Osnove mikroprocesorske elektronike

Vaja 11: AD pretvornik

V sklopu priprave na vajo ste poskrbeli za *inicializacijo strojne opreme*, ki je potrebna, da lahko merimo pozicijo osi "joysticka": AD pretvornik ste nastavili za delo v DMA načinu ter nastavili proženje AD pretvorbe s pomočjo časovnika. V sklopu vaje pa boste sedaj poskrbeli še za implementacijo funkcionalnosti "joysticka" na sistemskem nivoju: definirali boste ustrezne podatkovne strukture, poskrbeli za kalibracijo osi "joysticka" ter pripravili funkcijo, ki iz "surovih meritev" pozicije osi "joysticka" izračuna *relativno* pozicijo osi.

Naloge vaje

1. ***Novemu projektu dodajte modula*** `joystick.c` ***in*** `joystick.h`, znotraj katerega bomo v sklopu vaje implementirali *sistemski modul* za delo z "joystickom".

   Modula najdete v mapi "predloge". Dodajte ju v projektno mapo znotraj podmape "`System`" na ustrezno mesto.

2. ***Definirajte podatkovno strukturo, ki bo hranila vse potrebne parametre za delo z "joystickom".***

   Definirali jo boste s pomočjo naslednjih korakov:

   a) definirajte naštevni *tip* `joystick_axes_enum_t`, kjer boste *definirali imena vseh osi "joysticka"* (angl. axes). Definicijo umestite v datoteko `joystick.h`.

   b) Definirajte naštevni *tip* `joystick_buttons_enum_t`, kjer boste *definirali ime edine tipke* "joysticka" `JOY_BTN_FIRE`. Definicijo umestite v datoteko `joystick.h`.

   c) Definirajte *tip* "handle" strukture `joystick_handle_t`, ki pa bo hranila vse potrebne parametre za delo z "joystickom".

      Definiciji tipa dodajte sledeče podatkovne strukture:

      - `position_raw[]` – tabela, kamor bo DMA enota shranjevala "surove" rezultate AD pretvorbe (angl. raw measurements), torej meritve pozicije osi "joysticka";
      - `position_raw_min[]` – tabela, kjer bomo hranili informacijo o *najmanjšem* odklonu osi "joysticka",
      - `position_raw_max[]` – tabela, kjer bomo hranili informacijo o *največjem* odklonu osi "joysticka",
      - `position_raw_range[]` – tabela, kjer bomo hranili informacijo o *razponu odklona* osi "joysticka" (angl. axis range).

   d) Na podlagi zgornjega tipa definirajte *globalno* spremenljivko `joystick`, ki pa bo naša "handle" struktura za "joystick".

3. ***Definirajte dolžino medpomnilnika za shranjevanje informacije o pritisnjenih tipkah "joysticka".***

   Poskrbite, da bo dolžina medpomnilnika za tipke "joysticka" nastavljena na 16.

4. ***Poskrbite za inicializacijo "joystick" modula znotraj*** `JOY_init()` ***funkcije.***

   Pri inicializaciji boste poskrbeli za sledeče stvari:

   a) za tipko "joysticka" specificirate, na kateri GPIO pin in port je priključena,
   b) smiselno nastavite *začetne vrednosti sistemskih spremenljivk* "joysticka",
   c) shranite kazalca na "handle" strukturi za časovnik in AD pretvornik, ki sta potrebni, če želimo ti dve periferni enoti upravljati s HAL funkcijami,
   d) inicializirate medpomnilnik za tipke "joysticka",
   e) izvedete kalibracijo AD pretvornika s pomočjo HAL funkcije,
   f) zaženete AD pretvornik v DMA načinu,
   g) zaženete časovnik, da prične s štetjem,
   h) počakate toliko časa, da bo časovnik zagotovo že sprožil prvo AD pretvorbo.

5. ***Dopolnite implementacijo funkcij, ki so povezane z osmi "joysticka".***

   Funkcije, ki implementirajo funkcionalnost povezano s tipkami "joysticka" so že implementirane, saj ste se s takim problemom že srečali tekom oživljanja tipkovnice.

   Vaša naloga je, da dokončate implementacijo funkcij, ki so povezane z meritvijo pozicije osi "joysticka". To pa pomeni, da morate dopolniti sledeči funkciji:

   a) `JOY_calibrate()` – funkcija, ki poskrbi za *kalibracijo* "joysticka". Znotraj te funkcije poskrbite, da se zabeležijo ekstremni odkloni za posamezne osi "joysticka".

      O ideji kalibracije "joysticka" "joysticka" si lahko preberete v poglavju spodaj.

   b) `JOY_get_axis_position()` – funkcija, ki vrne *relativno* pozicijo osi "joysticka". Relativno pozicijo podaja v smislu procentualnega deleža celotnega razpona odklona osi.

      O ideji izračuna relativne pozicije "joysticka" si lahko preberete v poglavju spodaj.

6. ***Stestirajte in demonstrirajte delovanje "joysticka" s pomočjo testnih funkcij.***

   Delovanje "joysticka" boste stestirali s pomočjo pomožne "debug" funkcije `JOY_SCI_send_status()`, ki na serijski vmesnik SCI izpiše stanje ključnih sistemskih spremenljivk "joysticka" ter trenutno relativno lego osi "joysticka". S to funkcijo boste lahko spremljali sistemske spremenljivke "joysticka" v realnem času med kalibracijo in po njej.

   Delovanje tipke "joysticka" boste lahko preizkusili s funkcijo `JOY_button_demo()`, ki jo je potrebno malenkost dopolniti.

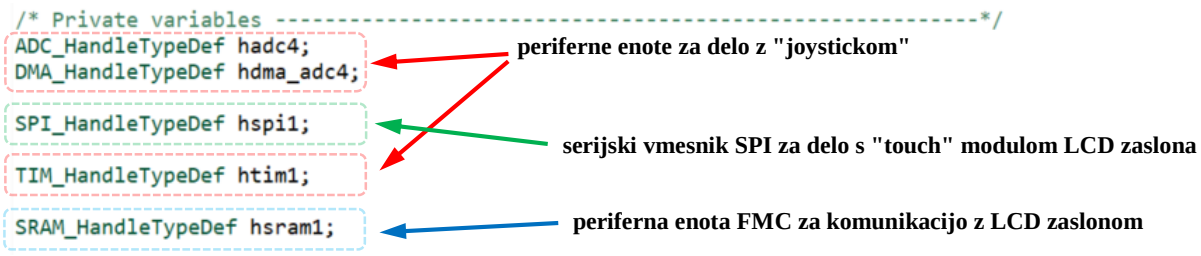
DODATNA NALOGA – Test "joysticka" s prižiganjem LEDice

V sklopu funkcije `JOY_LED_demo()` spišite testno kodo, ki "premika prižgano LEDico" od `LED7` do `LED0` glede na trenutno relativno pozicijo osi "joysticka".

Dodatna pojasnila

Delo s HAL knjižnico in "handle" strukture za periferne enote

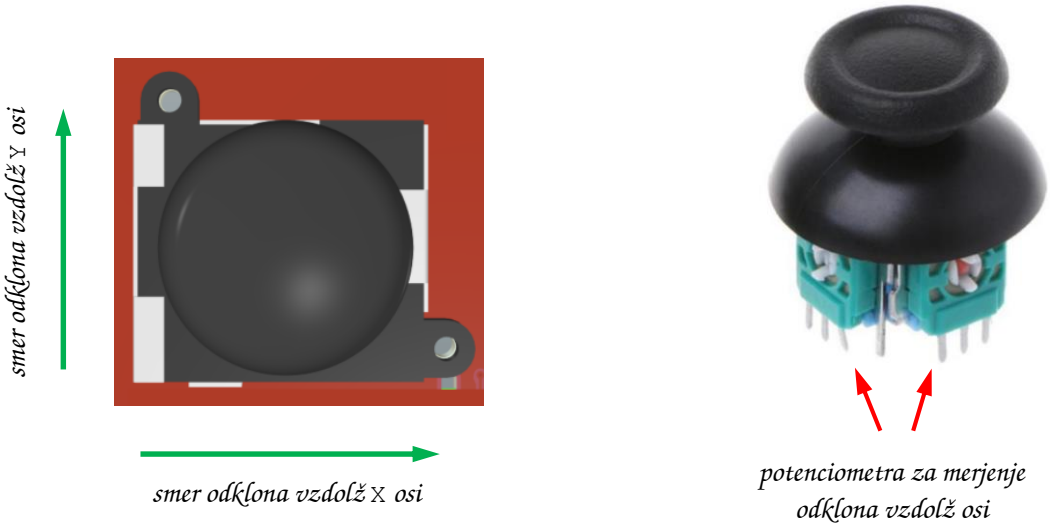
Do sedaj smo za upravljanje perifernih enot mikrokrmilnika pretežno uporabljali le *nizko-nivojske* funkcije LL knjižnice. Funkcije LL knjižnice so implementirane tako, da periferno enoto *upravljajo neposredno na nivoju registrov*. Funkcije HAL knjižnice pa uporabijo *višje-nivojski* pristop in vpeljejo uporabo "handle" strukture za upravljanje periferne enote. Če pogledate v datoteko `main.c`, boste našli odsek, kjer so definirane vse "handle" strukture, ki jih potrebujemo, če želimo upravljati periferne enote s pomočjo HAL knjižnice. Poglejte izsek spodaj.



HAL knjižnica uporablja idejo "handle" strukture na podoben način, kot smo "handle" strukture uporabljali mi pri implementaciji sistemskih modulov Miškota.

O kalibraciji osi "joysticka" in izračunu *relativne* pozicije osi

"Joystick" je *vhodna naprava* (angl. input device), ki omogoča vnos informacije s pomočjo pozicije oziroma orientacije krmilne ročice. Krmilna ročica se pri dvo-osnemu "joysticku" nahaja v nevtralni centralni legi in jo lahko odmaknemo v dveh smereh, ki jih tipično označujemo kot osi koordinatnega sistema in jih zato poimenujemo X in Y (glejte sliko spodaj).

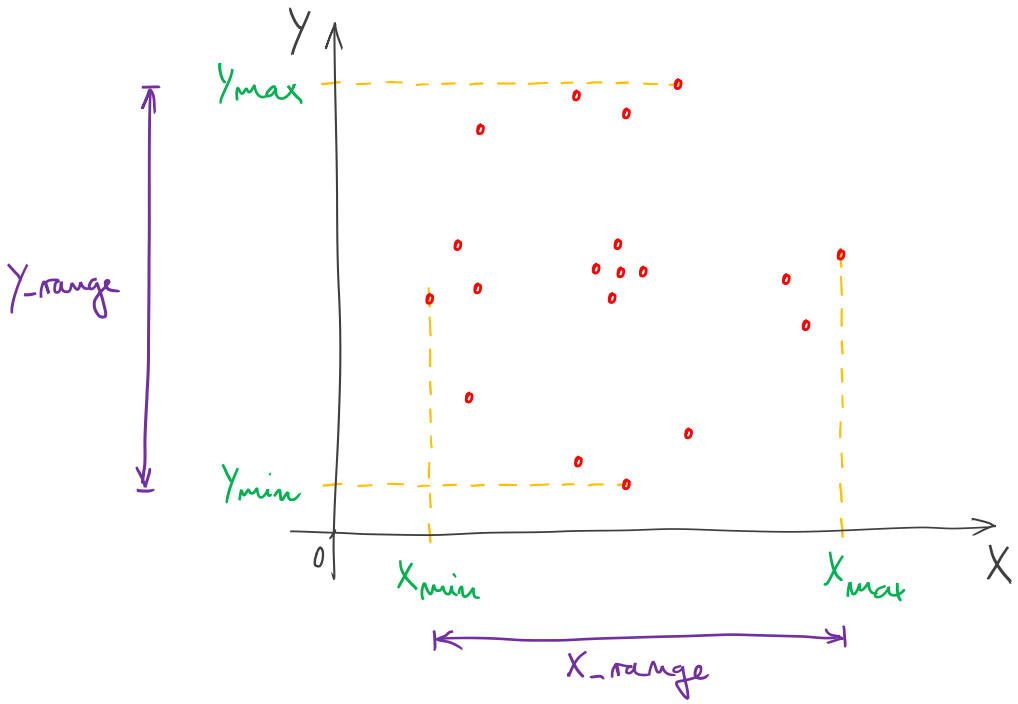


Detekcija odklona ročice vzdolž posamezne osi (angl. axis, množina angl. axes) je izvedena s pomočjo potenciometrov (glejte sliko zgoraj). "Joystick" je v smislu električnega vezja zgolj par spremenljivih uporovnih delilnikov. Sedaj pa premislimo, kaj je potrebno storiti pri procesu kalibracije osi "joysticka".

Kalibracija osi "joysticka"

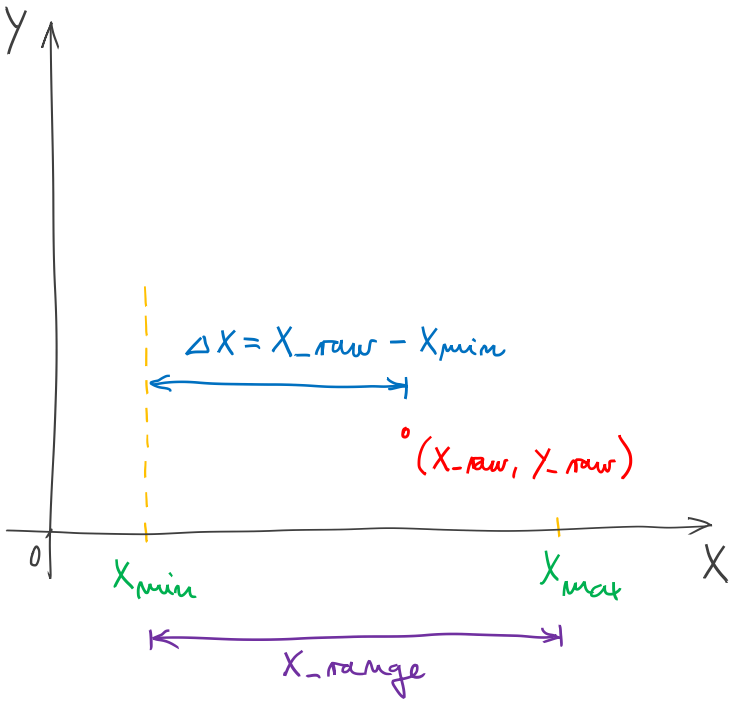
Pri kalibraciji osi "joysticka" želimo pravzaprav ugotoviti *ekstremni vrednosti* analognega signala, ki ga proizvede potenciometer posamezne osi, torej *minimalno* in *maksimalno* vrednost signala. Na ta način *dobimo informacijo, v katerem območju se bodo gibale meritve pozicije osi "joysticka"*, ko bomo "joystick" uporabljali. Tekom kalibracije je zato potrebno ročico "joysticka" potiskati *v skrajne lege* in *med tem beležiti vrednosti skrajno ležečih meritev*.

Idejo kalibracije ponazarja spodnja skica. Rdeče točke v koordinatni ravnini predstavljajo "*surove meritve*" (angl. raw measurements) pozicij obeh osi "joysticka". Pod terminom "surove meritve" mislimo vrednost, ki jo dobimo ob analogno-digitalni pretvorbi analognega signala. Ker imamo v našem primeru opravka z 12-bitnim AD pretvornikom, lahko torej za surove meritve pričakujemo ne-predznačene celoštevilske vrednosti na intervalu od 0 do 4095. Tekom kalibracije želimo poiskati vrednosti meritev v skrajnih legah osi in tako določiti parametre `X_MIN`, `X_MAX` ter `Y_MIN`, `Y_MAX`. Ko so skrajne lege osi znane, pa lahko določimo tudi *razpon odklona* posameznih osi (angl. range), torej parametra `X_RANGE` ter `Y_RANGE`. Vse te parametre bomo potrebovali, ko bomo želeli izračunati trenutno relativno pozicijo osi "joysticka".



Izračun *relativne* pozicije osi

Sedaj pa poglejmo še, kako iz *trenutne surove meritve pozicije* osi ob pomoči kalibracijskih parametrov določimo *trenutno relativno pozicij*o osi. Razložimo to na primeru meritve X osi s pomočjo spodnje skice.



Trenutni *absolutni odklon osi* ΔX določimo kot razliko med trenutno "surovo" meritvijo odmika ter minimalno vrednostjo odmika,

$$\Delta X = X_{\mathrm{RAW}} - X_{\mathrm{MIN}}.$$

Trenutni *relativni* odklon osi pa dobimo, če ta *absolutni* odklon izrazimo kot *procentualni delež* razpona celotnega odklona osi, torej

$$\Delta X_{\mathrm{REL}} = \frac{\Delta X}{X_{\mathrm{RANGE}}} \cdot 100\ \% = \frac{X_{\mathrm{RAW}} - X_{\mathrm{MIN}}}{X_{\mathrm{RANGE}}} \cdot 100\ \%$$

In taka relativna pozicija osi "joysticka" je informacija, ki jo je smiselno vpeljati pri delu z "joystickom". V nevtralni legi tako pričakujemo vrednosti relativne pozicije v okolici 50, v skrajnih legah pa vrednosti okoli 0 oziroma 100.