```
// ********************************************************************************** defines.h
// ***********************************************************************************************
#define ILI9341_CS   0
#define ILI9341_CD   1
#define ILI9341_WR   2
#define ILI9341_RD   3
#define ILI9341_RST  4
#define ILI9341_D0   5
#define ILI9341_MASK 0x1fff   // 0001 1111 1111 1111

#define ILI9341_TFTWIDTH 240  ///< ILI9341 max TFT width
#define ILI9341_TFTHEIGHT 320 ///< ILI9341 max TFT height
#define SWAP_BYTES(color) ((uint16_t)(color>>8) | (uint16_t)(color<<8))

#define ILI9341_NOP 0x00     ///< No-op register
#define ILI9341_SWRESET 0x01 ///< Software reset register
#define ILI9341_RDDID 0x04   ///< Read display identification information
#define ILI9341_RDDST 0x09   ///< Read Display Status

#define ILI9341_SLPIN 0x10  ///< Enter Sleep Mode
#define ILI9341_SLPOUT 0x11 ///< Sleep Out
#define ILI9341_PTLON 0x12  ///< Partial Mode ON
#define ILI9341_NORON 0x13  ///< Normal Display Mode ON

#define ILI9341_RDMODE 0x0A     ///< Read Display Power Mode
#define ILI9341_RDMADCTL 0x0B   ///< Read Display MADCTL
#define ILI9341_RDPIXFMT 0x0C   ///< Read Display Pixel Format
#define ILI9341_RDIMGFMT 0x0D   ///< Read Display Image Format
#define ILI9341_RDSELFDIAG 0x0F ///< Read Display Self-Diagnostic Result

#define ILI9341_INVOFF 0x20   ///< Display Inversion OFF
#define ILI9341_INVON 0x21    ///< Display Inversion ON
#define ILI9341_GAMMASET 0x26 ///< Gamma Set
#define ILI9341_DISPOFF 0x28  ///< Display OFF
#define ILI9341_DISPON 0x29   ///< Display ON

#define ILI9341_CASET 0x2A ///< Column Address Set
#define ILI9341_PASET 0x2B ///< Page Address Set
#define ILI9341_RAMWR 0x2C ///< Memory Write
#define ILI9341_RAMRD 0x2E ///< Memory Read

#define ILI9341_PTLAR 0x30    ///< Partial Area
#define ILI9341_VSCRDEF 0x33  ///< Vertical Scrolling Definition
#define ILI9341_MADCTL 0x36   ///< Memory Access Control
#define ILI9341_VSCRSADD 0x37 ///< Vertical Scrolling Start Address
#define ILI9341_PIXFMT 0x3A   ///< COLMOD: Pixel Format Set

#define ILI9341_FRMCTR1 0xB1 ///< Frame Rate Control (In Normal Mode/Full Colors)
#define ILI9341_FRMCTR2 0xB2 ///< Frame Rate Control (In Idle Mode/8 colors)
#define ILI9341_FRMCTR3 0xB3 ///< Frame Rate control (In Partial Mode/Full Colors)
#define ILI9341_INVCTR  0xB4 ///< Display Inversion Control
#define ILI9341_DFUNCTR 0xB6 ///< Display Function Control

#define ILI9341_PWCTR1 0xC0 ///< Power Control 1
#define ILI9341_PWCTR2 0xC1 ///< Power Control 2
#define ILI9341_PWCTR3 0xC2 ///< Power Control 3
#define ILI9341_PWCTR4 0xC3 ///< Power Control 4
#define ILI9341_PWCTR5 0xC4 ///< Power Control 5
#define ILI9341_VMCTR1 0xC5 ///< VCOM Control 1
#define ILI9341_VMCTR2 0xC7 ///< VCOM Control 2

#define ILI9341_RDID1 0xDA ///< Read ID 1
#define ILI9341_RDID2 0xDB ///< Read ID 2
#define ILI9341_RDID3 0xDC ///< Read ID 3
#define ILI9341_RDID4 0xDD ///< Read ID 4

#define ILI9341_GMCTRP1 0xE0 ///< Positive Gamma Correction
#define ILI9341_GMCTRN1 0xE1 ///< Negative Gamma Correction
//#define ILI9341_PWCTR6     0xFC

// ******************************************************** From Adafruit
#define ILI9341_SOFTRESET 0x01
#define ILI9341_SLEEPIN 0x10
#define ILI9341_SLEEPOUT 0x11
#define ILI9341_NORMALDISP 0x13
#define ILI9341_INVERTOFF 0x20
#define ILI9341_INVERTON 0x21
#define ILI9341_GAMMASET 0x26
#define ILI9341_DISPLAYOFF 0x28
#define ILI9341_DISPLAYON 0x29
#define ILI9341_COLADDRSET 0x2A
#define ILI9341_PAGEADDRSET 0x2B
#define ILI9341_MEMORYWRITE 0x2C
#define ILI9341_PIXELFORMAT 0x3A
#define ILI9341_FRAMECONTROL 0xB1
#define ILI9341_DISPLAYFUNC 0xB6
#define ILI9341_ENTRYMODE 0xB7
#define ILI9341_POWERCONTROL1 0xC0
#define ILI9341_POWERCONTROL2 0xC1
#define ILI9341_VCOMCONTROL1 0xC5
#define ILI9341_VCOMCONTROL2 0xC7
#define ILI9341_MEMCONTROL 0x36
#define ILI9341_MADCTL 0x36

#define ILI9341_MADCTL_MY 0x80
#define ILI9341_MADCTL_MX 0x40
#define ILI9341_MADCTL_MV 0x20
#define ILI9341_MADCTL_ML 0x10
#define ILI9341_MADCTL_RGB 0x00
#define ILI9341_MADCTL_BGR 0x08
#define ILI9341_MADCTL_MH 0x04
```

```
// Color definitions
#define ILI9341_BLACK 0x0000       ///<   0,   0,   0
#define ILI9341_NAVY 0x000F        ///<   0,   0, 123
#define ILI9341_DARKGREEN 0x03E0   ///<   0, 125,   0
#define ILI9341_DARKCYAN 0x03EF    ///<   0, 125, 123
#define ILI9341_MAROON 0x7800      ///< 123,   0,   0
#define ILI9341_PURPLE 0x780F      ///< 123,   0, 123
#define ILI9341_OLIVE 0x7BE0       ///< 123, 125,   0
#define ILI9341_LIGHTGREY 0xC618   ///< 198, 195, 198
#define ILI9341_DARKGREY 0x7BEF    ///< 123, 125, 123
#define ILI9341_BLUE 0x001F        ///<   0,   0, 255
#define ILI9341_GREEN 0x07E0       ///<   0, 255,   0
#define ILI9341_CYAN 0x07FF        ///<   0, 255, 255
#define ILI9341_RED 0xF800         ///< 255,   0,   0
#define ILI9341_MAGENTA 0xF81F     ///< 255,   0, 255
#define ILI9341_YELLOW 0xFFE0      ///< 255, 255,   0
#define ILI9341_WHITE 0xFFFF       ///< 255, 255, 255
#define ILI9341_ORANGE 0xFD20      ///< 255, 165,   0
#define ILI9341_GREENYELLOW 0xAFE5 ///< 173, 255,  41
#define ILI9341_PINK 0xFC18        ///< 255, 130, 198

// ******************************************************************************************** ILI9341.h
// ************************************************************************************************************
struct ILI9341 {
    ILI9341(int16_t w, int16_t h);

    void init();
    void set_command(uint8_t cmd);
    void command_param(uint8_t data);
    void write_data(void *buffer, int bytes);
    void write_data(const uint8_t *buffer, int bytes);
    void pin_reset();

    // ************************************************* Adafruit base
    void begin();

    // ************************************************* Defining the virtual functions

};

// ******************************************************************************************** ILI9341.cpp
static inline void init_pins() {
    gpio_init_mask(ILI9341_MASK);
    gpio_set_dir_out_masked(ILI9341_MASK);
    gpio_set_mask(ILI9341_MASK);
};

#define CS_ACTIVE  gpio_put(ILI9341_CS, 0)
#define CS_IDLE    gpio_put(ILI9341_CS, 1)
#define CD_COMMAND gpio_put(ILI9341_CD, 0)
#define CD_DATA    gpio_put(ILI9341_CD, 1)
#define WR_IDLE    gpio_put(ILI9341_WR, 1)
#define WR_STROBE  gpio_put(ILI9341_WR, 0); gpio_put(ILI9341_WR, 1)
#define RST_ACTIVE gpio_put(ILI9341_RST, 0)
#define RST_IDLE   gpio_put(ILI9341_RST, 1)

static inline void sio_write(const uint8_t *src, size_t len) {
    do {
        gpio_put_masked((0xff << ILI9341_D0), (*src << ILI9341_D0));
        WR_STROBE;

        len--;
        src++;
    } while (len > 0);
}

static inline void sio_write(void *src, size_t len) {
    char *x = (char *)src;
    do {
        gpio_put_masked((0xff << ILI9341_D0), (*x << ILI9341_D0));
        WR_STROBE;

        len--;
        x++;
    } while (len > 0);
}

// ************************************************************************************************************
ILI9341::ILI9341(int16_t w, int16_t h) {
}

void ILI9341::init() {
    init_pins();

    set_command(0x01); //soft reset
    sleep_ms(1000);

    set_command(ILI9341_GAMMASET);
    command_param(0x01);

/*
    // positive gamma correction
    set_command(ILI9341_GMCTRP1);
    write_data((const uint8_t[15]){ 0x0f, 0x31, 0x2b, 0x0c, 0x0e, 0x08, 0x4e, 0xf1, 0x37, 0x07, 0x10, 0x03, 0x0e, 0x09,
0x00 }, 15);

    // negative gamma correction
    set_command(ILI9341_GMCTRN1);
```

```
    write_data((const uint8_t[15]){ 0x00, 0x0e, 0x14, 0x03, 0x11, 0x07, 0x31, 0xc1, 0x48, 0x08, 0x0f, 0x0c, 0x31, 0x36,
0x0f }, 15);
*/
    // memory access control
    set_command(ILI9341_MADCTL);
    command_param(0x48);

    // pixel format
    set_command(ILI9341_PIXFMT);
    command_param(0x55);  // 16-bit

    // frame rate; default, 70 Hz
    set_command(ILI9341_FRMCTR1);
    command_param(0x00);
    command_param(0x1B);

    // exit sleep
    set_command(ILI9341_SLPOUT);

    // display on
    set_command(ILI9341_DISPON);

    // column address set
    set_command(ILI9341_CASET);
    command_param(0x00);
    command_param(0x00);  // start column
    command_param(0x00);
    command_param(0xef);  // end column -> 239

    // page address set
    set_command(ILI9341_PASET);
    command_param(0x00);
    command_param(0x00);  // start page
    command_param(0x01);
    command_param(0x3f);  // end page -> 319

    set_command(ILI9341_RAMWR);
};

void ILI9341::set_command(uint8_t cmd) {
    CS_ACTIVE;
    CD_COMMAND;
    sio_write(&cmd, 1);
    CD_DATA;
    CS_IDLE;
};

void ILI9341::command_param(uint8_t data) {
    CS_ACTIVE;
    sio_write(&data, 1);
    CS_IDLE;
};

void ILI9341::write_data(void *buffer, int bytes) {
    CS_ACTIVE;
    sio_write(buffer, bytes);
    CS_IDLE;
};

void ILI9341::write_data(const uint8_t *buffer, int bytes) {
    CS_ACTIVE;
    sio_write(buffer, bytes);
    CS_IDLE;
};

void ILI9341::pin_reset() {
    RST_ACTIVE;
    sleep_ms(1000);
    RST_IDLE;
};

// ********************************************************************************** Adafruit base
void ILI9341::begin() {
    init();
};

// ***********************************************
ILI9341 ili = ILI9341(ILI9341_TFTWIDTH, ILI9341_TFTHEIGHT);
```