

Opdrachten week 7 les 1 – ADC (Analoog Digitaal Converter)

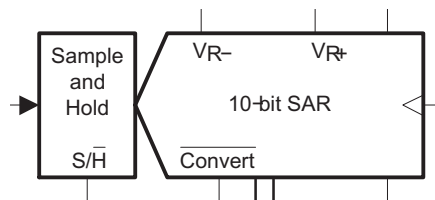
In de opdrachten van week 4 en 5 heb je geleerd een digitaal signaal in te lezen. Deze week ga je leren om een analoog signaal in te lezen in de microcontroller met behulp van de ingebouwde ADC (Analog Digital Converter). Je gaat met behulp van deze kennis de uitgang van een temperatuursensor inlezen om zodoende de omgevingstemperatuur te meten. We maken daarbij gebruik van de LM35DZ temperatuursensor die in je zakje met componenten aanwezig is. De datasheet van de LM35DZ kun je vinden op <http://www.ti.com/lit/ds/symlink/lm35.pdf>.

Tijdens deze les leer je:

- wat een ADC is;
- een analoge spanning in te lezen met de MSP430G2553;
- de temperatuursensor LM35DZ te gebruiken;
- de waarde van een variabele te plotten in Code Composer Studio;
- de temperatuur te laten zien op het oleddisplay.

Een ADC is een schakeling die een analoge spanning omzet naar een digitaal getal (een ‘sample’). Een ADC wordt onder andere gespecificeerd naar hoeveel bits dit sample heeft. Een N -bit ADC deelt een referentiespanning (V_{ref}) op in 2^N stapjes. De mate waarmee de ingangsspanning V_{in} verschilt van V_{ref} bepaalt de sample. Waarbij deze sample een numeriek bereik heeft van 0 tot $2^N - 1$.

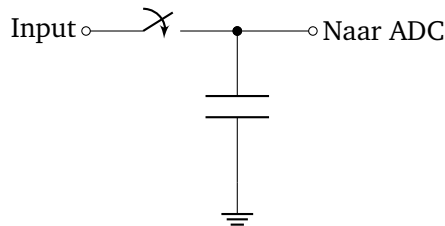
Voorbeeld: Een 12-bit ADC heeft $V_{ref} = 5\text{V}$. Aan de ingang staat een spanning van 3V. Het sample heeft dan een waarde van ongeveer $\frac{3}{5} \cdot (2^{12} - 1) = 2457$. Als er 5V aan de ingang zou staan dan zou de waarde van het sample 4095 zijn.



Figuur 1: Het ADC gedeelte in de μC (uit [figuur 22-1](#) van de User's Guide).

De analoge spanning die de ADC omzet naar een sample, moet wel constant blijven voor een bepaalde tijd. Dit gebeurt door het sample en hold (S&H) circuit, zie [figuur 1](#). De S&H schakeling bestaat simpel gezien uit een condensator en een schakelaar zoals in [scha-](#)

keling 1. Wanneer de schakelaar dichtgaat wordt de condensator opgeladen, wanneer de schakelaar opengaat blijft de condensator geladen. De spanning op de geladen condensator kan vervolgens worden gedigitaliseerd door de ADC.



Schakeling 1: Een simpel S&H circuit.

De opdrachten tot [opdracht 7.1.8](#) zijn uitzoekopdrachten om de juiste instelling voor de ADC te ontdekken. Dit zijn geen programmeeropdrachten. De informatie die wordt verzameld is echter wel nodig om later de registers correct in te stellen. **Maak dus notities!**

7.1.1 Deze week gaan we de ADC van de MSP430G2553 gebruiken. Zoek op in de [MSP-4302x53 datasheet](#)¹ hoeveel bit de ADC van de MSP430G2553 is. Zoek vervolgens in [hoofdstuk 22](#) van de MSP430x2xx Family User's Guide² op wat de omzettingsformule is voor deze ADC.

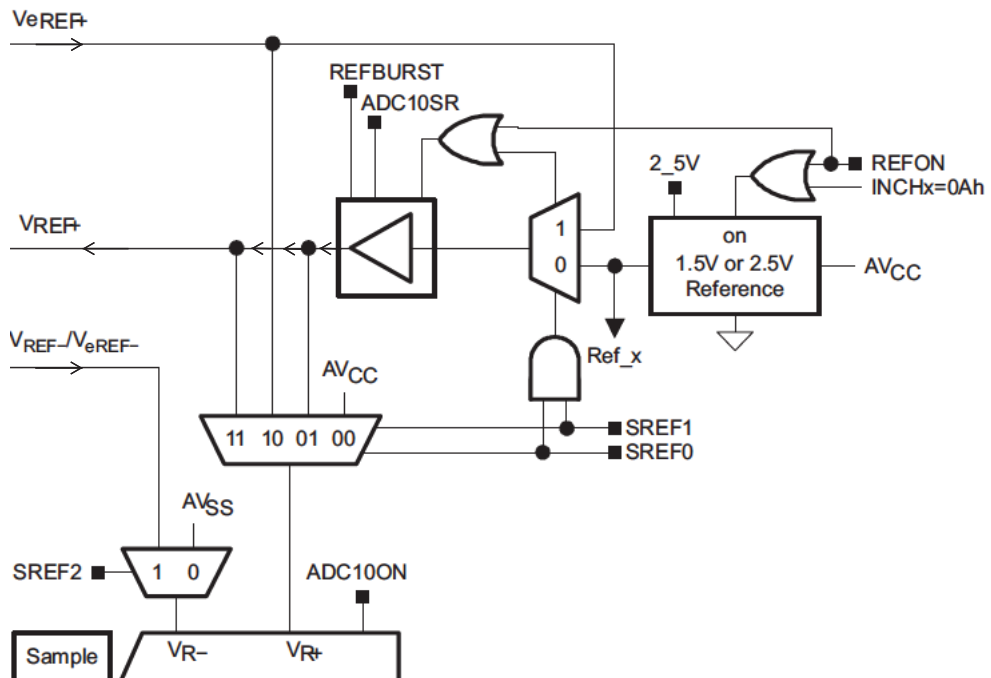
7.1.2 In de gevonden formule staan V_{R-} en V_{R+} . Zie ook [figuur 2](#). De ADC is in staat om externe referentiespanningen te gebruiken (signalen V_{REF+} en V_{REF-}), deze gaan wij niet gebruiken. Intern heeft de ADC een eigen referentiegenerator van 1,5 V of 2,5 V, deze gaan wij wel gebruiken. Deze is rechtsboven in [figuur 2](#) zichtbaar.

Zoek uit welke waarden de SREFx bits krijgen om V_{R-} aan AV_{SS} , en V_{R+} aan Ref_x te koppelen. Let hierbij op dat SREF0 en SREF1 ook de multiplexer erboven beïnvloeden!

Het referentiegeneratorsignaal gaat door een multiplexer en een buffer heen voordat het V_{REF+} wordt genoemd. De buffer en de referentiegenerator worden ingeschakeld m.b.v. de bit REFON. Op de betekenis van de bits REFBURST en ADC10SR komen we later terug.

¹ Deze kun je vinden in de Resource Explorer van CCS of online op: <http://www.ti.com/lit/ds/symlink/msp430g2553.pdf>.

² Deze kun je vinden in de Resource Explorer van CCS of online op: <http://www.ti.com/lit/ug/slau144k/slau144k.pdf>.



Figuur 2: Het bepalen van de referentiespanning voor de ADC.

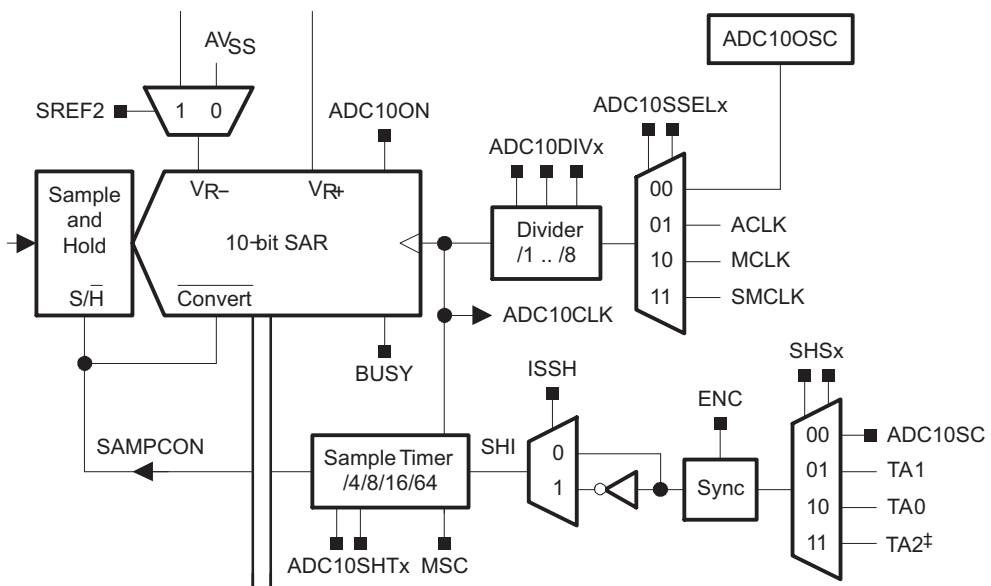
7.1.3 De LM35DZ geeft $10 \text{ mV}/^\circ\text{C}$. Zoek in de [datasheet van de LM35DZ](#) op wat de maximale temperatuur is die deze sensor kan meten. Wat is dus de maximale uitgangsspanning van deze sensor? Als we deze maximale uitgangsspanning willen kunnen meten, welke referentiespanning moeten we dan gebruiken als $V_{\text{REF}+}$ signaal van de ADC? Wat is dan het kleinste spanningsverschil dat je dan nog kunt meten met de (10-bit) ADC? Wat is de nauwkeurigheid van de sensor, uitgedrukt in $^\circ\text{C}$ en in mV? Is de nauwkeurigheid van de ADC voldoende om de sensor met maximale nauwkeurigheid uit te kunnen lezen? Zoek in [paragraaf 22.2.3](#) van de MSP430x2xx Family User's Guide op hoe je de referentiegenerator correct instelt. **Schrijf dit op.**

Naast de hoeveelheid bits wordt een ADC ook gespecificeerd naar hoe snel hij samples kan nemen. Het aantal samples per seconde (de 'sample rate') bepaalt op zijn beurt wat de maximaal waarneembare frequentie is van hetingangssignaal. Als een ADC 100 samples per seconde kan nemen, is het bijvoorbeeld niet mogelijk om een sinus (frequentie) van 1 kHz goed te kunnen meten; er worden simpelweg niet genoeg samples per periode genomen.

De sample rate moet minimaal twee keer zo hoog zijn als de hoogste frequentie die in het ingangssignaal voorkomt om dit signaal in zijn totaliteit te kunnen beschrijven.

De ADC heeft dan twee kloksignalen nodig: een kloksignaal om de conversieschakeling aan te sturen en een kloksignaal om de start van een conversie aan te kunnen geven bij de S&H schakeling. Gelukkig heeft de ADC zelf een interne oscillator van 5 MHz. Deze kunnen wij gebruiken zonder afhankelijk te zijn van de overige systeemklokken, wat het instellen van de ADC versimpelt. De volgende opdrachten gaan hierover.

7.1.4 Om de 5 MHz bron (ADC10OSC) te gebruiken moeten de bits ADC10SSELx ingesteld worden. Welke waarden krijgen deze bits? Door de divider op een deler van 8 te zetten kunnen wij langzaam samplen. We meten temperatuur dus snel samplen is overbodig en later zal blijken dat langzamer samplen ook energie zal besparen. Welke instelling moet ADC10DIVx krijgen?



Figuur 3: Kloksignalen van de ADC

Het opladen van de condensator van het S&H circuit (zie [schakeling 1](#)) kost tijd. Een hogere uitgangsimpedantie R_S van de sensor betekent een langere oplaadtijd. Dit wordt beschreven in [paragraaf 22.2.5.1](#) van de MSP430x2xx Family User's Guide.

- 7.1.5** De LM35DZ heeft een R_S van $0,1 \Omega$, wat voor een wachttijd t_{sample} geeft dit volgens [paragraaf 22.2.5.1](#) van de MSP430x2xx Family User's Guide? Hoeveel tikken van de ADC10CLK zijn dan nodig om tenminste die tijd te wachten? **Schrijf dit op**, dit gaan we later instellen.
- 7.1.6** [Paragraaf 22.2.5](#) van de MSP430x2xx Family User's Guide beschrijft hoe een conversie wordt gestart. In [figuur 3](#) is te zien hoe het SHI-signaal tot stand komt. Welke waarden moeten de bits SHSx hebben om de bit ADC10SC te koppelen aan het signaal SHI? Het signaal hoeft niet geïnverteerd te worden dus de bit ISSH moet nul zijn. **Schrijf dit op**. De bit ADC10SC wordt later gebruikt om een conversie te starten.
- 7.1.7** De ADC van de μC kan maar één signaal tegelijk samplen. Wij willen het signaal van P1.5 (channel A5) koppelen aan de ADC. Gebruik [figuur 22-1](#) uit de MSP430x2xx Family User's Guide om de juist instelling van de input-multiplexer (links in het figuur) te vinden.
- 7.1.8** Maak een nieuw project aan en noem dit LM35plot.
- 7.1.9** Voeg een functie toe om een ADC-waarde (0..1023) om te zetten naar een temperatuur in tiende graden Celcius (bijv. 37.5 graden is 375 tiende graden). Het return-type van de functie is dus **int**. Gebruik onder andere de formules die je kunt vinden in [paragraaf 22.2.1](#) van de MSP430x2xx Family User's Guide. Let op: deze functie kun je al maken zonder de ADC te gebruiken. Stel: de ADC geeft (bij een $V_{\text{REF}+}$ van 1,5 V) een waarde van 137. Deze waarde komt overeen met een spanning van $137/1023 * 1,5 = 0,201$ V. Wat betekent dat de LM35DZ een temperatuur meet van $20,1^\circ\text{C}$ oftewel 201 tiendegraden. De functie moet de waarde 137 dus omzetten naar de waarde 201.
- 7.1.10** Nu is het tijd om alles te integreren.
- A** Sluit de LM35DZ aan op P1.5 volgens <http://www.ti.com/lit/ds/symlink/lm35.pdf#page=3>³. **Let op: de pinout is bekeken vanaf de onderkant**. Als de LM35DZ heet wordt na aansluiten van de voeding, dan is hij verkeerd aangesloten.

³ Voor het oplettende oog: De LM35DZ werkt ook op een spanning van 3,6V

B [Figuur 22-5](#) in de MSP430x2xx Family User's Guide laat de statemachine zien om de ADC in 'single-channel single-conversion' modus te gebruiken. Deze modus gaan we nu toepassen door de registers correct in te stellen. Doe dit met behulp van de notities van de voorgaande opdrachten en de statemachine uit [figuur 22-5](#).


Stel eerst register ADC10CTL1 in⁴. Loop door [paragraaf 22.3.2](#) van de MSP430x2xx Family User's Guide om een waarde te selecteren voor elke beschreven bit. De bit ADC10DF mag hierbij 0 blijven zodat een unsigned getal wordt gesampled.

Stel vervolgens register ADC10CTL0 op dezelfde manier in. ADC10SR kan op 1 om energie te besparen (welke sample-rate hebben we nu?). REFBURST kan op 1 om energie te besparen. Zo staat de buffer alleen aan wanneer nodig. MSC moet in single-conversion modus op 0 staan. ENC moet op 1 om de ADC aan te zetten.

Zet hierbij ook de bit ADC10IE (interrupt enable) aan, maar laat ADC10SC uit. Dit alles doe je vóór de `while(1)`-loop in de `main()`-functie.

C Start een ADC-conversie binnen de `while(1)`-loop van de `main()`-functie door de bit ADC10SC hoog te maken. Gebruik `__delay_cycles` om na het starten van de conversie een halve seconde te wachten.

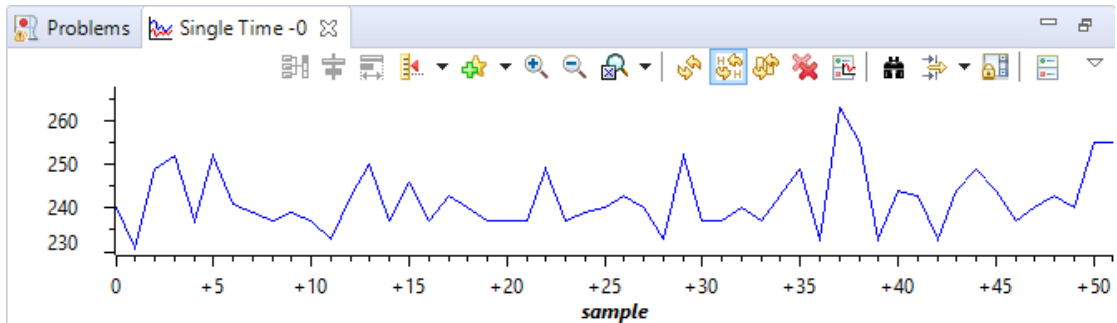
D Zoals beschreven in [figuur 22-5](#) wordt de ADC10IFG hoog gemaakt wanneer een conversie klaar is. Deze vlag roept een ISR aan met de ADC10_VECTOR, maar deze ISR moet nog wel worden geschreven. Maak de ADC10 interrupt service routine en zorg ervoor dat hierbinnen de functie van [opdracht 7.1.9](#) wordt aangeroepen om de waarde uit het ADC10MEM register om te zetten naar een temperatuur. De temperatuur kan worden opgeslagen in een nieuwe `static int` variabele. De `static` variabele wordt in het RAM bewaard zodat CCS dit later correct kan lezen om te plotten.

7.1.11 Als het klopt, doet de code nu het volgende: de ADC wordt correct ingesteld, een conversie wordt elke halve seconde gestart en de temperatuur wordt uitgerekend wanneer de conversie klaar is. Zet een breakpoint in de ADC10_VECTOR ISR en druk op . Een paar dingen kunnen gebeuren:

- De interrupt wordt nooit aangeroepen. Oorzaak kan zijn dat er nooit een conversie wordt gestart (in de `while(1)`-loop in de functie `main`) of dat het interrupt-systeem verkeerd is ingesteld. Los het probleem op.

⁴ Let op de benaming in code; Gebruik underscores. Bijv. `INCH_2` in plaats van `INCH2`.

- Het breakpoint werkt, maar de temperatuur klopt niet. Oorzaak kan zijn dat een andere ‘channel’, of een verkeerde referentiespanning, is geselecteerd. Bekijk de ADC10CTLx-registers in de ‘register debug view’ en los dit op.
- Het breakpoint werkt en de temperatuur klopt. Mooi! Nu kun je naar de volgende opdracht om de temperatuur te laten plotten.

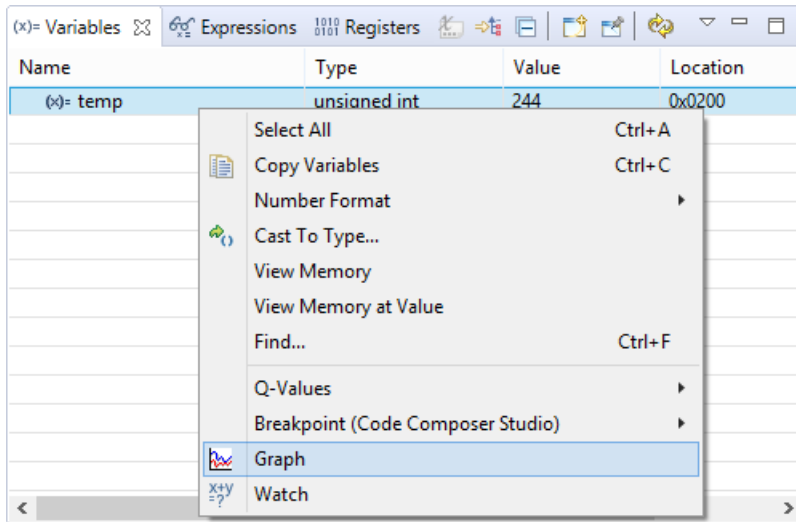


Figuur 4: Plotten van de temperatuur in CCS.

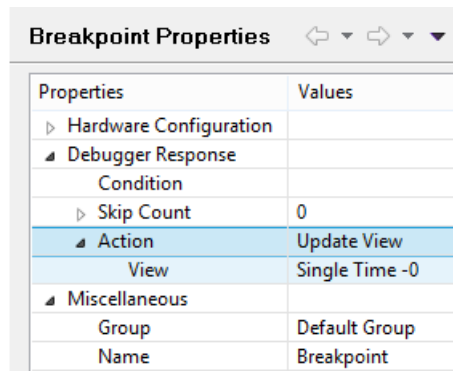
7.1.12 Code Composer Studio is in staat om de waarde van een variabele te plotten zoals weergegeven in [figuur 4](#). Om dit voor elkaar te krijgen moet je een aantal stappen nemen:

- Druk op om te stoppen bij het breakpoint in de ADC10_VECTOR ISR. In de ‘Variables’ window staat de variabele die je hebt gebruikt voor de temperatuur. Klik met de rechtermuisknop op de variabele in dit window en kies voor . Zie ook [figuur 5](#).
- De plot zal nu alleen updaten wanneer het breakpoint opnieuw wordt bereikt. Gelukkig kan dit geautomatiseerd worden. Druk met de rechtermuisknop op het breakpoint in de ISR en kies . Gebruik de instelling van [figuur 6](#) bij het venster wat opent. Deze instelling laat de plot elke keer updaten wanneer het breakpoint wordt bereikt zonder de uitvoer te stoppen.
- Druk nu op en de plot zou automatisch moeten updaten.

Waarschijnlijk ziet jouw plot eruit als [figuur 4](#). De waarden schommelen, dit wordt in de volgende les opgelost door de data te filteren. Waarom maakt de plot stappen en loopt hij niet continu door?



Figuur 5: Graph optie in Variables window.



Figuur 6: Breakpoint eigenschappen.

Verdieping


Het oledisplay wordt op de toets niet gebruikt, maar het is wel erg leuk om er mee te werken! Als voorbereiding op de volgende les is het nu tijd om het oledisplay te testen. In [les 7.3](#) mag je er uitgebreid mee spelen, maar deze les wordt slechts de temperatuur ermee weergegeven.

7.1.13 Fork het project via https://bitbucket.org/HR_ELEKTRO/ems10_adc_oled/fork op dezelfde manier als in [les 5.2](#). Clone de repository en importeer hieruit het project

naar de CCS workspace. Sluit het oleddisplay aan met SCL op P1.6 en SDA op P1.7. Vergeet ook de voeding niet. Run het project en de display zou een temperatuur moeten weergeven.

7.1.14 Kopieer de main- en ADC-code uit project LM35Plot naar het huidige project. In plaats van de temperatuur te plotten laten we die nu op het display zien. Roep, binnen de ADC-ISR, de `setTemp()` functie aan met jouw temperatuurvariabele. Druk op  en elke halve seconde wordt de temperatuur op het display weergegeven.

7.1.15 De `while(1)`-loop wacht nu actief met `__delay_cycles` totdat er een halve seconde is verlopen. Het starten van de conversie kan beter in een timer interrupt worden gedaan waardoor de processor tussendoor kan gaan slapen. Stel `Timer_A1` in zodat deze elke halve seconde een interrupt genereert. Start de ADC-conversie in deze timerinterrupt. Laat de CPU zo diep mogelijk slapen. Welk intern kloksignaal kun je het beste gebruiken als kloksignaal voor de timer?

7.1.16 In de volgende les gaan we verder met dit project, dus commit en push de nieuwe code. Om dit te doen kun je met de rechtermuisknop klikken op de projectnaam in de 'Project Explorer'. In het muismenu kun je dan vervolgens kiezen voor .

In [les 7.3](#) wordt het display nader bekeken!