Viikko 1: setup()-funktio

Johdanto viikko 1:

Kun projekti on oikeaoppisesti luotu xilinx SDK:ssa, viikolla on tarkoitus määritellä colors shieldin dm163 LED-ohjaimen käyttämien ohjausrekisterien muistiosoitteet koodissa ja tutustua bittimanipulointitapoihin ohjausbittien asettamiseksi näissä rekistereissä.

Tällä viikolla tutustutaan ohjausrekistereihin ja käytetään erityisesti ”control signals”-rekisteriä josta muun muassa käytetään raudan resetoinnin määrittävää bittiä, datalinja- ja sarjakellobittiä. Kaksi jälkimmäistä toimivat sarjamuotoiseen tiedonsiirtoon LED-ohjaimen kahteen siirtorekisteriin, jotka määrittävät kirkkauden sekä värit matriisissa.

Myös ”channel”-rekisteri on tarkoitus asettaa nollaksi. Jälkimmäiseen soveltuu niin sanottu suora asettaminen ilman bittimanipulaatiota.

Erityisesti hieman myöhemmin harjoitustyössä on oleellista omaksua ainakin jokin tapa yksittäisen bitin muuttamiseksi rekisterissä ylös tai alas niin etteivät muut bitit muutu. Niiden sisältö ei ole välttämättä tarkalleen tiedossa tai ovat sillä tavoin tilanneriippuvaisia, ettei voida aina olettaa kirjoitettavaksi niitä kaikkia kerralla. Bittimanipulaatioon kannattaa siis kiinnittää jo nyt huomiota ja harjoitella ensimmäisestä funktiosta lähtien.

Setup () -funktion tarkoituksena on alustaa määritellyt ohjausrekisterit nollatilaan. Resetoida dm163 käyttäen kontrollirekisterin resetointibittiä ja kirjoittaa 6-bittinen siirtorekisteri täyteen bittiä 1, täyden kirkkauden saavuttamiseksi ledeille. Lähtökohtaisesti rekisteri olisi täynnä nollia ja matriisiin oikeallakaan tavalla kirjoittaminen ei toisi haluttua lopputulosta.

Setup()-funktio ajetaan kerran ja vain kerran main.c tiedostossa.

Ohjeistukset:

Projektin luonti:

COMP.CE.100_exercise_guide.pdf sivut: 5–8.

Muistiosoitteiden määrittely ja käyttö:

Osoitteet, bittijärjestys ja niiden lyhenteet:

COMP.CE.100_exercise_guide.pdf sivu: 12.

DM163.pdf ohjausbittien toiminnallisuutta sivut: 2–6.

Lyhenteiden sanastoa:

harjotustyö_lisämateriaalia_v1.pdf sivu: 8.

C-kielinen määrittely ja käyttö:

COMP.CE.100_exercise_guide.pdf sivut: 10(define to word) ja 18 (pointers).

Käytä näistä toista.

Ohjausrekisterien fyysiset lähdöt:

Project_work_info_fall_20.pdf sivut: 17–20.

Bittimanipulaatio:

COMP.CE.100_exercise_guide.pdf sivut: 19–21. (AND, OR, XOR, NOT).

Sarjatietoliikenteen luominen ja 6-bittinen rekisteri:

Sarjamuotoinen tietoliikenne:

COMP.CE.100_exercise_guide.pdf sivut: 14.

harjotustyö_lisämateriaalia_v1.pdf sivu: 9.

6-bittiseen ”gammakorjaus”-rekisteriin kirjoittaminen:

DM163.pdf sivut: 3, 7 (CLK,SDA,SB).

colorsshield.pdf sivu: 3.

setup()

1. Aseta kontrolli- ja kanavarekisteri nollaksi. Mikä merkitys on SB-bitin nolla asennolla?
2. Ohjain on resetoitava aluksi. Mitä nyt tarkoittaa asynkroninen resetti?
3. Käytä komentoa usleep(500) rekisterimuutosten välissä, kun käytät rst-bittiä toiminnallisessa arvossa ja pois. Tämä on odotuskäsky mikrosekunneissa. Ei sovellu koodin hidastamiseksi muualla, siihen palataan keskeytyksissä.
4. 6-bittiseen rekisteriin kirjoitetaan 144 ykköstä. Mistä tämä luku muodostuu? Kuinka montaa lediä voimme ohjata yhtä aikaa? Entä 3 ledin RGB-lediä ?
5. Kirjoita 6-bittiseen siirtorekisteriin joko yhdellä tai kolmella for loopilla. Tulet tarvitsemaan 8-bittisen kanssa myöhemmin kolmea.
6. Vaihda lopuksi kontrollirekisterin kautta kirjoituskohteeksi 8-bittinen siirtorekisteri.

Pseudokoodivinkkejä gammakorjausrekisteriin kirjoittamiseen:

Yhdellä loopilla:

Mieti minkä ohjausbitin arvo kirjoitetaan rekisteriin ja milloin. Toista riittävästi.

Kolmella loopilla:

Ennen for loop kokonaisuutta määrittele vektori sisällöllä {63, 63, 63}.

Miten sama määriteltäisiin binaarina tai heksadesimaalina?

Tarvitset neljä apumuuttujaa esim. uint8_t i, a, b, t;

for i rgb-ledit

    for a värit

        t=vektori[a]

        for b 6-bittiä

            if(t & 0b100000) -> kirjoita rekisteriin 1

else -> kirjoita rekisteriin 0
shiftaa apumuuttujaa t <<

Kysymykset ovat johdattelevia ja niiden vastauksien etsiminen on itseä varten. (Ei arvostella)

Viikon jälkeen on tarkoituksena *ymmärtää ja osata*:

Ohjausrekisterien bittien merkitykset ja käyttö bittimanipulaatiolla. AND, OR, XOR, NOT, shiftaus, jne. 0b ja 0x alkuliitteet ja niiden merkitys.

Sarjamuotoinen tiedonsiirto ja siirtorekisteriin kirjoittaminen. Mikä kirjautuu ja milloin? Missä järjestyksessä täyttyy?

Viikon jälkeen on *valmiina*:

Projekti on onnistuneesti luotu xilinxiin.

Ohjausrekisterien muistiosoitteet on määritelty käyttöä varten.

Ohjausrekisterit on nollattu alussa.

Setup-funktio on valmis, joten ohjain on resetoitu, 6-bittinen siirtorekisteri on kirjoitettu täyteen ja kirjoituskohde on vaihdettu 8-bittiseen.

Jos ei ole, koko projektiin saa aina ohjausta tunneilla, vaikka ei olisi kyse viikon teemasta, mutta tarkoitus olisi tehdä valmiiksi syysiltoja ja mattermostia hyväksikäyttäen ennen seuraavaa harkkatuntia… Joka palasta lisää pisteitä!