Høgskulen på Vestlandet

BACHELOROPPGAVE

BO19E-34

MÅLING AV BAKGRUNNSSTRÅLING

Fakultet for ingeniør- og naturvitskap

Institutt for elektrofag

Ole Mikal Glenjen Kjepso
Dominik Karbowski

31. mai. 2019

Dokumentkontroll

Rapportens tittel:	*Dato/Versjon*
Måling av bakgrunnsstråling Measurement of background radiation	31. mai. 2019
	Rapportnummer: BO19E-34
Forfatter(e): Ole Mikal Glenjen Kjepso Dominik Karbowski	*Studieretning:* 16HKOM
	Antall sider m/vedlegg 40
Høgskolens veileder: Ilker Meric, Adis Hodzic, Kyrre Skjerdal	*Gradering:* Åpen
Eventuelle merknader: Vi tillater at oppgaven kan publiseres.	

Oppdragsgiver:	*Oppdragsgivers referanse:*
Høgskulen på Vestlandet	
Oppdragsgivers kontaktperson(er) (inklusiv kontaktinformasjon): Ilker Meric, ilker.meric@hvl.no	

Forord

Denne rapporten er en avsluttende oppgave i bachelorstudiet vårt innen kommunikasjonsteknologi ved Høgskulen på Vestlandet, våren 2019. Oppgaven har vært svært lærerik og utfordrende, og vi har tilegnet oss mye kunnskap som vi tar med oss videre.

Vi ønsker å takke våre veiledere Ilker Meric, Adis Hodzic og Kyrre Skjerdal for masse god hjelp med både litteratur og konstruktiv tilbakemelding gjennom hele oppgaven.

En ekstra takk til Adis Hodzic, studiekoordinator vår, for gode programmering tips knyttet til Arduino og tilbakemelding underveis i oppgaven. Oppgaven hadde vært mye mer vanskelig uten kunnskapen vi har fått fra han under hele studieløpet.

Takk til Lars Ekroll for å ferdigstille Geiger-røret slik at vi kunne koble den til kretsen vår, for lån av utstyr, og for å stille utfordrende spørsmål vi kunne bryne oss på.

Takk til Mona Øynes, høyskolelektor ved Institutt for helse og funksjon, for at vi fikk komme på besøk for å teste detektoren med kilder på MR-laben.

Vi vil også takke alle forelesere våre vi møtte i løpet av studietiden vår.

Til slutt vil vi takke for en utfordrende og spennende studietid ved Institutt for elektrofag!

Sammendrag

I denne bacheloroppgaven har vi tatt for oss måling av bakgrunnsstråling ved hjelp av et Geiger-Müller rør som strålingsdetektor. Vi har laget et komplett og mobilt måleinstrument. Instrumentet består av et Geiger-Müller rør, en mikrokontroller og en god del tilleggskomponenter som skal gjøre det mulig å registrere, telle og loggføre hvor mye radioaktivitet det er i et område.

Når et foton går inn i Geiger-røret vil det ha en viss sannsynlighet for å utløse et såkalt «skred» med elektroner. Da blir det sendt ut en analog puls fra katoden med relativ høy spenning. Denne pulsen måler vi ved hjelp av en mikrokontroller.

Alle registrerte pulser telles og loggføres på et SD-kort. Siden mikrokontrolleren ikke har mulighet å lagre dato og tid har vi tatt i bruk en klokke modul. Løsningen vår dermed loggfører også dato og klokkeslett på alle utførte målinger.

Antall registrerte pulser i en viss periode gir en god indikasjon på hvor mye bakgrunnsstråling det faktisk er på det stedet hvor målingene ble utført. Alle lange målinger av bakgrunnsstråling vi har utført lå mellom 29 og 31 tellinger per minutt i gjennomsnitt.

Vi har testet løsningen vår ved å utføre både korte og lange målinger av bakgrunnsstråling forskjellige steder på høyskolen. I tillegg til det har vi også testet løsningen vår med Americium-241 og Cesium-137 kilder på MR-laben på høyskolen.

Vi har også utviklet et lite program som kan hente alle målinger fra SD-kortet og lagre disse i separate filer på PC. Programmet har også mulighet å slette alle dataene på kortet, synkronisere klokke på mikrokontrolleren og konfigurere måleparametere.

Løsningen vår oppførte seg veldig stabilt både under testing og mens vi målte. Vi hadde dessverre ikke mulighet å sammenligne resultatene våre med en annen Geiger detektor.

Måleinstrumentet vårt kan brukes som den er men kan også være godt utgangspunkt for videre arbeid. På slutten av rapporten har vi beskrevet mulige utvidelser og forbedringer til løsningen vår.

Løsningen vår er lett å reprodusere og kan relativt enkelt utvides til en Internet of Things løsning. Fordelen med en sånn utvidelse er at man kan kontinuerlig overvåke stråling over et område uten å måtte hente data fra SD-kortet manuelt.

1 Innhold

1 Innledning

1.1 Oppdragsgiver

Oppdragsgiveren vår er Høgskulen på Vestlandet.

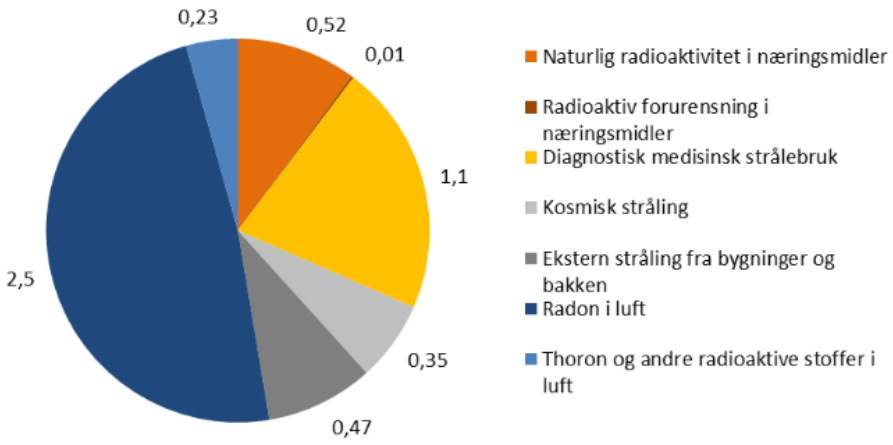
Som studenter begynte vi på Høgskolen i Bergen – HiB. men vi blir eksaminert i det som nå er fusjonert Høgskulen på Vestlandet – HVL. HVL er en av de største utdanningsinstitusjonene i landet, med om lag 16 000 studenter. HVL ble etablert 1. januar 2017, da Høgskolen i Bergen, Høgskulen i Sogn og Fjordane og Høgskolen Stord/Haugesund fusjonerte. HVL har fem campuser på Vestlandet: Førde, Sogndal, Bergen, Stord og Haugesund.

1.2 Problemstilling

1.2.1 Bakgrunnsstråling

Ioniserende stråling er et begrep som brukes om stråling med høy nok energi til å rive løs elektroner fra materie det treffer og dermed danne ioner. Dette kan både være partikler og elektromagnetisk stråling. Ioniserende stråling kan ødelegge cellevev, som kan skade biologisk materie. Store doser kan ha akutte skadevirkninger, men også mindre doser kan ha konsekvenser på lang sikt, som kreft.

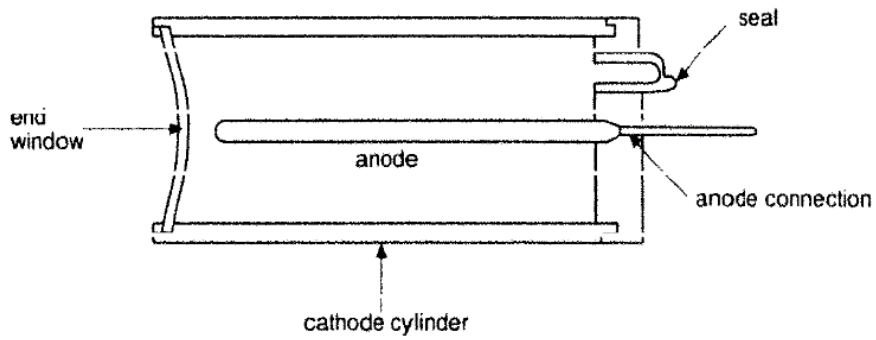
Naturlig bakgrunnsstråling er ioniserende stråling som er tilstede naturlig i vår hverdag, og utgjør mesteparten av strålingen vi blir utsatt for. Bakgrunnsstråling består både av naturlige og kunstige kilder. Hver dag blir vi eksponert for stråling som kommer fra verdensrommet, stråling fra konstruksjonsmateriale og radioaktive grunnstoffer fra jorden. Som man kan se i Figur 1 står radongass for nesten halvparten av stråledosen den norske befolkningen mottar. Det er også radioaktive isotoper i maten vi får i oss. Medisinsk stråling blir sett på som kunstig, eksempelvis en røntgenundersøkelse.



Figur 1 Samlet oversikt over bidragene til den gjennomsnittlige stråledosen (mSv/år) til befolkningen i Norge fra ulike kilder [1, p. 15]

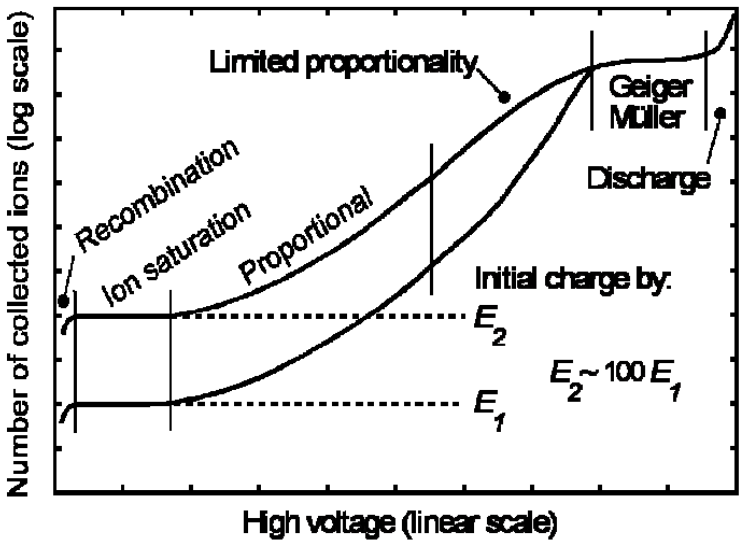
1.2.2 Geiger-Müller rør

Geiger-Müller rør er en simpel strålingsdetektor som ble introdusert i 1928 og er blant de mest brukte strålingsdetektorene grunnet at den er enkel å bruke, har en relativt lav kostnad og er robust i forhold til andre detektorer.



Figur 2 Konstruksjon av et typisk Geiger Müller rør [2, p. 3]

Dette er et sylinderformet metallrør fylt med en gassblanding med lavt trykk. Røret kobles til en høyspenningskilde der katoden er i rørveggene, mens anoden er en metalltråd i midten av røret. Dette gjør at det går et sterkt elektrisk felt gjennom gassen. Når et foton går inn i røret har det en viss sannsynlighet for å rive løs et elektron fra et gassmolekyl og danne et ionpar. Dette kalles fotoelektrisk effekt hvis all energien absorberes av elektronet, mens det kalles Compton-spredning dersom bare en del av fotonets bevegelsesmengde overføres til elektronet, og vi i tillegg får et restfoton. Det elektriske feltet vil akselerere elektronet mot anoden og det ioniserte molekylet mot katoden. Elektronet vil få nok energi til å kunne kollidere med andre atomer i gassen og eksitere dem. Dette vil skape en kjedereaksjon i røret kalt et Townsend skred. Eksiterte gassatomer vil sende ut UV-stråling som vil utløse flere Townsend-skred [3, p. 202]. Dette vil skape en strøm som er stor nok til å kunne måles.



Figur 3 Geiger-Müller regionen

Figur 3 viser oppførselen til forskjellige gassdetektorer. De to kurvene beskriver amplituden til pulsen på utgangen produsert av to fotoner med forskjellige energier. Detektorer som opererer ved lavere spenninger kan detektere energien til strålingen. I Geiger-Müller regionen vil man ikke kunne skille mellom energiforskjellene lenger. Den høye spenningen fører til at man får fullstendig utladning uavhengig av energi. Ved høyere spenninger vil det sterke feltet føre til at utladning kan skje uten at det utløses av radioaktiv stråling. Om detektoren opererer i sistnevnte region over lengre tid, vil det ta skade.

1.3 Hovedidé for løsningsforslag

Målet med oppgaven er å lage en fullstendig strålingsdetektor. Den skal kunne måle bakgrunnsstråling altså lave strålingsrater. Vi har fått et Geiger-rør fra en av våre veiledere og det er en *ZP1200* fra *Centronic*. Pulser fra Geiger-røret må registreres og telles av en mikrokontroller. Målingene må lagres på en strukturert form slik at disse dataene kan analyseres etterpå. For at det skal være praktisk å måle vil vi at vår løsning er kompakt og mobilt.

2 Kravspesifikasjon

Følgende kravspesifikasjoner ble gitt i oppgaveteksten:

- Et mikrokontroll (Arduino) program som skal kommunisere med sensoren og dataloggingssentralen skal utvikles og dokumenteres.
- Et program som kommuniserer med mikrokontrolleren (mottak/lagring/presentasjon av data) skal utvikles og dokumenteres.
- Et program som skal kjøre statistisk analyse av data fra sensoren for å detektere «hot spots» om det er noen. Programmet skal utvikles og dokumenteres av studenter.

3 Analyse av problemet

Oppgaven kan basert på kravspesifikasjonene deles i fire deler:

- Designe og lage en elektronisk krets
- Utvikle et mikrokontrollprogram som teller antall pulser
- Utvikle programvare som kan bearbeide data fra mikrokontrolleren
- Utvikle en effektiv metode for statistisk analyse av målte data

I tillegg til kravspesifikasjonene satt vi oss tidlig et mål om å gjøre løsningen mobil for å kunne gjøre målinger ved forskjellige lokasjoner også uten strømtilførsel. Setter derfor følgene krav til kretsdelen:

- Må omforme pulsen fra røret til et format mikrokontolleren kan tolke
- Må kunne forsynes med batteri
- Må kunne levere høy nok spenning til Geiger-røret
- Må kunne starte målinger ved hjelp en knapp
- Må ha mulighet til å lagre dataen lokalt

3.1 Centronic ZP1200

Røret gitt av oppdragsgiver er *ZP1200* fra *Centronic*. Katoden og anoden er laget av legeringer som i hovedsak består av jern og krom. Gassen inni røret består av neon, argon og brom. Operasjonsområdet til detektoren er 450 - 650 V. [4, p. 2]

Figur 4 Bilde av Centronic ZP1200

3.2 Utforming av mulige løsninger

Siden vi ønsker å gjøre løsningen mobil, slik at målinger enkelt kan utføres på forskjellige steder må løsningen kunne forsynes av batterier og kunne lagre data. Siden *ZP1200* behøver høyspenning må vi kunne konvertere til 500 V fra batteriene.

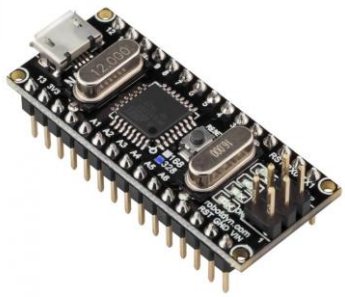
3.2.1 Vurderinger i forhold til verktøy og HW/SW komponenter

Siden en mikrokontroller Arduino ble nevnt i kravspesifikasjonene var naturlig å velge dette kortet til prosjektet vårt. Våre veiledere var samtidig åpne for andre løsninger som kunne vært bedre enn Arduino. Vi har også vurdert andre populære mikrokontrollere.

3.2.1.1 Arduino Nano

Arduino Nano er en liten og kompakt mikrokontroller med bygd inn *Atmel* sin *ATmega328* chip. Operasjonsfrekvens er på 16 MHz. Den har 8 analoge IO-porter og 13 digitale IO-porter. Micro-USB port kan brukes til strømforsyning eller seriell kommunikasjon med PC.

Denne type mikrokontroller hadde vi mest erfaring med fra før. Vi var borti Arduino i sammenheng med et programmeringsfag vi hadde i løpet av studietiden. Da lærte vi oss mye om selve kortet men også hvordan det kan programmeres. Dette var naturlig å velge dette men vi har likevel vurdert andre muligheter.



Figur 5 Arduino Nano

3.2.1.2 ESP

ESP-mikrokontrollere har mye tilfelles med vanlig Arduino men denne har mulighet for å koble mot og kommunisere over et WiFi nettverk. Det er en stor fordel med WiFi fordi vi kunne sende målte data til en database istedenfor å lagre de på et SD-kort. Da får man også mulighet å kunne overvåke og analysere data i sanntid. Dette kortet har også høyere klokkefrekvens. For eksempel *ESP8266* har klokkefrekvens på enten 80 MHz eller 160 MHz. Dette kortet har vi ingen erfaring med og ville krevd en del å sette seg inn i.

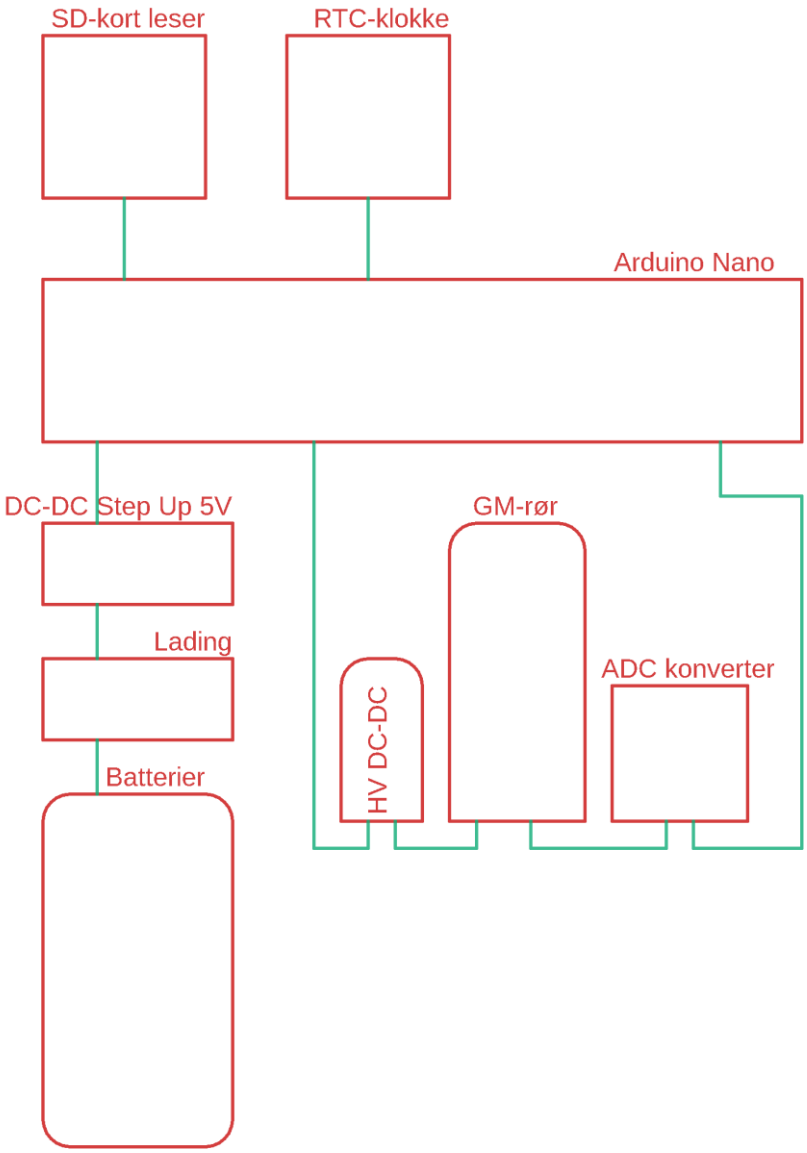
Figur 6 ESP8266 WiFi

3.3 Konklusjon

Vi har valgt Arduino Nano fordi det var kortet hadde vi mest erfaring med. 16 MHz burde være tilstrekkelig siden vi skal måle ved lave tellerater og at den ikke behøver å gjennomføre tunge kalkuleringer mens den måler. Dette er mye brukt mikrokontroller, så når vi møter problemer underveis vil det være mye hjelp å finne på nettet.

4 Realisering av valgt løsning

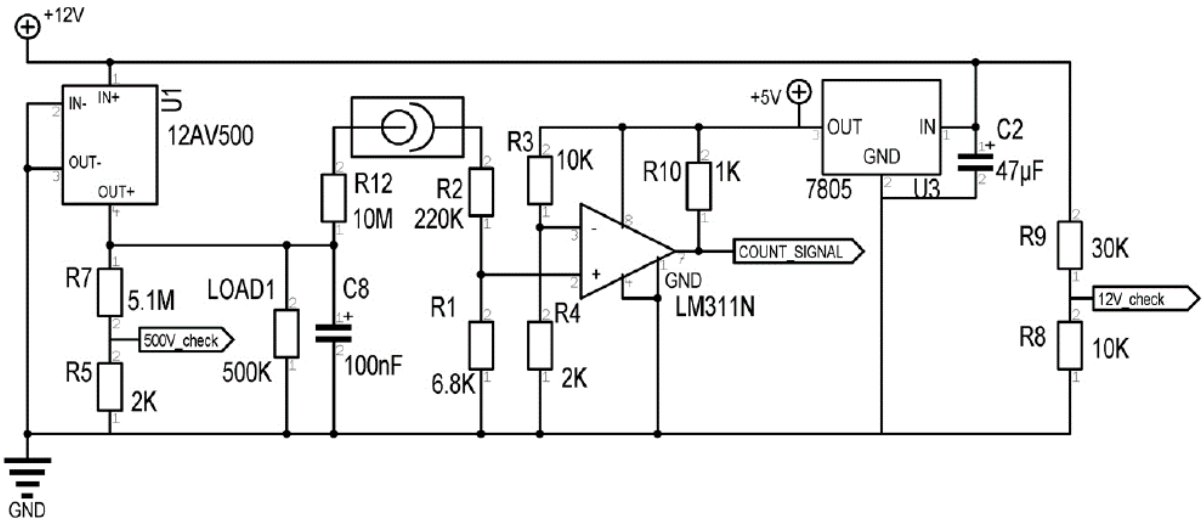
4.1 Blokkdiagram



Figur 7 Blokkdiagram som viser en grov oversikt av kretsen

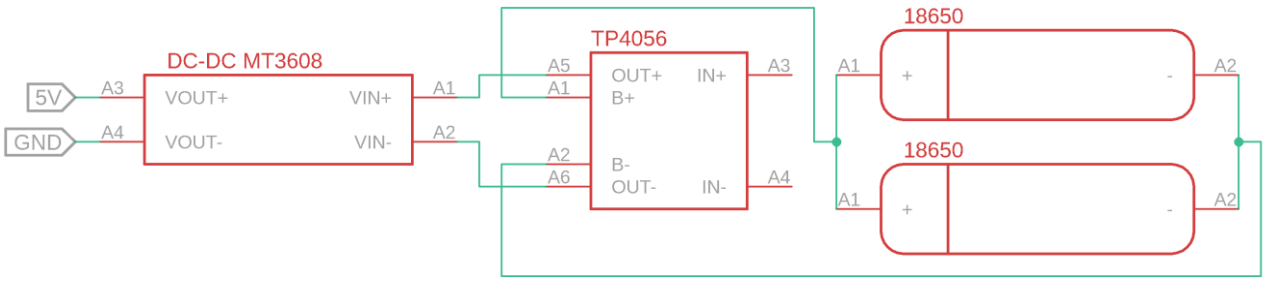
4.2 Krets

Vi har tatt utgangspunkt i en krets laget av Magne Lauritzen. Han laget en lignende krets i sammenheng med et prosjekt på Universitetet i Bergen. [5] Siden vi satt oss et mål å gjøre løsningen vår mobil måtte vi ta i bruk flere komponenter blant annet batterier, DC-DC omformere, SD-kort leser og RTC-klokke.



Figur 8 Kretstegning av løsningen til Magne Lauritzen [6, p. 4]

4.2.1 Strømforsyning til mikrokontrolleren



Figur 9 Strømforsyning – kretstegning

Strømforsyning består av tre elementer:

- 2 x 18650 Li-Po batterier (3300 mAh hver)
- TP4056 (Lade modul med overspenning beskyttelse)
- MT3608 (DC-DC Step-Up converter)

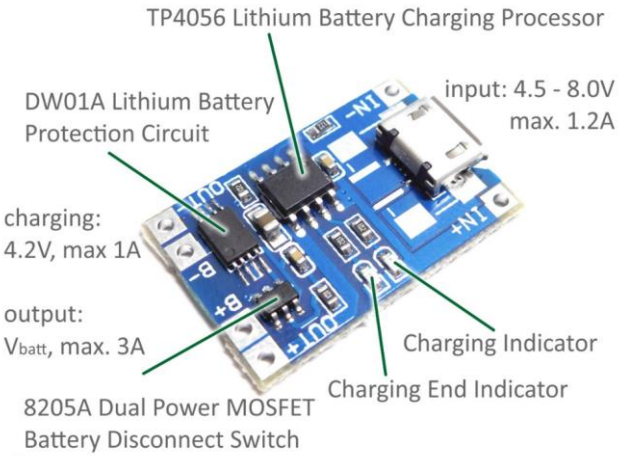
4.2.1.1 Batterier

Til strømforsyning bruker vi to 18650 batterier med kapasitet 3300 mAh altså 6600 mAh. Den nominelle kapasiteten avviker ofte fra virkeligheten så vi antar den reelle totale kapasiteten er på ca. 6000 mAh. Dette skal være nok til å utføre flere dagers måling uten å måtte lade batterier. Batterier av type 18650 har en spenning på 2,8 V når de er utladet og 4,2 V når de er fulladet.

Figur 10 Et eksempel på et batteri av type 18650

4.2.1.2 TP4056

TP4056 er en kompakt komponent som kan lade *18650* batterier. Den sørger for at batterier ikke overlades og ikke utlades helt siden dette kan ødelegge batteriene. *TP4056* har en mikro-USB port for strømtilførsel. Den kan lades med en vanlig 5 V mobillader. Vi har brukt en på 2,5 A først og da ble komponenten varm. Vi skjønte at det var for høy strømtilførsel og ifølge dokumentasjonen kan man lade med maks. 1 A. Vi hadde en på 1,5 A tilgjengelig, og selv om det er litt for mye, ser det ut til å ha fungert greit. Selve komponenten har to LED dioder som lyser grønt når batteriene er fulladet og rødt når de lades.



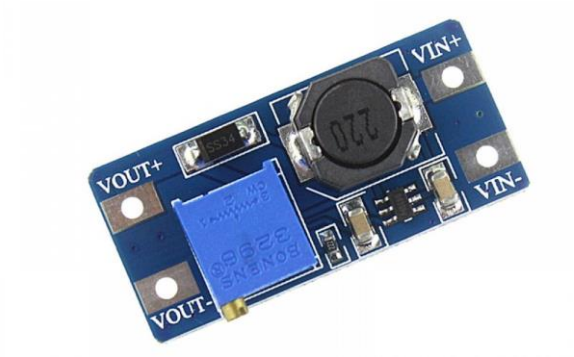
Figur 11 TP4056 med tilleggsinformasjon og parametere

4.2.1.3 MT3608

MT3608 er en enkel og justerbar DC-DC «step-up» konverter for å konvertere spenningen fra batteriene. Den justeres ved å skru på et potensiometer. Vi har justert den slik at den gir 6,4 V ut. VIN inngangen på Arduino opererer mellom 6 V og 9 V så 6,4 V fra *MT3608* er innenfor grensen. Varianten av *MT3608* vi har brukt tålte veldig lite og vi klarte å ødelegge noen av dem ved å kortslutte + og – utganger med en finger. Derfor kan det være lurt å lodde alle komponenter for så å koble til batteriene.

Input voltage	2 – 24V
Output voltage	5 – 28V
Max. Output current	2A
Conversion efficiency	< 96%

Figur 12 Parametere til MT3608

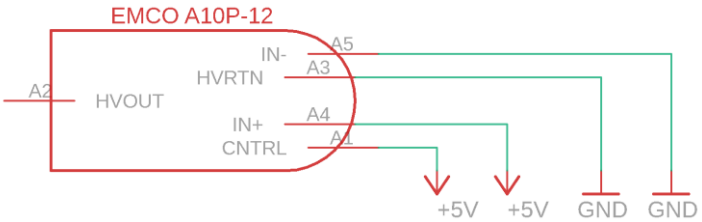


Figur 13 Bilde av MT3608

4.2.2 Høyspenning strømforsyning til Geiger-Müller røret

Nesten alle Geiger-rør krever høyspenning for at det skal være mulig at skred utløses. Vårt Geiger-rør opererer mellom 400 – 600 V. Produsenten anbefaler 500 V til å forsyne *ZP1200*. Vi valgte *A10P-12* fra *EMCO* som gjør det mulig å konvertere 5 V til 500 V. Varianten vi har bestilt opererer mellom 0 – 12 V på inngangen og 0 – 1000 V på utgangen.

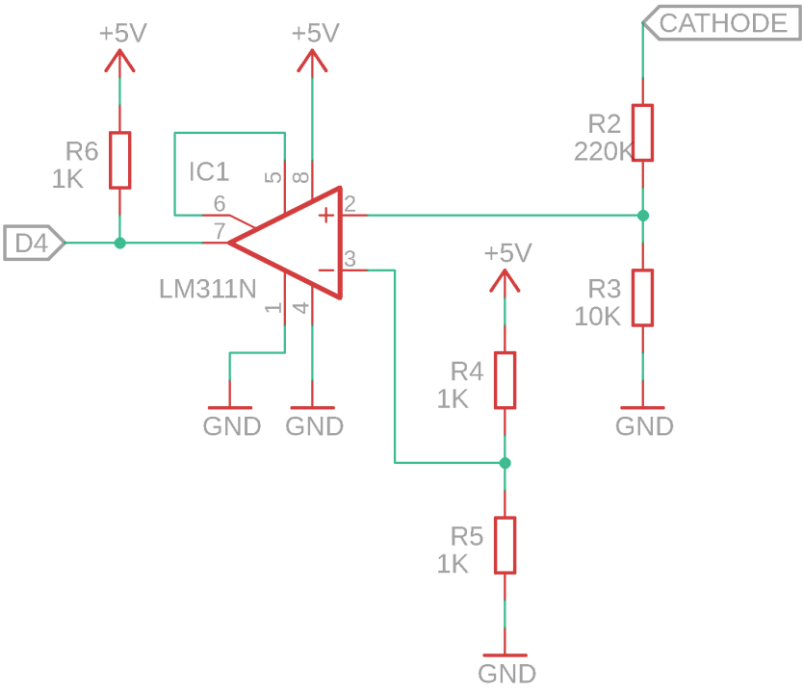
Figur 14 EMCO High Voltage Series



Figur 15 EMCO A10P-12 med oppkobling

Komponenten er enkel å bruke og å koble opp. Spenningen ut avhenger både av spenning på CNTRL inngangen, IN+ inngangen og last. Dersom spenningen på IN+ og CNTRL har samme verdi, vil denne være proporsjonal med spenningen på utgangen ifølge dokumentasjonen. Vi har brukt 5 V fra utgangen til Arduino inn på IN+ og CNTRL-inngangen. Da fikk vi fikk ca. 520 V på HV utgangen. IN- og HVRTN (High Voltage Return) koblet vi til jord fra Arduino altså Arduino og EMCO har felles jord. Første versjon av kretsen hadde separat jord, men da ble komparatorkretsen vanskeligere å designe. Dette skapte trøbbel og vi bestemte oss for felles jord for begge komponentene. CNTRL inngang på EMCO kan brukes til å finjustere spenningen mens IN+ holdes konstant på 5 V. Vi har brukt dette når vi skulle finne platået til Geiger-røret i seksjon 0. Komponenten ser ut til å fungere bra så lenge vi har en last med høy motstand, 10 MΩ. Da leverer den høy spenning, men med lav effekt.

4.2.3 Pulsdeteksjon

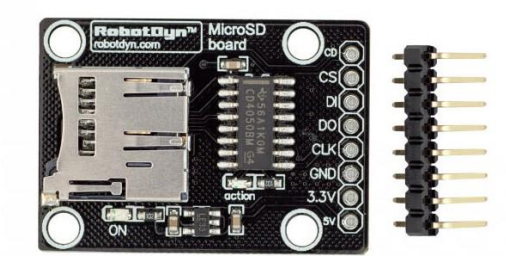


Figur 16 Komparator LM311 – oppkobling

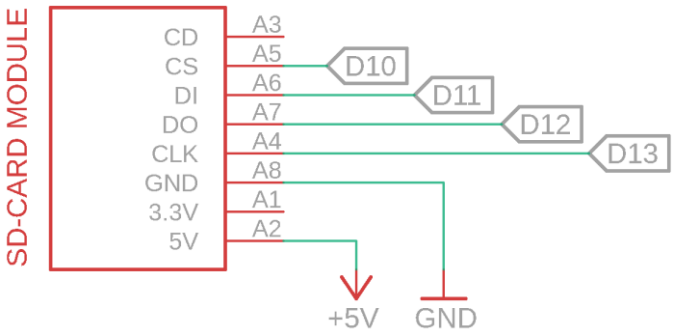
For å få en kontrollert puls på inngangen til Arduino, har vi brukt en enkel komparatorkrets. Den fungerer slik at dersom spenningen på plussinngangen er høyere enn den på minusinngangen, vil den på utgangen gi logisk høy. Dette gjør at det bare er pulser med en gitt størrelse som vil bli detektert av mikrokontrolleren slik at støy ikke kan risikere å bli registrert som en puls. Vi studerte pulsene på inngangen ved hjelp av et oscilloskop og målte at de var på rundt 8 V. Vi valgte en spenning på 2,5 V som terskelverdi på komparator, siden dette er over støyen med god margin. Senere innså vi at denne burde senkes, noe vi diskuterer i seksjon 5.3.

4.2.4 Loggføring av data

Arduino Nano har begrenset lagringsplass. Den har 2kB med SRAM, 32kB med flashminne og 1kB EEPROM [6]. Data lagret i SRAM vil gå tapt når strømmen kobles fra. Flashminne vil beholde dataene når strømmen er koblet fra, men det er her programvaren er lagret så vi ønsker ikke å lagre dataene her. EEPROM fungerer som en liten harddisk og er best egnet til å lagre data slik vi ønsker, men bare på 1kB som er for lite til å kunne loggføre flere tusen målinger. På grunn av dette måtte vi finne en løsning som kunne lagre større mengder av data. Vi fant ut at det er enkelt å bruke ett SD-kort sammen med Arduino, så vi bestilte en modul for dette.

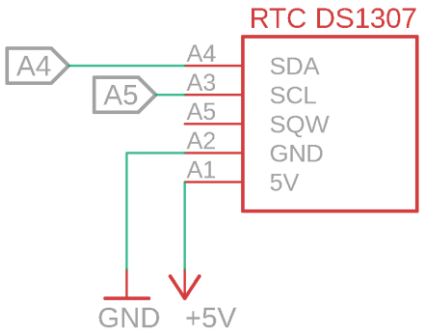


Figur 17 MicroSD board fra Robotdyn

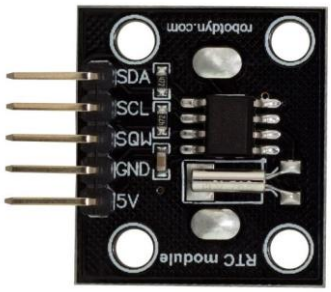


Figur 18 Oppkobling til SD-kort leser

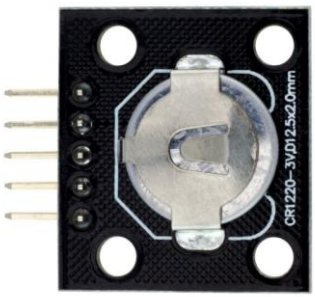
4.2.5 Klokke modul



Figur 19 RTC-klokke – oppkobling



Figur 20 RTC-klokke



Figur 21 RTC-klokke – baksiden

For å loggføre data er det også gunstig at dataene har et tidsstempel. RTC-klokken har et eget batteri og vil derfor holde styr på tiden selv om mikrokontrolleren er skrudd av. Etter å ha stilt klokken leste vi av tiden en uke senere og den viste fortsatt nøyaktig på sekundet.

4.3 Software

4.3.1 Arduino kode

Arduino koden består av 5 filer. To .cpp filer med tilhørende .h filer for håndtering av SD-kort og RTC-klokke. I tillegg har vi en .ino fil som utfører selve målingen. Flere filer var opprinnelig planlagt skulle legges til for håndtering av OLED.

Etter at programmet er startet vil det kontinuerlig sjekke for seriell input eller om knappen er trykket inn og utføre operasjoner ut ifra dette. Spesifiserte kommandoer fra seriell er:

r#	Sender måledata fra SD-kort over seriell
e#	Sletter alle målingsdata fra SD-kort
cyyyyMMddhhmmss#	Justerer klokkeslettet til RTC
s6000000010#	Justerer parametere i config.txt
v#	Sender parametere fra config.txt over seriell

For s# kommandoen vil de fem første symbolene representere antall millisekunder for et målingsintervall, mens de fem neste representerer antall intervaller i en måling. Eksempelet i tabellen vil derfor gjøre 10 målinger der hver måling er 60 sekunder. Når programmet kjører vil parameterne lagres som en variabel av type *«unsigned int»* som har en maksimumsverdi på 65535. Om man overgår dette, eller avviker fra de spesifiserte kommandoene, vil man få uforutsigbar oppførsel.

4.3.1.1 SD

I Arduino sine standardbibliotek finner man et bibliotek for lesing og skriving til SD-kort. Biblioteket baserer seg på Serial Periphal Interface (SPI) som «er en protokoll brukt av mikrokontrollere for å kommunisere med periferienheter over korte avstander». Ved hjelp av dette biblioteket kan man lese og skrive til SD-kortet ved hjelp av et par funksjoner.

På SD kortet er det to filer. En fil for konfigurasjon «config.txt» og en fil der vi lagrer målte data «fil.txt». Konfigurasjonsfilen vil leses idet programmet starter.

4.3.1.2 RTC

For å håndtere RTC-klokken bruker vi RTC-lib. Dette er et bibliotek fra *Adafruit* og lar Arduinoen hente datoen eller justere den ved å kalle `RTC.now()`.

4.3.1.3 Telling av pulser

Til telling av pulsene er funksjonen pulseIn() brukt, som er en del av standardbiblioteket til Arduino. Når funksjonen kalles vil den vente på at inngangen på en spesifikk pin vil gå fra lav til høy eller omvendt. Funksjonen vi returnere varigheten til pulsen i mikrosekunder. Dersom en puls ikke detekteres innen en oppgitt tid vil den returnere 0. [7]

Når en måling starter vil klokkeslettet hentes fra RTC og skrives til SD-kort for å fortelle starttidspunkt til målingen som skal utføres.

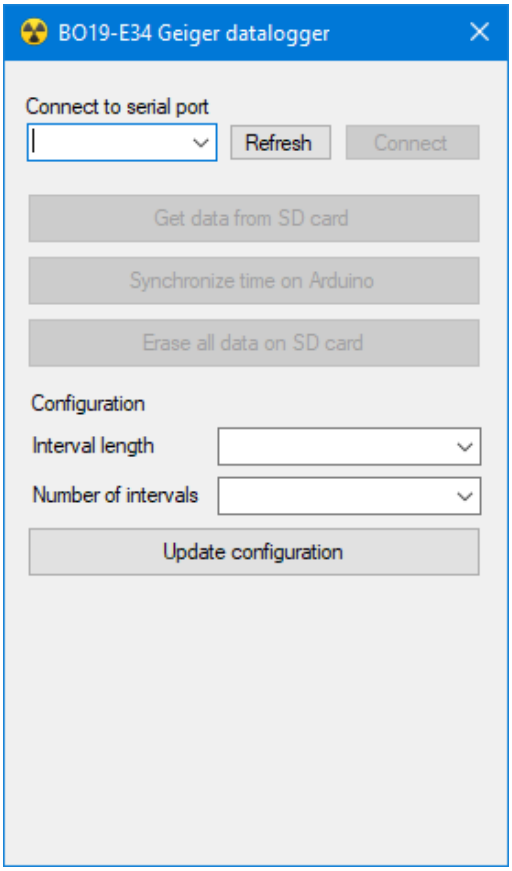
```
do

{

      duration = pulseIn(PULSE_PIN, HIGH, 1000);
      if (duration != 0)
            count++;
      currentTime = millis();
} while ((unsigned long)(currentTime - startTime) <= interval);
```

pulseIn() venter på at inngangen skal gå fra lav til høy i 1000 millisekunder. Når en verdi returneres sjekkes det om den er 0. Dersom den ikke er det vil tellevariabelen økes med én. Deretter sjekkes det om tiden som er gått er mindre enn det spesifiserte intervallet, og dersom ikke, vil loopen gjentas. Når intervallet er nådd vil antall tellinger lagres til SD-kort og et nytt intervall vil startes.

4.3.2 Data logger programmet

Kommandoene som er spesifisert i Arduinoprogrammet gjør at man kan styre den fra en PC-terminal over seriell. Siden Arduinoprogrammet ikke sjekker om kommandoene er på riktig format, kan dette føre til trøbbel. Man skal alltid forvente dårlig brukerinput og vi valgte derfor å la et desktopprogram sørge for å sende kommandoene, slik at det alltid er på riktig format.

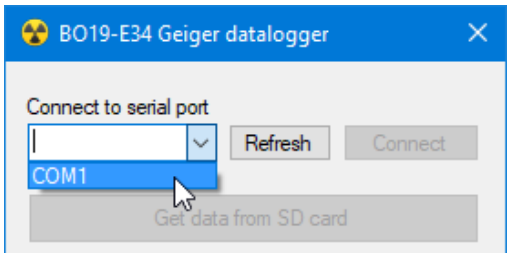
Vi har valgt *C#* som programmeringsspråk. Vi har brukt *Visual Studio 2019* som IDE (Integrated Development Environment). Vi har brukt *Windows Forms* som rammeverk til GUI (Graphical User Interface).



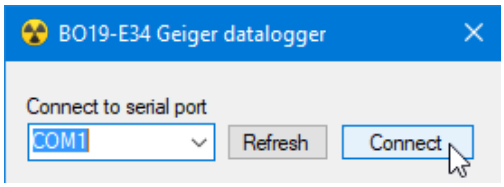
Figur 22 Datalogger

4.3.2.1 Funksjonalitet i programmet

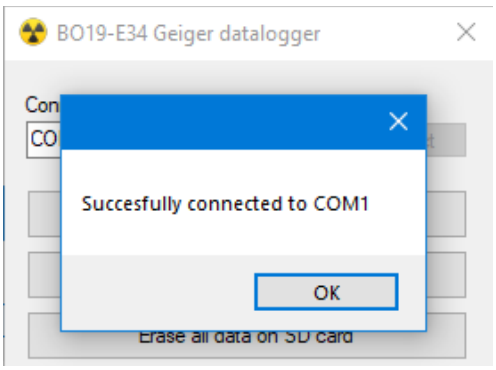
Når programmet har startet må man først velge USB porten Arduino er koblet til.



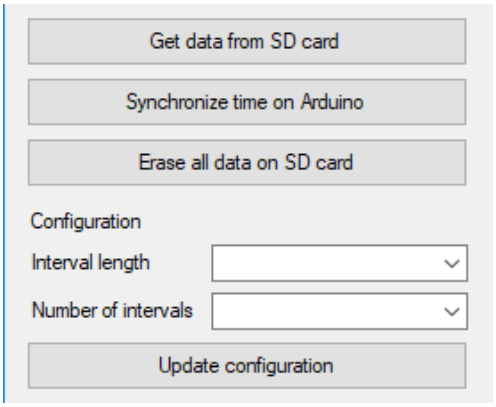
Da riktig port er valgt kan man trykke på «Connect».



Det skal komme opp et lite vindu med beskjed at oppkoblingen ble vellykket.



Nå har PC etablert en USB-tilkobling med Arduino og nå får man bruke funksjoner i programmet som ikke var tilgjengelig før.



Get data from SD card	Henter alle målte data fra SD kortet på Arduino og oppretter separat .csv fil for hver enkel måling. Filene lagres i samme mappe hvor .exe fil med programmet ligger.
Synchronize time on Arduino	Synkroniserer RTC-klokke på Arduino med klokke som er på PC.
Erase all data on SD card	Sletter alle malinger på SD kortet
Configuration	Konfigurerer lengde på måleintervall og antall intervaller.

4.3.2.2 Seriell kommunikasjon med USB

For at PC får kommunisere med Arduino må man først opprette USB-tilkobling. C# har en innebygd klasse som støtter USB kommunikasjon som heter `SerialPort`. Klassen ligger i «namespace» `System.IO.Ports`. For å kunne bruke SerialPort må man legge til `using System.IO.Ports;` i koden.

Vi har laget en enkel klasse `SerialHandler.cs` som tar seg av seriell kommunikasjon.

Vi har et objekt av klassen SerialPort.

```
public SerialPort serialPort = new SerialPort();
```

For å sjekke hvilke porter er tilgjengelige kaller vi funksjon `SerialPort.GetPortNames()` som returner en liste med alle porter.

```
public string[] GetAvailablePorts()
{
    return SerialPort.GetPortNames();
}
```

For å opprette en tilkobling må man sette opp `BaudRate`, velge port i `PortName` og kalle funksjon `Open()` på `SerialPort` objektet. Vi har valgt `115200` som baudrate. Opprinnelig brukte vi «default» verdi på `9600`, men dette var lite og overføring av data tok lang tid. Det er viktig at man setter opp samme baudrate både på PC og på Arduino.

```
public void ConnectToSerial(string port)
{
    try
    {
        serialPort.BaudRate = 115200;
        serialPort.PortName = port;
        serialPort.Open();
    }
    catch (Exception)
    {
        throw new Exception();
    }
}
```

Hvis koblingen er mislykket vil dette bli fanget opp av try-catch som vil hive en `Exception`.

Nå er vi klar å kommunisere med Arduino. Det er lurt å dobbelt sjekke om tilkoblingen er fortsatt åpen ved å bruke `IsOpen` «property» fra vårt `SerialPort` objektet. `IsOpen` er true hvis tilkoblingen er fortsatt tilgjengelig. For å sende en streng bruker vi `WriteLine()` funksjon.

```
public void SendCommand(string command)
{
    if (serialPort.IsOpen)
        serialPort.WriteLine(command);
}
```

Fra nå av kan vi bruke funksjonen `SendCommand(string command)` til å sende kommandoer til Arduino.

4.3.2.3 Klokke synkronisering

Siden vi har en funksjon `SendCommand(string command)` som tar en streng som parameter og sender det over seriell til Arduino så kan vi enkelt bruke den funksjonen til å synkronisere klokke.

Vi har laget en klasse `class GeigerHandler` som inneholder alle konfigurasjons funksjoner.

Arduino forventer en streng på følgende form:

```
c20190219120800#
```

I eksempelet over vil Arduino stille om dato og klokke til 19.02.2019 12:08:00.

I C# biblioteket finner vi en klasse `DateTime` og funksjonen `DateTime.Now.ToString()` som returner klokke og dato fra PC.

Vi har brukt den slik `DateTime.Now.ToString("yyyyMMddHHmmss")`

```
class GeigerHandler
{
    public static void SyncClockOnArduino(SerialHandler serialHandler)
    {
    serialHandler.SendCommand("c" + DateTime.Now.ToString("yyyyMMddHHmmss") + "#");
    }

/* ... */
}
```

4.3.2.4 Tømming av SD-kortet

For å tømme SD-kortet, altså alle målte data kan man bruke `e#` kommando. Vi gjør det helt likt som i forrige eksempel.

```
class GeigerHandler
{
     /* ... */
     public static void EraseSDCard(SerialHandler serialHandler)
     {
            serialHandler.SendCommand("e#");
     }
}
```

4.3.2.5 Konfigurasjon av parametere

For å konfigurere lengde på et måleintervall og antall intervaller må vi sende en kommando på lignende form `s6000000010#.` For å utføre en kontinuerlig måling kan man sette antall intervaller og lengden på et intervall til ønsket verdi, men ikke større enn 65535 `(unsigned small int)` som tilsvarer ca. 45 dagers lang måling. Programmet sørger for at kommandoen sendes på riktig format slik at Arduinoprogrammet oppfører seg som det skal.

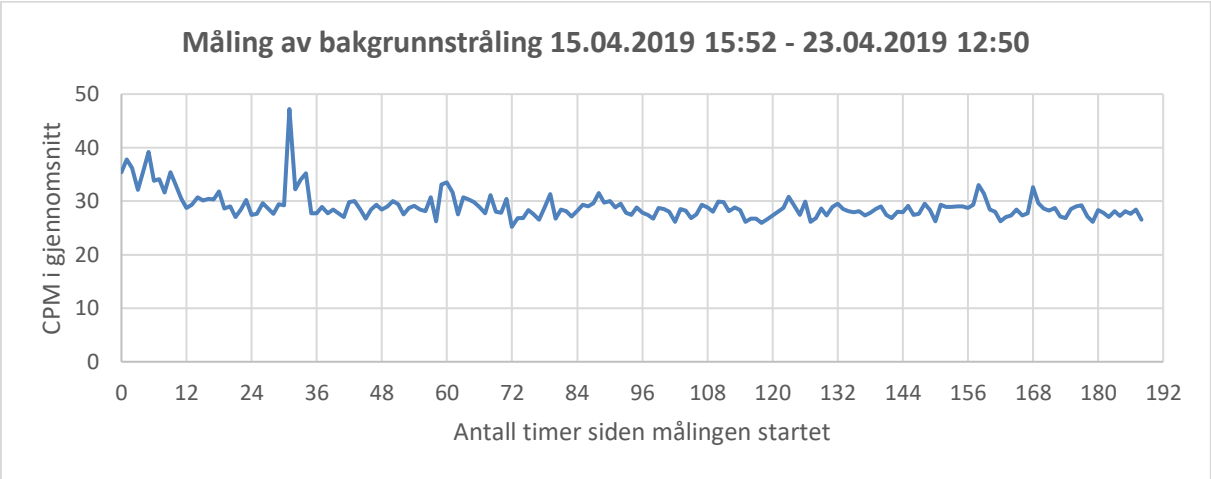
```
public static void ConfigureGeiger(SerialHandler serialHandler, string intervall,
string numberOfIntervals)
    {
        serialHandler.SendCommand("s" + intervall + numberOfIntervals + "#");
    }
```

5 Testing og målinger

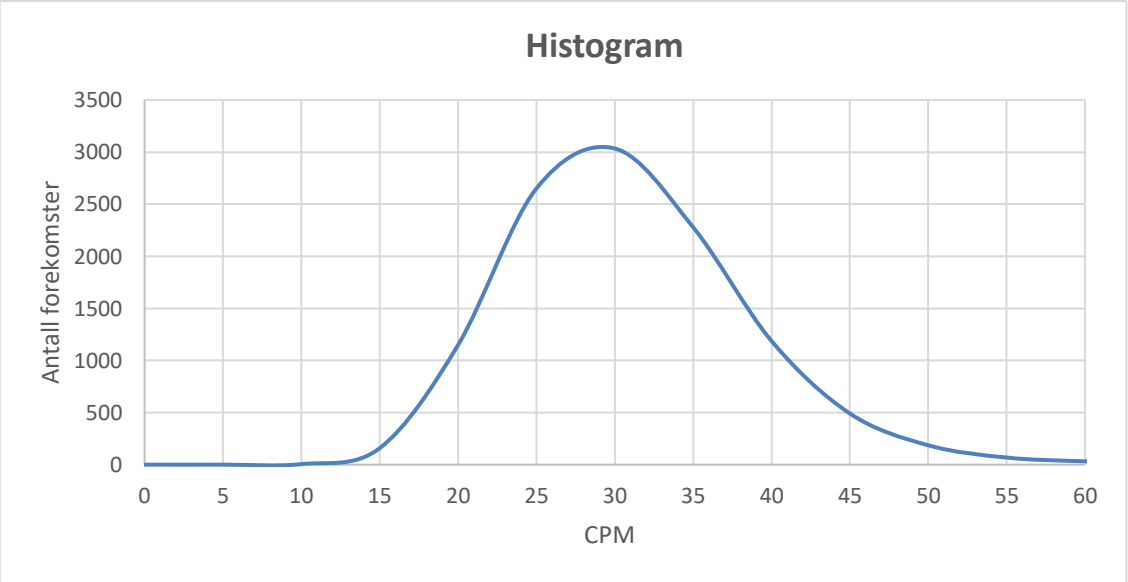
Mikrokontrolleren leverer forskjellige spenninger avhengig om en forsyner den med strøm fra batteriene eller fra USB-porten. 5,10 V med USB 4,98 V med batteri. Siden DC-DC konverter kontrolleres av denne spenningen vil også spenningen over røret bli høyere. 540 V med USB og 520 V med batteri. Måleresultatet blir derfor noe forskjellig. Den tenkte løsningen er batteridrevet så alle målinger er gjort med batterier. USB-port brukes bare til å overføre dataene.

5.1 Måling av bakgrunnsstråling

Vi gjorde en lang måling med den ferdige kretsen fra 15. til 23. april. Den er gjort på forskningslaben i fjerde etasje på høyskolen.



Figur 23 En ukes måling av bakgrunnsstråling



Figur 24 Histogram til en ukers måling

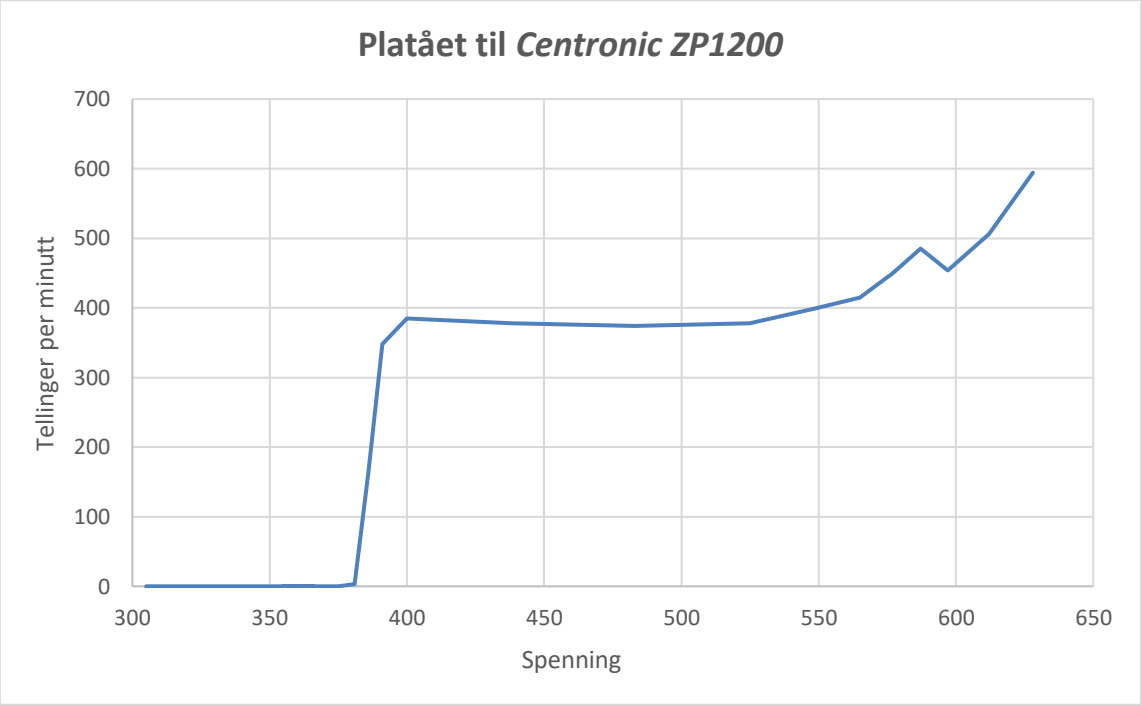
Figur 23 viser gjennomsnittlig tellerate for hver time siden målingen startet. Gjennomsnittet ligger på 29,08 målinger per minutt. Standardavviket er på 8,22 som er en del høyere enn det vi forventet hvis vi antar poisson-fordeling, som ville vært $\sqrt{\mu} = \sqrt{29{,}08} = 5{,}39$. En av årsakene kan være toppen i 30. time. Om dette skyldes en faktisk økning i bakgrunnsstråling eller en ustabilitet i systemet er vanskelig å si og det hadde vært lettere å konkludere dersom en hadde flere tellere som målte samtidig.

Vi gjorde flere tilsvarende målinger flere steder på skolen i 24 timer og observerte liknende resultater, det var ingenting som tyder på at det var en betydelig forskjell i bakgrunnsstråling mellom etasjene.

5.2 Geiger-Müller platå

15. april gjorde vi målinger på MR-laben på høyskolen. Hensikten var å se hvordan den ferdige kretsen ville oppføre seg med høyere stråledoser.

Vi gjorde målinger ved forskjellige spenninger i regionen å se om måleren får det flate platået vi forventer av et Geiger-Muller-rør. Vi plasserte en Cesium-137 kilde noen centimeter unna røret. Cesium-137 er en betakilde der noen henfall også vil sende ut ett gammafoton på 0,66 MeV, det er disse vi vil detektere.

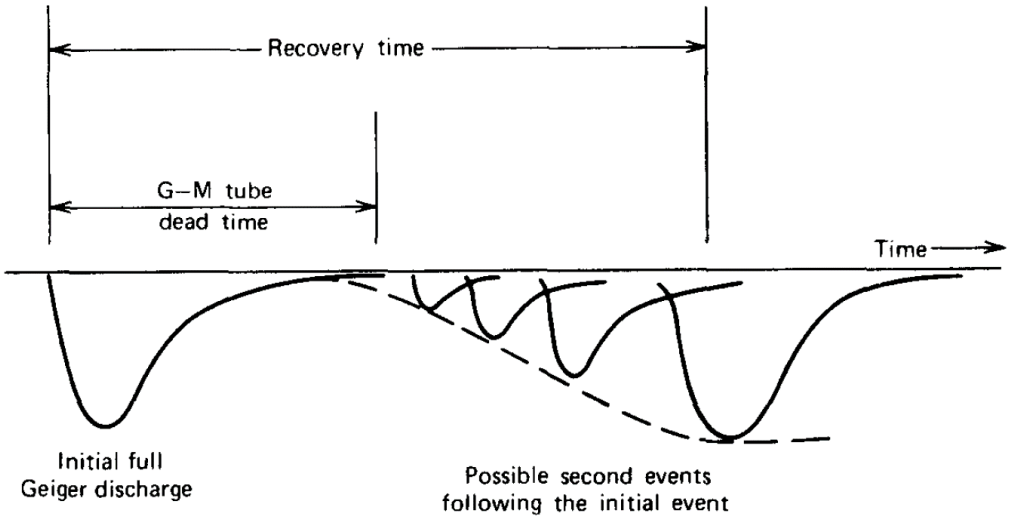


Figur 25 Platået til Centronic ZP1200

Grafen viser målingene gjort med forskjellig spenning i området 305 – 628 V. Når spenningen er under 380 V vil det elektriske feltet i gassen være for svakt til at skred vil kunne utløses. I regionen 400V – 550 V holder forholdet seg stabilt.

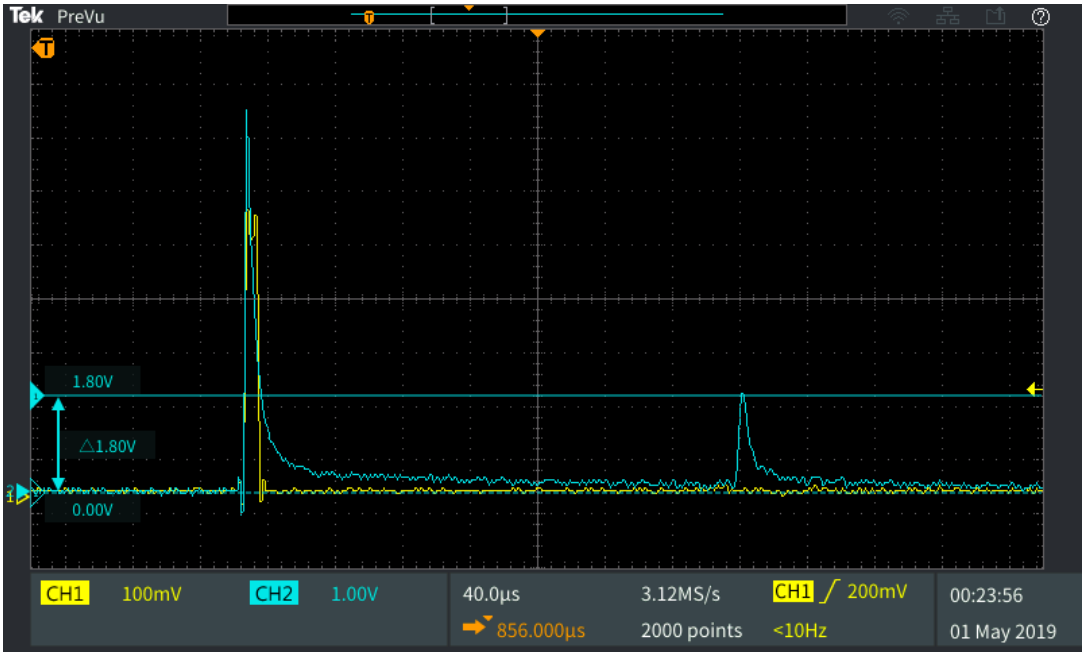
5.3 Dødtid

«Dead-time» eller dødtid er noe man må tenke på når man jobber med gassdetektorer. Dette er tiden det tar fra et foton blir detektert til systemet er i stand til å detektere et nytt. Selve røret har en dødtid, men for målinger er det mer interessant å snakke om den totale dødtiden til løsningen, altså summen av detektorens dødtid og dødtid som skyldes avlesningselektronikken.



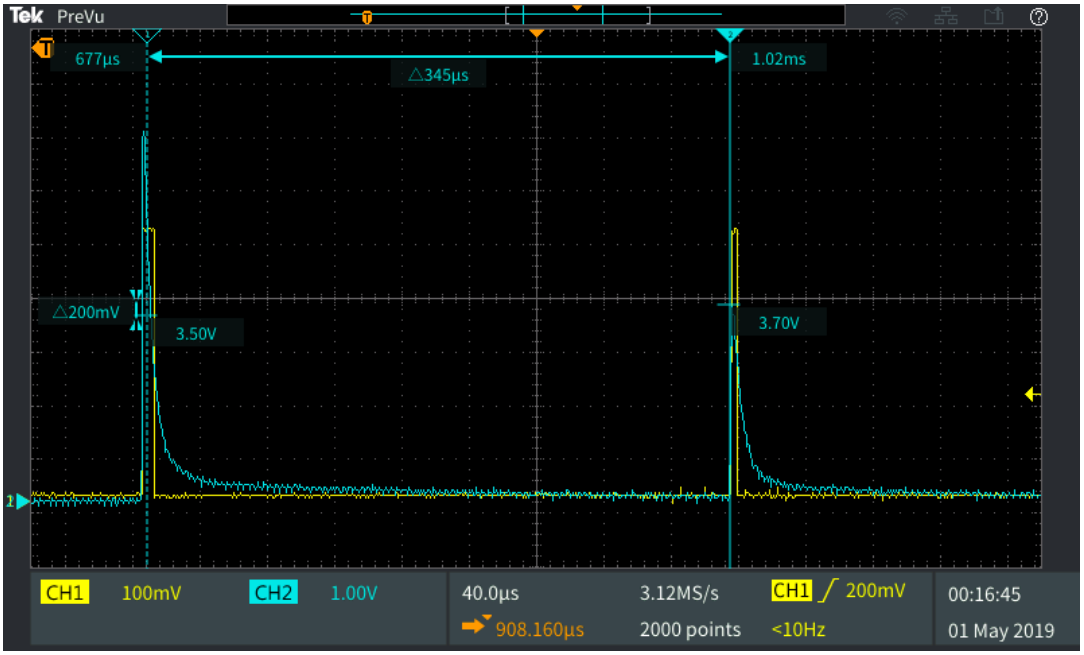
Figur 26 Illustrasjon av dødtid og gjenopprettelsestid i Geiger rør [3, p. 207]

Røret har ifølge dokumentasjonen en dødtid på 90 µs. Etter denne tiden vil røret være i stand til å sende ut en puls, men med lavere spenning enn den opprinnelige pulsen. Dette er fordi det trenger enda litt lengre tid for å gå tilbake til grunntilstanden. Dette kalles «Recovery-Time» eller gjenopprettelsestid. Den ferdige kretsen vil derfor ikke kunne detektere en ny puls før røret er i stand til å levere en puls over 2,5 V, som er terskelverdien på komparatoren. Dette fører også til at dersom det utløses ett nytt skred i tidsrommet mellom dette, vil målerens dødtid forlenges ytterligere.



Figur 27 To pulser genereres med så kort mellomrom at den ene ikke blir detektert

Figur 27 er en skjermdump av oscilloskop. Her klarte vi å fange en puls som forekommer i løpet av gjenopprettelsestiden til røret. Blå kanal viser spenningen ved inngangen til komparator, mens gul viser ved utgangen. Den første pulsen er over terskelverdien på 2,5 V og vil derfor bli registrert. Den andre pulsen er 1,80 V og vil dermed ikke bli registrert i det hele tatt. Denne pulsen ble generert 288 µs etter den opprinnelige puls.



Figur 28 To pulser genereres med stort nok mellomrom til at begge blir detektert

Vi klarte også å fange et lignende tilfelle i Figur 28. Her skjer pulsen også under gjenopprettelsestiden men denne gangen er pulsen 3,8 V og stor nok til å bli registrert. Den forekommer etter 345 µs. Vi vet derfor at systemet sin dødtid vil ligge et sted imellom 288 µs og 345 µs.

Poisson fordeling er en naturlig statistisk modell å bruke, siden det er snakk om antall ganger en hendelse inntreffer per tidsenhet og vi antar at en hendelsene er uavhengige av hverandre. Hvis vi antar bakgrunnsstrålingen *r* er 30 tellinger per minutt kan vi estimere sannsynligheten for at to hendelser inntreffer i løpet av et tidsintervall *t*. Sannsynligheten for at minst en hendelse vil inntreffe innen 300 µs ($5 * 10^{-6}$ min) blir dermed:

$$P(X > 0) = P(X = 0) = 1 - e^{-rt} = 1 - e^{-30*5*10^{-6}} = 0.14\ ‰$$

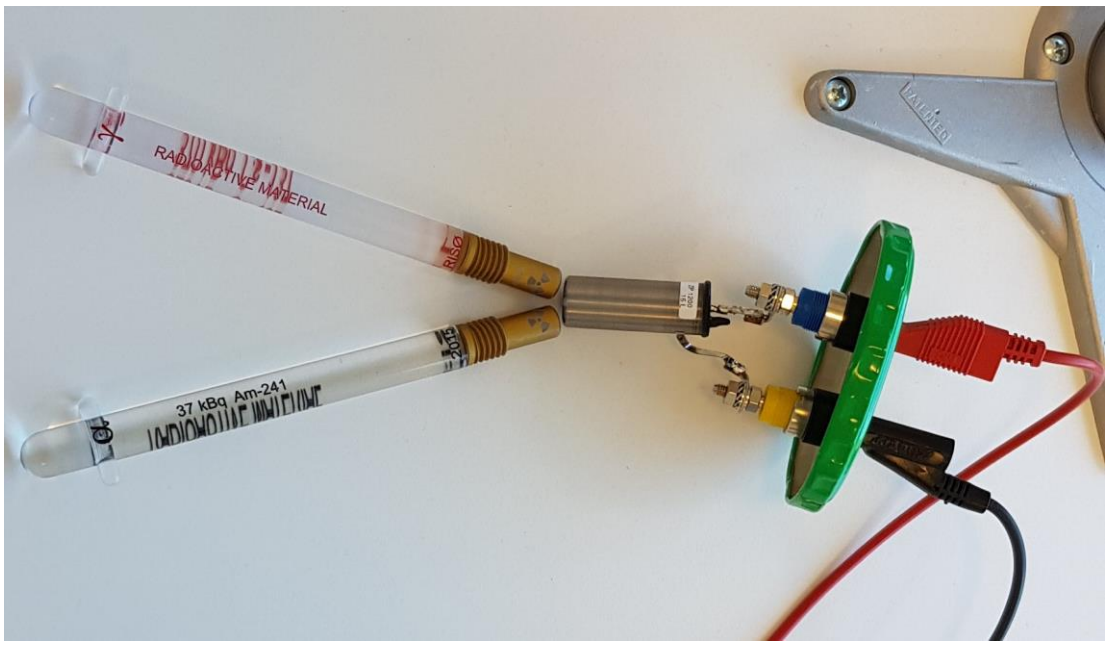
Ved lave stråledoser er det altså ikke en stor andel pulser man mister på grunn av dødtid. Ved høye tellerater derimot, vil det være nødvendig å senke terskelen lavere enn 2,5 V.

Dødtiden til et system kan estimeres også ved å måle telleraten til to kilder kombinert og deretter hver for seg. Dette krever at kildene har cirka samme aktivitet.

Dette gjelder om man antar at dødtiden ikke kan forlenges. Vi gjennomførte målinger for å beregne dette med kildene på radiograflaben.

Variabel	Kilde	Antall tellinger på 5 minutter
n_1	Cs-137a	55 754
n_2	Am-241	2 398
n_{12}	Cs-137a & Am-241	53 358
n_b	Bakgrunnsstråling	300

Resultatet ga oss lavere tellerate med 2 kilder enn med en kilde. Dette kan ikke brukes til å kalkulere dødtid. For å kalkulere dødtid burde man hatt omtrent like store kilder, noe vi ikke hadde tilgjengelig.



Figur 29 Måling med 2 kilder samtidig: Americium-241 og Cesium-137

6 Konklusjon

Systemet er en nokså primitiv løsning, men er i stand til å gjøre det som var målet med arbeidet. Den er i stand til å måle bakgrunnsstråling og den viste helt klart et utslag når vi økte stråledosen. 11 000 målinger per minutt med kilde i forhold til 30 per minutt uten kilde. Hvis vi godtar 30 tellinger per minutt som «normal» verdi for bakgrunnsstråling, kan vi bruke dette som referanse til å oppdage at bakgrunnsstrålingen øker.

Vi har arbeidet ganske greit innenfor rammene til den opprinnelige planen, men vi brukte mer tid på kretsen enn forventet. Dette gjorde at det ble mindre tid til målinger og dataanalyse. Dette er grunnen til at det er satt mest fokus på den tekniske løsningen i denne oppgaven.

6.1 Mulige utvidelser av kretsen

6.1.1 Bruk av en annen mikrokontroll

Klokkefrekvensen til *ATmega328* er på 16 MHz. Dette er høy nok frekvens til å kunne detektere bakgrunnsstråling med vår løsning som den er nå. Ved flere utvidelser og implementering av flere funksjoner kan dette være for lite. Med utvidelser mener vi funksjonalitet i detektoren som kan gjøre den mer robust og pålitelig for eksempel kontinuerlig overvåking av spenningen over Geiger-røret, temperatur, batterinivå, mulighet for varsling ved høyere stråling. En eventuell utvidelse til en IoT løsning ville også kreve mer strøm til kommunikasjon. Detektoren må kunne måle og sende data samtidig for eksempel hvert minutt. Dette kan løses ved å stoppe målingen, sende data og fortsette med målingen. Uansett hvor store utvidelsene måtte være kan det hende at *ATmega328* med sin frekvens på 16 MHz blir for lite. En god kandidat for å erstatte Arduino kan være en mikrokontroller fra ESP-familien for eksempel *ESP8266* som opererer på 80 MHz eller 160 MHz.

6.1.2 Designe og produsere egen PCB

Vi fikk dessverre ikke tid til å lodde sammen alle komponenter og lage et ferdig produkt. Løsningen vår er koblet ved hjelp av et koblingsbrett og ledninger. Dette kan være en kilde til uønsket oscillerende støy. Det er flere komponenter i kretsen som oscillerer som for eksempel DC-DC omformere. Oscillerende støy kan føre til ustabilt spenningsnivå på innganger til høyspenningsforsyningen og dette vil igjen føre til ustabil spenning over Geiger-røret. Vi kunne observere støy på ca. 200 mV på inngangen til komparator. Vi har forsøkt å dempe den ved hjelp av et enkelt lavpass filter. Dette ga ikke noe synlig resultat men årsaken kan være dårlig design av filteret eller behov for flere sånne filtrere i kretsen.

Det er flere verktøy tilgjengelig der man kan designe et PCB kort. Veldig populær er *Eagle* fra *Autodesk*. Ferdig design av kort sender man da til en av PCB produsentene. Komponenter på selve kortet må man lodde selv. Selve design og bestilling av PCB tar tid og må planlegges og bestilles god tid før man trenger det. I tillegg kan feil design føre til at man må bestille kortet en gang til. Uansett kan dette være en veldig utfordrende og lærerik erfaring, særlig for elektronikk studenter.

6.1.3 Redusere kapasitans i systemet

Lange ledninger kan øke kapasitans i systemet som kan føre til at Geiger-røret ikke fungerer stabilt. Det kan blant annet føre til lengre dødtid, forkorte levealderen til røret og forkorte Geiger platået. [2, p. 22]

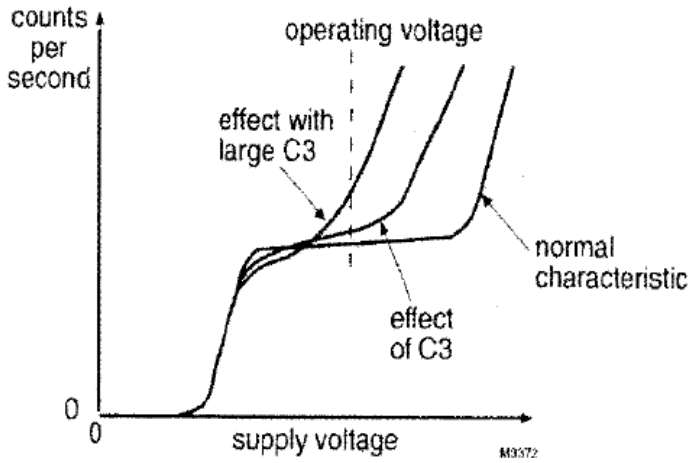


Figure 16 Effect of increased anode capacitance

Figur 30 Grafen viser resultatet av høyere kapasitans C3 mellom høyspenningskilden og anoden til Geiger-røret [2, p. 22]

6.1.4 Kontinuerlig overvåking og kontroll over høyspenning DC-DC omformer

Vår løsning har ikke mulighet å kunne kontrollere at det er riktig spenning både inn og ut fra DC-DC omformeren. Endringer i spenningen over Geiger-røret kan påvirke målinger og gi feil data. Vi har tenkt å bruke en analog inngang ved å bruke analogRead() til å sjekke at spenningen er innenfor en viss grense. Dette er en funksjon i Arduino biblioteket som gjør det mulig å kunne lese spenningsnivå på inngangen fra 0 – 5 V og presentere det om til et tall mellom 0 – 1023. Dette kan være problematisk fordi EMCO DC-DC omformer vi bruker er også avhengig av last på utgangen. Mest sannsynlig må man tilpasse spenningen på nytt etter man har lagt til flere motstander i systemet.

6.1.5 Schmitt trigger

For å få en kortere dødtid kan terskelverdien på komparator senkes. Problemet med å gjøre dette er at pulsen er flatere i dette området. Da kan man risikere at støyen vil svinge rundt terskelverdien slik at feil antall tellinger registreres.

En Schmitt-trigger er en komparatorkrets som er mer robust mot dette. Dersom løsningen vår brukes til å måle høy strålingsaktivitet vil dette være en bedre løsning enn vår enkle komparatorkrets.

6.1.6 «Active quenching» krets

Det er mulig å redusere dødtid ved at man kobler fra strømtilførselen til røret idet en puls registreres. Da vil det ikke skje noe ytterligere utladning og systemet vil raskere returnere til grunntilstanden. [8] Det finnes gode artikler og rapporter om «active quenching» for eksempel «A simple and efficient active quenching circuit for Geiger–Muller counters» av O. Vagle, Ø. Olsen og G.A. Johansen.

6.1.7 Av og på bryter

Mikrokontrolleren slås nå på ved at vi fysisk kobler ut batteriet. Dette er ikke ideelt og bør erstattes med en bryter.

6.1.8 IoT løsning

Dersom vi hadde hatt flere tellere kunne man ha spredt disse over et geografisk område. Da kunne man kartlagt bakgrunnsstrålingen. Telleren kunne jevnlig rapportert til en sentral som lagret dataene. Med flere tellere kan man også få bedre data til å trekke konklusjoner.

Referanser

[1] M. Komperød, E. Godske Friberg og A. L. Rudjord, «Stråledoser til befolkningen,» Statens strålevern, Østerås, 2015.

[2] Centronic, "Geiger Müller tubes," Centronic - ISS1, Croydon, UK.

[3] G. F. Knoll, Radiation Detection and Measurement, John Wiley & Sons Inc., 2000.

[4] I. Meric, G. A. Johansen, M. B. Holstad, A. F. Calderon and R. P. Gardner, "Enhancement of the intrinsic gamma-ray stopping efficiency of Geiger-Müller counters," Elsevier B.V., 2012.

[5] M. Lauritzen, «Project Report for PHYS117,» Universitetet i Bergen, Bergen, 2013.

[6] Arduino, «Arduino - Compare products,» [Internett]. Available: https://www.arduino.cc/en/products.compare. [Funnet 22. mai 2019].

[7] Arduino, «Arduino reference - pulseIn(),» [Internett]. Available: https://www.arduino.cc/reference/en/language/functions/advanced-io/pulsein/. [Funnet 30. april 2019].

[8] O. Vagle, Ø. Olsen og G.A. Johansen, «A simple and efficient active quenching circuit for Geiger–Muller counters,» Elsevier B.V., 2007.

[9] G. A Johansen, P. M Jackson, Radioisotope Gauges for Industrial Process Measurements, John Wiley & Sons, 2004.

Appendiks A Figurliste

Appendiks B Forkortelser og ordforklaringer

Geiger-rør	Geiger-Müller rør
UV-stråling	Ultrafiolett stråling
RTC	Real Time Clock
Li-Po	Lithium Polymer Battery
SPI	Serial Periphal Interface
IDE	Integrated Development Environment
GUI	Graphical User Interface
IoT	Internet of Things

Appendiks C Prosjektledelse og styring

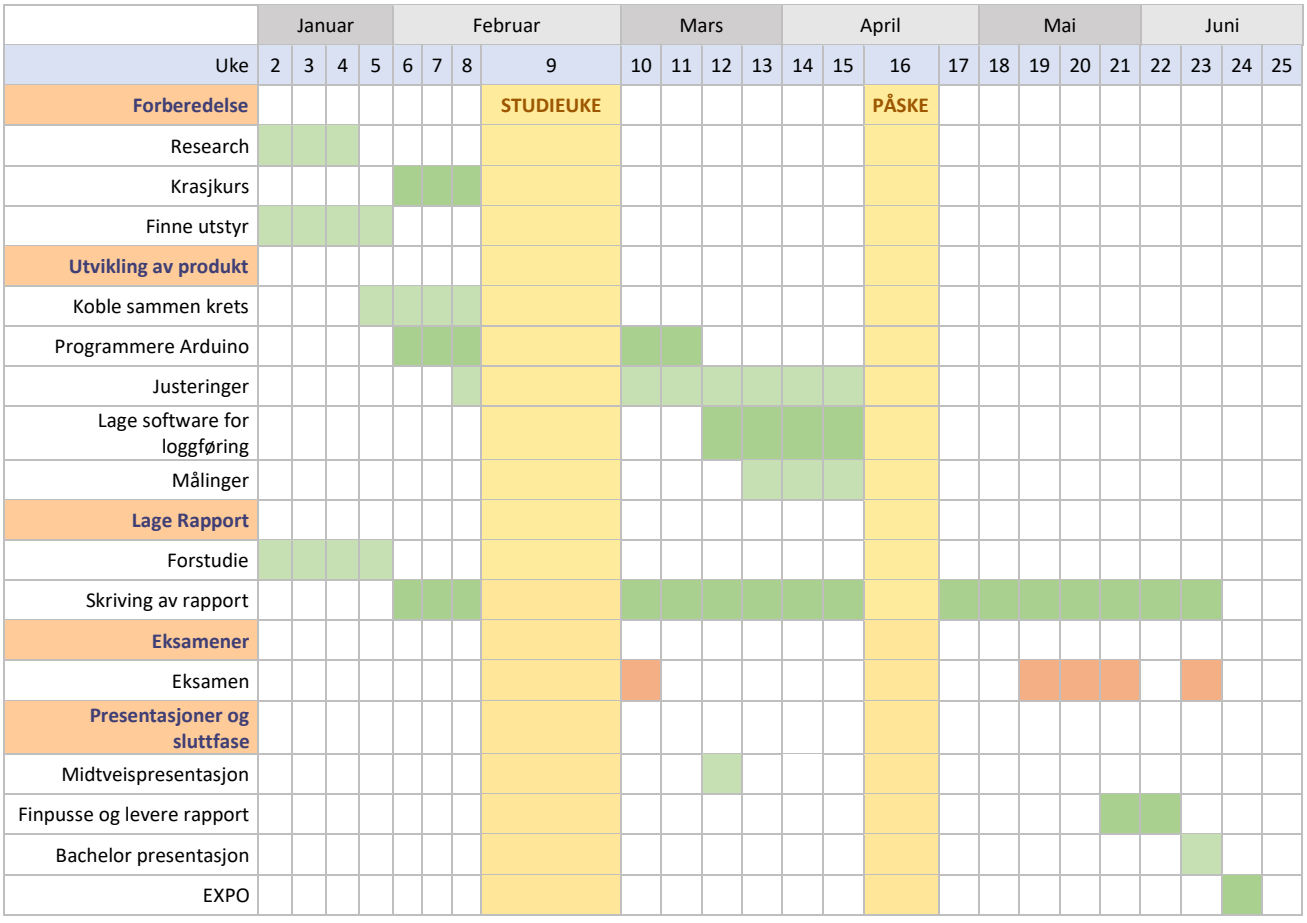
Vi har kommet i gang med arbeidet den 03. januar 2019 og da hadde vi også første møtet med en av våre veiledere, Ilker Meric. Vi har tenkt på mulige løsninger og har skrevet ned ideer og tanker.

C.1 Prosjektorganisasjon

Siden vi er kun to personer i gruppen så vi har ikke hatt noen spesiell rollefordeling underveis. Arbeidsoppgaver ble fordelt underveis. Vi har jobbet ofte begge to med samme oppgave samtidig, spesielt med programmering. For å organisere arbeidet har vi tatt i bruk *Microsoft Planner, Office365, SharePoint og GitHub.* Microsoft Planner var meget nyttig til å lage god oversikt over ting som må gjøres, er under arbeid og ble fullført.

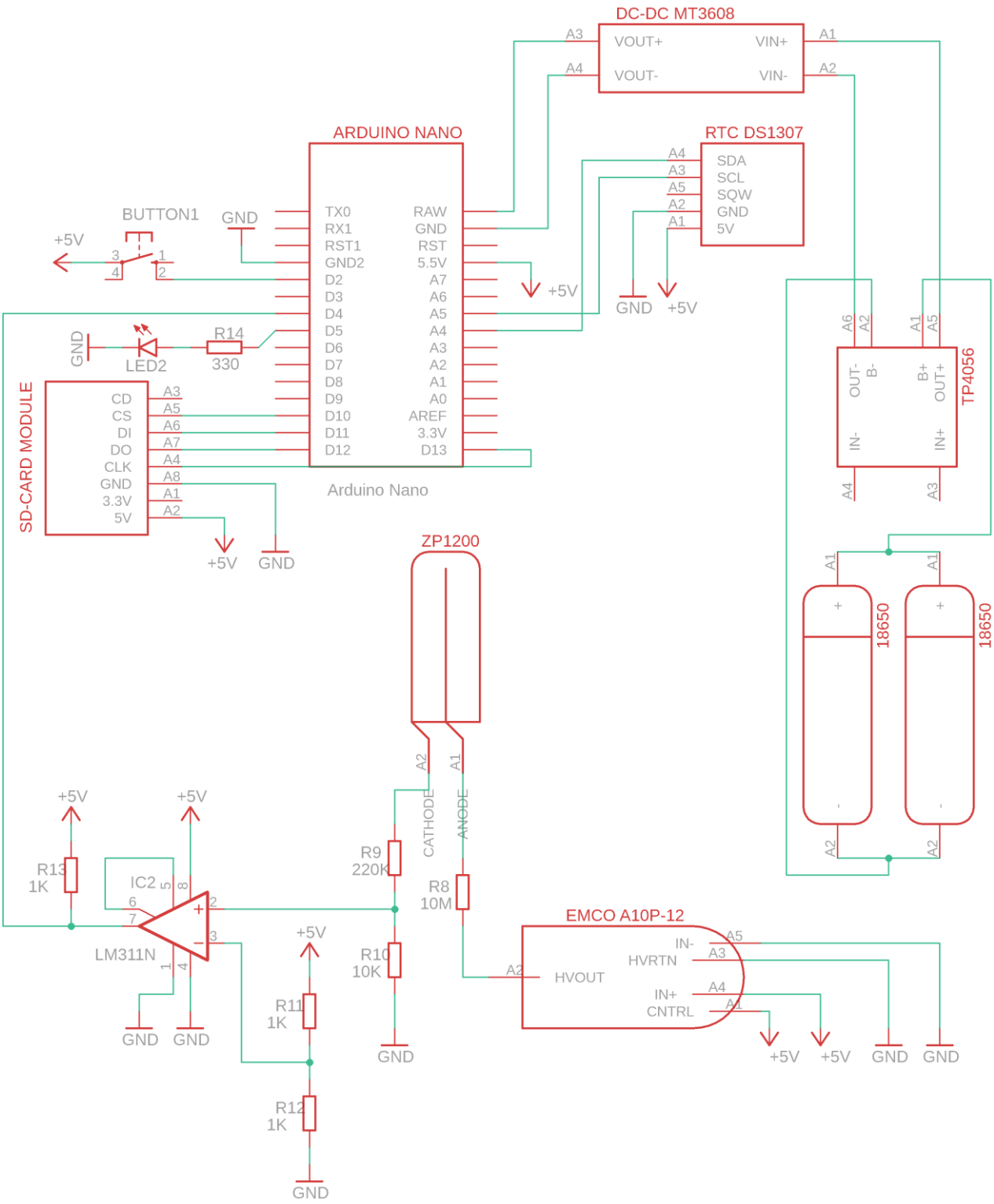
Vi har brukt *Visual Studio* til all skriving av programkode.

C.2 Fremdriftsplan



Figur 31 Fremdriftsplan

Appendiks D Kretstegning



Figur 32 Kretstegning

Appendiks E Kildekode

C# programmet: https://github.com/karbovski/bo19e34datalogger

Arduino kode: https://github.com/karbovski/bo19e34arduino

Appendiks F Dokumentasjon til komponenter

Dokumentasjonen til komponenter finner man i vedlagt mappe.

Appendiks G Komponent liste

Navn på komponenten	Produsent	Modell	Pris per enhet
Koblingsbrett	Ikke relevant		
Geiger-Müller rør	Centronic	ZP1200	~2000,00 NOK
Arduino Nano	Robotdyn	NANOV3	25,00 NOK
Batterier	VARICORE	18650	30,00 NOK
DC-DC høyspenning omformer	EMCO	A10P-12	750,00 NOK
DC-DC Step Up omformer	WAVGAT	MT3608	21,00 NOK
Lading komponent	WAVGAT	TP4056	17,00 NOK
Micro-SD kort leser til Arduino	Robotdyn	Ukjent	25,00 NOK
SD-kort	Kingston	16GB	50,00 NOK
RTC-klokke	Robotdyn	DS1307	12,00 NOK
Komparator	Texas Instruments	LM311	
Resistorer	Ikke relevant	1K	
		10K	
		220K	
		330	
		10M	
Button	Ikke relevant		
LED-diode	Ikke relevant		
		Totalt:	**2930,00 NOK**

Figur 33 Komponent liste