Autotallin ovi - Teoria

Sisällysluettelo

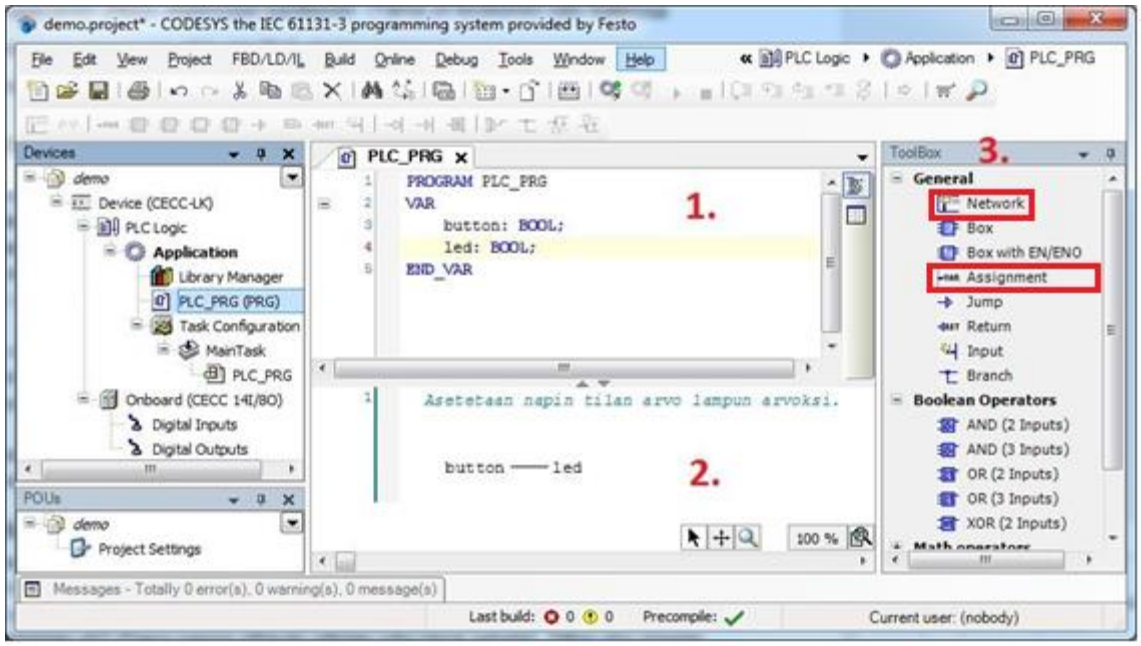
Oppimistavoitteet:

- FBD (Function Block Diagram) kielen alkeet
- Systemaattisen testauksen perusteet

FDB-editorin käyttö CODESYS-ohjelmointiympäristössä

CODESYS-ohjelmointiympäristön toimilohkoeditori jakautuu kahteen osaan, samalla tavalla kuin ST-kielen editori. Editorin yläikkunassa määritellään muuttujan (1) ja alaikkunassa muodostetaan ohjelman toimilohkologiikka (2). Toimilohkoja lisätään toimilohkoeditoriin vetämällä ne oikealla olevasta työkalupalkista, ToolBoxista (3).

FBD-kielessä ”Network” toimii perustana toimilohkoille. FBD-kielen ohjelma suositellaan kirjoittaa monelle riville, jotta se olisi selkeä. Uusia rivejä lisätään ohjelmaan vetämällä työkalupalkista general- välilehden alta ”Network” halutun rivin ylä- tai alapuolelle. Yksinkertaisin toimilohko-ohjelma koostuu vain sijoitusoperaatiosta, joka löytyy työkalupalkista general- välilehdeltä nimellä “Assignment” (Kuva 1).

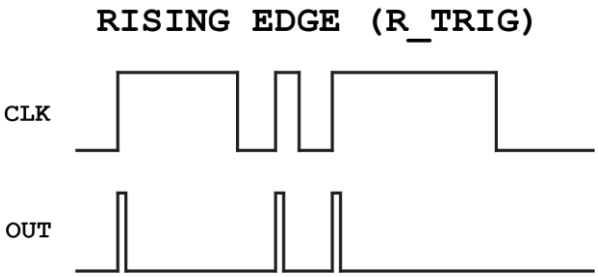


Kuva 1: FBD -editori

Autotallin ovea toteuttaessa kannattaa käyttää apuna MyCourses-sivusta löytyvää ”Opas toimilohko-ohjelmointiin” dokumenttia.

Autotallin oven ohjelmoiminen ja toimilohkojen käyttö

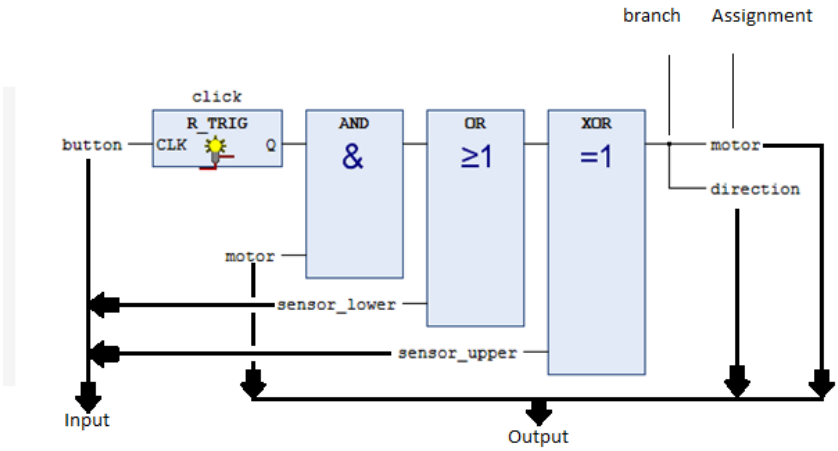
Autotallin oven on reagoitava napin painallukseen heti sen ollessa pohjassa. Tähän sopiva funktio on R_TRIG- lohko. R_TRIG lohkossa sisäänmenon nouseva reuna aikaansaa ulostulolle lyhyen pulssin. Ulostulo ei siis riipu siitä, kauanko sisäänmeno on aktiivinen, vaan ainoastaan sisäänmenon nousevasta reunasta.



Kuva 2: R_TRIG (PLC Academy)

R_TRIG - lohko löytyy CoDeSysistä Toolboxista kohdasta Function Blocks, josta se raahataan toimilohko editoriin. Tällöin luodaan instanssi lohkosta. Instanssille pitää myös antaa nimi muuttujiin. Tämä instanssinimi on siis muuttuja, jonka tyyppi on kyseisen toimilohkon nimi, joten tämä muuttuja pitää ilmoittaa muiden muuttujien tavoin ohjelman (PLC_PRG) VAR osassa seuraavalla syntaksilla: <instanssin nimi>:<toimilohkon nimi>;

Toimilohkoeditorin käyttöä on helppo lähestyä ajatuksella, että kaikki sisäänmenot tulevat vasemmalle ja ulostulot oikealle. Väliin vedetään tarvittavat toimilohkot, joiden avulla sisääntulon signaalit muutetaan halutuiksi ulostuloiksi. Huomaa, että ulostuloa on mahdollista käyttää myös sisäänmenosignaalina. Tätä voi tietyltä osin verrata Automaatio- ja systeemitekniikan perusteet kurssilta tuttuun takaisinkytkettyyn järjestelmään. Alla on kuvattu **mielivaltainen** esimerkki toimilohkoeditorin käytöstä.



Kuva 3: Esimerkki toimilohkoeditorin käytöstä

On hyvin yksilöllistä, miten editorin toiminta valkenee, kun sitä hiukan kokeilee. Kysy heti, jos jäät jumiin! C-kielen IF lausetta vastaavan toiminnallisuuden voi toteuttaa esimerkiksi XOR-lohkolla.

Testit

Testi määritellään siten että järjestelmän on oltava tietyssä tilassa testin alkaessa, minkä jälkeen sen syötteisiin (käyttöliittymälaitteet tai anturit) vaikutetaan tietyllä tavalla. Testaus on kattavaa, jos kaikissa mahdollisissa tiloissa testataan kaikki mahdolliset syötteiden kombinaatiot. Tässä harjoituksessa se on mahdollista, mutta isommissa sovelluksissa se on joko mahdotonta tai liian kallista. Testisuunnitelma on dokumentti, jonka perusteella kuka tahansa voi suorittaa samat testit. Siinä myös etukäteen määritellään, että miten järjestelmän pitäisi reagoida kuhunkin testiin. Testisuunnitelman voi kirjoittaa määrittelyjen perusteella ennen kuin ohjelmointia on aloitettu. On jopa parempi, että testien kirjoittaja ei tiedä miten ohjelmisto on toteutettu, koska tämä tietämättömyys estää ohjelmoijan ajatusvirheiden siirtymisen testisuunnitelmaan.