Tetris_STM_code

Generated by Doxygen 1.10.0

Chapter 1

Tetris Game

1.1 Introduction

Ce jeu de Tetris a entièrement été réalisé par Adrien Kerfriden, Lou Vacher et Pierre Pontet.

Il a été conçu pour fonctionner sur une carte STM32 L031K6.

Il suffit de brancher 5 boutons, sur les broches PA1, PA3, PA4, PA6 et PA7.

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Class Documentation

4.1 TETRIMINO Struct Reference

Structure to locate the current tetrimino in the game.

Public Attributes

- int **coordY**
- int **coordX**
- int **piece** [4][4]

4.1.1 Detailed Description

Structure to locate the current tetrimino in the game.

The documentation for this struct was generated from the following file:

- C:/Users/Pierre/Documents/Git_Repos/Tetris-STM/Tetris_STM_code/tetrisGame/tetrisGame.c

Chapter 5

File Documentation

5.1 C:/Users/Pierre/Documents/Git_Repos/Tetris-STM/Tetris_STM_↩ code/tetrisGame/tetrisGame.c File Reference

: Main program of the Tetris Game

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <tim.h>
#include "tetrisScoring.h"
#include "userOled.h"
```

Classes

- struct TETRIMINO

  Structure to locate the current tetrimino in the game.

Macros

- #define **PIECE_TOUCHED** 2
- #define **PIECE_FALLED** 1
- #define **FALL_COOLDOWN** 0

Enumerations

- enum game_state {
  PLAY_STATE = 0 , **SCORE_STATE** = 1 , **SPAWN_STATE** = 2 , **GAMEOVER_STATE** = 3 ,
  START_STATE = 4 }

  Game States.
- enum inputs {
  gauche = -1 , **droite** = 1 , **fall** = 2 , **rotate** = 3 ,
  menu = 4 }

  Inputs du jeu, permet de simplifier le code.

Functions

- void addPieceIntoPlayZone (TETRIMINO *currentTetrimino)

  Permet de placer la pièce dans le jeu.
- void removePieceFromPlayZone (TETRIMINO *currentTetrimino)

  Permet de supprimer la pièce du jeu.
- int isClippingInPlayZone (TETRIMINO *currentTetrimino)

  Permet de détecter si la pièce rentrerait en collision avec les blocs déjà présents.
- int falling (TETRIMINO *currentTetrimino)

  Permet de gérer la chute de la pièce.
- void **printPlayZone** (void)

  Affiche la play zone sur le terminal.
- int movePiece (TETRIMINO *currentTetrimino, int direction)

  Permet de déplacer une pièce vers la gauche ou la droite.
- int rotatePiece (TETRIMINO *currentTetrimino)

  Permet de pivoter de 90° une pièce, seulement si elle ne va pas clip.
- int getLinesCompleted ()

  Calculate amount of lines completed, and shifts down the playZone accordingly.
- void removeLine (int line)

  Remove the selected line (fill it with 0)
- void **resetPlayZone** ()

  Reset the PlayZone.
- void moveDownPlayZone (int startLine)

  Move all the play zone above "startLine" down by one line.
- void randomPiece (TETRIMINO *currentTetrimino)

  Select a random piece among the tetriminos.
- void **tetrisInit** ()

  Permet d'initialiser le jeu.
- int tetrisGame ()

  Permet de lancer le jeu.
- void **reduceFallDelay** ()

  Reduce the delay of fallTime.
- void HAL_GPIO_EXTI_Callback (uint16_t GPIO_Pin)

  Gestion des interruptions.
- void HAL_TIM_PeriodElapsedCallback (TIM_HandleTypeDef *htim)

  Fin du timer pour lire les boutons.

Variables

- int pieceZigzag [4][4]

  Zigzag piece in 4 by 4 grid.
- int pieceS [4][4]

  S piece in 4 by 4 grid.
- int pieceBar [4][4]

  Bar piece in 4 by 4 grid.
- int pieceSquare [4][4]

  Square piece in 4 by 4 grid.
- int pieceL [4][4]

  L piece in 4 by 4 grid.
- int pieceJ [4][4]

J piece in 4 by 4 grid.

- int pieceT [4][4]

  T piece in 4 by 4 grid.

- int playZone [23][16]

  PlayZone.

- uint32_t **nextFallTime** = 0
- int **fallDelay** = 1000
- int **rotateFallDelay** = 500
- int **moveFallDelay** = 500
- int **userInput** = 0

5.1.1 Detailed Description

: Main program of the Tetris Game

Date

08 février 2024

5.1.2 Function Documentation

5.1.2.1 addPieceIntoPlayZone()

```
void addPieceIntoPlayZone (
            TETRIMINO * currentTetrimino )
```

Permet de placer la pièce dans le jeu.

Parameters

∗currentTetrimino	Current tetrimino used

5.1.2.2 falling()

```
int falling (
            TETRIMINO * currentTetrimino )
```

Permet de gérer la chute de la pièce.

Parameters

∗currentTetrimino	Current tetrimino used

Return values

state	PIECE_TOUCHED, or PIECE_FALLED

5.1.2.3 getLinesCompleted()

```
int getLinesCompleted ( )
```

Calculate amount of lines completed, and shifts down the playZone accordingly.

Return values

linesCount	number of Lines completed

5.1.2.4 HAL_GPIO_EXTI_Callback()

```
void HAL_GPIO_EXTI_Callback (
            uint16_t GPIO_Pin )
```

Gestion des interruptions.

Parameters

GPIO_Pin	Pin sur lequel a lieu l'interruption

5.1.2.5 HAL_TIM_PeriodElapsedCallback()

```
void HAL_TIM_PeriodElapsedCallback (
            TIM_HandleTypeDef * htim )
```

Fin du timer pour lire les boutons.

Parameters

∗*htim*	timer

5.1.2.6 isClippingInPlayZone()

```
int isClippingInPlayZone (
            TETRIMINO * currentTetrimino )
```

Permet de détecter si la pièce rentrerait en collision avec les blocs déjà présents.

Parameters

∗*currentTetrimino*	Current tetrimino used

Return values

state	1 if would clip, 0 if not

5.1.2.7 moveDownPlayZone()

```
void moveDownPlayZone (
            int startLine )
```

Move all the play zone above "startLine" down by one line.

Parameters

startLine	line which got cleared

5.1.2.8 movePiece()

```
int movePiece (
            TETRIMINO * currentTetrimino,
            int direction )
```

Permet de déplacer une pièce vers la gauche ou la droite.

Parameters

∗currentTetrimino	Current tetrimino used
direction	userInputs

Return values

state	1 si déplacement OK, 0 si non

5.1.2.9 randomPiece()

```
void randomPiece (
            TETRIMINO * currentTetrimino )
```

Select a random piece among the tetriminos.

Parameters

currentTetrimino	list of tetriminos

5.1.2.10 removeLine()

```
void removeLine (
            int line )
```

Remove the selected line (fill it with 0)

Parameters

line	line selected

5.1.2.11 removePieceFromPlayZone()

```
void removePieceFromPlayZone (
            TETRIMINO * currentTetrimino )
```

Permet de supprimer la pièce du jeu.

Parameters

∗*currentTetrimino*	Current tetrimino used

5.1.2.12 rotatePiece()

```
int rotatePiece (
            TETRIMINO * currentTetrimino )
```

Permet de pivoter de 90° une pièce, seulement si elle ne va pas clip.

Parameters

∗*currentTetrimino*	Current tetrimino used

Return values

state	1 if rotated and 0 if clipping

5.1.2.13 tetrisGame()

```
int tetrisGame ( )
```

Permet de lancer le jeu.

Return values

0	si erreur

5.1.3 Variable Documentation

5.1.3.1 pieceBar

```
int pieceBar[4][4]
```

Initial value:

```
= {   {0,1,0,0},
                        {0,1,0,0},
                        {0,1,0,0},
                        {0,1,0,0}}
```

Bar piece in 4 by 4 grid.

5.1.3.2 pieceJ

```
int pieceJ[4][4]
```

Initial value:

```
= { {0,0,0,0},
                    {0,0,1,0},
                    {0,0,1,0},
                    {0,1,1,0}}
```

J piece in 4 by 4 grid.

5.1.3.3 pieceL

```
int pieceL[4][4]
```

Initial value:

```
= { {0,0,0,0},
                    {0,1,0,0},
                    {0,1,0,0},
                    {0,1,1,0}}
```

L piece in 4 by 4 grid.

5.1.3.4 pieceS

```
int pieceS[4][4]
```

Initial value:

```
=   {{0,0,0,0},
                    {0,0,1,1},
                    {0,1,1,0},
                    {0,0,0,0}}
```

S piece in 4 by 4 grid.

5.1.3.5 pieceSquare

```
int pieceSquare[4][4]
```

Initial value:

```
= {{0,0,0,0},
                        {0,1,1,0},
                        {0,1,1,0},
                        {0,0,0,0}}
```

Square piece in 4 by 4 grid.

5.1.3.6 pieceT

```
int pieceT[4][4]
```

Initial value:

```
= { {0,0,0,0},
                    {1,1,1,0},
                    {0,1,0,0},
                    {0,0,0,0}}
```

T piece in 4 by 4 grid.

5.1.3.7 pieceZigzag

```
int pieceZigzag[4][4]
```

Initial value:

```
= {{0,0,0,0},
                        {1,1,0,0},
                        {0,1,1,0},
                        {0,0,0,0}}
```

Zigzag piece in 4 by 4 grid.

5.1.3.8 playZone

```
int playZone[23][16]
```

Initial value:

```
=
    {{1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,1},
    {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1},
    {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1},
    {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1}}
```

PlayZone.

5.2 tetrisGame.h

```
/*
 * tetrisGame.h
 *
 *  Created on: Feb 7, 2024
 *      Author: PONTET
 */

#ifndef TETRISGAME_H_
#define TETRISGAME_H_

int tetrisGame();
void reduceFallDelay();

#endif /* TETRISGAME_H_ */
```

5.3 C:/Users/Pierre/Documents/Git_Repos/Tetris-STM/Tetris_STM_↩ code/tetrisGame/tetrisLeveling.c File Reference

: Program for Tetris Leveling system

```
#include "tetrisLeveling.h"
#include "tetrisGame.h"
```

Functions

- uint8_t getLevelNumber ()
  Return levelNumber.
- void newLevelCalculate (uint8_t _numberOfLineCompleted)
  Verify and change levelNumber.

Variables

- uint8_t **levelNumber** = 0

5.3.1 Detailed Description

: Program for Tetris Leveling system

Date

08 février 2024

5.3.2 Function Documentation

5.3.2.1 getLevelNumber()

```
uint8_t getLevelNumber ( )
```

Return levelNumber.

Return values

levelNumber	

5.3.2.2 newLevelCalculate()

```
void newLevelCalculate (
            uint8_t _numberOfLineCompleted )
```

Verify and change levelNumber.

Parameters

_numberOfLineCompleted	each level requires a numberOfLine completed to be incremented

5.4 tetrisLeveling.h

```
00001 /*
00002  * tetrisLeveling.h
00003  *
00004  *  Created on: Feb 7, 2024
00005  *      Author: PONTET
00006  */
00007
00008 #ifndef TETRISLEVELING_H_
00009 #define TETRISLEVELING_H_
00010
00011 #include "main.h"
00012
00013 uint8_t getLevelNumber();
00014 void newLevelCalculate(uint8_t);
00015
00016 #endif /* TETRISLEVELING_H_ */
```

5.5 C:/Users/Pierre/Documents/Git_Repos/Tetris-STM/Tetris_STM_↩ code/tetrisGame/tetrisScoring.c File Reference

: Program for Tetris Scoring system

```
#include "tetrisScoring.h"
#include "ssd1306_fonts.h"
#include "ssd1306.h"
#include <stdio.h>
#include "string.h"
#include "tetrisLeveling.h"
#include "userOled.h"
```

Functions

- uint32_t getScore ()

  Getter of score.
- void calculateScore (int _numberOfLineCompleted)

  Calculate the Score.
- void **addScorePiece** ()

  Add points to the score when a Piece fall to its place.
- void addScoreLine (uint8_t _numberOfLineCompleted)

  Add points to the score, using numberOfLineCompleted and levelNumber as factors.
- void **printScore** ()

  Print the score on the screen.

Variables

- uint32_t **tetrisScore** = 0
- uint8_t **piecePosee** = 100

5.5.1 Detailed Description

: Program for Tetris Scoring system

Date

08 février 2024

5.5.2 Function Documentation

5.5.2.1 addScoreLine()

```
void addScoreLine (
            uint8_t _numberOfLineCompleted )
```

Add points to the score, using numberOfLineCompleted and levelNumber as factors.

Parameters

_numberOfLineCompleted	numberOfLineCompleted in one fall by a piece

5.5.2.2 calculateScore()

```
void calculateScore (
            int _numberOfLineCompleted )
```

Calculate the Score.

Parameters

_numberOfLineCompleted	

5.5.2.3 getScore()

```
uint32_t getScore ( )
```

Getter of score.

Return values

Return	the value of score

5.6 tetrisScoring.h

```
00001 /*
00002  * tetrisScoring.h
```

```
00003  *
00004  *  Created on: Feb 1, 2024
00005  *      Author: PONTET
00006  */
00007
00008 #ifndef TETRISSCORING_H_
00009 #define TETRISSCORING_H_
00010
00011 #include "main.h"
00012
00013
00014 uint32_t getScore();
00015 void calculateScore(int);
00016 void addScorePiece();
00017 void addScoreLine(uint8_t);
00018
00019 void printScore();
00020
00021 #endif /* TETRISSCORING_H_ */
```

5.7 C:/Users/Pierre/Documents/Git_Repos/Tetris-STM/Tetris_STM_↩ code/tetrisGame/userOled.c File Reference

: Others functions for Oled drawing

```
#include "userOled.h"
#include "tetrisScoring.h"
```

Functions

- void **drawBorder** ()

  Draw the border of the game.
- void drawTile (uint8_t x, uint8_t y, SSD1306_COLOR color)

  Draw a normal tile.
- void drawPlayZone (int stack[23][16])

  Draw the entire play zone.
- void **drawGameOver** ()

  Plays the gameOver Animation.
- void **drawTetrisStartGame** ()

  Plays the StartMenu Animation.
- void **clearPlayZone** ()

  Clear the PlayZone.
- char ssd1306_WriteCharVertical (char ch, FontDef Font, SSD1306_COLOR color)

  Ecrit un caractère verticalement.
- char ssd1306_WriteStringVertical (char ∗str, FontDef Font, SSD1306_COLOR color)

  Ecrit une chaîne de caractères verticalement.
- void ssd1306_SetCursorVertical (uint8_t x, uint8_t y)

  Déplace le curseur à sa position Verticale.
- void drawBorderDecor (SSD1306_COLOR color)

  Draw the decor.
- void **drawTetriminos** ()

  Affiche les tetriminos en bas du StartMenu.
- void **Clignotement_Ecran** ()

  Fait "clignoter" l'écran en inversant ses couleurs pendant un court instant.
- void **Clignotement_Click** ()

  Fait "clignoter" le mot "Click" à l'écran de démarrage, non bloquant.

5.7.1 Detailed Description

: Others functions for Oled drawing

Date

08 février 2024

5.7.2 Function Documentation

5.7.2.1 drawBorderDecor()

```
void drawBorderDecor (
            SSD1306_COLOR color )
```

Draw the decor.

Parameters

color	couleur sélectionnée

5.7.2.2 drawPlayZone()

```
void drawPlayZone (
            int stack[23][16] )
```

Draw the entire play zone.

Parameters

stack[][]	the play zone of the game

5.7.2.3 drawTile()

```
void drawTile (
            uint8_t x,
            uint8_t y,
            SSD1306_COLOR color )
```

Draw a normal tile.

Parameters

x,y	position de la Tile à changer
color	couleur de la Tile

5.7.2.4 ssd1306_SetCursorVertical()

```
void ssd1306_SetCursorVertical (
            uint8_t x,
            uint8_t y )
```

Déplace le curseur à sa position Verticale.

Parameters

x,y	: position du curseur à changer

5.7.2.5 ssd1306_WriteCharVertical()

```
char ssd1306_WriteCharVertical (
            char ch,
            FontDef Font,
            SSD1306_COLOR color )
```

Ecrit un caractère verticalement.

Parameters

ch	caractère à afficher
Font	police sélectionnée
color	couleur sélectionnée

5.7.2.6 ssd1306_WriteStringVertical()

```
char ssd1306_WriteStringVertical (
            char * str,
            FontDef Font,
            SSD1306_COLOR color )
```

Ecrit une chaîne de caractères verticalement.

Parameters

str	chaîne de caractères à afficher
Font	police sélectionnée
color	couleur sélectionnée

5.8 userOled.h

```
00001 /*
00002  * userOled.h
00003  *
00004  *  Created on: Feb 7, 2024
00005  *      Author: VACHER
```

```
00006  */
00007
00008 #ifndef USEROLED_H_
00009 #define USEROLED_H_
00010
00011 #include "main.h"
00012 #include "ssd1306.h"
00013
00014 void drawBorder();
00015 void drawTile(uint8_t x, uint8_t y, SSD1306_COLOR color);
00016 void drawPlayZone(int stack[23][16]);
00017 void drawGameOver();
00018 void ssd1306_SetCursorVertical(uint8_t x, uint8_t y);
00019 char ssd1306_WriteStringVertical(char* str, FontDef Font, SSD1306_COLOR color);
00020 void drawTetrisStartGame();
00021 void clearPlayZone();
00022 void drawBorderDecor(SSD1306_COLOR color);
00023 void drawTetriminos();
00024 void Clignotement_Ecran();
00025 void Clignotement_Click();
00026
00027 #endif /* USEROLED_H_ */
```

Index