Osnove mikroprocesorske elektronike

Priprava 6: Tipkovnica

Namen te priprave je, da si na podlagi projekta prejšnje vaje pripravite izhodiščni projekt za reševanje vaje s tipkovnico. Namreč, *z vsako naslednjo vajo bomo gradili na že obstoječih rešitvah iz prejšnjih vaj* vse do konca semestra, ko boste na sistemskem nivoju v celoti oživili Miškota in nato na njem implementirali končno aplikacijo. V sklopu priprave se boste tudi podučili o *cikličnem medpomnilniku*, ki ga bomo uporabljali tekom naslednjih vaj.

Priprava na vajo

Doma pripravite naslednje stvari:

1. ***s pomočjo orodja*** `STM Cube Project Copy` ***ustvarite nov projekt z imenom***

   `VAJA_06-keyboard`

   `STM Cube Project Copy` je preprosto orodje, ki vam na podlagi že obstoječega projekta pomaga ustvariti nov projekt tako, da datoteke projekta skopira in poskrbi za ustrezna preimenovanja znotraj projekta. Navodila za uporabo najdete znotraj arhivske datoteke, s katero je prišlo orodje `STM Cube Project Copy`.

   Kot *izhodiščni projekt* za kopiranje uporabite projekt uspešno rešene prejšnje vaje `VAJA_05-LEDs`. Nov projekt naj se nahaja znotraj vašega "STM Cube workspace" direktorija.

   Na tak način boste lahko za vsako vajo ustvarili nov projekt, ki bo temeljil na vseh vaših dotedanjih rešitvah.

2. ***V orodju "STM Cube IDE" uvozite novi projekt*** `VAJA_06-keyboard`.

   Novonastali projekt je potrebno še *uvoziti med aktivne projekta* Cube IDE okolja. Opis postopka za uvažanje projekta najdete med navodili za uporabo orodja `STM Cube Project Copy`.

3. ***Novemu projektu dodajte modula*** `buf.c` ***in*** `kbd.c`.

   Modul `buf.c` vsebuje podporo za delo s *cikličnim medpomnilnikom*. Tak medpomnilnik bomo uporabili pri implementaciji tipkovnice tako, da bo hranil informacijo o pritisnjenih tipkah, na katere se še nismo odzvali.

   Znotraj modula `kbd.c` pa boste implementirali funkcionalnost tipkovnice v sklopu te vaje.

   Datoteke modulov dodajte v podmapo "`System`" na ustrezno mesto. Datoteke najdete v mapi "dodatno" gradiva laboratorijske vaje.

4. ***Podučite se o uporabi modula za ciklični medpomnilnik*** `buf.c`.

   Pomembno je, da *okvirno razumete* delovanje cikličnega medpomnilnika ter da znate modul `buf.c` *pravilno uporabiti*, saj bomo medpomnilnik uporabljali tudi pri naslednjih vajah.

   Vso potrebno gradivo na temo cikličnega medpomnilnika najdete v dokumentu "Ciklični medpomnilnik" v mapi "dodatno".

5. ***Preučite, kateri pini mikrokrmilnika so uporabljeni za branje stanja tipk.***

   Pomagajte si seveda z električno shemo Miškota.

6. ***V grafičnem vmesniku CubeMX nastavite te pine kot digitalne vhode z vklopljenim notranjim zgornjim uporom (tj. "pull-up" uporom).***

   Lastnosti uporabljenih GPIO pinov nastavljate znotraj že znane rubrike:

   "Pinout & Configuration → System Core → GPIO → Configuration".

7. ***Shranite nastavitve v CubeMX in tako zgenerirajte novo avtomatsko kodo za inicializacijo digitalnih GPIO vhodov.***

   Podobno kot pri prejšnji vaji, ste s tem zadnjim korakom poskrbeli, da se bo *dodala programska koda za inicializacijo vseh digitalnih vhodov*, ki jih bomo uporabljali za branje tipk tipkovnice. Poskrbeli ste torej za *inicializacijo strojne opreme* in tako je vse pripravljeno, da pričnemo implementirati *funkcije za delo s tipkovnico* na višjem, *sistemskem nivoju*!

8. ***Preučite, katero nizko-nivojsko funkcijo bomo potrebovali za branje stanja posamezne tipke.***

   Podobno kot pri prejšnji vaji, si bomo pri implementaciji *sistemskih* funkcij za delo s tipkovnico pomagali z *nizko-nivojsko* funkcijo (angl. low-level, LL), s katero bomo brali stanje digitalnih vhodov, kamor so priključene tipke.

   Pomagate si lahko z uradno dokumentacijo za "LL knjižnico".

   Namig: sedaj že veste, da je uradna dokumentacija nerodna, saj ne vsebuje preglednega seznama LL funkcij, iz katerega bi lahko hitro uganili, katero funkcijo potrebujete. Tak seznam lahko dobite znotraj orodja STM Cube IDE, če smiselno uporabite funkcijo "auto-comple". Glejte namig spodaj.

NE POZABITE DOMA PRIPRAVLJENI PROJEKT PRINESTI S SEBOJ NA VAJO!

To storite tako, da celotno projektno mapo stisnete v arhiv (npr. zip) in prinesete s seboj ta arhiv. Na vajah pa bomo pripravljeni projekt le še uvozili.

Namigi

Uporaba "auto-complete" funkcionalnosti za pridobitev seznama funkcij iz LL knjižnice

STM Cube IDE orodje vsebuje t. i. *"auto-complete" funkcionalnost*, ki vam pomaga hitro dopolniti izraze, ki ste se jih namerili zapisati v urejevalnik programske kode.

Ideja: kaj če bi to "auto-complete" funkcionalnost uporabili tako, da bi z njo hitro *prišli do seznama vseh funkcij iz ustrezne LL knjižnice*?

Kako to storiti?

Razmišljamo takole. Vemo, da želimo uporabiti LL funkcijo za delo z GPIO periferijo. In vemo, da so funkcije LL (in tudi HAL knjižnice) vedno poimenovane po znanem vzorcu:

- na začetku je pripona LL, ki pove, da imamo opravka z "low-layer" knjižnico (rumeni del na sliki spodaj)
- nato sledi kratica, ki nakaže, s katero periferijo upravljajo LL funkcije, v našem primeru torej GPIO (modri del na sliki spodaj)
- ter na koncu sledi ime funkcije, ki nakazuje funkcionalnost LL funkcije (vijolični del na sliki spodaj)

Poglejte primer iz uradne dokumentacije:

Function name

__STATIC_INLINE void LL_GPIO_SetPinMode (GPIO_TypeDef * GPIOx, uint32_t Pin, uint32_t Mode)

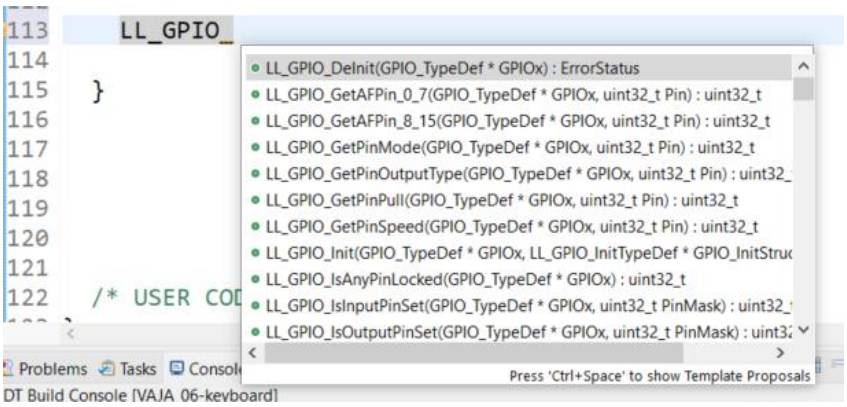
Function description

Configure gpio mode for a dedicated pin on dedicated port.

In tako že slutimo rešitev: v STM Cube IDE moramo pričeti pisati izraz oblike

```
LL_GPIO_
```

ter nato uporabiti funkcijo "auto-complete". **Funkcijo "auto-complete" sprožite tako, da pritisnete bližnjico "CTRL + SPACE".** Poglejte rezultat na sliki spodaj. Odpre se vam seznam vseh LL_GPIO funkcij in potrebno je le še pobrskati po seznamu in izbrati smiselno funkcijo.



Slika 1 - CTRL+SPACE bližnjica sproži "auto-complete" funkcijo, ki vam lahko hitro pomaga najti željeno funkcijo iz knjižnice.