Univerzitet u Nišu
Elektronski Fakultet

Mbed razvojno okruženje i grafički displej sa ILI9326 kontrolerom

Mentor:
prof. Dr Branislav Petrović

Student:
Nenad Radulović (13152)

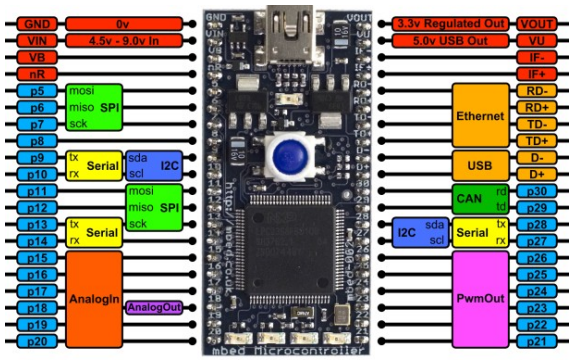
Sadržaj

Index slika

1. Uvod

1.1. mbed mikrokontroler modul

Mbed je serija razvojnih ploča bazirane na ARM mikrokontrolerima, koji su dizajnirani za brzo projektovanje.



Slika 1: mbed mikrokontroler modul

Mbed NXP LPC1768 mikrokontroler modul je projektovan za izradu prototipova raznih uređaja, naročito onih koji uključuju Ethernet, USB i fleksibilnost brojnih perifernih interfejsa i FLASH memoriju. Proizveden je u malom DIP pakovanju za upotrebu sa raznim proto PCB pločama i postoji ugrađeni USB FLASH programator/debager.

Zasnovan je na NXP LPC1768 mikrokontroleru, sa 32-bitnim ARM Cortex-M3 jezrom koje radi na 96MHz. Sadrži 512KB FLASH memorije, 64KB RAM i mnoštvo interfejsa uključujući i ugrađeni Ethernet, USB, CAN, SPI, I2C, ADC, DAC, PWM i druge U/I interfejse. Slika iznad prikazuje često korišćene interfejse i njihovu lokaciju.

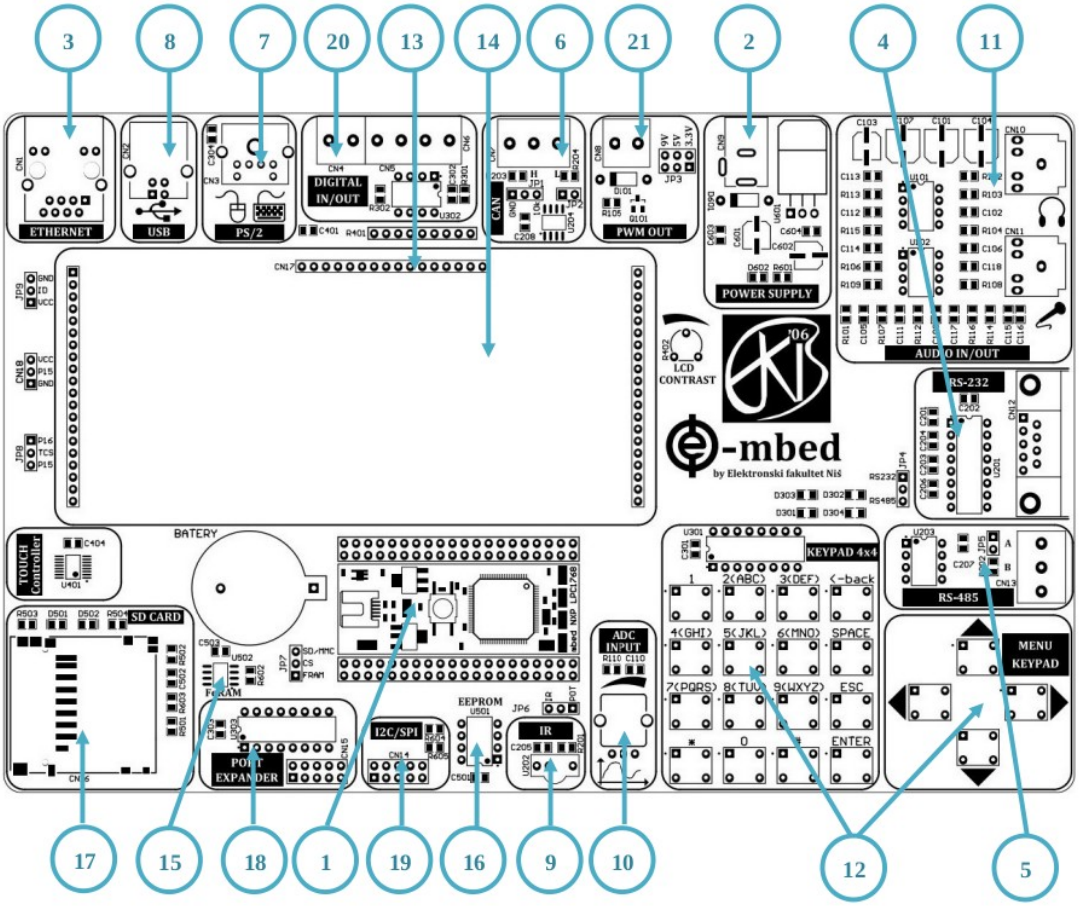
Mbed mikrokontroleri pružaju iskusnim embedded programerima moćnu i produktivnu platformu za izradu prototipova. Za programere neiskusne u oblasti 32-bitnih mikrokontrolera, mbed pruža pristupačno rešenje za izradu projekta uz raspoložive biblioteke, resurse i podrške koju pruža mbed zajednica.

Karakteristike:

- NXP LPC1768 MCU
- ARM® Cortex™-M3 jezgro visokih performansi
- 96MHz, 32KB RAM, 512KB FLASH
- Ethernet, USB Host/Device, 2xSPI, 2xI2C, 3xUART, CAN, 6xPWM, 6xADC, GPIO
- 40-pinsko DIP kućište, 54x26mm
- 5V USB ili 4.5-9V napajanje
- Ugrađen USB drag 'n' drop FLASH programator
- Online Kompajler
- C/C++ razvojno okruženje na visokom nivou

1.2. ELFAK-mbed board

Razvojni sistem ESRG predstavlja razvojni alat pogodan za programiranje i eksperimentisanje sa mbed mikrokontrolerom kompanije NXP. Brojni moduli, među kojima su TFT displej osetljiv na dodir, 2x16 alfanumerički LCD, tastatura, koji se nalaze na razvojnom sistemu omogućavaju jednostavnu i brzu simulaciju rada finalnog uređaja.



Slika 2: Struktura ELFAK-mbed razvojne ploče

1. mbed mikrokontroler
2. napajanje
3. Ethernet konektor
4. RS-232 konektor
5. RS-485 konektor
6. CAN konektor
7. PS/2 konektor
8. USB konektor
9. IR prijemnik
10. Analogni ulaz
11. Audio In/Out
12. Tastatura
13. LCD alfanumerički displej 2x16
14. TFT displej
15. Serijski RAM
16. Serijski EEPROM
17. SD/MMC
18. Port ekspander
19. Digital In/Out
20. PWM out

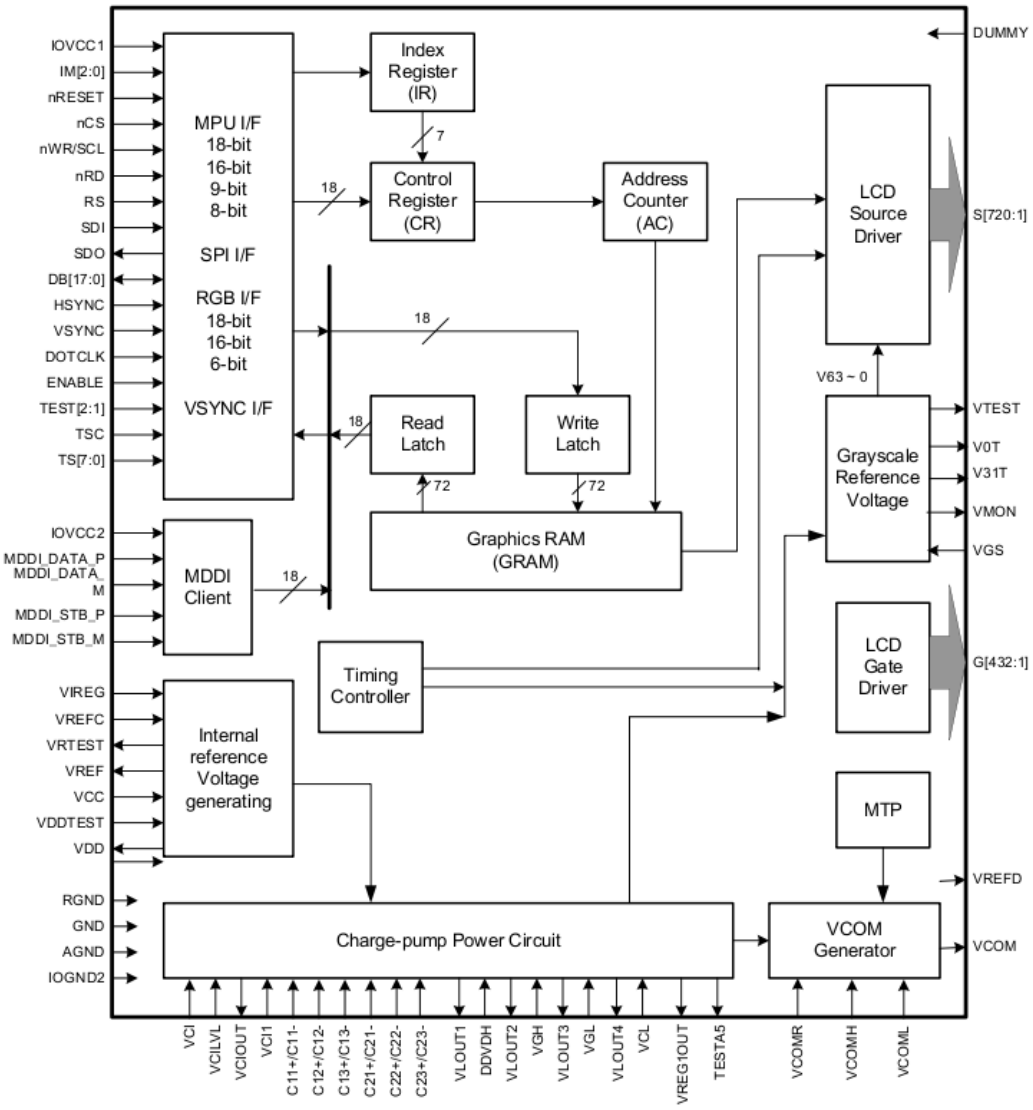
1.3. ILI9326 grafički kontroler

ILI9326 je SoC drajver za TFT LCD sa 262.144 boja i rezolucijom do 240x432 pixela. ILI9326 poseduje 720 kanala source drajvera i 432 kanala gate drajvera, RAM za prikaz slike do rezolucije 240x432 i sva potrebna kola za generisanje visokih napona. Podržava četiri fizička interfejsa:

- i80 MPU sistem (širina magistrale 8/9/16/18 bita)
- VSYNC interfejs
- SPI interfejs
- RGB 6/16/18 interfejs

Kombinacija brzog upisa u RAM sa RGB ili VSYNC interfejsom omogućava da se preko funkcija prozora prikaže dinamička slika na željenoj lokaciji, a statična slika u ostalim delovima ekrana.

ILI9326 može da radi sa 1,65V interfejsom, podržava štedljiv režim rada sa prikazom do 8 boja i režime za smanjenje potrošnje (sleep mode).



Slika 3: Blok dijagram ILI9326 kontrolera

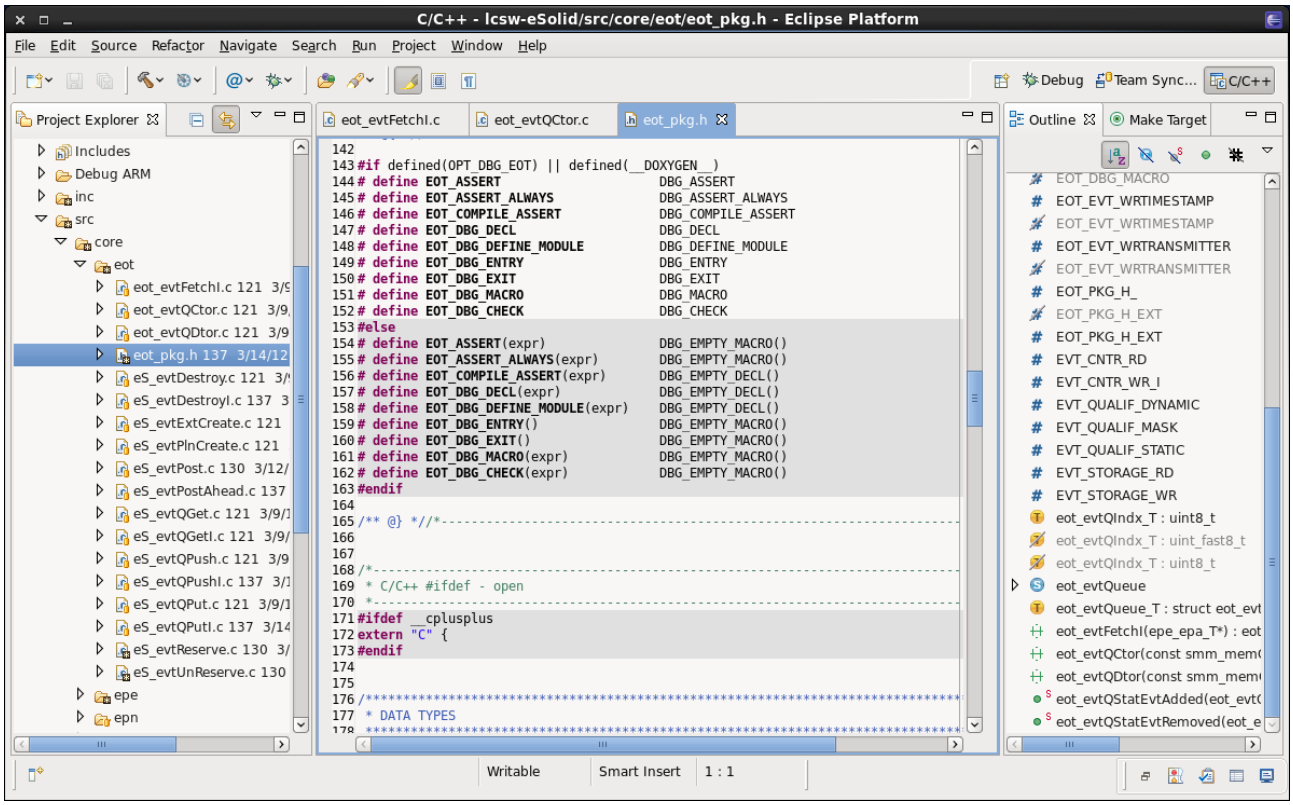
Karakteristike:

- jedan čip za LCD TFT displej
- 240x432 pixela sa 262.144 boja
- poseduje 720 kanala source drajvera i 432 kanala gate drajvera
- interna memorija sa kapacitetom 233.288 B
- burst upis u RAM velike brzine
- raznolik sistemski interfejs
- interni oscilator i hardverski reset monitor
- reverzibilni smer pomeranja source/gate drajvera
- funkcija prozora
- gamma korekcija boja
- sadrži sva neophodna step-up kola za generisanje napona za rad TFT LCD displeja
- funkcije za smanjenje potrošnje
- generisanje pogonskih napona
    - Source/VCOM napon
        - DVDH – GND = 4,5V ~ 6,0V
        - VCL – GND = -2.0V ~ -3,0V
        - VCI – VCL ≤ 6,0V
    - Napon na gate drajveru
        - VGH + GND = 10V ~ 16V
        - VGL – GND = -5V ~ -15V
        - VGH – VGL ≤ 32V
    - Napon na VCOM drajveru
        - VCOMH = 3,0V ~ (DVDH – 0,5)V
        - VCOML = (VCL + 0,5)V ~ 0V
        - VCOMH – VCOML ≤ 6,0V
- Kondezator u step-up kolu: samo Cst

1.4. Eclipse razvojno okruženje

Eclipse je open source zajednica, čiji su projekti namenjeni za izgradnju otvorene razvojne platforme koja se sastoji od velikog broja frameworka i dodataka što nudi veliku mogućnost proširenja funkcionalnosti, od alatki za kreiranje izvršnih datoteka i ostalih alatki koje se koriste u životnom ciklusu softvera. Eclipse fondacija je ne-profitabilna korporacija koja je podržana od strane članova koji učestvuju u razvoju open source zajednice i čitavog ekosistema koji pruža dodatne usluge i proizvode bazirani na Eclipse proizvodima.

Eclipse projekat je bio kreiran od strane IBM-a 2001 godine i podržan je bio od strane konzorcijuma softverskih proizvođača. Eclipse fondacija je kreirana početkom 2004 kao nezavisna, ne-profitabilna korporacija koja se ponaša kao upravnik Eclipse open source zajednice.



Slika 4: Izgled Eclipse razvojnog okruženja

Funkcionalnost razvojnog okruženja se proširuje instaliranjem odgovarajućih dodataka. Samo po sebi Eclipse ne nudi napredne mogućnosti za razvoj C/C++ aplikacija, te je stoga potrebno instalirati nekoliko potrebnih dodataka. Konfiguracija je minimalna, a dobija se moćna i vrlo fleksibilna alatka za razvoj.

1.4.1. Instalacija kompletnog razvojnog okruženja

Za početak je potrebno instalirati sledeće:

- Java Runtime Environment

  Eclipse je java aplikacija i za rad je potreban java izvršno okruženje.

- Eclipse

  Koristimo CDT projekat, a on je dodatak za Eclipse, zbog toga nam je potreban Eclipse. U ovom projektu je korišćena verzija 3.6.2 poznata pod imenom *Helios*. Eclipse je dostupan na adresi: *http://www.eclipse.org/downloads/*

- Eclipse C/C++ razvojne alatke (CDT projekat)

  CDT je dodatak za Eclipse koji ga transformiše u moćno okruženje za razvoj C/C++ aplikacija. Korišćena je verzija 7.0.2. Dodatak je dostupan na adresi: *http://www.eclipse.org/cdt/downloads.php*

- GNU C/C++ razvojne alatke

  CDT projekat koristi standardni GNU C/C++ alatke za kompajliranje koda, kreiranje projekata i debagiranje aplikacija. U ove alatke spadaju GNU Compiler Collection (GCC), make i GNU Project Debugger (GDB). Postoje nekoliko varijanti ovih alatki.

- GNU ARM dodatak

  Za rad sa ARM kompajlerima potrebno je instalirati ovaj dodatak koji proširuje funkcionalnost CDT projekta. Dodatak je dostupan na adresi: *http://sourceforge.net/projects/gnuarmeclipse/*

CDT projekat pruža potpuno funkcionalno C/C++ integrisano razvojno okruženje koje je bazirano na Eclipse platformi. CDT projekat sadrži sledeće:

- podrška za kreiranje projekta i organizovani build sistem za razne kompajlere,
- standardni make build sistem,
- navigacija izvornog koda,
- razne alatke za statičnu analizu koda u koje spadaju generisanje prikaza za pozive funkcija, pregledač uključivanja datoteka, pregledač makroa, prikaz hijerarhije tipova podataka, kod editor sa naglašavanjem sintakse, refaktor koda,
- vizuelne alatke za debagiranje sa prikazom memorije, registara i instrukcija.

1.4.1.1. Instalacija GNU ARM dodatka

- Korak 1:

  Kliknuti na "*Help*" -> "*Install New Software...*"

- Korak 2:

  Pojavljuje se nov prozor "*Install*". Kliknuti na "*Add...*", a zatim na "*Archive...*". U novom prozoru izabrati prethodno skinut GNU ARM Eclipse Plug-in, klikom na "*Open*", a zatim na "*OK*".

- Korak 3:

  U prozoru "*Install*" se u listi pojavila stavka "*CDT GNU Cross Development Tools*". Potrebno je štiklirati ovu stavku i kliknuti na "*Next >*".

- Korak 4:

  Potrebno je kliknuti ponovo na "*Next >*", zatim potvrditi da se slažete sa licencom i kliknuti na "*Finish*". Eclipse će započeti sa skidanjem potrebnih paketa sa interneta. U slučaju da se pojavi "*Security Warning*" klikom na "*OK*" nastavlja se sa instalacijom. Nakon skidanja i instalacije paketa, Eclipse će ponuditi opciju restartovanja. Klikom na "*Restart Now*" Eclipse će se restartovati.

1.4.1.2. Instalacija kompajler toolchain-a

Postoji nekoliko aktuelnih kompajlera. Neki su toolchain-ovi su komercijalni, dok postoje i sasvim funkcionalne i dobro podržane open source varijante.

Varijante toolchain-a

- *Potpuna open source varijanta* - podrazumeva dobavljanje izvornog koda za sve potrebne alatke i njihovo kompajliranje. Ova solucija nudi najviše fleksibilnosti jer se programeru ostavlja mogućnost da izmeni i prilagodi sistemske biblioteke i alatke prema svom nahođenju. Međutim, ova metoda iziskuje dosta potrebnog znanja o alatkama i vremena za njihovu konfiguraciju i kompajliranje. Da bi se olakšao pomenuti problem na adresi: *https://github.com/esden/summon-arm-toolchain* je dostupna skripta koja automatski dobavlja izvorni kod za razvojne alatke, vrši njihovu konfiguraciju, a zatim i njihovo kompajliranje.
- *Kompajlirane open-source varijante* - postoje kompajlirane open-source varijante koje su sa stanovišta korisnika mnogo lakše za upotrebu nego potpuna open source varijanta. Najpoznatije varijante su:
  - *Yagarto* toolchain – razvijana od strane zajednice, dostupna samo za Windows platformu
  - *CodeSourcery Lite* – open source varijanta kompanije Mentor Graphics, dostupana za Linux i Windows platformu.

U ovom projektu je korišćen *CodeSourcery Lite* toolchain. Instalacija je sasvim jednostavna. Bitno je napomenuti da se za putanju instalacije MORA izabrati putanja koja u sebi nema SPACE karaktere.

1.4.2. Konfiguracija razvojnog okruženja

1.4.2.1. Dodavanje toolchain-a u putanju

Napomena:

Ovo podešavanje je vezano za *workspace*. Ukoliko se *workspace* promeni, onda se mora ponovo izvršiti ova procedura.

- Korak 1:

  Sada je potrebno ubaciti promenljive koje govore Eclipse-u gde da potraži alatke i biblioteke. To se vrši klikom na "*Window*" -> "*Preferences*". U novom prozoru se bira "*C/C++*" -> "*Build*" -> "*Environment*".

- Korak 2:

  U Environment sekciji kliknuti na dugme "*Select...*". U novom prozoru treba štiklirati promenljivu *Path*.

Podešavanja za Windows operativni sistem

Proveriti da li su instalacione putanje za dati toolchain navedene. Ukoliko nisu, dodati ih. Putanje se međusobno razdvajaju ";" karakterom.

Primer za Yagarto toolchain:

```
C:\Program Files\CollabNet\Subversion Client; C:\WINDOWS\system32; C:\WINDOWS; C:\yagarto\bin;
C:\yagarto-tools-20100703\bin
```

Podešavanja za Linux operativni sistem

Dodati putanju toolchain-a u PATH promenljivu. Dovoljno je napisati:

```
${PATH}:[putanja/ka/toolchain-u]
```

1.4.2.2. Integracija Eclipse sa Doxygen-om

Doxygen je dokumentacioni sistem za C, C++, Java, Objective C, Python, Fortran, VHDL, PHP i druge programske jezike. On pomaže na sledeće načine:

- generiše on-line dokumentacioni pregledač (HTML) ili off-line referentno uputstvo (LATEX) iz samih izvornih datoteka. Takođe, postoji i podrška za generisanje dokumentacije u drugim formatima, kao što su RTF (Ms Word), PostScript, PDF, Unix man stranice. Dokumentacija se izvlači direktno iz izvornih datoteka čime se postiže laka sinhronizacija između dokumentacije i izvornog koda.
- Doxygen se može konfigurisati da kreira dokumentaciju za izvorne datoke koje nisu interno dokumentovane. Ovo može da pomogne da se lakše upozna veliki softverski projekat. Prilikom kreiranja dokumentacije vrši se generisanje određenih grafika koji prikazuju međusobnu zavisnost funkcija, podataka, tipova podataka i slično.

Doxygen se razvija za Linux i Mac OS X sisteme, ali se može pokrenuti sa velikim uspehom i pod Windows sistemima.

Napomena:

- Ovo podešavanje je vezano za *workspace*. Ukoliko se *workspace* promeni, onda se mora ponovo izvršiti ova procedura.

  Potrebno je instalirati *eclox* plugin, koji se nalazi na sledećoj adresi:

  http://home.gna.org/eclox

1.4.2.3. Podešavanje šablona

Šabloni omogućavaju konzistentan izgled izvornog koda.

Napomena:

- Ovo podešavanje je vezano za *workspace*. Ukoliko se *workspace* promeni, onda se mora ponovo izvršiti ova procedura.

Treba učitati podrazumevane šablone za izvorne datoteke. To se vrši odabirom: “*Window*“ -> “*Preferences*“ -> “*C/C++*“ -> “*Code Style*“ -> “*Code Template*“. U novom prozoru kliknuti na “*Import...*“ i izabrati datoteke šablona sa fajl sistema.

1.4.2.4. Podešavanje stila kodiranja

Stil kodiranja omogućava lakše formatiranje C koda i konzistentan izgled što doprinosi preglednosti koda, a samim tim i njegovo lakše razumevanje.

Napomene:

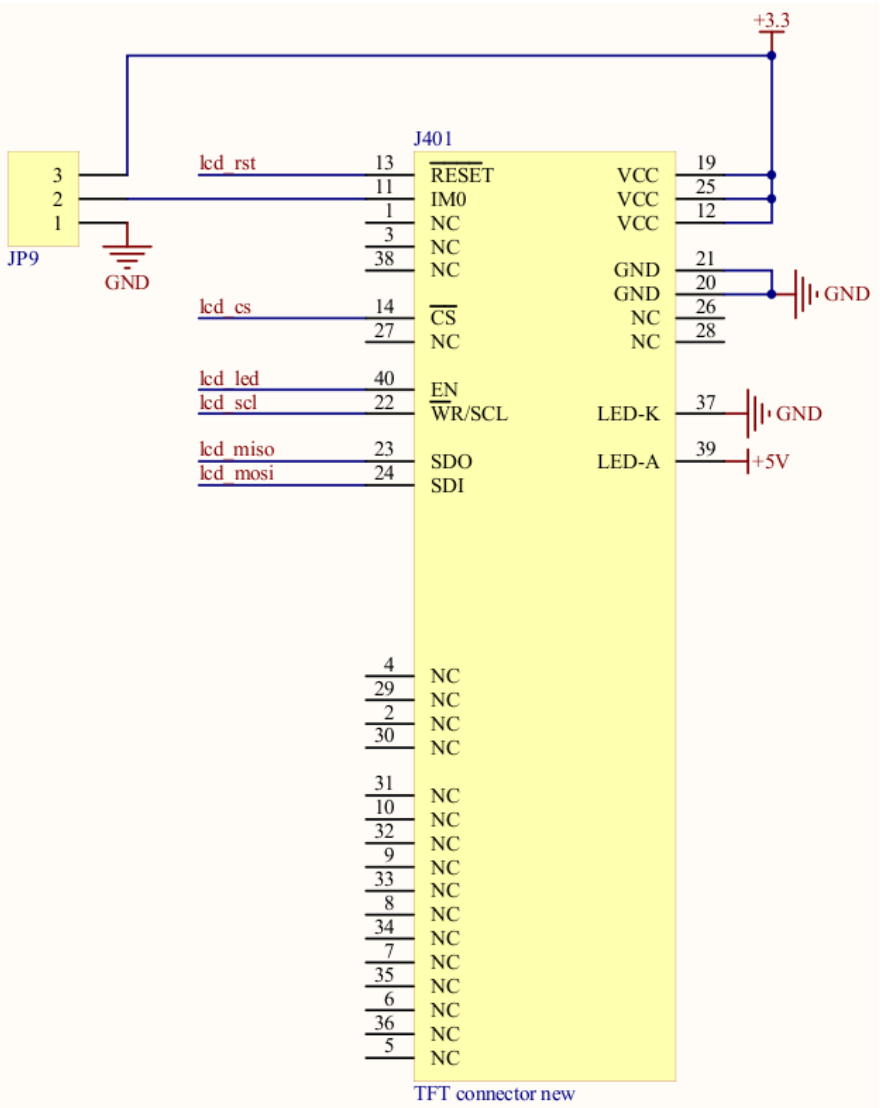
- Ovo podešavanje je vezano za *workspace*. Ukoliko se *workspace* promeni, onda se mora ponovo izvršiti ova procedura.
- Stil kodiranja se zasniva na D0020 dokumentu i uvek prati najnoviju verziju dokumenta.

Treba učitati podrazumevani šablon stila kodiranja za izvorne datoteke. To se vrši odabirom: “*Window*“ -> “*Preferences*“ -> “*C/C++*“ -> “*Code Style*“. U novom prozoru kliknuti na “*Import...*“ i izabrati datoteke šablona stila u fajl sistemu.

2. Hardver

2.1. mbed – TFT kontroler interfejs

TFT displej rezolucije 240x432 se koristi za prikaz grafičkih sadržaja. Drajver i kontroler za displej ILI9326 se povezuje sa mbed mikrokontrolerom preko SPI magistrale. Kontrolni signal *lcd_cs* se dovodi na ILI9326 kontroler preko kratkospajača JP8 koji mora biti u gornjem položaju.



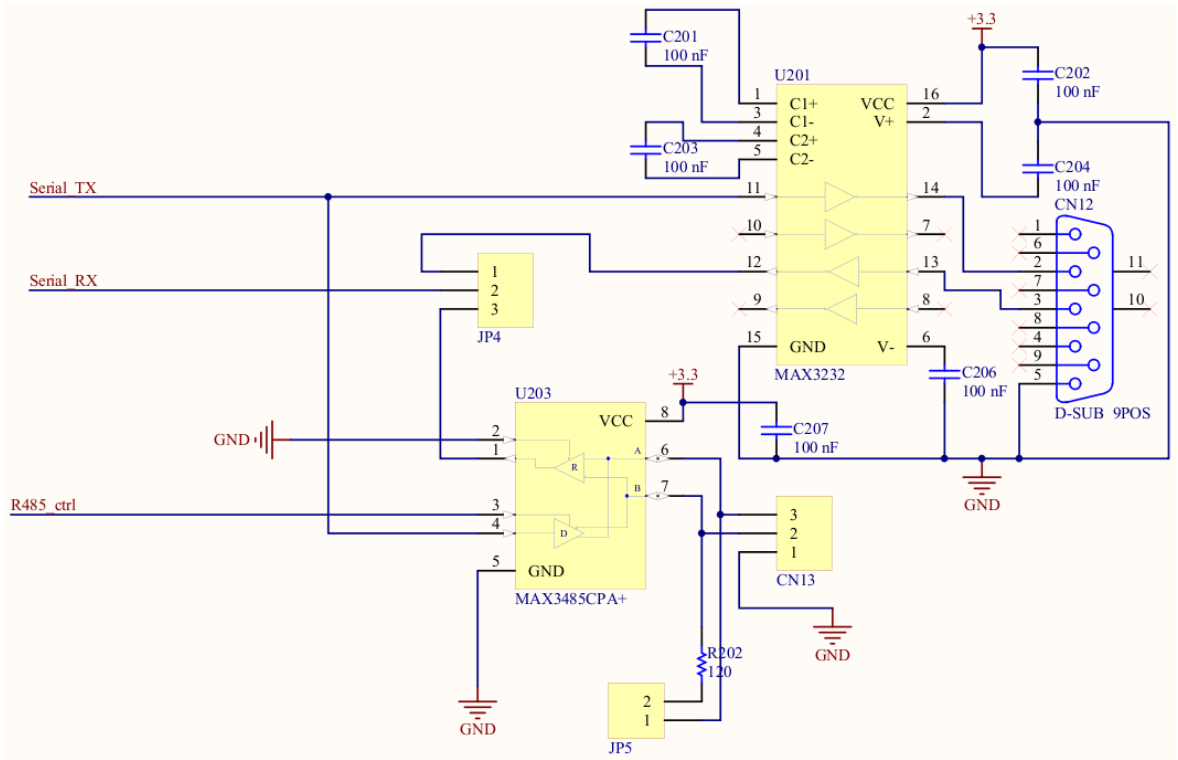
Slika 5: Konektor za TFT displej sa kontrolerom

Kratkospajač JP9 se koristi za postavljanje vrednosti ID na pin displej kontrolera. U gornjem položaju ID = „1“, a u donjem je ID = „0“.

Mbed mikrokontroler pored SPI kontrolnih signala vrši i kontrolu osvetljenja TFT displeja preko kontrolnog signala *lcd_led*. Signal *lcd_rst* se koristi za dovođenje ILI9326 kontrolera u inicijalno stanje.

2.2. RS232 interfejs

Radi potrebe testiranja ispravnosti rada ILI9326 drajvera i aplikacije za prikaz podataka, koristio se RS232 interfejs sa PC računarom kako bi se simulirao rad nekog akvizicionog sistema koji generiše podatke koje treba prikazati na displeju. RS232 komunikacija se obavlja preko 9-pinskog SUB-D konektora i MAX232 kola, koje prilagođava naponske nivoe. Signali *Tx* i *Rx* se nalaze na pinovima 28 i 27, respektivno. Da bi RS232 komunikacija bila moguća potrebno je da se kratkospajač JP4 postavi u gornji položaj.



Slika 6: Šema RS232 interfejsa

2.3. ILI9326 grafički kontroler i LCD displej

ILI9326 poseduje sistemski interfejs za čitanje/upis kontrolnih registara i grafičke memorije (GRAM) i RGB ulaze za prikaz pokretne slike. Korisnik je u mogućnosti da odabere optimalan interfejs za prikaz statične i/ili pokretne slike. Svi podaci se skladište u interni RAM kako bi se smanjila nepotrebna komunikacija i ažurirarali samo delovi slike koji se menjaju. Korisnik je, takođe, u mogućnosti da odabere deo slike ekrana, poznat pod imenom prozor (*window*), i vrši ažuriranje slike samo u tom delu displeja.

2.3.1. Interfejsi ILI9326 kontrolera

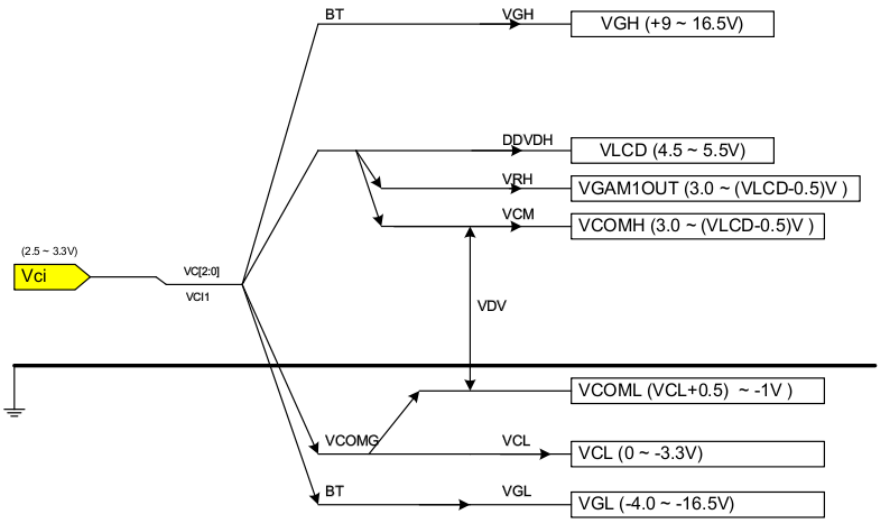
Sistemski interfejsi koje ILI9326 kontroler podržava su sledeći:

IM2	IM1	IM0/ID	Tip interfejsa	DB pin
0	0	0	i80 – sistem, 18-bita	DB[17:0]
0	0	1	i80 – sistem, 9-bita	DB[17:9]
0	1	0	i80 – sistem, 16-bita	DB[17:10], DB[8:1]
0	1	1	i80 – sistem, 8-bita	DB[17:10]
1	0	ID	Serial Peripheral Interface (SPI)	SDI, SDO
1	1	*	Nevalidno podešavanje	
1	1	1	MDDI interfejs	

Mbed mikrokontroler razvojni sistem podržava isključivo SPI interfejs, tako da su konfiguracioni pinovi ILI9326 kontrolera fiksirani na 1,0,x konfiguraciju, gde se LSB bit podešava kratkospajačem JP9.

2.3.2. Naponski nivoi

Za potrebe rada LCD displeja potrebni su različiti naponi. Kontroler ILI9326 generiše različite napone prema sledećoj šemi:



Slika 7: Generisanje napona za LCD displej

Naponi DDVDH, VGH, VGL, i VCL su niži od idealnih naponskih nivoa zbog struja potrošača na krajevima naponskih regulatora. Kontroler omogućava da se za generisanje napona koristi eksterni izvor referentnog napona ili interni izvor referentnog napona. U ovom projektu se koristi interni izvor referentnog napona koji se napaja sa Vci pina.

3. Softver

Radi preglednosti drajver softver je podeljen u dve celine, modula. Prvu modul čini ILI9326 drajver niskog nivoa. U ovom modulu se nalaze funkcije koje omogućavaju komunikaciju sa ILI9326 kontrolerom i osnovne operacije nad njim. Drugu celinu čini ILI9326 drajver višeg nivoa. U ovoj celini se nalaze funkcije za crtanje osnovnih grafičkih elemenata, ispis teksta i prikaz slike. Rad displej drajvera niskog nivoa se konfiguriše u datoteci ILI9326_cfg.h.

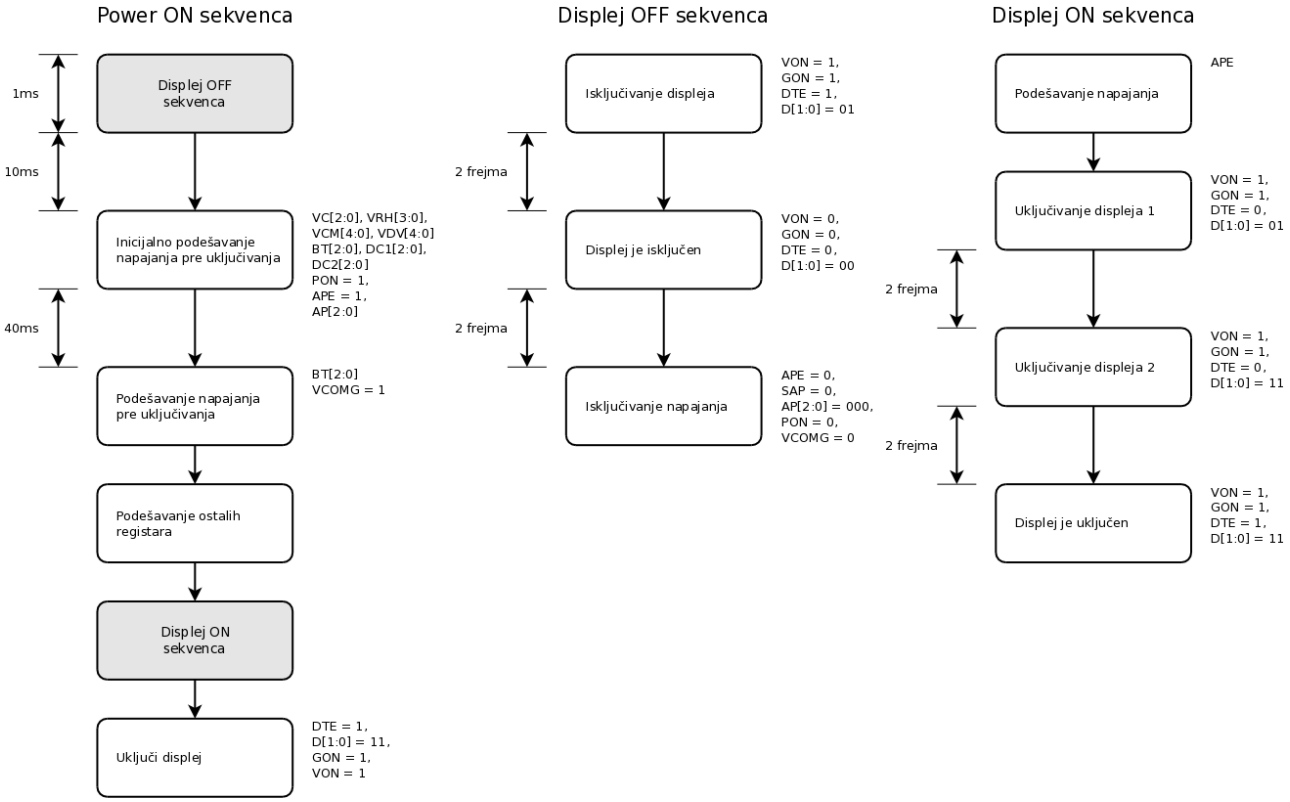
3.1. Spi interfejs

Za efikasnu SPI komunikaciju sa ILI9326 kontrolerom se koriste brze funkcije iz NXP biblioteke za periferiju mikrokontrolera. U toku inicijalizacije displej kontrolera koristi se spora SPI komunikacija, a nakon inicijalizacije se prelazi na brzu SPI komunikaciju čija se brzina podešava preprocesorskom promenljivom *OPT_SPI_SPEED*. Radi postizanja što veće efikasnosti koda pozivaju se funkcija NXP biblioteke sa isključenim internim proverama (*assert* makroi).

3.2. Upravljanje ILI9326 grafičkim kontrolerom

3.2.1. Inicijalizacija

Uključivanje displej kontrolera se mora izvršiti po prikazanoj sekvenci. Data sekvenca osigurava da se uspešno uključe step-up kola i regulatori radi generisanja potrebnih napona za rad LCD displeja. Sekvenca definiše i vremenske intervale koji se moraju ispoštovati kako bi se stabilizovao rad internog oscilatora za step-up kola.



Slika 8: Sekvenca za uključivanje displeja

3.2.2. Režimi upisa boje pixela

Kontroler ILI9326 podražava dva osnovna režima rada za zapis boje pixela na displeju. To su:

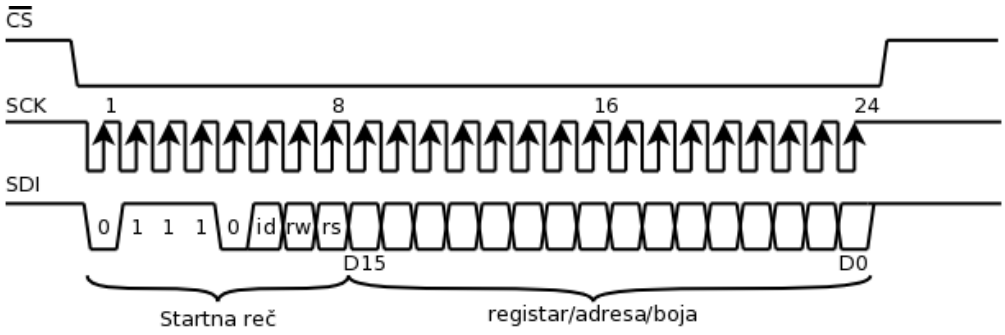
- manje efikasan, ali fleksibilan metod, *jedan pixel* metod
- drugi metod je vrlo brz metod, ali sa dosta ograničenja, *burst* metod.

3.2.2.1. Jedan pixel metod

Ovaj metod omogućava da se na displeju upali tačno jedan pixel. Da bi se to izvršilo potrebno je prvo poslati adresu pixela u GRAM memoriji kontrolera, a zatim poslati i boju tog pixela. To se vrši na sledeći način:

1. šalje se start bajt sa oznakom da se upisuje u indeksni registar
2. SPI se prebaci u 16-bitni režim
3. u indeksni registar se upiše adresa registra za horizontalnu adresu
4. SPI se prebaci u 8-bitni režim
5. šalje se start bajt sa oznakom da se upisuje u data registar
6. SPI se prebaci u 16-bitni režim
7. u data registar se upiše podatak koji želimo da upišemo u registar za horizontalnu adresu
8. ponavljaju se svi koraci od 1 do 7 za registar za vertikalnu adresu
9. šalje se start bajt sa oznakom da se upisuje u indeksni registar
10. SPI se prebaci u16-bitni režim
11. u indeksni registar se upiše adresa registra za RAM podatke
12. SPI se prebavi u 8-bitni režim
13. šalje se start bajt sa oznakom da se upisuje u data registar
14. SPI se prebaci u 16-bitni režim
15. u data registar se pošalje podatak za boju piksela
16. SPI se prebaci u 8-bitni režim

Deo sekvence je prikazan na sledećoj slici:



Slika 9: Prikaz komunikacije sa kontrolerom

Kao što se može videti, ovaj metod je veoma neefikasan što se tiče brzine jer zahteva pristup različitim registrima što iziskuje slanje mnoštvo start bajtova.

3.2.2.2. Burst metod

Ovaj metod je mnogo efikasniji od prethodnog. Ideja ja da se kontroleru kaže u kom deli displeja želimo da upišemo boje piksela i da se nakon toga pošalje niz podataka koje predstavljaju boje. Kontroler interno sam inkrementira brojače za vertikalnu i horizontalnu adresu dok prihvata boje pojedinačnih piksela. Ovim je eliminisana suvišna i nepotrebna komunikacija između kontrolera.

3.3. Organizacija drajvera

Kao što je već ranije pomenuto ILI9326 drajver je podeljen na drajver niskog nivoa i drajver visokog nivoa. U datoteci ILI9326_lld.c je drajver niskog nivoa, čiji je interfejs dostupan u datoteci ILI9326_lld.h. Drajver visokog nivoa se nalazi u datoteci ILI9326.c, a interfejs je u datoteci ILI9326.h. Drajver niskog nivoa se konfiguriše u datoteci ILI9326_cfg.h, dok drajver visokog nivoa nema konfiguraciju.

- ILI9326_lld.c – drajver niskog nivoa
- ILI9326_lld.h – interfejs drajvera niskog nivoa
- ILI9326.c – drajver
- ILI9326.h – interfejs drajvera
- ILI9326_cfg.h – konfiguracija drajvera

3.3.1. ILI9326 drajver niskog nivoa

Funkcije:

```
void I_resetDevice(
    void);
```

Ovo je najkritičnija funkcija ILI9326 drajvera. Ona dovodi ILI9326 kontroler u poznato stanje i vrši konfiguraciju napajanja za displej. Ovu funkciju treba pozvati pre ostalih funkcija za LCD displej.

```
void I_drawPoint(
    uint16_t xCord,
    uint16_t yCord,
    uint16_t color);
```

Ova funkcija postavlja boju jednog piksela po *jedan pixel* metodi koja je objašnjena ranije.

```
uint16_t I_getMaxX(
    void);
```

Vraća maksimalnu X koordinatu displeja za trenutnu orijentaciju.

```
uint16_t I_getMaxY(
    void);
```

Vraća maksimalnu X koordinatu displeja za trenutnu orijentaciju.

```
void I_windowReset(
    void);
```

Postavlja opseg iscrtavanja na ceo displej.

```
void I_windowSet(
    uint16_t x1,
    uint16_t y1,
    uint16_t x2,
    uint16_t y2);
```

Postavlja prozor na zadate koordinate. Prozori se koriste u burst metodi zapisa boje piksela.

```
void I_orientSet(
    lcdOrient_T orientation);
```

Postavlja orijentacija displeja. Ova funkcija se koristi ukoliko je potrebno da se orijentacija displeja dinamički menja u toku izvršavanja aplikacije.

```
lcdOrient_T I_orientGet(
    void);
```

Vraća trenutno postavljenu orijentaciju.

```
void I_pixelWRbegin(void);
```

Započinje proces upisa boje piksela burst metodom.

```
void I_pixelWR(uint16_t color);
```

Upis boje piksela burst metodom.

```
void I_pixelWRend(void);
```

Zaustavlja burst upis boje piksela.

3.3.2. ILI9326 drajver

Funkcije:

```
void drawCutLine(
    uint16_t x1,
    uint16_t y1,
    uint16_t x2,
    uint16_t y2,
    uint16_t color);
```

Crta isprekidanu liniju na displeju.

```
void drawLine(
    uint16_t x1,
    uint16_t y1,
    uint16_t x2,
    uint16_t y2,
    uint16_t color);
```

Crta liniju na displeju.

```
void drawText(
    uint16_t x,
    uint16_t y,
    const char *pText,
    const int8_t (*font)[5],
    uint8_t height,
    uint16_t color);
```

Stampa tekst na displeju.

```
void drawFilledRectangle(
    uint16_t x1,
    uint16_t y1,
    uint16_t x2,
    uint16_t y2,
    uint16_t color);
```

Crta pravougaonik ispunjen bojom.

```
void drawRectangle(
    uint16_t x1,
    uint16_t y1,
    uint16_t x2,
    uint16_t y2,
    uint16_t color);
```

Crta pravougaonik.

```
void drawFlashImage(
    const uint8_t * image,
    uint16_t x,
    uint16_t y);
```

Postavlja sliku iz ROM/FLASH memorije na LCD. Generator slike i opis formata zapisa slike se moze naci u Microchipovoj biblioteci za grafičke aplikacije.

3.4. Test aplikacija za prikaz primljenih podataka

Da bi se potvrdila ispravnost rada drajvera za displej kontroler napravljena je mala test aplikacija koja ima za zadatak da podatke koji pristižu u sistem prikaže u obliku grafikona na displeju. Podaci koji pristižu se tretiraju kao signali koji se odmeravaju na nekom udaljenom mestu. Grafikon prikazuje izmerenu vrednost u funkciji vremena.

Radi jednostavnijeg programiranja test aplikacije i veće fleksibilnosti pribeglo se objektnim tehnikama programiranja. U tu svrhu kreirana su dve klase objekata: objekat grafikon i objekat signal.

Objekat grafikon je opisan strukturom:

```
typedef struct gridObject {
    uint16_t x;
    uint16_t y;
    uint16_t xSize;
    uint16_t ySize;
    uint16_t gridColor;
    uint16_t backColor;
    uint8_t  numRow;
    uint8_t  numCol;
} gridObject_T;
```

gde je:

- x x pozicija,
- y y pozicija,
- xSize x velicina grafikona,
- ySize y velicina grafikona,
- gridColor boja mrezice,
- backColor boja pozadine,
- numRows broj redova u mrezici,
- numCol broj kolona u mrezici.

Objekat grafikon se kreira funkcijom *newGridObject* kojoj se predaje instanca opisne strukture i svi podaci koji je opisuju, na primer:

```
gridObject_T    grid;
newGridObject(
    &grid,
    GRID_X_POS,
    GRID_Y_POS,
    GRID_X_SIZE,
    GRID_Y_SIZE,
    DARKGRAY,
    BLACK,
    6,
    4);
drawGridObject(&grid)
```

U prvom redu se instacira struktura grafikon objekta. Zatim se ova struktura predaje funkciji za kreiranje. Na ovaj način se kreira grafikon *grid* sa pozicijom *GRID_X_POS* i *GRID_Y_POS*, veličine *GRID_X_SIZE* i *GRID_Y_SIZE*, sa bojom mrežice *DARKGRAY* i pozadinskom bojom *BLACK*. Mrežica je podeljena na 6 redova i 4 kolona. Funkcijom *drawGridObject* se vrši iscrtavanje grafikona na ekran.

Na vrlo sličan način se vrši kreiranje i iscrtavanje signal objekta.

```
signalObject_T  redSig;
newSignal(
    &redSig,
    &grid,
    RED);
drawSignal(
    &redSig,
    newSample);
```

Prvom linijom se instacira struktura signal objekta. Funkcijom *newSignal* se kreira signal objekat *redSig* koji se prikazuje na grafikonu *grid* i boje *RED*. Funkcijom *drawSignal* se vrednost promenljive *newSample* prikazuje na grafikonu i vrši linearna interpolacija sa prethodnom vrednošću *redSig* signala.

U test aplikaciji je napravljen jedan objekat grafikon na kome se prikazuju dva signala, crvene i zelene boje, respektivno. Posredstvom uart-a podaci pristižu u sistem, a na grafikonu se pojavljuju krive koje pokazuju vrednost crvenog i zelenog signala.

4. Prilog

4.1. ILI9326.h

```
/*****************************************************************************************************
 * This file is part of ILI9326
 *
 * Copyright (C) 2011, 2012 - Nenad Radulovic
 *
 * ILI9326 is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 2 of the License, or
 * (at your option) any later version.
 *
 * ILI9326 is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with ILI9326; if not, write to the Free Software
 * Foundation, Inc., 51 Franklin St, Fifth Floor,
 * Boston, MA  02110-1301  USA
 *
 * web site:    http://blueskynet.dyndns-server.com
 * e-mail  :    blueskyniss@gmail.com
 *****************************************************************************************************/

#ifndef ILI9326_H_
#define ILI9326_H_

/*****************************************************************************************************
 * INCLUDE FILES
 *****************************************************************************************************/

#include "ILI9326_lld.h"
#include "font.h"

/*---------------------------------------------------------------------------------------------------*
 * EXTERNS
 *---------------------------------------------------------------------------------------------------*//** @cond */

#ifdef ILI9326_H_VAR
# define ILI9326_H_EXT
#else
# define ILI9326_H_EXT extern
#endif


/** @endcond*//*****************************************************************************************
 * DEFINES
 *****************************************************************************************************/

/*---------------------------------------------------------------------------------------------------*//**
 * @name        Predefinisane boje
 * @{ *//*-----------------------------------------------------------------------------------------------*/

#define BLACK           RGBCONV(0U  , 0U  , 0U  )
#define BRIGHTBLUE      RGBCONV(0U  , 0U  , 255U)
#define BRIGHTGREEN     RGBCONV(0U  , 255U, 0U  )
#define BRIGHTCYAN      RGBCONV(0U  , 255U, 255U)
#define BRIGHTRED       RGBCONV(255U, 0U  , 0U  )
#define BRIGHTMAGENTA   RGBCONV(255U, 0U  , 255U)
#define BRIGHTYELLOW    RGBCONV(255U, 255U, 0U  )
#define BLUE            RGBCONV(0U  , 0U  , 128U)
#define GREEN           RGBCONV(0U  , 128U, 0U  )
#define CYAN            RGBCONV(0U  , 128U, 128U)
#define RED             RGBCONV(128U, 0U  , 0U  )
#define MAGENTA         RGBCONV(128U, 0U  , 128U)
#define YELLOW          RGBCONV(255U, 255U, 128U)
```

```
#define BROWN           RGBCONV(255U, 128U, 0U  )
#define LIGHTGRAY       RGBCONV(128U, 128U, 128U)
#define DARKGRAY        RGBCONV(64U , 64U , 64U )
#define LIGHTBLUE       RGBCONV(128U, 128U, 255U)
#define LIGHTGREEN      RGBCONV(128U, 255U, 128U)
#define LIGHTCYAN       RGBCONV(128U, 255U, 255U)
#define LIGHTRED        RGBCONV(255U, 128U, 128U)
#define LIGHTMAGENTA    RGBCONV(255U, 128U, 255U)
#define WHITE           RGBCONV(255U, 255U, 255U)

/** @} *//*-------------------------------------------------------------------------------------------*/


/*****************************************************************************************************
 * MACRO's
 *****************************************************************************************************/

#define RGBCONV(red, green,blue )                                                       \
    ((((red) >> 3) << 11) | (((green) >> 2) << 5) | ((blue) >> 3))


/*-------------------------------------------------------------------------------------------*
 * C++ zastita - pocetak
 *-------------------------------------------------------------------------------------------*/
#ifdef __cplusplus
extern "C"
{
#endif

/*****************************************************************************************************
 * FUNCTION PROTOTYPES
 *****************************************************************************************************/

/*-------------------------------------------------------------------------------------------*/
/**
 * @brief       Crta sprekidanu liniju na displeju
 *
 * @param       x1                      x koordinata pocetka
 * @param       y1                      y koordinata pocetka
 * @param       x2                      x koordinata kraja
 * @param       y2                      y koordinata kraja
 * @param       color                   boja linije
 */
/*-------------------------------------------------------------------------------------------*/
void drawCutLine(
    uint16_t x1,
    uint16_t y1,
    uint16_t x2,
    uint16_t y2,
    uint16_t color);

/*-------------------------------------------------------------------------------------------*/
/**
 * @brief       Crta liniju na displeju
 *
 * @param       x1                      x koordinata pocetka
 * @param       y1                      y koordinata pocetka
 * @param       x2                      x koordinata kraja
 * @param       y2                      y koordinata kraja
 * @param       color                   boja linije
 */
/*-------------------------------------------------------------------------------------------*/
void drawLine(
    uint16_t x1,
    uint16_t y1,
    uint16_t x2,
    uint16_t y2,
    uint16_t color);

/*-------------------------------------------------------------------------------------------*/
/**
 * @brief       Stampa tekst na displeju
 *
```

```
 * @param       x                       x koordinata teksta,
 * @param       y                       y koordinata teksta,
 * @param       pText                   pokazivac na tekst,
 * @param       font                    pokazivac na font,
 * @param       height                  debljina (visina) fonta,
 * @param       color                   boja teksta.
 */
/*-----------------------------------------------------------------------------------------------*/
void drawText(
    uint16_t x,
    uint16_t y,
    const char *pText,
    const int8_t (*font)[5],
    uint8_t height,
    uint16_t color);

/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       Crta pravugaonik ispunjen bojom @c color.
 *
 * @param       x1                      x koordinata prve tacke,
 * @param       y1                      y koordinata prve tacke,
 * @param       x2                      x koordinata zadnje tacke,
 * @param       y2                      y koordinata zadnje tacke,
 * @param       color                   boja ispune.
 * @note        Moraju biti zadovoljeni sledeci uslovi:
 *              - x1 < x2
 *              - y1 < y2
 *              U suprotnom, ponasanje funkcije je nedefinisano.
 */
/*-----------------------------------------------------------------------------------------------*/
void drawFilledRectangle(
    uint16_t x1,
    uint16_t y1,
    uint16_t x2,
    uint16_t y2,
    uint16_t color);

/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       Crta pravugaonik boje @c color.
 *
 * @param       x1                      x koordinata prve tacke,
 * @param       y1                      y koordinata prve tacke,
 * @param       x2                      x koordinata zadnje tacke,
 * @param       y2                      y koordinata zadnje tacke,
 * @param       color                   boja ispune.
 * @note        Moraju biti zadovoljeni sledeci uslovi:
 *              - x1 < x2
 *              - y1 < y2
 *              U suprotnom, ponasanje funkcije je nedefinisano.
 */
/*-----------------------------------------------------------------------------------------------*/
void drawRectangle(
    uint16_t x1,
    uint16_t y1,
    uint16_t x2,
    uint16_t y2,
    uint16_t color);

/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       Postavlja sliku iz ROM/FLASH memorije na LCD.
 *
 * @param       image                   Pokazivac na podatke slike,
 * @param       x                       x koordinata pocetka slike na LCD-u,
 * @param       y                       y koordinata pocetka slike na LCD-u.
 * @note        Slika je u 16-bitnom formatu.
 *
 *              Generator slike i opis formata zapisa slike se moze naci u
 *              Microchipovoj biblioteci za graficke aplikacije.
 */
/*-----------------------------------------------------------------------------------------------*/
```

```
void drawFlashImage(
    const uint8_t * image,
    uint16_t x,
    uint16_t y);


/*-------------------------------------------------------------------------------------------*
 * C++ zastita - kraj
 *-------------------------------------------------------------------------------------------*/
#ifdef __cplusplus
}
#endif


/*****************************************************************************************
 * CONFIGURATION ERRORS
 *****************************************************************************//** @cond */


/** @endcond *//** @} *//*************************************************************************
 * END of ILI9326.h
 *****************************************************************************************/
#endif /* ILI9326_H_ */
```

4.2. ILI9326.c

```
/*****************************************************************************************
 * This file is part of ILI9326
 *
 * Copyright (C) 2011, 2012 - Nenad Radulovic
 *
 * ILI9326 is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 2 of the License, or
 * (at your option) any later version.
 *
 * ILI9326 is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with ILI9326; if not, write to the Free Software
 * Foundation, Inc., 51 Franklin St, Fifth Floor,
 * Boston, MA  02110-1301  USA
 *
 * web site:    http://blueskynet.dyndns-server.com
 * e-mail  :    blueskyniss@gmail.com
 *****************************************************************************************/


/*****************************************************************************************
 * INCLUDE FILES
 *****************************************************************************************/

#include "ILI9326.h"

/*****************************************************************************************
 ***                                 I M P L E M E N T A T I O N                       ***
 *****************************************************************************************/


/*****************************************************************************************
 * GLOBAL FUNCTION DEFINITIONS
 *****************************************************************************************/

void drawFlashImage(
    const uint8_t * image,
    uint16_t x,
    uint16_t y) {

    const uint16_t * tmpPtr;
    uint16_t xSize;
```

```
    uint16_t ySize;
    uint32_t m;
    uint32_t i;

    tmpPtr = (uint16_t *)image;
    ++tmpPtr;
    ySize = *tmpPtr;
    ++tmpPtr;
    xSize = *tmpPtr;
    ++tmpPtr;
    I_windowSet(x, y, x + xSize - 1, y + ySize - 1);
    m = xSize * ySize;

    I_pixelWRbegin();
    for (i = 0; i < m; i++) {
        I_pixelWR(*tmpPtr);
        tmpPtr++;
    }
    I_pixelWRend();
    I_windowReset();
}

void drawLine(
    uint16_t x1,
    uint16_t y1,
    uint16_t x2,
    uint16_t y2,
    uint16_t color) {

    uint16_t dy;
    uint16_t dx;
    uint16_t i = 0;
    int16_t  addx = 1;
    int16_t  addy = 1;
    int16_t  p;
    int16_t  diff;

    diff = ((int16_t)x2 - x1);

    if (diff < 0) {
        diff *= -1;
    }
    dx = diff;
    diff = ((int16_t)y2 - y1);

    if (diff < 0) {
        diff *= -1;
    }
    dy = diff;

    if (x1 > x2) {
        addx = -1;
    }

    if (y1 > y2) {
        addy = -1;
    }

    if (dx >= dy) {
        dy *= 2;
        p = dy - dx;
        diff = p - dx;

        for (; i<=dx ; ++i) {
            I_drawPoint(x1, y1, color);

            if (p < 0) {
                p  += dy;
                x1 += addx;
            } else {
                p  += diff;
                x1 += addx;
                y1 += addy;
```

```
            }
        }
    } else {
        dx *= 2;
        p = dx - dy;
        diff = p - dy;

        for (; i<=dy; ++i) {
            I_drawPoint(x1, y1, color);

            if (p < 0) {
                p  += dx;
                y1 += addy;
            } else {
                p  += diff;
                x1 += addx;
                y1 += addy;
            }
        }
    }
}

void drawCutLine(
    uint16_t x1,
    uint16_t y1,
    uint16_t x2,
    uint16_t y2,
    uint16_t color) {

    uint16_t dy;
    uint16_t dx;
    uint16_t i = 0;
    int16_t  addx = 1;
    int16_t  addy = 1;
    int16_t  p;
    int16_t  diff;

    diff = ((int16_t)x2 - x1);

    if (diff < 0) {
        diff *= -1;
    }
    dx = diff;
    diff = ((int16_t)y2 - y1);

    if (diff < 0) {
        diff *= -1;
    }
    dy = diff;

    if (x1 > x2) {
        addx = -1;
    }

    if (y1 > y2) {
        addy = -1;
    }

    if (dx >= dy) {
        dy *= 2;
        p = dy - dx;
        diff = p - dx;

        for (; i <= dx ; ++i) {

            if ((i & 0x01) == 0x0) {
                I_drawPoint(x1, y1, color);
            }

            if (p < 0) {
                p  += dy;
                x1 += addx;
            } else {
```

```
                p  += diff;
                x1 += addx;
                y1 += addy;
            }
        }
    } else {
        dx *= 2;
        p = dx - dy;
        diff = p - dy;

        for (; i <= dy; ++i) {

            if ((i & 0x01) == 0x0) {
                I_drawPoint(x1, y1, color);
            }

            if (p < 0) {
                p  += dx;
                y1 += addy;
            } else {
                p  += diff;
                x1 += addx;
                y1 += addy;
            }
        }
    }
}

void drawText(
    uint16_t x,
    uint16_t y,
    const char *pText,
    const int8_t (*font)[5],
    uint8_t height,
    uint16_t color) {

    uint16_t i;
    uint16_t j;
    uint16_t k;
    uint16_t l;
    uint16_t m;
    uint16_t n;
    uint16_t tmp;
    int8_t pointData[5];
    const int8_t *tmpPtr;

    n = x;

    while(*pText != '\0'){
        tmpPtr = (font + *pText - ' ')[0];

        for (i=0;i<5;i++) {
            pointData[i] = *tmpPtr++;
        }

        switch (*pText) {

            case '\n': {
                y += 8 * height;
                break;
            }

            case '\r': {
                x = n;
                break;
            }

            default: {

                if ((x + 5 * height) >= I_getMaxY()){
                    x = n;
                    y += 8 * height;
                }
```

```
                for (j = 0; j < 5; ++j, x += height) {

                    for (k = 0; k < 7; k++) {
                        tmp = (0x01 << k);

                        if ((pointData[j] & tmp) == tmp) {

                            for (l = 0; l < height; ++l) {

                                for (m=0; m < height; ++m) {
                                    I_drawPoint(x + m, y + k * height + l, color);
                                }
                            }
                        }
                    }
                }
                x++;
                break;
            }
        }
        pText++;
    }
}

void drawFilledRectangle(
    uint16_t x1,
    uint16_t y1,
    uint16_t x2,
    uint16_t y2,
    uint16_t color) {

    uint32_t i;
    uint32_t m;

    I_windowSet(x1, y1, x2, y2);
    m = ((uint32_t)(x2 - x1 + 1)) * ((uint32_t)(y2 - y1 + 1));

    I_pixelWRbegin();
    for (i = 0; i < m; i++) {
        I_pixelWR(color);
    }
    I_pixelWRend();
    I_windowReset();
}

void drawRectangle(
    uint16_t x1,
    uint16_t y1,
    uint16_t x2,
    uint16_t y2,
    uint16_t color) {


    I_drawHLine(
        x1,
        y1,
        x2 - x1,
        color);
    I_drawHLine(
        x1,
        y2,
        x2 - x1,
        color);
    I_drawVLine(
        x1,
        y1,
        y2 - y1,
        color);
    I_drawVLine(
        x2,
        y1,
        y2 - y1,
```

```
        color);
}

/*****************************************************************************************************
 * CONFIGURATION ERRORS
 *****************************************************************************//** @cond */


/** @endcond *//** @} *//*************************************************************************
 * END of ILI9326.c
 *****************************************************************************************************/
```

4.3. ILI9326_lld.h

```
/*****************************************************************************************************
 * This file is part of ILI9326
 *
 * Copyright (C) 2011, 2012 - Nenad Radulovic
 *
 * ILI9326 is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 2 of the License, or
 * (at your option) any later version.
 *
 * ILI9326 is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with ILI9326; if not, write to the Free Software
 * Foundation, Inc., 51 Franklin St, Fifth Floor,
 * Boston, MA  02110-1301  USA
 *
 * web site:    http://blueskynet.dyndns-server.com
 * e-mail  :    blueskyniss@gmail.com
 *****************************************************************************************************/

#ifndef ILI9326_LLD_H_
#define ILI9326_LLD_H_

/*****************************************************************************************************
 * INCLUDE FILES
 *****************************************************************************************************/


#include <stdint.h>
#include "ILI9326_regs.h"

#include "ILI9326_cfg.h"
#include "lpc17xx_clkpwr.h"
#include "lpc17xx_pinsel.h"
#include "lpc17xx_ssp.h"
#include "lpc17xx_gpio.h"
#include "wait_api.h"

/*---------------------------------------------------------------------------------------------------*
 * EXTERNS
 *---------------------------------------------------------------------------*//** @cond */

#ifdef ILI9326_LLD_H_VAR
# define ILI9326_LLD_H_EXT
#else
# define ILI9326_LLD_H_EXT extern
#endif


/** @endcond*//*************************************************************************************
 * DEFINES
 *****************************************************************************************************/
```

```
/*-------------------------------------------------------------------------------------------*
 * C++ zastita - pocetak
 *-------------------------------------------------------------------------------------------*/
#ifdef __cplusplus
extern "C"
{
#endif

/*********************************************************************************************
 * DATA TYPES
 *********************************************************************************************/

typedef enum lcdOrient {
    I_ORIENT_NORMAL,
    I_ORIENT_0,
    I_ORIENT_90,
    I_ORIENT_180,
    I_ORIENT_270,
} lcdOrient_T;

/*********************************************************************************************
 * FUNCTION PROTOTYPES
 *********************************************************************************************/

/*-------------------------------------------------------------------------------------------*/
/**
 * @brief       Crta horizontalnu liniju.
 *
 * @param       x                       x koordinata,
 * @param       y                       y koordinata,
 * @param       length                  duzina linije,
 * @param       color                   boja linije.
 */
/*-------------------------------------------------------------------------------------------*/
void I_drawHLine(
    uint16_t x,
    uint16_t y,
    uint16_t length,
    uint16_t color);

/*-------------------------------------------------------------------------------------------*/
/**
 * @brief       Crta vertikalnu liniju.
 *
 * @param       x                       x koordinata,
 * @param       y                       y koordinata,
 * @param       length                  duzina linije,
 * @param       color                   boja piksela.
 */
/*-------------------------------------------------------------------------------------------*/
void I_drawVLine(
    uint16_t x,
    uint16_t y,
    uint16_t length,
    uint16_t color);

/*-------------------------------------------------------------------------------------------*/
/**
 * @brief       Postavlja jedan piksel.
 *
 * @param       xCord                   x koordinata,
 * @param       yCord                   y koordinata,
 * @param       color                   boja piksela.
 */
/*-------------------------------------------------------------------------------------------*/
void I_drawPoint(
    uint16_t xCord,
    uint16_t yCord,
    uint16_t color);

/*-------------------------------------------------------------------------------------------*/
/**
 * @brief       Vraca maksimalnu X koordinatu za trenutnu orijentaciju
```

```
 *
 * @return
 */
/*-----------------------------------------------------------------------------------------------*/
uint16_t I_getMaxX(
    void);

/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       Vraca maksimalnu Y koordinatu za trenutnu orijentaciju
 *
 * @return
 */
/*-----------------------------------------------------------------------------------------------*/
uint16_t I_getMaxY(
    void);

/*-----------------------------------------------------------------------------------------------*/
/**
* @brief       Izvrsava inicijalizaciju displej kontrolera i postavljanje na
*              podrazumevane vrednosti.
*
*              Funkcija je radjena po power-on algoritmu koji je dat na
*              120 str. ILI9326 datasheet-a.
*              Display-on sekvenca je data na 118 str.
*/
/*-----------------------------------------------------------------------------------------------*/
void I_resetDevice(
    void);

/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       Postavlja prozor iscrtavanja na ceo LCD displej.
 */
/*-----------------------------------------------------------------------------------------------*/
void I_windowReset(
    void);

/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       Postavlja prozor na zadate koordinate.
 *
 * @param x1
 * @param y1
 * @param x2
 * @param y2
 */
/*-----------------------------------------------------------------------------------------------*/
void I_windowSet(
    uint16_t x1,
    uint16_t y1,
    uint16_t x2,
    uint16_t y2);

/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       Postavlja orijentaciju LCD displeja
 *
 * @param       orientation             Orijentacija
 * @arg         I_ORIENT_NORMAL         - postavlja orijentaciju na podrazumevanu vrednost
(OPT_DISP_ORIENTATION)
 * @arg         I_ORIENT_0              - Orijentacija 0
 * @arg         I_ORIENT_90             - Orijentacija 90
 * @arg         I_ORIENT_180            - Orijentacija 180
 * @arg         I_ORIENT_270            - Orijentacija 270
 */
/*-----------------------------------------------------------------------------------------------*/
void I_orientSet(
    lcdOrient_T orientation);

/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       Vraca trenutno postavljenu orijentaciju displeja
```

```
 *
 * @return
 * @retval      I_ORIENT_0              - Orijentacija 0
 * @retval      I_ORIENT_90             - Orijentacija 90
 * @retval      I_ORIENT_180            - Orijentacija 180
 * @retval      I_ORIENT_270            - Orijentacija 270
 */
/*-----------------------------------------------------------------------------------------------*/
lcdOrient_T I_orientGet(
    void);

/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       Zapocinje proces sukcesivnog upisa boja pixela.
 *
 * @note        Mora se pozvati pre I_pixelWR() funkcije.
 */
/*-----------------------------------------------------------------------------------------------*/
void I_pixelWRbegin(void);

/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       Upis boje piksela
 *
 * @param       color                   Boja pixela.
 */
/*-----------------------------------------------------------------------------------------------*/
void I_pixelWR(uint16_t color);

/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       Zaustavlja proces sukcesivnog upisa boja pixela.
 */
/*-----------------------------------------------------------------------------------------------*/
void I_pixelWRend(void);


/*-----------------------------------------------------------------------------------------------*
 * C++ zastita - kraj
 *-----------------------------------------------------------------------------------------------*/
#ifdef __cplusplus
}
#endif


/*************************************************************************************************
 * CONFIGURATION ERRORS
 ************************************************************************************//** @cond */


/** @endcond *//** @} *//*************************************************************************
 * END of ILI9326_lld.h
 *************************************************************************************************/
#endif /* ILI9326_LLD_H_ */
```

4.4. ILI9326_lld.c

```
/*************************************************************************************************
 * This file is part of ILI9326
 *
 * Copyright (C) 2011, 2012 - Nenad Radulovic
 *
 * ILI9326 is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 2 of the License, or
 * (at your option) any later version.
 *
 * ILI9326 is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
```

```
 * along with ILI9326; if not, write to the Free Software
 * Foundation, Inc., 51 Franklin St, Fifth Floor,
 * Boston, MA  02110-1301  USA
 *
 * web site:    http://blueskynet.dyndns-server.com
 * e-mail  :    blueskyniss@gmail.com
 ************************************************************************************************/

/************************************************************************************************
 * INCLUDE FILES
 ************************************************************************************************/

#include "ILI9326_lld.h"

/************************************************************************************************
 * LOCAL MACRO's
 ************************************************************************************************/

#if defined(__GNUC__)
# define C_INLINE                       static inline __attribute__((always_inline))
#else
# define C_INLINE                       static
#endif

#define VALID_BOUNDARIES(x,y)                                                       \
    ((window_.x1 =< x) && (window_.x2 => x) && (window_.y1 =< y) && (window_.y2 => y))

/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       CS = 1
 */
/*-----------------------------------------------------------------------------------------------*/
#define LCD_DISABLE()                                                               \
    GPIO_SetValue(OPT_LCD_CS_PORT, (1U << OPT_LCD_CS_PIN))

/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       CS = 0
 */
/*-----------------------------------------------------------------------------------------------*/
#define LCD_ENABLE()                                                                \
    GPIO_ClearValue(OPT_LCD_CS_PORT, (1U << OPT_LCD_CS_PIN))

/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       LED = 0
 */
/*-----------------------------------------------------------------------------------------------*/
#define LCD_LED_DISABLE()                                                           \
    GPIO_ClearValue(OPT_LCD_LED_PORT, (1U << OPT_LCD_LED_PIN))

/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       LED = 1
 */
/*-----------------------------------------------------------------------------------------------*/
#define LCD_LED_ENABLE()                                                            \
    GPIO_SetValue(OPT_LCD_LED_PORT, (1U << OPT_LCD_LED_PIN))

/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       RST = 0
 */
/*-----------------------------------------------------------------------------------------------*/
#define LCD_RESET_ACTIVE()                                                          \
    GPIO_ClearValue(OPT_LCD_RST_PORT, (1U << OPT_LCD_RST_PIN))

/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       RST = 1
 */
/*-----------------------------------------------------------------------------------------------*/
#define LCD_RESET_INACTIVE()                                                        \
```

```
    GPIO_SetValue(OPT_LCD_RST_PORT, (1U << OPT_LCD_RST_PIN))

/*-------------------------------------------------------------------------------------------------*/
/**
 * @brief       Makro koji formira odgovarajuci start bajt
 */
/*-------------------------------------------------------------------------------------------------*/
#define LCD_START_BYTE(func)                                                        \
    (0x70 | (OPT_DEVICE_ID << 2) | (func))

/*-------------------------------------------------------------------------------------------------*/
/**
 * @brief       Prebacivanje u 8-bitni rezim rada SPI modula
 */
/*-------------------------------------------------------------------------------------------------*/
#define SPI_MODE_8B()                                                               \
    do {                                                                            \
        while(LPC_SSP0->SR & (1 << 4));                                             \
        SSP_Cmd(LPC_SSP0, DISABLE);                                                 \
        SSP_cfgStruct.Databit = SSP_DATABIT_8;                                      \
        SSP_Init(LPC_SSP0, &SSP_cfgStruct);                                        \
        SSP_Cmd(LPC_SSP0, ENABLE);                                                  \
    } while (0)

/*-------------------------------------------------------------------------------------------------*/
/**
 * @brief       Prebacivanje u 16-bitni rezim rada SPI modula
 */
/*-------------------------------------------------------------------------------------------------*/
#define SPI_MODE_16B()                                                              \
    do {                                                                            \
        while(LPC_SSP0->SR & (1 << 4));                                             \
        SSP_Cmd(LPC_SSP0, DISABLE);                                                 \
        SSP_cfgStruct.Databit = SSP_DATABIT_16;                                     \
        SSP_Init(LPC_SSP0, &SSP_cfgStruct);                                        \
        SSP_Cmd(LPC_SSP0, ENABLE);                                                  \
    } while (0)

/*-------------------------------------------------------------------------------------------------*/
/**
 * @brief       Inicijalizacija GPIO modula
 */
/*-------------------------------------------------------------------------------------------------*/
#define GPIO_INIT()                                                                 \
    do {                                                                            \
        PINSEL_CFG_Type PinCfg;                                                     \
        PinCfg.Funcnum = PINSEL_FUNC_0;                                             \
        PinCfg.OpenDrain = PINSEL_PINMODE_OPENDRAIN;                                \
        PinCfg.Pinmode = PINSEL_PINMODE_TRISTATE;                                   \
        PinCfg.Portnum = OPT_LCD_LED_PORT;                                          \
        PinCfg.Pinnum = OPT_LCD_LED_PIN;                                            \
        PINSEL_ConfigPin(&PinCfg);                                                  \
        GPIO_SetDir(OPT_LCD_LED_PORT, (1U << OPT_LCD_LED_PIN), 1);                  \
        GPIO_SetDir(OPT_LCD_CS_PORT, (1U << OPT_LCD_CS_PIN), 1);                    \
        GPIO_SetDir(OPT_LCD_RST_PORT, (1U << OPT_LCD_RST_PIN), 1);                  \
    } while (0)

/*-------------------------------------------------------------------------------------------------*/
/**
 * @brief       Inicijalizacija SPI modula - mala brzina
 */
/*-------------------------------------------------------------------------------------------------*/
#define SPI_INIT()                                                                  \
    do {                                                                            \
        PINSEL_CFG_Type PinCfg;                                                     \
        CLKPWR_SetPCLKDiv(CLKPWR_PCLKSEL_SSP0, CLKPWR_PCLKSEL_CCLK_DIV_1);          \
        PinCfg.Funcnum = PINSEL_FUNC_2;                                             \
        PinCfg.OpenDrain = PINSEL_PINMODE_NORMAL;                                   \
        PinCfg.Pinmode = PINSEL_PINMODE_TRISTATE;                                   \
        PinCfg.Portnum = PINSEL_PORT_0;                                             \
        PinCfg.Pinnum = 15;                                                         \
        PINSEL_ConfigPin(&PinCfg);                                                  \
        PinCfg.Pinnum = 17;                                                         \
```

```
        PINSEL_ConfigPin(&PinCfg);                                          \
        PinCfg.Pinnum = 18;                                                 \
        PINSEL_ConfigPin(&PinCfg);                                          \
        SSP_ConfigStructInit(&SSP_cfgStruct);                               \
        SSP_cfgStruct.ClockRate = 10000;                                    \
        SSP_cfgStruct.CPHA = OPT_SSP_PHA;                                   \
        SSP_cfgStruct.CPOL = OPT_SSP_POL;                                   \
        SSP_cfgStruct.Databit = SSP_DATABIT_8;                              \
        SSP_cfgStruct.FrameFormat = SSP_FRAME_SPI;                          \
        SSP_Init(LPC_SSP0, &SSP_cfgStruct);                                 \
        SSP_Cmd(LPC_SSP0, ENABLE);                                          \
    } while (0)

/*-------------------------------------------------------------------------------------------------*/
/**
 * @brief       Inicijalizacija SPI modula - velika brzina
 */
/*-------------------------------------------------------------------------------------------------*/
#define SPI_INIT_FASTEST()                                                  \
    do {                                                                    \
        SSP_Cmd(LPC_SSP0, DISABLE);                                         \
        SSP_ConfigStructInit(&SSP_cfgStruct);                               \
        SSP_cfgStruct.ClockRate = OPT_SPI_SPEED;                            \
        SSP_cfgStruct.CPHA = OPT_SSP_PHA;                                   \
        SSP_cfgStruct.CPOL = OPT_SSP_POL;                                   \
        SSP_cfgStruct.Databit = SSP_DATABIT_8;                              \
        SSP_cfgStruct.FrameFormat = SSP_FRAME_SPI;                          \
        SSP_Init(LPC_SSP0, &SSP_cfgStruct);                                 \
        SSP_Cmd(LPC_SSP0, ENABLE);                                          \
    } while (0)

/*-------------------------------------------------------------------------------------------------*/
/**
 * @brief       Slanje podataka sa SPI modula
 */
/*-------------------------------------------------------------------------------------------------*/
#define SPI_TX(data)                                                        \
    SSP_SendData(LPC_SSP0, data)

/*-------------------------------------------------------------------------------------------------*/
/**
 * @brief       Cekanje u milisekundama
 */
/*-------------------------------------------------------------------------------------------------*/
#define WAIT_MS(time)                                                       \
    wait_ms(time)

/*-------------------------------------------------------------------------------------------------*/
/**
 * @brief       Pocetak sukcesivnog upisa boja pixela
 */
/*-------------------------------------------------------------------------------------------------*/
#define WR_PIXEL_BEGIN()                                                    \
    do {                                                                    \
        LCD_ENABLE();                                                       \
        SPI_TX(LCD_START_BYTE(INDEX_WR));                                   \
        SPI_MODE_16B();                                                     \
        SPI_TX(I_GRAM_WR_DATA);                                             \
        SPI_MODE_8B();                                                      \
        LCD_DISABLE();                                                      \
        LCD_ENABLE();                                                       \
        SPI_TX(LCD_START_BYTE(REG_WR));                                     \
        SPI_MODE_16B();                                                     \
    } while (0)

/*-------------------------------------------------------------------------------------------------*/
/**
 * @brief       Upis boje jednog piksela
 */
/*-------------------------------------------------------------------------------------------------*/
#define WR_PIXEL(color)                                                     \
    SPI_TX(color)
```

```
/*-----------------------------------------------------------------------------------------------*/
/**
 * @brief       Zavrsetak sukcesivnog upisa boja
 */
/*-----------------------------------------------------------------------------------------------*/
#define WR_PIXEL_END()                                                          \
    do {                                                                        \
        SPI_MODE_8B();                                                          \
        LCD_DISABLE();                                                          \
    } while (0)

/*************************************************************************************************
 * LOCAL GLOBAL VARIABLES
 *************************************************************************************************/

static SSP_CFG_Type SSP_cfgStruct;
static lcdOrient_T orientation_;


/*************************************************************************************************
 * LOCAL FUNCTION PROTOTYPES
 *************************************************************************************************/

C_INLINE uint16_t getMaxX(
    void);

C_INLINE uint16_t getMaxY(
    void);

C_INLINE void wrIndex(
    uint16_t index);

C_INLINE void wrData(
    uint16_t data);

C_INLINE void setReg(
    uint16_t reg,
    uint16_t value);

void setAddres(
    uint16_t x,
    uint16_t y);


/*************************************************************************************************
 ***                                 I M P L E M E N T A T I O N                               ***
 *************************************************************************************************/


/*************************************************************************************************
 * GLOBAL FUNCTION DEFINITIONS
 *************************************************************************************************/

void I_drawPoint(
    uint16_t x,
    uint16_t y,
    uint16_t color) {

    setAddres(
        x,
        y);
    setReg(
        I_GRAM_WR_DATA,
        color);
}

void I_drawHLine(
    uint16_t x,
    uint16_t y,
    uint16_t length,
    uint16_t color) {

    uint32_t i;
```

```
    for (i = 0; i < length; i++) {
        I_drawPoint(
            x + i,
            y,
            color);
    }
}

void I_drawVLine(
    uint16_t x,
    uint16_t y,
    uint16_t length,
    uint16_t color) {

    uint32_t i;

    setAddres(x, y);

    WR_PIXEL_BEGIN();
    for (i = 0; i < length; i++) {
        WR_PIXEL(color);
    }
    WR_PIXEL_END();
}

uint16_t I_getMaxX(
    void) {

    return (getMaxX());
}

uint16_t I_getMaxY(
    void) {

    return (getMaxY());
}

void I_orientSet(
    lcdOrient_T orientation) {

    switch (orientation) {

        case I_ORIENT_0 : {
            orientation_ = I_ORIENT_0;
            setReg(
                I_ENTRY_MODE,
                BIT_BGR | BITS_IDIR(0x03));
            break;
        }

        case I_ORIENT_90 : {
            orientation_ = I_ORIENT_90;
            setReg(
                I_ENTRY_MODE,
                BIT_BGR | BIT_AM | BITS_IDIR(0x02));
            break;
        }

        case I_ORIENT_180 : {
            orientation_ = I_ORIENT_180;
            setReg(
                I_ENTRY_MODE,
                BIT_BGR);
            break;
        }

        case I_ORIENT_270 : {
            orientation_ = I_ORIENT_270;
            setReg(
                I_ENTRY_MODE,
                BIT_BGR | BIT_AM | BITS_IDIR(0x01));
            break;
```

```
        }

        default : {
#if (OPT_DISP_ORIENTATION == 0)
            orientation_ = I_ORIENT_0;
            setReg(
                I_ENTRY_MODE,
                BIT_BGR | BITS_IDIR(0x03));
#elif (OPT_DISP_ORIENTATION == 90)
            orientation_ = I_ORIENT_90;
            setReg(
                I_ENTRY_MODE,
                BIT_BGR | BIT_AM | BITS_IDIR(0x02));
#elif (OPT_DISP_ORIENTATION == 180)
            orientation_ = I_ORIENT_180;
            setReg(
                I_ENTRY_MODE,
                BIT_BGR);
#elif (OPT_DISP_ORIENTATION == 270)
            orientation_ = I_ORIENT_270;
            setReg(
                I_ENTRY_MODE,
                BIT_BGR | BIT_AM | BITS_IDIR(0x01));
#endif
            break;
        }
    }
}

lcdOrient_T I_orientGet(
    void) {

    return orientation_;
}

void I_pixelWR(
    uint16_t color) {

    WR_PIXEL(color);
}

void I_pixelWRbegin(
    void) {

    WR_PIXEL_BEGIN();
}

void I_pixelWRend(
    void) {


    WR_PIXEL_END();
}

void I_resetDevice(
    void) {

    GPIO_INIT();
    SPI_INIT();
    LCD_LED_DISABLE();
    LCD_RESET_INACTIVE();
    LCD_DISABLE();
    WAIT_MS(100);
    LCD_RESET_ACTIVE();
    WAIT_MS(10);
    LCD_RESET_INACTIVE();
    WAIT_MS(10);

/*-----------------------------------------------------------------------------------------------*
 * Display OFF setting
 *-----------------------------------------------------------------------------------------------*/
    setReg(
        0x0702,
```

```
        0x3008);                                                        /* Set internal
timing,don't change this value              */
    setReg(
        0x0705,
        0x0036);                                                        /* Set internal
timing,don't change this value              */
    setReg(
        0x070B,
        0x1213);                                                        /* Set internal
timing,don't change this value              */
    setReg(
        I_DISP_CTRL_1,
        0x0000);
    setReg(
        I_PWR_CTRL_1,
        0x0000);
    setReg(
        I_PWR_CTRL_3,
        0x0000);
    setReg(
        I_PWR_CTRL_4,
        0x0000);
    WAIT_MS(10);

/*-----------------------------------------------------------------------------------------------*
 * Power supply initial setting VC, VDV, VRH, VCM
 *-----------------------------------------------------------------------------------------------*/
    setReg(
        I_PWR_CTRL_2,
        BITS_VC(OPT_VC));                                               /* Vc = 2.475V
*/
    WAIT_MS(10);
    setReg(
        I_PWR_CTRL_3,
        BITS_VRH(OPT_VRH));                                             /* Vrh = 4.33V
*/
    WAIT_MS(10);
    setReg(
        I_VCOMH,
        OPT_VCM);                                                       /* Vcm = 4.33V
*/
    WAIT_MS(10);
    setReg(
        I_PWR_CTRL_4,
        BITS_VDV(OPT_VDV));                                             /* Vdv = 5.6V
*/
    WAIT_MS(100);

/*-----------------------------------------------------------------------------------------------*
 * Power supply operation setting, part 1
 *-----------------------------------------------------------------------------------------------*/
    setReg(
        I_PWR_CTRL_1,
        BITS_BT(0x0) | BITS_AP(OPT_AP));                                /* BT = 0, AP =
gamma and source driver config              */
    WAIT_MS(10);
    setReg(
        I_PWR_CTRL_2,
        BITS_VC(OPT_VC) | BITS_DC0(OPT_DC0) | BITS_DC1(OPT_DC1));       /* DC0 i DC1 izvori
konstantne struje                         */
    WAIT_MS(10);
    setReg(
        I_PWR_CTRL_3,
        BITS_VRH(OPT_VRH) | BIT_PON | BIT_PSON | BIT_VREG1R(OPT_VREG1R));       /* PON = 1, Vgl
output is enabled                           */
    WAIT_MS(10);
    setReg(
        I_PWR_CTRL_1,
        BITS_BT(0x0) | BITS_AP(OPT_AP) | BIT_APE);                      /* APE = 1, start
the generation of power supply             */
    WAIT_MS(200);

/*-----------------------------------------------------------------------------------------------*
```

```
 * Power supply operation setting, part 2
 *-------------------------------------------------------------------------------------------------*/
    setReg(
        I_PWR_CTRL_1,
        BITS_BT(OPT_BT) | BITS_AP(OPT_AP) | BIT_APE);                          /* set Vgh, Vgl,
Vcl, Vlcd                                    */
    WAIT_MS(10);
    setReg(
        I_PWR_CTRL_4,
        BITS_VDV(OPT_VDV) | BIT_VCOMG);                                        /* VCOMG = 1, VCOM
output low is fixed to VCOML              */
    WAIT_MS(200);

/*-------------------------------------------------------------------------------------------------*
 * Ostala podesavanja
 *-------------------------------------------------------------------------------------------------*/
    setReg(
        I_GRAM_ADDR_H_SET,
        0x0000);
    setReg(
        I_GRAM_ADDR_V_SET,
        0x0000);
    setReg(
        I_DISP_CTRL_2,
        0x0808);                                                               /* set the back
porch and front porch                        */
    setReg(
        I_DISP_CTRL_3,
        0x0000);                                                               /* set non-display
area refresh cycle ISC[3:0]               */
    setReg(
        I_RGB_DISP_INTF_CTRL_1 ,
        0x0000);                                                               /* RGB interface
setting 16 bits color depth                */
    setReg(
        I_RGB_DISP_INTF_CTRL_2,
        0x0000);                                                               /* RGB interface
polarity                                     */

    /*
     * Podesavanje gamma krive
     */
    setReg(
        I_GAMMA_CTRL_1,
        0x0000);
    setReg(
        I_GAMMA_CTRL_2,
        0x0707);
    setReg(
        I_GAMMA_CTRL_3,
        0x0606);
    setReg(
        I_GAMMA_CTRL_4,
        0x0006);
    setReg(
        I_GAMMA_CTRL_5,
        0x0A09);
    setReg(
        I_GAMMA_CTRL_6,
        0x0203);
    setReg(
        I_GAMMA_CTRL_7,
        0x0005);
    setReg(
        I_GAMMA_CTRL_8,
        0x0007);
    setReg(
        I_GAMMA_CTRL_9,
        0x0400);
    setReg(
        I_GAMMA_CTRL_10,
        0x000A);
```

```
    /*
     * Podesavanje GRAM oblasti
     */
    setReg(
        I_H_ADDR_START_POS,                                             /* Horizontal GRAM
Start Address                           */
        0x0000);
    setReg(
        I_H_ADDR_END_POS,                                               /* Horizontal GRAM
End Address                             */
        OPT_DISP_X);
    setReg(
        I_V_ADDR_START_POS,                                             /* Vertical GRAM
Start Address                             */
        0x0000);
    setReg(
        I_V_ADDR_END_POS,                                               /* Vertical GRAM End
Address                                 */
        OPT_DISP_Y);
    setReg(
        I_BASE_IMG_DISP_CTRL_1,
        BITS_NL(0x35));
    setReg(
        I_BASE_IMG_DISP_CTRL_2,
        BIT_REV);
    setReg(
        I_BASE_IMG_DISP_CTRL_3,                                         /* Sets the
scrolling amount of base image                  */
        0x0000);

    /*
     * Kontrola prikaza parcijalne slike
     */
    setReg(
        I_PART_IMG_1_DISP_POS,                                          /* Partial Image 1
Display Position                        */
        0x0000);
    setReg(
        I_PART_IMG_1_AREA_START,                                        /* Partial Image 1
RAM Start Address                       */
        0x0000);
    setReg(
        I_PART_IMG_1_AREA_END,                                          /* Partial Image 1
RAM End Address                         */
        0x0000);
    setReg(
        I_PART_IMG_2_DISP_POS,                                          /* Partial Image 2
Display Position                        */
        0x0000);
    setReg(
        I_PART_IMG_2_AREA_START,                                        /* Partial Image 2
RAM Start Address                       */
        0x0000);
    setReg(
        I_PART_IMG_2_AREA_END,                                          /* Partial Image 2
RAM End Address                         */
        0x0000);

    /*
     * Kontrola panel-a
     */
    setReg(
        I_PANEL_INTF_CTRL_1,
        BITS_RTNI(0x10));
    setReg(
        I_PANEL_INTF_CTRL_2,
        0x0000);
    setReg(
        I_PANEL_INTF_CTRL_4,
        BITS_RTNE(0x2));
    setReg(
        I_FRAME_RATE_COLOR_CTRL,
        0x0000);
```

```
/*-------------------------------------------------------------------------------------------------*
 * Display ON sekvenca
 *-------------------------------------------------------------------------------------------------*/
    setReg(
        I_PWR_CTRL_1,
        BITS_BT(OPT_BT) | BITS_AP(OPT_AP) | BIT_APE | BIT_SAP);                /* SAP = 1, Source
driver is enabled                          */
    WAIT_MS(10);
    setReg(
        I_DISP_CTRL_1,
        BIT_BASEE | BIT_VON | BIT_GON | BITS_D(0x01));
    WAIT_MS(10);
    setReg(
        I_DISP_CTRL_1,
        BIT_BASEE | BIT_VON | BIT_GON | BITS_D(0x03));
    WAIT_MS(10);
    setReg(
        I_DISP_CTRL_1,
        BIT_BASEE | BIT_VON | BIT_GON | BIT_DTE | BITS_D(0x03));
    WAIT_MS(10);
    SPI_INIT_FASTEST();
    WAIT_MS(10);
    I_orientSet(
        I_ORIENT_NORMAL);
    LCD_LED_ENABLE();
}

void I_windowReset(
    void) {

    setReg(
        I_H_ADDR_START_POS,
        0);
    setReg(
        I_H_ADDR_END_POS,
        OPT_DISP_X);
    setReg(
        I_V_ADDR_START_POS,
        0);
    setReg(
        I_V_ADDR_END_POS,
        OPT_DISP_Y);
}

void I_windowSet(
    uint16_t x1,
    uint16_t y1,
    uint16_t x2,
    uint16_t y2) {

    uint16_t mx1;
    uint16_t mx2;
    uint16_t my1;
    uint16_t my2;
    uint16_t sizeY;
    uint16_t sizeX;

    if (x2 > getMaxY()) {                                                       /* Ovde su obrnute X
i Y koordinate                         */
        x2 = getMaxY();
    }

    if (y2 > getMaxX()) {
        y2 = getMaxX();
    }

    sizeX = x2 - x1;
    sizeY = y2 - y1;

    switch (orientation_) {

        case I_ORIENT_90 : {
```

```
            mx1 = getMaxY() - y1 - sizeY;
            mx2 = getMaxY() - y1;
            my1 = x1;
            my2 = x1 + sizeX;
            break;
        }

        case I_ORIENT_180 : {                                                   /* Ovde su obrnute X
i Y koordinate                          */
            mx1 = getMaxX() - y1 - sizeY;
            mx2 = getMaxX() - y1;
            my1 = getMaxY() - x1 - sizeX;
            my2 = getMaxY() - x1;
            break;
        }

        case I_ORIENT_270 : {
            mx1 = y1;
            mx2 = y1 + sizeY;
            my1 = getMaxX() - x1 -sizeX;
            my2 = getMaxX() - x1;
            break;
        }

        default : {
            mx1 = x1;
            mx2 = x2;
            my1 = y1;
            my2 = y2;
            break;
        }
    }
//    setReg(
//        I_H_ADDR_START_POS,
//        mx1);
//    setReg(
//        I_H_ADDR_END_POS,
//        mx2);
//    setReg(
//        I_V_ADDR_START_POS,
//        my1);
//    setReg(
//        I_V_ADDR_END_POS,
//        my2);

    setReg(
        I_H_ADDR_START_POS,
        mx1);
    setReg(
        I_H_ADDR_END_POS,
        mx2);
    setReg(
        I_V_ADDR_START_POS,
        my1);
    setReg(
        I_V_ADDR_END_POS,
        my2);
    setAddres(
        x1,
        y1);
}


/**************************************************************************************************
 * LOCAL FUNCTION DEFINITIONS
 *************************************************************************************************/

C_INLINE uint16_t getMaxX(
    void) {

    if ((I_ORIENT_90 == orientation_) || (I_ORIENT_270 == orientation_)) {

        return (OPT_DISP_Y);
```

```
    } else {

        return (OPT_DISP_X);
    }
}

C_INLINE uint16_t getMaxY(
    void) {

    if ((I_ORIENT_90 == orientation_) || (I_ORIENT_270 == orientation_)) {

        return (OPT_DISP_X);
    } else {

        return (OPT_DISP_Y);
    }
}

C_INLINE void wrIndex(
    uint16_t index) {

    LCD_ENABLE();
    SPI_TX(LCD_START_BYTE(INDEX_WR));
    SPI_MODE_16B();
    SPI_TX(index);
    SPI_MODE_8B();
    LCD_DISABLE();
}

C_INLINE void wrData(
    uint16_t data) {

    LCD_ENABLE();
    SPI_TX(LCD_START_BYTE(REG_WR));
    SPI_MODE_16B();
    SPI_TX(data);
    SPI_MODE_8B();
    LCD_DISABLE();
}

C_INLINE void setReg(
    uint16_t reg,
    uint16_t value) {

    wrIndex(
        reg);
    wrData(
        value);
}

void setAddres(
    uint16_t x,
    uint16_t y) {

    uint32_t addr;

    switch (orientation_) {

        case I_ORIENT_90 : {
            addr = (x * 0x100) + (getMaxY() - y);
            break;
        }

        case I_ORIENT_180 : {
            addr = ((getMaxY() - x) * 0x100) +(getMaxX() - y);                  /* Obrnute su X i Y
koordinate                                  */
            break;
        }

        case I_ORIENT_270 : {
            addr = ((getMaxX() - x) * 0x100) + y;
            break;
        }
```

```
        default : {
            addr = (y * 0x100) + x;
            break;
        }
    }

    setReg(
        I_GRAM_ADDR_H_SET,
        addr & 0x00FF);
    setReg(
        I_GRAM_ADDR_V_SET,
        addr >> 8);
}


/*****************************************************************************************************
 * CONFIGURATION ERRORS
 *****************************************************************************//** @cond */


/** @endcond *//** @} *//*************************************************************************
 * END of ILI9326_lld.c
 *****************************************************************************************************/
```

4.5. ILI9326_cfg.h

```
/*****************************************************************************************************
 * This file is part of ILI9326
 *
 * Copyright (C) 2011, 2012 - Nenad Radulovic
 *
 * ILI9326 is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 2 of the License, or
 * (at your option) any later version.
 *
 * ILI9326 is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with ILI9326; if not, write to the Free Software
 * Foundation, Inc., 51 Franklin St, Fifth Floor,
 * Boston, MA  02110-1301  USA
 *
 * web site:    http://blueskynet.dyndns-server.com
 * e-mail  :    blueskyniss@gmail.com
 *****************************************************************************************************/

#ifndef ILI9326_CFG_H_
#define ILI9326_CFG_H_

/*---------------------------------------------------------------------------------------------------*/
/**
 * @brief       Konfiguracija IM0 pin-a kada se koristi SPI komunikacija.
 */
/*---------------------------------------------------------------------------------------------------*/
#define OPT_DEVICE_ID                   0

/*---------------------------------------------------------------------------------------------------*/
/**
 * @brief       Orijentacija displeja
 *
 *              Moguce vrednosti:
 *              - 0
 *              - 90
 *              - 180
 *              - 270
 */
/*---------------------------------------------------------------------------------------------------*/
#define OPT_DISP_ORIENTATION            180
```

```
/*-------------------------------------------------------------------------------------------------*/
/**
 * @brief       X velicina - pogledati str 109 zasto je X = 240, a Y = 432.
 */
/*-------------------------------------------------------------------------------------------------*/
#define OPT_DISP_X                      239U

/*-------------------------------------------------------------------------------------------------*/
/**
 * @brief       Y velicina
 */
/*-------------------------------------------------------------------------------------------------*/
#define OPT_DISP_Y                      431U

/*-------------------------------------------------------------------------------------------------*/
/**
 * @brief       Brzina SPI interfejsa
 */
/*-------------------------------------------------------------------------------------------------*/
#define OPT_SPI_SPEED                   48000000U

/*-------------------------------------------------------------------------------------------------*/
/**
 * @brief       Polaritet signala SPI modula
 *
 *              Pogledati LPC1768 datasheet.
 *
 *              Moguce vrednosti:
 *              - 0
 *              - 1
 *              - 2
 *              - 3
 */
/*-------------------------------------------------------------------------------------------------*/
#define OPT_SPI_POLARITY                3


/** @} *//*-------------------------------------------------------------------------------------------*/
/*-------------------------------------------------------------------------------------------------*//**
 * @name        Podesavanja za pinove
 * @{ *//*-------------------------------------------------------------------------------------------*/
#define OPT_LCD_RST_PORT                0
#define OPT_LCD_RST_PIN                 25
#define OPT_LCD_CS_PORT                 0
#define OPT_LCD_CS_PIN                  24
#define OPT_LCD_LED_PORT                2
#define OPT_LCD_LED_PIN                 1

/** @} *//*-------------------------------------------------------------------------------------------*/
/*-------------------------------------------------------------------------------------------------*//**
 * @name        Podesavanja LCD kontrolera
 * @{ *//*-------------------------------------------------------------------------------------------*/

/**
 * @brief       Podesavanje glavne naponske reference
 *
 *              Vci - napon napajanja: (2,5 - 3,3)V
 *              Vci1 = VC * Vci
 */
#define OPT_VC                          0x04

/**
 * @brief       Podesavanje Vrh napona
 *
 *              Vrh = VRH * Vci1
 */
#define OPT_VRH                         0x0B

/**
 * @brief       Podesavanje Vcm napona
 *
 *              Vcm = VCM * Vrh
```

```
 */
#define OPT_VCM                         0x1F

/**
 * @brief       Podesavanje Vdv napona
 *
 *              Vdv = VDV * Vrh
 */
#define OPT_VDV                         0x1F

/**
 * @brief       Izvori konstantne struje 1
 *
 *              Podrazumevano: 0x06
 */
#define OPT_DC0                         0x00

/**
 * @brief       Izvori konstantne struje 2
 *
 *              Podrazumevano: 0x06
 */
#define OPT_DC1                         0x01

/**
 * @brief       Izvori konstantne struje u OPAMP-ovima
 */
#define OPT_AP                          0x01

/**
 * @brief       Podesavanje Vgh, Vgl, Vcl, Vlcd
 *
 *              BT = 0x04
 *              Vgh = Vci1 * 5 = 12,375V
 *              Vgl = Vci1 * 4 = 9,9V
 *              Vcl = -Vci1 = -2.475V
 *              Vlcd = Vci1 * 2 = 4,95V
 */
#define OPT_BT                          0x05

/**
 * @brief       Koristi se interni VREG generator
 */
#define OPT_VREG1R                      1

/** @} *//*-------------------------------------------------------------------------------------*/


/*****************************************************************************************
 * CONFIGURATION ERRORS
 *****************************************************************************//** @cond */

#if (OPT_SPI_POLARITY == 0)
# define OPT_SSP_POL                    SSP_CPOL_HI
# define OPT_SSP_PHA                    SSP_CPHA_FIRST
#elif (OPT_SPI_POLARITY == 1)
# define OPT_SSP_POL                    SSP_CPOL_HI
# define OPT_SSP_PHA                    SSP_CPHA_SECOND
#elif (OPT_SPI_POLARITY == 2)
# define OPT_SSP_POL                    SSP_CPOL_LO
# define OPT_SSP_PHA                    SSP_CPHA_FIRST
#elif (OPT_SPI_POLARITY == 3)
# define OPT_SSP_POL                    SSP_CPOL_LO
# define OPT_SSP_PHA                    SSP_CPHA_SECOND
#endif


/** @endcond *//** @} *//*************************************************************************
 * END of ILI9326_cfg.h
 ****************************************************************************************/
#endif /* ILI9326_CFG_H_ */
```

4.6. main.cpp

```
/*****************************************************************************************************
 * This file is part of ILI9326
 *
 * Copyright (C) 2011, 2012 - Nenad Radulovic
 *
 * ILI9326 is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 2 of the License, or
 * (at your option) any later version.
 *
 * ILI9326 is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with ILI9326; if not, write to the Free Software
 * Foundation, Inc., 51 Franklin St, Fifth Floor,
 * Boston, MA  02110-1301  USA
 *
 * web site:    http://blueskynet.dyndns-server.com
 * e-mail  :    blueskyniss@gmail.com
 *****************************************************************************************************/

/*****************************************************************************************************
 * INCLUDE FILES
 *****************************************************************************************************/

#include "mbed.h"
#include "ILI9326.h"

/*****************************************************************************************************
 * LOCAL DEFINES
 *****************************************************************************************************/

/*-----------------------------------------------------------------------------------------------*//**
 * @name        Pozicija i velicina grafika na displeju
 * @{ *//*-------------------------------------------------------------------------------------------*/

#define GRID_X_POS                      20
#define GRID_Y_POS                      40
#define GRID_X_SIZE                     256
#define GRID_Y_SIZE                     128

/** @} *//*-----------------------------------------------------------------------------------------*/

/**
 * @brief       Stanja FSM automata
 */
typedef enum signalFsm {
    STATE_RED_SIG, //!< STATE_RED_SIG
    STATE_GREEN_SIG//!< STATE_GREEN_SIG
} signalFsm_T;

/*****************************************************************************************************
 * LOCAL DATA TYPES
 *****************************************************************************************************/

/**
 * @brief       Struktura grafikona
 */
typedef struct gridObject {
    uint16_t x;
    uint16_t y;
    uint16_t xSize;
    uint16_t ySize;
    uint16_t gridColor;
    uint16_t backColor;
    uint8_t  numRow;
    uint8_t  numCol;
} gridObject_T;
```

```
/**
 * @brief       Struktura signala (objekat)
 */
typedef struct signalObject {
    struct gridObject * grid;
    uint16_t    x;
    uint16_t    y;
    uint16_t    color;
} signalObject_T;

/*****************************************************************************************
 * LOCAL GLOBAL VARIABLES
 *****************************************************************************************/

/**
 * @brief       Konfiguracija testnog UART interfejsa
 */
Serial      uart(p28, p27);

/**
 * @brief       Statusna LED dioda
 */
DigitalOut  status(LED1);

/*****************************************************************************************
 * LOCAL FUNCTION PROTOTYPES
 *****************************************************************************************/

/**
 * @brief       Kreira nov grafikon
 *
 * @param       aGrid                   Pokazivac na struktura grafikona,
 * @param       xPos                    x pozicija,
 * @param       yPos                    y pozicija,
 * @param       xSize                   x velicina grafikona,
 * @param       ySize                   y velicina grafikona,
 * @param       gridColor               boja mrezice,
 * @param       backColor               boja pozadine,
 * @param       rows                    broj redova u mrezici,
 * @param       columns                 broj kolona u mrezici.
 */
void newGridObject (
    gridObject_T    * aGrid,
    uint16_t        xPos,
    uint16_t        yPos,
    uint16_t        xSize,
    uint16_t        ySize,
    uint16_t        gridColor,
    uint16_t        backColor,
    uint8_t         rows,
    uint8_t         columns);

/**
 * @brief       Crta grafikon koji je opisan u @c aGrid
 *
 * @param       aGrid                   Pokazivac na grafikon koji treba nacrtati.
 */
void drawGridObject (
    gridObject_T    * aGrid);

/**
 * @brief       Kreira nov signal objekat za prikaz na gradikonu.
 *
 * @param       aSignal                 Pokazivac na strukturu signal objekta,
 * @param       aGrid                   pokazivac na grafikon gde treba da se crta signal,
 * @param       color                   boja signala na grafikonu.
 */
void newSignal(
    signalObject_T  * aSignal,
    gridObject_T    * aGrid,
    uint16_t        color);
```

```
/**
 * @brief       Dodaje novu izmerenu vrednost @c aSignalSample u signal @c aSignal
 *
 * @param       aSignal                 Pokazivac na strukturu signala,
 * @param       aSignalSample           nova izmerena vrednost signala.
 */
void drawSignal(
    signalObject_T  * aSignal,
    uint16_t        aSignalSample);


/*****************************************************************************************
 ***                                 I M P L E M E N T A T I O N                       ***
 *****************************************************************************************/

int main() {
    signalFsm_T     state;
    gridObject_T    grid;
    signalObject_T  redSig;
    signalObject_T  greenSig;

    I_resetDevice();
    I_orientSet(
        I_ORIENT_180);
    drawFilledRectangle(
        0,
        0,
        431,
        239,
        BLACK);
    drawText(
        20,
        20,
        "Acquired data:",
        &ARIAL[0],
        1,
        YELLOW);
    drawText(
        20,
        188,
        "Legend:",
        &ARIAL[0],
        1,
        YELLOW
        );
    drawText(
        20,
        198,
        "Stream 1",
        &ARIAL[0],
        1,
        RED
        );
    drawText(
        20,
        208,
        "Stream 2",
        &ARIAL[0],
        1,
        GREEN
        );
    drawText(
        300,
        20,
        "Hor: 64 samples",
        &ARIAL[0],
        1,
        YELLOW
        );
    drawText(
        300,
        30,
        "Ver: 16%",
```

```
        &ARIAL[0],
        1,
        YELLOW
        );
    newGridObject(
        &grid,
        GRID_X_POS,
        GRID_Y_POS,
        GRID_X_SIZE,
        GRID_Y_SIZE,
        DARKGRAY,
        BLACK,
        6,
        4);
    drawGridObject(
        &grid);
    newSignal(
        &redSig,
        &grid,
        RED);
    newSignal(
        &greenSig,
        &grid,
        GREEN);
    state = STATE_RED_SIG;

    while (true) {

        if (uart.readable()) {
            status = !status;

            switch (state) {
                case STATE_RED_SIG : {
                    drawSignal(
                        &redSig,
                        uart.getc() << 8);
                    state = STATE_GREEN_SIG;
                    break;
                }

                case STATE_GREEN_SIG : {
                    drawSignal(
                        &greenSig,
                        uart.getc() << 8);
                    state = STATE_RED_SIG;
                    break;
                }

                default : {
                    break;
                }
            }
        }
    }
    return (0);
}


/*****************************************************************************************
 * LOCAL FUNCTION DEFINITIONS
 *****************************************************************************************/

void newGridObject (
    gridObject_T    * aGrid,
    uint16_t        xPos,
    uint16_t        yPos,
    uint16_t        xSize,
    uint16_t        ySize,
    uint16_t        gridColor,
    uint16_t        backColor,
    uint8_t         rows,
    uint8_t         columns) {
```

```
    aGrid->x = xPos;
    aGrid->y = yPos;
    aGrid->xSize = xSize;
    aGrid->ySize = ySize;
    aGrid->gridColor = gridColor;
    aGrid->backColor = backColor;
    aGrid->numRow = rows;
    aGrid->numCol = columns;
}


void drawGridObject (
    gridObject_T    * aGrid) {

    uint16_t x1;
    uint16_t y1;
    uint8_t  i;
    float    dref;
    float    dtmp;

    x1 = aGrid->x + aGrid->xSize;
    y1 = aGrid->y + aGrid->ySize;
    drawRectangle(
        aGrid->x,
        aGrid->y,
        x1,
        y1,
        WHITE);
    drawFilledRectangle(
        aGrid->x + 1U,
        aGrid->y + 1U,
        x1 - 1U,
        y1 - 1U,
        aGrid->backColor);
    dtmp = 0.0f;
    dref = (float)(aGrid->ySize - 2U) / (float)(aGrid->numRow);

    for (i = 1U; i < (aGrid->numRow); i++) {
        dtmp += dref;
        drawCutLine(
            aGrid->x + 1U,
            aGrid->y + dtmp,
            x1 - 1U,
            aGrid->y + dtmp,
            aGrid->gridColor);
    }
    dtmp = 0.0f;
    dref = (float)(aGrid->xSize - 2U) / (float)(aGrid->numCol);

    for (i = 1U; i < (aGrid->numCol); i++) {
        dtmp += dref;
        drawCutLine(
            aGrid->x + dtmp,
            aGrid->y + 1U,
            aGrid->x + dtmp,
            y1 - 1U,
            aGrid->gridColor);
    }
}

void newSignal(
    signalObject_T  * aSignal,
    gridObject_T    * aGrid,
    uint16_t        color) {

    aSignal->grid = aGrid;
    aSignal->x = 0U;
    aSignal->y = aGrid->ySize - 1U;
    aSignal->color = color;

}

void drawSignal(
```

```
    signalObject_T  * aSignal,
    uint16_t        aSignalSample) {

    uint16_t xCord;
    uint16_t yCord;
    uint16_t xCordNext;
    uint16_t yCordNext;

    if ((aSignal->x + 1U) > (aSignal->grid->xSize - 1U)) {
        aSignal->x = 0U;
        drawGridObject(
            aSignal->grid);
    }
    xCord = aSignal->grid->x + 1U + aSignal->x;
    yCord = aSignal->grid->y + 1U + aSignal->y;
    aSignal->x++;
    aSignal->y = aSignalSample / (UINT16_MAX / aSignal->grid->ySize);

    if (aSignal->y > aSignal->grid->ySize - 1U) {
        aSignal->y = aSignal->grid->ySize - 1U;
    }
    xCordNext = aSignal->grid->x + 1U + aSignal->x;
    yCordNext = aSignal->grid->y + 1U + aSignal->y;
    drawLine(
        xCord,
        yCord,
        xCordNext,
        yCordNext,
        aSignal->color);
}


/*****************************************************************************************************
 * CONFIGURATION ERRORS
 *****************************************************************************//** @cond */


/** @endcond *//** @} *//*************************************************************************
 * END of main.c
 *****************************************************************************************************/
```