Opdrachten week 5 les 2 – Libraries en toestandsmachines

In de vorige les is aandacht besteed aan het werken met functies in C en heb je geleerd om de interne klokfrequentie van de MSP430G2553 in te stellen. Ook heb je in de vorige les geleerd hoe je een knop kunt ontdenderen.

In deze les leer je:

- hoe je functies in een apart bestand kunt plaatsen waardoor je ze eenvoudiger kunt hergebruiken;
- hoe je een private git repository aan kan maken op bitbucket om je programma's in op te slaan;
- hoe je verschillende versies van je programma kan beheren met behulp van git en CCS;
- een toestandsmachine te implementeren als een programma in een microcontroller.

In de vorige les heb je verschillende functies geschreven. De functies `zet_led_aan`, `zet_led_uit` en `knop_is_ingedrukt` verbergen de details van de bitwise-operatoren maar zijn niet erg gemakkelijk herbruikbaar in andere programma's omdat er in deze functies van wordt uitgegaan dat de leds en de drukknop op bepaalde pinnen zijn aangesloten. Ook wordt er vanuit gegaan dat deze pinnen correct geconfigureerd zijn en dat de drukknop in een pull-down configuratie is aangesloten.

In de vorige les heb je ook de functie `zet_klok_op_MHz` geschreven. Deze functie verbergt de complexiteit van het zetten van de frequentie van de interne klokgenerator van de MSP430-G2553. Deze functie is herbruikbaar in andere programmma's voor de MSP430G2553. Om deze functie eenvoudig herbruikbaar te maken kunnen we deze functie in een apart bestand plaatsen. Aan dit bestand kunnen we dan nog meer herbruikbare functies toevoegen. Zo'n verzameling van meerdere herbruikbare functies wordt in het boek[1] een library genoemd.

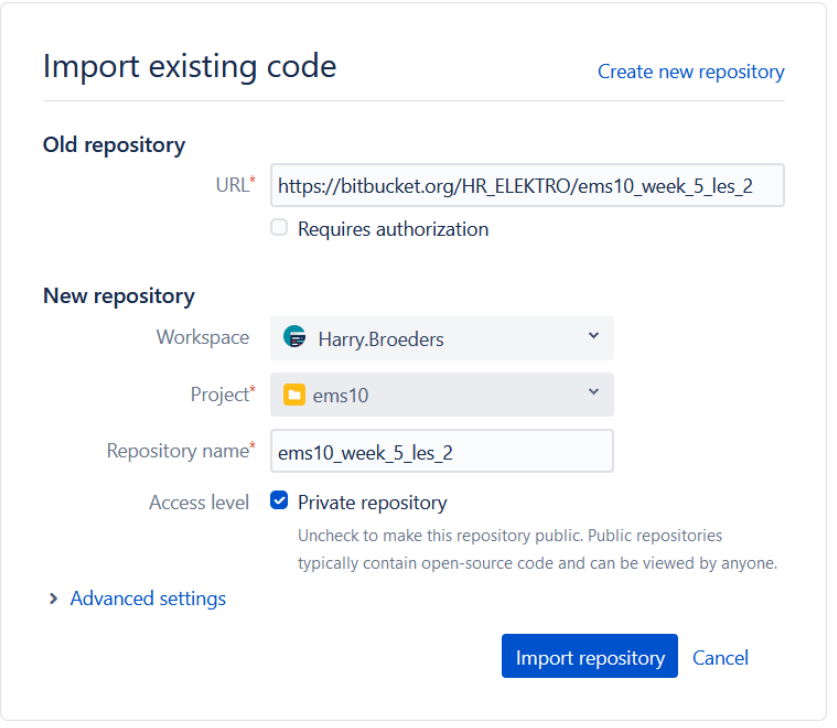
Lees paragraaf 3.1 en 3.2 van het boek.

In deze les leer je hoe een kleine library kunt maken met daarin functies om de GPIO pinnen van de MSP430G2553 aan te sturen en uit te lezen. We gaan deze library beheren in een git repository zodat we de verschillende versies van de herbruikbare functies altijd terug kunnen vinden. Een eerste opzet van deze library hebben wij al voor je gemaakt en op bitbucket beschikbaar gemaakt.

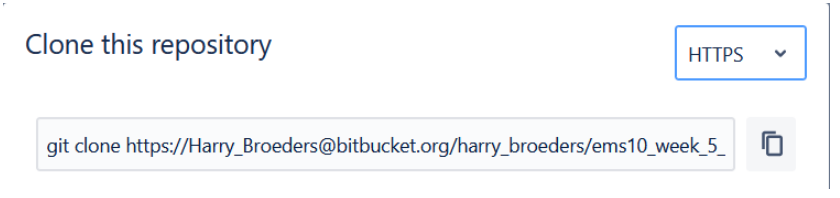
[1] Carl Burch. *C for Python programmers*. 2011. URL: http://www.cburch.com/books/cpy/

Volg de onderstaande instructies nauwkeurig op:

- Ga naar https://bitbucket.org/ en klik op 'Get it free' en vervolgens op 'Next'.
- Klik op de knop "Ga verder met Microsoft".
- Meld je aan met je e-mailadres van school.
- Log in op je schoolaccount.
- Microsoft meldt dat Atlassian (het bedrijf achter Bitbucket) machtigingen heeft aangevraagd. Klik op de knop "Accepteren".
- Bitbucket stuurt je nu een mailtje om je e-mailadres te bevestigen.
- Voer de verificatiecode uit de mail in en klik op de knop 'Verifieer je account'.
- Er wordt nu gevraagd om een workspace aan te maken. Kies een naam voor je workspace bijvoorbeeld je eigen naam en klik op de knop 'Create'.
- Klik vervolgens op de knop 'Import repository'.
- Vul de volgende URL in:
  `https://bitbucket.org/HR_ELEKTRO/ems10_week_5_les_2`
- Vul de Project name in: `ems10`
- Vul de Repository name in: `ems10_week_5_les_2` en geef aan dat je een private repository aan wilt maken, zie figuur 1.
- Klik op de knop 'Import repository'. Hiermee maak je een eigen kopietje van het door de docenten aangemaakte repository, waarin je jouw eigen code kunt opnemen en beheren. Dit wordt in het jargon van git ook wel een 'Fork' genoemd. Gefeliciteerd. Je bent nu de eigenaar van je eigen git repository op bitbucket!
- Dit repository bevindt zich in de cloud. Om dit repository te kunnen gebruiken in Code Composer Studio moet je er een lokale copy van maken op de machine waarop je CCS draait. Dit wordt in het jargon van git een 'Clone' genoemd. Je kan op verschillende manieren zo'n clone aanmaken. Wij doen dit vanuit CCS en hebben daarvoor een git clone commando nodig dat je kunt vinden door op de knop Clone te klikken (rechts boven). In mijn geval is dat: `git clone https://Harry_Broeders@bitbucket.org/harry_broeders/ems10_week_5_les_2.git`. Je kunt dit commando eenvoudig kopiëren met het knopje aan de rechterkant, zie figuur 2.
- Start nu Code Composer Studio (CCS).
- Selecteer het menu Window » Perspective » Open Perspective » Other... en kies vervolgens voor Git en klik op 'Open'.
- Klik op 'Clone a Git repository'.

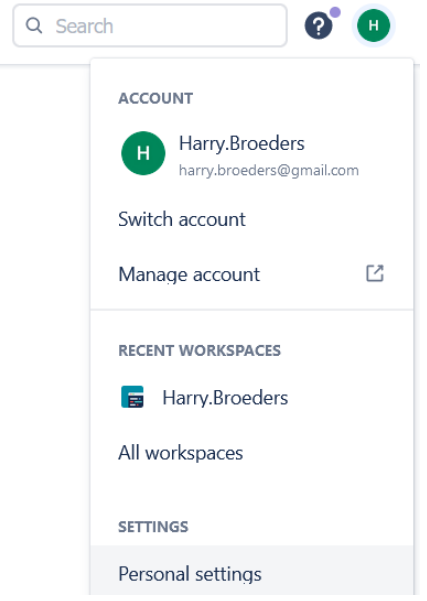


Figuur 1: Het importeren van een repository.

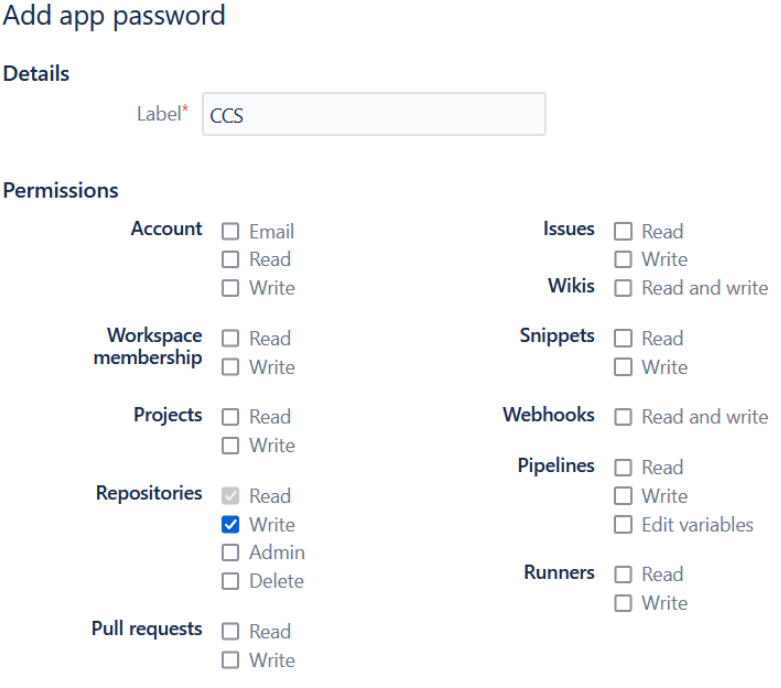


Figuur 2: Met het knopje aan de rechterkant kun je het commando eenvoudig kopiëren.

- Plak het zojuist gekopieerde git clone commando achter URI:.
- Je moet nu eerst een zogenoemd App password aanmaken in bitbucket om dit repository te kunnen clonen. Ga naar bitbucket en log in (indien nodig). Klik op het 'bolletje' rechtsboven en kies voor 'Personal settings', zie figuur 3.
- Klik op 'App passwords' en vervolgens op 'Create app password'. Vul een label in, bijvoorbeeld: 'CCS' en zet een vinkje bij 'Repositories' 'Write', zie figuur 4.
- Klik op de knop 'Create' en kopieer het aangemaakt App password.

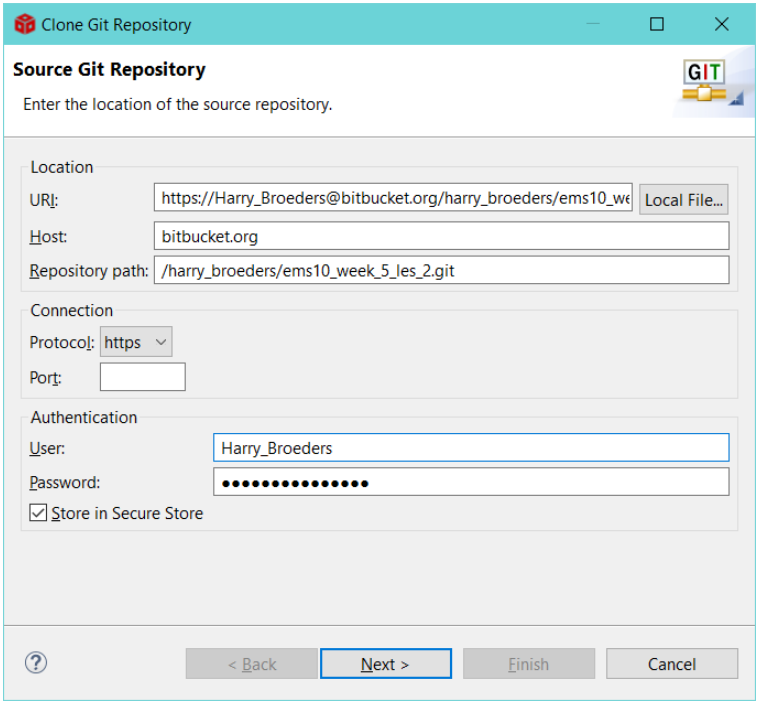


Figuur 3: Kies 'Personal settings' om een App password aan te maken.



Figuur 4: Het aanmaken van een App password.

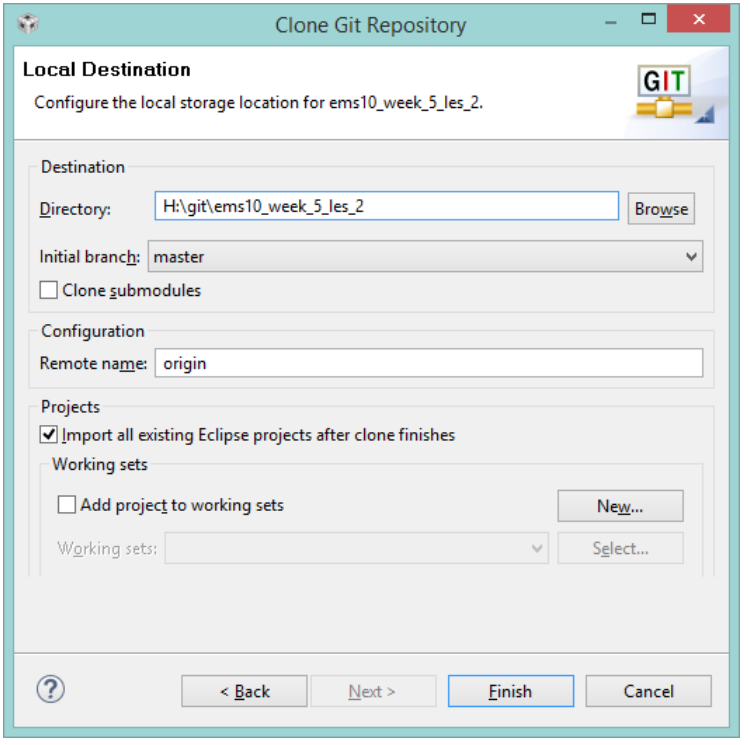
- Ga nu weer naar CCS en vul nu het zojuist op bitbucket aangemaakte App password in, zie figuur 5. Als je niet steeds je App password in wilt voeren als je het repository wilt bijwerken, dan kun je een vinkje zetten bij 'Store in Secure Store'. Klik vervolgens op 'Next'.



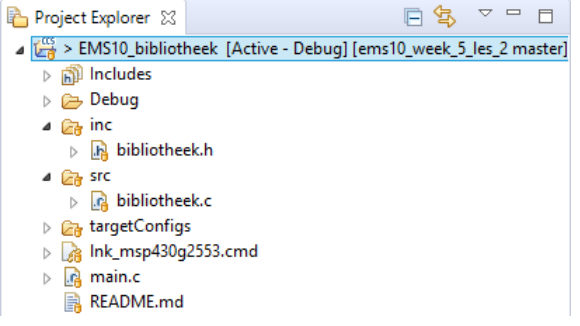
Figuur 5: Het toevoegen van een Git repository in CCS.

- Klik nogmaals op 'Next'.
- Kies een geschikt 'directory' (bijvoorbeeld thuis op je C-schijf of op school op je H-schijf).
- Zet een vinkje bij 'Import all existing Eclipse projects after clone finishes', zie figuur 6.
- Klik vervolgens op 'Finish'.
- Selecteer het menu Window » Perspective » Open Perspective » Other... en kies vervolgens voor CCS Edit en klik op OK.
- Als het goed is, zie je nu het project `EMS10_bibliotheek`. Als je dit project open klikt, dan zie je dat het twee `.c`-files en een `.h`-file bevat, zie figuur 7.

In het bestand `bibliotheek.h` zijn de prototypes van een aantal herbruikbare functies gegeven. Open dit bestand en bestudeer de code. Er staan een aantal dingen in die waarschijnlijk nieuw voor je zijn.



Figuur 6: Het toevoegen van een Git repository in CCS.



Figuur 7: De inhoud van het project `EMS10_bibliotheek`.

- De regels 8, 9 en 61 bevatten zogenaamde preprocessor directives die ervoor zorgen dat er geen fouten ontstaan als deze headerfile per ongeluk meerdere keren ge-include wordt. Elke headerfile dient zo'n zogenoemde include guard te bevatten.
- In tegenstelling tot wat het boek beweert kent de programmeertaal C, sinds 1999, wel een datatype `bool`. Zie http://en.cppreference.com/w/c/types/boolean. Op

regel 11 wordt de standaard includefile `stdbool.h` ge-include zodat het datatype `bool` en de booleaanse constanten **`false`** en **`true`** in de bibliotheek gebruikt kunnen worden.

- Op regel 12 wordt de standaard includefile `stdint.h` ge-include. **Lees nu bijlage B van het handboek**[2].
- Op regel 29 wordt het enumeratietype `Richting` gedefinieerd. Variabelen van dit type kunnen de in de **`enum`** opgesomde waarden bevatten. In dit geval wordt het type `Richting` gebruikt op regel 30 als type van de laatste parameter van de functie `zet_pin_richting`. Bij het aanroepen van deze functie kunnen we dus als laatste argument de waarde `input` of `output` meegeven.

In het bestand `main.c` is de applicatiecode die gebruik maakt van de functies uit de bibliotheek gegeven. Open dit bestand en bestudeer de code. Begrijp je, met behulp van de informatie uit de file `bibliotheek.h`, wat de functionaliteit van deze applicatiecode is?[3]

Zoek in het boek op wat de C operatoren zijn voor de logische NOT, OR en AND bewerkingen.[4]

Lees nu paragraaf 2.4 punt 9 in het boek.

In het bestand `bibliotheek.c` zijn de implementaties van de in `bibliotheek.h` gedeclareerde functies gegeven. Ze zijn echter nog niet allemaal ingevuld. Open dit bestand en bestudeer de code. Merk op dat de functie `zet_pin_richting` niets doet als er ongeldige argumenten meegegeven worden.

Als het goed is, kun je nu alle code die gegeven is in `bibliotheek.c` begrijpen. Vraag je docent om uitleg als dit niet zo is.

5.2.1 Vul de ontbrekende code in, in het bestand `bibliotheek.c`, zodat de applicatiecode werkt zoals bedoeld is. Het gewenste gedrag van de functies is beschreven in het bestand `bibliotheek.h`. Zorg ervoor dat de functies niets doen als er ongeldige argumenten meegegeven worden.

---

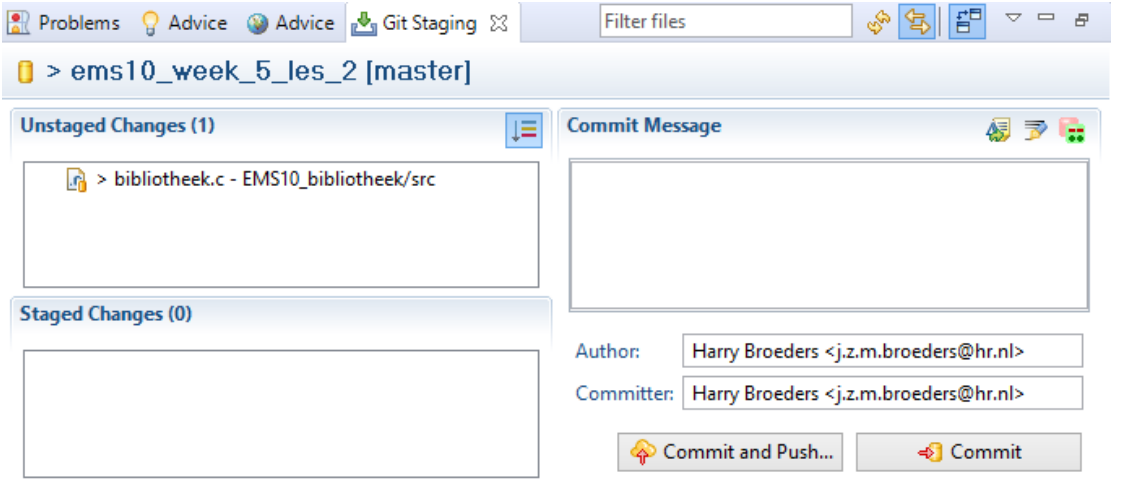
[2] Daniël Versluis. *Microprocessor Programmeren in C*. 2018. URL: https://bytebucket.org/HR_ELEKTRO/ems10/wiki/Handboek/EMS10_handboek_ebook.pdf.

[3] Zolang de drukknop aangesloten op pin P1.0 wordt ingedrukt, brandt de led op pin P2.1.

[4] Respectievelijk `!`, `||` en `&&`. Let er goed op dat je de logisch OR-operator `||` niet verward met de bitwise OR-operator `|`. Let er ook goed op dat je de logisch AND-operator `&&` niet verward met de bitwise AND-operator `&`.

Je kunt de wijzigingen in het project nu vanuit CCS committen en naar bitbucket pushen. Volg de onderstaande instructies nauwkeurig op:

- Klik met je rechtermuisknop op de projectnaam in de 'Project Explorer'.
- Kies vervolgens voor 'Team' en 'Commit'.
- Rechtsonder verschijnt nu het 'Git Staging' window, zie figuur 8.



Figuur 8: Het 'Git Staging' window in CCS.

- In dit geval is er slechts één file gewijzigd. Je kunt deze file van het 'Unstaged Changes' window naar het 'Staged Changes' window slepen zodat ze ge-commit kunnen worden.
- Vul een commit message in en klik op 'Commit en Push...' om de wijzigingen te committen en op te slaan op bitbucket.

In de gegeven applicatie moet je de knop de hele tijd ingedrukt houden als je de led wilt laten branden. Deze applicatie kan gebruiksvriendelijker gemaakt worden door flankdetectie toe te passen. Het programma reageert dan alleen op het indrukken van de drukknop (en niet op het loslaten ervan).

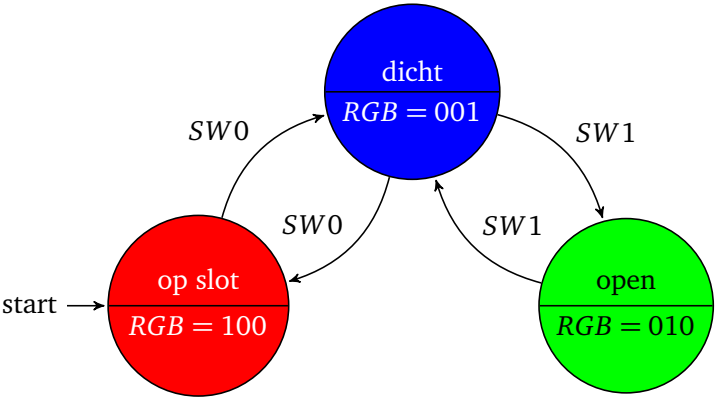
5.2.2 Pas de gegeven applicatie aan zodat de groene led aangaat nadat op de drukknop is gedrukt. Als de knop losgelaten wordt, moet de led blijven branden. Als de drukknop nogmaals wordt ingedrukt moet de groene led doven. Als de knop weer losgelaten wordt, moet de led uit blijven. Enzovoort. Het is nu erg belangrijk dat de drukknop correct ontdenderd is! Begrijp je waarom?

Bij het vak ELE10 heb je geleerd om het gedrag van een digitaal systeem te specificeren met behulp van een toestandsmachine.

Een toestandsmachine kan ook relatief eenvoudig gerealiseerd worden als een C-programma in een microcontroller.

Lees nu hoofdstuk 8 van het handboek.

5.2.3 Implementeer de in figuur 9 gegeven toestandsmachine.



Figuur 9: Eenvoudige toestandsmachine.

Het gaat hier om een toestandsmachine van een deur. De deur kan zich in drie toestanden bevinden:

- op slot, dit wordt aangegeven door de RGB led rood te laten branden;
- dicht, dit wordt aangegeven door de RGB led blauw te laten branden;
- open, dit wordt aangegeven door de RGB led groen te laten branden.

De deur kan alleen geopend worden en/of op slot gedaan worden vanuit de toestand dicht. Door één maal op drukknop $SW0$ te drukken kan de deur van het slot worden gehaald respectievelijk op slot worden gedaan. Door één maal op drukknop $SW1$ te drukken kan de deur geopend respectievelijk gesloten worden. De knoppen moeten dus ontdenderd worden en er moet flankdetectie worden toegepast. Als het programma start, moet de deur zich in de toestand op slot bevinden[5].

[5] Een stap-voor-stap-uitwerking van deze opdracht kun je vinden op de wiki.

Oefening

Je kunt een video bekijken waarin een deel van de stof van deze les wordt uitgelegd, zie tabel 1.

Tabel 1: Video over de stof van week 5 les 2.

Link	Tijd	Beschrijving
Introductie versiebeheer	02:43	Waarom passen we versiebeheer toe bij het ontwikkelen van software?

Verdieping

Een uitgebreider tutorial voor het gebruik van git in Code Composer Studio vind je hier: `https://software-dl.ti.com/ccs/esd/documents/sdto_ccs_git-with-ccs.html`.