CyberStorm-Basilisk

0.0.1

Generated by Doxygen 1.13.1

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 __display_struct Struct Reference

`#include <display.h>`

Public Attributes

- shiftreg sreg
- uint16_t data
- bool show_state

3.1.1 Member Data Documentation

3.1.1.1 data

`uint16_t __display_struct::data`

3.1.1.2 show_state

`bool __display_struct::show_state`

3.1.1.3 sreg

`shiftreg __display_struct::sreg`

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

- include/drako/display.h

3.2 at28c64b Struct Reference

`#include <at28c64b.h>`

Public Attributes

- uint32_t data_bus
- uint32_t addr_bus
- uint8_t we
- uint8_t oe
- uint8_t ce
- bool data_dir

3.2.1 Member Data Documentation

3.2.1.1 addr_bus

`uint32_t at28c64b::addr_bus`

3.2.1.2 ce

`uint8_t at28c64b::ce`

3.2.1.3 data_bus

`uint32_t at28c64b::data_bus`

3.2.1.4 data_dir

`bool at28c64b::data_dir`

3.2.1.5 oe

`uint8_t at28c64b::oe`

3.2.1.6 we

`uint8_t at28c64b::we`

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

- include/drako/hardware/at28c64b.h

3.3 sn74hc595n Struct Reference

`#include <sn74hc595n.h>`

Public Attributes

- uint8_t ser
- uint8_t rclk
- uint8_t srclk
- uint8_t oe

3.3.1 Member Data Documentation

3.3.1.1 oe

`uint8_t sn74hc595n::oe`

3.3.1.2 rclk

`uint8_t sn74hc595n::rclk`

3.3.1.3 ser

`uint8_t sn74hc595n::ser`

3.3.1.4 srclk

`uint8_t sn74hc595n::srclk`

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

- include/drako/hardware/sn74hc595n.h

3.4 terminal_command_struct Struct Reference

`#include <terminal.h>`

Public Attributes

- char ** argv
- size_t argc

3.4.1 Member Data Documentation

3.4.1.1 argc

`size_t terminal_command_struct::argc`

3.4.1.2 argv

`char** terminal_command_struct::argv`

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

- include/drako/terminal.h

Chapter 4

File Documentation

4.1 include/drako/display.h File Reference

```
#include <stdbool.h>
#include "drako/hardware/sn74hc595n.h"
```

Classes

- struct __display_struct

Macros

- #define DRKO_DISPL_0 0x003F
- #define DRKO_DISPL_1 0x0006
- #define DRKO_DISPL_2 0x005B
- #define DRKO_DISPL_3 0x004F
- #define DRKO_DISPL_4 0x0066
- #define DRKO_DISPL_5 0x006D
- #define DRKO_DISPL_6 0x007D
- #define DRKO_DISPL_7 0x0007
- #define DRKO_DISPL_8 0x007F
- #define DRKO_DISPL_9 0x006F
- #define DRKO_DISPL_A 0x0077
- #define DRKO_DISPL_B 0x007C
- #define DRKO_DISPL_C 0x0039
- #define DRKO_DISPL_D 0x005E
- #define DRKO_DISPL_E 0x0079
- #define DRKO_DISPL_F 0x0071
- #define DRKO_DISPL_P 0x0080
- #define DRKO_DISPM_0 0x3F00
- #define DRKO_DISPM_1 0x0600
- #define DRKO_DISPM_2 0x5B00
- #define DRKO_DISPM_3 0x4F00
- #define DRKO_DISPM_4 0x6600
- #define DRKO_DISPM_5 0x6D00

- #define DRKO_DISPM_6 0x7D00
- #define DRKO_DISPM_7 0x0700
- #define DRKO_DISPM_8 0x7F00
- #define DRKO_DISPM_9 0x6F00
- #define DRKO_DISPM_A 0x7700
- #define DRKO_DISPM_B 0x7C00
- #define DRKO_DISPM_C 0x3900
- #define DRKO_DISPM_D 0x5E00
- #define DRKO_DISPM_E 0x7900
- #define DRKO_DISPM_F 0x7100
- #define DRKO_DISPM_P 0x8000

Typedefs

- typedef struct __display_struct display

Functions

- void display_init (display *disp)
- void display_show (display *disp)
- void display_hide (display *disp)
- void display_clear (display *disp)
- void display_write (display *disp, uint16_t data)
- void display_hex (display *disp, uint8_t byte)
- uint16_t byte2disp (uint8_t data)

4.1.1 Macro Definition Documentation

4.1.1.1 DRKO_DISPL_0

```
#define DRKO_DISPL_0 0x003F
```

4.1.1.2 DRKO_DISPL_1

```
#define DRKO_DISPL_1 0x0006
```

4.1.1.3 DRKO_DISPL_2

```
#define DRKO_DISPL_2 0x005B
```

4.1.1.4 DRKO_DISPL_3

```
#define DRKO_DISPL_3 0x004F
```

4.1.1.5 DRKO_DISPL_4

```
#define DRKO_DISPL_4 0x0066
```

4.1.1.6 DRKO_DISPL_5

```
#define DRKO_DISPL_5 0x006D
```

4.1.1.7 DRKO_DISPL_6

```
#define DRKO_DISPL_6 0x007D
```

4.1.1.8 DRKO_DISPL_7

```
#define DRKO_DISPL_7 0x0007
```

4.1.1.9 DRKO_DISPL_8

```
#define DRKO_DISPL_8 0x007F
```

4.1.1.10 DRKO_DISPL_9

```
#define DRKO_DISPL_9 0x006F
```

4.1.1.11 DRKO_DISPL_A

```
#define DRKO_DISPL_A 0x0077
```

4.1.1.12 DRKO_DISPL_B

```
#define DRKO_DISPL_B 0x007C
```

4.1.1.13 DRKO_DISPL_C

```
#define DRKO_DISPL_C 0x0039
```

4.1.1.14 DRKO_DISPL_D

```
#define DRKO_DISPL_D 0x005E
```

4.1.1.15 DRKO_DISPL_E

```
#define DRKO_DISPL_E 0x0079
```

4.1.1.16 DRKO_DISPL_F

```
#define DRKO_DISPL_F 0x0071
```

4.1.1.17 DRKO_DISPL_P

```
#define DRKO_DISPL_P 0x0080
```

4.1.1.18 DRKO_DISPM_0

```
#define DRKO_DISPM_0 0x3F00
```

4.1.1.19 DRKO_DISPM_1

```
#define DRKO_DISPM_1 0x0600
```

4.1.1.20 DRKO_DISPM_2

```
#define DRKO_DISPM_2 0x5B00
```

4.1.1.21 DRKO_DISPM_3

```
#define DRKO_DISPM_3 0x4F00
```

4.1.1.22 DRKO_DISPM_4

```
#define DRKO_DISPM_4 0x6600
```

4.1.1.23 DRKO_DISPM_5

```
#define DRKO_DISPM_5 0x6D00
```

4.1.1.24 DRKO_DISPM_6

```
#define DRKO_DISPM_6 0x7D00
```

4.1.1.25 DRKO_DISPM_7

```
#define DRKO_DISPM_7 0x0700
```

4.1.1.26 DRKO_DISPM_8

```
#define DRKO_DISPM_8 0x7F00
```

4.1.1.27 DRKO_DISPM_9

```
#define DRKO_DISPM_9 0x6F00
```

4.1.1.28 DRKO_DISPM_A

```
#define DRKO_DISPM_A 0x7700
```

4.1.1.29 DRKO_DISPM_B

```
#define DRKO_DISPM_B 0x7C00
```

4.1.1.30 DRKO_DISPM_C

```
#define DRKO_DISPM_C 0x3900
```

4.1.1.31 DRKO_DISPM_D

```
#define DRKO_DISPM_D 0x5E00
```

4.1.1.32 DRKO_DISPM_E

```
#define DRKO_DISPM_E 0x7900
```

4.1.1.33 DRKO_DISPM_F

```
#define DRKO_DISPM_F 0x7100
```

4.1.1.34 DRKO_DISPM_P

```
#define DRKO_DISPM_P 0x8000
```

4.1.2 Typedef Documentation

4.1.2.1 display

```
typedef struct __display_struct display
```

4.1.3 Function Documentation

4.1.3.1 byte2disp()

```
uint16_t byte2disp (
            uint8_t data)
```

4.1.3.2 display_clear()

```
void display_clear (
            display * disp)
```

4.1.3.3 display_hex()

```
void display_hex (
            display * disp,
            uint8_t byte)
```

4.1.3.4 display_hide()

```
void display_hide (
            display * disp)
```

4.1.3.5 display_init()

```
void display_init (
            display * disp)
```

4.1.3.6 display_show()

```
void display_show (
            display * disp)
```

4.1.3.7 display_write()

```
void display_write (
            display * disp,
            uint16_t data)
```

4.2 display.h

Go to the documentation of this file.

```
00001 #ifndef __drako_display
00002 #define __drako_display
00003
00004 #include <stdbool.h>
00005 #include "drako/hardware/sn74hc595n.h"
00006
00007 // Least-Significant Digit
00008 #define DRKO_DISPL_0 0x003F
00009 #define DRKO_DISPL_1 0x0006
00010 #define DRKO_DISPL_2 0x005B
00011 #define DRKO_DISPL_3 0x004F
00012 #define DRKO_DISPL_4 0x0066
00013 #define DRKO_DISPL_5 0x006D
00014 #define DRKO_DISPL_6 0x007D
00015 #define DRKO_DISPL_7 0x0007
00016 #define DRKO_DISPL_8 0x007F
00017 #define DRKO_DISPL_9 0x006F
00018 #define DRKO_DISPL_A 0x0077
00019 #define DRKO_DISPL_B 0x007C
00020 #define DRKO_DISPL_C 0x0039
00021 #define DRKO_DISPL_D 0x005E
00022 #define DRKO_DISPL_E 0x0079
00023 #define DRKO_DISPL_F 0x0071
00024 #define DRKO_DISPL_P 0x0080
00025
00026 // Most-Significant Digit
00027 #define DRKO_DISPM_0 0x3F00
00028 #define DRKO_DISPM_1 0x0600
00029 #define DRKO_DISPM_2 0x5B00
00030 #define DRKO_DISPM_3 0x4F00
00031 #define DRKO_DISPM_4 0x6600
00032 #define DRKO_DISPM_5 0x6D00
00033 #define DRKO_DISPM_6 0x7D00
00034 #define DRKO_DISPM_7 0x0700
00035 #define DRKO_DISPM_8 0x7F00
00036 #define DRKO_DISPM_9 0x6F00
00037 #define DRKO_DISPM_A 0x7700
00038 #define DRKO_DISPM_B 0x7C00
00039 #define DRKO_DISPM_C 0x3900
00040 #define DRKO_DISPM_D 0x5E00
00041 #define DRKO_DISPM_E 0x7900
00042 #define DRKO_DISPM_F 0x7100
00043 #define DRKO_DISPM_P 0x8000
00044
00045 typedef struct __display_struct {
00046     shiftreg sreg;
00047     uint16_t data;
00048     bool show_state;
00049 } display;
00050
00051
00052 void display_init(display* disp);
00053 void display_show(display* disp);
00054 void display_hide(display* disp);
00055 void display_clear(display* disp);
00056 void display_write(display* disp, uint16_t data);
00057 void display_hex(display* disp, uint8_t byte);
00058 uint16_t byte2disp(uint8_t data);
00059
00060 static inline void display_select(display* disp) {
00061     shiftreg_select(&disp->sreg);
00062     if (disp->show_state)
00063         display_show(disp);
00064     else
00065         display_hide(disp);
00066 }
00067
00068 #endif
```

4.3 include/drako/hardware/at28c64b.h File Reference

```
#include <hardware/gpio.h>
#include <pico/stdlib.h>
#include <stdint.h>
#include <stdbool.h>
```

Classes

- struct at28c64b

Macros

- #define EEPROM_OP_DELAY 1000

Typedefs

- typedef struct at28c64b eeprom

Functions

- void _eeprom_gpio_init (eeprom *prom)
  Initialize GPIO pins used for EEPROM.
- void _eeprom_data_in (eeprom *prom)
  Sets data bus to input mode.
- void _eeprom_data_out (eeprom *prom)
  Sets data bus to input mode.
- void _eeprom_set_idle_condition (eeprom *prom)
  Sets all control pins to HIGH (inactive)
- void _eeprom_set_read_condition (eeprom *prom)
  Sets EEPROM read condition. Must be called before executing a READ.
- void _eeprom_set_write_condition (eeprom *prom)
  Sets EEPROM write condition. Must be called before executing a WRITE.
- void _eeprom_execute_write (eeprom *prom)
  Executes WRITE. Must be called AFTER setting a WRITE condition.
- void eeprom_init (eeprom *prom, uint32_t data_bus, uint32_t addr_bus, uint8_t we, uint8_t oe, uint8_t ce)
  Initialize a previously allocated EEPROM struct.
- void eeprom_select (eeprom *prom)
  Allows EEPROM to take control of shared GPIO pins.
- void eeprom_read8 (eeprom *prom, uint32_t addr, uint8_t *buff)
  Reads byte at specified address into buffer.
- void eeprom_write8 (eeprom *prom, uint32_t addr, uint8_t data)
  Write byte to EEPROM at specified address.

4.3.1 Macro Definition Documentation

4.3.1.1 EEPROM_OP_DELAY

```
#define EEPROM_OP_DELAY 1000
```

4.3.2 Typedef Documentation

4.3.2.1 eeprom

```
typedef struct at28c64b eeprom
```

4.3.3 Function Documentation

4.3.3.1 _eeprom_data_in()

```
void _eeprom_data_in (
            eeprom * prom)
```

Sets data bus to input mode.

Parameters

prom	Pointer to EEPROM struct

4.3.3.2 _eeprom_data_out()

```
void _eeprom_data_out (
            eeprom * prom)
```

Sets data bus to input mode.

Parameters

prom	Pointer to EEPROM struct

4.3.3.3 _eeprom_execute_write()

```
void _eeprom_execute_write (
            eeprom * prom)
```

Executes WRITE. Must be called AFTER setting a WRITE condition.

Parameters

prom	Pointer to EEPROM struct

4.3.3.4 _eeprom_gpio_init()

```
void _eeprom_gpio_init (
            eeprom * prom)
```

Initialize GPIO pins used for EEPROM.

Parameters

prom	Pointer to EEPROM struct

4.3.3.5 _eeprom_set_idle_condition()

```
void _eeprom_set_idle_condition (
            eeprom * prom)
```

Sets all control pins to HIGH (inactive)

Parameters

prom	Pointer to EEPROM struct

4.3.3.6 _eeprom_set_read_condition()

```
void _eeprom_set_read_condition (
            eeprom * prom)
```

Sets EEPROM read condition. Must be called before executing a READ.

Parameters

prom	Pointer to EEPROM struct

4.3.3.7 _eeprom_set_write_condition()

```
void _eeprom_set_write_condition (
            eeprom * prom)
```

Sets EEPROM write condition. Must be called before executing a WRITE.

Parameters

prom	Pointer to EEPROM struct

4.3.3.8 eeprom_init()

```
void eeprom_init (
            eeprom * prom,
            uint32_t data_bus,
            uint32_t addr_bus,
            uint8_t we,
            uint8_t oe,
            uint8_t ce)
```

Initialize a previously allocated EEPROM struct.

Parameters

prom	Pointer to EEPROM struct
data_bus	GPIO mask of pins used for EEPROM I/O
addr_bus	GPIO mask of pins used for EEPROM addressing
we	Write Enable GPIO pin
oe	Output Enable GPIO pin
ce	Chip Enable GPIO pin

4.3.3.9 eeprom_read8()

```
void eeprom_read8 (
            eeprom * prom,
            uint32_t addr,
            uint8_t * buff)
```

Reads byte at specified address into buffer.

Parameters

prom	Pointer to EEPROM struct
addr	Address to read from
buff	Buffer for storing read data

4.3.3.10 eeprom_select()

```
void eeprom_select (
            eeprom * prom)
```

Allows EEPROM to take control of shared GPIO pins.

Parameters

prom	Pointer to EEPROM struct

4.3.3.11 eeprom_write8()

```
void eeprom_write8 (
            eeprom * prom,
            uint32_t addr,
            uint8_t data)
```

Write byte to EEPROM at specified address.

Parameters

prom	Pointer to EEPROM struct
addr	Address at which to write data
data	Byte data to write to address

4.4 at28c64b.h

Go to the documentation of this file.

```
00001 #ifndef __drako_at28c64b
00002 #define __drako_at28c64b
00003
00004 #include <hardware/gpio.h>
00005 #include <pico/stdlib.h>
00006 #include <stdint.h>
00007 #include <stdbool.h>
00008
00009 #define EEPROM_OP_DELAY 1000
00010
00011 typedef struct at28c64b {
00012     uint32_t data_bus;
00013     uint32_t addr_bus;
00014     uint8_t we, oe, ce;
00015     bool data_dir;
00016 } eeprom;
00017
00018 void _eeprom_gpio_init(eeprom* prom);
00019 void _eeprom_data_in(eeprom* prom);
```

```
00020 void _eeprom_data_out(eeprom* prom);
00021 void _eeprom_set_idle_condition(eeprom* prom);
00022 void _eeprom_set_read_condition(eeprom* prom);
00023 void _eeprom_set_write_condition(eeprom* prom);
00024 void _eeprom_execute_write(eeprom* prom);
00025
00026 void eeprom_init(eeprom* prom, uint32_t data_bus, uint32_t addr_bus, uint8_t we, uint8_t oe, uint8_t
      ce);
00027 void eeprom_select(eeprom* prom);
00028
00029 void eeprom_read8(eeprom* prom, uint32_t addr, uint8_t* buff);
00030 void eeprom_write8(eeprom* prom, uint32_t addr, uint8_t data);
00031
00032 #endif
```

4.5 include/drako/hardware/sn74hc595n.h File Reference

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

Classes

- struct sn74hc595n

Macros

- #define SR_DELAY_US 10

Typedefs

- typedef struct sn74hc595n shiftreg

Functions

- void shiftreg_init (shiftreg *sreg, uint8_t ser, uint8_t rclk, uint8_t srclk, uint8_t oe)
- void shiftreg_select (shiftreg *sreg)
- void shiftreg_pulse_clock (shiftreg *sreg)
- void shiftreg_latch (shiftreg *sreg)
- void shiftreg_oe_hi (shiftreg *sreg)
- void shiftreg_oe_lo (shiftreg *sreg)
- void shiftreg_shift1 (shiftreg *sreg, bool bit)
- void shiftreg_shift8 (shiftreg *sreg, uint8_t byte)
- void shiftreg_shift16 (shiftreg *sreg, uint16_t data)
- void shiftreg_put1 (shiftreg *sreg, bool bit)
- void shiftreg_put8 (shiftreg *sreg, uint8_t byte)
- void shiftreg_put16 (shiftreg *sreg, uint16_t data)

4.5.1 Macro Definition Documentation

4.5.1.1 SR_DELAY_US

```
#define SR_DELAY_US 10
```

4.5.2 Typedef Documentation

4.5.2.1 shiftreg

```
typedef struct sn74hc595n shiftreg
```

4.5.3 Function Documentation

4.5.3.1 shiftreg_init()

```
void shiftreg_init (
            shiftreg * sreg,
            uint8_t ser,
            uint8_t rclk,
            uint8_t srclk,
            uint8_t oe)
```

4.5.3.2 shiftreg_latch()

```
void shiftreg_latch (
            shiftreg * sreg)
```

4.5.3.3 shiftreg_oe_hi()

```
void shiftreg_oe_hi (
            shiftreg * sreg)
```

4.5.3.4 shiftreg_oe_lo()

```
void shiftreg_oe_lo (
            shiftreg * sreg)
```

4.5.3.5 shiftreg_pulse_clock()

```
void shiftreg_pulse_clock (
            shiftreg * sreg)
```

4.5.3.6 shiftreg_put1()

```
void shiftreg_put1 (
            shiftreg * sreg,
            bool bit)
```

4.5.3.7 shiftreg_put16()

```
void shiftreg_put16 (
            shiftreg * sreg,
            uint16_t data)
```

4.5.3.8 shiftreg_put8()

```
void shiftreg_put8 (
            shiftreg * sreg,
            uint8_t byte)
```

4.5.3.9 shiftreg_select()

```
void shiftreg_select (
            shiftreg * sreg)
```

4.5.3.10 shiftreg_shift1()

```
void shiftreg_shift1 (
            shiftreg * sreg,
            bool bit)
```

4.5.3.11 shiftreg_shift16()

```
void shiftreg_shift16 (
            shiftreg * sreg,
            uint16_t data)
```

4.5.3.12 shiftreg_shift8()

```
void shiftreg_shift8 (
            shiftreg * sreg,
            uint8_t byte)
```

4.6 sn74hc595n.h

Go to the documentation of this file.

```
00001 #ifndef __drako_sn74hc595n
00002 #define __drako_sn74hc595n
00003
00004
00005 #include <stdint.h>
00006 #include <stdbool.h>
00007
00008
00009 #define SR_DELAY_US 10
00010
00011
00012 typedef struct sn74hc595n {
00013     uint8_t ser;
00014     uint8_t rclk;
00015     uint8_t srclk;
00016     uint8_t oe;
00017 } shiftreg;
00018
00019
00020 void shiftreg_init(shiftreg* sreg, uint8_t ser, uint8_t rclk, uint8_t srclk, uint8_t oe);
00021 void shiftreg_select(shiftreg* sreg);
00022 void shiftreg_pulse_clock(shiftreg* sreg);
00023 void shiftreg_latch(shiftreg* sreg);
00024
00025 void shiftreg_oe_hi(shiftreg* sreg);
00026 void shiftreg_oe_lo(shiftreg* sreg);
00027
00028 void shiftreg_shift1(shiftreg* sreg, bool bit);
00029 void shiftreg_shift8(shiftreg* sreg, uint8_t byte);
00030 void shiftreg_shift16(shiftreg* sreg, uint16_t data);
00031
00032 void shiftreg_put1(shiftreg* sreg, bool bit);
00033 void shiftreg_put8(shiftreg* sreg, uint8_t byte);
00034 void shiftreg_put16(shiftreg* sreg, uint16_t data);
00035
00036 #endif
```

4.7 include/drako/terminal.h File Reference

```
#include <stdio.h>
#include <stdbool.h>
#include <pico/stdlib.h>
#include <tusb.h>
```

Classes

- struct terminal_command_struct

Macros

- #define DRKO_TERM_BUFSIZE 256
- #define DRKO_TERM "[-BASILISK-]"

Typedefs

- typedef struct terminal_command_struct terminal_command

Functions

- void _terminal_clean_string (char *str, char *buf, size_t nbuf)
  Cleans string of leading and tailing spaces, tabs, and/or newline chars.
- void terminal_get_line (char *buf, size_t n)
  Reads in a line from terminal connection. Does not include return character.
- void terminal_get_command (terminal_command *tcmd)
  Gets command from terminal in argc/argv format.
- void terminal_command_free (terminal_command *tcmd)

4.7.1 Macro Definition Documentation

4.7.1.1 DRKO_TERM

```
#define DRKO_TERM "[-BASILISK-]"
```

4.7.1.2 DRKO_TERM_BUFSIZE

```
#define DRKO_TERM_BUFSIZE 256
```

4.7.2 Typedef Documentation

4.7.2.1 terminal_command

```
typedef struct terminal_command_struct terminal_command
```

4.7.3 Function Documentation

4.7.3.1 _terminal_clean_string()

```
void _terminal_clean_string (
            char * str,
            char * buf,
            size_t nbuf)
```

Cleans string of leading and tailing spaces, tabs, and/or newline chars.

Parameters

str	String to clean
buf	Buffer to store cleaned string
nbuf	Length of storage buffer

4.7.3.2 terminal_command_free()

```
void terminal_command_free (
            terminal_command * tcmd)
```

4.7.3.3 terminal_get_command()

```
void terminal_get_command (
            terminal_command * tcmd)
```

Gets command from terminal in argc/argv format.

Parameters

tcmd	Pointer to terminal_command buffer.

Note

Do not forget to free the terminal_command object after using.

4.7.3.4 terminal_get_line()

```
void terminal_get_line (
            char * buf,
            size_t n)
```

Reads in a line from terminal connection. Does not include return character.

Parameters

buf	String buffer to store read data in.
n	Length of provided buffer.

4.8 terminal.h

Go to the documentation of this file.

```
00001 #ifndef __drako_terminal
00002 #define __drako_terminal
00003
00004 #include <stdio.h>
00005 #include <stdbool.h>
00006
00007 #include <pico/stdlib.h>
00008 #include <tusb.h>
00009
00010
00011
00012
00013 #define DRKO_TERM_BUFSIZE 256
00014 #define DRKO_TERM "[-BASILISK-]"
00015
00016
00017
00018
00019 typedef struct terminal_command_struct {
00020     char** argv;
00021     size_t argc;
00022 } terminal_command;
00023
00024
00025
00026
00027 void _terminal_clean_string(char* str, char* buf, size_t nbuf);
00028
00029 void terminal_get_line(char* buf, size_t n);
00030 void terminal_get_command(terminal_command* tcmd);
00031 void terminal_command_free(terminal_command* tcmd);
00032
00033
00034
00035
00041 static inline bool _terminal_is_valid_char(char c) {
00042     return (c == ' ' || c == '\n' || c == '\t');
00043 }
00044
00045
```

```
00046
00047
00051 static inline void _terminal_greet() {
00052     printf(
00053         "|================================================================================|\n"
00054         "||------------------------------------------------------------------------------||\n"
00055         "||                                                                              ||\n"
00056         "||  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ||\n"
00057         "||  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ||\n"
00058         "||  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ||\n"
00059         "||  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ||\n"
00060         "||  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ||\n"
00061         "||  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ||\n"
00062         "||  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄  ▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄  ||\n"
00063         "||                                                                              ||\n"
00064         "||                                                                              ||\n"
00065         "||------------------------------------------------------------------------------||\n"
00066         "|================================================================================|\n"
00067     );
00068     printf("%s CONNECTION: ESTABLISHED\n", DRKO_TERM);
00069     printf("%s Welcome to DRAKO OS.\n", DRKO_TERM);
00070     printf("%s Type 'commands' to view list of valid commands.", DRKO_TERM);
00071 }
00072
00073
00074
00075
00079 static inline void terminal_open_connection() {
00080     // await connection
00081     while (!tud_cdc_connected()) {
00082         gpio_put(25, 1);
00083         sleep_ms(250);
00084         gpio_put(25, 0);
00085         sleep_ms(250);
00086     }
00087
00088     // display greeting
00089     _terminal_greet();
00090
00091     // display terminal promp
00092     printf("%s >$", DRKO_TERM);
00093 }
00094
00095
00096
00097
00102 static inline bool terminal_is_connected() {
00103     return tud_cdc_connected();
00104 }
00105
00106
00107
00108
00109 #endif
```

4.9 src/drako/display.c File Reference

```
#include "drako/display.h"
```

Functions

- void display_init (display *disp)
- void display_show (display *disp)
- void display_hide (display *disp)
- void display_clear (display *disp)
- void display_write (display *disp, uint16_t data)
- void display_hex (display *disp, uint8_t byte)
- uint16_t byte2disp (uint8_t data)

4.9.1 Function Documentation

4.9.1.1 byte2disp()

```
uint16_t byte2disp (
            uint8_t data)
```

4.9.1.2 display_clear()

```
void display_clear (
            display * disp)
```

4.9.1.3 display_hex()

```
void display_hex (
            display * disp,
            uint8_t byte)
```

4.9.1.4 display_hide()

```
void display_hide (
            display * disp)
```

4.9.1.5 display_init()

```
void display_init (
            display * disp)
```

4.9.1.6 display_show()

```
void display_show (
            display * disp)
```

4.9.1.7 display_write()

```
void display_write (
            display * disp,
            uint16_t data)
```

4.10 src/drako/hardware/at28c64b.c File Reference

```
#include "drako/hardware/at28c64b.h"
```

Functions

- void eeprom_init (eeprom *prom, uint32_t data_bus, uint32_t addr_bus, uint8_t we, uint8_t oe, uint8_t ce)
  Initialize a previously allocated EEPROM struct.
- void eeprom_select (eeprom *prom)
  Allows EEPROM to take control of shared GPIO pins.
- void eeprom_read8 (eeprom *prom, uint32_t addr, uint8_t *buff)
  Reads byte at specified address into buffer.
- void eeprom_write8 (eeprom *prom, uint32_t addr, uint8_t data)
  Write byte to EEPROM at specified address.

4.10.1 Function Documentation

4.10.1.1 eeprom_init()

```
void eeprom_init (
            eeprom * prom,
            uint32_t data_bus,
            uint32_t addr_bus,
            uint8_t we,
            uint8_t oe,
            uint8_t ce)
```

Initialize a previously allocated EEPROM struct.

Parameters

prom	Pointer to EEPROM struct
data_bus	GPIO mask of pins used for EEPROM I/O
addr_bus	GPIO mask of pins used for EEPROM addressing
we	Write Enable GPIO pin
oe	Output Enable GPIO pin
ce	Chip Enable GPIO pin

4.10.1.2 eeprom_read8()

```
void eeprom_read8 (
            eeprom * prom,
            uint32_t addr,
            uint8_t * buff)
```

Reads byte at specified address into buffer.

Parameters

prom	Pointer to EEPROM struct
addr	Address to read from
buff	Buffer for storing read data

4.10.1.3 eeprom_select()

```
void eeprom_select (
            eeprom * prom)
```

Allows EEPROM to take control of shared GPIO pins.

Parameters

prom	Pointer to EEPROM struct

4.10.1.4 eeprom_write8()

```
void eeprom_write8 (
            eeprom * prom,
            uint32_t addr,
            uint8_t data)
```

Write byte to EEPROM at specified address.

Parameters

prom	Pointer to EEPROM struct
addr	Address at which to write data
data	Byte data to write to address

4.11 src/drako/hardware/at28c64b_hidden.c File Reference

```
#include "drako/hardware/at28c64b.h"
```

Functions

- void _eeprom_data_in (eeprom ∗prom)
  Sets data bus to input mode.
- void _eeprom_data_out (eeprom ∗prom)
  Sets data bus to input mode.
- void _eeprom_gpio_init (eeprom ∗prom)
  Initialize GPIO pins used for EEPROM.
- void _eeprom_set_idle_condition (eeprom ∗prom)
  Sets all control pins to HIGH (inactive)
- void _eeprom_set_read_condition (eeprom ∗prom)
  Sets EEPROM read condition. Must be called before executing a READ.
- void _eeprom_set_write_condition (eeprom ∗prom)
  Sets EEPROM write condition. Must be called before executing a WRITE.
- void _eeprom_execute_write (eeprom ∗prom)
  Executes WRITE. Must be called AFTER setting a WRITE condition.

4.11.1 Function Documentation

4.11.1.1 _eeprom_data_in()

```
void _eeprom_data_in (
            eeprom * prom)
```

Sets data bus to input mode.

Parameters

prom	Pointer to EEPROM struct

4.11.1.2 _eeprom_data_out()

```
void _eeprom_data_out (
            eeprom * prom)
```

Sets data bus to input mode.

Parameters

prom	Pointer to EEPROM struct

4.11.1.3 _eeprom_execute_write()

```
void _eeprom_execute_write (
            eeprom * prom)
```

Executes WRITE. Must be called AFTER setting a WRITE condition.

Parameters

prom	Pointer to EEPROM struct

4.11.1.4 _eeprom_gpio_init()

```
void _eeprom_gpio_init (
            eeprom * prom)
```

Initialize GPIO pins used for EEPROM.

Parameters

prom	Pointer to EEPROM struct

4.11.1.5 _eeprom_set_idle_condition()

```
void _eeprom_set_idle_condition (
            eeprom * prom)
```

Sets all control pins to HIGH (inactive)

Parameters

prom	Pointer to EEPROM struct

4.11.1.6 _eeprom_set_read_condition()

```
void _eeprom_set_read_condition (
            eeprom * prom)
```

Sets EEPROM read condition. Must be called before executing a READ.

Parameters

prom	Pointer to EEPROM struct

4.11.1.7 _eeprom_set_write_condition()

```
void _eeprom_set_write_condition (
            eeprom * prom)
```

Sets EEPROM write condition. Must be called before executing a WRITE.

Parameters

prom	Pointer to EEPROM struct

4.12 src/drako/hardware/sn74hc595n.c File Reference

```
#include "drako/hardware/sn74hc595n.h"
#include <pico/stdlib.h>
```

Functions

- void shiftreg_init (shiftreg *sreg, uint8_t ser, uint8_t rclk, uint8_t srclk, uint8_t oe)
- void shiftreg_select (shiftreg *sreg)
- void shiftreg_pulse_clock (shiftreg *sreg)
- void shiftreg_latch (shiftreg *sreg)
- void shiftreg_oe_hi (shiftreg *sreg)
- void shiftreg_oe_lo (shiftreg *sreg)
- void shiftreg_shift1 (shiftreg *sreg, bool bit)
- void shiftreg_shift8 (shiftreg *sreg, uint8_t byte)
- void shiftreg_shift16 (shiftreg *sreg, uint16_t data)
- void shiftreg_put1 (shiftreg *sreg, bool bit)
- void shiftreg_put8 (shiftreg *sreg, uint8_t byte)
- void shiftreg_put16 (shiftreg *sreg, uint16_t data)

4.12.1 Function Documentation

4.12.1.1 shiftreg_init()

```
void shiftreg_init (
            shiftreg * sreg,
            uint8_t ser,
            uint8_t rclk,
            uint8_t srclk,
            uint8_t oe)
```

4.12.1.2 shiftreg_latch()

```
void shiftreg_latch (
            shiftreg * sreg)
```

4.12.1.3 shiftreg_oe_hi()

```
void shiftreg_oe_hi (
            shiftreg * sreg)
```

4.12.1.4 shiftreg_oe_lo()

```
void shiftreg_oe_lo (
            shiftreg * sreg)
```

4.12.1.5 shiftreg_pulse_clock()

```
void shiftreg_pulse_clock (
            shiftreg * sreg)
```

4.12.1.6 shiftreg_put1()

```
void shiftreg_put1 (
            shiftreg * sreg,
            bool bit)
```

4.12.1.7 shiftreg_put16()

```
void shiftreg_put16 (
            shiftreg * sreg,
            uint16_t data)
```

4.12.1.8 shiftreg_put8()

```
void shiftreg_put8 (
            shiftreg * sreg,
            uint8_t byte)
```

4.12.1.9 shiftreg_select()

```
void shiftreg_select (
            shiftreg * sreg)
```

4.12.1.10 shiftreg_shift1()

```
void shiftreg_shift1 (
            shiftreg * sreg,
            bool bit)
```

4.12.1.11 shiftreg_shift16()

```
void shiftreg_shift16 (
            shiftreg * sreg,
            uint16_t data)
```

4.12.1.12 shiftreg_shift8()

```
void shiftreg_shift8 (
            shiftreg * sreg,
            uint8_t byte)
```

4.13 src/drako/terminal.c File Reference

```
#include "drako/terminal.h"
#include <stdlib.h>
```

Functions

- void _terminal_clean_string (char *str, char *buf, size_t nbuf)

  Cleans string of leading and tailing spaces, tabs, and/or newline chars.
- void terminal_get_line (char *buf, size_t n)

  Reads in a line from terminal connection. Does not include return character.
- void terminal_get_command (terminal_command *tcmd)

  Gets command from terminal in argc/argv format.
- void terminal_command_free (terminal_command *tcmd)

4.13.1 Function Documentation

4.13.1.1 _terminal_clean_string()

```
void _terminal_clean_string (
            char * str,
            char * buf,
            size_t nbuf)
```

Cleans string of leading and tailing spaces, tabs, and/or newline chars.

Parameters

str	String to clean
buf	Buffer to store cleaned string
nbuf	Length of storage buffer

4.13.1.2 terminal_command_free()

```
void terminal_command_free (
            terminal_command * tcmd)
```

4.13.1.3 terminal_get_command()

```
void terminal_get_command (
            terminal_command * tcmd)
```

Gets command from terminal in argc/argv format.

Parameters

tcmd	Pointer to terminal_command buffer.

Note

Do not forget to free the terminal_command object after using.

4.13.1.4 terminal_get_line()

```
void terminal_get_line (
            char * buf,
            size_t n)
```

Reads in a line from terminal connection. Does not include return character.

Parameters

buf	String buffer to store read data in.
n	Length of provided buffer.

4.14 src/main.c File Reference

```
#include <pico/stdlib.h>
#include <pico/rand.h>
#include <stdio.h>
#include <tusb.h>
#include "drako/display.h"
#include "drako/hardware/at28c64b.h"
```

Functions

- void display_shell (display *disp)
- void eeprom_test (eeprom *prom)
- void full_test (eeprom *prom, display *disp)
- int main ()

4.14.1 Function Documentation

4.14.1.1 display_shell()

```
void display_shell (
            display * disp)
```

4.14.1.2 eeprom_test()

```
void eeprom_test (
            eeprom * prom)
```

4.14.1.3 full_test()

```
void full_test (
            eeprom * prom,
            display * disp)
```

4.14.1.4 main()

```
int main ()
```

Index