

Vaja 3

Testna plošča, ADC, LCD

Naloga

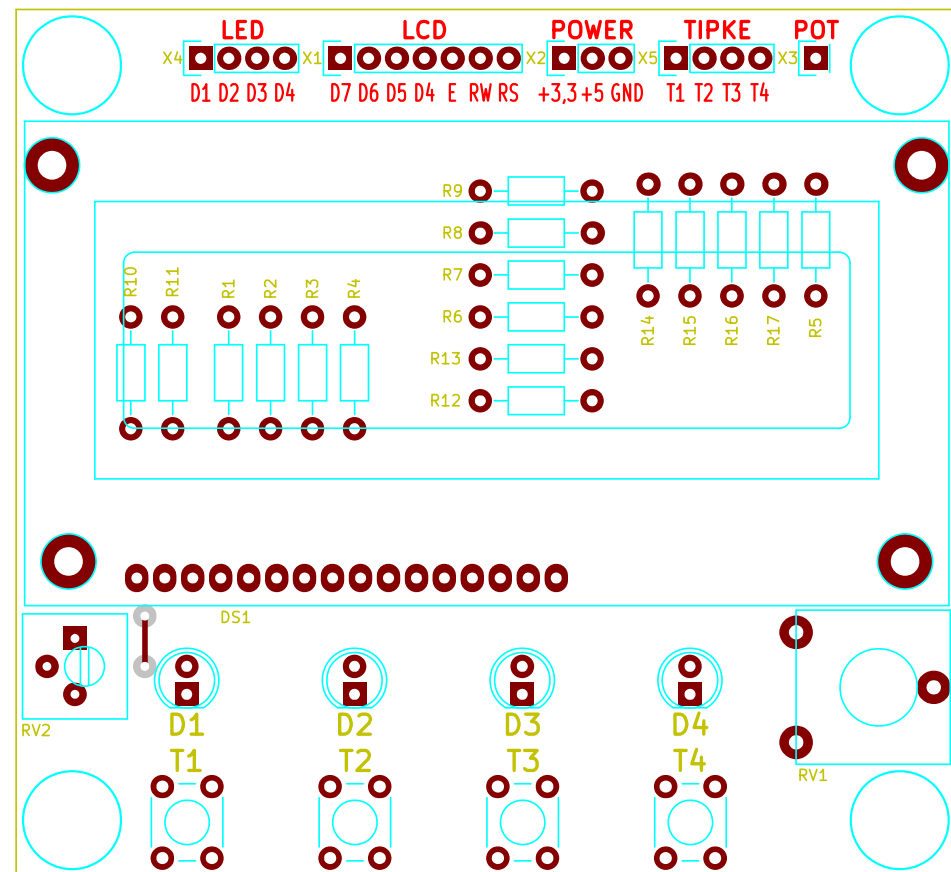
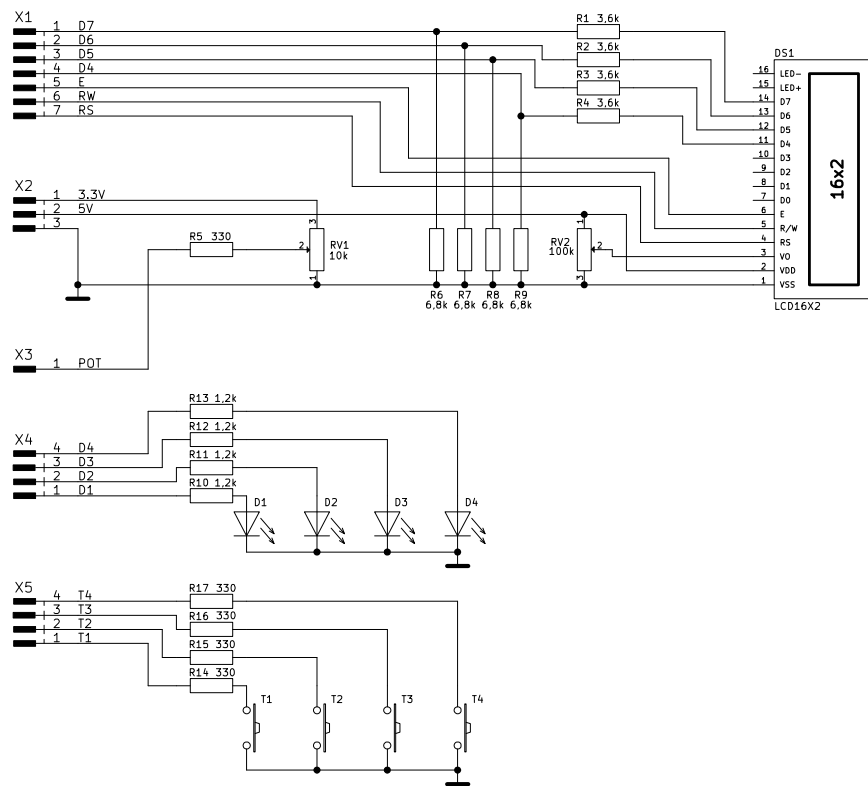
Napišite program, ki bo na zaslonu LCD prikazoval stanje potenciometra POT

Napišite funkcijo **void adc_setup(void)**, ki bo na Arduino pinu A0 (pin št. 40, SAM3X8E pin PA16), vklopila funkcijo A/D pretvornika (A/D kanal 7)

Napišite funkcijo **uint32_t adc_read(void)**, ki naj sproži A/D pretvorbo, počaka na rezultat in vrne vrednost v obliki nepredznačenega 32 bitnega celega števila

Stanje potenciometra prikazujte na zaslonu LCD na vsake 0.5 skunde
npr. **"AD = xxxx"**, kjer je xxxx 12 bitno celo število z intervala [0, 4095]

Testna ploščica



Testna ploščica - povezave

Pazite na pravilno priključitev napajalnih linij (GND, 3.3V, 5V)

Tipke (glej vajo 2)

LED (glej vajo 2)

Potenciometer:

Povežite pin POT s testne ploščice na Arduino pin z oznako A0 (št.pina 40, SAM3X8E pin PA16, ADC kanal 7)

LCD:

LCD D4: Arduino pin 50, SAM3X8E pin PC13

LCD D5: Arduino pin 48, SAM3X8E pin PC15

LCD D6: Arduino pin 46, SAM3X8E pin PC17

LCD D7: Arduino pin 44, SAM3X8E pin PC19

LCD RS: Arduino pin 51, SAM3X8E pin PC12

LCD RW: Arduino pin 49, SAM3X8E pin PC14

LCD E: Arduino pin 47, SAM3X8E pin PC16

LCD

Za uporabo LCD zaslona je že napisan zelo preprost vmesnik

S spletne strani predmeta prenesite "lcd.c" (v mapo "app/src") in "lcd.h" (v mapo "app/include")

Vključite "lcd.h" v program

Funkcije:

void lcd_init(void)

Inicializira lcd zaslon, ki mora biti povezan na ustrezne pine (glej "lcd.h")

void lcd_driver(void)

Na lcd zaslon pošlje 32 znakov (2 vrstici x 16 znakov, 32 x 8 bitov), ki se nahajajo na naslovu shranjenem v globalnem kazalcu **lcd_string**

Kazalec drži naslov statičnega kosa pomnilnika (glej lcd.c)

```
char lcd_buff[33] = "                               ";  
char *lcd_string = lcd_buff;
```

LCD - vmesnik

1. Zapis vsebine v lcd medpomnilnik **lcd_buff** (nanj kaže kazalec **lcd_string**)

Na naslov v **lcd_string** je potrebno zapisati želeni tekst

Največ 32 znakov

Lahko z uporabo funkcije "sprintf(..)"

npr: **sprintf(lcd_string, "HELLO");**

Pazite na število znakov (buffer overflow)

Kaj, če želimo popraviti vsebino samo enega dela zaslona?

2. Klic gonilnika **lcd_driver();**

ADC - analogno digitalni pretvornik

Dodajte “sam/drivers/adc” v projekt (glej vajo 1)

Večina I/O linij je multipleksiranih (nanje je lahko priključenih več perifernih enot)

npr. PA16: SPCK1 (SPI1 clock), TD (SSC transmit data), AD7

Ustrezno funkcijo je potrebno izbrati – za ADC se nastavi avtomatsko pri vključitvi pripadajočega AD kanala

ADC

10 ali 12 bitna resolucija

16 A/D linij (na AD15 je temp. senzor), na Arduino pine je speljanih le 8 (pini A0 – A7, glej shemo)

Uporabite Arduino pin A0 → pin PA16, ADC kanal 7

Različni načini proženja (timer, pwm, zunanji, programski)

Nastavili bomo programsko proženje

Veliko dodatnih opsijskih možnosti, npr. Ojačenje, diferencialni vhod, ...(glej datasheet)

ADC nastavitve

Definicije:

"sam/drivers/adc/adc.h", "sam/utils/cmsis/sam3x/include/component/component_adc.h"

Kanal: ADC_CHANNEL_7

Arduino Due pin A0, PA16

Frekvenca: 6.4 MHz

Med MCK/2 (main clock) in MCK/512

Startup time: ADC_STARTUP_TIME_4 (= ADC_MR_STARTUP_SUT64 → 64 period AD ure)

Ob vklopu kanala, odvisen od prejšnjega stanja ADC (izklopljen, sleep)

Tracking time: 0 (glej datasheet formulo)

Vzorčenje signala (sample & hold), odvisen od notranje impedance signalnega vira

Transfer period: 1 (glej datasheet formulo)

Čas prenosa vzorčenega signala v vezje za pretvorbo (v tem času vzorčenje signala ne poteka)

Settling time: ADC_SETTLING_TIME_3 (= ADC_MR_SETTLING_AST17 → 17 period AD ure)

Pri različnih nastavitvah kanalov (ojačenje, offset, ...)

Resolucija: ADC_12_BITS (= ADC_MR_LOWRES_BITS_12 → 12 bitna resolucija)

Lahko tudi 10 bitna

Proženje: ADC_TRIG_SW

Programsko proženje

"FREERUN ": 0

Izklopimo neproženo vzorčenje

ADC – funkcije (glej adc.h, adc.c)

`sysclk_enable_peripheral_clock(ID_ADC)`

`uint32_t adc_init(Adc *p_adc, const uint32_t ul_mck, const uint32_t ul_adc_clock, const enum adc_startup_time startup)`

`void adc_configure_timing(Adc *p_adc, const uint8_t uc_tracking, const enum adc_settling_time_t settling, const uint8_t uc_transfer)`

`void adc_set_resolution(Adc *p_adc, const enum adc_resolution_t resolution)`

`void adc_configure_trigger(Adc *p_adc, const enum adc_trigger_t trigger, uint8_t uc_freerun)`

`void adc_enable_channel(Adc *p_adc, const enum adc_channel_num_t adc_ch)`

`void adc_start(Adc *p_adc)`

`uint32_t adc_get_status(const Adc *p_adc)`

`uint32_t adc_get_channel_value(const Adc *p_adc, const enum adc_channel_num_t adc_ch)`

ADC pretvorba

