



Faculteit Ingenieurswetenschappen
Vakgroep Elektronica en informatiesystemen
Voorzitter: Prof. Dr. Ir. J. Van Campenhout

Automatiseren van een meetopstelling voor het opmeten van het vermogenverbruik in beeldschermen

door

Jeroen Hoet

Promotor: Prof. Dr. Ir. J. Doutrelaigne

Scriptiebegeleider: Ir. A. Monté

Scriptie ingediend tot het behalen van de academische graad van
burgerlijk electrotechnisch ingenieur

Academiejaar 2005–2006



Faculteit Ingenieurswetenschappen
Vakgroep Elektronica en informatiesystemen
Voorzitter: Prof. Dr. Ir. J. Van Campenhout

Automatiseren van een meetopstelling voor het opmeten van het vermogenverbruik in beeldschermen

door

Jeroen Hoet

Promotor: Prof. Dr. Ir. J. Doutrelaigne

Scriptiebegeleider: Ir. A. Monté

Scriptie ingediend tot het behalen van de academische graad van
burgerlijk electrotechnisch ingenieur

Academiejaar 2005–2006

Voorwoord

Het gekozen thesisonderwerp sluit nauw aan bij de gevolgde ingenieursvakken tijdens de opleiding. Tijdens de lessen werd soms een link gelegd naar het onderzoek verricht in de vakgroep TFCG, waardoor de interesse voor een thesis bij deze vakgroep sterk is gegroeid. Voor deze thesis moesten enkele problemen onderzocht worden, waarna een oplossing praktisch uitgewerkt moest worden. De combinatie van onderzoek en praktische realisatie is een interessante uitdaging.

Tijdens de praktische realisatie van de thesis is ondervonden met hoeveel problemen men geconfronteerd wordt. Dit zijn meestal onvoorziene situaties, wat aantoont dat op ieder ogenblik met alles rekening moet gehouden worden. Probleemoplossend denken is daarom zeer belangrijk om tot een goed eindresultaat te komen.

Een woord van dank gaat uit naar Prof. J. Doutrelaigne voor het beschikbaar stellen van het interessante thesisonderwerp en de uitstekende begeleiding. Verder wil ik ir. Ann Monté bedanken voor de begeleiding en de hulp tijdens de realisatie van het meetcircuit. Voornamelijk tijdens de finale metingen werd veel samengewerkt, waarbij de goede samenwerking heeft geleid tot een werkend geheel.

Toelating tot bruikleen

“De auteur geeft de toelating deze scriptie voor consultatie beschikbaar te stellen en delen van de scriptie te kopiëren voor persoonlijk gebruik.

Elk ander gebruik valt onder de beperkingen van het auteursrecht, in het bijzonder met betrekking tot de verplichting de bron uitdrukkelijk te vermelden bij het aanhalen van resultaten uit deze scriptie.”

Automatiseren van een meetopstelling voor het opmeten van het vermogenverbruik in beeldschermen

door

Jeroen HOET

Scriptie ingediend tot het behalen van de academische graad van
burgerlijk electrotechnisch ingenieur

Academiejaar 2005–2006

Promotor: Prof. Dr. Ir. J. Doutrelaigne

Scriptiebegeleider: Ir. A. Monté

Faculteit Toegepaste Wetenschappen

Universiteit Gent

Vakgroep Elektronica en informatiesystemen

Voorzitter: Prof. Dr. Ir. J. Van Campenhout

Samenvatting

In het kader van een doctoraat wordt onderzoek gedaan naar het aanpassen van de aanstuursignalen, gebruikt om een beeld weg te schrijven in een bistabiel display. Verschillende optimalisatietechnieken worden toegepast om het vermogenverbruik te reduceren. Om vlot een vergelijking te maken tussen verschillende aanstuurschema's, is er nood aan een meetopstelling die volledig automatisch het vermogenverbruik opmeet.

In deze thesis wordt een meetopstelling gerealiseerd die met grote nauwkeurigheid het vermogenverbruik op de voedingslijnen opmeet. Een computerprogramma zorgt nadien dat deze meetwaarden in kaart worden gebracht en berekent het totale vermogenverbruik.

Trefwoorden

Stroommeting, precisie, analoog/digitaal-conversie, assembler, RS-232 seriële communicatie

Automatic Measurement Setup for Powermeasurement in Bistable Displays

Jeroen Hoet

Supervisor(s): Jan Doutreloigne, Ann Monté

Abstract—In this article, an automatic measurement setup is presented to measure powerconsumption in bistable displays. Because of the complex driving schemes and the use of high voltages, a solution to measure small currents in this high voltage environment is explained. This setup will then be used to easily evaluate different driving schemes for bistable displays.

Keywords—High voltage, power, precision, current measurement

I. INTRODUCTION

BISTABLE displays are displays that don't need a voltage source to keep an image. Only when writing a new image to the display, a voltage source is needed. This makes the displays especially suited for applications where the image refresh rate is small and where (battery) power is limited.

To write an image to a bistable display, complex waveforms with voltages up to 85V are needed. These waveforms will be generated by a driver chip that, for example, only has a battery to power the circuit. Special high voltage driver circuits have already been designed to minimize the power consumption[1].

When some programmable logic is added to these driver circuits that calculate the most efficient waveforms for a given image, power consumption can be further reduced. The two principles used to obtain an efficient waveform are the use of intermediate voltage levels and the adaptation of the waveform to the image[2].

To quickly evaluate these different waveforms and optimisations, an automated measurement setup is realised to easily measure the powerconsumption with great detail.

II. MEASURING POWER

The display driver makes use of 6 power lines to generate the correct waveform. Each line has a specific voltage: 5,35,45,55,60 and 85V. To measure power, a current has to be measured on these lines. In order to do this, a small shunt-resistor is added in each of these lines. When current flows through one of these lines, through the resistor, a small voltage drop will occur over the resistor. Using $V = R * I$, the current is easily extracted when this voltage drop is measured.

A. Measure Current

A big problem when measuring the small voltage drop over the resistor is that a high voltage V_{cm} is also present. To extract the usefull signal out of this high voltage environment, a differential amplifier (AD629) is used[3]. Because the differential amplifier only extracts the usefull signal without amplification, an extra amplifier is needed.

After amplification, the signal will be presented to a 10-bit analog-/digitalconvector (ADC), with a V_{ref} of 5V. The amplification will try to convert the maximum current I_{max} into V_{ref} .

This way the full scale of the ADC will be used. When using 10 bit and $V_{ref}=5V$, the smallest level will be:

$$\frac{V_{ref}}{2^{10} - 1} = 4.888mV \quad (1)$$

When using for example an amplification of 15 and a resistor of 56Ω , the smallest measurable current will be:

$$\frac{4.888mV}{R * A} = 6\mu A \quad (2)$$

In table I, for each line the shunt-resistor is shown together with I_{max} and the amplification. Also the smallest measurable current I_{min} is shown.

TABLE I
SETUP OF THE ANALOG MEASUREMENT CIRCUIT

Channel	$I_{max}(mA)$	R (Ω)	A	V_{max} (V)	I_{min} (μA)
85V	5.85	56	15	4.914	5.82
60V	5.47	56	16	4.901	5.46
55V	5.37	56	16	4.812	5.46
45V	5.17	56	17	4.922	5.16
35V	4.93	56	18	4.969	4.85
5V	0.93	220	24	4.913	0.93

The ADC has a multiplexer integrated on chip to choose between the 6 power-lines. The 10-bit presentation of the measured voltage will be send to a microprocessor wich will store these values into external RAM. When there is no more space to store values, measurement will stop and the complete content of the RAM will be send to the computer.

On the computer, an application will convert all these values into correct measurements and display them on a graph. As a final result, the application will calculate the area under the graph as the total powerconsumption in each line.

III. SOFTWARE

The microprocessor needs software to communicate with the ADC, the PC and to store values in RAM. The software is written in assembler.

Software to display the measurements on the computer and to calculate the powerconsumption, is written in c#. This is a language that is derived from c/c++ and other languages. The great advantage of c# is that it is easy to use complicated graphic components.

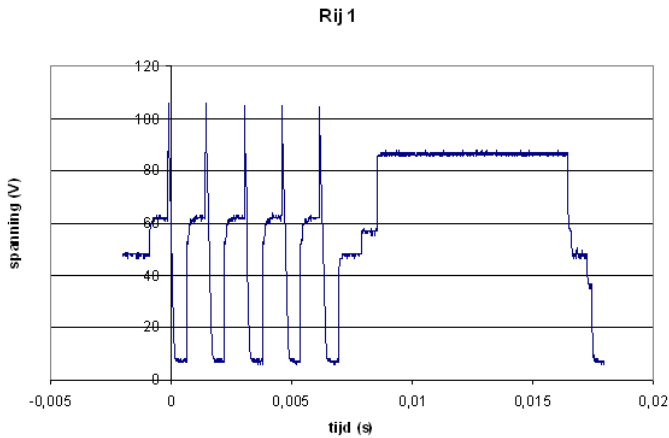


Fig. 1. Measurement of reset-pulses and image-info on a row

A. Microprocessor

There are two software versions available to control the measurement setup. The first one will measure all 6 channels until memory is full. Using 128kB external RAM, 5461 values can be measured on each channel. The second version will measure only one channel until memory is full. This makes it possible to measure 6 times longer on a selected channel. Both versions are integrated into one program.

On a reset, the microprocessor will initialise and startup the ADC. Then it waits for a startsignal coming from the PC. This startsignal will also determine which version of the program should run. If the PC sends 0xFF, all 6 channels will be measured. Otherwise the PC will send 0x2X, where the X represents the channel number. In this case the second version of the program will run.

The microprocessor will setup the ADC to measure the first channel or the channel selected by the computer. To ensure that only valid data will be measured, the circuits that write an image to the display will give a startsignal when an image is written. At this point the microcontroller will start measuring. Every $5,75\mu\text{s}$, a measurement is done.

When 6 channels are selected, the program will again setup the ADC to the next channel after measuring 5461 values and wait for an image to be written. When all 6 channels have been setup, measurement is completed. When only 1 channel is selected, measurement will stop when memory is full.

As a final step, the ADC will tell the computer that it's ready to send and will then send all data stored in RAM.

B. Pc

First the program on the pc will convert all values into the measured voltage at the ADC and correct it with an offset. This is done by:

$$V_{ADC} = \text{value} * \frac{V_{ref}}{1023} - V_{offset} \quad (3)$$

Using V_{ADC} , the power can be calculated using the channel setup as seen in table I:

$$P = V_{channel} * \frac{V_{ADC}}{A * R} \quad (4)$$

It is important to calibrate the program, otherwise when an error exists on for example the amplification or an offset-factor, a systematic error will be introduced. It is possible to change different settings such as: shunt-resistor, amplification, V_{ref} and the offset.

The power-values will then be plot on a graph and the total powerconsumption is calculated. If nessesary, these values can also be stored in an excel-file to reload it or make more calculations afterwards. Because of the high resolution of the measurements, it is possible to zoom in on the graph.

IV. TESTING

As a test, an image is written to the display and total power is measured. In figure 1, the waveform presented to a row of the display, is shown. Notice the five 60V reset pulses at the beginning and there timing.

Figure 2 shows the measured power on the 60V channel. There are 10 peaks, for each resetpuls two, at exactly the same timing of the resetpulses. After $10\mu\text{s}$, there seems to be noise but when zooming in, it shows that these are also pulses corresponding to the waveforms writing the image.

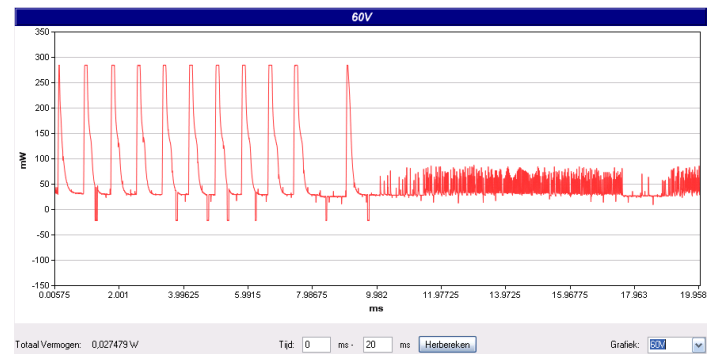


Fig. 2. Powermeasurement on channel 60V

V. CONCLUSION

As seen in figures 1 and 2, the measurement setup works fine. Every $5,57\mu\text{s}$ there is a measurement with great detail. This great detail is achieved after calibrating the PC-program to reduce systematic errors.

REFERENCES

- [1] J. Doutrelaigne, H. De Smet, and A. Van Calster, "A New Architecture for Monolithic Low-Power High-Voltage Display Drivers", Proceedings of the 20th International Display Research Conference IDRC2000, (Palm Beach, Florida, USA), pp. 115 - 118, September 2000.
- [2] A. Montè, J. Doutrelaigne, and A. Van Calster, "An Intelligent Driving Scheme for High-Voltage Display Drivers", Proceedings of the 11th International Display Workshops IDW '04, (Toki Messe, Niigata, Japan), pp. 1737-1740, (December 2004).
- [3] Charles Kitchin, Lew Counts, "A designer's guide to instrumentation amplifiers 2nd edition", <http://www.analog.com>.

Inhoudsopgave

1	Inleiding	1
2	Stroommeting	3
2.1	Probleemstelling	3
2.2	Meetcircuit	5
2.2.1	Oplossing 1: Opamp	5
2.2.2	Oplossing 2: Instrumentatieversterker	6
2.2.3	Oplossing 3: Differentiële versterker	7
2.3	Verwerking van de meetwaarden	8
3	Hardware: AnalooG	10
3.1	Component keuze	10
3.1.1	Differentiële versterker	10
3.1.2	Extra versterking: Opamp	11
3.1.3	AnalooG naar digitaal convertor	14
3.2	Schema	16
3.3	Metingen	17
3.3.1	Versterking	18
3.3.2	Offset	19
3.3.3	Ruis	19
3.3.4	Bescherming ADC	21
4	Hardware: Digitaal	23
4.1	Componenten	23
4.1.1	AnalooG Digitaal Convertor	23
4.1.2	Microcontroller	26

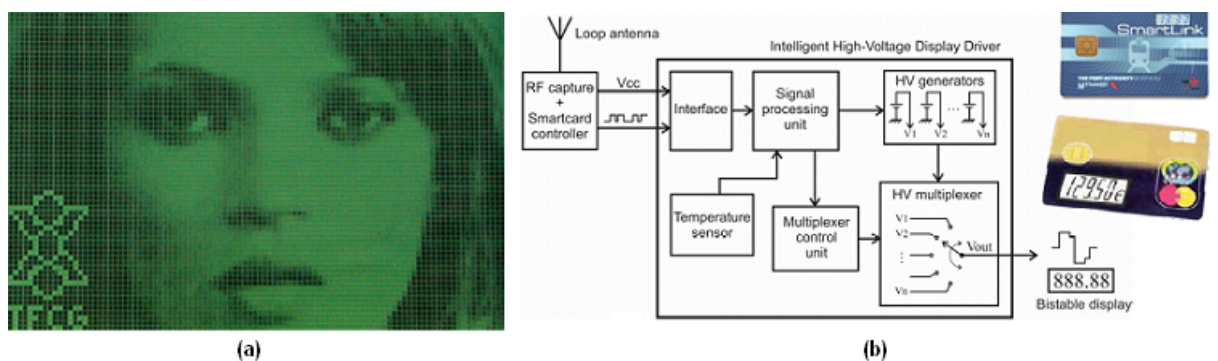
4.1.3	Ram Geheugen	27
4.1.4	RS-232 seriële communicatie	30
4.1.5	Randcomponenten	30
4.2	Schema	32
5	Software	35
5.1	Microcontroller	35
5.1.1	Seriële link	36
5.1.2	Ram-geheugen	39
5.1.3	ADC-controle	41
5.1.4	Volledige programma	46
5.2	Pc-programma	49
5.2.1	Seriële communicatie	49
5.2.2	Grafiek	51
5.2.3	Instellingen	52
6	Eindresultaat	56
6.1	Testen	56
6.2	Verbeteringen	57
6.2.1	Sneller meten	57
6.2.2	Negatieve stromen	58
7	Besluit	62
A	Bestelling	63
B	Foto meetcircuit	64
C	Programma-code: microcontroller	65

Hoofdstuk 1

Inleiding

Bij de vakgroep TFCG wordt onderzoek gedaan naar bistabiele beeldschermen. Dit zijn beeldschermen die zonder voedingsspanning hun beeld kunnen behouden. Enkel tijdens het inschrijven van een beeld wordt vermogen gedissipeerd. Deze beeldschermen zijn uitermate geschikt voor toepassingen met een lage beeldfrequentie ($<50\text{Hz}$), waarbij vermogenverbruik laag moet worden gehouden.

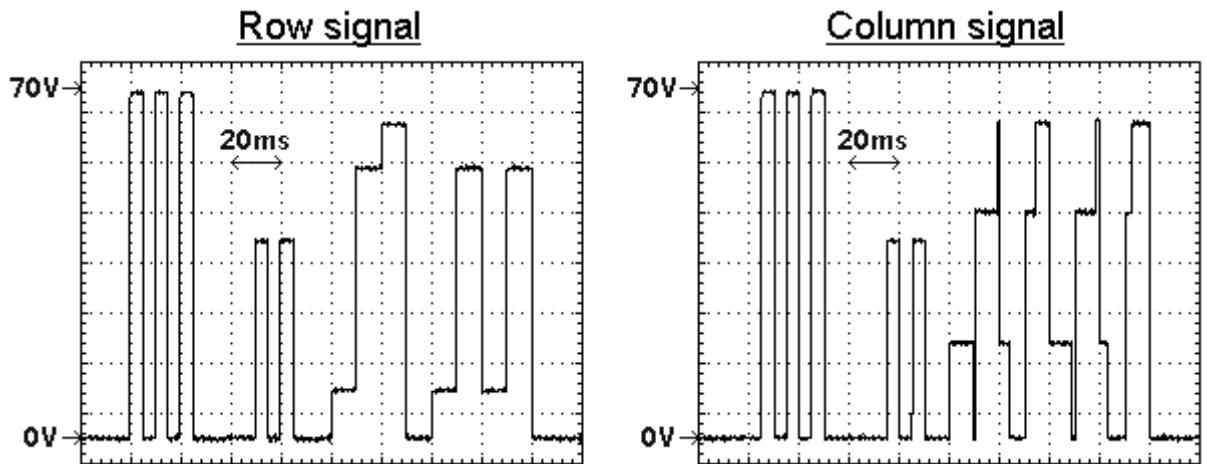
In figuur 1.1(a) is een voorbeeld weergegeven van een bistabiel beeldscherm. In figuur 1.1(b) wordt een bistabiel beeldscherm gebruikt in combinatie met een protonkaart. Het idee bestaat om telkens wanneer de kaart wordt gebruikt in een transactie, het saldo aan te passen en weer te geven op het display. Tijdens de transactie maakt de kaart contact met een terminal, waar een spanningsbron beschikbaar is om het beeld aan te passen. Zodra de kaart losgekoppeld wordt van de terminal, valt de spanning weg. Het beeld blijft echter wel behouden.



Figuur 1.1: Voorbeelden van bistabiele beeldschermen en hun toepassingen

Tijdens het wegschrijven van een beeld worden hoge spanningen tot 85V gebruikt om een

pixel van waarde te doen veranderen. Een voorbeeld van de complexe golfvormen die gebruikt worden om een beeld weg te schrijven, is weergegeven in figuur 1.2. Deze complexe golfvormen moeten gegenereerd worden vanuit een laagspanningsbron, zoals een batterij. Het is belangrijk om ook nu ervoor te zorgen dat het vermogenverbruik zo laag mogelijk blijft.



Figuur 1.2: Complexe golfvormen om een beeld weg te schrijven

Op hardware niveau is al voldoende onderzoek geleverd, waarbij de momenteel beschikbare aanstuurcircuits volledig geoptimaliseerd zijn naar vermogenverbruik[1]. Maar ook door een goede softwarematige aanpassing van de aanstuurgolven zelf, kan het vermogenverbruik gereduceerd worden.

In het kader van een doctoraat worden verschillende optimalisaties van de golfvormen bestudeerd. Er worden 3 verschillende optimalisatietechnieken gebruikt: gebruik van tussenniveaus (enkele μs), charge recycling en gebruik van beeldinformatie[2]. Voor meer informatie over de gebruikte technieken wordt verwezen naar de literatuur.

Deze thesis heeft als doel een geautomatiseerde meetopstelling te maken, zodat verschillende aansturingen en verschillende optimalisaties snel kunnen geëvalueerd worden naar hun vermogenverbruik.

Hoofdstuk 2

Stroommeting

De aanstuurchip die ervoor zal zorgen dat het correcte beeld wordt weggeschreven in het display, maakt gebruik van 6 voedingslijnen om de verschillende aanstuursignalen op te bouwen. Concreet zijn volgende spanningen beschikbaar: 85V, 60V, 55V, 45V, 35V en 5V. In elk van deze voedingslijnen wil men het vermogenverbruik kennen. Om het vermogenverbruik te bepalen is er nood aan 2 grootheden: stroom en spanning ($P = I * V$). De spanning is gekend, waardoor enkel nog de stroom in deze voedingslijnen moet gemeten worden.

Het meten van de stroom wordt gedaan met behulp van een kleine shunt-weerstand in iedere voedingslijn. Als er stroom door een voedingslijn vloeit, zal deze weerstand een (kleine) spanningsval veroorzaken. Hierdoor wordt het originele circuit beïnvloed, wat niet gunstig is. Deze shunt-weerstand wordt zo klein mogelijk gehouden, zodat deze weerstand het originele circuit weinig beïnvloed. Vanaf nu wordt naar een voedingslijn verwezen als er gesproken wordt over 'een kanaal'.

2.1 Probleemstelling

In eerste instantie wil men dat er een zo laag mogelijke spanningsval is over de meetweerstand bij de maximale stroomniveaus. Zoals reeds vermeld wordt de meetweerstand hiervoor zo klein mogelijk gekozen. Aan de andere kant moeten stromen tot $10\mu A$, indien mogelijk nog kleiner, meetbaar blijven. In combinatie met de kleine meetweerstand krijgt men kleine spanningsvallen. Het circuit dat de uiteindelijke meting zal verrichten moet een grote precisie vertonen.

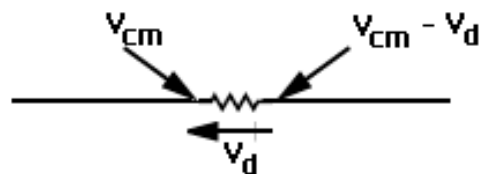
In tabel 2.1 zijn voor de verschillende voedingslijnen de maximale stroomniveau's weergegeven. Deze werden opgemeten in de ontwerpomgeving van Cadence. De gekozen meetweerstand zijn

Spanning(V)	I_{max} (mA)	R(Ω)	V_{max} (mV)	V_{min} (μ V)
85	5,85	56	327,6	311,974
60	5,47	56	306,32	305,474
55	5,37	56	300,72	305,474
45	5,17	56	289,52	287,505
35	4,93	56	276,08	245,145
5	0,93	220	204,71	192,614

Tabel 2.1: Stroom en spannings-niveau's

eveneens in de tabel weergegeven, samen met de maximale en minimale spanningsvallen over de weerstanden. De minimale spanningsval is eerst berekend voor een stroom van $10\mu\text{A}$, de gewenste precisie. Later, in het hoofdstuk over analoge hardware waar de analog-/digitaalconverter wordt besproken, zal blijken dat een precisie van $\approx 5\mu\text{A}$ haalbaar is. Om toch al een idee te vormen van de grootteorde van spanningen wordt in de tabel uitgegaan van een minimum van $5\mu\text{A}$. Later wordt dit minimum vastgelegd per kanaal.

Een volgende probleem stelt zich doordat er gewerkt wordt op hoogspanning. Hoogspanning in deze context wil zeggen: alles boven de voedingsspanning (5V). Zoals reeds vermeld, moeten zeer preciese metingen gedaan worden. De hoogspanningsomgeving waarin gewerkt wordt, vormt hiervoor nog een extra moeilijkheid. De gemeten spanning zal aan beide kanten van de meetweerstand hoog zijn, terwijl de verschilspanning (V_d) klein is. Dit is verduidelijkt in figuur 2.1. Het is duidelijk dat het nuttige signaal net deze zeer kleine afwijking is. Dit signaal moet gefilterd worden uit de hoogspanningsomgeving. In wat volgt wordt een oplossing voor dit probleem gezocht.

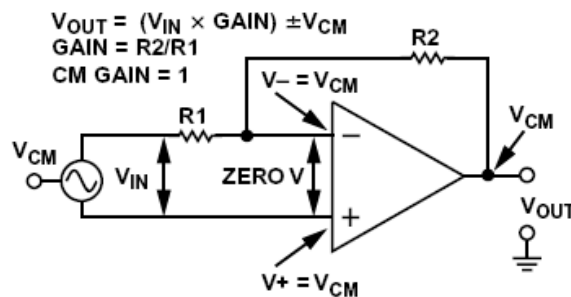
Figuur 2.1: Spanningsval: Nuttige spanning V_d tov. V_{cm}

2.2 Meetcircuit

De gemeten spanningsvallen zijn zo klein dat het noodzakelijk is om deze signalen te versterken. Dit is niet haalbaar met een passief circuit en de keuze werd snel gemaakt om actieve componenten te gebruiken.

2.2.1 Oplossing 1: Opamp

Als eerste oplossing wordt gedacht een normale opamp te gebruiken met de nodige feedback. In figuur 2.2 is een opamp gebruikt als versterker die ingesteld kan worden met behulp van de weerstanden R_1 en R_2 . Hetingangssignaal bestaat uit eeningangsspanning V_{in} , gesuperponeerd op een gemeenschappelijke 'common-mode' spanning V_{cm} . V_{cm} stelt in ons probleem de hoge spanning van een voedingslijn voor en V_{in} de variabele spanningsval, het nuttige signaal. Om V_{uit} te berekenen wordt gebruik gemaakt van superpositie:



Figuur 2.2: Opamp: V_{uit} ifv V_{in} en V_{cm}

- a) De invloed van V_{in} geeft ons de transfertfunctie van een gewone inverterende versterkerschakeling:

$$\frac{V_{in}}{R_1} = \frac{V_{uit}}{R_2}$$

$$V_{uit} = -V_{in} \frac{R_2}{R_1}$$

- b) De invloed van V_{cm} is bij deze schakeling belangrijk. Omwille van nullatorwerking staan op beide ingangen de spanning V_{cm} . Er vloeit geen stroom door R_1 en dus ook niet door R_2 aangezien er geen stroom in de opamp vloeit (verwaarloosbaar). Hierdoor wordt $V_{uit} = V_{cm}$.

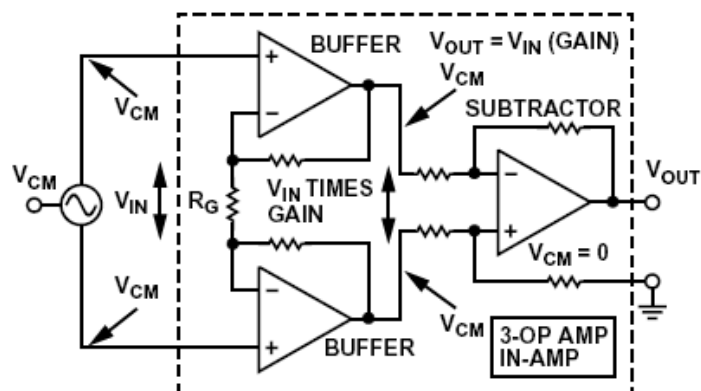
Als beide delen worden opgeteld, wordt $V_{uit} = -V_{in} * \frac{R_2}{R_1} + V_{cm}$. V_{in} wordt versterkt met de ingestelde versterking $-\frac{R_2}{R_1}$, maar ook V_{cm} wordt doorgelaten met versterking 1.

Deze oplossing is niet bruikbaar om ons probleem op te lossen omdat V_{cm} wordt doorgelaten. Als extra probleem moeten de opamps gevoed worden met een voedingsspanning die tot 85V gaat omdat V_{uit} niet hoger kan worden dan de voedingsspanning. Daarbij komt dat de opamp symmetrisch moet gevoed worden. Dit betekent dat er 2 extra hoogspanningen aanwezig moeten zijn per opamp.

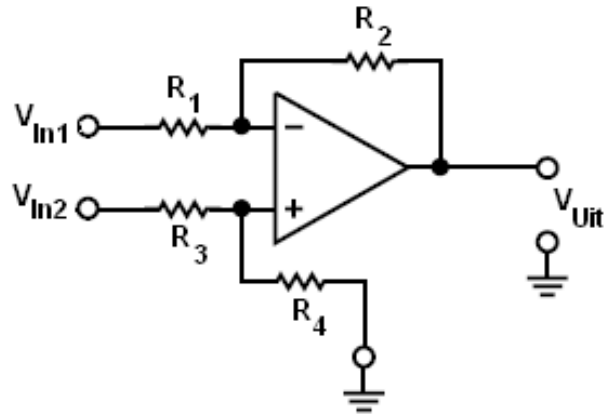
2.2.2 Oplossing 2: Instrumentatieversterker

Het belangrijke gebrek bij de vorige oplossing is het doorlaten van de common-mode spanning. Een gewone opamp biedt niet de mogelijkheid om deze spanning weg te filteren aan zijn uitgang. Voor deze toepassing bestaat er een andere soort actieve componenten, de instrumentatieversterkers (inamps). Deze inamps proberen zo goed mogelijk de common-mode spanning te verzwakken en de verschilspanning door te laten, eventueel zelfs te versterken[3].

In figuur 2.3 is een basis inamp circuit zichtbaar, opgebouwd uit 3 opamps. De eerste 2 opamps zijn geschakeld als volgerschakelingen. Dit wil zeggen dat ze het ingangssignaal gewoon doorgeven naar de volgende knoop. Indien nodig kan nog een weerstand R_G worden toegevoegd om het ingangssignaal te versterken. Belangrijk is dat deze opamps dienst doen als buffers. Deze zorgen voor een zeer hoge ingangsimpedantie van de inamp, waardoor de inamp zelf zo goed als geen invloed meer heeft op het te meten circuit. De laatste opamp doet dienst als verschilversterker.



Figuur 2.3: Instrumentatieversterker



Figuur 2.4: Verschilversterker: Berekening Vuit

Een verschilversterker wordt opgebouwd als in figuur 2.4. Veronderstel eerst dat alle weerstanden willekeurig zijn. Door superpositie krijgt men:

$$V_{uit} = \begin{cases} -V_{in1} \frac{R_2}{R_1} & \text{Bijdrage } V_{in1} \\ V_{in2} \frac{R_4}{R_3+R_4} \frac{R_1+R_2}{R_1} & \text{Bijdrage } V_{in2}. \end{cases} \quad (2.1)$$

Wanneer $\frac{R_4}{R_3} = \frac{R_2}{R_1}$, krijgt men:

$$V_{uit} = \frac{R_2}{R_1} (V_{in2} - V_{in1}) \quad (2.2)$$

Aan de uitgang is geen V_{cm} meer zichtbaar. Dit lijkt een goede oplossing voor het probleem, maar het probleem van een hoge voedingsspanning blijft bestaan. De twee opamps die dienen als buffers, zijn dezelfde als in oplossing 1. Daar is al aangetoond dat deze opamps moeten gevoed worden met een symmetrische spanning tot 85 volt, want niet evident is. De componenten zijn duur en de spanningen zijn niet zomaar beschikbaar.

Deze implementatie biedt wel een oplossing voor ons grootste probleem, de common-mode spanningen, maar zorgt voor een grote overhead door de extra voedingen.

2.2.3 Oplossing 3: Differentiële versterker

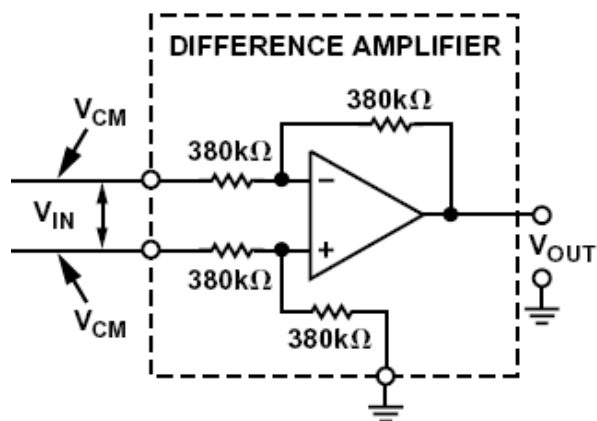
Bij een inamp wordt de common-mode spanning weggefilterd door de differentiële versterker. De buffers dienen enkel om een hoge ingangsimpedantie te krijgen. Deze differentiële versterkers

bestaan ook afzonderlijk, zonder de bufferschakeling[4]. In figuur 2.5 is een schema zichtbaar van een differentiële versterker.

Het is mogelijk om deze versterker na te bouwen met een normale opamp en externe weerstanden. Het is echter belangrijk dat de weerstanden goed gematched zijn om een zo hoog mogelijke CMRR te krijgen. Op een chip kunnen deze weerstanden zeer goed gematched worden, waardoor de CMRR zeer hoog wordt.

De CMRR geeft weer hoe goed de common mode is weggefilterd in het uitgangssignaal: $CMRR = A_d \frac{V_{cm}}{V_{uit}}$, met A_d de differentiële versterking. Wanneer door een fout op de weerstanden niet meer voldaan is aan $\frac{R_4}{R_3} = \frac{R_2}{R_1}$, zal formule 2.2 niet meer gelden en moet formule 2.1 worden gebruikt. Omdat V_{in1} en V_{in2} anders versterkt worden, zal V_{cm} niet wegvallen door beide bijdragen op te tellen.

Zoals te zien is in de figuur, zijn de weerstandswaarden hoog. Dit zorgt voor een goede CMRR en zorgt ervoor dat de component een hoge ingangsimpedantie behoudt. Aangezien de differentiële versterker eigenlijk zelf niet versterkt met deze keuze van de weerstanden (zie formule 2.2), is V_{uit} een zeer klein signaal. Dit signaal moet nog versterkt worden met een gewone opamp om verdere verwerking van het signaal mogelijk te maken.



Figuur 2.5: Differentiële versterker

2.3 Verwerking van de meetwaarden

De spanningsval zal gemeten worden over de meetweerstand en de common-mode spanning wordt weggefilterd door de differentiële versterker. Dit signaal wordt op zijn beurt versterkt door een opamp. Het uitgangssignaal kan dan gebruikt worden om te bemonsteren met een analog

naar digitaal convertor, die de analoge meetwaarde omzet in een bit-voorstelling. Zolang een beeld wordt weggeschreven worden metingen gedaan en deze metingen worden opgeslagen in een geheugen. Zodra de metingen gedaan zijn, wordt het geheugen uitgelezen en worden de metingen doorgezonden naar de computer voor verdere verwerking. Het uiteindelijke doel is het verkrijgen van een grafiek waarin het stroomverbruik is te zien per voedingslijn, samen met het totale vermogenverbruik in elk van de lijnen.

In de volgende hoofdstukken worden de componenten voor het analoge en het digitale gedeelte gekozen en worden de circuits opgesteld.

Hoofdstuk 3

Hardware: AnalooG

Nu er een oplossing is gevonden om de spanningsvallen over de meetweerstand nauwkeurig te meten, moet het geheel geïmplementeerd worden. De componenten worden zorgvuldig gekozen aan de hand van de vooropgestelde specificaties. Omdat op breadboard gewerkt wordt, moeten componenten gekozen worden met een dual-inline verpakking. Alle nodige componenten worden aangekocht bij farnell, een schema wordt opgesteld en het volledige circuit wordt uitgemeten om te testen dat de specificaties weldegelijk worden gehaald. Een volledige lijst met bestelde componenten is weergegeven in appendix A.

3.1 Component keuze

3.1.1 Differentiële versterker

De belangrijkste component in de meting is de differentiële versterker. In de literatuur[4] wordt hiervoor de AD629 van analog devices aangeraden. Dit is een differentiële versterker met zeer hoge common-mode ingangsbereik en een hoge CMRR. Een typische applicatie voor deze versterker is het meten van kleine stromen op hoogspanningslijnen met hoge nauwkeurigheid. De specificaties van deze component zijn de volgende:

- $\pm 270\text{V}$ common-mode spannings-bereik
- Ingangen beschermd tot $\pm 500\text{V}$
- Maximaal $20\mu\text{V}/^\circ\text{C}$ offset drift
- Minimaal 77dB CMRR

- Bandbreedte: 500kHz
- Voedingsspanning: $\pm 2,5V$ tot $\pm 18V$

De maximale ingangs-spanning en de common-mode spanning bedragen ongeveer 85V, wat zeker geen probleem vormt. Een bandbreedte van 500kHz wil zeggen dat pulsen tot $\frac{1}{2\pi f} = 0,2\mu s$ doorgelaten worden. Het is duidelijk dat alle specificaties voldoen aan de voorwaarden.

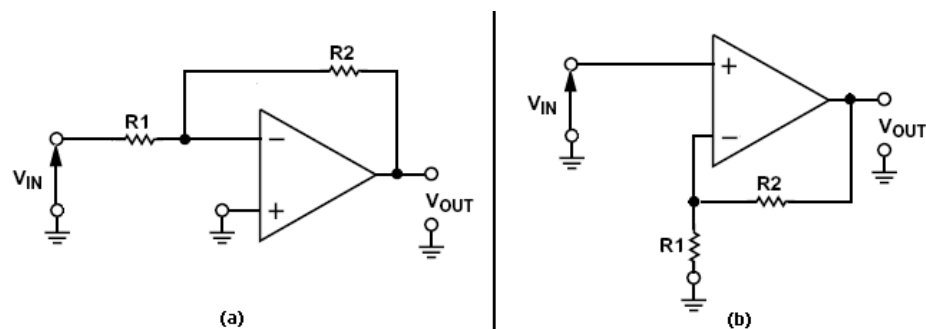
De bestelde component bij farnell is de AD629AN.

3.1.2 Extra versterking: Opamp

Het signaal afkomstig van de differentiële versterker is zeer zwak en moet nog versterkt worden. De versterker heeft best een lage offset en moet snel genoeg zijn. Verdere specificaties zijn er niet. De gekozen opamp is de OP07C van ST-electronics met volgende specificaties:

- Max $150\mu V$ offset, regelbaar met externe weerstand
- Lage drift: $0,5\mu V/^{\circ}C$
- Voedingsspanning: $\pm 3V$ tot $\pm 22V$

Er zijn twee mogelijkheden om een versterking te maken. Inverterend en niet inverterend. Beide circuits zijn weergegeven in respectievelijk figuur 3.1(a) en 3.1(b). Via een eenvoudige berekening, gebruik makende van de nullatorhypothese, krijgt men:

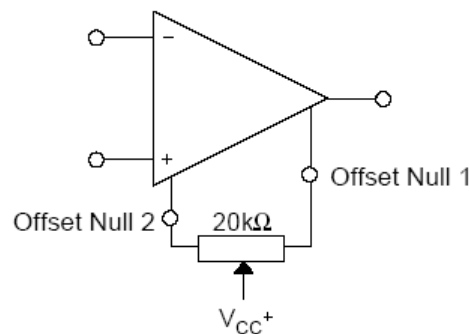


Figuur 3.1: (a) Inverterende versterker (b) Niet-inverterende versterker

$$V_{uit} = \begin{cases} -V_{in} \frac{R_2}{R_1} & \text{Inverterende versterker (a);} \\ V_{in} \frac{R_1 + R_2}{R_1} & \text{Niet inverterende versterker (b).} \end{cases} \quad (3.1)$$

Bij de inverterende versterker wordt de ingang aan de $V(-)$ -klem gehangen. Dit is eveneens de klem die opgenomen is in het feedback pad. Hierdoor wordt de ingangsweerstand bepaald door R_1 , welke niet oneindig groot is. Bij de niet-inverterende versterker is de ingang aan de $V(+)$ -klem gehangen, waardoor de ingangsimpedantie deze van de opamp is. Deze wordt aangenomen als oneindig groot, waardoor er geen stroom zal vloeien in de opamp en $V(-)$ niet zal beïnvloeden.

De offset spanning van de opamp kan met behulp van een externe regelbare weerstand (potentiometer) ingesteld worden zoals op figuur 3.2. Door een correcte instelling van de potentiometers kan de offset aan de uitgang gereduceerd worden tot 0,2mV. Later zal blijken dat er gewerkt wordt met een 10bit analoog naar digitaal-converter met referentiespanning 5V. Het kleinste mogelijke meetinterval wordt hierdoor $\frac{5}{2^{10}-1} = 4,888\text{mV}$. De extra lage offset kan in het slechtste geval enkel zorgen voor 1lsb fout.



Figuur 3.2: Offset instellen

Het is de bedoeling dat de uitgangen van de verschillende opamps gesampled worden door een analoog naar digitaal converter. In de volgende paragraaf wordt hier in meer detail op ingegaan. Op dit moment is het voldoende te weten dat deze omzetting zal gebeuren met een referentiespanning van 5V, de voedingsspanning van het digitale circuit. In het hoofdstuk over de stroommeting is in tabel 2.1 al berekend wat de minimale en maximale spanningsvallen zijn die gemeten worden over de meetweerstand. Met behulp van de opamps kan ervoor gezorgd worden dat deze maxima zo goed mogelijk overeenkomen met de 5V referentie spanning.

In tabel 3.1 is per kanaal weergegeven wat de maximale meetbare spanningsval (V_{max}) is over de meetweerstand. De versterking (A) wordt gekozen zodat $V_{max} * A \approx 5\text{V}$. Alle versterkingen zijn afgerond tot op een geheel getal. Het is niet nuttig te werken met een versterking van vb. 15,2626... omdat deze versterking niet realiseerbaar is met reële weerstanden. De afgeronde

Kanaal (V)	V_{max} (mV)	A	$V_{uit-max}$ (V)	I_{min} (μ A)
85	327,6	15	4,914	5,82
60	306,32	16	4,90112	5,46
55	300,72	16	4,81152	5,46
45	289,52	17	4,92184	5,16
35	276,08	18	4,96944	4,85
5	204,71	24	4,91304	0,93

Tabel 3.1: Ideale versterkingen

versterkingen dienen dan ook eerder als een richtlijn. In de tabel is ook de resulterende uitgangsspanning $V_{uit-max}$ weergegeven. I_{min} stelt de minimaal meetbare stroom per kanaal voor. Dit wordt bepaald door gebruik te maken van het kleinste meetinterval van 4,888mV aan de ADC. De minimale meetbare spanningen V_{min} zullen pas volledig bepaald worden wanneer de analog-/digitaalconverter is gedimensioneerd.

De versterkingsfactoren worden gerealiseerd met het niet-inverterende opamp-circuit uit figuur 3.1(b). Als eerste keuze wordt aangenomen dat $R_1 = 1k\Omega$. R_2 wordt dan bepaald via:

$$A = \frac{R_1 + R_2}{R_1} \quad (3.2)$$

$$R_2 = (A - 1) * R_1 \quad (3.3)$$

Niet bij alle versterkingsfactoren is $R_1 = 1k\Omega$ de beste keuze. Daar waar nodig zijn deze weerstanden aangepast om zo goed mogelijk de benodigde versterking te verkrijgen. In tabel 3.2 zijn de gekozen weerstanden per kanaal weergegeven samen met de uiteindelijke versterking. In de tabel zijn de weerstanden gekozen uit de E-24 reeks, waarbij een + teken staat voor een serieschakeling van 2 weerstanden. Er is gekozen om niet meer dan 2 weerstanden in serie te plaatsen omdat elke weerstand tollerantie-fouten (5%) introduceert en omwille van plaatsgebrek op de compacte breadboard.

Sommige van de spanningsversterkingen zijn groter geworden dan A_{ideaal} waardoor het gevaar bestaat dat de maximale spanningsval V_{max} versterkt wordt tot een waarde boven de referentiespanning van 5V. Deze waarden zijn niet meer meetbaar met de ADC aangezien het hoogste niveau op 5V ligt. Alles hierboven is niet meer te onderscheiden van deze waarde. Er zitten echter nog tollerantiefouten op de gekozen weerstanden. Bij de metingen zal blijken dat

Kanaal (V)	R_1 (k Ω)	R_2 (k Ω)	A	A_{ideaal}
85	1,2	16,0 + 1,6	15,6667	15
60	1,0	15,0	16,0	16
55	1,0	15,0	16,0	16
45	1,0	15,0 + 1,0	17	17
35	1,6	27,0 + 3,3	19,9375	18
5	1,6	39,0	25,375	25

Tabel 3.2: Gekozen weerstanden en versterking

de echte weerstandswaarden aanleiding geven tot versterkingsfactoren die wel binnen de goede grenzen liggen.

3.1.3 Analooq naar digitaal convertor

De omzetting van een analoge meetwaarde naar een digitale voorstelling wordt gedaan met behulp van een analooq naar digitaal-convertor (ADC). Een belangrijke parameter van een ADC is de resolutie, het aantal bits waarmee de analoge waarde wordt voorgesteld. Een resolutie van 8 bits komt overeen met $2^8 - 1 = 255$ niveaus. Als onze referentiespanning 5V ($= V_{DD}$) is, wil dit zeggen dat het kleinste niveau overeenstemt met $\frac{5V}{255} = 19,6mV$. Voor ons eerste kanaal met een versterking van ≈ 15 en een meetweerstand van 56Ω krijgt men:

$$\frac{19,6mV}{15} = 1,3mV \quad (3.4)$$

$$\frac{1,3mV}{56\Omega} = 23\mu A = I_{min} \quad (3.5)$$

De nauwkeurigheid kan opgedreven worden door een resolutie van 10 bit te nemen. 10 bit komt overeen met $2^{10} - 1 = 1023$ niveaus, waarbij het kleinste niveau overeenstemt met $5V/1023 = 4,883mV$. Voor het eerste kanaal geldt dit keer:

$$\frac{4,883mV}{15} = 0,326mV \quad (3.6)$$

$$\frac{0,326mV}{56\Omega} = 6\mu A = I_{min} \quad (3.7)$$

Deze nauwkeurigheid is hoger dan de oorspronkelijk opgestelde nauwkeurigheid, maar hoe nauwkeuriger gemeten kan worden, hoe beter. Belangrijk hier is dat de metingen snel genoeg

gedaan worden. Er is gekozen om verder te werken met een 10 bit ADC. In het deel over de metingen (paragraaf 3.3) zal per kanaal de minimaal meetbare stroom bepaald worden.

In totaal moeten op 6 kanalen metingen gedaan worden. Een eerste mogelijkheid is om 6 aparte ADC's te gebruiken, voor ieder kanaal één. Dit is een zeer snelle parallelle uitvoering, maar zorgt voor veel connecties naar de microcontroller die alles moet verwerken. De microcontroller heeft niet voldoende ingangen en uitgangen om 6 ADC's rechtstreeks te verbinden. Daarbij komt nog dat de microcontroller zelf alle waarden moet verwerken en naar een extern geheugen moet wegschrijven. Dit vergt enige verwerkingstijd. Door de zeer snelle toevoer van meetdata zal de microcontroller niet kunnen volgen.

Een andere oplossing is om slechts 1 ADC te gebruiken en telkens 1 kanaal te selecteren voor omzetting. Het selecteren van een kanaal kan gebeuren met behulp van een analoge multiplexer. Er zijn ADC's beschikbaar waarbij deze analoge multiplexer reeds geïntegreerd is op de chip. Er is geen extra externe component meer nodig en het selecteren van een kanaal voor conversie kan optimaal gebeuren.

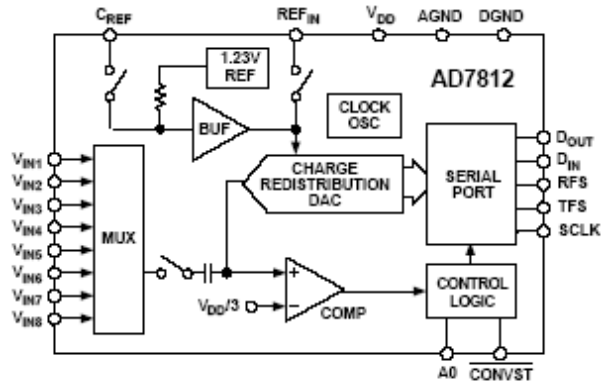
De snelste ADC die voldoet aan deze voorwaarden is de AD7812 van analog devices. Dit is een 10 bit ADC met een conversietijd van $2,3\mu\text{s}$. De tussenniveaus in de aanstuursignalen, van het beeldscherm, zijn $\approx 6\mu\text{s}$ lang. De ADC is snel genoeg om deze belangrijke niveaus te kunnen meten. De chip heeft een multiplexer aan boord die kan kiezen uit 8 ingangen. Het instellen van het juiste kanaal wordt gedaan via seriële communicatie met de microcontroller.

De 10-bit meetwaarden worden op hun beurt teruggestuurd via dezelfde seriële link naar de microcontroller. Dit is een groot voordeel aangezien hierdoor slechts een paar verbindingen met de microcontroller nodig zijn.

Over de digitale specificaties en eigenschappen van de ADC wordt meer verteld in het hoofdstuk over digitale hardware (hoofdstuk 4). Wat ons nu interesseert zijn de analoge eigenschappen van de ADC.

- Conversietijd: $2,3\mu\text{s}$
- DNL: ± 1 lsb max
- SNDR: 58dB min
- Offset error: ± 2 lsb max

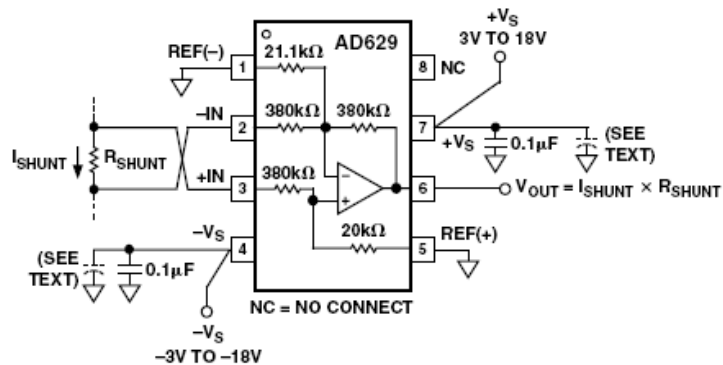
In figuur 3.3 is een schematische voorstelling zichtbaar van de opbouw van de ADC.



Figuur 3.3: Schema ADC

3.2 Schema

De datasheet van de differentiële opamp geeft aan hoe deze component moet aangesloten worden met de benodigde condensators (zie figuur 3.4). Er worden 2 condensators ($0,1\mu\text{F}$ en $10\mu\text{F}$) aangesloten per voedingspin ($V(+)$ en $V(-)$) om de voeding te ontkoppelen. Dit is nodig om ruis te onderdrukken en onstabiliteit in de voeding weg te werken.



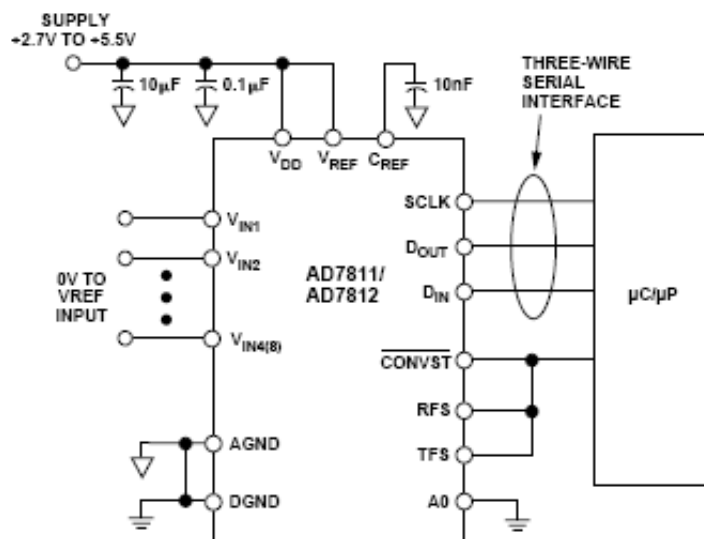
Figuur 3.4: Schema differentiële versterker

Het is mogelijk een voeding te gebruiken van $\pm 3\text{V}$ tot $\pm 18\text{V}$. Het is best slechts 1 voeding te gebruiken om de volledige schakeling te voeden. In eerste instantie is gekozen om een voeding van $\pm 15\text{V}$ te gebruiken, een veel gebruikte waarde bij opamps. Via een spanningsregelaar kan een andere stabiele spanning (5V) voorzien worden voor het digitale gedeelte van de schakeling. De spanningsregelaar werkt correct tot 7V ingangsspanning, waardoor de volledige schakeling in principe tot $\pm 7\text{V}$ kan gevoed worden.

De aansluiting voor de opamp is al uitvoerig besproken in paragraaf 3.1.2. De spanningsversterking wordt ingesteld door een correcte feedback rond de opamp, volgens het principe van een niet-inverterende versterker. Verder moet er nog een potentiometer worden aangesloten om de offset zo laag mogelijk te maken.

Bij de ADC moeten ook nog enkele capaciteiten worden toegevoegd voor de nodige ontkoppeling met voeding en referentiespanning. Een typische aansluiting is zichtbaar in figuur 3.5. Omdat er maar 6 kanalen zijn en 8 ingangen op de ADC moeten de overige 2 ingangen aan massa verbonden worden. Anders kan de interne multiplexer foutief werken. Een open ingang kan immers een signaal opvangen en substraatstromen introduceren, welke een correcte werking kunnen verstoren.

De signalen CONVST, RFS en TFS zullen anders aangesloten worden dan op de figuur. Dit wordt meer in detail besproken in het deel over de digitale hardware.



Figuur 3.5: Typische aansluiting voor de ADC

Het volledige schema van de analoge hardware is te zien in figuur 3.10. Een foto van de implementatie op breadboard is te zien in appendix B.

3.3 Metingen

Per kanaal worden verschillende ingangssignalen aangelegd aan de differentiële versterker. Op basis van deze signalen wordt de uitgang gemeten om de versterking, de offset en de ruisbijdrage

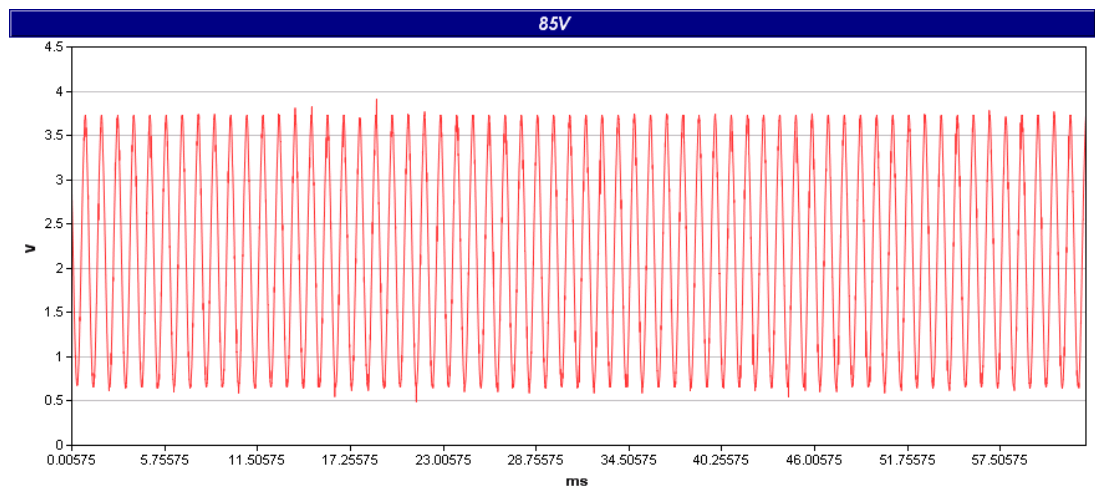
te bepalen op ieder kanaal. De metingen zijn gedaan met het volledige meetcircuit, inclusief computer-programma (zie hoofdstuk 6).

Alle metingen zijn ook vergeleken met een oscilloscoop. De metingen, gedaan door het meetcircuit, zijn dezelfde als deze gemeten met de oscilloscoop. Ter verduidelijking worden enkele grafieken getoond uit het computer-programma.

3.3.1 Versterking

Om de versterking te bepalen is aan iedere differentiële versterker een sinus-sigitaal met frequentie 1kHz aangelegd. De sinus heeft een amplitude van 100mV (200mV piek tot piek). De DC-offset van de sinus is 100mV, zodat de volledige sinus meetbaar is met het meetcircuit.

Met deze instelling wordt de sinus niet volledig weergegeven bij de spanning van 5V. Het hoogste niveau is dan: $0,25V * 25 = 6,25V$. Bij 5V is een kleinere sinus aangelegd met amplitude 50mV en offset 50mV. In figuur 3.6 is een meting getoond op kanaal 85V.



Figuur 3.6: Meting: sinus op kanaal 85V

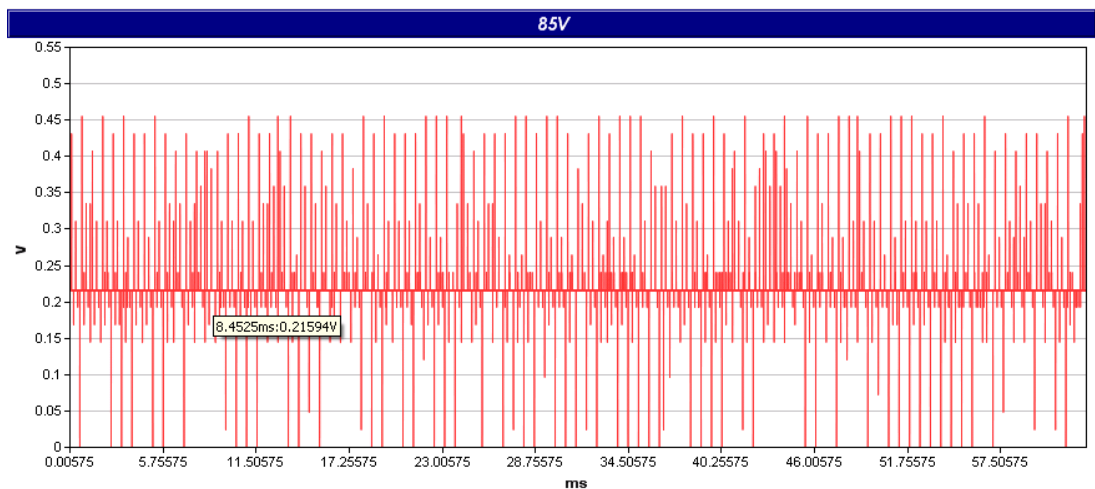
In tabel 3.3 zijn de piek-tot-piek waarden van de gemeten sinussen op de verschillende kanalen weergegeven, samen met de afgeronde versterking. In de laatste kolom is de berekende versterking A weergegeven uit tabel 3.2 ter vergelijking.

Kanaal (V)	V_{ptp} (V)	$A_{gemeten}$	A
85	3,11	15,55	15,6667
60	3,18	15,9	16
55	3,16	15,8	16
45	3,37	16,85	17
35	3,68	18,4	19,9375
5	2,27	22,7	25,375

Tabel 3.3: Gemeten versterking

3.3.2 Offset

Om de offset te bepalen op ieder kanaal worden beide ingangen van de differentiële versterker met massa verbonden. Opnieuw is een meting uitgevoerd op ieder kanaal. In figuur 3.7 is de gemeten offset weergegeven. De gemeten offset per kanaal wordt weergegeven in tabel 3.4.



Figuur 3.7: Meting: Offset op kanaal 85V

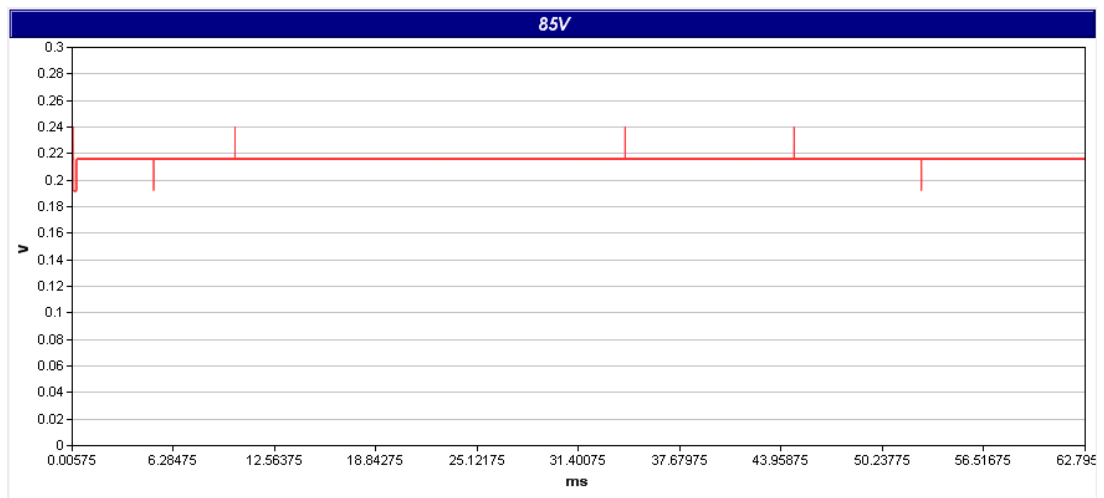
3.3.3 Ruis

In figuur 3.7 is te zien dat er ruis aanwezig is op het gemeten signaal. De tussentijd tussen 2 pieken bedraagt telkens: 0,006ms. Het periodieke karakter van deze ruis maakt het makkelijker om dit weg te filteren. De frequentie van deze ruis bedraagt: $f = \frac{1}{0,006ms} = 166,7kHz$. Dit komt overeen met een tijdsconstante: $\tau = \frac{0,006ms}{2\pi} = 9,5 \cdot 10^{-7}$. Als een condensator wordt toegevoegd

Kanaal (V)	Offset (mV)
85	215,9
60	313,4
55	306,9
45	416,4
35	501,6
5	178,6

Tabel 3.4: Gemeten offset per kanaal

aan de uitgang van de opamp, zal in combinatie met de feedbackweerstand een RC-filter met tijdsconstante $\tau = RC$, gevormd worden. In tabel 3.5 is per kanaal de bijhorende capaciteit C berekend om deze frequentie weg te filteren via de formule: $C = \frac{\tau}{R}$.



Figuur 3.8: Meting: Offset op kanaal 85V met RC-filter

In figuur 3.8 is van kanaal 85V opnieuw de offset gemeten, dit keer met een condensator om het RC-filter te realiseren. De uitgang is duidelijk verbeterd ten opzichte van figuur 3.7.

Wanneer dit filter gebruikt wordt om de golfvormen te meten tijdens het wegschrijven van een beeld, bestaat de kans dat de tussenniveaus (zie hoofdstuk 1) weggefilterd worden. Deze tussenniveaus hebben een tijdsduur van $\approx 6\mu s$. Een oplossing kan zijn om de ruis softwarematig weg te filteren zonder gebruik te maken van een RC-filter. In wat volgt wordt geen filter meer gebruikt, de ruisbijdrage is klein tov. het nuttige signaal.

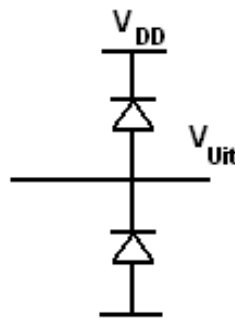
Kanaal (V)	R (k Ω)	C (pF)
85	18,8	50
60	16,0	60
55	16,0	60
45	17,0	56
35	31,9	30
5	40,6	23,5

Tabel 3.5: Berekening RC-filter

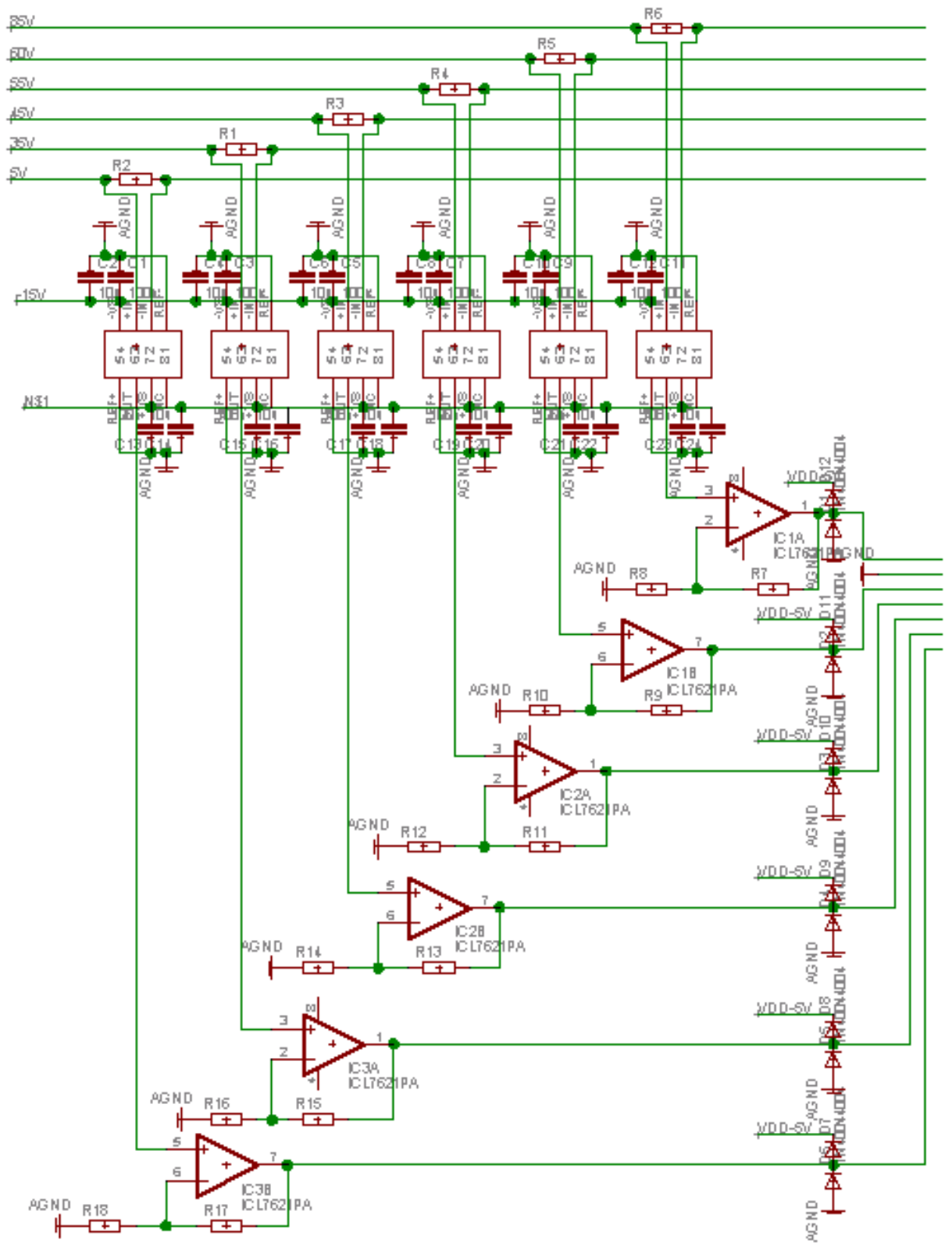
3.3.4 Bescherming ADC

De ADC kan volgens de datasheet geen negatieve ingangsspanning ($< -0,3V$) aan. Ook een spanning boven $V_{DD} + 0,3V = 5,3V$ kan de ADC beschadigen. Het meetcircuit zal echter ook negatieve spanningsvallen meten en versterken. Ook wanneer een spanning boven 5V wordt versterkt, moet de ADC beveiligd worden zodat deze niet kapot gaat.

Dit kan opgelost worden met behulp van diodes. Deze diodes zullen de ingangsspanning aan de ADC tussen de grenzen $-0,7V$ en $5,7V$ houden, zoals verduidelijkt in figuur 3.9. Dit is nog iets buiten de grenzen van de opgegeven waarden in de datasheet, maar vormen toch reeds een grote bescherming. Deze uiterste waarden zullen ook maar weinig bereikt worden.



Figuur 3.9: Diodeschakeling ter bescherming van ADC



Figuur 3.10: Schema analoge hardware

Hoofdstuk 4

Hardware: Digitaal

Nadat in het analoge hardware gedeelte een meting is gedaan van de spanning over de meetweerstand, wordt deze gedigitaliseerd door de ADC. De bekomen 10-bit voorstelling van iedere meting wordt met behulp van een microcontroller verwerkt en opgeslagen in RAM-geheugen. Er wordt een seriële link voorzien om te communiceren tussen de microcontroller en PC, zodat de meetwaarden kunnen opgevraagd en bewerkt worden. In dit hoofdstuk worden de verschillende componenten van de digitale schakeling in meer detail besproken.

4.1 Componenten

4.1.1 AnalooG Digitaal Convertor

In het hoofdstuk over de analoge hardware is reeds dieper ingegaan op de analoge werking van de ADC. In dit hoofdstuk worden de digitale aspecten besproken. Hieronder valt alles wat te maken heeft met de communicatie tussen ADC en microcontroller en het protocol dat gebruikt wordt om de ADC aan te sturen.

Om de ADC in te stellen moet een bitsequentie weggeschreven worden naar het 10 bit controle-register. Via dit register is het mogelijk het ingangskanaal te kiezen, de mode waarin de ADC werkt, de referentiespanning per kanaal, . . . In tabel 4.1 zijn de verschillende functies van het register verduidelijkt.

In deze toepassing zal de ADC gebruikt worden in de snelste modus, dit wil zeggen dat de ADC zichzelf niet mag uitschakelen na een conversie zodat er continu metingen gedaan worden. Op deze manier is het mogelijk 350kSPS, dit zijn 350k metingen per seconde, te halen. De externe pin A_0 wordt aan massa gehangen, waardoor de overeenkomstige bit in het controle-

9									0															
A0	PD1	PD0	V_{IN8}/\overline{AGND}	DIF/\overline{SGL}	CH2	CH1	CH0	\overline{CONVST}	EXTREF															
A ₀	Deze bit maakt het mogelijk meerdere ADC's op eenzelfde seriële link aan te sluiten. De ADC bevat eenzelfde input-pin A ₀ . Als de waarde van deze pin overeenkomt met de A ₀ waarde in het controleregister zal de ADC geselecteerd zijn voor communicatie. Als deze waarden verschillen wordt de seriële communicatie genegeerd en kan een andere ADC deze seriële link gebruiken																							
PD1,PD0	Deze bits bepalen de mode waarin de ADC gebruikt wordt. Er zijn 4 mogelijkheden. Meer uitleg over de gebruikte mode volgt bij de bespreking van het protocol																							
			<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">PD1</th> <th style="width: 10%;">PD0</th> <th style="width: 80%;">Mode</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">ADC volledig uitschakelen</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">Gedeeltelijk uitschakelen na een conversie</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">Volledig uitschakelen na een conversie</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">ADC inschakelen</td> </tr> </tbody> </table>							PD1	PD0	Mode	0	0	ADC volledig uitschakelen	0	1	Gedeeltelijk uitschakelen na een conversie	1	0	Volledig uitschakelen na een conversie	1	1	ADC inschakelen
PD1	PD0	Mode																						
0	0	ADC volledig uitschakelen																						
0	1	Gedeeltelijk uitschakelen na een conversie																						
1	0	Volledig uitschakelen na een conversie																						
1	1	ADC inschakelen																						
V_{IN8}/\overline{AGND}	Als deze bit op 1 staat, worden alle ingangen vergeleken met V_{IN8} , in het andere geval worden alle ingangen vergeleken met de massa																							
DIF/\overline{SGL}	Wanneer deze bit op 0 staat bepaalt V_{IN8}/\overline{AGND} hoe de ingangen gerefereerd zijn, als deze bit op 1 staat zijn alle ingangen differentiëel tov. elkaar: $V_{IN1}/V_{IN2}, V_{IN3}/V_{IN4}, V_{IN5}/V_{IN6}, V_{IN7}/V_{IN8}$																							
CH2,CH1,CH0	Met deze 3 bits wordt het ingangskanaal geselecteerd.																							
\overline{CONVST}	Wanneer deze bit op 1 komt, zal een conversie starten. Na de conversie wordt deze bit gereset naar 0. Het starten van een conversie kan ook gebeuren door te werken met de corresponderende CONVST input-pin																							
EXTREF	Als deze bit op 1 staat wordt de externe referentie als V_{ref} gebruikt.																							

Tabel 4.1: Controleregister van de ADC

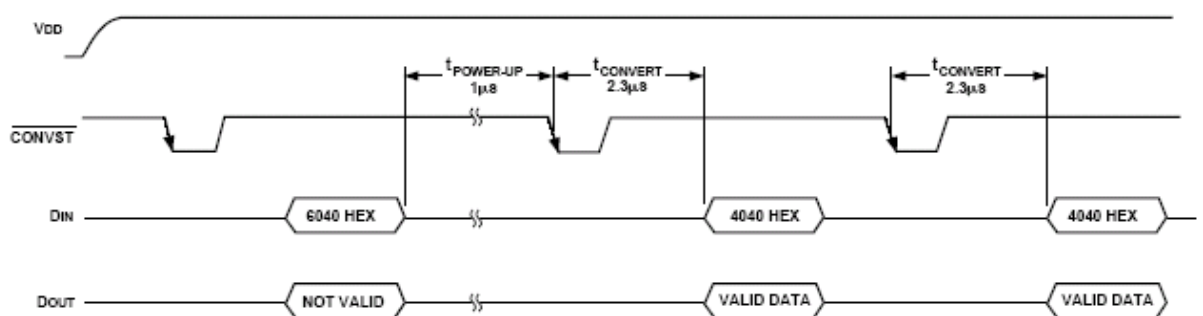
register ook op 0 moet worden gezet. Verder wordt V_{DD} als externe referentie gebruikt en worden alle ingangen tov. de massa gerefereerd. Samengevat wil dit zeggen dat in het controleregister de volgende bitsequentie wordt weggeschreven:

'00100XXX01': De bits met een X zijn variabel, deze bepalen het ingangskanaal

Om de ADC correct in te stellen, metingen te verrichten en de gemeten waarden door te zenden naar de microcontroller, moet een specifiek protocol gevolgd worden. De ADC moet eerst worden opgestart voor hij gebruikt kan worden. Dit wordt gedaan door bits PD1,PD0 = 11 weg te schrijven in het controle register. Na $1,8\mu\text{s}$ is de component volledig opgestart. In het controle-register worden PD1,PD0 terug aangepast, waarna gestart kan worden met meten.

Door de input CONVST laag te brengen wordt de ingang van het geselecteerde kanaal vastgehouden en start de conversie. Een conversie duurt ongeveer $2,3\mu\text{s}$, waarna de data beschikbaar is om door te sturen naar de microcontroller.

Als CONVST laag blijft tot het einde van de conversie, zal de ADC zichzelf uitschakelen zoals ingesteld in de PD1,PD0 bits. Wordt CONVST echter opnieuw hoog voor het einde van een conversie, dan zal de ADC niet uitschakelen en blijven werken in zijn snelle modus. Een volgende conversie kan starten zodra de vorige conversie gedaan is. Er moet wel op gelet worden dat er minstens 100ns gewacht wordt na het doorsturen van een meetwaarde voordat CONVST opnieuw laag wordt. Deze werking is nog eens verduidelijkt in figuur 4.1.



Figuur 4.1: Protocol: Meten in snelle mode

De communicatie tussen ADC en microcontroller maakt eveneens gebruik van een protocol. De pinnen RFS en TFS van de ADC bepalen of er ontvangen of verstuurd wordt. Ontvangen en versturen wordt in deze context gezien vanuit het standpunt van de microcontroller. Ontvangen vindt plaats wanneer de microcontroller een meting ontvangt, versturen wanneer de

microcontroller het controle-register beschrijft.

De communicatie maakt gebruik van 3 extra pinnen: D_{IN} , D_{OUT} en SCLK. Wanneer RFS hoog wordt zal een meetwaarde uit het interne schuifregister van de ADC naar buiten geschoven worden via D_{OUT} . De klokpulsen worden gegenereerd door de microcontroller op de SCLK ingang. In het andere geval als TFS laag wordt, zal D_{IN} zijn hoog impedante staat verlaten en wordt bij iedere klokpuls op SCLK een bit van het controleregister naar binnen geschoven.

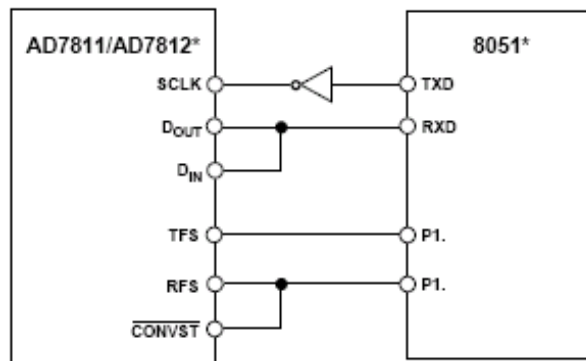
De microcontroller bevat een speciale UART poort om eenvoudig een seriële communicatie te implementeren. Hier wordt later dieper op ingegaan. Belangrijk is dat deze UART gebruik maakt van 2 pinnen, een TXD en RXD pin. Het kloksignaal wordt gegenereerd op de TXD pin en data wordt ontvangen en verstuurd via RXD. De pinnen D_{IN} en D_{OUT} van de ADC worden kortgesloten en beide verbonden met RXD. De link die ontstaat is half-duplex: men kan zowel verzenden als versturen, maar niet op hetzelfde tijdstip.

De schakeling kan nog vereenvoudigd worden door CONVST en RFS samen te nemen. Telkens RFS hoog wordt zal vanaf de eerste klokpuls een waarde worden verzonden naar de microcontroller. Aangezien CONVST opnieuw hoog wordt gebracht voordat de conversie klaar is, is het nuttig RFS hier aan te koppelen. Als het verzenden wordt gestart voordat de conversie klaar is, zal de vorige meetwaarde worden verzonden. Als gewacht wordt met verzenden tot de conversie klaar is, wordt de huidige meetwaarde verzonden.

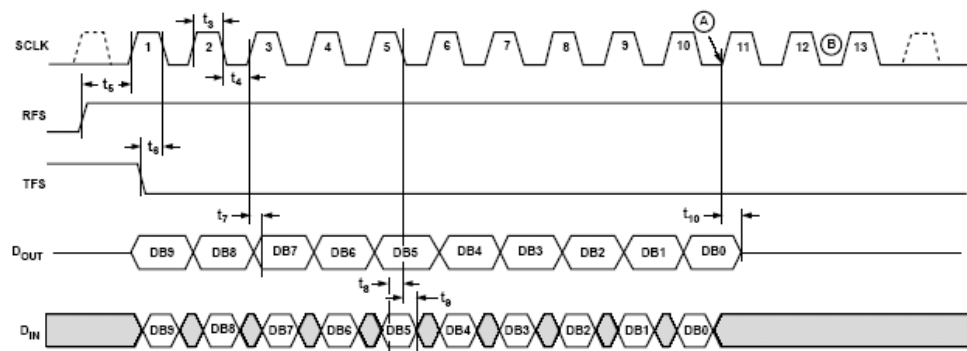
TFS kan in principe ook verbonden worden met RFS en CONVST, maar telkens TFS laag wordt verwacht de ADC dan ook dat er iets verzonden wordt. Als dit gekoppeld is aan CONVST verwacht de ADC dus bij iedere conversie dat er iets wordt weggeschreven. Dit kan handig zijn bij een full-duplex uitvoering, maar in dit geval is het beter TFS apart aan te sturen. De resulterende schakeling is schematisch weergegeven in figuur 4.2. Het verloop in de tijd van de communicatie is weergegeven in figuur 4.3. Op deze figuur is wel gebruik gemaakt van een full duplex verbinding. Wat hier vooral van belang is, is de klokflank na een RFS of TFS signaal die resp. het ontvangen en verzenden initialiseert (dalend bij TFS, stijgend bij RFS).

4.1.2 Microcontroller

De gebruikte microcontroller is de DS89C420 van Maxim/Dallas[5]. Deze werd al gebruikt in de bestaande schakeling en het is voor de hand liggend om deze dan ook te gebruiken voor de metingen. De microcontroller heeft een 8051 architectuur en instructieset met enkele uitbreidingen. Zo zijn er 2 seriële poorten ipv. 1, 3 timers en de belangrijke eigenschap dat er gemiddeld 1



Figuur 4.2: Aansluiting seriële communicatie met ADC



Figuur 4.3: Protocol: Seriële communicatie

instructie per klok-cyclus kan uitgevoerd worden. De maximale kristalfrequentie is 33Mhz. Als de microcontroller op volle snelheid werkt, kan 1 instructie worden uitgevoerd in 30ns.

Met behulp van de juiste rand-componenten kan deze microcontroller ook 'in system' geprogrammeerd worden en dit over dezelfde seriële link die gebruikt wordt om meetdata te versturen naar de pc. Een volledige beschrijving van deze microcontroller zou ons te ver leiden. De geïnteresseerde lezer wordt hiervoor doorverwezen naar de uitgebreide datasheet en userguide[5]. In het volgende hoofdstuk over de software worden de belangrijke onderdelen van de microcontroller verder besproken, met bijhorende assemblercode als voorbeeld.

4.1.3 Ram Geheugen

Het is de bedoeling dat gedurende een bepaalde tijd metingen gedaan worden. Al deze metingen worden opgeslagen in een extern geheugen om later opgevraagd te worden. Het interne geheugen van de microcontroller beschikt slechts over 1kB data-geheugen. Dit is uiteraard veel te weinig

om de enkele duizende metingen in op te slaan. De adresruimte van de microcontroller biedt de mogelijkheid om extern geheugen tot 64kB te gebruiken.

Om een idee te krijgen van de hoeveelheid meetdata die in 64kB kan opgeslagen worden, wordt een eerste ruwe schatting gemaakt. Later wordt dit nauwkeurig nagemeten zodra alle software en hardware samenwerkt. Iedere meetwaarde bevat 10 nuttige bits. Omdat alle waarden opgeslagen worden als een byte wordt deze meetwaarde aangevuld tot 16 bits = 2 bytes. Uitgerekend krijgt men:

$$64kB = 2^{16} = 65536\text{Bytes} \quad (4.1)$$

$$\frac{65536B}{2B} = 32768\text{metingen} \quad (4.2)$$

Deze waarden zijn afkomstig van 6 verschillende kanalen. Uit de vorige paragraaf over de ADC weet men dat op volle snelheid een conversie $2,3\mu\text{s}$ duurt. Stel dat met de extra overhead om de $2,5\mu\text{s}$ een nieuwe waarde beschikbaar is, dan kan men gedurende $13,7\text{ms}$ metingen per kanaal verrichten:

$$\frac{32768\text{metingen}}{6} \approx 5461 \frac{\text{metingen}}{\text{kanaal}} \quad (4.3)$$

$$5461 \frac{\text{metingen}}{\text{kanaal}} * \frac{2,5\mu\text{s}}{\text{metingen}} = 13,7 \frac{\text{ms}}{\text{kanaal}} \quad (4.4)$$

Een eerste keuze die gemaakt moet worden bepaalt het soort geheugen. Er zijn verschillende mogelijkheden: flash, RAM, EEPROM, ... Ram geheugen volstaat in deze toepassing omdat de waarden niet moeten bijgehouden worden nadat de voedingsspanning wegvalt. Maar er bestaan verschillende soorten RAM-geheugens: statisch- en dynamisch RAM-geheugen. Dynamisch RAM-geheugen heeft extra randapparatuur nodig en is ingewikkelder om aan te sturen dan statisch geheugen, waardoor de keuze snel gemaakt is.

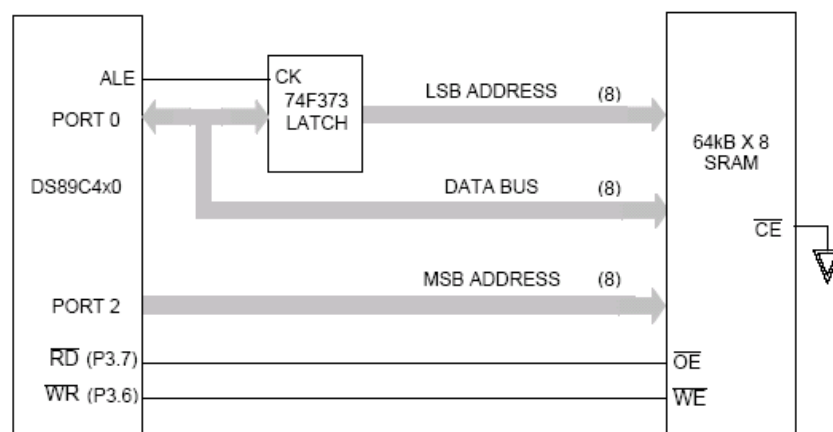
Een tweede belangrijke eigenschap van het geheugen is de toegangstijd. Onze microcontroller heeft de mogelijkheid snelle geheugentoeegang toe te laten. Indien het externe geheugen niet snel genoeg is, kan deze toegangstijd software-matig vertraagd worden. In deze toepassing is het van belang dat alles zo snel mogelijk verloopt, zodat tijdens het meten geen tijd verloren gaat door geheugentoeegangen.

Nu de nodige specificaties gekend zijn, wordt gezocht naar een goede component bij farnell. Het blijkt echter dat een 64kB versie niet beschikbaar is. Het alternatief is 2 geheugens van

32kB te combineren of 1 van 128kB te gebruiken. Het eerste alternatief zorgt dat er nog een extra chip bijkomt met alle nodige randchips. Het is veel eenvoudiger om een chip van 128kB te gebruiken waar slechts 64kB van gebruikt wordt. Als de hoogste adresbit van dit geheugen wordt verbonden met een output van de microcontroller kan op deze manier gewisseld worden tussen 2 geheugenbanken van 64kB. Op deze manier kan indien nodig toch de volledige 128kB gebruikt worden zodat het aantal metingen verdubbeld worden.

Er zijn verschillende modes waarin de microcontroller met het RAM-geheugen kan communiceren. De tijd om een byte weg te schrijven of op te halen uit het RAM-geheugen ligt enkel in mode0 vast. Andere mode's werken met behulp van een page-hit of een page-miss. Er wordt steeds eerst naar een bepaalde pagina gezocht in het geheugen. Wanneer data in dezelfde pagina beschikbaar is, spreekt men van een page-hit. In dit geval is de geheugen-toegang zeer snel. Wanneer echter de gezochte data zich in een andere pagina bevindt moet verder gezocht worden en spreekt men van een page-miss. In dit geval zal de geheugen-toegang lang duren. Omdat het in deze toepassing belangrijk is exact te weten hoe lang een geheugentoeegang duurt, wordt voor mode0 gekozen.

De bestelde component is de BS62LV1024. Dit is een statische CMOS RAM-chip met 128kB geheugen. De toegangstijd van deze chip is 70ns, wat voldoende is om zonder vertraging te communiceren met de microcontroller. In figuur 4.4 is te zien hoe het RAM-geheugen verbonden wordt met de microcontroller (mode0 operatie). De bespreking van de extra randchips volgt in het deel over de randcomponenten (paragraaf 4.1.5).



Figuur 4.4: Interconnectie tussen Ram-geheugen en microcontroller

4.1.4 RS-232 seriële communicatie

Er kan via de seriële poort van de computer gecommuniceerd worden met de microcontroller. De computer maakt hiervoor gebruik van het RS-232 protocol. Dit houdt onder andere in dat spanningen gebruikt worden van $\pm 12V$. De microcontroller beschikt eveneens over een seriële poort. Eén daarvan wordt gebruikt voor de communicatie met de ADC, de andere zal gebruikt worden om te communiceren met de computer. Aangezien de microcontroller werkt op 5V en de computer gebruik maakt van een 12V RS-232 protocol moet een omzetting gedaan worden naar het RS-232 protocol.

Ook hier bestaan veel geschikte chips die deze omzetting kunnen maken. Deze chips maken gebruik van ladingspompen om vanuit 5V een 12V signaal op te bouwen. Meestal maken deze chips dan ook gebruik van speciale en grote elektrolytische condensatoren. Makkelijker is als de chip gebruik maakt van standaard capaciteiten om de ladingspomp te implementeren. De MAX3232CPE is een chip die gebruik maakt van gewone $0,1\mu F$ capaciteiten. Dit zijn dezelfde capaciteiten die gebruikt worden om de voeding te ontkoppelen vóór een (digitale) chip.

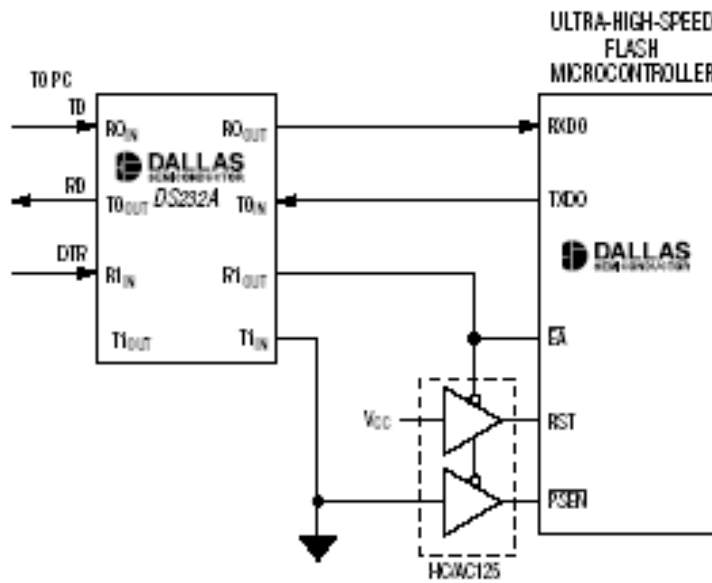
De MAX2323CPE is een RS-232 tranceiver met 2 kanalen aan boord. Enkele kenmerken:

- 3 tot 5,5V voedingsspanning
- Tot 1Mbps mogelijk

De seriële link met de computer zal gebruikt worden om alle meetdata uit het geheugen door te sturen naar de pc. Dezelfde seriële link kan echter ook gebruikt worden om de microcontroller te programmeren. Men spreekt van 'in system'-programmeren omdat de microcontroller niet uit het circuit moet gehaald worden om te programmeren. In figuur 4.5 is een schakeling voorgesteld die dit mogelijk maakt. Het seriëel programmeren van de microcontroller is mogelijk door het algoritme in figuur 4.6 te volgen. De extra randcomponenten zorgen voor de correcte signalen.

4.1.5 Randcomponenten

Om de verschillende digitale componenten te laten samenwerken, zijn nog enkele randcomponenten nodig. Zo moet de klok geïnverteerd worden bij de communicatie tussen microcontroller en ADC (zie ook figuur 4.2). Om de microcontroller te laten communiceren met het RAM-geheugen is er nog een extra latch nodig. Dit is een component die 8 bits kan vasthouden en weer doorgeven, wat nodig is om de 16bit adresruimte te adresseren vanuit één 8 bit poort (zie



Figuur 4.5: In-system programmeer circuit

figuur 4.4). Een laatste randcomponent vindt men terug in het circuit om de microcontroller te programmeren (zie figuur 4.5).

Deze 3 componenten worden geselecteerd uit de standaard 74HC reeks. Dit is een reeks met snelle CMOS-componenten:

- 74HC04: 6 invertoren op 1 chip
- 74HC125: 3 tri-state buffers
- 74HC373: 8 bit latch

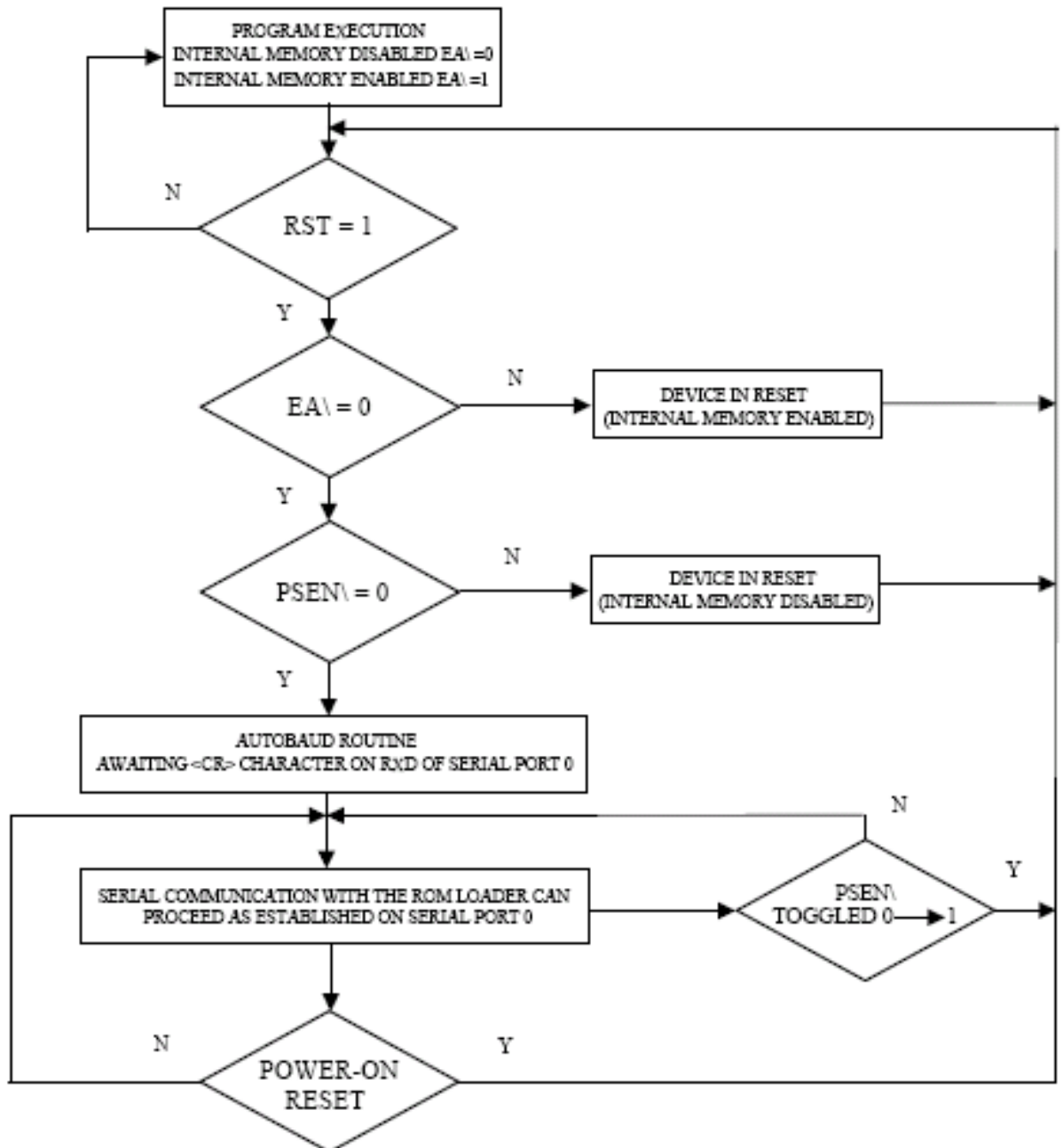
Naast deze componenten heeft de microcontroller nog nood aan een kristal om de frequentie te genereren. Er wordt gekozen voor de maximale frequentie: 32MHz. Er moet op gelet worden dat het kristal in zijn fundamentele mode resoneert op deze frequentie, anders kunnen fouten optreden.

Om testen te vergemakkelijken en om in het uiteindelijke programma bepaalde bewerkingen aan te geven, worden nog enkele leds besteld. Deze leds hebben reeds een weerstand aan boord zodat geen extra weerstand moet toegevoegd worden in serie met een led om ze te doen werken.

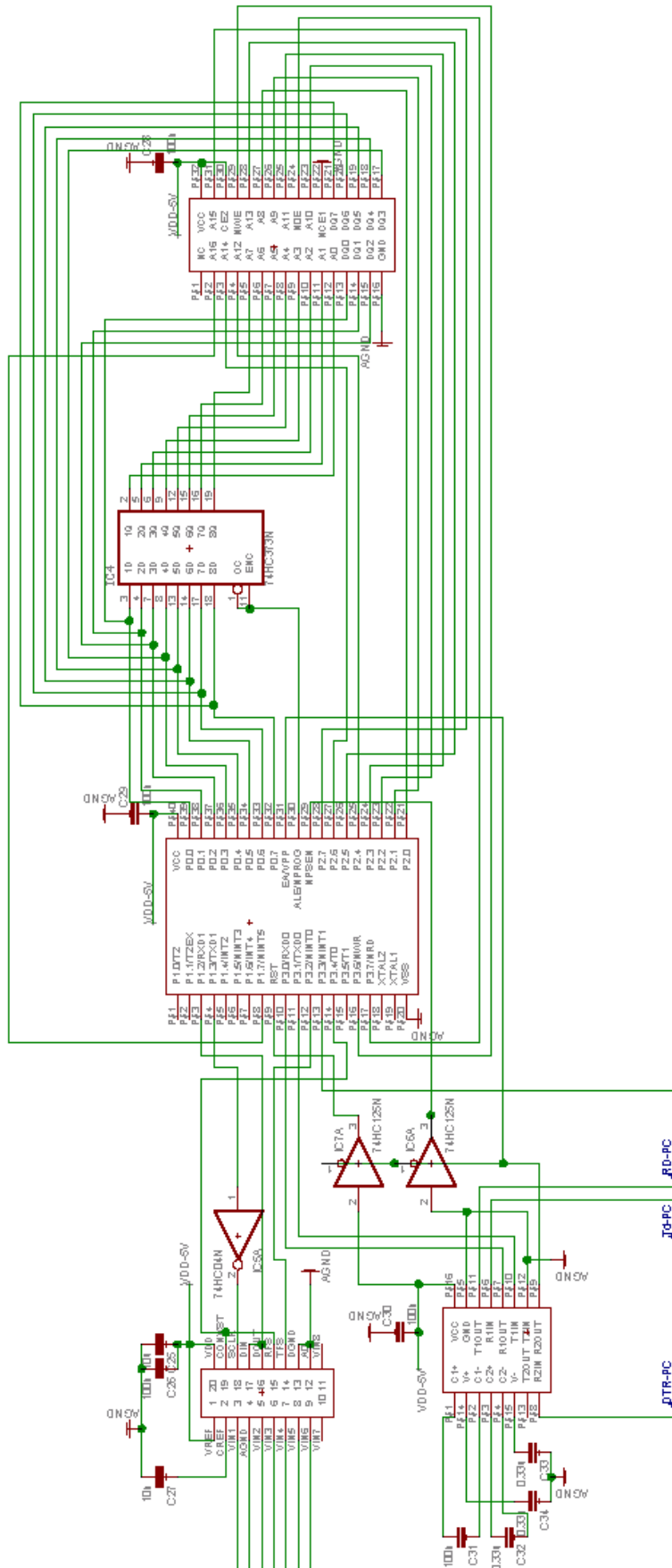
De volledige lijst van bestelde componenten is weergegeven in appendix A.

4.2 Schema

Alle componenten worden verbonden zoals aangegeven in de datasheets en reeds besproken in voorgaande paragrafen. Het volledige digitale schema is zichtbaar in figuur 4.7. In bijlage B is een foto zichtbaar van de volledige implementatie op breadboard.



Figuur 4.6: Algoritme om de uC seriëel te programmeren



Figuur 4.7: Digitaal schema

Hoofdstuk 5

Software

In de vorige hoofdstukken is het volledige circuit opgebouwd. De analoge hardware voert reeds correcte metingen uit, maar de digitale hardware werkt niet zonder software. In totaal moet de microcontroller 3 taken uitvoeren: communicatie met de ADC, communicatie met de computer en data wegschrijven naar geheugen. Elk van deze 3 taken kan uitgevoerd worden door correcte software te schrijven en in te laden in de microcontroller.

Eerst worden de verschillende onderdelen van het assemblerprogramma voor de microcontroller toegelicht. Nadien volgt een beschrijving van de pc-applicatie. In het volgende hoofdstuk worden beide onderdelen samengevoegd tot een werkend geheel.

5.1 Microcontroller

De gebruikte programmeertaal is assembler. Zoals reeds gezegd, wordt gebruik gemaakt van een uitgebreide 8051-instructieset. Stap voor stap wordt de code opgebouwd en samengevoegd. Het is belangrijk om er rekening met te houden dat het testen en localiseren van fouten niet evident is. Om beter te kunnen debuggen wordt eerst de seriële link met de computer geïmplementeerd. Zodra dit zonder fouten werkt, is het mogelijk om bepaalde variabelen tijdens de werking van het programma door te sturen naar de computer om te debuggen. In een tweede stap wordt het Ram-geheugen aangestuurd. Vervolgens komt de ADC aan beurt, waarna als laatste stap enkele versies van een volledig programma worden getest.

5.1.1 Seriële link

Het doel van dit programma is het genereren van enkele gekende getallen en door te sturen naar de computer. Op de computer kan dan gecontroleerd worden of de getallen correct zijn.

De seriële link met de computer maakt gebruik van poort0 op de microcontroller. De communicatie werkt in mode 1. Dit wil zeggen dat aan iedere byte een start- en stopbit wordt toegevoegd.

Zowel de microcontroller als de computer genereren een klok, het is belangrijk deze goed op elkaar af te stemmen (baudrate). Er kan pas een correcte communicatie tot stand worden gebracht als de computer en de microcontroller op dezelfde baudrate werken. Dit is de snelheid waarmee de verschillende symbolen worden doorgestuurd.

Het instellen van de baud-rate op de microcontroller wordt gedaan met behulp van een teller. Deze teller kan ingesteld worden om telkens op een bepaalde waarde te herstarten. In combinatie met de frequentie waarop de teller werkt kan een baud-rate gegenereerd worden.

Als teller wordt teller 1 gekozen. De teller wordt ingesteld zodat hij slechts in 8 bit telt en automatisch herstart. Het is niet nodig een interrupt te laten genereren telkens de teller zijn eindwaarde bereikt. Dit komt overeen met een werking in mode 1 voor teller 1. De gebruikte registers om de communicatie in te stellen zijn:

Register	Waarde	Werking
SCON0.7	1	Poort 0 instellen op mode 1
CKCON.4	T1M	Zie tabel 5.1
CKMOD.4	TH1M	Zie tabel 5.1
PMR.7,6,3	CD1,CD0,4X/2 \bar{X}	Zie tabel 5.1
PCON.7	SMOD_0	Baud-rate verdubbeling
TMOD.5	1	Timer 1 in mode 1
TCON.4	TR1	Timer 1 starten en stoppen
TL1,TH1		Laagste en hoogste byte van het teller-register

De overige bits in de registers moeten niet aangepast worden en staan default goed. Voor meer uitgebreide uitleg wordt verwezen naar de user-guide van de microcontroller[5]. Als de teller in mode 1 werkt, wordt in TH1 de waarde gestokeerd waarop de teller moet herstarten. TL1 is het register waar de huidige waarde van de teller beschikbaar is. Telkens TL1 = TH1 zal de teller herstarten.

Systeem-klok mode	PMR-register (CD1,CD0,4X/ $\overline{2X}$)	Timer1 input-frequentie: T1MH,T1M		
		00	01	1X
Klok *4	001	OSC / 12	OSC / 1	OSC / 0,25
Klok *2	000	OSC / 12	OSC / 2	OSC / 0,5
Klok *1: default	01X, 10X	OSC / 12	OSC / 4	OSC / 1
Klok / 1024	11X	OSC / 3072	OSC / 1024	OSC / 1024

Tabel 5.1: Instelling timer1-registers (X=dont-care bit)

De baud-rate kan nu ingesteld worden door TH1 in te stellen op basis van volgende formule:

$$TH1 = 256 - \frac{2^{SMODx} * \text{timer1inputfrequentie}}{32 * \text{baud} - \text{rate}} \quad (5.1)$$

Hoe hoger de frequentie genomen wordt, hoe moeilijker het is om een correcte match te krijgen met de baud-rate gegenereerd op de computer. Een kleine afwijking in de baudrate zal bij hogere frequentie sneller zorgen voor een fout tijdstip waarop data wordt ingelezen. In eerste instantie wordt gewerkt op 600 baud. Dit werkt zeker, waardoor het mogelijk wordt extra te debuggen.

Later wanneer het volledige ramgeheugen (128kB) wordt doorgezonden naar de computer, is duidelijk dat dit veel te traag is. Het volledig doorsturen neemt ongeveer 3 kwartier in beslag. De snelheid wordt opgevoerd door telkens een nieuwe baud-rate in te stellen en een groot aantal waarden door te sturen. Wanneer alle waarden correct toekomen, is een goede baudrate ingesteld.

In tabel 5.2 zijn per baud-rate de verschillende register-instellingen weergegeven, waarbij TH1 is berekend met formule 5.1. Experimenteel is vastgelegd dat tot 115200 baud een correcte communicatie mogelijk is. Dit is eveneens de snelst mogelijke baud-rate. Nu duurt het doorsturen van het volledige ramgeheugen ongeveer slechts 15s.

In bijlage C is de volledige assemblercode te vinden van het programma dat de uiteindelijke metingen zal uitvoeren. De code die met de seriële communicatie naar de computer te maken heeft, wordt nu verder toegelicht.

Baud	T1MH,T1M	SMOD_0	TH1	correct
600	0,0	0	76H	ok
9600	0,0	0	F7H	ok
14400	0,0	1	F5H	ok
19200	0,0	1	F7H	ok
56000	0,0	1	FDH	ok
115200	0,0	1	FEH	niet ok
115200	0,1	0	FEH	niet ok
115200	0,1	1	FBH	niet ok
115200	1,0	1	EEH	ok

Tabel 5.2: Geteste baud-rates met timer 1 instelling

De instelling van de verschillende registers:

```

MOV     SCON0 ,#40H           ; Comm 0 in mode 1
MOV     PCON ,#80H           ; SMOD_0 = 1
MOV     CKCON ,#00H          ; Timer1: T1M = 0
MOV     CKMOD ,#10H          ; Timer1: T1HM = 1: '00010000'
MOV     TMOD ,#20H           ; Timer 1 in mode 2: '00100000'
MOV     TCON ,#40H           ; Timer 1 starten: '01000000'
MOV     IE ,#90H             ; Interrupt voor comm 0 opzetten
MOV     TH1 ,#EEH            ; TH1 , 115200baud
MOV     TL1 ,#00H            ; Tellerregister

```

Zodra alle registers correct zijn ingesteld kunnen bytes verzonden en ontvangen worden. Er moet telkens gewacht worden tot de byte goed is verzonden of ontvangen. Dit wordt gerealiseerd door het register R0 op 0 te initialiseren. De interrupt die gegenereerd wordt als de transmissie voltooid is, zal ervoor zorgen dat R0 verhoogd wordt.

```

; Verzenden
MOV     R0 ,#00H
MOV     R1 ,#01H             ; '1' verzenden
MOV     SBUF0 ,R1
WACHT_TRANSMIT:
CJNE   R0 ,#01H,WACHT_TRANSMIT ; wachten tot byte is verzonden

```

```

; Ontvangen
MOV      R0 ,#00H
ORL      SCON0 ,#10H           ; Receive enable aanzetten
MOV      SBUF0 ,R1
WACHT_RECIEVE :
  CJNE   R0 ,#01H,WACHT_RECIEVE ; wachten tot byte is verzonden
MOV      A ,R3                 ; Ontvangen waarde beschikbaar in R3

```

De interrupt service routine voor de seriële communicatie controleert op basis van het SCON0 register of een byte is ontvangen of verzonden. Wanneer een byte ontvangen is wordt de ontvangstbuffer uitgelezen en het resultaat wordt opgeslagen in R5:

```

ORG      0023H                 ; Interrupt service routine van de
                                ; seriele poort 0 (naar pc)
ANL      SCON0 ,#EFH           ; Receive enable af
MOV      A ,SCON0
ANL      A ,#03H               ; Laatste 2 bits, TI&RI, afzonderen
CJNE    A ,#01H,TRANSMIT      ; RI=1: Een byte is ontvangen
RECIEVE: MOV      R5 ,SBUF0     ; Buffer-waarde opslaan in R3
TRANSMIT: ANL     SCON0 ,#FCH    ; TI&RI bit resetten
INC      R0
RETI

```

5.1.2 Ram-geheugen

Data wegschrijven naar het RAM-geheugen en terug uitlezen is vrij eenvoudig. Er bestaat een assembler-instructie, MOVX, die zorgt voor de externe toegang. Default gebruikt de microcontroller zelfs geen intern datageheugen. Er is 1kB aan intern datageheugen beschikbaar. Dit kan ingeschakeld worden door het PMR-register correct in te stellen.

De adresruimte van de microcontroller biedt slechts toegang tot 64kB geheugen. Om toch de volle capaciteit van het RAM-geheugen te gebruiken, wordt de hoogste adres-pin verbonden met P1.7 (zie hoofdstuk 4). Op deze manier wordt het geheugen opgedeeld in 2 banken.

Omdat er gewerkt wordt met een snelle microcontroller bestaat de kans dat het externe geheugen niet snel genoeg is. De microcontroller heeft, in mode 0, 2 cycli (8 klokpulsen) nodig voor 1 geheugentoeegang. Indien nodig kan de geheugentoeegang softwarematig langer worden gemaakt. Deze zogenaamde 'stretch'-waarde zorgt voor extra cycli tijdens een geheugentoeegang.

Register	Waarde	Werking
PMR.0	0	Geen intern geheugen (default)
CKCON.2-0 (MD2:0)	000	Stretch waarde 0
DPS.6	X	0: INC = increment, 1: INC = decrement
DPS.4	1	Automatisch INC bij een DPTR-instructie
ACON.7	0	Non-page mode (default)

Tabel 5.3: Instelling RAM registers

Dit wordt ingesteld in het CKCON register. Het RAM-geheugen in deze toepassing heeft een toegangstijd van 70ns. Dit komt overeen met 2,24 klokpulsen. Het is niet nodig een stretch-waarde in te voeren omdat het RAM-geheugen snel genoeg is.

Het geheugenadres wordt bijgehouden in het 16-bit DPTR register. Dit register bestaat uit 2 8-bit registers: DPH en DPL. DPH verwijst naar de hoogste byte en DPL naar de laagste. In het DPS register kan ingesteld worden dat DPTR automatisch verhoogt of verlaagt na een geheugentoeegang. Op deze manier is het niet meer nodig zelf een INC DPTR instructie uit te voeren.

In het DPL-register kan ingesteld worden wat het effect is van een INC instructie. Zo kan de INC instructie ingesteld worden om de datapointer te verlagen ipv te verhogen. Dit is nodig omdat er geen DEC instructie beschikbaar is voor 16-bit registers. Om de datapointer toch te verlagen moet de INC instructie gebruikt worden, met het DPS register ingesteld om 1 te verlagen.

Het volledige programma wordt getest door een gekende sequentie getallen te genereren en deze op te slaan in het ram-geheugen. Nadien worden al deze waarden terug uitgelezen en ter controle doorgezonden naar de computer via de seriële link die reeds is opgebouwd. De sequentie $x[i+1] = (x[i] + 1) \bmod 256$ mag niet worden gebruikt bij de test. Wanneer het RAM-geheugen niet functioneert (of zelfs niet aanwezig is), zal de microcontroller de waarden van deze sequentie teruggeven bij het inlezen.

Een samenvatting van de gebruikte registers met hun instelling is weergegeven in tabel 5.3. In het volgende programma worden de registers ingesteld en wordt een byte weggeschreven naar het RAM-geheugen. De datapointer wordt opnieuw goed gezet om dezelfde byte terug uit te lezen. Het programma illustreert alle nodige instructies om met het externe geheugen te werken.

```

MOV      ACON ,#00H           ; Non-page mode
ANL      P1 ,#7FH            ; Pin P1.7=0, Ram geheugen bank 0
ORL      P1 ,#10H           ;      P1.7=1, Ram geheugen bank 1
MOV      DPS ,#14H          ; '00010100' DPTR incr na een toegang
MOV      DPL ,#00H          ; Datapointer, laagste byte
MOV      DPH ,#00H          ; Datapointer, hoogste byte
MOV      A ,#01H
MOVX     @DPTR ,A           ; Waarde in A wordt weggeschreven
ORL      DPS ,#40H          ; INC voert DEC uit
INC      DPTR                ; DPTR = DPTR-1
MOVX     A ,@DPTR           ; Waarde A terug uitlezen

```

5.1.3 ADC-controle

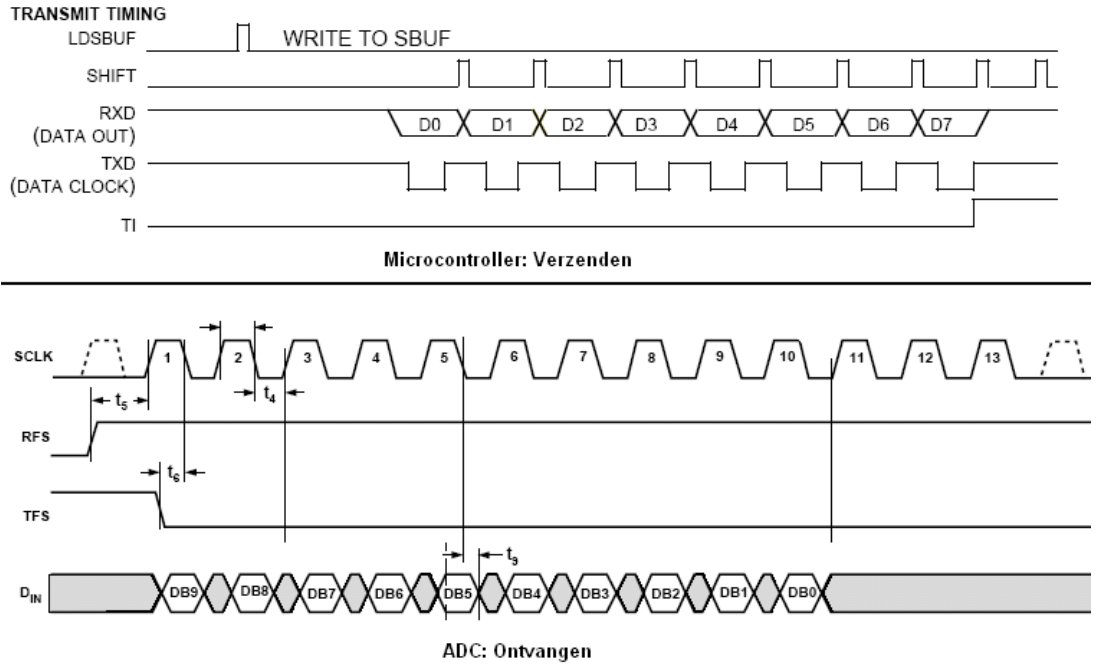
In hoofdstuk 4 is de werking van de ADC in grote lijnen reeds besproken. Dit was nodig om de ADC correct te kunnen aansluiten. In figuur 5.5 is schematisch weergegeven hoe het protocol moet geïmplementeerd worden. De controlestructuur zal worden bepaald wanneer een volledig programma wordt geïmplementeerd (zie 5.1.4).

De aansturing van de ADC wordt bepaald door het 10-bit controleregister. Hierin worden de verschillende kanalen geselecteerd en de mode waarin gewerkt wordt. De waarden van dit register zijn eveneens ter herinnering weergegeven in de flowchart.

De communicatie tussen de microcontroller en de ADC vraagt wat extra aandacht. Zowel het controleregister als de meetwaarden bestaan uit 10 bit. De microcontroller werkt met 8 bit. Om toch een goede communicatie tot stand te brengen stelt de datasheet voor om een 16 bit communicatie te gebruiken, met behulp van een 8 bit microcontroller. Men realiseert dit door 2 bytes te verzenden, de hoogste byte eerst.

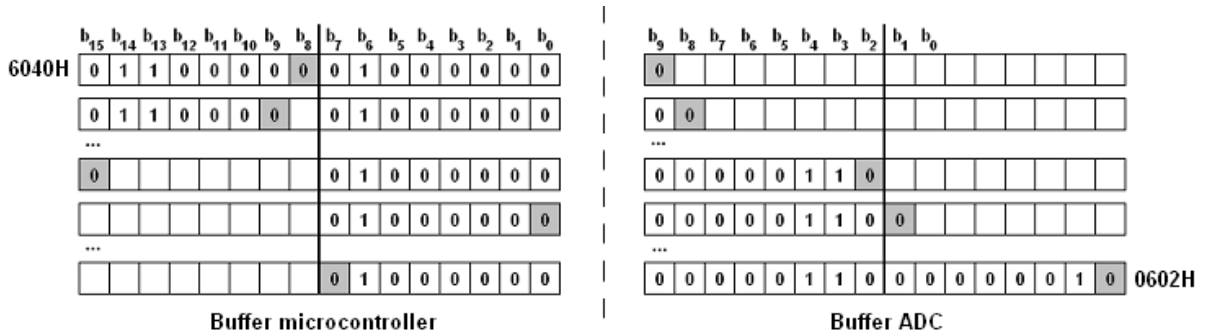
Het is niet mogelijk op deze manier correct te communiceren. De reden hiervoor is dat de microcontroller telkens de laagste bit eerst verzend, maar de ADC verwacht de hoogste bit eerst zoals aangegeven in figuur 5.1. Het probleem wordt verduidelijkt aan de hand van een voorbeeld:

Om de ADC op te starten moet 6040H verzonden worden (zie figuur 5.5). In bitsvoorstelling geeft dit: '01100000 01000000'. Dit wordt doorgezonden per byte, waarbij de hoogste byte eerst verzonden wordt. De ADC zal 6040H ipv 0602H ontvangen (zie figuur 5.2).



Figuur 5.1: Seriële communicatie met ADC

Voordat het 16-bit getal verzonden wordt, moeten alle bits omgekeerd worden per byte. De eerste bit wordt de laatste, de tweede bit de voorlaatste, ... : $b_i = b_{7-i}$ voor elk van de 2 bytes. Deze nieuwe voorstelling wordt doorgestuurd, opnieuw MSB eerst. Het roteren van de bits wordt in 9 stappen gedaan in assembler. Bit per bit wordt de nieuwe byte opgebouwd door telkens een bit te isoleren en goed te zetten. Al deze bijdragen worden op het einde samengevoegd via een OR instructie. Voor de volledige implementatie wordt verwezen naar de bijlage C.



Figuur 5.2: Voorbeeld: verzenden en ontvangen

Register	Waarde	Werking
SCON1.7-6	00	Mode 0
SCON1.5	1	Systeemklok /4 ipv /12
IE.6	1	Interrupt voor seriële communicatie op poort 1
SCON.4	X	0: Voor verzenden - 1: Voor ontvangen

Tabel 5.4: Instelling registers ADC

1. roteer 1 naar rechts AND #80H: $'b_00000000'$
2. roteer 3 naar rechts AND #40H: $'0b_1000000'$
3. roteer 3 naar links AND #20H : $'00b_200000'$
4. roteer 1 naar links AND #10H : $'000b_30000'$
5. roteer 1 naar rechts AND #08H: $'0000b_4000'$
6. roteer 3 naar rechts AND #04H: $'00000b_500'$
7. roteer 3 naar links AND #02H : $'000000b_60'$
8. roteer 1 naar links AND #01H : $'0000000b_7'$
9. Resultaat = OR alle vorige resultaten: $'b_0b_1b_2b_3b_4b_5b_6b_7'$

De seriële link met de ADC wordt geïmplementeerd met de seriële poort 1 van de microcontroller. De ADC verwacht dat deze communicatie in mode 0 werkt, wat wil zeggen dat de computer de klok genereert. Afhankelijk van de RFS en TFS pin wordt dan bij iedere klokpuls een databit ontvangen of verzonden. Voor deze mode moet geen baud-rate ingesteld worden. De systeemklok wordt gebruikt voor het genereren van de klokpulsen. Afhankelijk van de instelling van de SCON1.5 bit wordt de systeemklok gedeeld door 12 of door 4. Een overzicht van de gebruikte registers en hun instelling is weergegeven in tabel 5.4

De volledige aansturing van de ADC is eveneens terug te vinden in bijlage C. Om de code beter leesbaar te maken worden verschillende onderdelen van het programma toegelicht.

De seriële communicatie maakt, net zoals bij de communicatie met de computer, gebruik van een interrupt om aan te geven als een byte is verzonden of ontvangen. Ook hier wordt register R0 gebruikt als register om aan te geven wanneer een byte is verzonden of ontvangen. Wanneer een byte wordt ontvangen door de ADC, wordt deze tijdelijk opgeslagen in R5 om in het hoofdprogramma verder te kunnen verwerken. De interrupt-code is volledig analoog aan de code voor de seriële communicatie met de computer.

Voor een meting kan gedaan worden, moet de ADC opgestart worden. Dit wordt gedaan

door 6040H door te sturen. Nadien wordt een kanaal geselecteerd voor conversie door 2X40H door te zenden. De X moet vervangen worden door het kanaal nummer: 0 voor het 1ste kanaal, enz. Deze hoogste byte wordt in register R2 opgeslagen. Door R2 te incrementeren kan een volgend kanaal geselecteerd worden.

Zoals besproken moeten deze waarden doorgestuurd worden in gespiegelde versie. 6040H wordt zo 0602H en 2X40H wordt XX02H. Voor het omzetten van de 2X waarde naar zijn gespiegelde versie wordt het subprogramma 'SPIEGEL' aangeroepen. De om te zetten byte wordt weggeschreven in R3, het resultaat wordt eveneens in R3 teruggegeven. Na het opstarten van de ADC moet $1\mu\text{s}$ gewacht worden, voordat de eerste meting start. Deze μs kan gebruikt worden om de omzetting te doen. Een volledige omzetting neemt $2\mu\text{s}$ in beslag. Onderstaande code zorgt voor het correct opstarten van de ADC en de omzetting van het R2 register.

```

; Power up ADC... => 6040H wegschrijven naar ADC en 1us wachten
ADC_POWER: ANL      P3, #FBH          ; TFS = 0 -> Wegschrijven
MOV        R0, #00H
MOV        SBUF1, #06H              ; 0602H verzenden, MSB eerst
WACHT_ADC:
CJNE      R0, #01H, WACHT_ADC       ; Wachten tot data verzonden is
MOV        SBUF1, #02H              ; Volgende byte verzenden
WACHT_ADC1:
CJNE      R0, #02H, WACHT_ADC1     ; Wachten tot data verzonden is
MOV        A, R2                    ; Register voor kanaalselectie
                                                ; R2 spiegelen
MOV        R3, A
LCALL     SPIEGEL                   ; Dit duurt 2us
ORL        P3, #04H                 ; TFS = 1

```

De ADC is opgestart en er kan een meting gedaan worden. Opnieuw moet eerst een waarde worden weggeschreven in het controleregister van de ADC. Hiervoor wordt R3, de gespiegelde versie van R2, en 02H verzonden. Dit is volledig analoog aan bovenstaande programma, waarbij 06H vervangen wordt door R3.

Een conversie duurt $2,3\mu\text{s}$. Om geen tijd te verliezen, wordt al een vorige meting ontvangen tijdens de conversie van de huidige meting. De transmissie van 1 byte duurt $\approx 2,2\mu\text{s}$. Wegschrijven naar het RAM-geheugen $\approx 0,26\mu\text{s}$. Het ontvangen en opslaan van 2 bytes zal dan $\approx 5\mu\text{s}$ in beslag nemen. Dit is ruim voldoende om de huidige conversie te laten aflopen. Wanneer de

transmissie gedaan is kan onmiddellijk gestart worden met de volgende conversie.

Tijdens de allereerste conversie is het ook mogelijk om 2 bytes te ontvangen, maar deze zullen geen geldige waarden voorstellen. De bytes worden wel ontvangen en opgeslagen om een goede timing te behouden. Nadien wordt DPTR terug op 0000H gezet om de bytes te negeren. Register R7 bepaalt of sprake is van een eerste conversie.

Onderstaande programma start een omzetting en ontvangt de voorgaande omzetting. Deze 2 bytes worden weggeschreven in het RAM-geheugen. Register R7 wordt gecontroleerd. Indien dit de eerste conversie is, zal R7 gelijk zijn aan 00H en wordt de datapointer gereset:

```

INSTELLING_OK:
ANL      P3,#EFH                ; CONVST = 0 ; meting starten
MOV      R0,#00H
ORL      P3,#10H                ; CONVST = 1 ; puls lang genoeg
MOV      R0,#00H                ; Data onmiddellijk ontvangen
ORL      SCON1,#10H             ; '00110000': Recieve enable
WACHT_SERIAL1_2: CJNE      R0,#01H,WACHT_SERIAL1_2
MOV      A,R5
MOV      R6,A
ORL      SCON1,#10H             ; Receive enable aan
WACHT_SERIAL1_3: CJNE      R0,#02H,WACHT_SERIAL1_3 ;
MOV      A,R6                    ; Data wegschrijven: duur 0,57us
MOVX     @DPTR,A
MOV      A,R5
MOVX     @DPTR,A                ; Duur totaal: 2,2us*2 transmissie
                                   ; + 0,57us RAM-access = 4,97us
CJNE     R7,#00H,EERSTE_CONV_1
DEC      DPL                    ; Eerste conversie: DPTR terug op 0
DEC      DPL

```

Zolang er plaats is in het geheugen kunnen metingen uitgevoerd worden. Wanneer alle metingen gedaan zijn, moeten alle bytes opnieuw gespiegeld worden voor ze een correcte meting voorstellen. Opnieuw wordt het volledige geheugen doorlopen en telkens wordt het subprogramma 'SPIEGEL' aangeroepen. Op het einde van het programma worden alle meetwaarden doorgezonden naar de computer.

De bytes die toekomen op de computer moeten nog samengevoegd worden tot een 10-bit getal. Dit 10-bit getal stelt een niveau voor tussen 0 en 1023. Dit niveau wordt omgezet tot een

Kanaal	Multimeter (V)	ADC (V)
85V	4,84	4,831
60V	3,61	3,622
55V	2,39	2,408
45V	1,202	1,192
35V	0,001	0
5V	4,84	4,832

Tabel 5.5: Controle meetwaarden

spanning door het te vermenigvuldigen met het spanningsverschil tussen 2 niveau's:

$$\text{niveau} = \text{byte1} * 256 + \text{byte2} \quad (5.2)$$

$$\text{spanning} = \text{niveau} * \frac{V_{ref}}{1023} \quad (5.3)$$

Om de implementatie te testen worden gekende spanningen op de ADC aangesloten. Met behulp van een spanningsdeler worden 6 verschillende spanningen aangelegd. De spanningen worden nagemeten met een multimeter ter controle. In tabel 5.5 is per kanaal de spanning, gemeten met de multimeter, vergeleken met de spanning gemeten door de ADC.

De spanning gemeten door de ADC is gemeten met behulp van het uiteindelijke programma. Het programma kan ingesteld worden om 65533 metingen uit te voeren op eenzelfde kanaal. Al deze meetwaarden zijn niet altijd gelijk, hiervan is het gemiddelde genomen als gemeten waarde door de ADC. Dit is te verklaren omdat de gebruikte voeding niet stabiel genoeg is en omdat de weerstanden zelf ook ruis introduceren. De gemiddelde waarde wijkt een beetje af van de waarde gemeten met de multimeter. De multimeter zorgt zelf voor een andere uitmiddeling, waarvan dan een bepaalde momentopname wordt genomen. De meting met de multimeter is daarom ook niet volledig correct. Uit de tabel blijkt wel dat de meetwaarden slechts zeer weinig van elkaar verschillen, wat aantoont dat de ADC wel een goede meting doet.

5.1.4 Volledige programma

Tijdens het implementeren van een aansturing voor de ADC, werd reeds gebruik gemaakt van het RAM-geheugen en de seriële communicatie met de computer. Om 6 kanalen te kunnen meten en een vlotte samenwerking met het computerprogramma te verkrijgen moet een controlestructuur worden toegevoegd (zie figuur 5.5). Er zijn verschillende versies getest, hieruit is dan de beste versie gekozen.

Versie1: 6 kanalen tijdens 1 beeld

In deze eerste versie wordt slechts 1 maal een beeld weggeschreven. Gedurende deze tijd moeten voor ieder kanaal voldoende metingen gedaan worden. Na iedere meting zal van kanaal veranderd worden om een volgende meting te doen.

De tijd om 1 meting uit te voeren duurt $\approx 5\mu s$. Na iedere meting moet een nieuwe instelling worden gedaan van het controleregister van de ADC. Dit wil zeggen dat opnieuw 2 bytes moeten verzonden worden. Het verzenden van 1 byte duurt $\approx 2,2\mu s$, van 2 bytes dus $\approx 4,4\mu s$.

In het totaal zal een meting dan $\approx 9,4\mu s$ in beslag nemen. De tijd tussen 2 metingen is nagemeten met een teller en bedraagt $11,2\mu s$. Deze waarde ligt iets hoger dan de berekende omdat er geen rekening is gehouden met de uitvoertijd van de tussenliggende instructies.

Omdat er 6 kanalen gemeten worden voor er terug gesprongen wordt naar het eerste kanaal, bedraagt de tijd tussen 2 metingen op eenzelfde kanaal: $6 * 11,2\mu s = 67,2\mu s$. In de inleiding (hoofdstuk 1) is kort geschetst hoe het invoeren van tussenniveau's in de aanstuursignalen, het vermogenverbruik kunnen reduceren. Het is dan ook belangrijk deze tussenniveau's te meten. De duur van een tussenniveau bedraagt $\approx 6\mu s$. Het is duidelijk dat een tussentijd van $67,2\mu s$ hier niet voor geschikt is. De implementatie is dan ook niet nuttig voor deze toepassing.

Versie2: 1 kanaal per beeld, 6 beelden

Er wordt in de vorige versie veel tijd verloren omdat bij elke meting veranderd wordt van kanaal. Omdat tussen iedere meting van een bepaald kanaal, 5 andere metingen van andere kanalen zitten, is het niet mogelijk veel sneller te meten.

Er kan sneller gemeten worden door de ADC in te stellen op één kanaal, om vervolgens enkel nog dat kanaal te meten. De tijd tussen 2 verschillende metingen op een kanaal is nagemeten en bedraagt $5,75\mu s$. Deze tijd is afkomstig van de transmissie van 2 bytes ($\approx 4,2\mu s$), het weg-schrijven naar geheugen ($\approx 0,57\mu s$) en de extra instructies. Deze tijd is net voldoende om ook een meetwaarde te hebben van een tussenniveau in het aanstuursignaal.

Om toch van alle 6 de kanalen een meting te krijgen, moeten 6 beelden worden weggeschreven. Bij ieder beeld zal dan een ander kanaal geselecteerd worden. Wanneer het volledige geheugen gebruikt wordt, is het mogelijk om 10922 (=2AAAH) metingen per kanaal te doen. Om de $5,75\mu s$ wordt een meting gedaan. De totale meettijd per kanaal bedraagt:

$$\frac{10000H * 2\text{geheugenlocaties}}{2\frac{\text{metingen}}{\text{kanaal}} * 6\text{kanalen}} = 10922\frac{\text{metingen}}{\text{kanaal}} \quad (5.4)$$

$$10922\frac{\text{metingen}}{\text{kanaal}} * 5,75\frac{\mu\text{s}}{\text{kanaal}} = 63\frac{\text{ms}}{\text{kanaal}} \quad (5.5)$$

Het is belangrijk dat op een correct moment gestart wordt met meten. Liefst op dezelfde moment dat het beeld wordt weggeschreven, om zoveel mogelijk nuttige informatie te meten. Het circuit dat een beeld zal wegschrijven naar het beeldscherm, wordt aangepast om een signaal te genereren vlak voor het wegschrijven. Dit signaal wordt in de microcontroller gebruikt om een meting te starten.

Versie3: 1 kanaal, 1 beeld

De snelheid waarmee een beeld wordt weggeschreven naar het beeldscherm, kan worden ingesteld. In sommige situaties in een meting van 60ms voldoende om een volledige meting te doen. De snelheid kan worden verlaagd om de tussenniveaus beter te kunnen zien. Na 60ms is het beeld nog niet volledig weggeschreven, waardoor niet alle pulsen gemeten worden. In dit geval moet langer gemeten worden.

Er kan langer gemeten worden door slechts 1 kanaal te meten. De ADC wordt slechts één maal ingesteld op een bepaald kanaal. Nadien worden $6 * 10922 = 65532$ metingen verricht, zodat het RAM-geheugen vol staat. Op deze manier kan 6 maal zo lang gemeten worden: 360ms. Deze lange meettijd is zeker voldoende, ook wanneer de beeldsnelheid wordt verlaagd.

Een nadeel van deze methode is dat telkens maar één kanaal kan gemeten worden. In het uiteindelijke programma worden versie 2 en versie 3 samen geïmplementeerd.

Uiteindelijke programma

Het programma wordt gestart en er wordt gewacht op de computer om een startsignaal te geven. De computer stuurt welk kanaal gemeten moet worden. Als alle kanalen gemeten moeten worden stuurt de computer FFH door, anders zal 2XH doorgezonden worden, waarbij X opnieuw het kanaal voorstelt. Afhankelijk van de gekozen situatie wordt een ander programma opgestart om metingen te verrichten.

Wanneer FFH is doorgezonden zal versie 2 worden uitgevoerd. De microcontroller stelt de ADC in op het eerste kanaal en wacht op een startsein van het beeldscherm. Er worden 10922 metingen uitgevoerd op dit kanaal waarna de microcontroller stopt met meten. Na de meting zal

de microcontroller het volgende kanaal instellen in de ADC en opnieuw wachten op een startsein van het beelscherm. De metingen zijn afgelopen wanneer alle 6 de kanalen geselecteerd zijn.

Wanneer 2XH is doorgezonden zal versie 3 van het programma worden uitgevoerd. De microcontroller stelt de ADC in op het kanaal dat is geselecteerd door de computer. De microcontroller wacht op een startsein van het beelscherm, waarna 65532 metingen worden uitgevoerd. Omdat maar één kanaal wordt gemeten, betekent dit ook het einde van de metingen.

Na het meten moeten al deze waarden in een correcte vorm gezet worden door de bits te spiegelen. Het volledige geheugen wordt afgelopen en voor iedere byte wordt het subprogramma 'SPIEGEL' opgeroepen. Zodra alle waarden in de correcte vorm staan, zal de microcontroller een sein geven aan de computer dat deze klaar is om te verzenden. Alle waarden worden doorgezonden naar de computer voor verdere verwerking.

Een flowchart van dit programma is zichtbaar in figuur 5.6

5.2 Pc-programma

Zoals reeds vermeld in de inleiding van dit hoofdstuk, wordt *c#* gebruikt als programmeertaal. De syntax komt zeer goed overeen met deze van de gekende programmeertaal *c/c++*. Het grote voordeel van *c#* is dat moeilijke grafische vormen, zoals grafieken, eenvoudig kunnen geïmplementeerd worden. Ook het gebruik van de seriële poort is aanzienlijk makkelijker dan in *c/c++*. Omdat dit 2 belangrijke onderdelen vormen van de applicatie, is de keuze voor de hand liggend.

De ontwikkelomgeving waarin gewerkt wordt is de Microsoft Visual *c#* 2005, Express Edition. Dit is een gratis versie van Visual Studio.Net, welke gratis downloadbaar is.

5.2.1 Seriële communicatie

De seriële poort op de computer wordt gebruikt om te communiceren met de microcontroller. Om de seriële poort te gebruiken moet een object aangemaakt worden die deze seriële poort controleert. Vervolgens moet dit object correct ingesteld worden. De correcte COM-poort, dit geeft weer welke seriële poort de computer gebruikt, moet ingesteld worden samen met de baud-rate en de grootte van de ontvangst- en zend-buffer.

Dit alles moet instelbaar zijn door de gebruiker. Het is niet handig om het programma te hercompileren, telkens een andere COM-poort gekozen wordt. In figuur 5.3 is een schermafbeelding zichtbaar van de applicatie waar de seriële communicatie ingesteld kan worden. De

baud-rate kan ingesteld worden op 14400, 56000 en 115200 baud. Als blijkt dat op een bepaalde computer de standaard 115200 baud niet goed overeen komt met deze gegenereerd door de microcontroller, kan men steeds terugvallen op een tragere baud-rate. De assemblercode van de microcontroller moet dan wel aangepast worden volgens tabel 5.1.



Figuur 5.3: Instelling seriële poort

De programmacode maakt gebruik van een object met naam 'serialPort1' om de seriële poort te gebruiken. Wanneer op de knop 'Start Meting' wordt ingedrukt zal de seriële poort ingesteld worden. Vervolgens wordt een byte gezonden naar de microcontroller. Deze byte maakt de keuze tussen het meten van alle kanalen en het meten van een opgegeven kanaal. Nadien wacht de applicatie op de microcontroller. Wanneer deze een byte terugzendt, zal de applicatie data inlezen. Het inlezen van de data wordt gedaan in groepen van 8 bytes om het ontvangen te versnellen. Onderstaande code geeft de implementatie in c# weer:

```
this.serialPort1.Open(); // open seriële poort
this.serialPort1.BaseStream.Flush(); // buffers leegmaken
if (this.grafiekkeuze == 255)
    send[0] = System.Convert.ToByte(255);
else
    send[0] = System.Convert.ToByte(32 + this.grafiekkeuze);
this.serialPort1.Write(send,0,1); // start metingen
int temp = this.serialPort1.ReadByte(); //wachten op start van uC
while((l = this.serialPort1.Read(ontvangen, k, 8)) == 8 && k < lengte-8)
    k += 8;
this.serialPort1.Close(); // poort terug sluiten
```

Als alle data is ontvangen, wordt de seriële poort opnieuw gesloten. Zoals eerder vermeld, moeten 2 bytes gecombineerd worden tot een 10 bit-getal. Dit 10 bit-getal stelt een niveau voor, welk nog omgezet wordt naar een spanningsniveau. In figuur 5.3 is te zien dat 'conversie' kan

aangevinkt worden. Indien dit aangevinkt is zullen alle spanningsniveau's omgezet worden tot vermogen uitgedrukt in W via:

$$\frac{V_{\text{gemeten}}}{A * MeetR} = I_{\text{gemeten}} \quad (5.6)$$

$$I_{\text{gemeten}} * V_{\text{kanaal}} = P \quad (5.7)$$

Wanneer het vakje voor conversie niet aangevinkt is, wordt geen omzetting gedaan en is het resultaat de gemeten spanning aan de ADC. Dit is nuttig om de applicatie te calibreren of op fouten te testen.

5.2.2 Grafiek

Het grote voordeel aan *c#* is dat het eenvoudig is ingewikkelde grafische componenten te gebruiken. Een grafiek is geen standaard object in *c#*, maar op het internet zijn vele implementaties beschikbaar. Eén ervan is de bibliotheek 'ChartDirector'. Met behulp van deze bibliotheek kan eenvoudig een grafiek aangemaakt worden.

Onderstaande code maakt een grafiek aan en zet enkele punten op de grafiek. De assen worden gelabeld en de grafiek is aanklikbaar. Het is mogelijk om met de muis op een bepaald punt te staan en de exacte x- en y-waarde op te vragen.

```
this.winChartViewer1 = new ChartDirector.WinChartViewer();
ChartDirector.XYChart chart = new ChartDirector.XYChart(874, 409);
    // grootte van de XY-grafiek
chart.setPlotArea(30, 20, 840, 360);
chart.yAxis().setAutoScale(0.1, 0.2, 1);
chart.addLineLayer(meting_grafiek); // lijst punten
chart.xAxis().setLabels(xas);
chart.xAxis().setLabelStep(1000); // om de 1000 punten een label
chart.addTitle(titel, "Arial Bold Italic", 11, 0xffffffff).setBackground(0
    x000080, -1, 1);
chart.yAxis().setTitle("mW");
chart.xAxis().setTitle("ms");
winChartViewer1.Image = chart.makeImage();
winChartViewer1.ImageMap = chart.getHTMLImageMap("clickable", "", "title
    = '{xLabel}ms:{value}mW'");
```

Het totale vermogenverbruik wordt berekend als de totale oppervlakte onder de grafiek. Deze wordt berekend als volgt:

$$P_{tot} = \frac{\sum P_{meting} * t}{\text{Totaletijd}} \quad (5.8)$$

5.2.3 Instellingen

Wanneer een aanpassing aan het circuit wordt gedaan, zoals de versterking van de opamps die wordt veranderd, moet dit makkelijk aan te passen zijn in het programma. In figuur 5.4 is een schermafbeelding weergegeven van het formulier om de instellingen aan te passen. Per kanaal is telkens de maximale stroom, de meetweerstand en de versterkingsfactor instelbaar. Voor alle kanalen is globaal de referentiespanning V_{ref} van de ADC instelbaar.



Figuur 5.4: Instelling omzettingen

Als de referentiespanning van de ADC geen 5V bedraagt, moet deze aangepast worden in de applicatie om correcte omzettingen te maken.

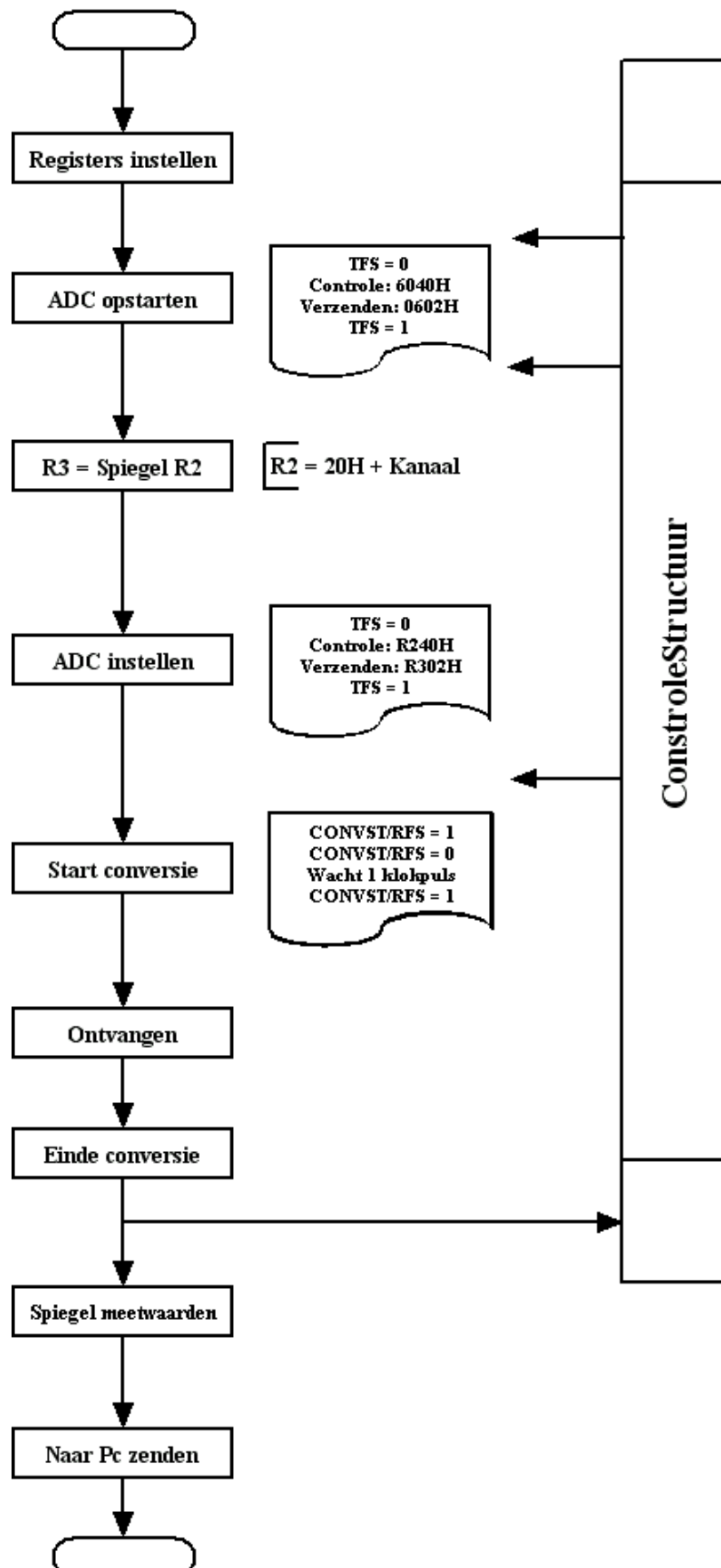
Wanneer de specificaties veranderen of wanneer aanpassingen gedaan worden aan de meetcircuits, moet de applicatie opnieuw gecalibreerd worden. Bij het aanpassen van de maximale stroom, de meetweerstand of de offset, zal het programma een nieuwe ideale versterking berekenen. De echte gemeten versterking kan worden ingevuld, waarna bepaald wordt wat de maximale spanning V_{max} aan de ingang van de ADC bedraagt en wat de minimaal meetbare stroom I_{min} wordt.

$$A_{ideaal} = \frac{V_{ref} - V_{offset}}{MeetR * I_{max}} \quad (5.9)$$

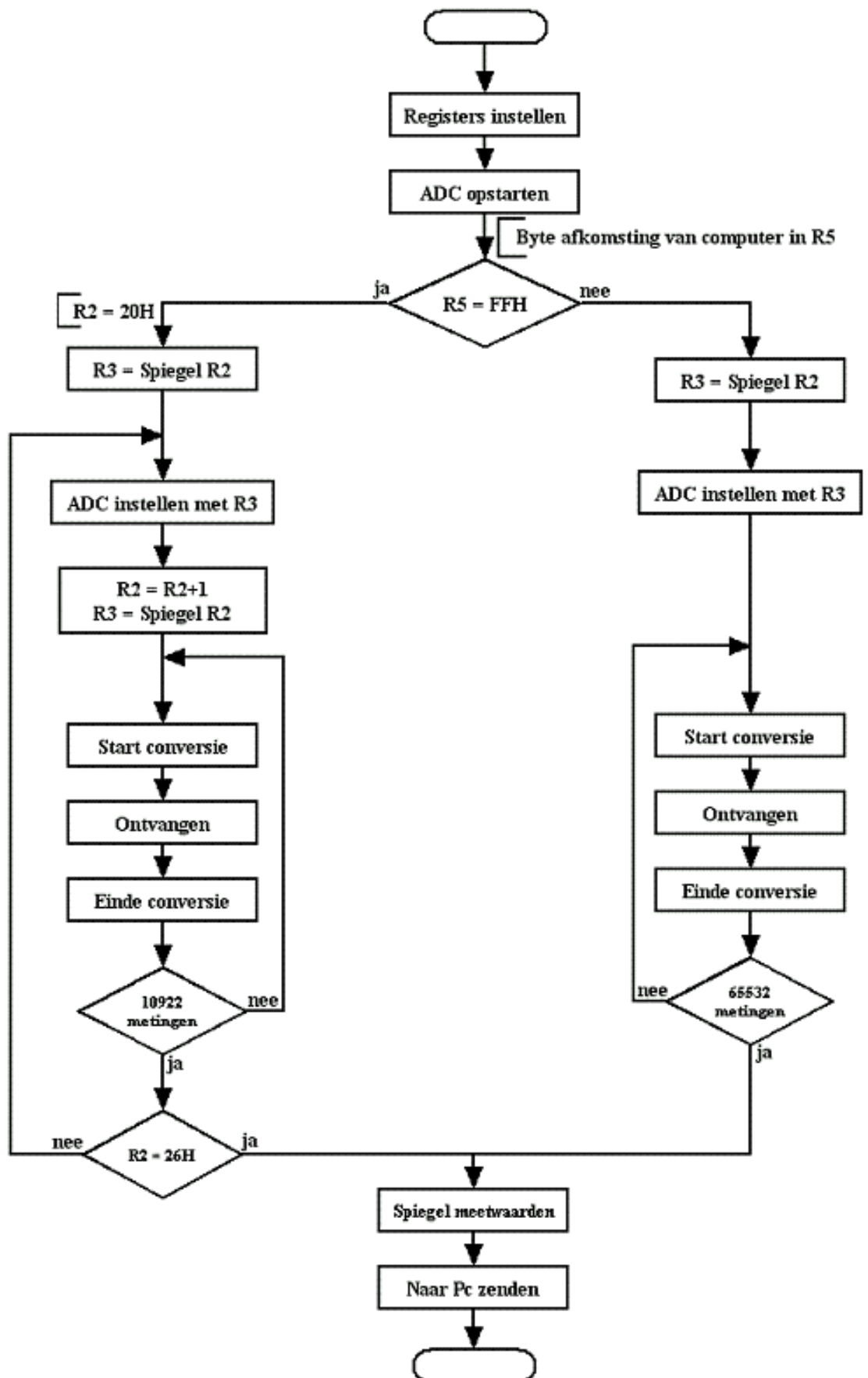
$$V_{max} = V_{offset} + I_{max} * MeetR * A \quad (5.10)$$

$$I_{min} = \frac{\frac{V_{ref}}{1023}}{MeetR * A} \quad (5.11)$$

Het aanpassen van de offset is belangrijk omdat de differentiële versterker een klein DC-niveau doorlaat. Deze offset wordt dan versterkt door de opamp waardoor een DC-spanning van enkele mV aanwezig is in het signaal aan de ADC. Het is niet mogelijk om deze offset volledig weg te regelen met de regelbare weerstanden rond de opamp, waardoor softwarematig een correctie wordt uitgevoerd. De ADC meet zeer nauwkeurig, maar zonder goede calibratie van offset, versterking, meetweerstand en V_{ref} in de applicatie, gaat de nauwkeurigheid verloren.



Figuur 5.5: Flowchart: Werking ADC



Figuur 5.6: Flowchart: Volledig programma

Hoofdstuk 6

Eindresultaat

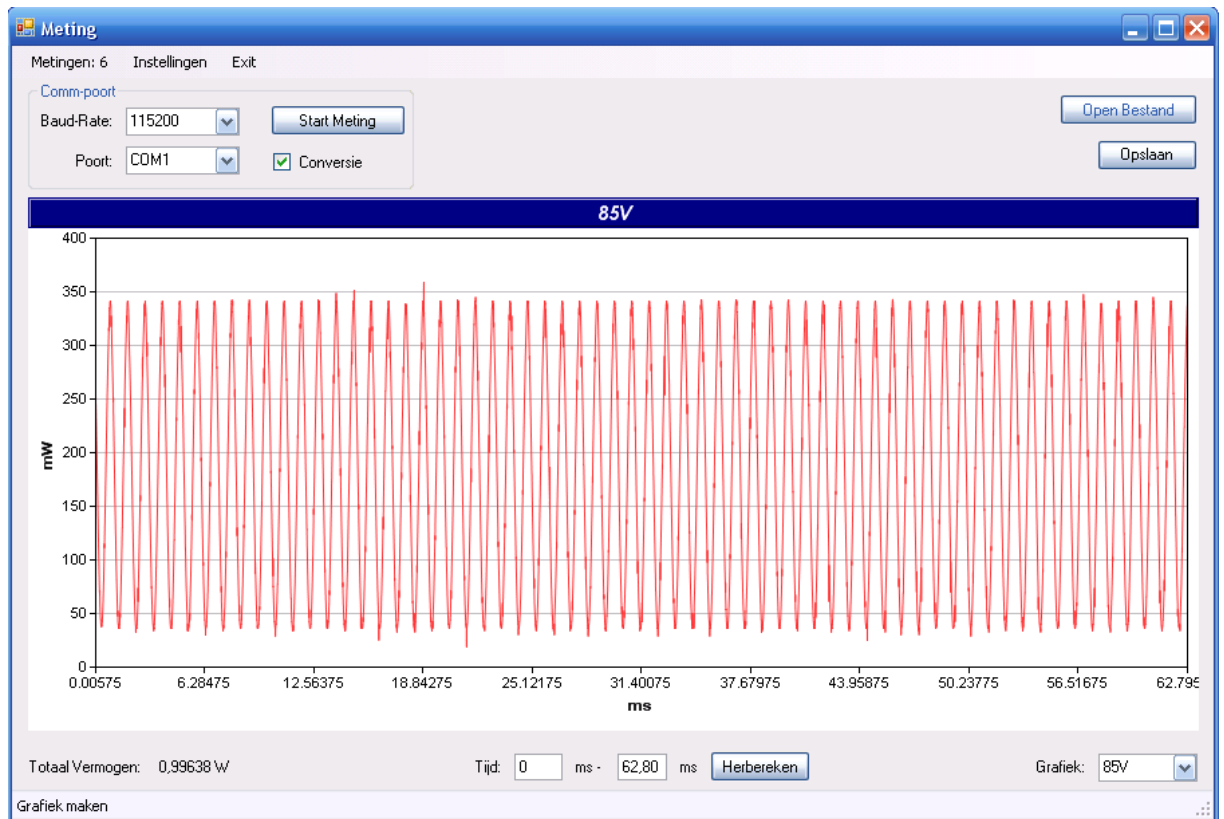
Een afbeelding van het volledige programma is weergegeven in figuur 6.1. In het menu, links bovenaan, kan het aantal te meten kanalen geselecteerd worden. In de afbeelding zijn 6 kanalen gemeten, waarbij het af te beelden kanaal rechts onderaan kan geselecteerd worden. Als het aantal kanalen op 1 gezet wordt, kan eveneens rechts onderaan een kanaal geselecteerd worden om te meten. Door in het menu op 'Instellingen' te klikken, zal het instellingenvenster uit figuur 5.4 worden opgeroepen.

Het programma geeft enkel een weergave van de grafiek en berekent het vermogenverbruik. Het is ook mogelijk om in te zoomen op een bepaalde tijd door een begintijd en een eindtijd op te geven. Voor verdere bewerkingen of om de meetwaarden te bewaren, kunnen de metingen opgeslagen worden als een excel file. Dit bestand kan later opnieuw geopend worden.

6.1 Testen

In hoofdstuk 3 zijn al enkele metingen gedaan met de volledige opstelling. Toen zijn enkele eenvoudige zaken zoals een offset en een sinus gemeten. Nu zal een meting gedaan worden van de verschillende signalen tijdens het wegschrijven van een beeld.

In figuur 6.4 zijn de verschillende resetpulsen zichtbaar die aanwezig zijn op een rij. Deze golfvorm is gemeten met de oscilloscoop. Let goed op de timing van de 60V resetpulsen. In figuur 6.5 is het gemeten vermogen weergegeven van kanaal 60V. Telkens er een resetpuls is, zijn duidelijk twee stroom-/vermogenpieken zichtbaar. De schijnbare ruis vanaf 10ms komt overeen met stroompiekjes afkomstig van het wegschrijven van een beeld. In figuur 6.6 is een detail genomen van 1 stroompiek tijdens een resetpuls. Figuur 6.7 geeft een detailopname van de



Figuur 6.1: Screenshot: Applicatie

pieken na 10ms.

Op de andere kanalen zijn eveneens metingen gedaan. De resultaten zijn vergelijkbaar met de stroompieken na 10ms op het 60V kanaal.

6.2 Verbeteringen

De tussentijd tussen 2 metingen bedraagt momenteel $5,75\mu\text{s}$. Dit is wel snel genoeg om voldoende metingen te maken, maar wanneer men meer metingen wenst bij de tussenniveaus is $5,75\mu\text{s}$ niet voldoende. Bij het meten is ontdekt dat ook negatieve stromen kunnen optreden, dit is met de huidige opstelling niet mogelijk. Ook hiervoor bestaat nog een verbetering.

6.2.1 Sneller meten

Het is mogelijk om de metingen nog sneller te maken door de tijd van een transmissie te verminderen. De ADC kan correct werken tot een klokfrequentie van 30MHz . Momenteel bedraagt deze klokfrequentie slechts $\frac{32\text{MHz}}{4} = 8\text{MHz}$. Het is niet mogelijk om deze kloksnelheid te verhogen

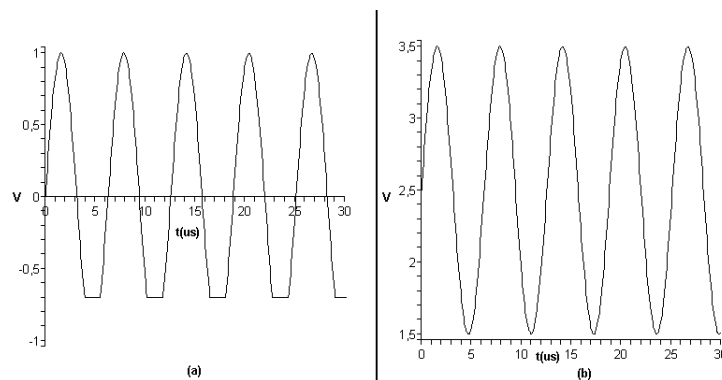
door gebruik te maken van de seriële poort. De seriële communicatie kan ook tot stand gebracht worden door zelf, met behulp van gewone input- en output-pinnen, een klok te genereren en data in te lezen. Bij een klokfrequentie van 30MHz zou de transmissie nog maar $\approx 0,5\mu\text{s}$ in beslag nemen. Zodra een conversie klaar is, wordt het mogelijk om opnieuw een conversie te starten ipv. te wachten tot de transmissie voltooid is.

Om nog sneller te meten kunnen verschillende borden, met dezelfde opstelling of snellere componenten, gebruikt worden. Elk meetbord start dan iets later met meten (vb. $1\mu\text{s}$). Dit is wel een duurdere oplossing, waarbij de verschillende borden onderling goed op elkaar moeten afgesteld zijn.

6.2.2 Negatieve stromen

Het meten van negatieve stromen is momenteel niet mogelijk. De reden hiertoe is dat de ADC geen negatieve spanningen aankan. In hoofdstuk 3 zijn diodes toegevoegd om de spanning steeds tussen $-0,7\text{V}$ en $5,7\text{V}$ te houden ter bescherming van de ADC. Negatieve spanningen kunnen toch gemeten worden als een offset wordt toegevoegd aan de spanning, net voor de ADC.

In figuur 6.2(a) is een sinus zichtbaar zonder offset. Zodra de sinus $-0,7\text{V}$ bereikt zullen de diodes verhinderen dat de spanning nog zal verlagen. In figuur 6.2(b) is dezelfde sinus zichtbaar, met een offset van $2,5\text{V}$ (de helft van V_{ref}). De sinus kan nu wel volledig gemeten worden.



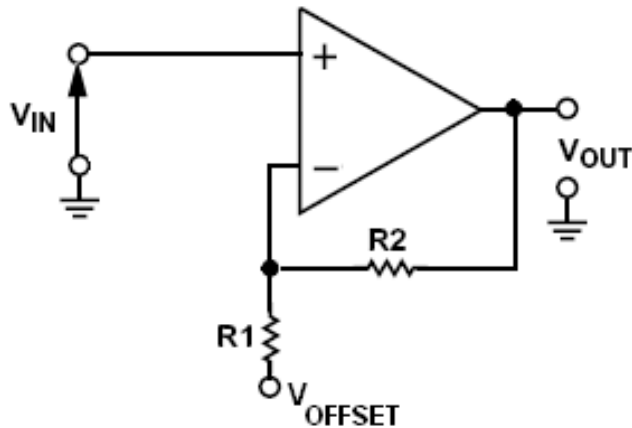
Figuur 6.2: (a) Sinus zonder offset, afgevlakt op $-0,7\text{V}$ (b) Sinus met offset $2,5\text{V}$

De eenvoudigste manier om een offset toe te voegen aan de te meten spanning is door gebruik te maken van een opamp. In figuur 6.3 is opnieuw een niet-inverterende versterker zichtbaar. Deze is analoog aan figuur 3.1(b), waarbij de negatieve klem nu verbonden wordt met V_{offset}

ipv. de massa. Via superpositie krijgt men 2 bijdragen:

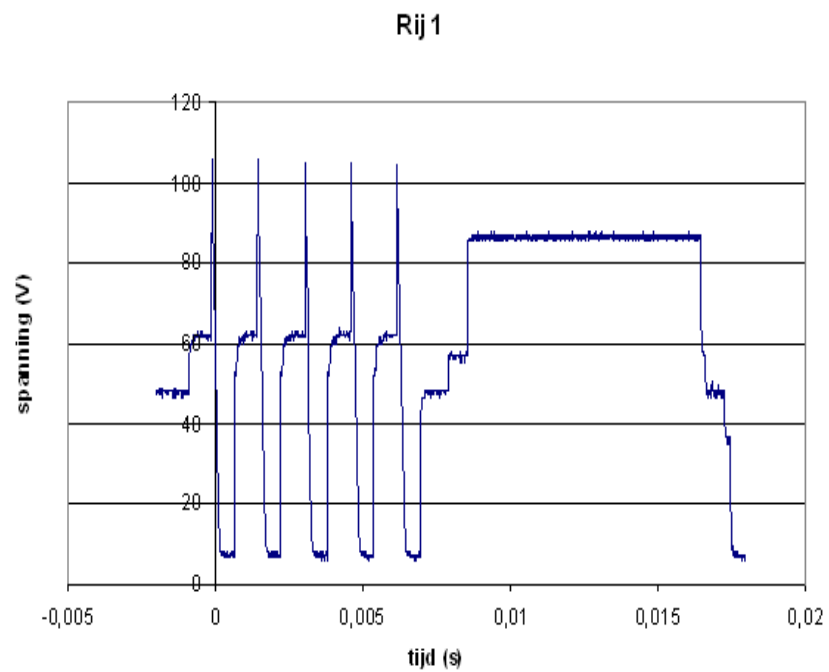
$$V_{uit} = \begin{cases} V_{in} \frac{R_1+R_2}{R_1} & \text{Bijdrage } V_{in}; \\ -V_{offset} \frac{R_2}{R_1} & \text{Bijdrage } V_{offset}. \end{cases} \quad (6.1)$$

Samen krijgt men: $V_{uit} = V_{in} \frac{R_1+R_2}{R_1} - V_{offset} \frac{R_2}{R_1}$. Omdat dit dezelfde opstelling is als de reeds gebruikte opamps voor extra versterking, kunnen deze opamps hergebruikt worden. Omdat voor ieder kanaal deze opamps andere waarden voor R_1 en R_2 gebruiken, moet voor iedere opamp een andere V_{offset} gebruikt worden. Het is makkelijker om voor ieder kanaal een extra opamp toe te voegen, met dezelfde R_1 en R_2 .

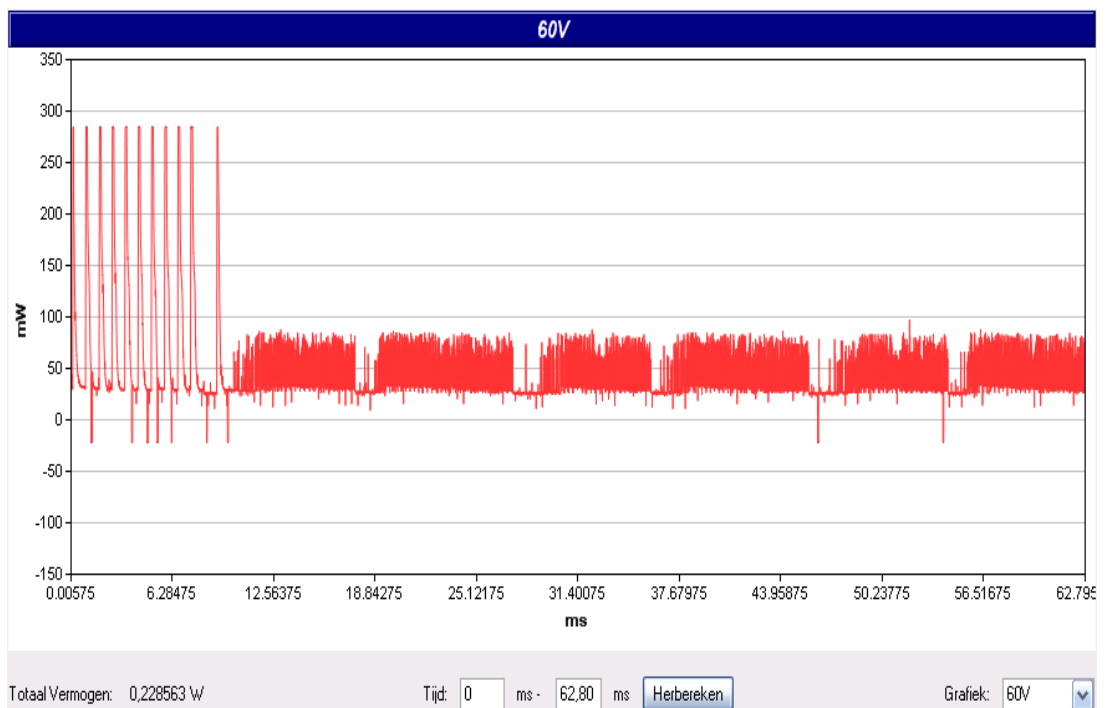


Figuur 6.3: Niet inverterende versterker met offset

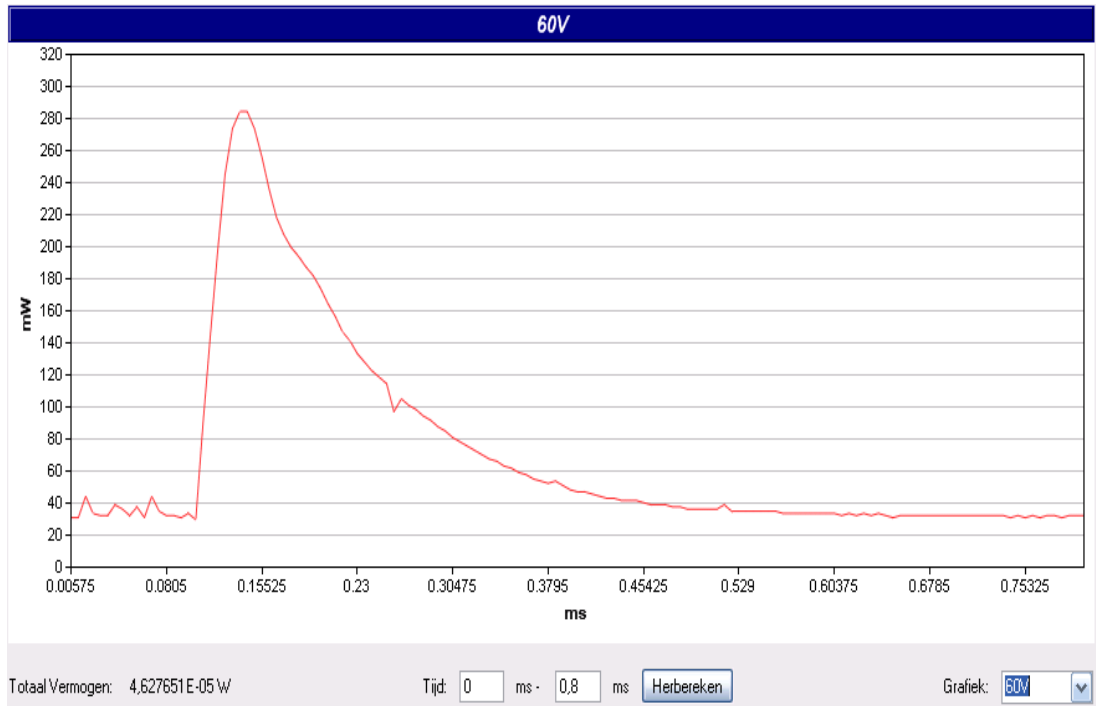
Het meetcircuit voegt nu een extra versterking en een offset toe aan het signaal voor de ADC. De pc-applicatie biedt de mogelijkheid om in het instellingenvenster (figuur 5.4) de totale versterking, samen met de offset, aan te passen. Zodra een nieuwe offset is ingevoerd in dit venster, zal de applicatie een nieuwe versterking voorstellen, zodat I_{max} meetbaar blijft. Deze versterking wordt gerealiseerd door de feedback van de opamps aan te passen. Door het aanpassen van de offset worden ook negatieve meetwaarden weergegeven.



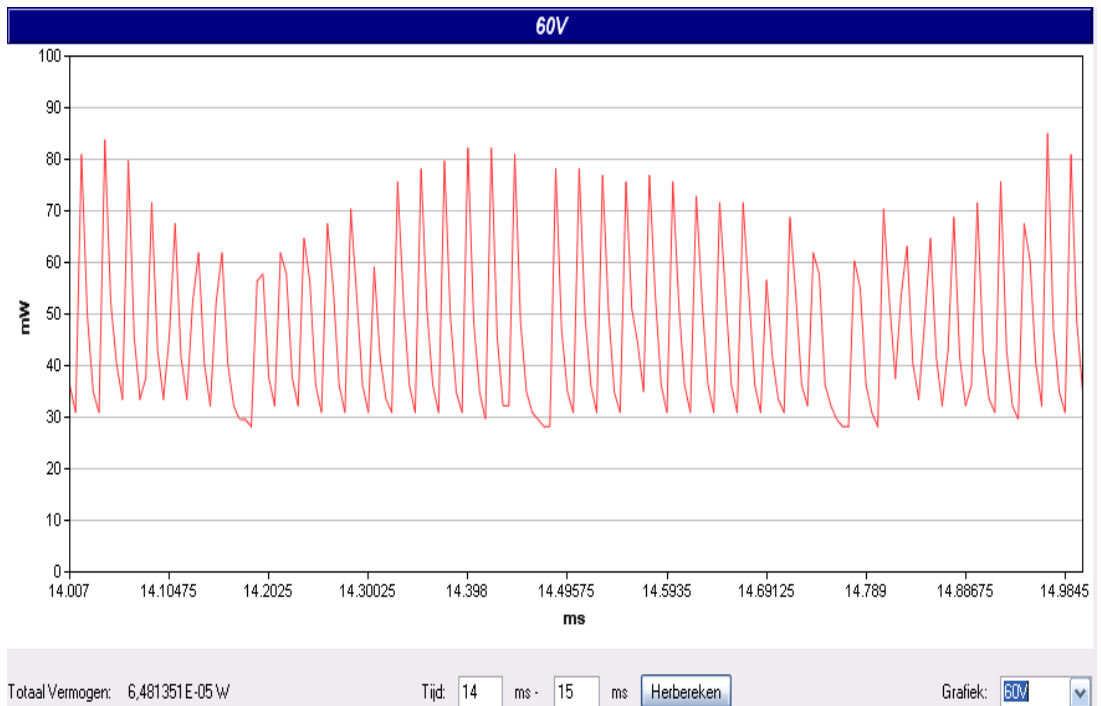
Figuur 6.4: Meting: Resetpulsen en beeldinformatie op een rij



Figuur 6.5: Meting: Stroommeting op 60V



Figuur 6.6: Meting: Detail stroommeting op 60V: 0 tot 0,8ms



Figuur 6.7: Meting: Detail stroommeting op 60V: 12 tot 13ms

Hoofdstuk 7

Besluit

Het opzet van deze thesis was om het vermogenverbruik van bistabiele beeldschermen automatisch op te meten. De complexe golfvormen, nodig om een beeld weg te schrijven, kunnen met behulp van verschillende technieken geoptimaliseerd worden om het vermogenverbruik te reduceren. Het gerealiseerde meetcircuit kan ingezet worden om deze optimalisaties snel te evalueren. Om het vermogenverbruik te kunnen opmeten worden meetweerstand in de verschillende voedingslijnen aangebracht.

De analoge hardware is in staat om een zeer kleine spanningsval over de meetweerstand te meten, zelfs wanneer de common-mode spanning 85V bedraagt. De differentiële versterker laat nog een kleine extra DC-offset spanning door en voert ruis in. Na versterking wordt een signaal tussen -0,7V en 5,7V aangeboden aan de ADC.

De digitale hardware maakt om de $5,75\mu\text{s}$ samples van de meetwaarden. Er is keuze om slechts 1 kanaal te meten, of alle 6 de kanalen. Het meten van 1 kanaal maakt het mogelijk gedurende een langere periode metingen te verrichten. Het beeld-circuit geeft een startsignaal aan het meet-circuit telkens een beeld weggeschreven wordt, waarna het meetcircuit metingen uitvoert. De metingen worden met een grote nauwkeurigheid verricht (tot $\approx 5\mu\text{A}$).

De computerapplicatie kan, via een seriële link met het meet-circuit, de meetwaarden opvragen en verder verwerken. Het is belangrijk het programma goed te calibreren om correcte omzettingen te behouden. Bij een fout op de offset, V_{ref} , de meetweerstand of de versterking zal er een systematische fout aanwezig zijn in alle meetwaarden. Het programma biedt de mogelijkheid de verschillende offsets en versterkingen aan te passen.

Als uiteindelijke test zijn metingen verricht tijdens het wegschrijven van een beeld. De bekomen resultaten voldoen aan de verwachtingen.

Bijlage A

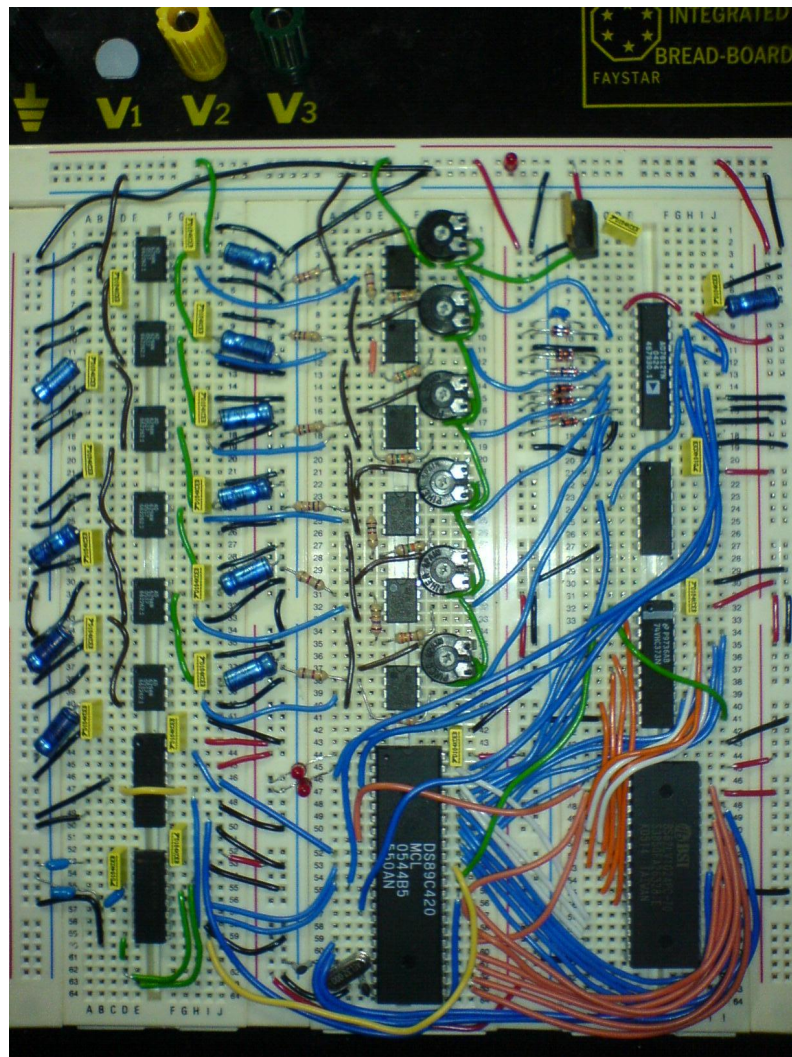
Bestelling

Volgende componenten zijn besteld bij Farnell.com:

Beschrijving	Ordernummer	Stuks	Prijs (€)	Totaal (€)
Differentiële versterker: AD629AN	388-3042	6	7,52	45,12
Opamp: OP07C	690-624	6	1,95	11,7
10 bit ADC: AD7812	283-710	1	16,24	16,24
128kB RAM (70ns): BS62LV1024PC-70	413-6184	1	5,80	5,80
RS232-seriële communicatie: Max3232CPE	305-0269	1	4,12	4,12
Invertor: MM74HC04N	378-252	1	0,30	0,30
3 state Buffer: MM74HC125N	378-458	1	0,47	0,47
8 bit Latch: 74VHC373N	684-119	1	0,66	0,66
32Mhz-Xtal: HC49/4H	569-884	1	4,02	4,02
Leds HE RED 5Vdc	885-976	10	0,37	3,70
			Totaal:	92,13

Bijlage B

Foto meetcircuit



Figuur B.1: Foto meetcircuit

Bijlage C

Programma-code: microcontroller

```
; Volledige programma
; Meting: Aparte meting per kanaal, dubbel geheugengebruik
; Keuze uit 1 kanaal meten of 6 kanalen

; Overzicht:
; - Seriele communicatie op 115200 baud
; - Schrijven naar ram geheugen, geen stretch value nodig
; - ADC-aansturing
; - Bitspiegeling van aansturing tijdens conversie, ontvangen waarden
;   achteraf spiegelen en correct vormen
; - Stopt als geheugen volledig is volgeschreven

; Gebruikte Registers: BANK0
; R0 = wachten
; R1 = Tellen
; R2 = ADC-instelling in originele versie
; R3 = spiegeldoorgeefadres
; R4 = spiegelhulpadres
; R5&R6 = data-adres
; R7 = Eerste conversie
;
; BANK1
; R2 & R3: Tellers

; Gebruikte Pinnen:
```

```

; Poort 0 en Poort 2
; P1.2 en P1.3 = communicatie ADC
; P1.5 en P1.6 = toestand (nog maken)
; P1.7: bank geheugen
; P3.2 = TFS
; P3.4 = CONVST & RFS
; P3.3 = Start: eerst hoog, dan laag
; P1.1 en P1.2 = lampje

; Uitleg ADC:
;   TFS = 0 => D_IN dus intern register updaten = schrijven
;   RFS = 1 => D_OUT dus data ontvangen vanaf klok = lezen
;   CONVST = 0 => Conversie starten en voor 2,3us terug hoogbrengen.
;   Nadien uitlezen door RFS hoog te brengen
;   Na conversie 100ns wachten
;   seriele communicatie via seriele comm poort 1, mode 0
;*****

;CPU      "8051.TBL"
;HOF      "INT8"

P1        EQU    90H
P1_1      EQU    90H
P1_2      EQU    91H
P3        EQU    B0H
SCON0     EQU    98H
PCON      EQU    87H
T2CON     EQU    C8H
SBUF0     EQU    99H
TCON      EQU    88H
TMOD      EQU    89H
TH1       EQU    8DH
TL1       EQU    8BH
CKCON     EQU    8EH
IE        EQU    A8H
PCON      EQU    87H
CKMOD     EQU    96H

```



```

    ANL    SCON1,#EFH            ; Receive enable af
    MOV    A,SCON1
    ANL    A,#03H              ; enkel checken op laatste 2 bit
    CJNE   A,#01H,SEND1        ;
RECEIVE1: MOV    R5,SBUF1
SEND1:    ANL    SCON1,#FCH      ; TI&RI bit resetten
    INC    R0
    RETI
;*****
RESET:
    MOV    P1,#FEH            ; Lamp 1 brandt
    ; Initialisatie van de registers
    MOV    SCON0,#40H         ; Comm 0 in mode 1 '10000000'
    MOV    PCON,#80H          ; PCON: '10000000'
    MOV    CKCON,#28H         ; Timer1: T1M = 0 ; '00101000'
    MOV    CKMOD,#10H         ; Timer1: T1HM = 0 ; '00000000'
    MOV    TMOD,#20H          ; Timer 1 in mode 2: '00100000'
    MOV    TH1,#EEH           ; TH1 = EEH: 115200 baud
    MOV    TL1,#00H
    MOV    ACON,#00H          ; '00000000' Non-Page mode
    ANL    P1,#7FH            ; Pin P1.7 af, msb RAM = 0
    MOV    DPS,#14H           ; '00010100' auto DPTR incr
    MOV    DPL,#00H
    MOV    DPH,#00H           ; Startadres in het ramgeheugen
    ; timer ingesteld, geheugentoeegang ok, seriele comm ok
;*****
; ADC controle:
;*****
    ; initialisatie van alle registers,... voor de ADC toegang
    MOV    SCON1,#20H         ; Seriele comm: mode 0
    ORL    IE,#C0H            ; '11xxxxx' interrupt seriële 1
    ORL    P1,#0CH            ;
    ORL    P3,#10H            ; -> CONVST op 1 => geen conversie
    ORL    TMOD,#02H          ; 'xxxxxxx1x' => Mode 2 voor timer0
    MOV    R2,#20H            ; kanaal 0, mode 1
START:

```

```

    ORL    IE,#10H                ; Wachten op start van de computer
    MOV    R0,#00H
    MOV    TCON,#40H             ; Timer 1 starten: '01000000'
    ORL    SCON0,#10H           ; Receive enable aan
WACHT_SERIAL0: CJNE    R0,#01H,WACHT_SERIAL0
    ANL    IE,#EFH                ; startsignaal gegeven
    ANL    SCON0,#EFH
    ANL    P1,#FDH                ; lamp2 aan

                                ; Power up ADC... => 6040H wegschrijven naar ADC
ADC_POWER: ANL    P3,#FBH        ; TFS = 0 -> Wegschrijven
    MOV    R0,#00H                ;
    MOV    SBUF1,#06H            ; 0602H zenden
WACHT_ADC: CJNE    R0,#01H,WACHT_ADC    ; Wachten tot data verzonden
    MOV    SBUF1,#02H            ; laatste byte
WACHT_ADC1: CJNE    R0,#02H,WACHT_ADC1   ; Wachten tot data verzonden
    ORL    P3,#04H                ; TFS = 1

                                ; ADC opgestart
                                ; keuze van de computer
    CJNE    R5,#FFH,TUSSEN_KANALEN_1 ; 1 kanaal meten geselecteerd

;*****
; 6 kanalen meten
;*****
    MOV    A,R2
    MOV    R3,A
    LCALL  SPIEGEL                ; Duur: 2us
METING_START:
    ORL    PSW,#08H                ; BANK1 kiezen
    MOV    R2,#00H
    MOV    R3,#00H
    ANL    PSW,#F7H                ; BANK0 kiezen
    MOV    R7,#00H                ; eerste conversie

    ANL    P3,#FBH                ; TFS = 0 -> Wegschrijven
    MOV    R0,#00H                ;

```

```

        MOV     SBUF1,R3                ; XX02H zenden
WACHT_ADC2: CJNE     R0,#01H,WACHT_ADC2    ; Wachten tot verzonden
        MOV     SBUF1,#02H            ; laatste byte
WACHT_ADC3: CJNE     R0,#02H,WACHT_ADC3    ; Wachten tot verzonden
        ORL     P3,#04H                ; TFS = 1
        ; Volgende kanaal berekenen
        INC     R2
        MOV     A,R2
        MOV     R3,A
        LCALL   SPIEGEL

        ANL     P1,#FEH                ; lamp1 aan
WACHT_EXTERNE1: MOV     A,P3            ; startsignaal hoog
        ANL     A,#08H
        CJNE     A,#08H,WACHT_EXTERNE1
WACHT_EXTERNE2: MOV     A,P3            ; startsignaal laag
        ANL     A,#08H
        CJNE     A,#00H,WACHT_EXTERNE2
        ORL     P1,#01H                ; lamp1 uit
        LJMP    INSTELLING_OK

TUSSEN_KANALEN_1: LJMP    KANALEN_1
        ; Adc ingesteld op correcte kanaal
INSTELLING_OK: ANL     P3,#EFH          ; CONVST = 0
        MOV     R0,#00H
        ORL     P3,#10H                ; CONVST = 1 ; puls lang genoeg

        MOV     R0,#00H                ; Data onmiddelijk ontvangen
        ORL     SCON1,#10H             ; '00110000': Recieve enable
WACHT_SERIAL1_2: CJNE     R0,#01H,WACHT_SERIAL1_2
        MOV     A,R5
        MOV     R6,A
        ORL     SCON1,#10H            ; Receive enable
WACHT_SERIAL1_3: CJNE     R0,#02H,WACHT_SERIAL1_3 ;

        MOV     A,R6                    ; Data wegschrijven: duur 0,57us
        MOVX    @DPTR,A

```

```

MOV      A,R5
MOVX    @DPTR,A
CJNE    R7,#00H,EERSTE_CONV_1
DEC     DPL                                ; Eerste conversie niet opslaan
DEC     DPL                                ; Duur totaal: 2,2us*2 transmissie
                                           ; + 0,57us RAM-access = 4,97us
EERSTE_CONV_1: CJNE    R7,#01H,METING_EINDE    ; Eerste meting
ORL     PSW,#08H                            ; BANK1 kiezen
INC     R2
CJNE    R2,#00H,INC_R1

INC     R3                                ; R1 1 verhogen, R1,R0 = 2AAA?
INC_R1: CJNE    R3,#2AH,METING_EINDE
CJNE    R2,#AAH,METING_EINDE
LJMP    METING_EINDE_2

TUSSENSPRONG: LJMP    METING_START
METING_EINDE: ANL     PSW,#F7H                ; BANK0 kiezen
MOV     R7,#01H
LJMP    INSTELLING_OK

METING_EINDE_2: ANL     PSW,#F7H                ; BANK0
CJNE    R2,#23H,BANK1_GEHEUGEN
MOV     DPL,#00H                            ; Volgende bank in RAM
MOV     DPH,#00H
ORL     P1,#80H                              ; RAM: Bank1

BANK1_GEHEUGEN: CJNE    R2,#26H,TUSSENSPRONG    ; 6x uitvoeren
ANL     P1,#FEH                              ; Lamp 1 aan;
LJMP    DATA_OK

;*****
; 1 kanaal meten
;*****
KANALEN_1:
MOV     A,R5
MOV     R3,A

```

```

        LCALL    SPIEGEL                ; Duur: 2us
METING_START_1: ORL        PSW,#08H    ; BANK1 kiezen
        MOV     R2,#00H
        MOV     R3,#00H
        ANL    PSW,#F7H                ; BANK0 kiezen
        MOV     R7,#00H                ; eerste conversie

        ANL    P3,#FBH                ; TFS = 0 -> Wegschrijven
        MOV     R0,#00H
        MOV     SBUF1,R3              ; XX02H zenden
WACHT_ADC2_1: CJNE     R0,#01H,WACHT_ADC2_1    ; Wachten tot verzonden
        MOV     SBUF1,#02H            ; laatste byte
WACHT_ADC3_1: CJNE     R0,#02H,WACHT_ADC3_1    ; Wachten tot verzonden
        ORL    P3,#04H                ; TFS = 1
        ANL    P1,#FEH                ; lamp1 aan
WACHT_EXTERNE1_1: MOV     A,P3        ; startsignaal hoog
        ANL    A,#08H
        CJNE   A,#08H,WACHT_EXTERNE1_1
WACHT_EXTERNE2_1: MOV     A,P3        ; startsignaal laag
        ANL    A,#08H
        CJNE   A,#00H,WACHT_EXTERNE2_1
        ORL    P1,#01H                ; lamp1 uit

        ; Adc ingesteld op correcte kanaal
        MOV     R3,#00H
INSTELLING_OK_1: ANL     P3,#EFH      ; CONVST = 0
        MOV     R0,#00H
        ORL    P3,#10H                ; CONVST = 1 ; puls lang genoeg

        MOV     R0,#00H                ; Data onmiddelijk ontvangen
        ORL    SCON1,#10H            ; '00110000': Recieve enable
WACHT_SERIAL1_2_1: CJNE   R0,#01H,WACHT_SERIAL1_2_1
        MOV     A,R5
        MOV     R6,A
        ORL    SCON1,#10H            ; Receive enable aan
WACHT_SERIAL1_3_1: CJNE   R0,#02H,WACHT_SERIAL1_3_1 ;

```

```

MOV      A,R6                ; Data wegschrijven
MOVX     @DPTR,A
MOV      A,R5
MOVX     @DPTR,A            ; Data wegschrijven in het geheugen
CJNE     R7,#00H,EERSTE_CONV_1_1
DEC      DPL                ; Eerste conversie
DEC      DPL                ; Duur totaal: 2,2us*2 transmissie
                                ; + 0,57us RAM-access = 4,97us
EERSTE_CONV_1_1: CJNE     R7,#01H,METING_EINDE_1    ; Eerste meting
INC_R1_1: MOV      R0,DPH
          CJNE     R0,#FFH,INSTELLING_OK_1 ; Datapointer = FFFC?
          MOV      R0,DPL
          CJNE     R0,#FCH,INSTELLING_OK_1
          MOV      DPL,#00H                ; bank 1 in ram
          MOV      DPH,#00H
          ORL      P1,#80H                ; bank 1 van ram
          INC      R3
          CJNE     R3,#02H,INSTELLING_OK_1
          LJMP     METING_EINDE_2_1
METING_EINDE_1:
          MOV      R7,#01H
          LJMP     INSTELLING_OK_1
METING_EINDE_2_1:
          ANL      P1,#7FH                ; bank0
          ANL      P1,#FEH                ; Lamp 1 aan;

                                ; Data gemeten, goedgezetten via spiegel
DATA_OK: ANL      P1,#7FH                ; bank0 ram
TRANSMIT_START: MOV      IE,#90H
          MOV      DPL,#00H                ; Adrespointer resetten
          MOV      DPH,#00H
          MOV      R2,#00H
DATA_AANPASSEN: MOVX     A,@DPTR
          MOV      R3,A
          LCALL    SPIEGEL
          MOV      A,R3
          MOV      R6,A

```

```
MOVX    A , @DPTR
MOV     A , R3
MOV     R5 , A                ; Bits Gespiegeld , nu samenzetten
RL      A                    ; LSB: rotatie over 2 naar links
RL      A                    ; en alle hoogste bits = 0
ANL     A , #03H
MOV     R5 , A                ; LSB ok
MOV     A , R6                ; MSB: rotatie over 2 naar links
RL      A                    ; 6 hoogste bits or met lsb
RL      A                    ; en 2 laagste opslaan
MOV     R6 , A
ANL     A , #FCH
ORL     A , R5
MOV     R5 , A                ; LSB volledig ok
MOV     A , R6
ANL     A , #03H
MOV     R6 , A                ; MSB volledig ok
ORL     DPS , #40H           ; geheugenpointer resetten
INC     DPTR
INC     DPTR
ANL     DPS , #BFH           ; DPS goedzetten
MOVX    @DPTR , A            ; MSB opslaan
MOV     A , R5
MOVX    @DPTR , A            ; LSB opslaan
MOV     A , DPL
CJNE   A , #00H , DATA_AANPASSEN ; blijven data aanpassen
MOV     A , DPH
CJNE   A , #00H , DATA_AANPASSEN
INC     R2                    ; Datapointer reset
MOV     DPL , #00H
MOV     DPH , #00H
ORL     P1 , #80H            ; bank1_ramgeheugen
CJNE   R2 , #02H , DATA_AANPASSEN
ANL     P1 , #7FH            ; bank0 ramgeheugen
; Data aangepast => zenden
ANL     P1 , #FDH            ; Lamp 2 aan;
MOV     DPL , #00H
```

```

MOV     DPH,#00H
MOV     TCON,#40H           ; Timer 1 starten: '01000000'
MOV     IE,#90H           ; Interrupt voor seriele comm 0

MOV     R0,#00H
MOV     A,#FFH
MOV     SBUF0,A

WACHT_SERIAL0_0: CJNE     R0,#01H,WACHT_SERIAL0_0
                ; startsignaal gegeven

ORL     P1,#02H           ; Lamp 2 uit;
MOV     R5,#00H

SENDING: MOV     R0,#00H
MOVX    A,@DPTR           ; Data terugsturen naar de pc
MOV     SBUF0,A

WACHT_SERIAL0_1: CJNE     R0,#01H,WACHT_SERIAL0_1
INC_R1_2: MOV     R0,DPL
CJNE    R0,#FCH,SENDING   ; Datapointer = FFFC?
MOV     A,P1             ; Lamp 2 flikkeren
XRL     A,#02H
MOV     P1,A
MOV     R0,DPH
CJNE    R0,#FFH,SENDING
INC     R5
MOV     DPL,#00H
MOV     DPH,#00H
ORL     P1,#80H           ; bank 1 geheugen
CJNE    R5,#02H,SENDING
ORL     P1,#02H           ; lamp2 uit
LJMP    RESET

;*****
; Bits spiegelen:
; a) rotate 1 right and #80H
; b) rotate 3 right and #40H
; c) rotate 3 left and #20H
; d) rotate 1 left and #01H
; e) rotate 1 right and #08H

```

```
; f) rotate 3 right and #04H
; g) rotate 3 left and #02H
; h) rotate 1 left and #01H
;
;   Doorgeefadres: R3; hulp R4

SPIEGEL: MOV     A,R3                ; a)
          RR      A
          ANL     A,#80H
          MOV     R4,A
          MOV     A,R3                ; b)
          RR      A
          RR      A
          RR      A
          ANL     A,#40H
          ORL     A,R4
          MOV     R4,A
          MOV     A,R3                ; c)
          RL      A
          RL      A
          RL      A
          ANL     A,#20H
          ORL     A,R4
          MOV     R4,A
          MOV     A,R3                ; d)
          RL      A
          ANL     A,#10H
          ORL     A,R4
          MOV     R4,A
          MOV     A,R3                ; e)
          RR      A
          ANL     A,#08H
          ORL     A,R4
          MOV     R4,A
          MOV     A,R3                ; f)
          RR      A
          RR      A
```

```
RR      A
ANL    A , #04H
ORL    A , R4
MOV    R4 , A
MOV    A , R3          ; g)
RL     A
RL     A
RL     A
ANL    A , #02H
ORL    A , R4
MOV    R4 , A
MOV    A , R3          ; h)
RL     A
ANL    A , #01H
ORL    A , R4
MOV    R3 , A
RET

;*****
```

EINDE :

END

Bibliografie

- [1] J. Doutreloigne, H. De Smet, and A. Van Calster, “A New Architecture for Monolithic Low-Power High-Voltage Display Drivers”, Proceedings of the 20th International Display Research Conference IDRC2000, (Palm Beach, Florida,USA), pp. 115 - 118, September 2000.
- [2] A. Monté, J. Doutreloigne, and A. Van Calster, “An Intelligent Driving Scheme for High-Voltage Display Drivers”, Proceedings of the 11th International Display Workshops IDW '04, (Toki Messe, Niigata, Japan), pp. 1737-1740, (December 2004).
- [3] Sergio Franco, “Design with operational amplifiers and analog integrated circuits (p73-91)”, Third edition, 2002 San Francisco State University
- [4] Charles Kitchin, Lew Counts, “A designer’s guide to instrumentation amplifiers 2nd edition”, <http://www.analog.com>.
- [5] DS89C420 Microcontroller, http://www.maxim-ic.com/quick_view2.cfm/qv_pk/2963

Lijst van figuren

1.1	Voorbeelden van bistabiele beelschermen en hun toepassingen	1
1.2	Complexe golfvormen om een beeld weg te schrijven	2
2.1	Spanningsval: Nuttige spanning V_d tov. V_{cm}	4
2.2	Opamp: V_{uit} ifv V_{in} en V_{cm}	5
2.3	Instrumentatieversterker	6
2.4	Verschilversterker: Berekening V_{uit}	7
2.5	Differentiële versterker	8
3.1	(a) Inverterende versterker (b) Niet-inverterende versterker	11
3.2	Offset instellen	12
3.3	Schema ADC	16
3.4	Schema differentiële versterker	16
3.5	Typische aansluiting voor de ADC	17
3.6	Meting: sinus op kanaal 85V	18
3.7	Meting: Offset op kanaal 85V	19
3.8	Meting: Offset op kanaal 85V met RC-filter	20
3.9	Diodeschakeling ter bescherming van ADC	21
3.10	Schema analoge hardware	22
4.1	Protocol: Meten in snelle mode	25
4.2	Aansluiting seriële communicatie met ADC	27
4.3	Protocol: Seriële communicatie	27
4.4	Interconnectie tussen Ram-geheugen en microcontroller	29
4.5	In-system programmeer circuit	31
4.6	Algoritme om de uC seriëel te programmeren	33

4.7	Digitaal schema	34
5.1	Seriële communicatie met ADC	42
5.2	Voorbeeld: verzenden en ontvangen	42
5.3	Instelling seriële poort	50
5.4	Instelling omzettingen	52
5.5	Flowchart: Werking ADC	54
5.6	Flowchart: Volledig programma	55
6.1	Screenshot: Applicatie	57
6.2	(a) Sinus zonder offset, afgevlakt op -0,7V (b) Sinus met offset 2,5V	58
6.3	Niet inverterende versterker met offset	59
6.4	Meting: Resetpuls en beeldinformatie op een rij	60
6.5	Meting: Stroommeting op 60V	60
6.6	Meting: Detail stroommeting op 60V: 0 tot 0,8ms	61
6.7	Meting: Detail stroommeting op 60V: 12 tot 13ms	61
B.1	Foto meetcircuit	64

Lijst van tabellen

2.1	Stroom en spannings-niveau's	4
3.1	Ideale versterkingen	13
3.2	Gekozen weerstanden en versterking	14
3.3	Gemeten versterking	19
3.4	Gemeten offset per kanaal	20
3.5	Berekening RC-filter	21
4.1	Controleregister van de ADC	24
5.1	Instelling timer1-registers (X=dont-care bit)	37
5.2	Geteste baud-rates met timer 1 instelling	38
5.3	Instelling RAM registers	40
5.4	Instelling registers ADC	43
5.5	Controle meetwaarden	46

