

Proyecto Hardware

Generado por Doxygen 1.8.6

Sábado, 19 de Diciembre de 2015 12:07:25

Índice general

1	Índice de estructura de datos	1
1.1	Estructura de datos	1
2	Índice de archivos	3
2.1	Lista de archivos	3
3	Documentación de las estructuras de datos	5
3.1	Referencia de la Estructura BITMAP	5
4	Documentación de archivos	7
4.1	Referencia del Archivo 8led.h	7
4.1.1	Documentación de las funciones	7
4.1.1.1	D8Led_blink_symbol	7
4.1.1.2	D8Led_current_symbol	7
4.1.1.3	D8Led_init	8
4.1.1.4	D8Led_symbol	8
4.2	Referencia del Archivo Bmp.h	8
4.2.1	Descripción detallada	9
4.2.2	Documentación de las funciones	9
4.2.2.1	BitmapPop	9
4.2.2.2	BitmapPush	9
4.2.2.3	BitmapView	9
4.2.2.4	CursorPush	9
4.2.2.5	CursorView	9
4.3	Referencia del Archivo Button.h	10
4.3.1	Descripción detallada	10
4.3.2	Documentación de las funciones	10
4.3.2.1	action	10
4.3.2.2	Button_init	10
4.3.2.3	Button_low_next	10
4.3.2.4	Button_low_update_screen	11
4.3.2.5	Button_next	11

4.3.2.6	Button_reconfigure_range	11
4.3.2.7	Button_set_valor_actual	11
4.3.2.8	Button_update_screen	11
4.3.2.9	Button_valor_actual	11
4.4	Referencia del Archivo lcd.h	11
4.4.1	Descripción detallada	13
4.4.2	Documentación de los 'defines'	13
4.4.2.1	LCD_Active_PutPixel	13
4.4.2.2	LCD_PutPixel	13
4.5	Referencia del Archivo still-alive-lyrics.h	13
4.5.1	Descripción detallada	14
4.6	Referencia del Archivo sudoku_2015.h	14
4.6.1	Descripción detallada	14
4.6.2	Documentación de las funciones	14
4.6.2.1	celda_cambiar_candidatos	14
4.6.2.2	celda_es_candidato	15
4.6.2.3	celda_es_error	15
4.6.2.4	celda_es_pista	15
4.6.2.5	celda_leer_valor	15
4.6.2.6	celda_poner_valor	15
4.6.2.7	init_game	16
4.6.2.8	sudoku_candidatos_arm	16
4.6.2.9	sudoku_empty_grid	16
4.6.2.10	sudoku_recalcular	16
4.7	Referencia del Archivo sudoku_collection_san.h	16
4.7.1	Descripción detallada	17
4.7.2	Documentación de las funciones	17
4.7.2.1	sudoku_collection_descomprime	17
4.7.3	Documentación de las variables	17
4.7.3.1	cuadrículaCasiResuelta	17
4.7.3.2	cuadrículas	17
4.8	Referencia del Archivo sudoku_graphics.h	17
4.8.1	Descripción detallada	18
4.8.2	Documentación de las funciones	18
4.8.2.1	sudoku_graphics_draw_base	18
4.8.2.2	sudoku_graphics_draw_state	18
4.8.2.3	sudoku_graphics_draw_time	18
4.8.2.4	sudoku_graphics_fill_from_data	19
4.8.2.5	sudoku_graphics_mark_error	19
4.8.2.6	sudoku_graphics_print_final_screen	19

4.8.2.7	sudoku_graphics_print_instructions	19
4.8.2.8	sudoku_graphics_print_still_alive	19
4.8.2.9	sudoku_graphics_print_title_screen	19
4.8.2.10	sudoku_graphics_put_number_in_square	20
4.8.2.11	sudoku_graphics_remark_error_in_square	20
4.8.2.12	sudoku_graphics_remark_square	20
4.8.2.13	sudoku_graphics_update_lcd	20
4.9	Referencia del Archivo Timer2.h	20
4.9.1	Descripción detallada	20
4.9.2	Documentación de las funciones	21
4.9.2.1	Timer2_Empezar	21
4.9.2.2	Timer2_Inicializar	21
4.9.2.3	Timer2_Leer	21
4.9.2.4	Timer2_Reiniciar	21
Índice		22

Capítulo 1

Índice de estructura de datos

1.1. Estructura de datos

Lista de estructuras con una breve descripción:

BITMAP	5
----------------------------------	---

Capítulo 2

Indice de archivos

2.1. Lista de archivos

Lista de todos los archivos documentados y con descripciones breves:

8led.h	Funciones de control del display 8-segmentos	7
aperture-logo-bitmap.h	??
Bmp.h	Definicion mapas de bits del LCD	8
Button.h	Modulo que gestiona los rebotes de los botones	10
lcd.h	Funciones de visualizacion y control LCD	11
still-alive-lyrics.h	Archivo de recursos con los creditos finales	13
sudoku_2015.h	Modulo que contiene funciones relacionadas con el analisis de sudokus	14
sudoku_collection_san.h	Fichero de recursos que contiene cuadrículas	16
sudoku_graphics.h	Modulo que actua de capa de abstraccion entre el juego y la pantalla	17
Timer2.h	Modulo que gestiona un contador	20
title-bitmap.h	??
common/ 44b.h	??
common/ 44blib.h	??
common/ def.h	??
common/ option.h	??

Capítulo 3

Documentación de las estructuras de datos

3.1. Referencia de la Estructura BITMAP

Campos de datos

- INT8U **ucFlags**
- INT8U **ucBitsPix**
- INT16U **usWidth**
- INT16U **usHeight**
- INT32U **ulTransparentColor**
- INT8U * **pucStart**

La documentación para esta estructura fue generada a partir del siguiente fichero:

- [Bmp.h](#)

Capítulo 4

Documentación de archivos

4.1. Referencia del Archivo 8led.h

Funciones de control del display 8-segmentos.

Funciones

- void `D8Led_init` (void)
Inicializacion del sistema del 8 segmentos.
- void `D8Led_symbol` (int value)
Cambia el valor en el 8 segmentos.
- int `D8Led_current_symbol` (void)
Devuelve el simbolo actualmente en el 8 segmentos.
- void `D8Led_blink_symbol` (int value, int ms)
Pone temporalmente un nuevo simbolo el 8 segmentos.

4.1.1. Documentación de las funciones

4.1.1.1. void `D8Led_blink_symbol` (int value, int ms)

Cambia temporalmente el simbolo en el 8 segmentos al valor value (este parametro ha de cumplir las mismas condiciones que para su uso en la funcion `D8Led_symbol`). El nuevo simbolo permanecera en el 8 segmentos durante los siguientes ms milisegundos. A continuacion se volvera a poner el valor inicial. Es una funcion bloqueante.

Parámetros

in	value	Valor a poner en el 8 segmentos
in	ms	Milisegundos que permanecera el simbolo en el 8 segmentos

4.1.1.2. int `D8Led_current_symbol` (void)

Devuelve el valor en el 8 segmentos, interpretado como un numero decimal en el rango [0,15]

Devuelve

Simbolo actualmente en el 8 segmentos

4.1.1.3. void D8Led_init (void)

Inicializa el sistema del 8 segmentos, y pone el simbolo 0 en el mismo

4.1.1.4. void D8Led_symbol (int value)

Cambia el simbolo en el 8 segmentos al numero pasado, interpretado en hexadecimal. En caso de que el valor no pueda ser mostrado con un unico digito hexadecimal, no cambia el numero en el 8 segmentos.

Parámetros

<code>in</code>	<code>value</code>	Valor a poner en el 8 segmentos
-----------------	--------------------	---------------------------------

4.2. Referencia del Archivo Bmp.h

Definicion mapas de bits del LCD.

```
#include "def.h"
```

Estructuras de datos

- struct [BITMAP](#)

'defines'

- #define **BLACK** 0xf
- #define **WHITE** 0x0
- #define **LIGHTGRAY** 0x5
- #define **DARKGRAY** 0xa
- #define **TRANSPARENCY** 0xff

'typedefs'

- typedef struct [BITMAP](#) **STRU_BITMAP**
- typedef struct [BITMAP](#) * **pSTRU_BITMAP**

Funciones

- void [BitmapView](#) (INT16U x, INT16U y, [STRU_BITMAP](#) Stru_Bitmap)
Display bitmap in virtual buffer.
- void **BitmapViewHorizontallyCentered** (INT16U y, [STRU_BITMAP](#) Stru_Bitmap)
- void **BitmapViewVerticallyCentered** (INT16U x, [STRU_BITMAP](#) Stru_Bitmap)
- void **BitmapViewCentered** ([STRU_BITMAP](#) Stru_Bitmap)
- void [BitmapPush](#) (INT16U x, INT16U y, [STRU_BITMAP](#) Stru_Bitmap)
Push bitmap data into LCD active buffer.
- void [BitmapPop](#) (INT16U x, INT16U y, [STRU_BITMAP](#) Stru_Bitmap)
Pop bitmap data into LCD active buffer.
- void [CursorInit](#) (void)
Cursor init.
- void [CursorView](#) (INT16U x, INT16U y)

Cursor display.

- void **CursorPush** (INT16U x, INT16U y)

Cursor push.

- void **CursorPop** (void)

Cursor pop.

4.2.1. Descripción detallada

Autor

4.2.2. Documentación de las funciones

4.2.2.1. void BitmapPop (INT16U x, INT16U y, STRU_BITMAP Stru_Bitmap)

Parámetros

in	x	X coordinate of the position to pop into
in	y	Y coordinate of the position to pop into
in	Stru_Bitmap	Bitmap to pop

4.2.2.2. void BitmapPush (INT16U x, INT16U y, STRU_BITMAP Stru_Bitmap)

Parámetros

in	x	X coordinate of the position to draw
in	y	Y coordinate of the position to draw
in	Stru_Bitmap	Bitmap to draw

4.2.2.3. void BitmapView (INT16U x, INT16U y, STRU_BITMAP Stru_Bitmap)

Parámetros

in	x	X coordinate of the position to draw
in	y	Y coordinate of the position to draw
in	Stru_Bitmap	Bitmap to draw

4.2.2.4. void CursorPush (INT16U x, INT16U y)

Parámetros

in	x	X coordinate of the position to push
in	y	Y coordinate of the position to push

4.2.2.5. void CursorView (INT16U x, INT16U y)

Parámetros

in	x	X coordinate of the position to draw
in	y	Y coordinate of the position to draw

4.3. Referencia del Archivo Button.h

Modulo que gestiona los rebotes de los botones.

Funciones

- int [Button_valor_actual](#) (void)
- int [Button_next](#) (void)
- void [Button_init](#) (int min, int max)
- void [action](#) (int n)
- void [Button_set_valor_actual](#) (int n)
- void [Button_low_next](#) (void)
- int [Button_update_screen](#) ()
- void [Button_low_update_screen](#) ()
- void [Button_reconfigure_range](#) (int new_min, int new_max)

4.3.1. Descripción detallada

Modulo que se encarga de gestionar los botones y su uso en el proyecto, inicializarlos y sincronizar el 7 segmentos con el valor interno del boton.

Autor

Guillermo Robles Gonzalez

4.3.2. Documentación de las funciones

4.3.2.1. void action (int *n*)

Accion asociada a la pulsacion del boton de codigo n

Parámetros

in	<i>n</i>	Codigo de boton pulsado
----	----------	-------------------------

4.3.2.2. void Button_init (int *min*, int *max*)

Iniciar el sistema de botones, poniendo la pantalla al valor minimo pasado

Parámetros

in	<i>min</i>	Valor minimo que aparecera en la pantalla
in	<i>max</i>	Valor maximo que aparecera en la pantalla

4.3.2.3. void Button_low_next (void)

Baja el flag interno de next

4.3.2.4. void Button_low_update_screen ()

Baja el flag interno indicando que se han realizado acciones

4.3.2.5. int Button_next (void)

Informa de si se ha de avanzar

Devuelve

1 en caso de que el flag este activo, 0 en caso contrario

4.3.2.6. void Button_reconfigure_range (int new_min, int new_max)

Reconfigura el rango que maneja el boton, solo seran validos valores en el rango [0,15]

Parámetros

in	new_min	Nuevo minimo
in	new_max	Nuevo maximo

4.3.2.7. void Button_set_valor_actual (int n)

Ajusta el valor de la cuenta interna del boton, actualizando el display. Ha de pertenecer al rango al cual esta actualmente configurado el boton.

Parámetros

in	n	nuevo valor a poner
----	---	---------------------

4.3.2.8. int Button_update_screen ()

Indica si ha sido realizada alguna accion que afecte al estado interno del boton, tanto la pulsacion de un boton como la repeticion de una accion

Devuelve

1 en caso de que el flag este activo, 0 en caso contrario

4.3.2.9. int Button_valor_actual (void)

Valor actual de la cuenta interna

Devuelve

valor actual

4.4. Referencia del Archivo lcd.h

Funciones de visualizacion y control LCD.

```
#include "def.h"
```

'defines'

- #define **TLCD_160_240** (0)
- #define **VLCD_240_160** (1)
- #define **CLCD_240_320** (2)
- #define **MLCD_320_240** (3)
- #define **ELCD_640_480** (4)
- #define **SLCD_160_160** (5)
- #define **LCD_TYPE** MLCD_320_240
- #define **SCR_XSIZE** (320)
- #define **SCR_YSIZE** (240)
- #define **LCD_XSIZE** (320)
- #define **LCD_YSIZE** (240)
- #define **MODE_MONO** (1)
- #define **MODE_GREY4** (4)
- #define **MODE_GREY16** (16)
- #define **MODE_COLOR** (256)
- #define **Ascii_W** 8
- #define **XWIDTH** 6
- #define **BLACK** 0xf
- #define **WHITE** 0x0
- #define **LIGHTGRAY** 0x5
- #define **DARKGRAY** 0xa
- #define **TRANSPARENCY** 0xff
- #define **HOZVAL** (LCD_XSIZE/4-1)
- #define **HOZVAL_COLOR** (LCD_XSIZE*3/8-1)
- #define **LINEVAL** (LCD_YSIZE -1)
- #define **MVAL** (13)
- #define **M5D**(n) ((n) & 0x1ffff)
- #define **MVAL_USED** 0
- #define **ARRAY_SIZE_MONO** (SCR_XSIZE/8*SCR_YSIZE)
- #define **ARRAY_SIZE_GREY4** (SCR_XSIZE/4*SCR_YSIZE)
- #define **ARRAY_SIZE_GREY16** (SCR_XSIZE/2*SCR_YSIZE)
- #define **ARRAY_SIZE_COLOR** (SCR_XSIZE/1*SCR_YSIZE)
- #define **CLKVAL_MONO** (12)
- #define **CLKVAL_GREY4** (12)
- #define **CLKVAL_GREY16** (12)
- #define **CLKVAL_COLOR** (10)
- #define **LCD_BUF_SIZE** (SCR_XSIZE*SCR_YSIZE/2)
- #define **LCD_ACTIVE_BUFFER** (0xc300000)
- #define **LCD_VIRTUAL_BUFFER** (0xc300000 + LCD_BUF_SIZE)
- #define **LCD_PutPixel**(x, y, c)
- #define **LCD_Active_PutPixel**(x, y, c)
- #define **GUISWAP**(a, b) {a^=b; b^=a; a^=b;}

Funciones

- **INT8U LCD_GetPixel** (INT16U usX, INT16U usY)
- **void Lcd_Clr** (void)
- **void Lcd_Test** (void)
- **void Lcd_Dma_Trans** (void)
- **void LcdVirtualToTrue** (void)
- **void LcdClrRect** (INT16 usLeft, INT16 usTop, INT16 usRight, INT16 usBottom, INT8U ucColor)
- **void Lcd_Draw_Box** (INT16 usLeft, INT16 usTop, INT16 usRight, INT16 usBottom, INT8U ucColor)

- void **Lcd_Draw_Box_inverted** (INT16 usLeft, INT16 usTop, INT16 usRight, INT16 usBottom, INT8U ucColor)
- void **Lcd_Draw_Filled_Box** (INT16 usLeft, INT16 usTop, INT16 usRight, INT16 usBottom, INT8U ucColor)
- void **Lcd_Draw_Line** (INT16 usX0, INT16 usY0, INT16 usX1, INT16 usY1, INT8U ucColor, INT16U usWidth)
- void **Lcd_Draw_HLine** (INT16 usX0, INT16 usX1, INT16 usY0, INT8U ucColor, INT16U usWidth)
- void **Lcd_Draw_VLine** (INT16 usY0, INT16 usY1, INT16 usX0, INT8U ucColor, INT16U usWidth)
- void **Lcd_Draw_HLine_pointed** (INT16 usX0, INT16 usX1, INT16 usY0, INT8U ucColor, INT16U usWidth)
- void **Lcd_Draw_VLine_pointed** (INT16 usY0, INT16 usY1, INT16 usX0, INT8U ucColor, INT16U usWidth)
- void **Lcd_Draw_HLine_inverted** (INT16 usX0, INT16 usX1, INT16 usY0, INT16U usWidth)
- void **Lcd_Draw_VLine_inverted** (INT16 usY0, INT16 usY1, INT16 usX0, INT16U usWidth)
- void **Lcd_Anti_Disp** (INT16U usX0, INT16U usY0, INT16U usX1, INT16U usY1)
- void **Lcd_DisplayChar** (INT16U usX0, INT16U usY0, INT8U ForeColor, INT8U ucChar)
- void **Lcd_DisplayChar_inverted** (INT16U usX0, INT16U usY0, INT8U ForeColor, INT8U ucChar)
- void **Lcd_DisplayShort** (INT16 sX, INT16 sY, INT16U usInt)
- void **Lcd_Circle** (INT8 X, INT8 Y, INT16 radius, INT8U ForeColor)
- void **Zdma0Done** (void) __attribute__((interrupt("IRQ")))
- void **Lcd_DspAscll6x8** (INT16U usX0, INT16U usY0, INT8U ForeColor, INT8U *pucChar)
- void **Lcd_DspAscll8x16** (INT16U x0, INT16U y0, INT8U ForeColor, INT8U *s)
- void **Lcd_DspAscll8x16HorizontallyCentered** (INT16U y0, INT8U ForeColor, INT8U *s)
- void **Lcd_DspAscll8x16HorizontallyCentered_inverted** (INT16U y0, INT8U ForeColor, INT8U *s)
- void **Lcd_DspHz16** (INT16U x0, INT16U y0, INT8U ForeColor, INT8U *s)
- void **ReverseLine** (INT32U ulHeight, INT32U ulY)
- void **ReverseSquare** (INT32U ulX0, INT32U ulY0, INT32U ulX1, INT32U ulY1)

4.4.1. Descripción detallada

Versión

<P6-ARM>

4.4.2. Documentación de los 'defines'

4.4.2.1. #define LCD_Active_PutPixel(x, y, c)

Valor:

```
(* (INT32U *) (LCD_ACTIVE_BUFFER + (y) * SCR_XSIZE / 2 + (319 - (x)) / 8 * 4)) = \
((( (* (INT32U *) (LCD_ACTIVE_BUFFER + (y) * SCR_XSIZE / 2 + (319 - (x)) / 8 * 4)) & \
(~ (0xf0000000 » ((319 - (x)) % 8) * 4))) | ((c) « (7 - (319 - (x)) % 8) * 4))
```

4.4.2.2. #define LCD_PutPixel(x, y, c)

Valor:

```
(* (INT32U *) (LCD_VIRTUAL_BUFFER + (y) * SCR_XSIZE / 2 + ( (x)) / 8 * 4)) = \
((( (* (INT32U *) (LCD_VIRTUAL_BUFFER + (y) * SCR_XSIZE / 2 + ( (x)) / 8 * 4)) & \
(~ (0xf0000000 » (( (x)) % 8) * 4))) | ((c) « (7 - ( (x)) % 8) * 4))
```

4.5. Referencia del Archivo still-alive-lyrics.h

Archivo de recursos con los creditos finales.

'defines'

- #define **STILL_ALIVE_SIZE** 117

Variables

- char * **still_alive** [STILL_ALIVE_SIZE]

4.5.1. Descripción detallada

Autor

Guillermo Robles

4.6. Referencia del Archivo sudoku_2015.h

Modulo que contiene funciones relacionadas con el analisis de sudokus.

```
#include <inttypes.h>
```

'defines'

- #define **CELDA** uint16_t

Enumeraciones

- enum { **NUM_FILAS** = 9, **NUM_COLUMNAS** = 16, **TAM_REGION** = 3, **NUM_REGION** = 3 }

Informacion de la cuadrícula.

Funciones

- void [init_game](#) (void)
- void [celda_cambiar_candidatos](#) (uint8_t valor, CELDA cuadrícula[NUM_FILAS][NUM_COLUMNAS], uint8_t fila, uint8_t columna)
- int [sudoku_candidatos_arm](#) (CELDA cuadrícula[NUM_FILAS][NUM_COLUMNAS], uint8_t fila, uint8_t columna)
- int [sudoku_recalcular](#) (CELDA cuadrícula[NUM_FILAS][NUM_COLUMNAS])
- int [celda_es_error](#) (CELDA celda)
- int [celda_es_pista](#) (CELDA celda)
- int [celda_es_candidato](#) (CELDA celda, uint8_t valor)
- void [celda_poner_valor](#) (CELDA *celdaptr, uint8_t val)
- uint8_t [celda_leer_valor](#) (CELDA celda)
- void [sudoku_empty_grid](#) (CELDA cuadrícula[NUM_FILAS][NUM_COLUMNAS])

4.6.1. Descripción detallada

Autor

Guillermo Robles Gonzalez

4.6.2. Documentación de las funciones

- 4.6.2.1. void [celda_cambiar_candidatos](#) (uint8_t *valor*, CELDA *cuadrícula*[NUM_FILAS][NUM_COLUMNAS], uint8_t *fila*, uint8_t *columna*) [inline]

Funcion que modifica los posibles candidatos de una celda, eliminando una pista dada

Parámetros

in	valor	Pista que se eliminara
in	cuadricula	Cuadricula a modificar
in	fila	Coordenada fila de la casilla a modificar
in	columna	Coordenada columna de la casilla a modificar

4.6.2.2. int celda_es_candidato (CELDA celda, uint8_t valor) [inline]

Devuelve un numero mayor que cero en caso de que la celda sea pista inicial, devuelve 0 en caso contrario

Parámetros

in	celda	Celda a comprobar
----	-------	-------------------

Devuelve

0 si y solo si la celda no es pista

4.6.2.3. int celda_es_error (CELDA celda) [inline]

Devuelve un numero mayor de 0 si y solo si la celda esta marcada como error, devuelve 0 en caso contrario

Parámetros

in	celda	Celda a comprobar
----	-------	-------------------

Devuelve

0 en caso de que la celda sea correcta

4.6.2.4. int celda_es_pista (CELDA celda) [inline]

Devuelve un numero mayor que cero en caso de que la celda sea pista inicial, devuelve 0 en caso contrario

Parámetros

in	celda	Celda a comprobar
----	-------	-------------------

Devuelve

0 si y solo si la celda no es pista

4.6.2.5. uint8_t celda_leer_valor (CELDA celda) [inline]

Devuelve el numero en la celda dada (0 si es vacia)

Parámetros

out	celda	Celda a consultar
-----	-------	-------------------

4.6.2.6. void celda_poner_valor (CELDA * celdaptr, uint8_t val) [inline]

Actualiza el valor de la celda dada al valor dado

Parámetros

out	<i>celda</i>	Celda a cambiar
in	<i>val</i>	Nuevo valor

4.6.2.7. void init_game (void)

Inicializa el juego

4.6.2.8. int sudoku_candidatos_arm (CELDA *cuadrícula*[NUM_FILAS][NUM_COLUMNAS], uint8_t *fila*, uint8_t *columna*)

Funcion que dado una casilla en una cuadrícula, actualiza sus candidatos

Parámetros

in	<i>cuadrícula</i>	Cuadrícula a modificar
in	<i>fila</i>	Coordenada fila de la casilla a modificar
in	<i>columna</i>	Coordenada columna de la casilla a modificar

Devuelve

0 en caso de que la casilla este vacia, !=0 en caso contrario

4.6.2.9. void sudoku_empty_grid (CELDA *cuadrícula*[NUM_FILAS][NUM_COLUMNAS])

Pone cualquier casilla que no sea pista a 0

Parámetros

out	<i>cuadrícula</i>	Cuadrícula a editar
-----	-------------------	---------------------

4.6.2.10. int sudoku_recalcular (CELDA *cuadrícula*[NUM_FILAS][NUM_COLUMNAS])

Funcion que dado una cuadrícula, actualiza todos sus candidatos y reajusta los errores como sea necesario, usando como hoja una funcion ARM.

Parámetros

out	<i>cuadrícula</i>	Cuadrícula a modificar
-----	-------------------	------------------------

Devuelve

Numero de casillas vacias, o -1 en caso de que haya errores

4.7. Referencia del Archivo sudoku_collection_san.h

Fichero de recursos que contiene cuadrículas.

```
#include "sudoku_2015.h"
```

```
'defines'
```

```
■ #define SUDOKU_COLLECTION_SAN_H_
```

- #define **NUM_CUADRICULAS** 10
- #define **COMPRESSED_SIZE** 41

Funciones

- void [sudoku_collection_descomprime](#) (char *compressed, CELDA destiny[NUM_FILAS][NUM_COLUMNAS])

Variables

- char [cuadriculas](#) [NUM_CUADRICULAS][COMPRESSED_SIZE]
Coleccion de cuadriculas comprimidas.
- char [cuadriculaCasiResuelta](#) [COMPRESSED_SIZE]
Cuadricula especial casi resuelta.

4.7.1. Descripción detallada

Modulo que contiene cuadriculas en formato comprimido, ademas de funciones de descompresion

Autor

Guillermo Robles Gonzalez

4.7.2. Documentación de las funciones

4.7.2.1. void [sudoku_collection_descomprime](#) (char * *compressed*, CELDA *destiny*[NUM_FILAS][NUM_COLUMNAS])

Funcion que convierte del formato comprimido al formato normal de uso

Parámetros

in	<i>compressed</i>	Cuadricula compresada
out	<i>destiny</i>	Zona de memoria en la que se descomprimira la cuadricula

4.7.3. Documentación de las variables

4.7.3.1. char [cuadriculaCasiResuelta](#)[COMPRESSED_SIZE]

Cuadricula especial en la cual solo resta introducir un valor

4.7.3.2. char [cuadriculas](#)[NUM_CUADRICULAS][COMPRESSED_SIZE]

Coleccion de cuadriculas en formato compresado

4.8. Referencia del Archivo sudoku_graphics.h

Modulo que actua de capa de abstraccion entre el juego y la pantalla.

```
#include "sudoku_2015.h"
#include "44blib.h"
#include "44b.h"
#include "def.h"
```

'defines'

- #define **SUDOKU_NUM_CUADS** 9
- #define **SUDOKU_SQUARE_SIZE** 18
- #define **SUDOKU_X0** 20
- #define **SUDOKU_Y0** 20
- #define **SUDOKU_FONT_HEIGHT** 16
- #define **SUDOKU_FONT_LENGTH** 8
- #define **ASCII_NUMBER_BASE** 48

Funciones

- void [sudoku_graphics_draw_base](#) ()
- void [sudoku_graphics_fill_from_data](#) (CELDA cuadrícula[NUM_FILAS][NUM_COLUMNAS])
- void [sudoku_graphics_update_lcd](#) ()
- void [sudoku_graphics_print_final_screen](#) (int tiempo_juego_s, int tiempo_calculos_ms, int errores)
- void [sudoku_graphics_put_number_in_square](#) (INT8 x, INT8 y, INT8 number, INT8U ucColor)
- void [sudoku_graphics_remark_square](#) (INT8 x, INT8 y)
- void [sudoku_graphics_mark_error](#) (INT8 x, INT8 y, INT8 error)
- void [sudoku_graphics_remark_error_in_square](#) (INT8 x, INT8 y, INT8 error)
- void [sudoku_graphics_print_title_screen](#) ()
- void [sudoku_graphics_print_instructions](#) ()
- void [sudoku_graphics_print_still_alive](#) (int lineNumber)
- void [sudoku_graphics_draw_state](#) (int state, int number)
- void [sudoku_graphics_draw_time](#) (int time_playing_s, int time_calculating_ms)

4.8.1. Descripción detallada

Modulo que contiene un conjunto de funciones que interactuan con el aspecto grafico del sudoku, y que abstrae las operaciones sobre la pantalla en un conjunto de operaciones de alto nivel

Autor

Guillermo Robles Gonzalez

4.8.2. Documentación de las funciones**4.8.2.1. void [sudoku_graphics_draw_base](#) ()**

Dibuja la base del sudoku (cuadrícula, numeracion de la misma, frase de informacion)

4.8.2.2. void [sudoku_graphics_draw_state](#) (int *state*, int *number*)

Dibuja el estado de seleccion en el que nos encontramos state=0 => esperando fila state=1 => esperando columna state=2 => esperando valor

Parámetros

in	<i>state</i>	Estado actual, en forma textual
in	<i>number</i>	Valor actual del boton

4.8.2.3. void [sudoku_graphics_draw_time](#) (int *time_playing_s*, int *time_calculating_ms*)

Dibuja el tiempo, tanto el pasado como el de los calculos

Parámetros

in	<i>time_playing_s</i>	Tiempo de juego, en segundos
in	<i>time_calculating- _ms</i>	Tiempo de calculo, en milisegundos

4.8.2.4. void sudoku_graphics_fill_from_data (CELDA *cuadrícula*[NUM_FILAS][NUM_COLUMNAS])

Rellena el sudoku con la informacion de la cuadrícula dada

Parámetros

in	<i>cuadrícula</i>	Cuadrícula de la que se cojera la informacion
----	-------------------	---

4.8.2.5. void sudoku_graphics_mark_error (INT8 *x*, INT8 *y*, INT8 *error*) [inline]

Marca el error dado en la casilla dada (Negando los colores en su interior)

Parámetros

in	<i>x</i>	Posicion x de la casilla
in	<i>y</i>	Posicion y de la casilla
in	<i>error</i>	Error a marcar (intervalo 1-9)

4.8.2.6. void sudoku_graphics_print_final_screen (int *tiempo_juego_s*, int *tiempo_calculos_ms*, int *errores*)

Imprime la pantalla final, si el numero de errores es 0, se imprimira la pantalla de exito, en caso contrario se imprimira un mensaje de fracaso.

Parámetros

in	<i>tiempo_juego_- ms</i>	Tiempo que ha durado el juego, en ms
in	<i>tiempo_calculos- _ms</i>	Tiempo que han durado los calculos, en ms
in	<i>tiempo_- renderizado_ms</i>	Tiempo de renderizado, en ms
in	<i>errores</i>	Numero de errores actual

4.8.2.7. void sudoku_graphics_print_instructions ()

Imprime la pantalla de instrucciones

4.8.2.8. void sudoku_graphics_print_still_alive (int *lineNumber*)

Imprime 6 lineas de Still Alive a partir de la linea dada

Parámetros

in	<i>lineNumber</i>	Línea a partir de la cual comenzar a imprimir
----	-------------------	---

4.8.2.9. void sudoku_graphics_print_title_screen ()

Imprime la pantalla de título

4.8.2.10. `void sudoku_graphics_put_number_in_square (INT8 x, INT8 y, INT8 number, INT8U ucColor) [inline]`

Rellena una casilla con el numero dado

Parámetros

in	x	Posicion x de la casilla a rellenar
in	y	Posicion y de la casilla a rellenar
in	number	Numero con el cual rellenarla
in	ucColor	Color de letra

4.8.2.11. `void sudoku_graphics_remark_error_in_square (INT8 x, INT8 y, INT8 error)`

Remarca una marca de error en una casilla

Parámetros

in	x	Posicion x de la casilla
in	y	Posicion y de la casilla
in	error	Error a remarcar

4.8.2.12. `void sudoku_graphics_remark_square (INT8 x, INT8 y) [inline]`

Remarca una casilla de forma visible al usuario (Aplicando un enmarcado)

Parámetros

in	x	Posicion x de la casilla
in	y	Posicion y de la casilla

4.8.2.13. `void sudoku_graphics_update_lcd ()`

Actualiza el LCD con la informacion del sudoku

4.9. Referencia del Archivo Timer2.h

Modulo que gestiona un contador.

```
#include "44b.h"
```

Funciones

- void [Timer2_Inicializar](#) (void)
- void [Timer2_Empezar](#) (void)
- void [Timer2_Reiniciar](#) (void)
- uint32_t [Timer2_Leer](#) (void)

4.9.1. Descripción detallada

Modulo que gestiona un timer de precision milisegundos, usa el timer 2

Autor

Guillermo Robles Gonzalez

4.9.2. Documentación de las funciones

4.9.2.1. void Timer2_Empezar (void)

Funcion que comienza la cuenta.

4.9.2.2. void Timer2_Inicializar (void)

Funcion que inicializa el contador

4.9.2.3. uint32_t Timer2_Leer (void)

Devuelve el contador interno en microsegundos

Devuelve

Tiempo en microsegundos desde el ultimo reinicio del contador

4.9.2.4. void Timer2_Reiniciar (void)

Funcion que resetea el contador interno a 0

Índice alfabético

8led.h, [7](#)
 D8Led_blink_symbol, [7](#)
 D8Led_current_symbol, [7](#)
 D8Led_init, [7](#)
 D8Led_symbol, [8](#)

action
 Button.h, [10](#)

BITMAP, [5](#)

BitmapPop
 Bmp.h, [9](#)

BitmapPush
 Bmp.h, [9](#)

BitmapView
 Bmp.h, [9](#)

Bmp.h, [8](#)
 BitmapPop, [9](#)
 BitmapPush, [9](#)
 BitmapView, [9](#)
 CursorPush, [9](#)
 CursorView, [9](#)

Button.h, [10](#)
 action, [10](#)
 Button_init, [10](#)
 Button_low_next, [10](#)
 Button_low_update_screen, [10](#)
 Button_next, [11](#)
 Button_reconfigure_range, [11](#)
 Button_set_valor_actual, [11](#)
 Button_update_screen, [11](#)
 Button_valor_actual, [11](#)

Button_init
 Button.h, [10](#)

Button_low_next
 Button.h, [10](#)

Button_low_update_screen
 Button.h, [10](#)

Button_next
 Button.h, [11](#)

Button_reconfigure_range
 Button.h, [11](#)

Button_set_valor_actual
 Button.h, [11](#)

Button_update_screen
 Button.h, [11](#)

Button_valor_actual
 Button.h, [11](#)

celda_cambiar_candidatos
 sudoku_2015.h, [14](#)

celda_es_candidato
 sudoku_2015.h, [15](#)

celda_es_error
 sudoku_2015.h, [15](#)

celda_es_pista
 sudoku_2015.h, [15](#)

celda_leer_valor
 sudoku_2015.h, [15](#)

celda_poner_valor
 sudoku_2015.h, [15](#)

cuadrículaCasiResuelta
 sudoku_collection_san.h, [17](#)

cuadrículas
 sudoku_collection_san.h, [17](#)

CursorPush
 Bmp.h, [9](#)

CursorView
 Bmp.h, [9](#)

D8Led_blink_symbol
 8led.h, [7](#)

D8Led_current_symbol
 8led.h, [7](#)

D8Led_init
 8led.h, [7](#)

D8Led_symbol
 8led.h, [8](#)

init_game
 sudoku_2015.h, [16](#)

LCD_Active_PutPixel
 lcd.h, [13](#)

LCD_PutPixel
 lcd.h, [13](#)

lcd.h, [11](#)
 LCD_Active_PutPixel, [13](#)
 LCD_PutPixel, [13](#)

still-alive-lyrics.h, [13](#)

sudoku_2015.h, [14](#)
 celda_cambiar_candidatos, [14](#)
 celda_es_candidato, [15](#)
 celda_es_error, [15](#)
 celda_es_pista, [15](#)
 celda_leer_valor, [15](#)
 celda_poner_valor, [15](#)
 init_game, [16](#)
 sudoku_candidatos_arm, [16](#)

- sudoku_empty_grid, [16](#)
 - sudoku_recalcular, [16](#)
- sudoku_candidatos_arm
 - sudoku_2015.h, [16](#)
- sudoku_collection_descomprime
 - sudoku_collection_san.h, [17](#)
- sudoku_collection_san.h, [16](#)
 - cuadrículaCasiResuelta, [17](#)
 - cuadriculas, [17](#)
 - sudoku_collection_descomprime, [17](#)
- sudoku_empty_grid
 - sudoku_2015.h, [16](#)
- sudoku_graphics.h, [17](#)
 - sudoku_graphics_draw_base, [18](#)
 - sudoku_graphics_draw_state, [18](#)
 - sudoku_graphics_draw_time, [18](#)
 - sudoku_graphics_fill_from_data, [19](#)
 - sudoku_graphics_mark_error, [19](#)
 - sudoku_graphics_print_final_screen, [19](#)
 - sudoku_graphics_print_instructions, [19](#)
 - sudoku_graphics_print_still_alive, [19](#)
 - sudoku_graphics_print_title_screen, [19](#)
 - sudoku_graphics_put_number_in_square, [19](#)
 - sudoku_graphics_remark_error_in_square, [20](#)
 - sudoku_graphics_remark_square, [20](#)
 - sudoku_graphics_update_lcd, [20](#)
- sudoku_graphics_draw_base
 - sudoku_graphics.h, [18](#)
- sudoku_graphics_draw_state
 - sudoku_graphics.h, [18](#)
- sudoku_graphics_draw_time
 - sudoku_graphics.h, [18](#)
- sudoku_graphics_fill_from_data
 - sudoku_graphics.h, [19](#)
- sudoku_graphics_mark_error
 - sudoku_graphics.h, [19](#)
- sudoku_graphics_print_final_screen
 - sudoku_graphics.h, [19](#)
- sudoku_graphics_print_instructions
 - sudoku_graphics.h, [19](#)
- sudoku_graphics_print_still_alive
 - sudoku_graphics.h, [19](#)
- sudoku_graphics_print_title_screen
 - sudoku_graphics.h, [19](#)
- sudoku_graphics_put_number_in_square
 - sudoku_graphics.h, [19](#)
- sudoku_graphics_remark_error_in_square
 - sudoku_graphics.h, [20](#)
- sudoku_graphics_remark_square
 - sudoku_graphics.h, [20](#)
- sudoku_graphics_update_lcd
 - sudoku_graphics.h, [20](#)
- sudoku_recalcular
 - sudoku_2015.h, [16](#)
- Timer2.h, [20](#)
 - Timer2_Empezar, [21](#)
 - Timer2_Inicializar, [21](#)
 - Timer2_Leer, [21](#)
 - Timer2_Reiniciar, [21](#)
- Timer2_Empezar
 - Timer2.h, [21](#)
- Timer2_Inicializar
 - Timer2.h, [21](#)
- Timer2_Leer
 - Timer2.h, [21](#)
- Timer2_Reiniciar
 - Timer2.h, [21](#)