I2C-Drivrutin

Författare: Alexander Solman Andersson
Mail: alexander.solmanandersson@yh.nackademin.se
Mjukvaruutvecklare, Inbyggda system och IoT

Innehållsförteckning

Inledning

Denna dokumentation förklarar processen i utvecklingen av denna I2C-drivrutin och stegen som tagits för att nå slutmålet samt komplikationerna under utvecklingen.
För att säkerställa att drivrutinerna är i funktionsdugliga så använder vi oss utav SSD1306 OLED skärmen.

Projekt uppkomst

Detta projekt skulle initialt utvecklas med hjälp utav proteus, men då demo varianten utav denna mjukvara inte fungerar för att simulera så har projektet fått göras hypotetiskt och baserat på referens material samt annan dokumentation som finns bifogat.

Drivrutins utveckling

Vid utveckling utav drivrutiner så bör man bekanta sig med relevant dokumentation för det man utvecklar emot. Om det är stm32 mikrokontroller så finns dokumentation på tillverkarens hemsida. Man bör också försäkra sig om att man är införstådd i hur komponenterna fungerar så att konfigurering emot det protokoll och enheter man använder blir korrekt.
Då vi har valt att använda oss utav I2C protokollet så finns den mesta informationen om hur detta bör sättas upp under kapitlet för I2C.

I2C.h

Drivrutinen som vi använder oss utav här har tre stycken funktioner som är nödvändiga för att nyttja den i fall där man behöver upprätta kommunikation med detta protokoll för envägs kommunikation. Denna drivrutin gör det inte möjligt för användaren att läsa data från enheter utan bara att skriv till dem. Den första funktionen som hittas är uppsättningen och där förbereder vi pins för SDA och SCL vilka är de två pins vi behöver för I2C kommunikation. Vi säkerställer att vi har tillgång till klockfrekvens samt att dessa konfigureras till I2C genom ”alternate function mode”.
Efter det går vi in mer på I2C själv och vi konfigurerar då hastigheter för att enheterna ska matcha i dataöverföringen. Vi gör det också möjligt för oss att hantera fel som kan uppstå under kommunikation.
Möjligheten att kunna skriva till enheter finns via skriv funktionen som vi implementerat och ordningen som görs skiljer ofta protokollen lite åt. Vi ser till att kommunikations bussen inte är i användning innan vi försöker skicka data. Vi inleder sedan kommunikation och meddelar att vi vill skriva samt inväntar svar på att vi kan skicka data följt utav ett stop. Skulle ett fel inträffa under kommunikationen så har vi även en funktion som kan hantera detta. Efter kommunikationen så kollar vi om det finns flaggor som rests. Dessa är acknowledge fail, arbitration lost och bus error och det vi gör om flaggorna för dessa är satta är att om initiera kommunikationen och försöka skicka data igen.

Medföljande program

Till denna drivrutin så finns det två stycken filer bifogade. Dessa är till för att kunna säkerställa funktionen utav drivrutinen. Den ena filen testar att köra koden samt uppvisar hur användandet av drivrutinen sker. Det finns även en fil för att initiera SSD1306 enheten som använder sig utav drivrutinens skriv funktion. Det finns också ”dummy” kod som innehåller hexadecimala värden som skrivs till ssd1306 enhetens RAM som kan visa datumet 2023.

Testning och problem

Då det inte funnit möjlighet till simulering eller hårdvara för att kunna testa mjukvaran som är skapt så finns det väldigt lite testning gjord. Koden är jämförd med tillverkarens dokumentation samt betrodda figurer inom branchen.

Det har inte funnits speciellt många problem under utvecklingen då det inte går att testköra koden för att se om logiken är rätt. Däremot så har det funnit problematik att kunna finna all information angående vilka register som behöver beröras samt hur man implementerar I2C. Samt har det funnits problematik kring proteus och även att finna bibliotek för att kunna hantera fel och komma åt NVIC registret i stm32.

Dagbok

Dag 1: Under dag ett så testades utvecklingsverktygen som skulle användas samt så undersöktes möjligheterna för vilka projekt som kunde genomföras. Efter att ha bekantats med dokumentation och möjligheterna till att kunna simulera eller införskaffa sig hårdvaran så landade det i att varken simulering eller införskaffning skulle vara möjligt. När projekt iden hade satts så skapades strukturen lokalt och arbetet började.

Dag 2: Påbörjandet av drivrutinerna för I2C gjordes och uppsättning av GPIOB blev klart. Förberedande arbete skedde genom att studera ”Phil’s lab” uppsättning av I2C.

Dag 3: Initierings funktionen för I2C avslutades under dagen samt strukturen för skriv funktionen skapades. Omstrukturering av kod och ytterligare kommentarer gjordes då det saknade relevant förklaring till vad som skedde.

Dag 4: Under dagen så avslutades hela filen för I2C och det mesta arbetet gick till att implementera fel hantering samt funktionen för detta gjordes. Skriv funktionen avslutades också och alla binära tal/hexadecimal tal fick macros istället.

Dag 5: Efter en hel förmiddag för att förstå hur ssd1306 skulle initieras så lyckades jag få till command-arrays för att initiera samt att preppa datan som skulle skrivas till dess RAM. Den drivande funktionen avslutades också för att få ett komplett test program.