Osnove mikroprocesorske elektronike

Vaja 10: LCD zaslon

Pri tej vaji bomo oživili LCD zaslon. *Uporabno funkcionalnost* LCD zaslona bomo dosegli tako, da bomo uspešno oživili sledeče pod-module:

1) *modul za osvetljevanje* LCD zaslona z zadnje strani (angl. LCD backlight),
2) modul za izris grafike na LCD zaslon *na osnovnem nivoju* preko vmesnika ILI9341,
3) *grafično knjižnico* `uGUI`, ki bistveno poenostavi izris grafike na zaslon,
4) *modul za detekcijo lokacije pritiska* na LCD zaslon (angl. LCD touch sreen functionality) na podlagi čipa XPT2046.

Vse te module bomo povezali znotraj modula `lcd.c`. Tekom vaje boste poskrbeli za dopolnitev implementacije modulov ter za specifikacijo vseh potrebnih parametrov, da posamezni pod-moduli lahko delujejo pravilno.

Naloge vaje – osvetlitev LCD zaslona

Vaša naloga je, da *dopolnite implementacijo* za krmiljenje osvetlitve LCD zaslona znotraj `lcd_backlight.c` modula:

1. ***definirajte podatkovni tip za "handle" strukturo*** `LCD_BKLT_handle_t`, ki bo hranila vse, kar je potrebno, da upravljamo osvetlitev LCD zaslona.

2. ***Na podlagi tega podatkovnega tipa definirajte "privatno" globalno strukturno spremenljivko*** `LCD_backlight`.

3. ***Implementirajte spodnje "javne" funkcije s sledečo vsebino:***

   a) `LCD_BKLT_init()` – funkcija, ki poskrbi za inicializacijo "handle" strukture za upravljanje LCD osvetlitve, za omogočitev delovanja časovnika, omogočitev delovanja *kanala* časovnika ter za določitev začetne vrednosti jakosti osvetlitve LCD zaslona ob inicializaciji sistema.

   b) `LCD_BKLT_get_brightness()` – funkcija, ki vrne trenutno nastavitev relativne jakosti osvetlitve LCD zaslona.

   c) `LCD_BKLT_set_brightness()` – funkcija, ki poskrbi za nastavitev relativne jakosti osvetlitve LCD zaslona.

   d) `LCD_BKLT_on()` – funkcija, ki prižge osvetlitev LED zaslona in nastavi jakost osvetlitve na specificirano privzeto vrednost, ki jo hranimo v "handle" strukturi.

   e) `LCD_BKLT_off()` – funkcija, ki ugasne osvetlitev LED zaslona.

4. ***Stestirajte in demonstrirajte delovanje modula za osvetljevanje LCD zaslona s pomočjo testne funkcije*** `LCD_BKLT_demo()`**.**

Naloge vaje – uporaba LCD zaslona z ILI9341

V sklopu priprave na vajo ste s pomočjo orodja CubeMX poskrbeli za *inicializacijo strojne opreme* mikrokrmilnika, ki je potrebna za komunikacijo z LCD zaslonom. Sedaj pa boste v programski kodi poskrbeli še, da se *definirajo vsi potrebni parametri, ki po potrebni za delovanje knjižnice* za delo z LCD zaslonom s krmilnikom ILI9341. Pri tem se boste spoznali s pogostim programerskim pristopom:

> C knjižnice so pogosto implementirane tako, da mora programer, ki želi knjižnico uporabiti, *definirati vse potrebne parametre* za delovanje knjižnice *s pomočjo makrojev v zglavni .h datoteki*.

Vaše naloge so sledeče:

5. ***V zglavni datoteki*** `lcd_ili9341.h` ***definirajte sledeče parametre:***

   a) *specificirajte širino paralelnega podatkovnega vodila*, ki je uporabljen za komunikacijo z LCD zaslonom (oziroma čipom ILI9341);

   b) *specificirajte resolucijo* LCD zaslona.

   Pri tem si pomagajte s spodnjim izsekom iz podatkovnega lista LCD zaslona.

3.General Specifications

Item	Dimension	Unit
Size	2.8”	
Dot Matrix	240 x RGB x 320(TFT)	dots
Module dimension	50.0(W) x 69.2(H) x 3.6(D)	mm
Active area	43.2 x 57.6	mm
Dot pitch	0.06 x 0.18	mm
LCD type	TFT, Normally White, Transmissive	
TFT Control IC	ILI9341 or equivalent	
TFT Interface	8-bit/16-bit CPU (build in controller)	
View Direction	6 o’clock	
Gray Scale Inversion Direction	12 o’clock	
Aspect Ratio	3:4	
Backlight Type	LED,Normally White	
With /Without TP	With RTP	
Surface	Glare	

6. ***V zglavni datoteki*** `lcd.h` ***definirajte sledeče parametre:***

   a) *specificirajte kateri pin mikrokrmilnika* se uporablja za implementacijo "LCD reset" signala.

   Pazite: ker za upravljanje GPIO digitalnih vhodov/izhodov *uporabljamo LL knjižnico* (in ne HAL knjižnice), je potrebno pri specifikaciji pina uporabiti makro iz *LL knjižnice* (podobno kot smo to storili pri vajah z LEDicami in tipkovnico). To pa pomeni, da je potrebno v `lcd` modul *vključiti še manjkajočo LL knjižnico za podporo za delo z GPIO vhodi/izhodi.*

```
// LCD modul združuje vse preostale module, ki implementirajo
// polno uporabnost LCD zaslona in jo nadgradijo z grafično knjižnico:
//
//      - podpora LL knjižnice za delo z GPIO digitalnimi vhodi/izhodi
//      - upravljanje LCD zaslona s pomočjo ILI9341 vmesnika (lcd_ili9341.h)
//      - upravljanje osvetlitve LCD zaslona (lcd_backlight.h)
//      - vključitev grafične knjižnice za delo z zaslonom (ugui.h)
//      - upravljanje s senzorjem dotika zaslona (angl. touch screen, XPT2046_touch.h)
//
#include
#include "lcd_ili9341.h"
#include "lcd_backlight.h"
#include "ugui.h"
#include "XPT2046_touch.h"
```

7. ***V glavnem programu poskrbite za ustrezno inicializacijo LCD zaslona.***

   Za inicializacijo uporabite ustrezno funkcijo `lcd.c` modula.

8. ***Preizkusite delovanje LCD zaslona s pomočjo že implementirane demo testne funkcije.***

   Za demonstracijo uporabite ustrezno funkcijo `lcd.c` modula.

Naloge vaje – grafična knjižnica `uGUI`

Da si *poenostavimo delo z izrisom grafike na LCD zaslon*, bomo uporabili odprto-kodno knjižnico `uGUI`, ki ste jo spoznali že tekom priprave na vajo. Knjižnico je potrebno prirediti platformi, na kateri se izvaja. In to boste storili na enak način kot v prejšnjem primeru: da definirate ustrezne makro parametre v zglavni datoteki.

4. ***V zglavni datoteki*** `ugui_config.h` ***definirajte sledeče parametre:***

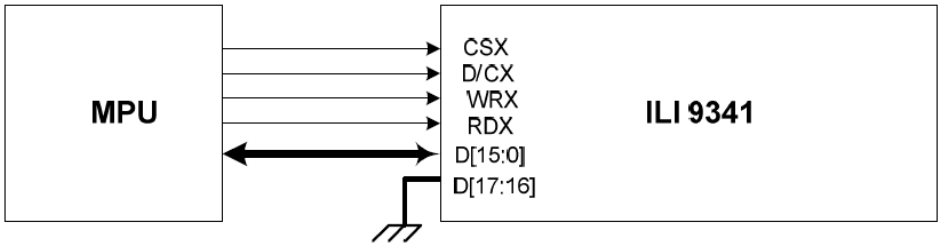
   a) specificirati je potrebno, kateri *barvni format* (angl. color pallete data format) naj se uporablja za *kodiranje informacije* o tem, kakšne barve naj bo posamezni piksel na LCD zaslonu.

   Zahteva: izbrati je potrebno tak barvni format, da bi celotno informacijo o barvi piksla lahko *prenesli v enem ciklu preko paralelnega podatkovnega vodila* do LCD zaslona, hkrati pa bi za to rešitev radi zagotovili največjo *barvno globino* (tj. največ različnih barv, angl. color depth).

   Informacijo, ki jo je potrebno poiskati, lahko najdete na spodnjem izseku iz podatkovnega lista.

8.2. 16-bit Parallel MCU Interface

The 8080- I system 16-bit parallel bus interface of ILI9341V can be selected by setting hardware pin IM[3:0] to"0001".The following shown figure is the example of interface with 8080- I MCU system interface.



Different display data format is available for two colors depth supported by listed below.
- 65K-Colors, RGB 5, 6, 5 -bits input data.
- 262K-Colors, RGB 6, 6, 6 -bits input data.

65K color: 16-bit/pixel (RGB 5-6-5 bits input)

One pixel (3 sub-pixels) display data is sent by 1 transfer when DBI [2:0] bits of 3Ah register are set to "101".

Count	0	1	2	3	…	238	239	240
D/CX	0	1	1	1	…	1	1	1
D15		0R4	1R4	2R4	…	237R4	238R4	239R4
D14		0R3	1R3	2R3	…	237R3	238R3	239R3
D13		0R2	1R2	2R2	…	237R2	238R2	239R2
D12		0R1	1R1	2R1	…	237R1	238R1	239R1
D11		0R0	1R0	2R0	…	237R0	238R0	239R0
D10		0G5	1G5	2G5	…	237G5	238G5	239G5
D9		0G4	1G4	2G4	…	237G4	238G4	239G4
D8		0G3	1G3	2G3	…	237G3	238G3	239G3
D7	C7	0G2	1G2	2G2	…	237G2	238G2	239G2
D6	C6	0G1	1G1	2G1	…	237G1	238G1	239G1
D5	C5	0G0	1G0	2G0	…	237G0	238G0	239G0
D4	C4	0B4	1B4	2B4	…	237B4	238B4	239B4
D3	C3	0B3	1B3	2B3	…	237B3	238B3	239B3
D2	C2	0B2	1B2	2B2	…	237B2	238B2	239B2
D1	C1	0B1	1B1	2B1	…	237B1	238B1	239B1
D0	C0	0B0	1B0	2B0	…	237B0	238B0	239B0

b) ***specificirati je potrebno, katere tipe pisave*** (v žargonu "fonte") bi želeli uporabljati za izpis besedila na LCD zaslon.

Vključite le tiste "fonte", ki so potrebni za delovanje sledečih funkcij:

- `LCD_uGUI_init()`,
- `LCD_uGUI_demo_Misko3()`.

Pojasnilo: `uGUI` knjižnica ponuja možnost izpisa besedila. Za ta namen ima definirane različne "fonte". Vsaka taka definicija "fonta", ki ga želite uporabljati, bo potrebovala svoj prostor v "flash" pomnilniku, ki pa ni zanemarljivo majehn. Iz tega sledi, da je smiselno varčevati na porabi "flash" pomnilnika in vključiti le tiste "fonte", ki jih resnično potrebujete za implementacijo sistema.

5. ***Preučite programsko kodo dveh temeljnih funkcij***, ki povežeta *visoko-nivojsko* grafično knjižnico `uGUI` z *nizko-nivojsko implementacijo izrisa grafike* na LCD zaslon.

   Ideja te povezave je sledeča: če za `uGUI` knjižnico implementiramo funkciji, ki znata narisati en sam piksel na zaslon oziroma en sam barvni pravokotnik na LCD zaslon, potem zna `uGUI` knjižnica s pomočjo teh dveh temeljnih funkcij na zaslon izrisati poljubno grafiko! Ideja je podobna kot pri implementaciji prirejene `printf()` funkcije, kjer je bilo potrebno implementirati temeljno funkcijo `_write()`.

```
// Implementacija funkcije za izris enega samega piksla na zaslon.
void UserPixelSetFunction(UG_S16 x, UG_S16 y, UG_COLOR c)
{
    ILI9341_SetDisplayWindow(x, y, 1, 1);
    ILI9341_SendData((LCD_IO_Data_t *)&c, 1);
}


// Implementacija funkcije za izris pravokotnika na zaslon.
UG_RESULT _HW_FillFrame_(UG_S16 x, UG_S16 y, UG_S16 w, UG_S16 h, UG_COLOR c)
{
    LCD_FillRect(x, y, w, h, c);

    return UG_RESULT_OK;
}
```

Tekom inicializacije knjižnice `uGUI` "knjižnici `uGUI` povemo", da naj uporablja ti dve temeljni funkciji. Poglejte kodo spodaj.

```
// Inicializacija uGUI knjižnice za delo z našim LCD zaslonom.
void LCD_uGUI_init(void)
{
    // Inicializacija uGUI knjižnice: registracija funkcije za izris enega piksla na zaslon,
    // specifikacija resolucije zaslona.
    UG_Init(&gui, UserPixelSetFunction, ILI9341_GetParam(LCD_WIDTH), ILI9341_GetParam(LCD_HEIGHT));

    // Nastavitev "default" fontov in barv za besedilo in ozadje.
    UG_FontSelect(&FONT_8X12);
    UG_SetForecolor(C_WHITE);
    UG_SetBackcolor(C_BLACK);

    // Registracija funkcij za izris pravokotnika.
    UG_DriverRegister(DRIVER_FILL_FRAME, (void *)_HW_FillFrame_);
    UG_DriverEnable(DRIVER_FILL_FRAME);
}
```

6. ***V glavnem programu poskrbite za ustrezno inicializacijo*** `uGUI` ***knjižnice za delo za našim LCD zaslonom.***

   Za inicializacijo uporabite ustrezno funkcijo `lcd.c` modula.

7. ***Preizkusite delovanje*** `uGUI` ***knjižnice zaslona s pomočjo že implementirane demo testne funkcije.***

   Za demonstracijo uporabite ustrezno funkcijo `lcd.c` modula.

Naloge vaje – modul za detekcijo pritiska na zaslon

Modul za detekcijo pritiska na LCD zaslon boste oživili na podoben način kot prejšnje LCD pod-module: v zglavni datoteki `XPT2046_touch.h` boste definirali ustrezne makro parametre.

8. ***V zglavni datoteki*** `XPT2046_touch.h` ***definirajte sledeče parametre:***

   a) kateri pin se uporablja za signal "IRQ pending",
   b) kateri pin se uporablja za signal "chip select",
   c) število meritev lokacije pritiska, ki naj se uporabi za izračun povprečne lokacije pritiska,
   d) v kateri orientaciji se uporablja LCD zaslon,
   e) kakšna je ločljivost LCD zaslona.

9. ***Modul za detekcijo pritiska preizkusite s pomočjo ustrezne demo testne funkcije.***

   Pojasnilo: modul za detekcijo pritiska na zaslon ne potrebuje posebne inicializacije. Modul je praktično pripravljen na uporabo takoj, ko se ustrezno inicializira serijski vmesnik SPI za komunikacijo s čipom XPT2046. Za to pa ste poskrbeli s pomočjo orodja CubeMX. Vse kar je potrebno nato še storiti, da modul za detekcijo deluje, je le še specificirati, kateri SPI vmesnik se uporablja za komunikacijo. To pa je že storjeno v zglavni datoteki (glejte spodaj) s pomočjo pred-definiranega HAL makroja.

```
// -------------------- User-defined parameters ---------------------
//
// POZOR: za uspešno uporabo modula je potrebno predhodno poskrbeti
// za ustrezno nizko-nivojsko inicializacijo digitalnih GPIO vhodov in
// izhodov ter SPI vmesnika za komunikacijo s čipom XPT2046.
// Za to v našem primeru poskrbi CubeMX z avtomatsko generirano kodo.


// Potrebno je vključiti podporo za delo s HAL knjižnico. Osnovna knjižnjica
// je bila namreč predelana tako, da predvideva implementacijo s pomočjo
// HAL knjižnice za komunikacijo preko serijskega vmesnika SPI.
#include "stm32g4xx_hal.h"


// Warning! Use SPI bus with < 2.5 Mbit speed, better ~650 Kbit to be save.
#define XPT2046_SPI_PORT        hspi1
```