HAN E/ESE, ELT-ESE-2C

ECSL Regeltechniek practicum

Opdracht 1

bouw funkties voor impulsresponsie, stapresponsie, hellingresponsie en blokgolfresponsie

Eerste stappen

1. Installeer eerst het meegeleverde programma "RGTControl" met de geleverde Win32 of Apple installer.
2. Laad de STM32 software in CLion (het top level projekt genaamd Practicum_STM32F412DiscoveryBoard of Practicum_STM32L432NucleoBoard met het CMakeListst.txt in de projekt directory)
3. Verander het RGT.CMake bestand op regels 19-22. Maak tot kommentaar (met #) :

   # set(**practicumopdracht DebugOpdracht**)

   and verwijder de kommentaar '#' bij :

   set(**practicumopdracht InputResponsieOpdracht**)

   Doe een CMake Cache reset.
4. CLion herlaad nu de softwarebasis voor opdracht1.

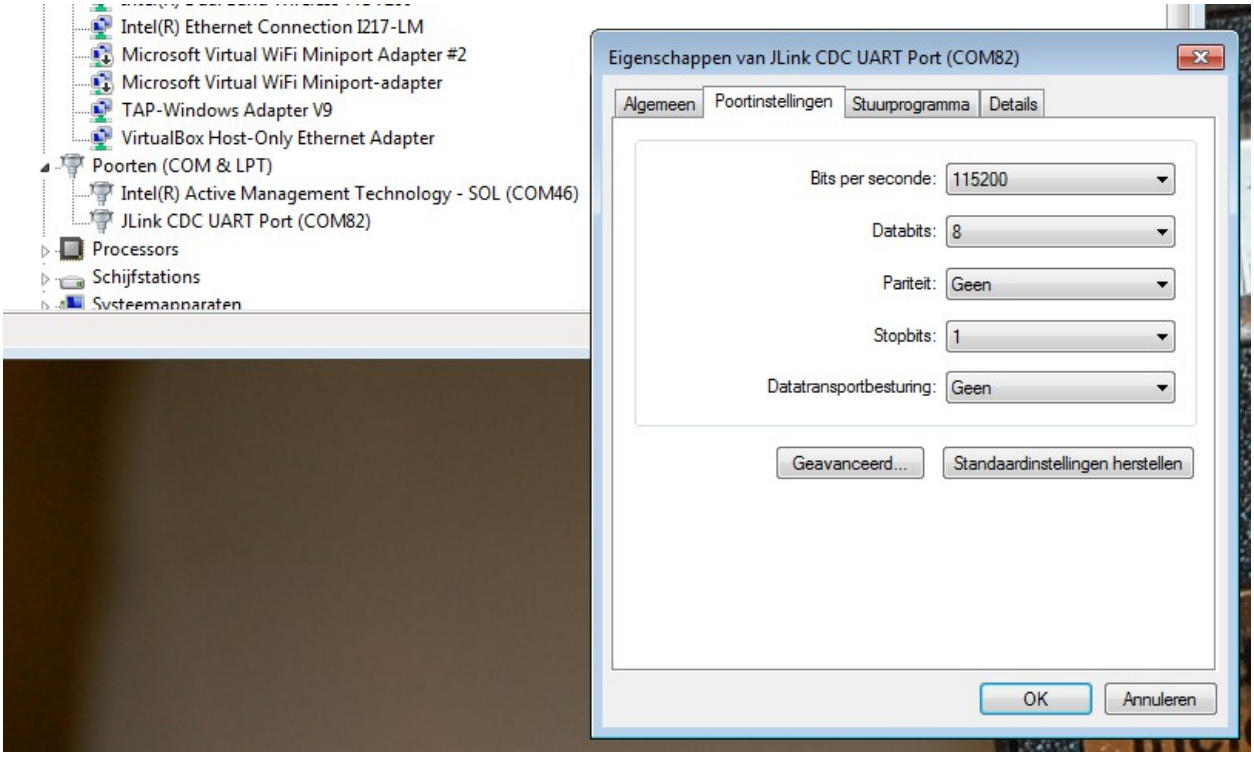
Alle werk voor deze opdracht resideert in het sub projekt "STUDENT" waar het bestand "InputResponsieSessie.cpp" zich bevindt. Jullie oplossing moet worden ingevuld in de lege (met #error gemarkeerde) delen van het bestand. Net zoals in de debugopdracht zijn grote delen van de software voorbereid en jullie behoeven slechts de relevante onderdelen in te vullen.

Geen laag-niveau driver implementatie (om direkt de ADC en DAC aan te sturen) is noodzakelijk. Uiteraard mogen jullie de software doorzoeken om te kijken hoe dingen worden gedaan. CLion toont jullie hoe de klassestrukturen in elkaar zitten en wat de basisklasse is van de klasse waarin jullie werken, namelijk RGTProces.

Verbinding tussen desktop PC en het embedded target

Na dat op de STM32 de software is ingeladen en loopt, kan er geprobeerd worden om een verbinding te maken vanaf de desktop PC.

Let op bij gebruik van het STM32L432NucleoBoard : zorg er op Windows voor dat de serieele poort over de juiste instellingen beschikt :



Stel in als : 115200 BPS, 8 databits, no parity, one stop bit and no control signals.

De opdracht

Er zijn vier signalen die benodigd zijn in regeltechniek experimenten:

- Impuls
- Stap
- Helling
- Blokgolf

In inputresponsie.cpp moeten jullie vier klassefunkties van *StudentInputResponsie* implementeren. Met deze funkties genereer je (op basis van het samplemoment) signalen die naar het proces worden gestuurd.

Alle funkties zijn opgebouwd op de volgende manier:

<signaal waarde in volts> *signaal funktie naam* (sample index als input)

{

…..

```
}
```

Bouw een output in Spanning gebaseerd op de waarde van het bemonsteringsmoment. Gebruik de amplitude van het ingestelde punt met behulp van `getSetPoint()`.

Om te kijken wat de gebruiker gedefinieerde vertraging is, gebruik je de funktie `getStartDelay()`.

Ook moet een geschikte bemonsteringtijd worden ingesteld op de microcontroller. Bepaal eerst een geschikte bemonseringfrequentie. Bestudeer de board documentatie om de proceseigenschappen te kennen, en volg de regel van Shannon theorema om tot een goede frequentie te komen. Daarna moet de frequentie worden toegepast. De gevolgde procedure verschilt per gebruikt practicumbord.

De toegepaste methode kan in de `void RGTProces::setSampleTime()` funktie worden gebruikt. Let op dat in C++ je soms expliciet de klassenaam moet meegeven in een afgeleide klasse om de funktie in de basisklasse te kunnen gebruiken.

De bemonsteringtijd moet overeenkomen met 2Hz of groter, want anders werkt de buffering naar de GUI te traag en toont het GUI programma een time-out.

Het aansturen van de DAC en de ADC hoef je NIET zelf te implementeren, dit is al in de onderliggende software geregeld.

Deze karakteristieken moeten worden geimplementeerd:

- Zet de hellingfunktie op een verloop van 1V/200 monsters

In het geval dat jullie willen weten waar het programma start: het start in main(), zoals gedefinieerd in main.cpp. Lt op dat er een RTOS (FreeRTOS) meedraait, en dat een extra taak (FReeRTOS noemt dit een *thread*) de controle over de LED op het board heeft.

Test en uitvoering

Gebruik de Ozone debugger om de binary in de microcontroller in te laden (gebaseerd op het gegenereerde bestand *ozoneConfig.jdebug* file in de Debug/gegenereerd directory in jullie CLion bouwomegeving. Test de software en indien nodig, herschijf de code en test opnieuw totdat alles naar wens werkt.

Inleveren

- Toon de instrukteur tijdens het practicum een werkende implementatie.
- Schrijf een kort verslag met codestukjes en screenshots (gebruik een tool als MWSnap of op Apple, `+j` `Shift`+4) van jullie werk. Maximale grootte : 2 bladzijden. De samenvatting moet door het team gemaakt zijn en van het pdf type zijn – geen andere formaten worden geaccepteerd. Geef het verslag uitsluitend de naam: **<nickname>_opdr1.pdf**

- Lever alle verslagen in een bundeling samen aan het einde van de practicumterm, dus niet elke week.
- Voeg geen aangeleverde code toe, de instrukteur kent zijn eigen software al.
- Voorzie het verslag op de voorkant in de rechterbovenhoek van:
  - Jullie nickname.
  - Klas.
  - Achternamen.
  - Studentnummers.
  - Het aantal uren dat je aan de opdracht hebt gewerkt.
  - **Deadline: voor aanvang van de volgende les!**

Ir drs E.J Boks, Oktober 2020