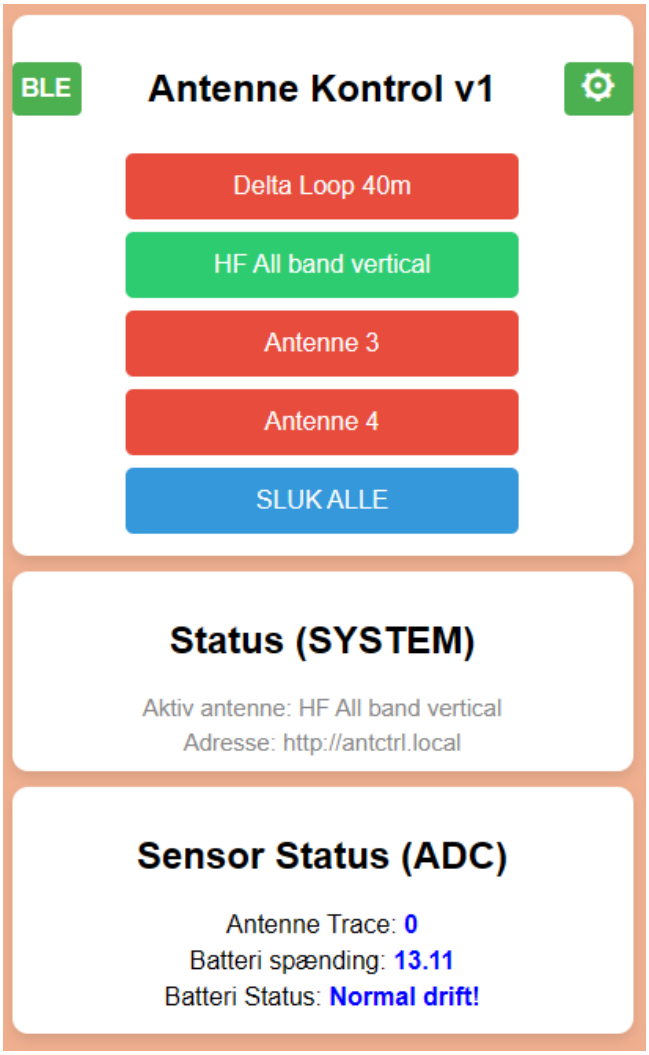
13-11-2026

Lav selv en antenneomskifter



Christian Knop
OZ1KLM

Lav selv en antenne omskifter helt fra start

Indholdsfortegnelse

Introduktion 2
Den indledende fase 2
Fra IDE til færdigt emne 3
IDÉ 3
Konkretisering 4
Antennestik 4
Relæer 4
Processor modul 5
1. Kritiske PSRAM og Flash pins (Må ALDRIG bruges) 5
2. Strapping Pins (Brug med stor forsigtighed) 6
3. Forbindelser til dine specifikke funktioner (Fastlagte pins) 6
Strømforsyning 6
Relæ komponenter 7
Antenne RF-trace kreds 7
Dataindsamling 8
Diagram 9
Print Layout 11
Oprettelse af produktions data for JLCPCB 12
Addon til projektet 13
Software for antennekontrollen 13
Opbygning af fjern betjeningen 13
Web-sidernes struktur 14
Forsiden (”/”) ROOT eller HOME 14
Her lidt om konfigurations siden 16
1. Tilføj pladsholdere i din PROGMEM HTML 16
2. Hent data fra NVS under opstart (**setup**) 17
3. Send HTML med NVS-data (**GET**) 18
4. Gem nye data i NVS ved **POST** 18
NVS hvad er det og hvad bruges det til 19
Forsiden af Antenne kontrollerens web 22
Opdatering af Firmware 24
Monterings vejledning 25
Stykliste 27
Component Liste 28
Afrunding 31

Lav selv en antenne omskifter helt fra start

Introduktion

Mange kender mig allerede for mangt og meget, men for en ordens skyld lige en lille introduktion til mig og hvorfor.

Navnet er Christian Knop, har gennem nu snart en menneskealder, på en eller anden måde altid været i berøring med noget der er elektrisk. Allerede som barn kom jeg i stødet bogstaveligt talt, og på trods af at jeg har prøvet en del gennem min tid, har min far gerne villet hjælpe mig til at lave noget, altså ikke gå og lalle den af, derfor fik jeg lidt lekture om elektronik, lavede små byggesæt, snakkede i Walki-Talkie. Far arbejdede selv med elektronik det meste af hans liv, og var også radioamatør, så æblet er ikke faldet særligt langt fra stammen. 1983 Startede jeg på min uddannelse som elektronikmekaniker, og tog i den forbindelse også min amatørradio licens. Jeg blev udlært i1987 på B&O i Struer og var ansat på B&O frem til 2004. Jeg har gennem de 20 år tilegnet mig rigtig megen viden om ikke bare elektronik, men også computere og netværk og senere har jeg også undervist i programmering, netværk og computerteknik. I dag er jeg så det vi kalder PHD d.v.s. Pensionist Hele Dagen, og har kastet mig over det ene projekt efter det andet, og nu er det altså en antenne omskifter.

Jeg har selvfølgelig lige som mange andre, også lavet alt muligt andre ting også nogle med 50års erfaring, men det er en helt anden historie.

Når jeg går i gang med et projekt, er det typisk fordi der er nogen der har fundet ud af, hvordan jeg skal gøres nysgerrig, så jeg ikke kan sige nej, men falder i gryden gang på gang.
Her kan nævnes:

- Startede med at sætte mig ind i DMR-systemet, og hjalp andre med deres radioer, skrev ikke mindre end to vejledninger som findes på OZ3EDR.dk hjemmesiden.
- Et løbende projekt med udgangspunkt i Krystalradio og fremstilling af en sådan, med sigte på skoleelever i udskolingen.
- Opsøgte en del information om Theremin, og dette endte med et selvbyg projekt til komplet instrument bestående af eget print layout, hjemme lavet og 3D printet kabinet samt tilhørende antenner.
- En morsetræner, som ikke er gjort helt færdigt grundet rigtig mange dumme undskyldninger.
- Nu! En antenne omskifter helt fra bunden med software, print og hvad der ellers hører til.

Denne gang har jeg så valgt at gribe tingene lidt anderledes an, da jeg egentlig vil forsøge at facilitere mine kollegaer til selv at lave et eller andet, ved at guide dem lidt i dette skriv. Om jeg så lykkes med dette, må tiden så vise.
Grundlaget for dette projekt kommer fra Carsten Grøn OZ9AAR som står bag det print som har været mit udgangspunkt, og Kristian Rasmussen OZ5KR som har lavet den software som jeg har taget ind i projektet og omskrevet plus lavet en del tilføjelser. Kristian og jeg vil være dem der står som kreatører af projektet, når vi frigiver det til eget brug.

--------===!===--------

Den indledende fase

Når nu man skal lave noget nyt, skal der jo mange tanker i spil, og her kan man godt komme lidt eller meget på glatis. Jeg har fået en IDE, men hvordan skal det se ud, hvad skal det kunne og hvordan skal det laves, altså rigtig mange 'hv' ord, ord der kan være svært at svare på, netop fordi man måske er god til at komme med ideer, men ikke selv har evnen til at bringe det til verden, det kan være det tager for lang tid eller skal bruges lige nu, og hvis det er nu man skal bruge det, så er der kun en vej, KØB, KØB, KØB som Mette siger. Men nej det er for nemt, det er meget mere spændende at lave det selv.

Derfor vil jeg tage dig med på en guidet tur, hvor vi skal runde valg af hvad skal vi lave, hvad skal vi bruge, hvordan gør vi så det.

Fra IDE til færdigt emne

Mange vil sige, lav en kravspecifikation, dette er en meget tung vej, med Review undervejs og meget andet, da jeg tror at jeg taber jer alle inden vi er i gang, ja måske er det alligevel allerede sket, så derfor bliver det en middel vej.

IDÉ

Vi har fået den Idé at lave et projekt for fremstilling af en antenne omskifter, ikke mekanisk som de fleste bruger, nej en der benytter relæer og med styring over Wi-Fi.

- Den skal have én indgang og fire udgange
- Vi drømmer om at den skal kunne klare 1KW, men lad os nu se hvor langt vi kommer med dette ønske.
- Det er lige nu at alle idéerne skal på bordet altså brainstorming, nogle idéer sies fra, andre markeres med 'need to have' og andre som 'nice to have', herefter vælges hvad vi egentlig vil tilstræbe os at lave.

Fjernstyringen skal være lavet med en lille Micro computer der har indbygget Wi-Fi og Blue-Tooth, og ønskerne listes her.

- Web interface som er enkel at bruge samt nemt at finde på nettet
- Non-blocking IO styring af relæer samt status.
- Vi har talt om analog måling også, så derfor bliver oplægget med to ADC-indgange og Non-blocking målinger også
- Mulighed for at kunne ændre på konfigurationen
  - Her tænkes navne på relæudgangene, tilknyttede IO porte.
  - Valg af relæ type (MONO-STABIL / BI-STABIL).
  - Mulighed for at kunne til eller fravælge blokering for relæskift, når senderen er aktiv.
  - Mulighed for at skifte IO for Relæer og andre IO der er valgt som en del af projektet, er fjernet igen i de seneste opdateringer.
    Det syntes som en god ide dengang vi startede på ide fasen.

    Vi har valgt nogle faste IO til følgende
    - 8 styk GPIO for håndtering af Bistabile og Monostabile relæer.
    - 1 styk GPIO indgang til RF-trace.
    - 1 styk GPIO indgang til måling af forsyning, så der kan udlæses en forsynings status, som , viser den påtrykte spænding på indgangsstikket, hvor der er taget højde for spændingsfald over indgangsdioden. Status tekst er baseret på niveauer for LiFePO4).
    - Derudover er der reserveret 2 styk GPIO til I2C, 4 styk GPIO til SPI for kommunikation med andre enheder.
  - Nye tiltag
    - Et ledigt område på print bruges til fumlebræt med et antal PADs (201 stk.)
    - Ledige GPIO føres ud til PADs så de kan tilsluttes
    - I2C og SPI føre ud til stik på bagsiden af printet
    - Power samt 4 ekstra pins tilsluttes via Phønix 6 pol blok med skruer.
    - Måling af RF-signal ved fødeledning indgang
    - Måling af forsyningsspænding.
    - Indføre mulighed for Bluetooth BLE når man arbejder offline
    - Indføre mulighed for opdatering af firmware over WiFi

Dette var de idéer som lige nu er på tegnebrættet, nu skal vi så have disse idéer bearbejdet med, hvad skal der så til for at indfri disse idéer.

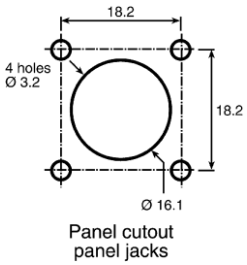
Konkretisering

Antennestik

Vi skal bruge 4 stik til antenne udgange og 1 stik til indgangen på antenneomskifteren, i alt 5 stik, og udgangs-punktet bliver SO-239 stik med 4 monterings huller i flangen, eller anden eks. N stik efter samme model.







Her skal bruges

- 5 styk SO-239 eller N stik
- 20 styk M3x5mm standoff
- Mindst 20 styk stjerne skiver eller fjederskiver anbefaler hele 40 styk.
- 20 styk M3 sekskant møtrikker
- 20 styk M3 skruer hvor lænden skal svare til 2,5mm + stik flange tykkelse + kabinet materiale tykkelsen

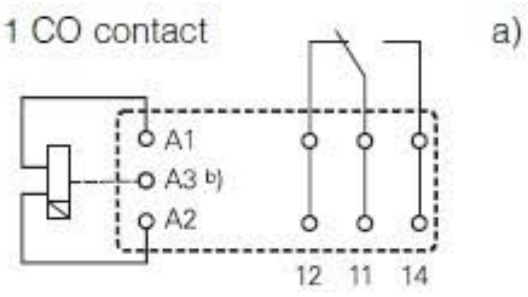
Relæer

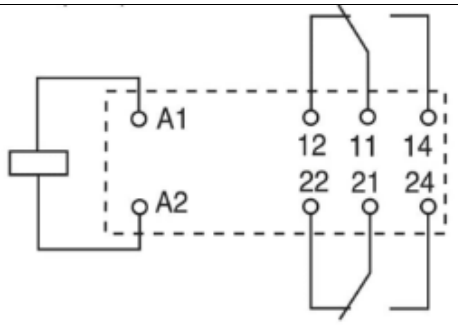


BISTABIL relæ RT314F12



MONOSTABIL relæ RT424012





Til hver antenne udgang skal der bruges et styk relæ, altså fire styk i alt, og disse har jeg så valgt skal være enten af typen MONO-STABIL eller BI-STABIL relæer og disse bliver af type SCHRACK POWER PCB RELAY, hvor den mono-stabile er en RT424012F, hvor F udgaven har en højere tilladt spole temperatur. Den bi-stabile relæ er en RT314F12 relæ. Disse relæer hører ikke til den billige klasse, men vil man det billigt, så er man også ligeglad med sin radio og meget andet.

Den store forskel ud over kontakt typen, hvor den bistabile har et kontaktsæt, og den monostabile har to kontakt sæt, så skal den bistabile bare pulses med max 50ms puls, hvor den monostabile skal have konstant signal.

Processor modul



Som processor modul er valget faldet på en YD-ESP32-S3-N16R8 Dev kit v1.4, den er bestykket med Wi-Fi BT og med mulighed for en ekstern antenne for bedre rækkevidde, desuden har den rimelig god processor kraft med op til 240MHz har 16MB flash samt 8MB RAM, ulempen ved den megen ekstra Flash og RAM er at det koster nogle af de IO pins der ellers ville være til rådighed, men i dette projekt går det nok også.

Rundt omkring dette modul skal som afkobling bruges

- 3 styk 1nF
- 3 styk 100 nF
- 2 styk 22 'pins pindhead sockets'

Af ledige GPIO kan listes: [1, 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 21, 38, 39, 40, 41, 42, 47] heraf har jeg så valgt at reservere dem som vist neden for:

- Relæ Set [7, 10, 16, 18]
- Relæ Reset [6, 9, 15, 17] bruges kun til BI-STABIL relæer
- ADC In 1 [4] Bruges til antenne signal sniffer (RF-Trace)
- ADC In 2 [5] Bruges til monitorering af indgangsforsyningen
- Reserveret SPI [11, 12, 13, 14]
- Reserveret I2C [2, 8]
- resterende ledige [1, 21, 38, 39, 40, 41, 42, 47] i alt 8 styk.

Hvis man ønsker at ændre på print layoutet skal man være opmærksom på at ændre i softwaren også.

Ønsker man ikke at gøre brug af I2C og SPI har man faktisk yderlige 6 GPIO til rådighed.

Ønsker man eks. flere indgange til ADC-målinger, så skal der lige rokeres rundt på signalerne og her skal bemærkes at vi har ADC kanal 1 og 2 at vælge mellem, men da Wi-Fi er i brug, har vi faktisk kun kanal 1 til rådighed dette betyder at vi kan bruge GPIO 1-10 til dette og i denne opstilling har jeg brugt dem til andre ting.

Her nogle meget vigtige informationer om de øvrige GPIOer på dette MCU board

1. Kritiske PSRAM og Flash pins (Må ALDRIG bruges)

Da dit modul er en **N16R8**-variant, bruger det **Octal (8-bit) PSRAM**. Det betyder, at chippen beslaglægger et udvidet antal pins til den interne hukommelse. Hvis du forbinder noget til disse, vil dit program crashe øjeblikkeligt.

- **GPIO 19 og 20**: Forbundet direkte til Native USB-C porten (bruges til USB CDC/JTAG).
- **GPIO 43 og 44**: Forbundet direkte til UART/COM-porten (via CH340-chippen til programmering og seriel monitor), Hardware Serial 0.
- **GPIO 26 til 32**: Optaget internt af **16MB Flash-hukommelsen**.
- **GPIO 35 til 37**: Optaget internt af **8 MB Octal PSRAM (R8)**.
  På almindelige ESP32-S3 uden Octal RAM kan nogle af disse bruges, men på dit N16R8-modul er de helt spæret.

2. Strapping Pins (Brug med stor forsigtighed)

Disse pins bestemmer, hvordan ESP32-S3 starter op (f.eks. om den skal gå i "bootloader/flashing mode" eller starte applikationen normalt). Hvis du trækker dem høj (3.3V) eller lav(GND) under opstart med eksternt kredsløb, vil modulet fejle i at boote:

- **GPIO 0**: Bruges til BOOT-knappen. Skal være høj (pull-up) under normal opstart.
- **GPIO 3**: Kontrollerer JTAG-debugging.
- **GPIO 45**: Bestemmer spændingen til Flash/PSRAM (VDD_SPI). Lad den være helt afbrudt.
- **GPIO 46**: Kontrollerer print af ROM-beskeder ved opstart og skal holdes lav (pull-down).

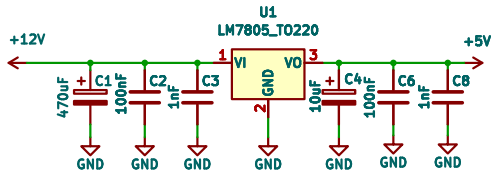
Tip: Du kan godt bruge f.eks. GPIO 0 som et input til en trykknap efter opstart, men du må ikke holde knappen nede, når du tænder for strømmen.

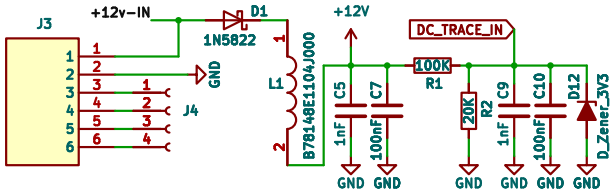
3. Forbindelser til dine specifikke funktioner (Fastlagte pins)

For at opfylde dine krav til onboard RGB, I2C og SPI uden at ramme konflikter, skal du tildele dem således:

- **Onboard RGB (NeoPixel):** Er på dette board altid hardwired til **GPIO 48**.
- **I2C (Hardware):** Brug **GPIO 8 (SDA)** og **GPIO 2 (SCL)**. Dette er de mest kompatible pins.
- **SPI (Hardware):** Brug **GPIO 11 (MOSI)**, **GPIO 12 SCK)**, **GPIO 13 (MISO)** og **GPIO 14 CS)**.
  - *Vigtigt:* Da disse fire SPI-pins ligger på **ADC2**, må du *ikke* bruge dem som analoge indgange, hvis du har Wi-Fi tændt. Men som digital SPI-bus fungerer de perfekt og lynhurtigt, uanset Wi-Fi!

Strømforsyning





Der skal bruges 12v DC til dette projekt, og her er der så behov for et indgangsfilter for blokering af HF-tilbageslag samt diode spærring for fejlstrøm. Dertil skal også bruges en 5v DC-forsyning til processor modulet og denne spænding laves fra de 12v.

Her er der også tilføjet en spændings deler til brug for monitoreringen af forsyningen.

Senest er der tilføjet en 3,3v zener diode til beskyttelse af indgangen for forsynings måling nemlig D12.

Dette betyder at her skal vi bruge følgende

- Et indgangsstik (Phønix 6 pol blok med skruer) for
  - 12v power
  - 4 ledige I/O
- En 100uH filter spole
- Gerne en sikring på 12v forsyningen Droppet i denne omgang
- En 1N5822 indgangs diode i serie med forsyningen for forkert polariseret spænding
- En LM7805 til 5v forsyningen
- Diverse afkoblings kondensatorer på forsyning og filter samt diverse andre steder for at komme højfrekvent signal fejl i forkøbet, typisk 1nF og 100nF, dertil kommer 470uF lyt, 10uF tantal
- En spændingsdeler bestående af en 100K og en 20K modstand, samt afkobling.

Lav selv en antenne omskifter helt fra start

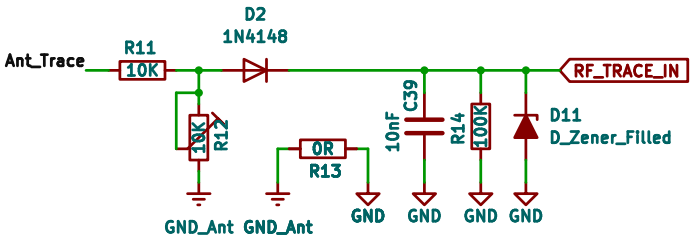
Relæ komponenter

Ud over et relæ af typen BISTABIL eller MONOSTABIL, skal der bruges følgende

- For et styk BISTABIL relæ, skal bruges
  - 2 styk flyback dioder af typen 1N5819
  - 2 styk BS170 FET-transistorer
  - 2 styk 1K modstande som gate modstande, til at sikre mod utilsigtet utidigt on/off af relæet
  - 2 styk 1nF til afkobling af 'gate' mod RF
  - 2 styk 100nF til afkobling af gate mod RF
  - 2 styk 10nF afkobling af 'drain' mod RF
  - 1 styk Jumper for elektrisk indkobling af Reset kredsløbet, er tilføjet for at undgå problemer.
- For et styk MONOSTABIL relæ, skal bruges
  - 1 styk flyback dioder af typen 1N5819
  - 1 styk BS170 FET-transistorer
  - 1 styk 1K modstande som gate modstande, til at sikre mod utilsigtet utidigt on/off af relæet
  - 1 styk 1nF til afkobling af 'gate' mod RF
  - 1 styk 100nF til afkobling af gate mod RF
  - 1 styk 10nF afkobling af 'drain' mod RF
  - 1 styk Jumper for elektrisk indkobling af forsyning på relæ, er indført for at undgå problemer.

Da der er 4 styk relæ kredse så skal der bruges 4 gange de komponenter der er beskrevet ovenfor, altså for BISTABIL kreds 8 styk af hver komponent, og for MONOSTABIL kreds skal bruges 4 styk af hver komponent

Antenne RF-trace kreds



Her laves to signal baner i printet ved antenne føde indgang, den ene direkte på RF-indgangen på top laget i printet, den anden som en lille tynd bane på laget lige under, til oplysning kan fortælles at der er fire lag i printet.

Den lille bane inde i printet kaldes Ant_Trace og signaler tappes ud over en 10K modstand, og med en 10K trimmer til stel for signal justering således at vi lige akkurat kan måle de 5W ved SSB, herfra tappes signalet ud og ensrettes med en 1N4148 traditionel små signals diode og ind over en 10nF kondensator parallelt med både en 100K modstand og en spændings begrænser (3,3v zener diode), herfra videre til ADC-indgangen på processoren. Bemærk at der er vist en 0R modstand, er bare en lus mellem de to stel punkter af hensyn til måling som ikke bare er svævende med meget støj.

Stykliste for denne kreds:

- 1 styk 10K
- 1 styk 10K trimmer
- 1 styk 1N4148
- 1 styk 10nF
- 1 styk 100K
- 1 styk 3,3v Zener diode

--------===!===--------

Til alt dette er der lavet dataindsamling som følger

Dataindsamling

Vi starter med at fremskaffe de datablade der er nyttige i forbindelse med denne konstruktion. Hvad ved vi at vi skal bruge?

- ☑ Relæer, udgangspunkt to slags MONO-STABIL RT424012F og BI-STABIL RT314F12
- ☑ Antenne stik SO239
- ☑ M3 x 5 mm standoff samt M3 møtrikker, sæt af 4 pr. SO239 stik, i alt 20 sæt
- ☑ DC-stik for 12v tilslutning, ændret til Phønix stik med 6 POL
- ☑ 100uH indgangsfilter, her skal vi nok lige have kigget på om det filter jeg har, er god nok (MAX 630mA)
- ☑ 5v DC strømforsyning, LM7805
- ☑ 1N5822 diode i serie med indgangs spændingen, for at sikre mod fejl polarisering.
- ☑ MCU YD-ESP32-S3 N16R8, her var der lidt udfordringer, hvor mange GPIOs kan faktisk bruges og hvilke kan absolut ikke bruges, tilmed er der mange versioner af dette board med meget forskellige layout. Så her skal man faktisk være meget klar i spyttet, ellers får man de forkerte informationer.
  Jeg har været meget langt i softwareudviklingen samt print layout, da jeg løb ind i en gamechanger. Jeg arbejdede med de forkerte informationer, dette er der nu rettet op på.
- ☑ FET af typen BS170, transistorer der skal drive relæerne, skal kunne klare opgaven uden de store problemer derfor dette valg.
- ☑ 1N5819 dioder. Almen viden for styring af relæer er at der SKAL være flyback dioder på relæerne, da der ellers med sikkerhed bliver ødelagt elektronik. Her har jeg så valgt 1N5819.
- ☑ Da hele dette opsæt skal kunne arbejde i et kraftigt HF-miljø, har der skullet en del flere kondensatorer i kredsløbet en der lige var regnet med fra starten, og om jeg er helt i hus, vides p.t. ikke men der skulle også lige et indgangsfilter i strømforsyningen.
- ☑ Da vi også ønsker at lave måling af RF-signal, samt forsyningsspænding, har jeg også indhentet informationer om dette, samt software løsninger til selvsamme.
- ☑ Der blev også ytret ønsker om måske at kunne måle spændingen på strømforsyningen, er dette også behandlet, og løsning implementeret.
- ☑ Da vi også har rundet det at kunne bruge denne løsning offline i forbindelse med Field Day og Florafauna, har vi nu også valgt at implementere Bluetooth BLE kontrol svarende til den oprindelige styring, dog kun med drifts håndtering, alt andet foregår stadig via WiFi.
  - Omkring Bluetooth BLE skal det lige bemærkes at, der er et minimums krav til de enheder der skal kunne snakke med Antenne Kontrolleren, som er version 5.0, dette kan have betydning for eventuelle Windows 10 installationer af ældre oprindelse.
  - På Android kan man IKKE bruge standard web browseren medmindre det er en Google browser.
  - På Apple produkter skal der bruges en anden browser i stedet for Safari, nemlig Bluefy eller WebBle som kan hentes gratis i App Store. Dette skyldes at OS er totalt lukket af sikkerheds mæssige grunde.
  - Dette har jo medført at der skulle laves en ekstra webkode til offline brug som understøtter BLE, så denne kode ligger nu på en github.com side hvorfra opdatering af denne også styres
- ☑ Efter at have arbejdet utrolig meget med udvikling af dette produkt, og talrige opdateringer via kabel blev nærmest en nødvendighed at skulle kunne opdatere firmwaren via WiFi, så dette er også blevet løst.

Diagram

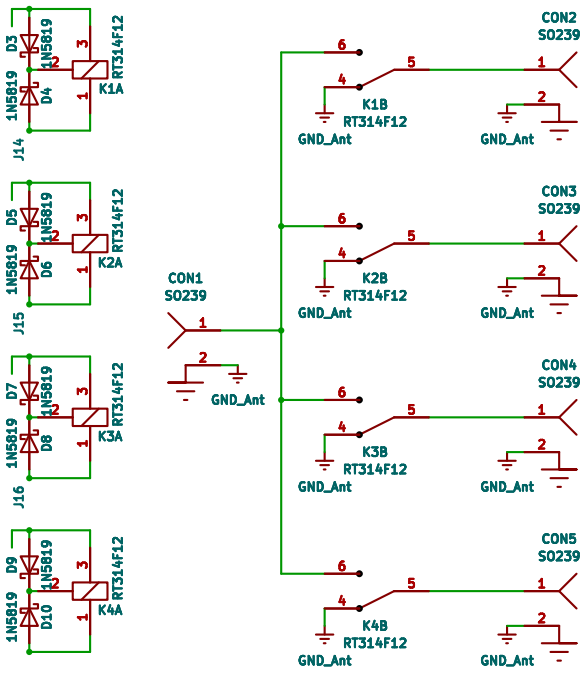
For at begynde på et diagram, kan man jo selvfølgelig bare gå i gang for derefter af finde ud af, at der mangler da både det ene og det andet.

Jeg har besluttet at ikke standard komponenter og lidt mere lægges ind i projektets egne biblioteker og dermed frigøres fra KiCad installationens biblioteker. Dette indbefatter diagramsymboler, footprints, og 3D objekter

Og er man hjemmevant i KiCad er det jo en smal sag at håndtere oprettelse og vedligeholdelse af netop det lokale bibliotek, som jeg har navngivet 'OZ1KLM_OZ5KR'

De forskellige komponentsymboler m.m. hentes bl.a. via https://componentsearchengine.com/

Med komponentdatabasen på plads, så laver vi diagrammet og her starter jeg med relæerne samt antennestikkene



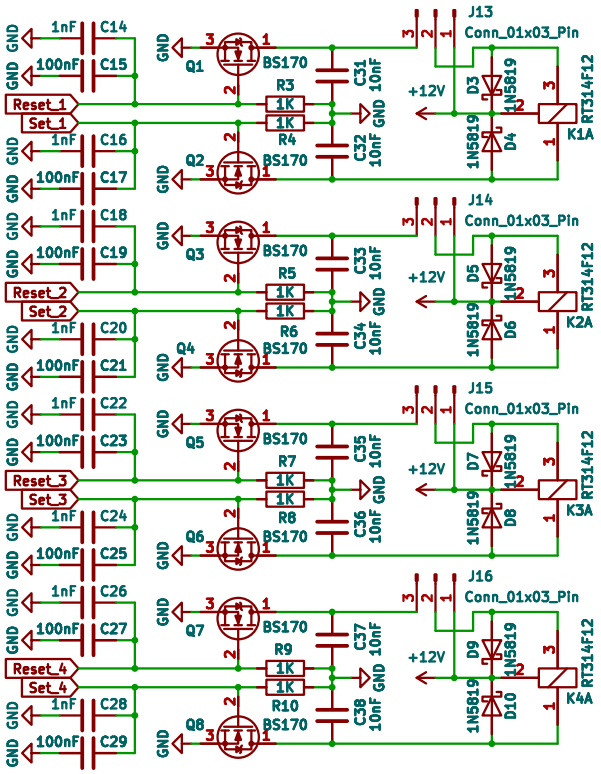
Her har jeg startet med at placere SO239 stikkene og lagt stel til GND_Ant for ikke at blande HF og digitale signalerne sammen.

Relæ symbolerne består af 2 dele en A og en B del. A delen er relæspolerne og B delene er relækontakterne.

På relæspolerne er monteret flyback dioder for at beskytte elektronikken mod de ødelæggende signaler fra relæerne når de afbrydes og sender en omvendt polariseret spænding retur.

Hvis der vælges MONO-STABIL relæer undlades D3, D5, D7, D9 og jumperne flyttes fra default position, de øvrige komponenter for reset kredsløbet udelades også, d.v.s. Q1, Q3, Q5 og Q7 samt C31, C33, C35 og C37 undlades da de ikke bruges.

De resterende komponenter for RESET 1, RESET 2, RESET 3 og RESET 4 skal stadig monteres af hensyn til ESP32-S3 computeren, som stadig har fat i disse.



Her er relæstyringens FET-transistorer med Set og Reset signalerne, samt en del kondensatorer for afkobling af HF. I seneste udgave, er der tilføjet Jumpere i Reset kredsløbet, til omgåelse af mulige problemer.

Kredsløbet som grundlæggende består af et relæ pr. antenne udgang og en til to Field Effekt Transistorer (FET) af typen BS170 suppleret med en til to flyback dioder af typen 1N5819 til at sikre transistoren med tilbageslags spændingen fra relæspolen som jo er omvendt polariseret.

Der er brugt t 1K modstande til at sikre mod utilsigtede støj signaler på transistorernes gate. Derudover er der brugt en del afkoblings kondensatorer for utilsigtede HF/RF-signaler.

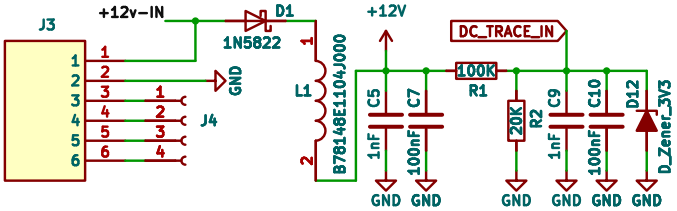
Lav selv en antenne omskifter helt fra start

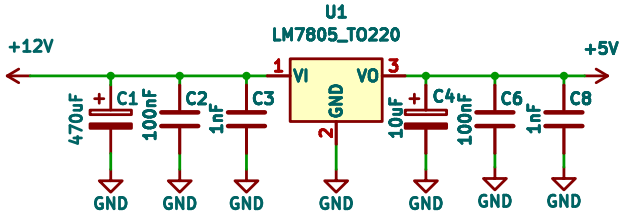
Funktion: BISTABIL relæ pulses med max 50ms pulse på henholdsvis Set og Reset indgangene for at skifte mellem ON/OFF hvilket betyder at selv om strømmen afbrydes, så er der stadig forbindelse gennem relæet, og dette huskes også i programmet.

Funktion: MONOSTABIL relæ skal konstant have spænding over relæets spole for at der er signal igennem relæet d.v.s. at relæet er ON, hvis spændingen forsvinder, da afbrydes alle antenner.

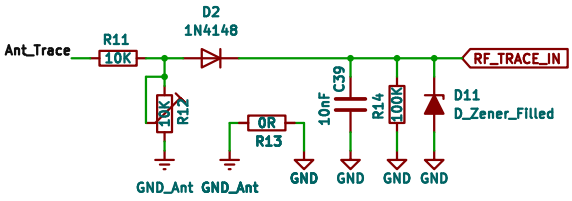
Der har været lavet diagram og printudlæg hvor der også var et kredsløb til at kontrollere for om relæ kredsløbet stadig fungerer, men det er blevet fravalgt.

Strømforsyning 12V DC og 5V DC laves således

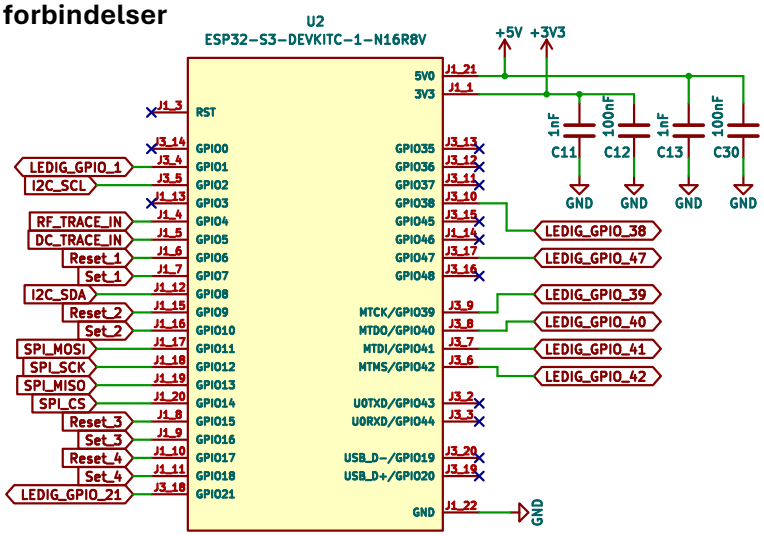




Her har vi RF trace kredsløbet, som måler RF signalet fra ca. 5W til 1000W SSB, signalet bruges til at kunne blokere for relæskift under TX sending.

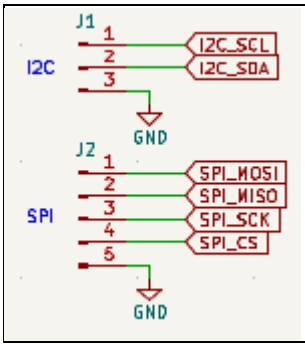
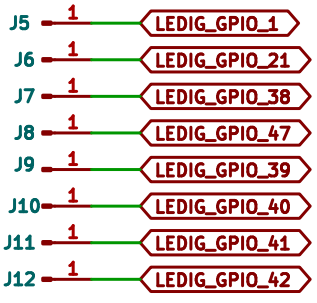


Her har vi ESP32-S3 N16R8 modulet med afkoblinger og forbindelser



Jeg har her i dette diagram valgt at benytte mig af Etiketter for signalerne, dette skyldes at jeg vil forenkle diagrammet således at det bliver nemmere at finde uf af hvor de forskellige signaler skal hen, i stedet for at bruge rigtig megen tid på at føre signaler eller signal-busser rundt på tegningen i et tæt edderkoppespind, hvor overblikket meget hurtigt mistes. Håber at det er lykkes at forenkle dette tilstrækkeligt.

Efter ønske har jeg også lavet således at de GPIO der ikke er i brug trækkes ud og gøres tilgængelige for nye ting, tilmed er I2C og SPI-signalerne også trukket ud til et stik

	Disse stik er placeret på bagsiden af printet under processoren
	Disse signaler er trukket ud for enden af processor printet sammen med et lille PAD grid fumle område med p.t. 201 huller med lodde ø.

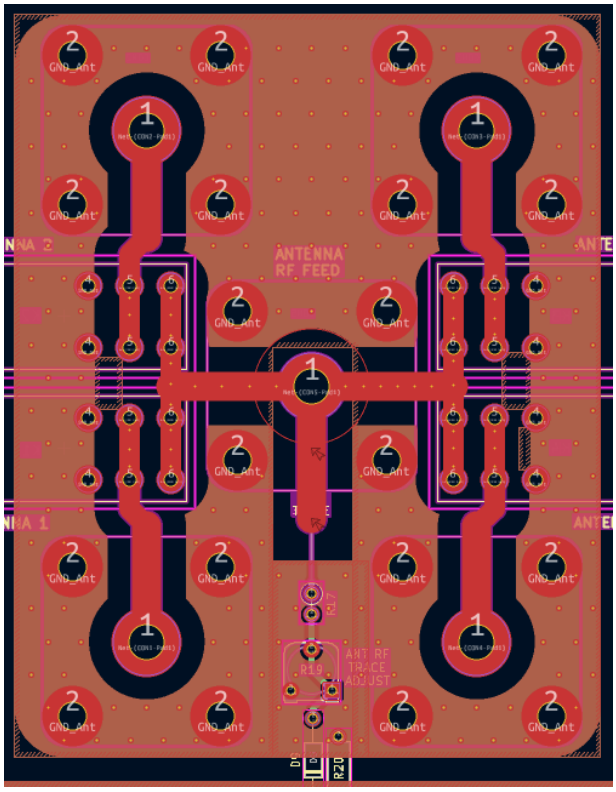
Print Layout

Grund printet er baseret på det som Carsten Grøn har lavet, bare med nogle ændringer her og der.

Det oprindelige print var lavet med de fire udgange via SO-239 stik men indgangen var lavet med et antenne-kabel loddet direkte på printet, dette har jeg ændret til endnu et SO-239 stik. Relæerne er også skiftet samt flyback dioderne, da disse var af SMD typen. Styringen var ikke en del af det oprindelige print, men dette er det blevet nu.

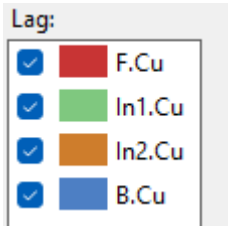
Måske er det i dette projekt ikke det helt rigtige at gøre, på grund af den øgede HF-indstråling på printet, der nødvendigvis måtte være resultatet af dette, med håber at tilstrækkeligt med afkoblinger kan afhjælpe eventuelle problemer.

Her set udsnittet hvor stikkene er placeret og relæernes kontakter



Først har jeg placeret disse 5 stik, og de 4 relæer, så har jeg lavet kobberforbindelserne mellem relæerne og stikkenes center pin og lavet dette helt ens på alle fire lag i dette 4 lags print.

Så har jeg lavet en kobberzone som er det der ses her og lagt den på det net der hedder GND_Ant, og er defineret som antenne stel under samme navn i diagrammet, zonen

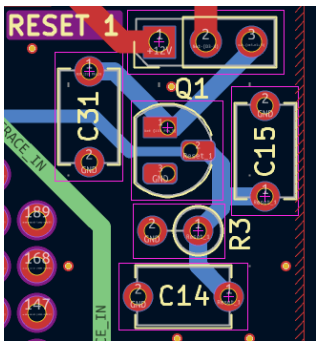


vælges til at dække alle 4 lag samt have en hjørneradius på 6mm, Afstand: 3mm, Minimum bredde: 0,25mm, Padforbindelser: Massiv, Termiske parametre: 0,5mm, Hjørneudjævning: Filet, Fjern øer: Altid

For at modvirke afstandskravet på 3mm ved RF-signal tracer har jeg lavet en ny zone med standard afstande i en størrelse der tilstrækkeligt dækker det område som hoved-zonen ellers laver fjernelse af kobber.

Derudover er der lavet nogle undtagelser for at fjerne nogle utilsigtede spidser i kobberet.

For at lave en mere kompakt stel har jeg spredt en masse vias ud over hele printet, de ligger med 4mm afstand på den vandrette linje og med 4mm mellem linjerne med en forskydning på 2mm til siden.



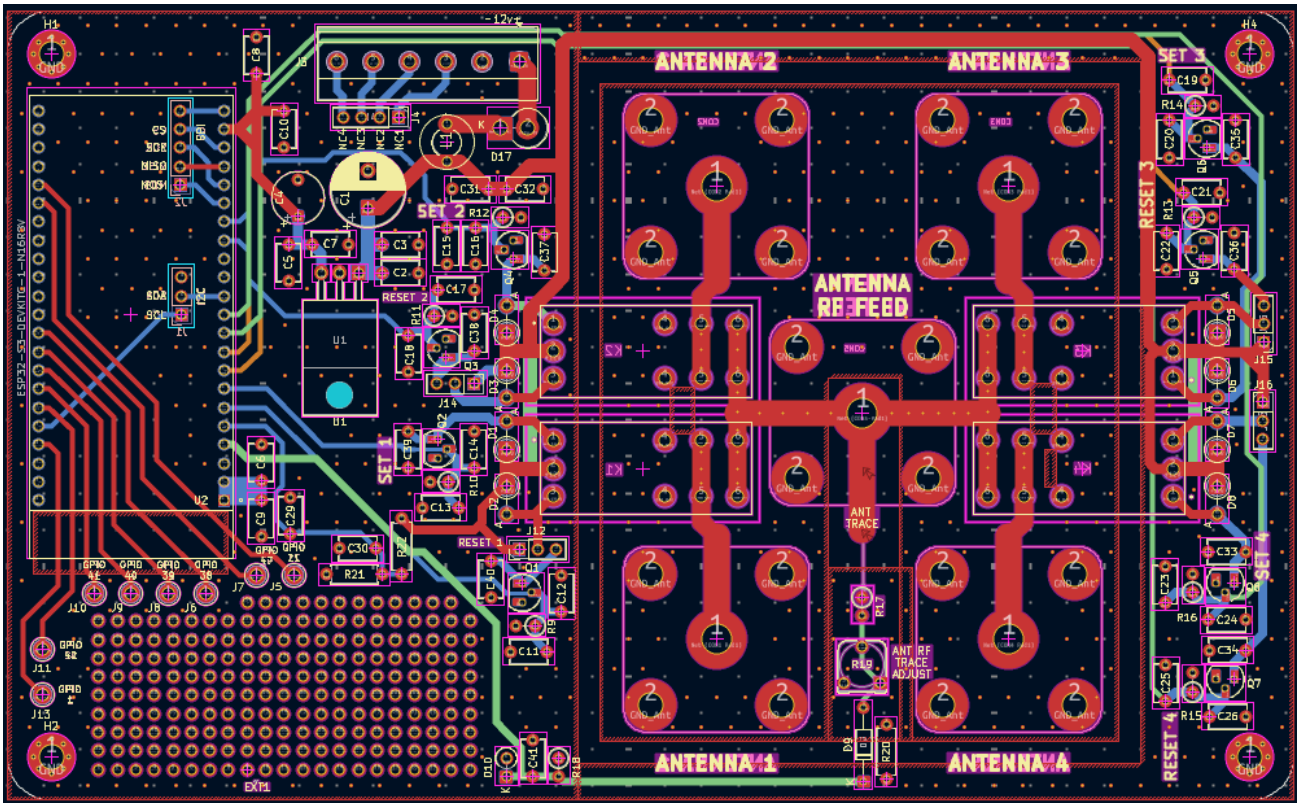
Her vises et af otte, styrings kredsløb for en relæspole, i dette tilfælde reset for relæ 1

her i toppen har vi FET-transistoren som er forbundet til relæspolen med en lodret stående 1K samt tre styk afkoblings kondensatorer for at minimere HF indflydelse

Om der skal afskærmning til vides ikke p.t. men vi tror på at det nok skal gå i dette kredsløb.

Lav selv en antenne omskifter helt fra start

Her ses printet i sin helhed uden ZONE fyldning



Printets dimensioner er 176mm langt og 108mm bredt, hvis det syntes for stort, så skal det deles op i to én med styring og én med relæer og antennestik, selvfølgelig en mulighed, men lad os nu lige se hvad stemningen er.

Tirsdag aften den 18. august 2026 blev vi enige om at sende printet i produktion for første gang, prisen for de første print er ca. 103,40 kr. pr. styk, altså ca. 517 kr. for 5 print inclusive afgifter og forsendelse.

Oprettelse af produktions data for JLCPCB

For at lave produktionsdata af et PCB hos eks. JLCPCB er der flere veje

1. Den tunge vej, skabe sig et overblik over hvad skal jeg bruge og hvordan m.v.
2. Installere plugin: **Fabrication Toolkit**
3. Installere plugin: **KiKit**

Jeg har faktisk begge plugin installeret, men jeg bruger KiKit til mine produktionsprint. Jeg laver både direkte produktions data, og paneler af mindre print forud for produktionsdataene.

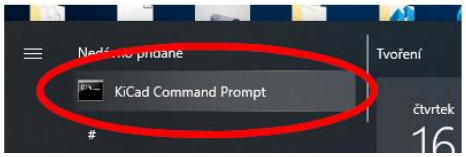
Hvordan installeres denne plugin, umiddelbart kan den installeres fra plugin manageren men den virker ikke, du skal installere via KiCad kommando konsol med pip, vejledning findes under følgende link:

homepage: https://github.com/yaqwsx/KiKit, og her finder du link til:
https://yaqwsx.github.io/KiKit/latest/installation/intro/, igen her findes følgende

- Linux guide
- Windows guide
- macOS guide

Installation on Windows

To install KiKit on Windows, you have to open "KiCAD Command Prompt". You can find it in the start menu:



udfør følgende: pip install kikit
for teste for OK: kikit –help
hvis denne fejler, luk konsollen og start en ny prøv så igen. Husk herefter at installere GUI interfacet i PCM

Hvordan kommer vi så frem til produktions data, her SKAL du være sikker på at der ikke er ubehandlede fejl både under update PCB fra schematic, samt DRC, når alt er OK, så laver vi de endelige data.
Grund vejledning er: **kikit fab jlcpcb [OPTIONS] BOARD OUTPUTDIR**
Kommandoen for dette projekt er: (**BEMÆRK** de tre linjer nedenfor skal indtastes/indsættes som én linje)

```
kikit fab jlcpcb --autoname
"C:\PCB_Produktion\Antenne_Switch_1KLM_5KR_August_2026_V5.3\Antenne_Switch_1KLM_5KR_August_2026_V5.3.kicad_pcb" "C:\PCB_Produktion\Antenne_Switch_1KLM_5KR_August_2026_V5.3\Production"
```

Dette betyder at der skal laves fabrikations data til **JLCPCB**, og hvor PCB filen og dens navn findes, samt hvor den skal afleveres, nemlig i et biblioteket **Production** inde i projektet.

Addon til projektet

Jeg har fremstillet nogle metalskærme til antenneområdet, de er lavet i fire dele, og for at kunne genskabe disse, har jeg fremstillet 3D printet plast skabeloner til opmærkning af huller og bukkelinjer samt klippe linjer.

Disse skabeloner er lavet specielt til nogle hvidblik plader fundet på klubbens lager, og kan ikke umiddelbart bruges til andre dimensioner af plader, men det løser jeg nok senere. Filerne til 3D print er endnu ikke frigivet.

Der er også fremstillet et 3D printet kabinet til prototypen, hvor tegningerne skal tilrettes den nye produktions udgave af printpladen.

Software for antennekontrollen

Opbygning af fjern betjeningen

Fjernbetjeningen er bestemt til at være en webside hvor alt fra konfiguration af både WiFi og konstruktion til styring af relæerne og aflæsning af status sker fra forskellige sider bestående af en ”ROOT (/)” også kaldet home, og dertil et antal under sider med bl.a. konfiguration, derom senere.

På forsiden af dette skriv kan man finde et billede som er et eksempel på den fjernbetjening man møder, når man opretter forbindelse til relæboksen.

Der er lavet flere sider og undersider på dette interface og disse er i skrivende stund:

- Forside som også kaldes ROOT (”/”)
- Konfigurations side kaldet CONFIG (”/config”)
- Informations side kaldet STATUS (”/status”)
- En side med funktionen GEM (”/save”)
- En side med funktionen nulstil konfiguration (”/resetNVS-Conf”)
- En side med funktionen nulstil til fabriks-indstilling (”/resetNVS-Fac”)
- En side med funktionen udskriv NVS indhold ("/showNVS")
- En side med funktionen oplåsning for NVS-adgang ("/unlock-kernel"), bemærk oplåsningen låser automatisk efter 60 sekunder, eller hvis man enten laver reload af siden eller vælger andre sider, samt ved valg af reset funktionerne.
  BEMÆRK! At ved nulstilling af konfiguration nulstilles navne m.m. til default. Ved fabriksinstilling fjernes WiFI angangen, som derfor skal konfigureres forfra.
- En side med funktionen enable Bluetooth BLE funtionalitet (”/loadBle-Func”)
- En side til opdatering af firmware (”/hemmelig-update”)

Web-sidernes struktur

Alle websiderne i dette projekt er lavet efter følgende grundmodel,
const char page_xxx_html[] PROGMEM = R"rawliteral(WEB INDHOLD)rawliteral";
hvor page_xxx_html er navnet på den enkelte webkode stykke eks. **page_head_html** og disse er placeret i hver sin fil. Det vil sige at Header og Footer findes i en fil, Index/root som består af to web tekster findes i en fil. Disse filer samles således, der startes med en ”String samletHTML” text variabel hvor i filernes tekster indlæses som følger:

Index siden samles således: 1. page_head_html 2. index_body_top1_html 3. Her indsættes Antenne knapperne fra programkoden 4. index_body_top2_html 5. page_foot_html	Konfigurations siden samles således: 1. page_head_html 2. page_config_html 3. page_foot_html
Status siden samles således: 1. page_header_html 2. status_body_top_html 3. page_foot_html	Remote opdatering består af en samlet side: 1. page_update_html

Fælles for disse sider er at efter de er samlet, skiftes de dynamiske felter ud med de dynamiske data, og herefter sendes de så til web-serveren.
Jeg har valgt at hver webside placeres efter samme system og i hver sin .cpp fil med tilhørende include fil .h som peger på .cpp filen.

Baggrunden for at samle de forskellige websider i forskellige filer, i stedet for at lave det hele i den samme fil, skal findes i det at, de fleste sider i den oprindelige software er bygget op med en html tekst streng variabel, hvor alt indlæses linje for linje således:

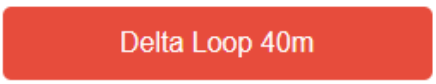
String Html = ”web content ”;

Html += “web content ” og så videre.

Hvilket kan være godt nok for nogen, men når der skal fejlfindes på dette, så står hele webindholdet på én linje, hvilket ikke lige er så rart, og gør det næsten umuligt at overskue. Derfor har jeg valgt at skrive siderne med den rigtige indrykning m.m., og det at lave flere filer med hver sin web del, er også for nemmere og hurtigere at kunne tilgå de rigtige dele af webserverens koder. En strukturerings metode jeg ofte bruger ved større programmer, så de forskellige ting adskilles fra hinanden, f.eks. finder man også koden til udskrivning af NVS systemets indhold i sine egne filer i projektet.

Forsiden (”/”) ROOT eller HOME

Bestykkes med fem knapper, fire til at vælge antenne udgange og en til at afbryde dem alle. Knapperne til antenneudgangene vil være RØDE når de er slukket og GRØNNE når de er tændte, teksten vil vare den bruger definerede tekst, eks.



Knappen for ”SLUK ALLE” vil som udgangspunkt være BLÅ, men der vil måske med tiden være mulighed for at ændre denne farve under konfigurationen, men er i skrivende stund fravalgt.

Lav selv en antenne omskifter helt fra start

Ved valg af antenne er der nogle ting der skal tages højde for:

- Hvis RF Tracer registrerer en aktiv sender, må der ikke kunne ændres på relæerne, derfor er der en variabel hvor denne status styres samt sendes ud til webklienterne, så man kan se hvorfor man ikke kan ændre noget.
- Er relæet et BISTABILT relæ, da skal der bruges to udgange til at styre hvert relæ, en til Set og en til Reset, derfor skal der bruges pulse metode med en defineret puls bredde på default 50ms.
  Da jeg har set at det godt kan tage nogen tid inden status ændrer sig ved relæ/udgangs valg, alt efter hvor lang tid man bruger på pulse bredden ved relæskift også selvom jeg har optimeret dette, derfor har jeg valgt en anden måde at opdatere relæerne på, det betyder at det ikke er web sidens front kode der ændrer relæerne, men blot sender signal ned til det bagved liggende system om hvad der skal ske.
- Er relæet et MONOSTABILT relæ, da skal der være strøm på relæet indtil der slukkes for antenne-udgangen. Lige som ved BISTABILT relæ, sendes der også besked om ændringer ned til samme underliggende system og at tænde eller slukke.
- Der skal sende status ud til webklienterne med navne og status (aktiv eller ej), og dette skal ske fra det bagved liggende system der også behandler beskeden om ændringer på relæer, d.v.s. hvilke antenner der skal være aktiv, samt sender status tilbage til websiden om hvilke relæer der nu er aktiv.
- Relæstyringen skal være af typen NON-BLOCKING, d.v.s at web interfacet ikke hænger mens relæ kontrollen arbejder, med skift og pulser.
- Relæ detaljernes styring vil ikke være synlig på siden, og kommer kun frem under konfigurationen

Under de fem relæknapper har vi et system status felt og her vises følgende:

- Information om låsning af knapper ved PTT aktiv, dette er en rød tekst som advarsel
- Information om hvilken antenne der er aktiv, blot et supplement til knapstatus.
- Information om relæboksens web adresse eks. http://antctrl.local, som kan bruges til at finde boksen på nettet ved normal drift.

Nederst findes endnu et felt, hvor der p.t. er to ADC-port data der udlæses. Den første viser den målte værdi på for RF Trace målinger, hvor den anden udlæser forsyningsspændingen i volt.

Øverste højre hjørne skal indeholde et tandhjul som indikerer, at her kan du finde konfigurationen (”/config”)

Webkoden samles af Root Handleren, som tidligere beskrevet ud fra page_head_html, index_body_top1_html, data med antenne valgs knapperne der opbygges on the fly, efterfulgt af index_body_top2_html, og slutteligt page_foot_html.

Disse html tekster ligger placeret i flash koden, og samles ved indlæsning af siden i en browser, kommunikationen frem og tilbage sker med websocket herved sparer man indlæsningen af siden hver gang der laves ændringer, i stedet opdateres kun de felter der har ændret sig.

Den valgte metode for lagring af webkode, er en beslutning om at have et bedre overblik over HTML koden under fremstillingen af softwaren samt debug i webbrowseren, i kontrast til metoden hvor hver linje i HTML koden tilføjes linje efter linje i kildeteksten hvilket ville betyde at overblikket for HTML koden går helt tabt og sværere at rette til.

Den egentlige kode for ROOT vil bestå af noget kode for placering og farver m.m. som grund model, dertil kommer så knapper som også fungerer som relæstatus felter med navn på antennen og en farve for tændt og slukket. Knapperne genereres under hver gang siden indlæses med de opdaterede informationer.

Alle felter der skal kunne ændre indhold, status eller farver, har tilknyttede navnefelter så der kan opdateres informationer straks fra webkodens sockets funktioner, uden at siden skal genindlæses først, og hvis flere klienter forbundet til denne web, opdateres disse også straks der sker ændringer.

Her lidt om konfigurations siden

Konfigurations siden fungerer lidt anderledes her laves der ændringer lokalt, i webbrowseren og gemmes først når der vælges gem, herved sendes alle data til serveren altså ESP32 kredsen hvor de lagres i både ram og flash.

Der er dog nogle undtagelser bl.a. en Låse knap til at give adgang til NVS-data, som er det sted hvor alle konfigurationer gemmes, herunder også WiFi konfigurationen.

Her beskrives lidt om metoden for konfigurations sidens sammensætning og funktionalitet.

I modsætning til indekssiden ”/root” som er en http og websocket side hvor alle ændringer opdateres straks, så er konfigurations siden "/config" bygget efter POST metoden, dette betyder at alle ændringer sendes til servere på én gang via en submit knap, her **SKAL** man passe på med hvad man laver inde i området før submit knappen. Dette har jeg konstateret i forbindelse med oprettelse af knapper der kalder funktioner i script området, som derefter sender små post beskeder tilbage til serveren. Uanset om disse funktioner er tomme eller laver noget, så medfører kald til disse en øjeblikket **BOOT af ESP32-S3** processoren, så derfor er disse knapper flyttet ned mellem to andre knapper, dog skjult indtil de skal bruges.

Nedenfor ses et eksempel på en POST side lagret i flash, med en titel og overskrift, en device hvor navnet kan ændres, en interval-værdi der også kan ændres, og til sidst en submit der sender data tilbage til serveren.

Bemærk at disse inputfelter indeholder følgende %NAVN% og %INTERVAL% dette er nøgler som udskiftes med de variable data inden de sendes til serveren, og når de sendes tilbage med submit kigger man efter felterne ’devicename’ og ’interval’ for at kunne ændre opsætning med de nye data. Eventuelle scripts befinder sig i et script område mellem dit egentlige web indhold og den afsluttende </body> som vist nedenfor.

1. Tilføj pladsholdere i din PROGMEM HTML

```
const char page_head_html[] PROGMEM = R"rawliteral(
<!DOCTYPE html>
<html>
  <head><meta charset='UTF-8'><title>NVS Indstillinger</title></head>
  <body>
    <h2>Enhedsindstillinger</h2>
    <form method='POST' action='/save'>
      <label>Enhedsnavn:</label><br>
      <input type='text' name='devicename' value='%NAVN%'><br><br>
      <label>Måleinterval (sekunder):</label><br>
      <input type='number' name='interval' value='%INTERVAL%'><br><br>
      <input type='submit' value='Gem indstillinger'>
    </form>
    <script>
    </script>
  </body>
</html>
)rawliteral";
```

Nedenfor ses et eksempel på en arduino web program kode, startende med inkludering af nødvendige biblioteker, efterfulgt af nødvendige globale variabler til brug for hele projektet.

Under variabler findes eks. klasse definitioner hvor man f.eks. tager WebServer class og laver et object af denne, dette fordi koden ikke kan bruges direkte, så derfor laves objektet server af typen WebServer hvor man nu bruger server.funktion/var som adgang til WebServer funktionerne.

Under 'void setup(){}' funktionen laver man alt det der skal lavet i forbindelse med opstart af projektet, altså initialisering af variabler, porte og meget andet eks. opstart af webserveren og dens funktioner. BEMÆRK setup kaldes kun én gang for hver opstart af processoren.

Derimod funktionen 'void loop(){}' kaldes fra systemets mainloop system, som styrer alt omkring systemet udenom dit program, d.v.s. kernen bag det hele, dette betyder også at hvis du laver en uendelig loop inde i funktionen loop(), så dør alle services som du er afhængig af eks. timere, serielle port styringer m.m.

I eksemplet nedenfor hentes data fra NVS området og bruges til eks. webserverens data, og her er det sådan at hvis data ikke findes i NVS, så initieres variablerne med en default værdi, som du selv bestemmer.

2. Hent data fra NVS under opstart (**setup**)

```
#include <WiFi.h>
#include <WebServer.h>
#include <Preferences.h> // Bruges til NVS på ESP32

WebServer server(80);
Preferences preferences;

// Globale RAM-variabler til dine dynamiske data
String enhedsNavn;
int maaleInterval;

void setup() {
  Serial.begin(115200);

  // Åbn NVS-rummet "indstillinger" i read-only tilstand (false)
  preferences.begin("indstillinger", false);

  // Hent data. Hvis de ikke findes i NVS endnu, bruges standardværdierne til højre
  enhedsNavn = preferences.getString("devicename", "Standard_ESP32");
  maaleInterval = preferences.getInt("interval", 60);

  preferences.end(); // Luk NVS midlertidigt

  // Server-ruter
  server.on("/", HTTP_GET, handleRoot);
  server.on("/save", HTTP_POST, handleSave);
  server.begin();
}

void loop() {
  server.handleClient();
}
```

I eks. 3 vises en funktion handleRoot til at håndtere siden '/' kaldet root, her hentes webkoden fra PROG_MEM hvorefter nøglerne NAVN og INTERVAL erstattes med data hentet fra NVS i eks. 2 hvorefter webkoden sendes til webserveren som svar på 'http://DinWeb.local/' kaldet fra en klient

3. Send HTML med NVS-data (**GET**)

```
void handleRoot() {
  // Kopier HTML-skabelonen fra flash til en dynamisk RAM-streng
  String html = String(FPSTR(page_head_html));

  // Erstat pladsholdere med dine NVS-data
  html.replace("%NAVN%", enhedsNavn);
  html.replace("%INTERVAL%", String(maaleInterval));

  // Send den færdige side til browseren
  server.send(200, "text/html", html);
}
```

Kaldet fra klienten som resulterede i at klienten fik præsenteret konfigurations siden, gør at der nu kan laves nogle ændringer i de data der blev præsenteret på denne side, og når der trykkes på submit knappen der typisk i dette tilfælde vil hedde 'SAVE and RESTART', med et link til '/save' som medfører at funktionen handleSave nedenfor bliver kaldt.

Her kontrollere om valide data, hvorefter de globale RAM-variabler opdateres med de nye data, og efterfølgende sendes de også til NVS, så de ikke går tabt ved genstart. Til sidst sendes der svar til klienten om det er gået godt eller ej, samt besked til klienten om at indlæse root siden igen.

4. Gem nye data i NVS ved **POST**

```
void handleSave() {
  // Tjek om felterne findes i POST-anmodningen
  if (server.hasArg("devicename") && server.hasArg("interval")) {

    // 1. Opdater globale RAM-variabler
    enhedsNavn = server.arg("devicename");
    maaleInterval = server.arg("interval").toInt();

    // 2. Gem permanent i NVS (åbn i skrivetilstand = false)
    preferences.begin("indstillinger", false);
    preferences.putString("devicename", enhedsNavn);
    preferences.putInt("interval", maaleInterval);
    preferences.end();

    // 3. Send en bekræftelse eller send brugeren tilbage til forsiden
    server.send(200, "text/html", "<h3>Indstillinger gemt i NVS! Enheden
genindlæser...</h3><script>setTimeout(function(){window.location.href='/';},2000);</script>");
  } else {
    server.send(400, "text/plain", "Fejl: Manglende data i POST");
  }
}
```

NVS hvad er det og hvad bruges det til

NVS er et Non Volatile Storage areal i processorens flash, det er her systemet gemmer vigtige informationer bl.a. WiFi data dog kun seneste brugte acces point, for ikke at afsløre vitale informationer om andres net, og selv om du kan aflæse NVS nøglerne så er netop disse data krypteret og dermed ikke læsbare.

Her gemmer vi også løbende konfigurations data og antennevalgene, dermed er vores vigtigske dynamiske data både tilgængelig i RAM og gemt i FLASH området NVS.

Hvilke data er det så vi gemmer: Her ses hvordan data indlæses fra NVS samt hvad der sker hvis dataene ikke er tilgængelig d.v.s. ødelagte eller manglende

Syntaks: prefs.get'var_type'(Data felt navn, default værdi hvis ingen)

BEMÆRK! Der kan være ændringer i den seneste version der er udgivet

```
void loadConfiguration() {
  // prefs.begin("antSwitch", true);
  bool SafetySave = false;
  prefs.begin(nvsDataName, true);
  // 1. 2.
  SafetySave = prefs.getString("webTitle", "") == "" ? true : false;

  webTitle = prefs.getString("webTitle", "Antenne Kontrol 10.0");
  mdnsName = prefs.getString("mdns", "antctrl");
  isBluetoothMode = prefs.getBool("use_ble", false); // Default er Wi-Fi (false)
  // 3.
  for (int i = 0; i < 4; i++) {
    relays[i].name = prefs.getString(("rName" + String(i)).c_str(), "Antenne " + String(i + 1));
    relays[i].SetPin = prefs.getInt(("rPinSet" + String(i)).c_str(), defRelays[0 + i]);
    relays[i].ResetPin = prefs.getInt(("rPinRes" + String(i)).c_str(), defRelays[4 + i]);
    relays[i].state = prefs.getBool(("rState" + String(i)).c_str(), false);
    relays[i].startState = prefs.getBool(("rStart" + String(i)).c_str(), false);
  }
  // 4.
  ADC_Ports[0].name = prefs.getString(("aName0"), "Antenne Trace");
  ADC_Ports[0].AdPin = prefs.getInt(("adPin0"), adcPins[0]);
  // 5. 6.
  adcInterval = prefs.getLong("aInterval", (long)2000);
  // Punkt 6. fjernet fremad rettet
  // pttPin = prefs.getInt("pttPin", PTT_PIN);
  // 7. - 9. 10.   >>>> Punkt 8 fjernet fremadrettet <<<<
  relayTypCfg.type = (RelayType)prefs.getUChar("rType", BISTABLE);  // preferences.putUChar("rType",
(uint8_t)relayTypCfg.type);
  relayTypCfg.startMode = prefs.getInt("rStartMode", 0);
  relayTypCfg.RelayPulseMs = prefs.getLong("rPulseMs", (long)50);
  // 11. 12. 13. 14.
  pttInterlockEnabled = prefs.getBool("pttInLock", false);
  // punkt 12. switchDelayEnabled erstattet med BleFuntionalityEnabled
  BleFuntionalityEnabled = prefs.getBool("enable_ble", false);
  // Punkt 13. fjernet fremad rettet
  // PTT_ACTIVE_LOW = prefs.getBool("pttActLow", true);
  relayTypCfg.useFixedStartState = prefs.getBool("fixedStart", false);
  // 15. 16. 17.
  relayTypCfg.fixedStartAntenna = prefs.getInt("fixStrtAnt", 1);
```

```
  relayTypCfg.lastSelectedAntenna = prefs.getInt("lastSlctAnt", 0);
  relayTypCfg.activeAntenna = prefs.getInt("activAnt", 0);
  // 18.
  // RGB Neopixel LED onboard farve
  rgbR = prefs.getUChar("rgbR", 25);
  rgbG = prefs.getUChar("rgbG", 138);
  rgbB = prefs.getUChar("rgbB", 10);
  prefs.end();
  // Hvis vi starter fra schratch gemmer vi lige default data
  if(SafetySave){saveConfiguration();}
}
```

Kort fortalt så gemmes:

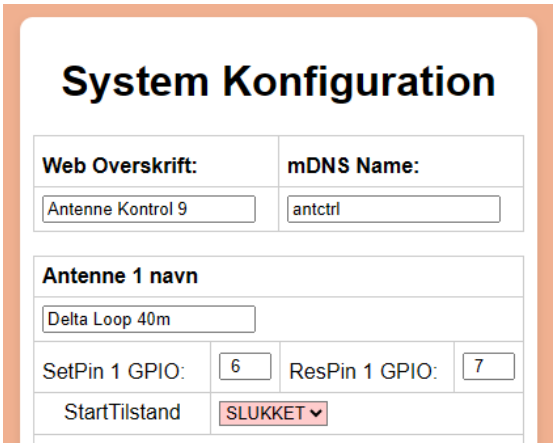
- **Web tittel**
- **mDNS navn**
- **Bluetooth mode (er bluetooth eller WiFi aktiv, det enten eller, ikke samtidigt)**
- **Alle fire relæers følgende data**
    - **Relæ navn**
    - **SetPin GPIO pin nr.**
    - **ResetPin GPIO nr.**
    - **State** aktiv antenne state, altså om denne antenne er aktiv lige nu
    - **StartState** om det er denne antenne der skal være fast opstarts forbindelse
- **De to ADC kanalers data**
    - **Portens navn**
    - **Portens GPIO pin nr.**
- **ADC måle interval**
- **Relæ type BI_STABIL / MONO_STABIL**
- **Startmode Fast eller ej**
- **Relæ puls millisekunder (max ca. 50ms)**
- **PTT interlock Enable**
- **BLE Funtionalitet Enable**
- **Fixed Start Enable**
- **Fixed Start Antenne**
- **Sidst Valgte Antenne**
- **Aktiv Antenne**
- **RGB data der bruges til NEOPIXEL leddioden på processor kortet**
    - **R for rød**
    - **G for grøn**
    - **B for blå**

Hvis man ønsker at se hvad der faktisk findes i NVS, kan man tilslutte en seriel terminal og udlæse NVS ved at tilgå konfigurations siden, lave unlock NVS og så tilgå denne inden 60sec. Tilgå ”**http://antctrl/showNVS**” eller skriv showNVS på den serielle terminal, herefter udskrives alle nøgler i denne terminal

Hvis man ønsker at slette NVS og starte forfra, dette kan man tilgå fra konfigurationssiden efter unlock, hvorefter man kan fjerne din personlige konfiguration eller alt med fabriks indstillinger.

Lav selv en antenne omskifter helt fra start

Se det var lidt om metoden til at fremstille "/config" siden, og nu lidt mere detaljer om denne.



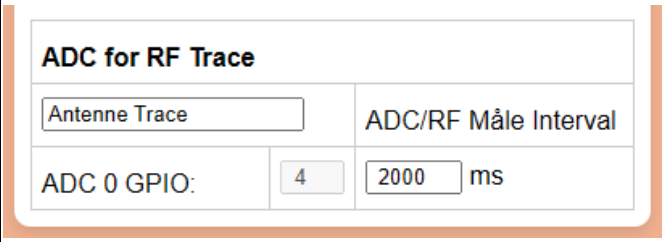
Øverst på konfigurations siden ses dette.

En web overskrift som kan ændres efter eget ønske mDNS navnet antctrl kan også ændres efter eget ønske

Antenne 1 navnet her: Delta Loop 40m kan ændres efter eget ønske

De GPIO der bruges til henholdsvis Set og Reset kan ikke ændres, de er der kun for info, tidligere kunne de rettes, men med et fast layout giver det ikke mening at kunne rette disse.

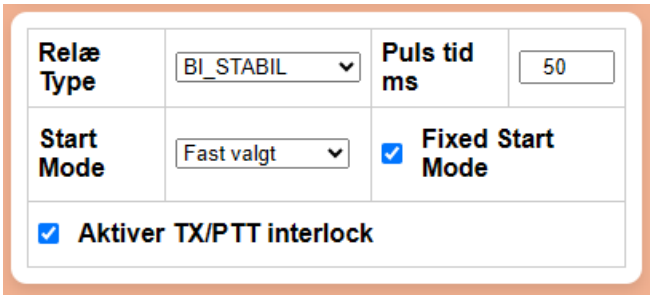
Feltet StartTilstand, bruges til at vælge om netop dette relæ skal være det relæ der er aktiv ved genstart, hvis der er valg fast opstart.



Her vises ADC kanalen for RF Trace, hvor ADC-kanal 0 GPIO 4, der bruges til at lytte efter et HF signal fra ca. 5W og op til ca. 1000W for at kunne blokere for antenneskift under TX

Navnet kan ændres, men GPIO er fast.
Måle interval kan ændres men er kun til brug for signal udlæsning på forsiden.

ADC 1 GPIO 5 som ikke længere fremgår af denne side, er inddraget til visning af strømforsyningens spænding baseret på LiFePO4 batteri tilstande.



Næst nederst på siden findes muligheden for at ændre relætypen nemlig BI_STABIL eller MONO_STABIL.

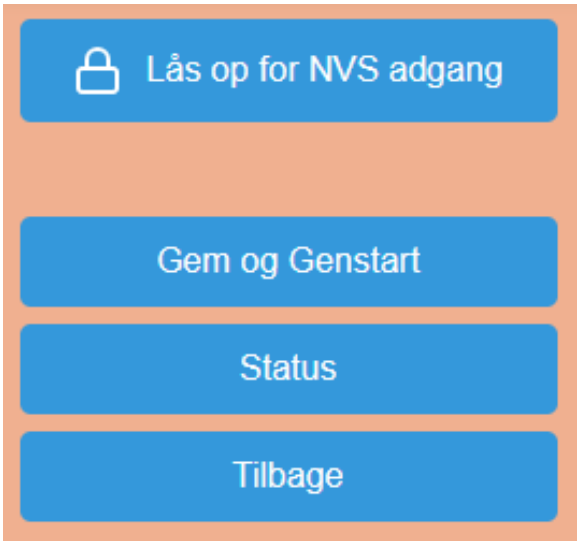
Start Mode: her kan man vælge hvilken opstartsmode du ønsker, betinget af **Fixed Start Mode** er valgt

1. Ingen antenne
2. Sidst valgte (default)
3. Fast valgt antenne

Hvis Fixed start mode ikke er valgt, er sidst valgte der gælder, uanset om det er ingen eller en af de fire antenner.

Hvis der bruges BI_STABIL relæ er det muligt at ændre puls tiden for disse relæer, værdien bør ikke være større end 50ms. Tallet er sat til max 50.

TX/PTTInterlock mode bruges til at aktivere eller deaktivere RF Trace blokering, for antenneskift under TX



Nederst på konfigurations siden findes disse knapper, **Lås op for NVS adgang**, **Gem og Genstart**, **Status**, samt **Tilbage**

Gem og Genstart kalder på "/save" siden som ikke er synlig, men blot udfører valgte funktion.

Status kalder på "/status" siden hvor man kan se forskellige system info.

Tilbage bringer brugeren tilbage til den forrige side

Låse knappen lukker op for to knapper som kommer frem mellem **Status** og **Tilbage** knapperne, disse er dog kun synlige i 60 sekunder så forsvinder de igen. Med disse knapper kan enten slette konfigurationen eller bringe systemet tilbage til fabriksindstillingerne.



Sådan ser den ud når der låses op for funktioner

1. Der er nu åben for http://antctrl.local/showNVS som medfører at hele NVS hukommelsen udskrives på den serielle port
2. Der er også åbent for serielle kommandoer, men del fjernes nok senere igen
3. Der er også åbent for firmware opdatering
4. og så selvfølgelig kan der nu også vælges
    a. Nulstilling af konfiguration
    b. Nulstilling til fabriks indstilling
    c. At slå Bluetooth BLE funktionaliteten til eller fra som en skifte funktion

Efter ca. 2 min. lukkes der automatisk igen, eller hvis der udføres en af disse beskrevne valg.

Forsiden af Antenne kontrollerens web

Forsiden som også ses på omslaget, har et par detaljer vi nok lige skal gennemgå, noget af det er allerede beskrevet i tekst på side 15, her tager vi lige fat i den grafiske del



Når BLE funktionen er slået fra, som den er default, så sidens top således ud.
Det lille tandhjul i det grønne felt er en knap der bruges til at kalde Konfigurationen/Setup frem



Når BLE funktionen er tilgængelig ser hoved siden således ud i toppen.
BLE feltet bruges vælge BLE mode hvorved denne side ikke kan bruges, før der igen skiftes tilbage igen.

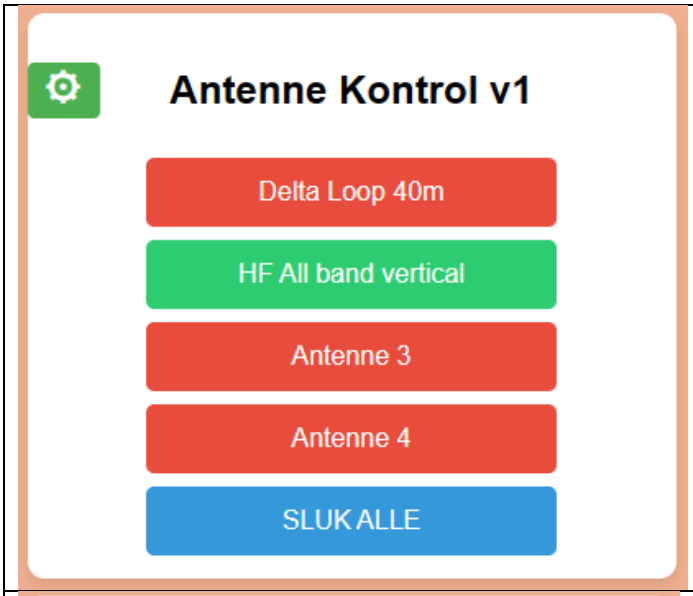
Lav selv en antenne omskifter helt fra start

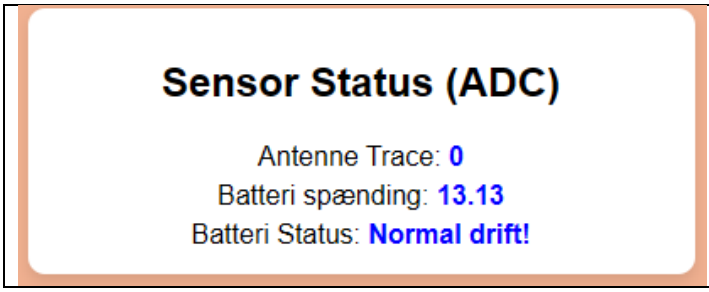


Dette er toppen af BLE koden, hvor feltet med WiFi er knappen der skifter over til WiFi igen og denne side vil ikke kunne bruges før der skiftes tilbage til BLE.

Pair feltet er også en knap den bruges til at forbinde dig med Antenne Kontrolleren med BLE.

Hvis har du laver opdatering af siden med eks. F5 medens du er forbundet, skal du igen forbinde med Pair, da du har smidt forbindelsen.

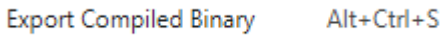
	Her ser du antenne vælgerne som er de røde og grønne felter altså knapper, som du har mulighed for at navngive efter eget valg under konfigurationen Disse knapper har numrene 1, 2, 3, 4 fra toppen og ned. Den grønne er den aktive antenne og de røde er slukket, den nederste som er blå, slukker for alle og dermed alle røde og ingen antenne tilsluttet til din radio.
	Når der skiftes til Bluetooth BLE ser siden nu sådan ud, og kræver at du skifter tilbage til WiFi inden du kan opdateres siden igen.
	Dette felt er hvor du kan få aktuel status, som hvilken antenne der aktiv som også vises på knapperne, hvilket var et nyt tiltag nemlig at sætte farver på knapperne. Her ses også URL adressen vist i grafikken Hvis relæskift bliver blokeret af en aktiv sender, kommer der en rød besked op lige over Aktiv antenne feltet.



Her vil du kunne se måle resultater for Antenne Trace.
Batteri spændingen vises også her samt en tekst baseret status for batteriet baseret på LIPOFE

Opdatering af Firmware

Som vanligt kan man selvfølgelig opdatere sin firmware på Antenne Kontrolleren ved at ændre i den medfølgende kildetekst også kaldet source, den skal compileres med Arduino IDE version 2.3.10 eller højere.

Når man har afsluttet rettelserne og kompileret koden kan man uploade via USB kabel, alternativt kan man også lave en opdatering over WiFi stedet, det kræver bare at du er i besiddelse af den binære kode ”xxxx.ino.bin”, og er instrueret i at bruge denne funktion.
den binære kode får du ved at vælge Sketch i menuen og herefter 

her efter kan du taste Alt + Ctrl + K hvorefter en stifinder vinduer lukker op i din source bibliotek, her finder du et bibliotek der hedder <build>, nede i den sti finder du dine binære filer og du skal bruge den der slutter på ”.ino.bin”



Når man ønsker at opdatere firmwaren på Antenne Kontrolleren og har fået de nødvendige informationer for at kunne lave denne opdatering Så får man adgang til dette billede, hvor man vælger filen en den nye firmware med et navn der slutter med ”.ino.bin” og derefter vælger Opdater Firmware, nu kommer der en lille bjælke frem der viser opdateringens upload Progress Bar.

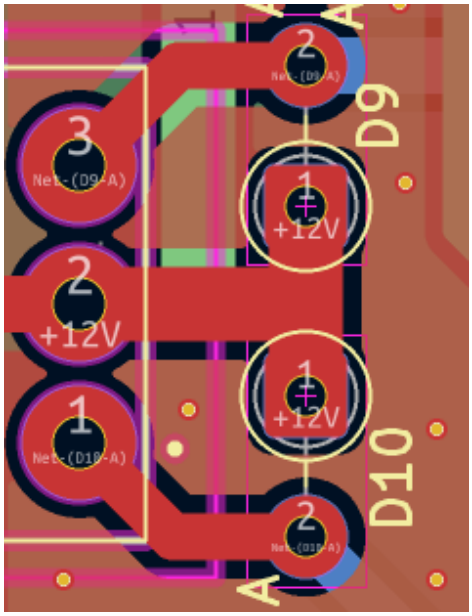
Når du er færdig med at uploade din nye firmware, kan du roligt lukke denne fane igen, og vende tilbage til den fane hvor din Antenne Kontrollers konfigurations side er, og her vælge tilbage til forsiden.

Hvis du ikke har lavet nogen fatale fejl, så skulle kontrolleren gerne virke igen.

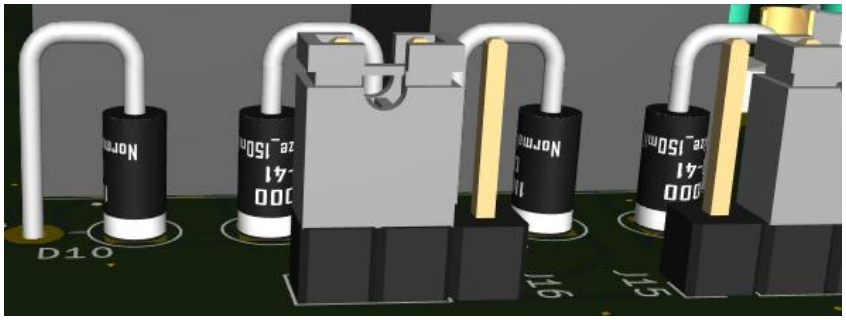
Monterings vejledning

Brug diagrammet som vejledning, og husk nu at vende alle komponenter rigtigt da der ellers kan forekomme hvid røg der slippes løs, når der sættes strøm til.

BEMÆRK at flyback dioderne ved relæerne SKAL vende rigtigt.



Som det ses her for den BI_STABILE relæ der er 12V plus placeret på midtebenet, så dioderne skal monteres med Katoderne nedad i den gule ring hvor der også står +12V og den anden ende skal bukkes rundt så den kan komme ned i det andet hul markeret med A for Anode.
Lige som der er vist på billedet neden for.



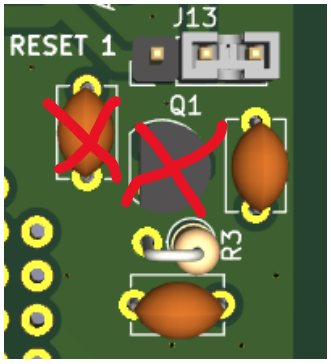
For MONO_STABIL relæ undlades D3, D5, D7 og D9 disse bliver by-passet af en jumper/eller lus alt efter hvad man vælger et montere

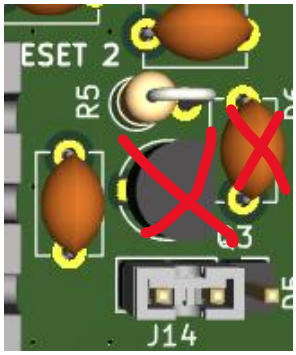


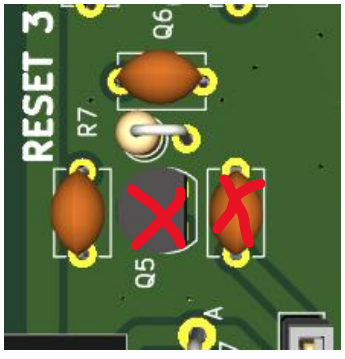
Her vises den ene af de fire jumpere som er J13, J14, J15, J16
Jumperen er vist i default stilling som er gældende for Bistabile relæer, ved monostabile relæer flyttes jumperen blot en plads.

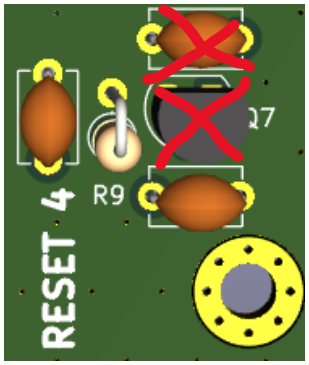
I nedenstående kredsløb undlades transistorerne, og de fire 10nF, de øvrige komponenter bibeholdes af hensyn til ESP32-S3 computeren, for at beskytte denne mod HF-indstråling.

Monterer man alligevel transistorerne er dette også OK, de vil ikke være i brug og er beskyttet af jumper delen som er indført for nemmere at kunne omstille mellem bistabilt og monostabilt kredsløb.









Lav selv en antenne omskifter helt fra start

Her er der lavet lidt plads som kan bruges til notater m.m., men reelt bare et lille frispark for at få siderne til at gå op med et multiplum af 4 så dokumentet passer til en folder.

Stykliste

Her kommer den samlede stykliste med priser.

Stykliste tekst og Priser	Forventet Pris i DKK	
	MONOSTABIL	BISTABIL
1. Færdig produceret PCB for Antenne Kontrol version 4 August 2026	130,00	130,00
2. 5 stk. SO-239 stik	66,80	66,80
3. 20 stk. M3x5mm Standoff 710-971050324 ved 100 stk	30,00	30,00
4. 20 stk. stjerneskiver	6,80	6,80
5. 20 stk. M3 møtrikker	4,00	4,00
6. 20 stk. M3 monterings skruer	8,00	8,00
7. 4 stk. Relæer		
a. BI-STABIL RT314F12		151,00
i. 8 stk. 1N5819 dioder for relæspoler		10,90
ii. 8 stk. BS170 FET		11,30
b. MONO_STABIL RT424012F	116,00	
i. 4 stk. 1N5819 dioder for relæspoler	5,45	
ii. 4 stk. BS170 FET	5,65	
8. 4 stk. Jumper for relæstyringen	?	?
9. 1 stk. YD-ESP32-S3 N16R8 v1.4 med ekstern antenne	60,40	60,40
10. 2 stk. Single Row Female Pin Header 1X22Pin til ESP32 board	3,59	3,59
11. 1 stk. Phønix MKDSN1,5/ 6-5 - PCB terminal block vare nr. 1729050	15,42	15,42
12. 1 stk. LM7805_TO220	3,44	3,44
13. 1 stk. 470uF elektrolyt radial ø10mm benafstand 5mm	?	?
14. 1 stk 10uF tantal	?	?
15. 14 stk. 100nF	?	?
16. 9 stk. 10nF	?	?
17. 14 stk. 1nF	?	?
18. 1 stk. 511-1N5822 diode for polariserings fejl beskyttelse ved 100 stk	1,93	1,93
19. 1 stk. 100uH filter spole til forsyning ind	1,50	1,50
20. 8 stk. 1K modstande	?	?
21. 1 stk. 10K modstand	?	?
22. 1 stk. 10K trimmer Runtrun RM-065	?	?
23. 1 stk. 20K modstand.....	?	?
24. 2 stk. 100K modstand	?	?
25. 1 stk. 3,3v Zener diode, 771-NZX3V0C-133 ved 100 stk	1,22	1,22
26. 1 stk. 4148 ensretter diode.....	?	?
27. Metal kabinet	?	?
28. Plastkasse udenom metal kassen (Pæn indpakning)	?	?
Samlet priser	460,20	506,30

BEMÆRK: priser markeret med ? betyder at de hentes fra lager og derfor ikke koster noget.

Told og forsendelse kommer oveni, men jo flere styk jo bedre pris, dog skal siges at de fleste af ovenstående priser er de endelige priser dags dato. Printet kan godt blive lidt dyrere grundet de nye told satser.

Bemærk at nogle af komponenterne måske kan skaffes lidt billigere i Kina, dette har jeg også benyttet mig af, dog har jeg lagt priser på eks. SO-239 stikkene til ca. 13,36 kr. stykket og så er de skaffet i EU, her er det således at vi har nogle på lager til 0,00 kr. men ved større ordrer skal vi ud af huset for at skaffe disse.

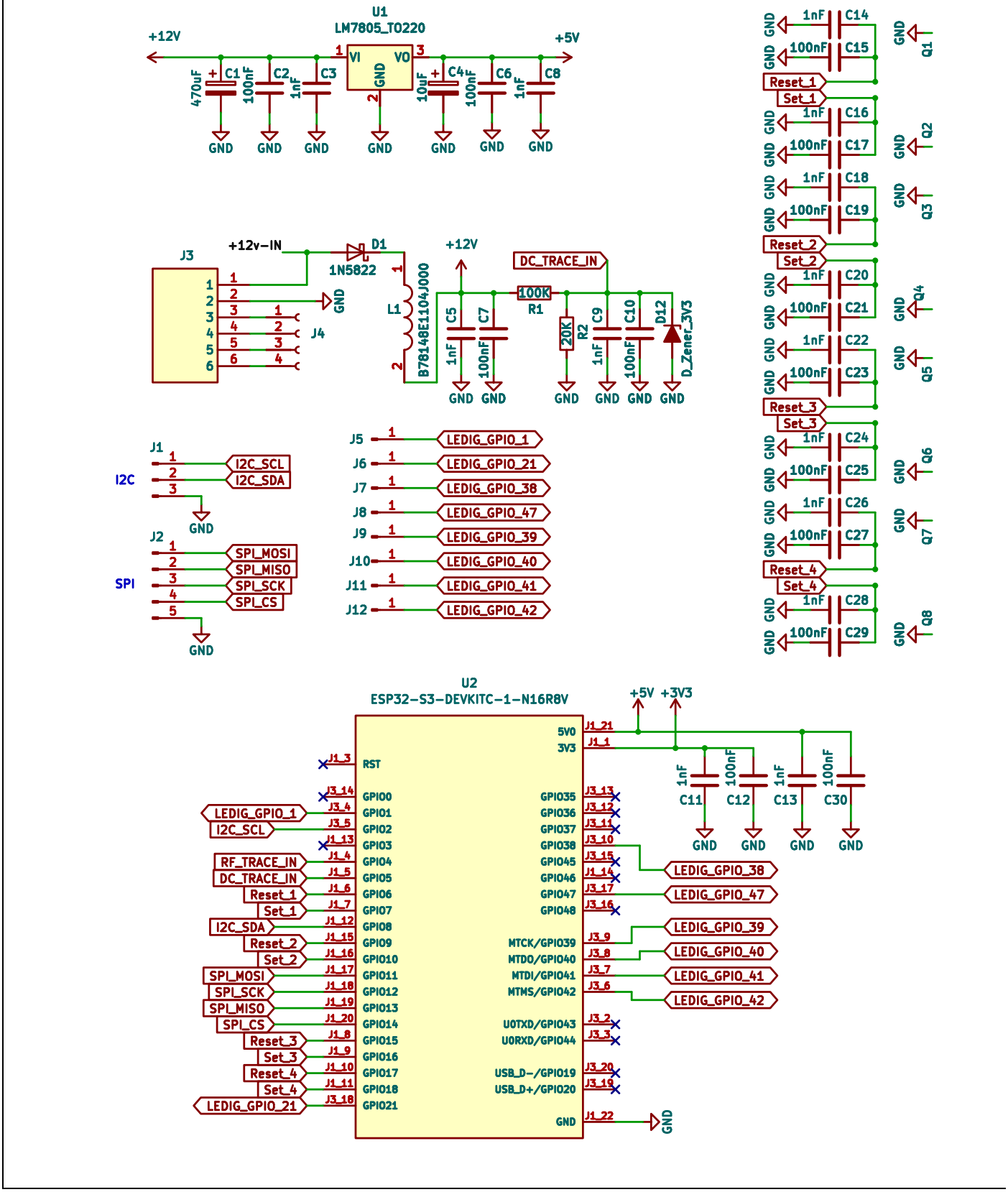
Vi har set en del forskellige priser på bl.a. relæerne, og har i den 21.10.2026 fundet priser i EU på ca. 37,75 kr for et bistabilt relæ, og ca. 29,00 kr. for et monostabilt relæ.

Component Liste

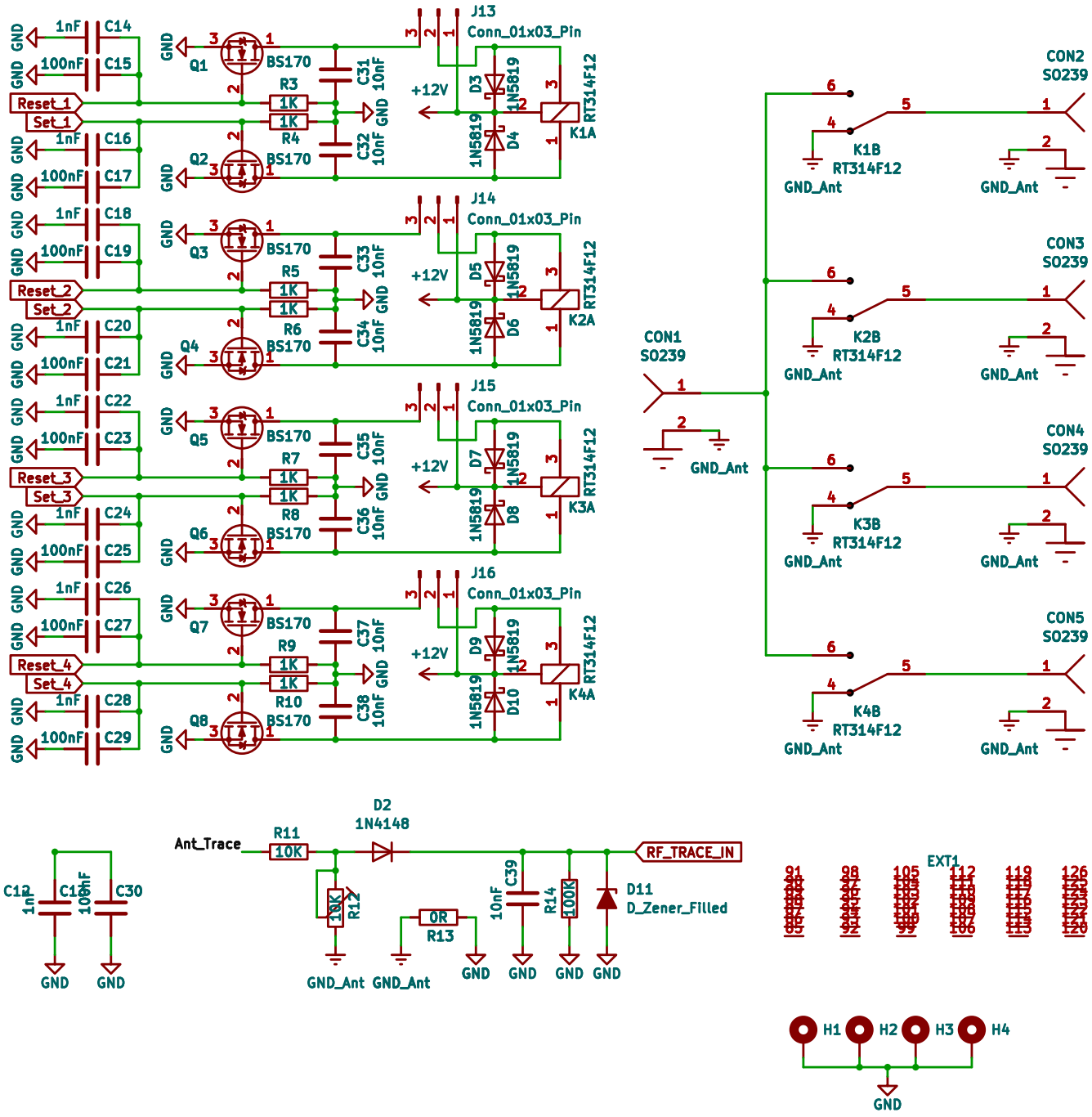
R1	100K	C35	10nF
R2	20K	C36	10nF
R3	1K	C37	10nF
R4	1K	C38	10nF
R5	1K	C39	10nF
R6	1K		
R7	1K	Q1	BS170 FET
R8	1K	Q2	BS170 FET
R9	1K	Q3	BS170 FET
R10	1K	Q4	BS170 FET
R11	10K	Q5	BS170 FET
R12	10K TRIM Runtrun RM-065	Q6	BS170 FET
R13	0R	Q8	BS170 FET
R14	100K		
		K1	RT314F12 BiStab eller RT424012 MonoStab
C1	470uF 25V	K2	RT314F12 BiStab eller RT424012 MonoStab
C2	100nF	K3	RT314F12 BiStab eller RT424012 MonoStab
C3	1nF	K4	RT314F12 BiStab eller RT424012 MonoStab
C4	10uF		
C5	1nF	D1	1N5822
C6	100nF	D2	1N4148
C7	100nF	D3	1N5819
C8	1nF	D4	1N5819
C9	1nF	D5	1N5819
C10	100nF	D6	1N5819
C11	1nF	D7	1N5819
C12	100nF	D8	1N5819
C13	1nF	D9	1N5819
C14	1nF	D10	1N5819
C15	100nF	D11, D12	3,3V Zener (771-NZX3V0C-133
C16	1nF		
C17	100nF	CON1	SO-239
C18	1nF	CON2	SO-239
C19	100nF	CON3	SO-239
C20	1nF	CON4	SO-239
C21	100nF	CON5	SO-239
C22	1nF		
C23	100nF	U1	LM7805_TO220
C24	1nF	U2	YD-ESP32-S3 N16R8 V1.4 med extern ant.
C25	100nF		2 stk. Female Pin Header 1X22Pin
C26	1nF	L1	100uH filter spole
C27	100nF		
C28	1nF	J1	3 POL Pin Header
C29	100nF	J2	5 POL Pin Header
C30	100nF	J3	Phoenix MKDSN1,5/ 6-5 - PCB term. block
C31	10nF	J13	3 POL Pin Header for jumper
C32	10nF	J14	3 POL Pin Header for jumper
C33	10nF	J15	3 POL Pin Header for jumper
C34	10nF	J16	3 POL Pin Header for jumper

For øvrige stumper se prisliste

DIA


U1
LM7805_TO220
+12V
+5V
VI
VO
GND
C1
C2
C3
C4
C6
C8
470uF
100nF
1nF
10uF
J3
+12v-IN
D1
1N5822
J4
L1
B78148E1104J000
+12V
DC_TRACE_IN
100K
R1
20K
R2
C5
C7
C9
C10
D12
D_Zener_3V3
J1
I2C
I2C_SCL
I2C_SDA
J2
SPI
SPI_MOSI
SPI_MISO
SPI_SCK
SPI_CS
J5
J6
J7
J8
J9
J10
J11
J12
LEDIG_GPIO_1
LEDIG_GPIO_21
LEDIG_GPIO_38
LEDIG_GPIO_47
LEDIG_GPIO_39
LEDIG_GPIO_40
LEDIG_GPIO_41
LEDIG_GPIO_42
C14
C15
C16
C17
C18
C19
C20
C21
C22
C23
C24
C25
C26
C27
C28
C29
Reset_1
Set_1
Reset_2
Set_2
Reset_3
Set_3
Reset_4
Set_4
Q1
Q2
Q3
Q4
Q5
Q6
Q7
Q8
U2
ESP32-S3-DEVKITC-1-N16R8V
+5V
+3V3
5V0
3V3
RST
GPIO00
GPIO01
GPIO02
GPIO03
GPIO04
GPIO05
GPIO06
GPIO07
GPIO08
GPIO09
GPIO10
GPIO11
GPIO12
GPIO13
GPIO14
GPIO15
GPIO16
GPIO17
GPIO18
GPIO21
GPIO35
GPIO36
GPIO37
GPIO38
GPIO45
GPIO46
GPIO47
GPIO48
MTCK/GPIO39
MTDO/GPIO40
MTDI/GPIO41
MTMS/GPIO42
U0TXD/GPIO43
U0RXD/GPIO44
USB_D-/GPIO19
USB_D+/GPIO20
C11
C12
C13
C30
RF_TRACE_IN

GRAM


J13
Conn_01x03_Pin
J14
Conn_01x03_Pin
J15
Conn_01x03_Pin
J16
Conn_01x03_Pin
Reset_1
Set_1
Reset_2
Set_2
Reset_3
Set_3
Reset_4
Set_4
Q1
Q2
Q3
Q4
Q5
Q6
Q7
Q8
BS170
R3
R4
R5
R6
R7
R8
R9
R10
1K
+12V
D3
D4
D5
D6
D7
D8
D9
D10
1N5819
K1A
K2A
K3A
K4A
RT314F12
K1B
K2B
K3B
K4B
CON1
CON2
CON3
CON4
CON5
SO239
GND_Ant
D2
1N4148
Ant_Trace
R11
10K
R12
R13
0R
R14
100K
C39
10nF
D11
D_Zener_Filled
RF_TRACE_IN
EXT1
H1
H2
H3
H4
GND

Afrunding

Det var så en lille tur gennem projektet Lav selv en antenne omskifter helt fra start, jeg håber at jeg har været rundt omalt det der er væsentlig omkring fremstilling af en antenne omskifter.

Dokumentet har været et levende dokument med løbende ændringer igennem flere måneder, og jeg mener at dette dokument har nået et niveau hvor det kan overgå til Release Version 1.0 November 2026, og ja der kan da stadig komme opdateringer.

Source og binær filernes version nummer 10.1 er et privat udviklings stadie og vil også skifte til Release Version1.0.0 November 2026

Hermed ønsker jeg dig held og lykke med projektet og eventuelle nye projekter som jeg måske har givet dig lyst til selv at binde an med.

Mvh Christian Knop alias OZ1KLM og Kristian Rasmussen OZ5KR som har været med på side linjen