guess.\r\n");
        }
        else { //checking number which user entered
            if (guess == number) { // user guessed number
                UART4_puts("Congratulations! You guessed it!\r\n");
                won = 1;
            }
            else if (guess < number) { // user didnt guess number
                UART4_puts("Too low!\r\n");
            }
            else { // user didnt guess number
                if (guess == 11) //Reset counters of wins and losses
                {
                   reset_flag = 1;
                   guesses_left = 0;
                }
                else UART4_puts("Too high!\r\n");
                }
            }
    }
    if (!reset_flag)
    {
         if (won)  //if user did guess number
             {
              num_won++;
             }
         if (!won) // if user didnt guess number
            {
            sprintf(buffer, "My number was %d.\r\n", number);
            UART4_puts(buffer);
            num_lost++;
            }
        if (num_won == 5) //if user have 5 wins in result gradual guessing
            {
            sprintf(buffer, "[Congratulations, you get jackpot!!!!]:\r\nwon number = %d\r\nlost number = %d
      \r\n", num_won, num_lost);
            UART4_puts(buffer);
            num_won = 0;
            num_lost = 0;
            }
        if (num_lost == 5) //if user have 5 loses in result gradual guessing
            {
            sprintf(buffer, "[Sorry, but you don't get jackpot]:\r\nwon number = %d\r\nlost number = %d\r\n
      ", num_won, num_lost);
            UART4_puts(buffer);
            num_lost = 0;
            num_won = 0;
            }
    }
    if (reset_flag)
    {
         num_lost = 0;
         num_won = 0;
         UART4_puts("Counters of wins and losses were reset!\r\n");
         reset_flag = 0;

    }
     // common quantity of wins and loses
     sprintf(buffer, "Won: %d\r\nLost: %d\r\n", num_won, num_lost);
     UART4_puts(buffer);
}
```

3.3 D:/GIT/TheConnectedMCU_Labs/bkarachok/lab4_UART_GAME/game.h File Reference

```
#include <xc.h>
#include "UART.h"
```

Macros

- #define NUM_GUESSES (3)
- #define TIME_OUT (-1)
- #define ILLEGAL_CHAR (-2)

Functions

- void GuessingGame (void)

3.3.1 Macro Definition Documentation

3.3.1.1 ILLEGAL_CHAR

```
#define ILLEGAL_CHAR (-2)
```

3.3.1.2 NUM_GUESSES

```
#define NUM_GUESSES (3)
```

3.3.1.3 TIME_OUT

```
#define TIME_OUT (-1)
```

3.3.2 Function Documentation

3.3.2.1 GuessingGame()

```
void GuessingGame (
              void  )
```

```
int guess, number, guesses_left, won=0, num_won=0, num_lost=0, reset_flag=0;
char buffer[80];

while (1) { // new game
    UART4_puts("\r\nGuessed integer number in the range from 0 to 10.\r\n");
    number = rand() % 10; // random value of number from 0 to 10;
    won = 0;
    for (guesses_left=NUM_GUESSES; (guesses_left>0) && (!won); guesses_left--) {
        sprintf(buffer, "You have %d guesses left.\r\n", guesses_left);
        UART4_puts(buffer);
        UART4_puts("What is this number?? Please type it and press enter.\r\n");
        UART4_puts("If you want to reset counters of wins and losses, type 11.\r\n");
        guess = Get_Guess();
        UART4_puts("\r\n");
        if (guess == ILLEGAL_CHAR) { //cheking correct data
            UART4_puts("That's not a number! You lost your guess.\r\n");
        }
        else { //checking number which user entered
            if (guess == number) { // user guessed number
                UART4_puts("Congratulations! You guessed it!\r\n");
                won = 1;
            }
            else if (guess < number) { // user didnt guess number
                UART4_puts("Too low!\r\n");
            }
            else { // user didnt guess number
                if (guess == 11) //Reset counters of wins and losses
                {
                   reset_flag = 1;
                   guesses_left = 0;
                }
                else UART4_puts("Too high!\r\n");
                }
            }
    }
    if (!reset_flag)
    {
         if (won)  //if user did guess number
             {
              num_won++;
             }
         if (!won) // if user didnt guess number
            {
            sprintf(buffer, "My number was %d.\r\n", number);
            UART4_puts(buffer);
            num_lost++;
            }
        if (num_won == 5) //if user have 5 wins in result gradual guessing
            {
            sprintf(buffer, "[Congratulations, you get jackpot!!!!]:\r\nwon number = %d\r\nlost number = %d
      \r\n", num_won, num_lost);
            UART4_puts(buffer);
            num_won = 0;
            num_lost = 0;
            }
        if (num_lost == 5) //if user have 5 loses in result gradual guessing
            {
            sprintf(buffer, "[Sorry, but you don't get jackpot]:\r\nwon number = %d\r\nlost number = %d\r\n
      ", num_won, num_lost);
            UART4_puts(buffer);
            num_lost = 0;
            num_won = 0;
            }
    }
    if (reset_flag)
    {
         num_lost = 0;
         num_won = 0;
         UART4_puts("Counters of wins and losses were reset!\r\n");
         reset_flag = 0;

    }
     // common quantity of wins and loses
     sprintf(buffer, "Won: %d\r\nLost: %d\r\n", num_won, num_lost);
     UART4_puts(buffer);
}
```

3.4 D:/GIT/TheConnectedMCU_Labs/bkarachok/lab4_UART_GAME/main.c File Reference

```
#include <stdint.h>
#include <stdbool.h>
#include "user.h"
```

Functions

- int32_t main (void)

3.4.1 Function Documentation

3.4.1.1 main()

```
int32_t main (
            void  )

// Initialize I/O and Peripherals for application
InitApp();

GuessingGame();
```

3.5 D:/GIT/TheConnectedMCU_Labs/bkarachok/lab4_UART_GAME/README.md File Reference

3.6 D:/GIT/TheConnectedMCU_Labs/bkarachok/lab4_UART_GAME/UART.c File Reference

```
#include "UART.h"
```

Functions

- void UART4_init (void)
- char UART4_getc (void)
- void UART4_putc (char c)
- void UART4_puts (char *s)
- void UART4_test (void)

3.6.1 Function Documentation

3.6.1.1 UART4_getc()

```
char UART4_getc (
              void  )
```

```
while (!U4STAbits.URXDA)
    ; // wait until character received
return U4RXREG; // read character
```

3.6.1.2 UART4_init()

```
void UART4_init (
              void  )
```

```
RPF8R = 2; // PPS for U4RX from pin F2
U4RXR = 11; // PPS for U4TX to pin F8

U4STAbits.UTXEN = 1;   // enable transmit pin
U4STAbits.URXEN = 1;   // enable receive pin
U4BRG           = ((100 * 1000000) / (16 * 115200)) - 1;
U4MODEbits.ON   = 1;   // enable UART
```

3.6.1.3 UART4_putc()

```
void UART4_putc (
              char c )
```

```
while (U4STAbits.UTXBF)
    ;  // wait until transmit buffer empty
U4TXREG = c;  // transmit character
```

3.6.1.4 UART4_puts()

```
void UART4_puts (
              char * s )
```

```
while (*s != '\0')
    UART4_putc (*s++);
```

3.6.1.5 UART4_test()

```
void UART4_test (
              void  )
```

```
char c;

UART4_puts("Hello, World!\r\n");
UART4_puts("Please press a key.\r\n");

for (;;)
{
    c = UART4_getc();
    UART4_putc('[');
    UART4_putc (c);
    UART4_puts("]\r\n");
}
```

3.7 D:/GIT/TheConnectedMCU_Labs/bkarachok/lab4_UART_GAME/UART.h File Reference

```
#include <xc.h>
```

Functions

- void UART4_init (void)
- char UART4_getc (void)
- void UART4_putc (char c)
- void UART4_puts (char *s)
- void UART4_test (void)

3.7.1 Function Documentation

3.7.1.1 UART4_getc()

```
char UART4_getc (
              void  )
```

```
while (!U4STAbits.URXDA)
    ; // wait until character received
return U4RXREG; // read character
```

3.7.1.2 UART4_init()

```
void UART4_init (
              void  )
```

```
RPF8R = 2; // PPS for U4RX from pin F2
U4RXR = 11; // PPS for U4TX to pin F8

U4STAbits.UTXEN = 1;   // enable transmit pin
U4STAbits.URXEN = 1;   // enable receive pin
U4BRG           = ((100 * 1000000) / (16 * 115200)) - 1;
U4MODEbits.ON   = 1;   // enable UART
```

3.7.1.3 UART4_putc()

```
void UART4_putc (
              char c )
```

```
while (U4STAbits.UTXBF)
    ;  // wait until transmit buffer empty
U4TXREG = c;  // transmit character
```

3.7.1.4 UART4_puts()

```
void UART4_puts (
              char * s )
```

```
while (*s != '\0')
    UART4_putc (*s++);
```

3.7.1.5 UART4_test()

```
void UART4_test (
              void  )
```

```
char c;

UART4_puts("Hello, World!\r\n");
UART4_puts("Please press a key.\r\n");

for (;;)
{
    c = UART4_getc();
    UART4_putc('[');
    UART4_putc (c);
    UART4_puts("]\r\n");
}
```

3.8 D:/GIT/TheConnectedMCU_Labs/bkarachok/lab4_UART_GAME/user.c File Reference

```
#include <stdint.h>
#include <stdbool.h>
#include "user.h"
#include <sys/attribs.h>
```

Functions

- void InitGPIO (void)
- void InitApp (void)

3.8.1 Function Documentation

3.8.1.1 InitApp()

```
void InitApp (
             void  )
```

```
// Initialize peripherals
InitGPIO();
UART4_init();
```

3.8.1.2 InitGPIO()

```
void InitGPIO (
             void  )
```

```
   // Setup functionality and port direction
   // LED output
// Disable analog mode if present
   ANSELG &= ~((1 << 6) | (1 << 15));
   ANSELB &= ~(1 << 11);
   // Set direction to output
   TRISG &= ~((1 << 6) | (1 << 15));
   TRISB &= ~(1 << 11);
   TRISD &= ~(1 << 4);
   // Turn off LEDs for initialization
   LD1_PORT_BIT = 0;
   LD2_PORT_BIT = 0;
   LD3_PORT_BIT = 0;
   LD4_PORT_BIT = 0;
   // Button inputs
   // Disable analog mode
   ANSELA &= ~(1 << 5);
   // Set directions to input
   TRISA |= (1 << 5)|(1 << 4);

   TRISDCLR = 1<<11;
```

3.9 D:/GIT/TheConnectedMCU_Labs/bkarachok/lab4_UART_GAME/user.h File Reference

```
#include <stdint.h>
```

Macros

- #define LD1_PORT_BIT LATGbits.LATG6
- #define LD2_PORT_BIT LATDbits.LATD4
- #define LD3_PORT_BIT LATBbits.LATB11
- #define LD4_PORT_BIT LATGbits.LATG15
- #define BTN1_PORT_BIT PORTAbits.RA5
- #define BTN2_PORT_BIT PORTAbits.RA4

Functions

- void InitApp (void)
- void DelayMs (int t)

3.9.1 Macro Definition Documentation

3.9.1.1 BTN1_PORT_BIT

```
#define BTN1_PORT_BIT PORTAbits.RA5
```

3.9.1.2 BTN2_PORT_BIT

```
#define BTN2_PORT_BIT PORTAbits.RA4
```

3.9.1.3 LD1_PORT_BIT

```
#define LD1_PORT_BIT LATGbits.LATG6
```

3.9.1.4 LD2_PORT_BIT

```
#define LD2_PORT_BIT LATDbits.LATD4
```

3.9.1.5 LD3_PORT_BIT

```
#define LD3_PORT_BIT LATBbits.LATB11
```

3.9.1.6 LD4_PORT_BIT

```
#define LD4_PORT_BIT LATGbits.LATG15
```

3.9.2 Function Documentation

3.9.2.1 DelayMs()

```
void DelayMs (
            int t )
```

3.9.2.2 InitApp()

```
void InitApp (
            void  )
```

```
// Initialize peripherals
InitGPIO();
UART4_init();
```

Index