Running over STM8

by pancake & brainstorm

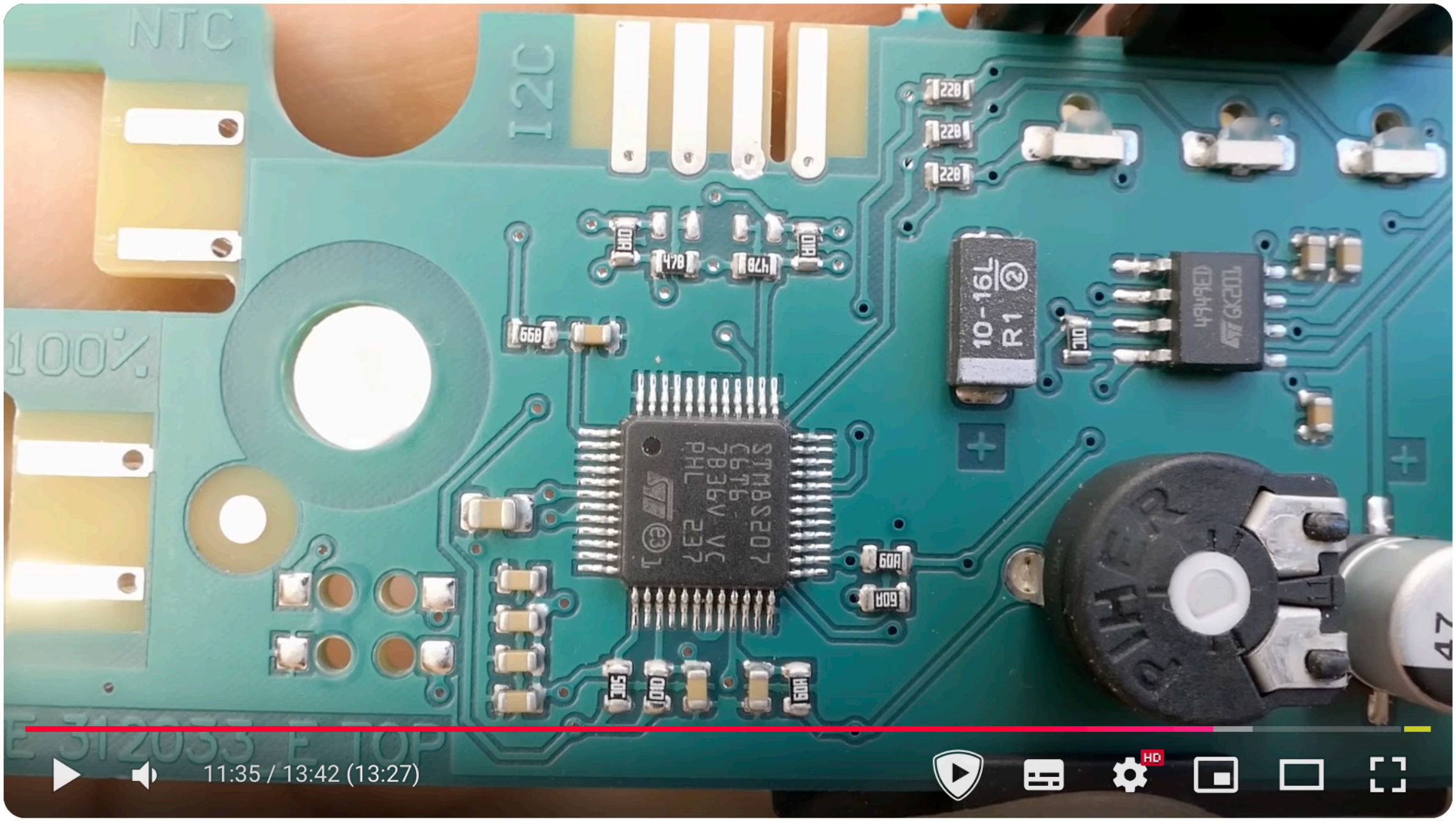
Agenda

1. STM8: What is this?
2. STM8: Motivation
3. STM8: e-waste target
4. STM8: PCBs reversing
5. STM8: radare2 improvements
6. LLMs: r2ai's decai for firmware decompilation
7. SVD: Improvements outside r2land
8. Conclusion
9. Future

STM8

*The STM8 is an **8-bit microcontroller family by STMicroelectronics**.*

*The STM8 microcontrollers use an extended variant of the ST7 microcontroller architecture. STM8 microcontrollers are particularly **low cost for a full-featured 8-bit microcontroller**.*



What's inside a tankless water heater

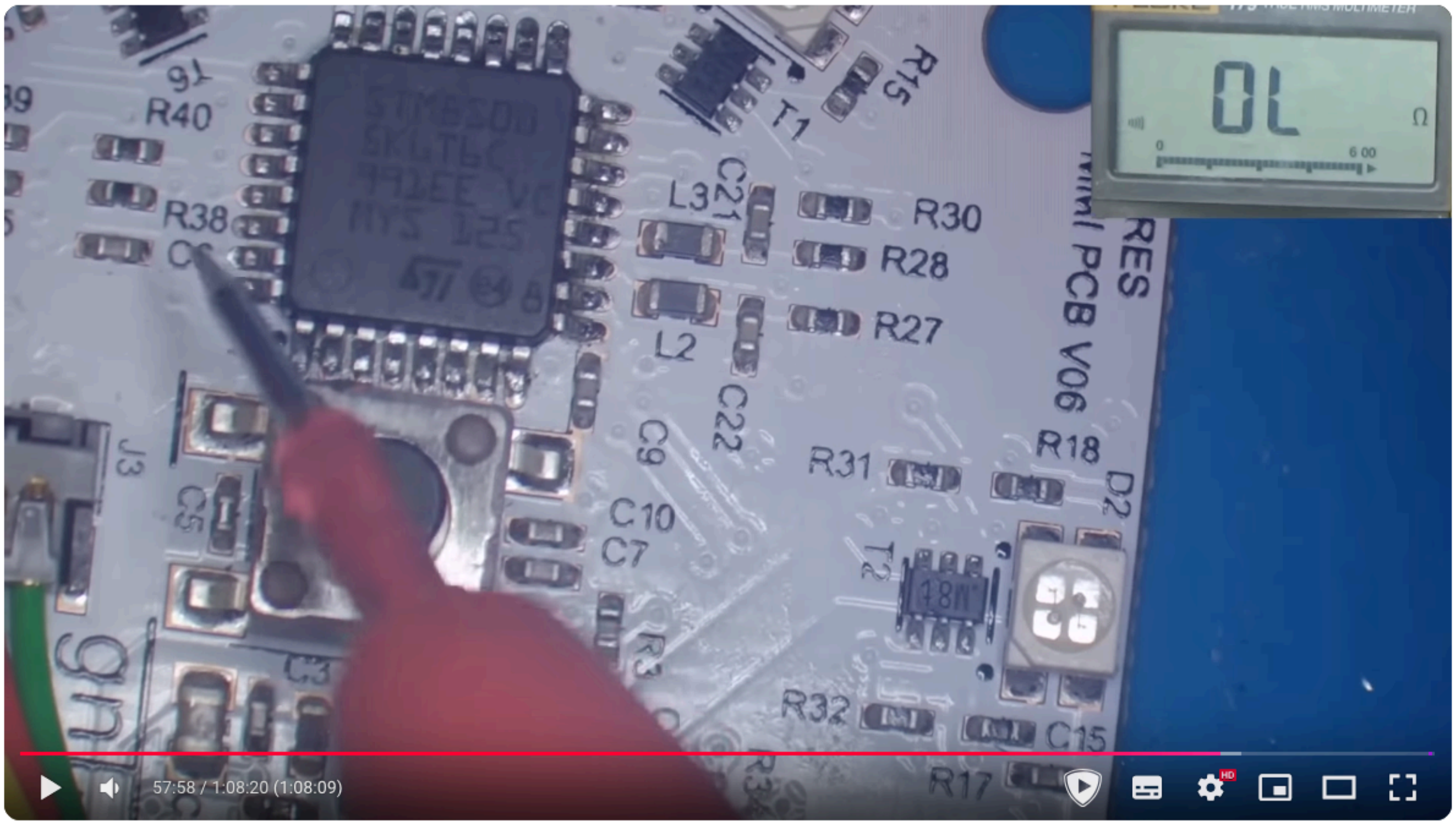
DiodeGoneWild
233 000 prenumeranter

Prenumererar

1 986

Dela

Thanks



Faulty VertuoPlus Nespresso Coffee Machine | Can I Fix it?

Buy it Fix it
48 900 prenumeranter

Prenumererar

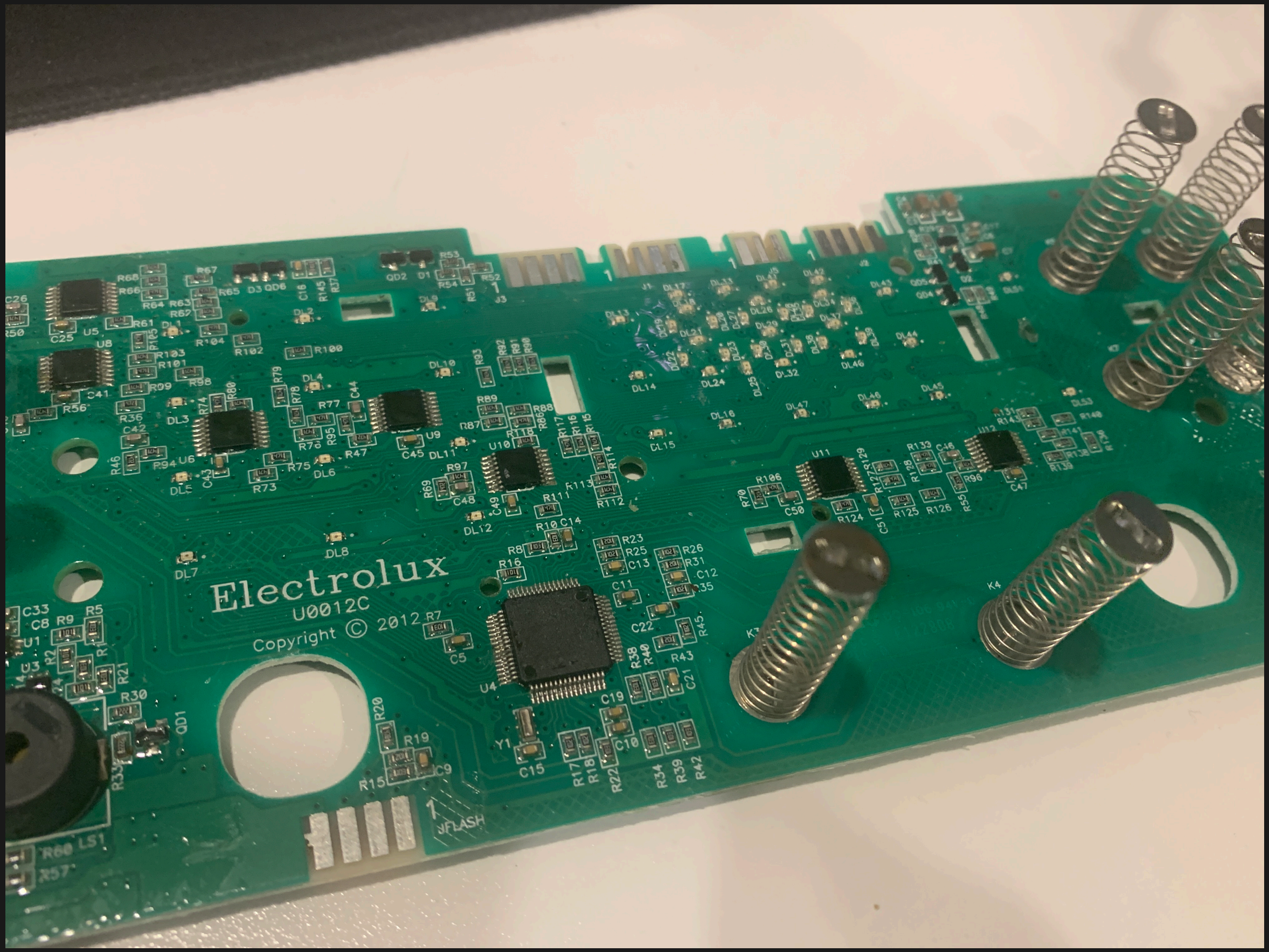
2 670

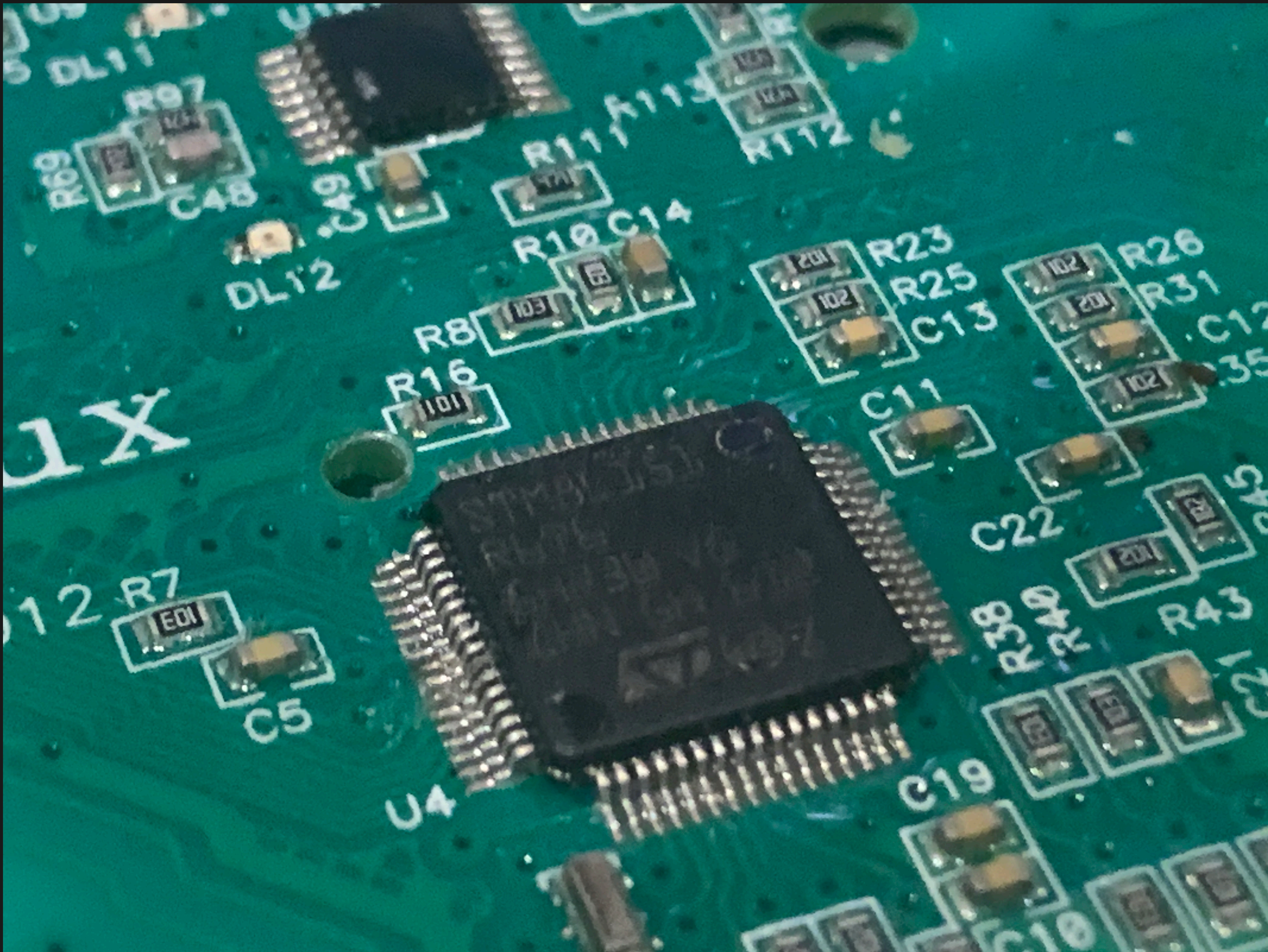
Dela

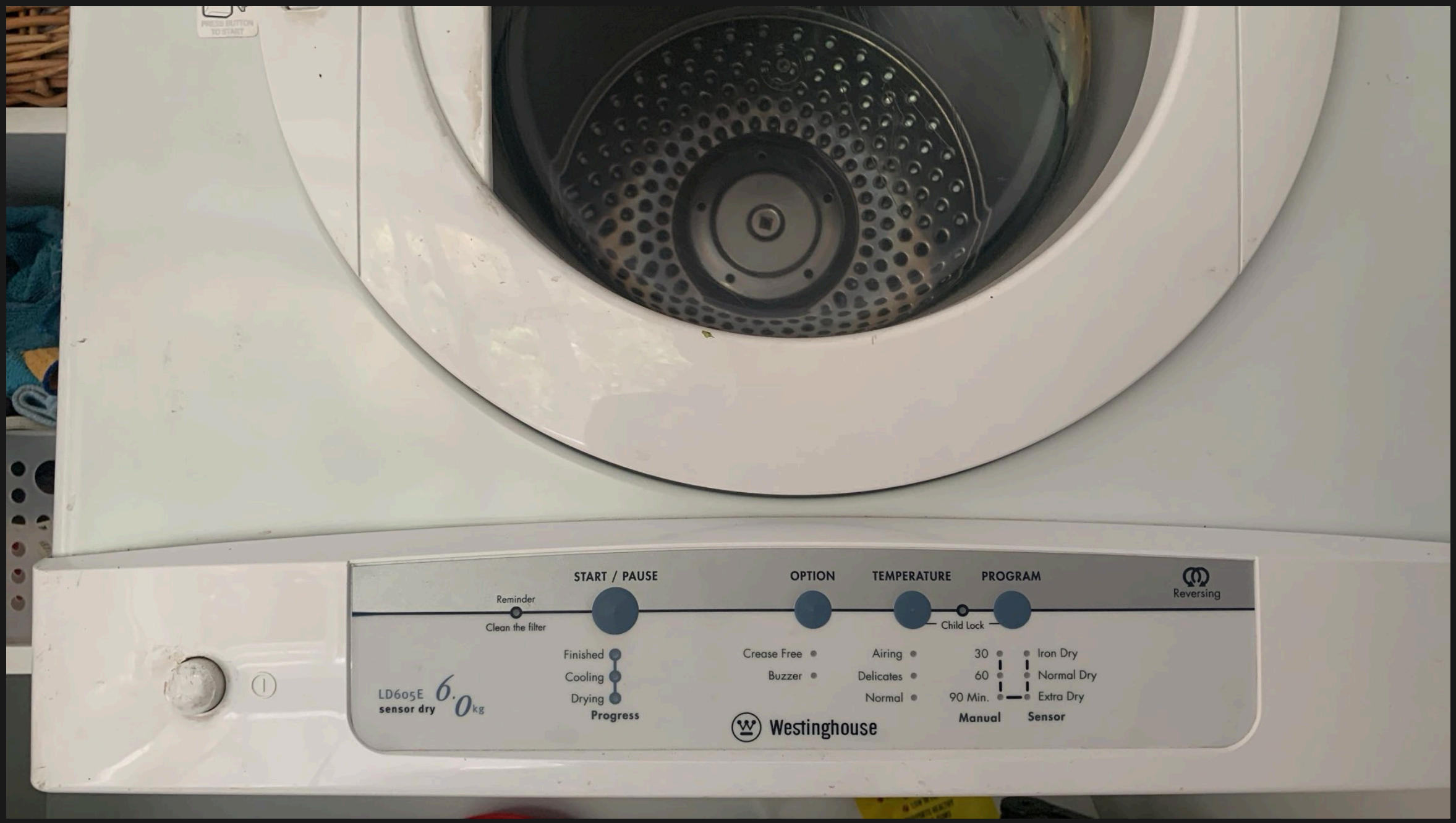
Spara

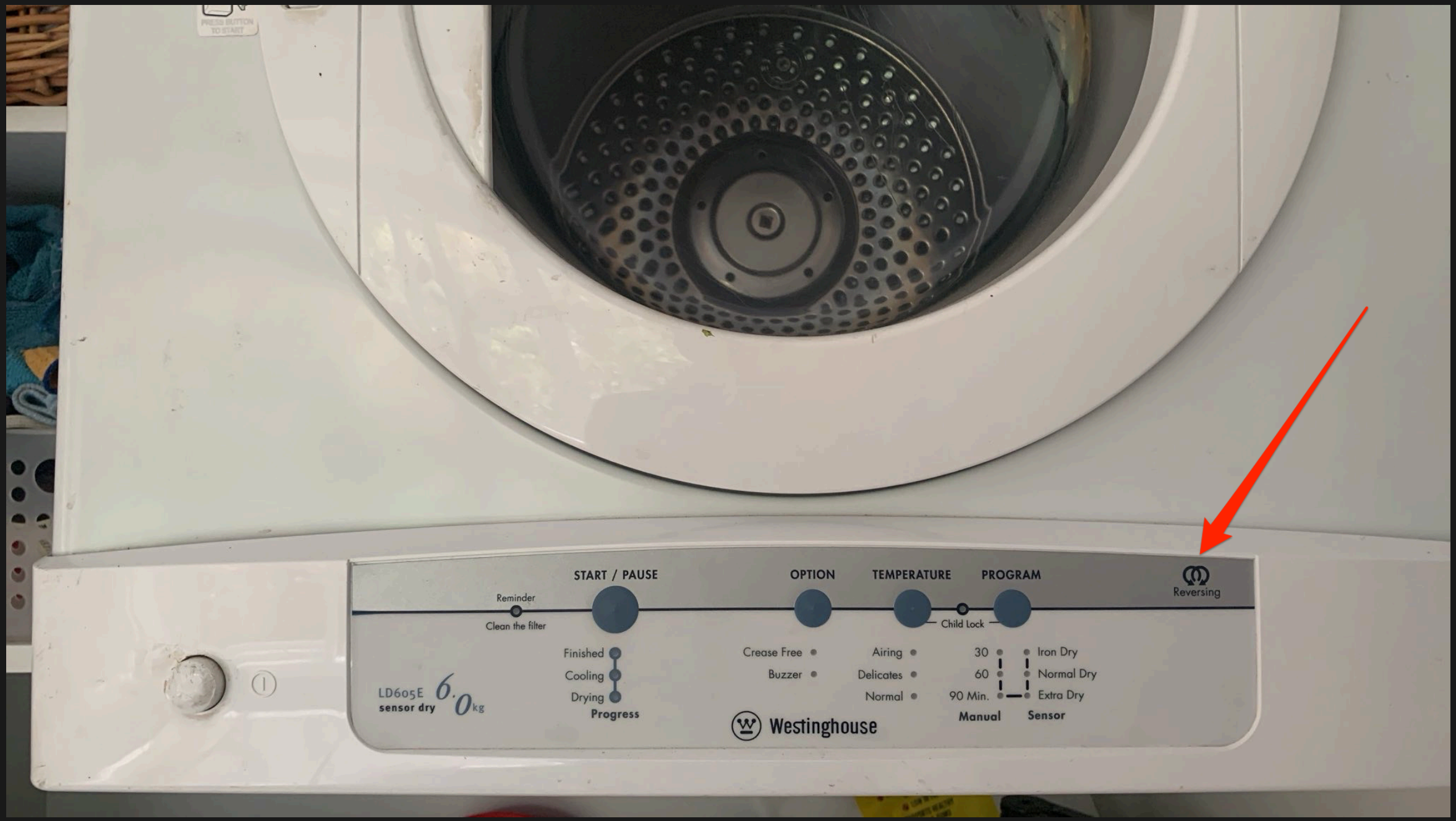
https://www.youtube.com/watch?v=8xZH0ccpOas


EWW12753 | ECOINVERTER | 7.5Kg/4.5Kg
Off
Vapour Refresh
Cottons
Mixed
Delicates
WashDry 60
Wool
Quick 18
Energy Saver
Temp
Spin
90° 60° 50°
40° 30°
1200 800 600
400
Dry Time Delay Start
Dryness Prewash
Vapour
Time Manager
Start/Pause
Add Clothes
WOOLMARK
APPAREL CARE
10
YEAR


Electrolux
U0012C
Copyright © 2012


DL11
R97
R69
C48
C49
DL12
R111
R113
R112
R10
C14
R23
R25
C13
R26
R31
R8
R16
C11
C22
R7
C5
U4
C19
R38
R40
R43


START / PAUSE
OPTION
TEMPERATURE
PROGRAM
Reversing
Reminder
Clean the filter
Child Lock
Finished
Cooling
Drying
Progress
Crease Free
Buzzer
Airing
Delicates
Normal
30
60
90 Min.
Manual
Iron Dry
Normal Dry
Extra Dry
Sensor
LD605E 6.0kg
sensor dry
Westinghouse

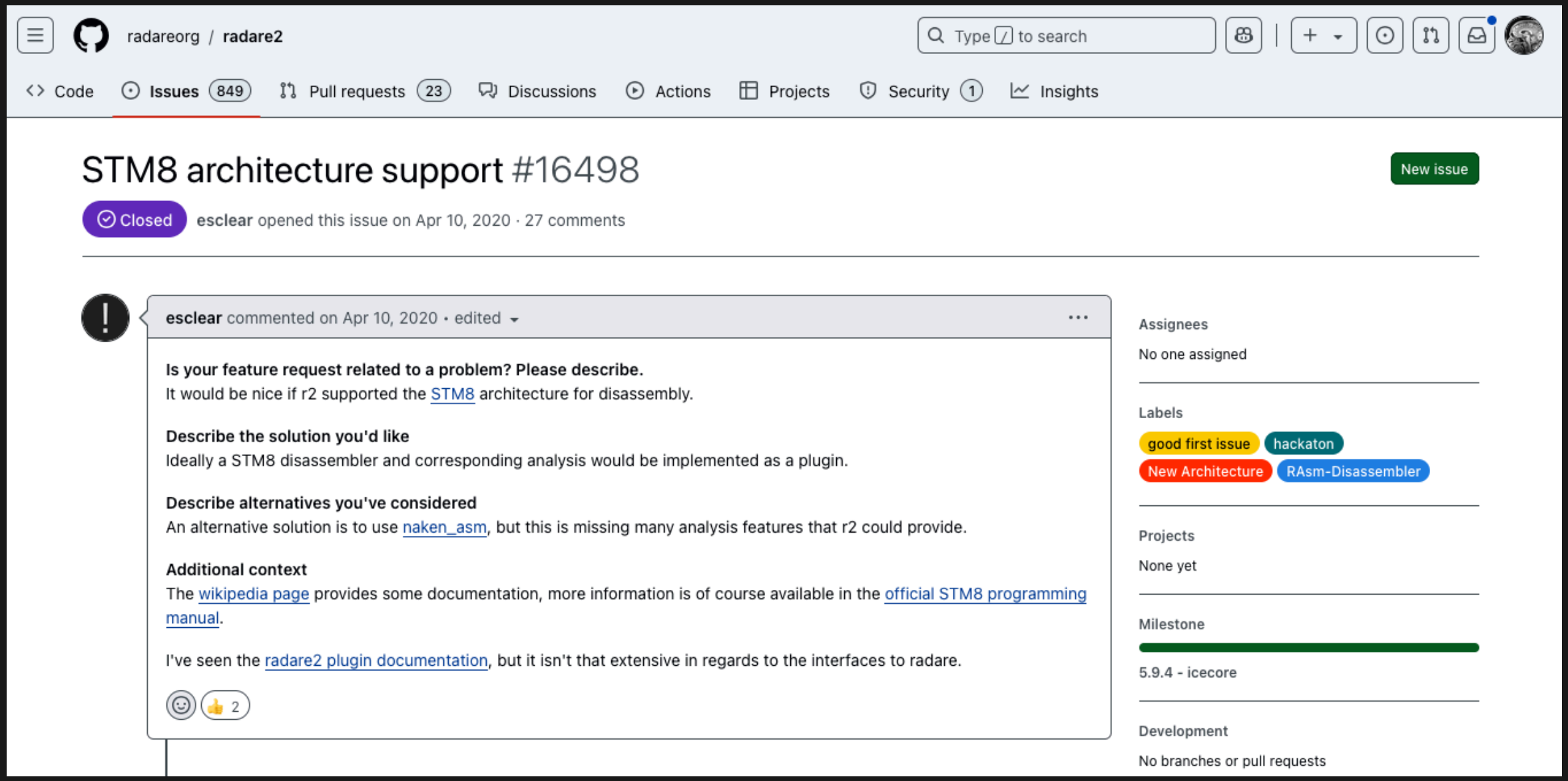

START / PAUSE
OPTION
TEMPERATURE
PROGRAM
Reversing
Reminder
Clean the filter
Child Lock
Finished
Cooling
Drying
Progress
Crease Free
Buzzer
Airing
Delicates
Normal
30
60
90 Min.
Manual
Iron Dry
Normal Dry
Extra Dry
Sensor
LD605E 6.0kg
sensor dry
Westinghouse


Reversing


RIGHT
TO
REPAIR

today's e-waste: a treadmill

Motivation



https://github.com/radareorg/radare2/issues/16498

Challenges

- No remote (RF) controller found. No sniffing 👃.
- No idea if this device **works** to begin with.
- Mains voltage == 💀 if not careful.
- Very limited time (and space).

Let's goooo anyway! 💪

Try Ghidra?

Ghidra's STM8 support

https://github.com/esaulenka/ghidra_STM8/issues/11

Low-level Error: Overlapping input varnodes #11

Open **brainstorm** opened this issue 2 weeks ago · 0 comments

brainstorm commented 2 weeks ago

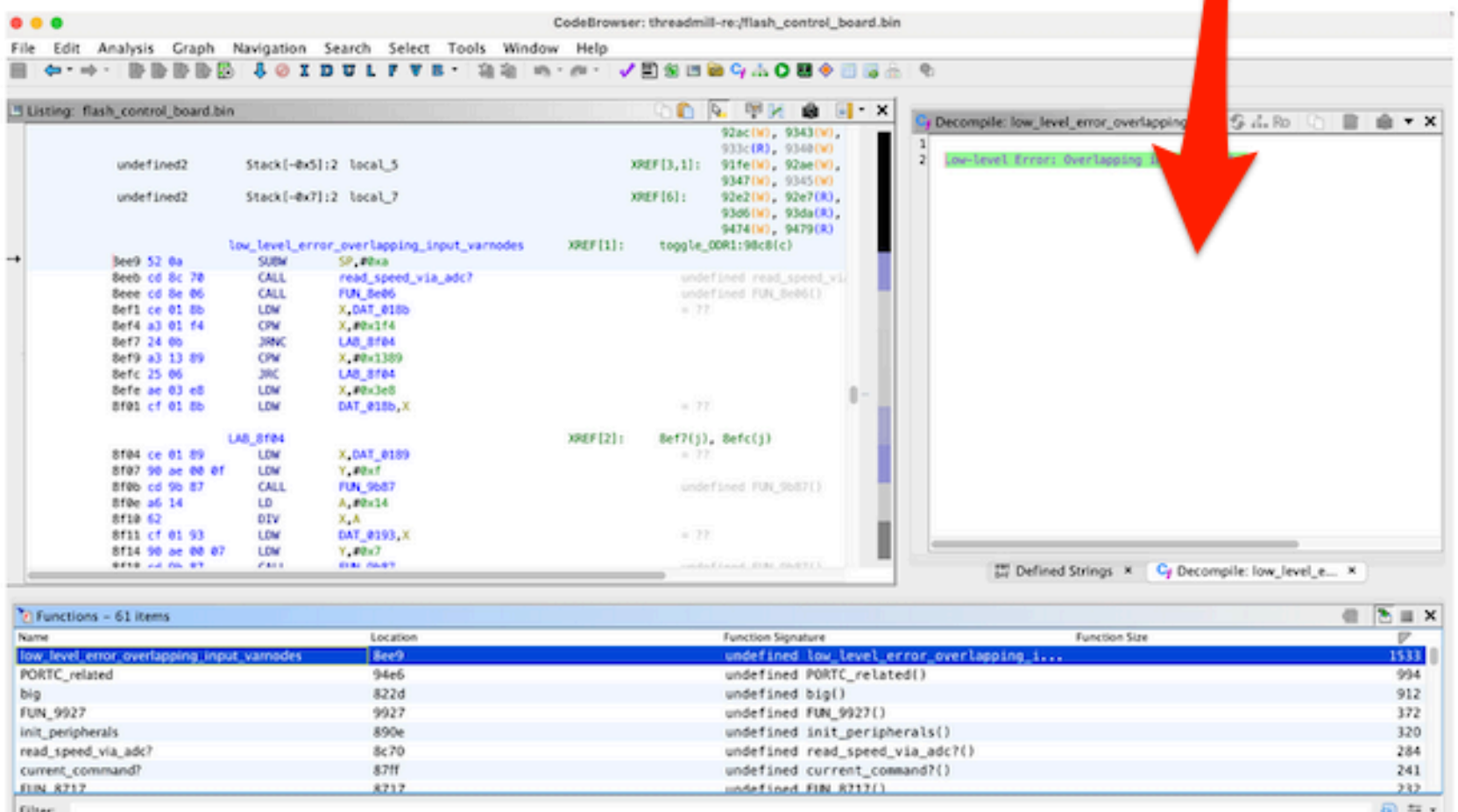
I'm reversing this STM8S003 firmware (flash.bin file):

https://github.com/brainstorm/treadmill-re/tree/master/control

The decompilation seems to fail (offset 0x8ee9)... any ideas on what might be causing this error?:

No upstream support... yet

Decompilation fails



Ghidra's ~~STM8~~ ST7 support

NationalSecurityAgency / **ghidra**

Type / to search

Code | Issues 1.4k | **Pull requests 270** | Discussions | Actions | Wiki | Security 4 | Insights

ST7 processor definition. #3631

Open **agatti** wants to merge 1 commit into `NationalSecurityAgency:master` from `agatti:stm_st7`

Conversation 40 | Commits 1 | Checks 0 | Files changed 12

agatti commented on Nov 15, 2021 — Contributor

This PR adds a CPU from ST Microelectronics (ST7) and a new family (STM). The plan is to add the other 4 processors in the family (ST6, STM8, ST9 and ST10) one at a time to ease eventual merging.

More information on the ST7: https://en.wikipedia.org/wiki/ST6_and_ST7 - this CPU might be considered legacy but from what I saw it surprisingly shows up in plenty of places. Its role is more or less taken over by STM8 these days, but according to the product page it is still in production.

The language file is based upon the latest available description on ST's own site - the document can be found here, free of charge.

mumbel reviewed on Nov 15, 2021 — View reviewed changes

https://github.com/NationalSecurityAgency/ghidra/pull

Binary Ninja?

https://github.com/Vector35/community-plugins/#binary-ninja-plugins

README MIT license

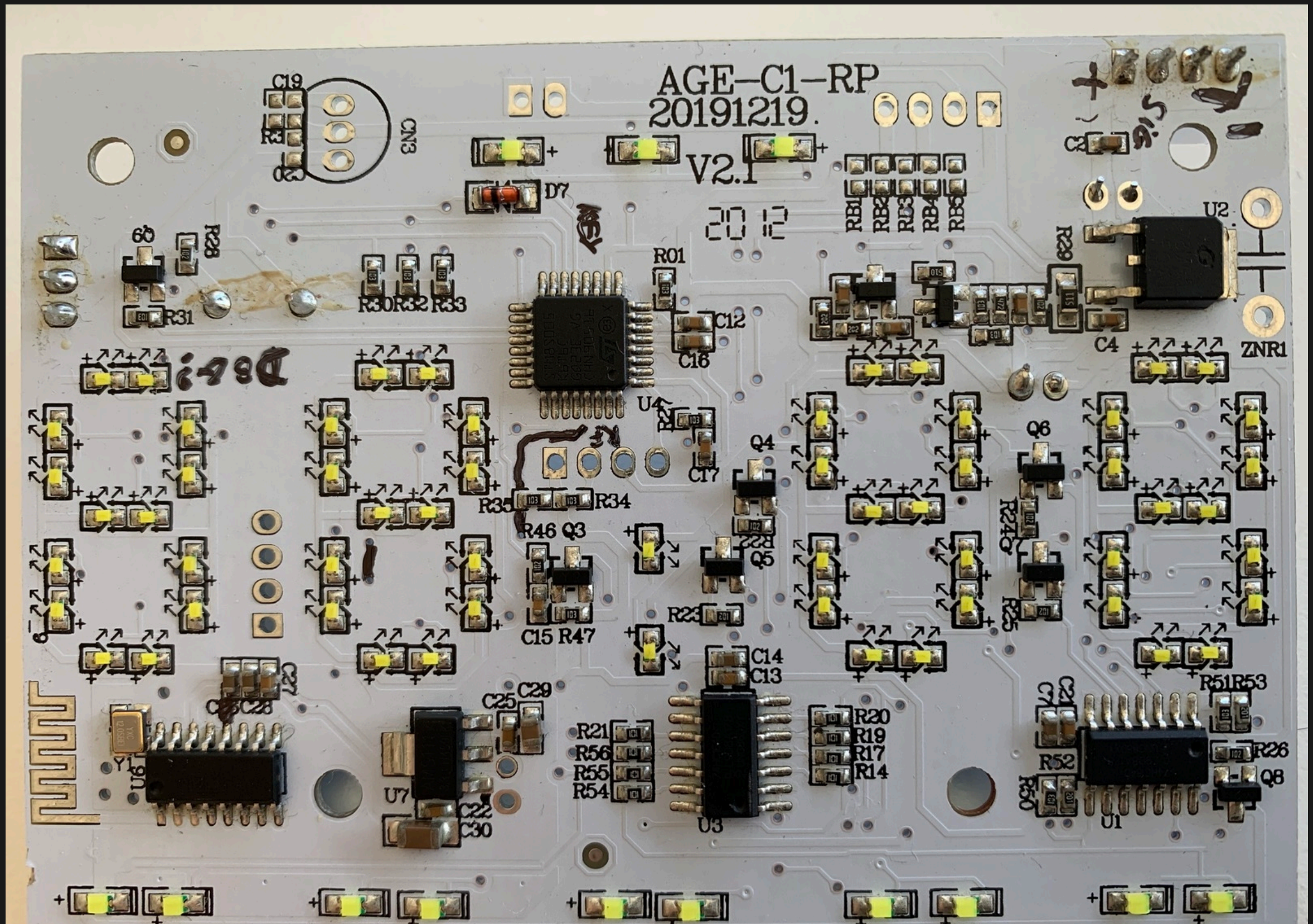
Binary Ninja Plugins

Nope :____(

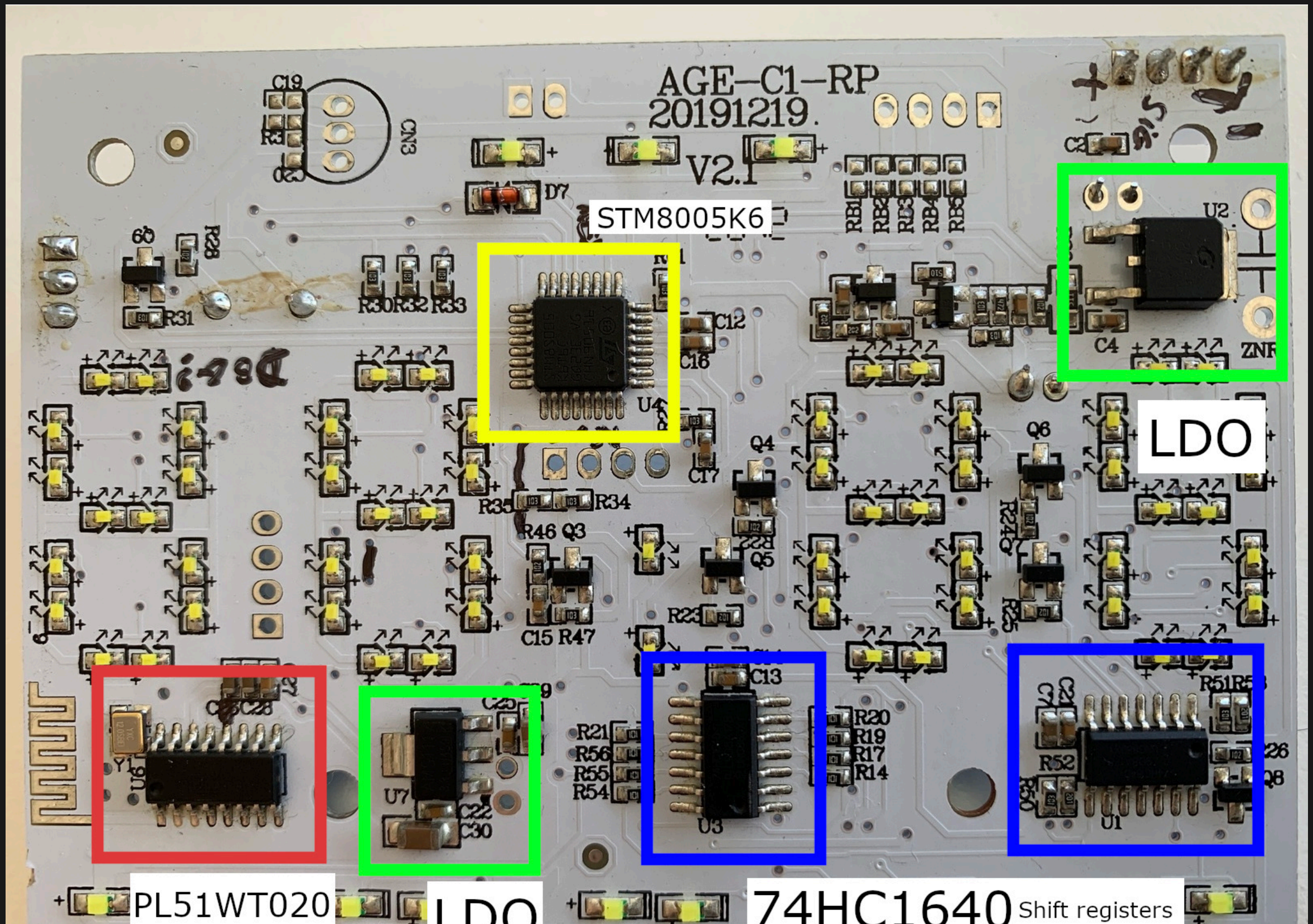
PluginName	Author	Description	Last Updated	
0CD	b0bb	Quality of life utilities for obsessive compulsive CTF enthusiasts.	python3	20
		Add information recovered by tool		

stm8 Highlight All Match Case Match Diacritics Whole Word

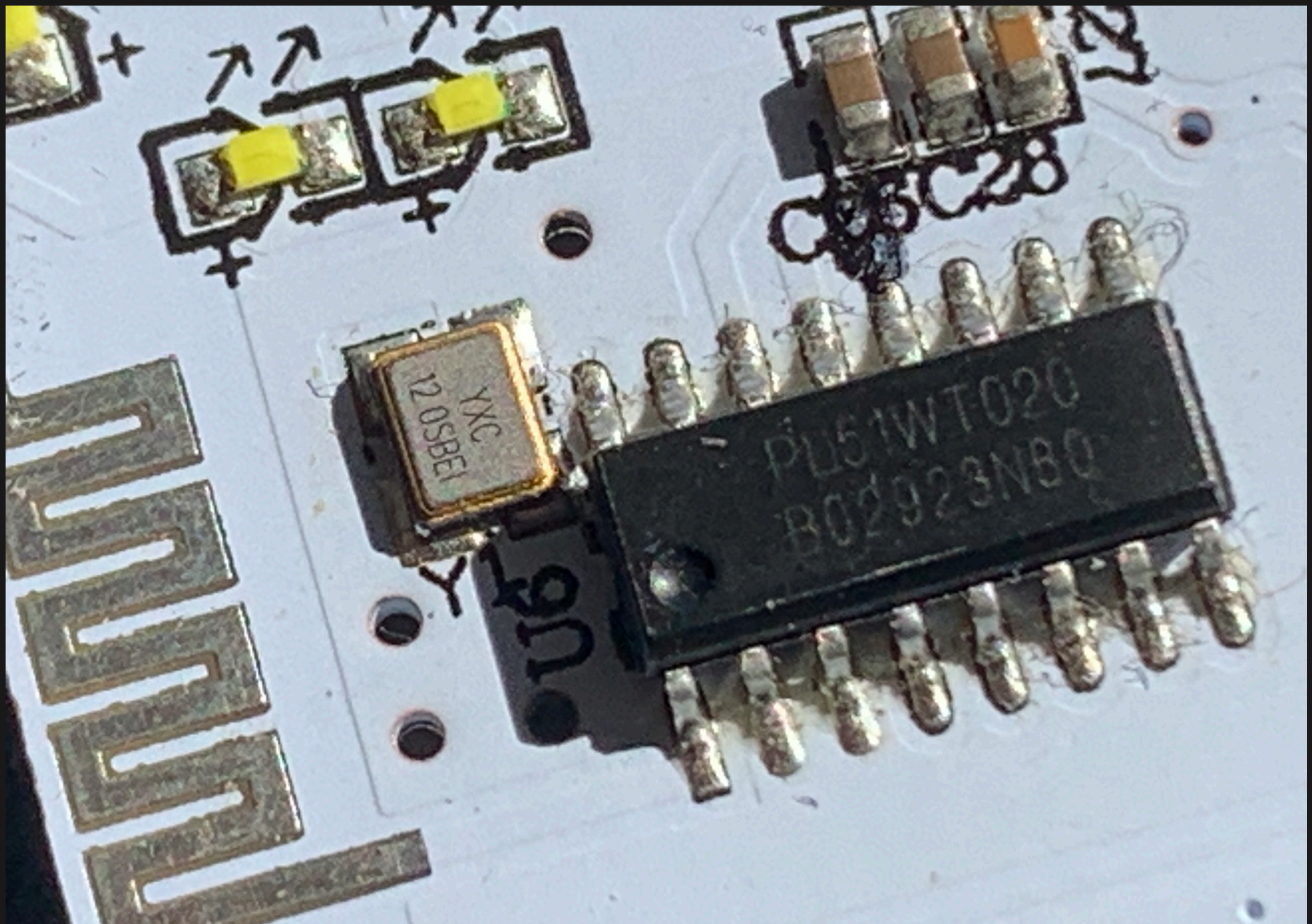
PCB: display


AGE-C1-RP
20191219.
V2.1
C19
CN3
D7
R28
R31
R30
R32
R33
R01
C12
C16
U4
RB1
RB2
RB3
RB4
RB5
U2
C4
ZNR1
Q4
Q6
R35
R34
R46
Q3
C15
R47
R23
Q5
C14
C13
C23
C28
C29
C25
Y1
U6
U7
C22
C30
R21
R56
R55
R54
U3
R20
R19
R17
R14
R51
R53
R52
R26
Q8
U1

Display PCB annotated

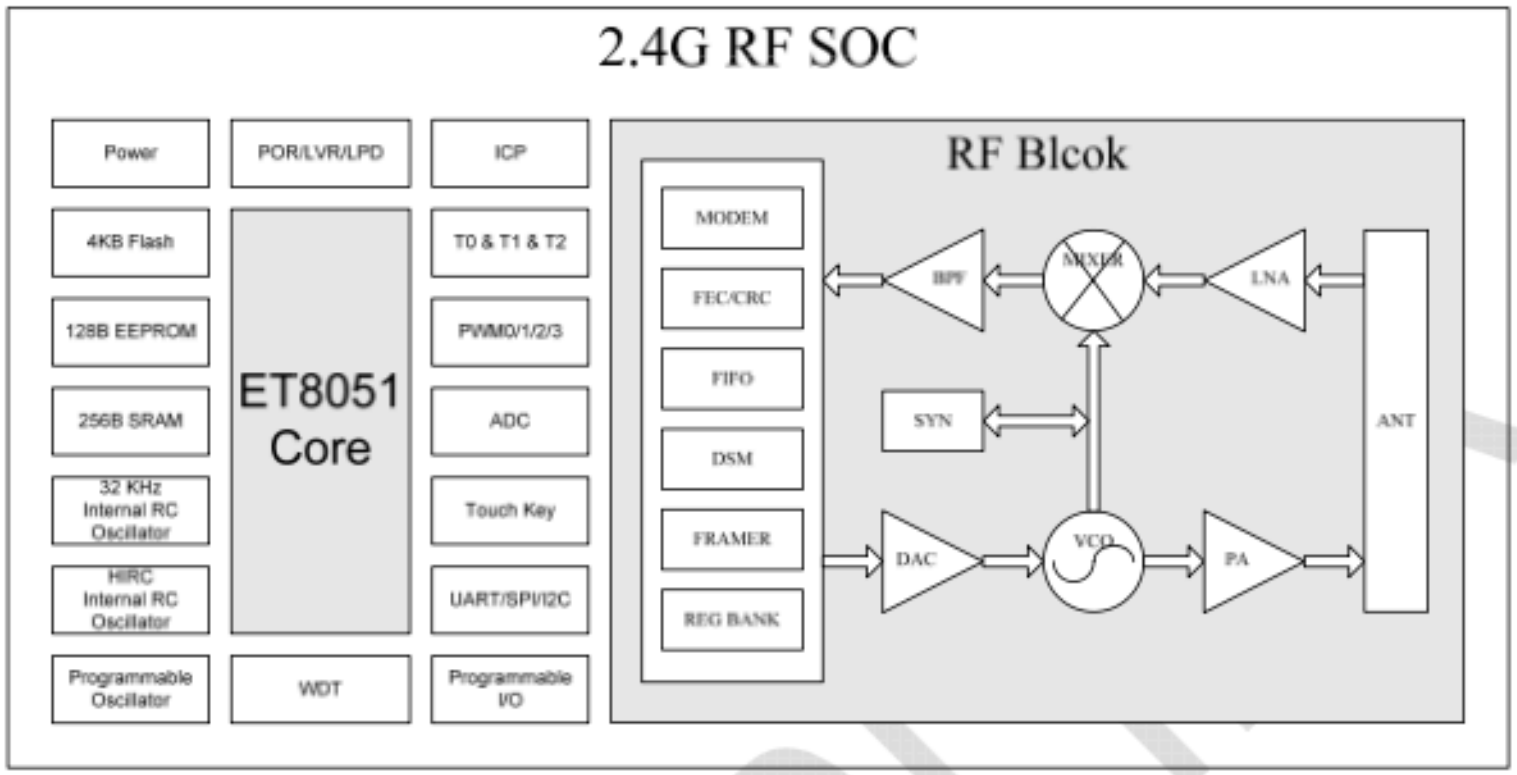


Display PCB: 2.4Ghz RF module



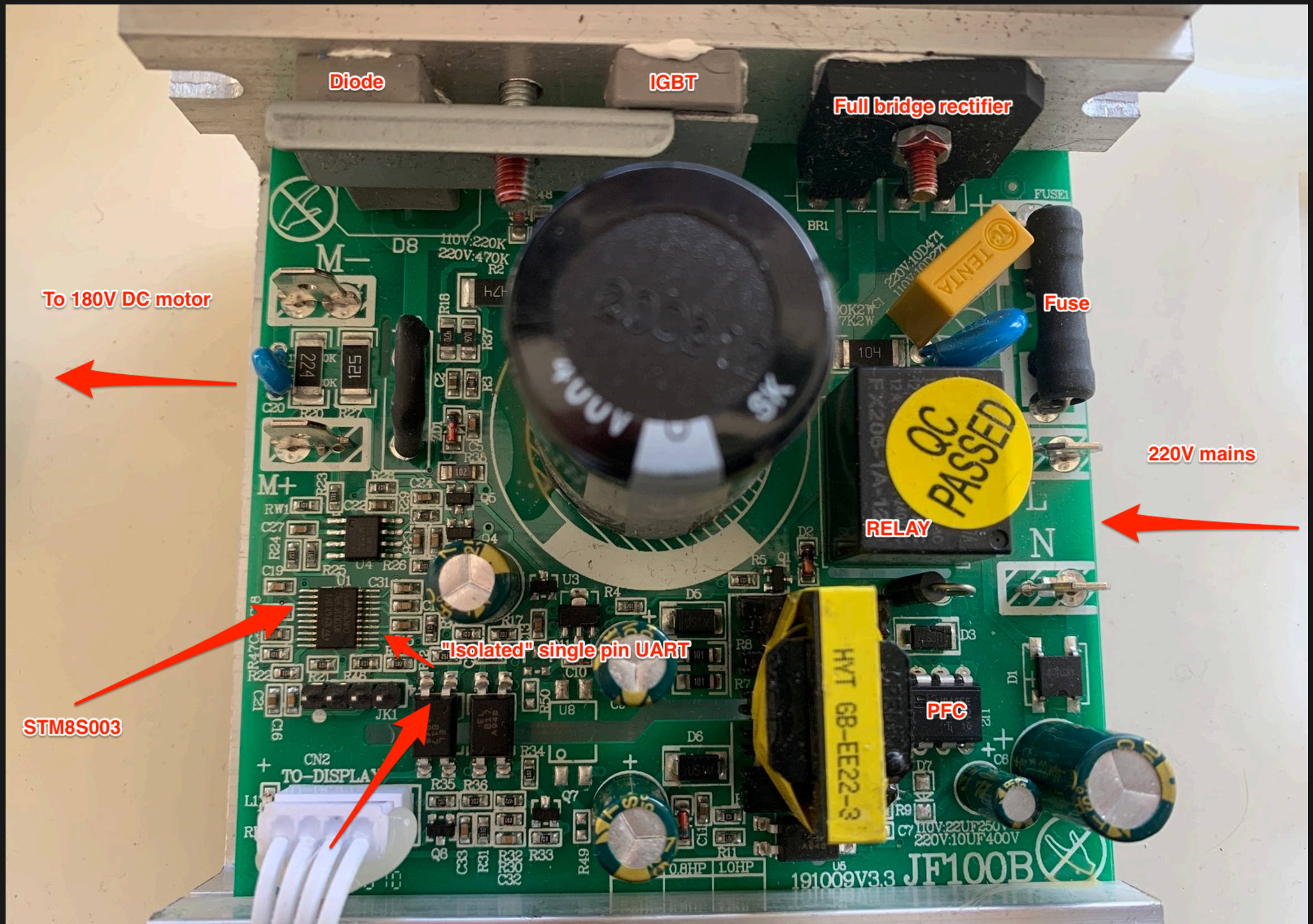
PL51WT020

5 Block Diagram

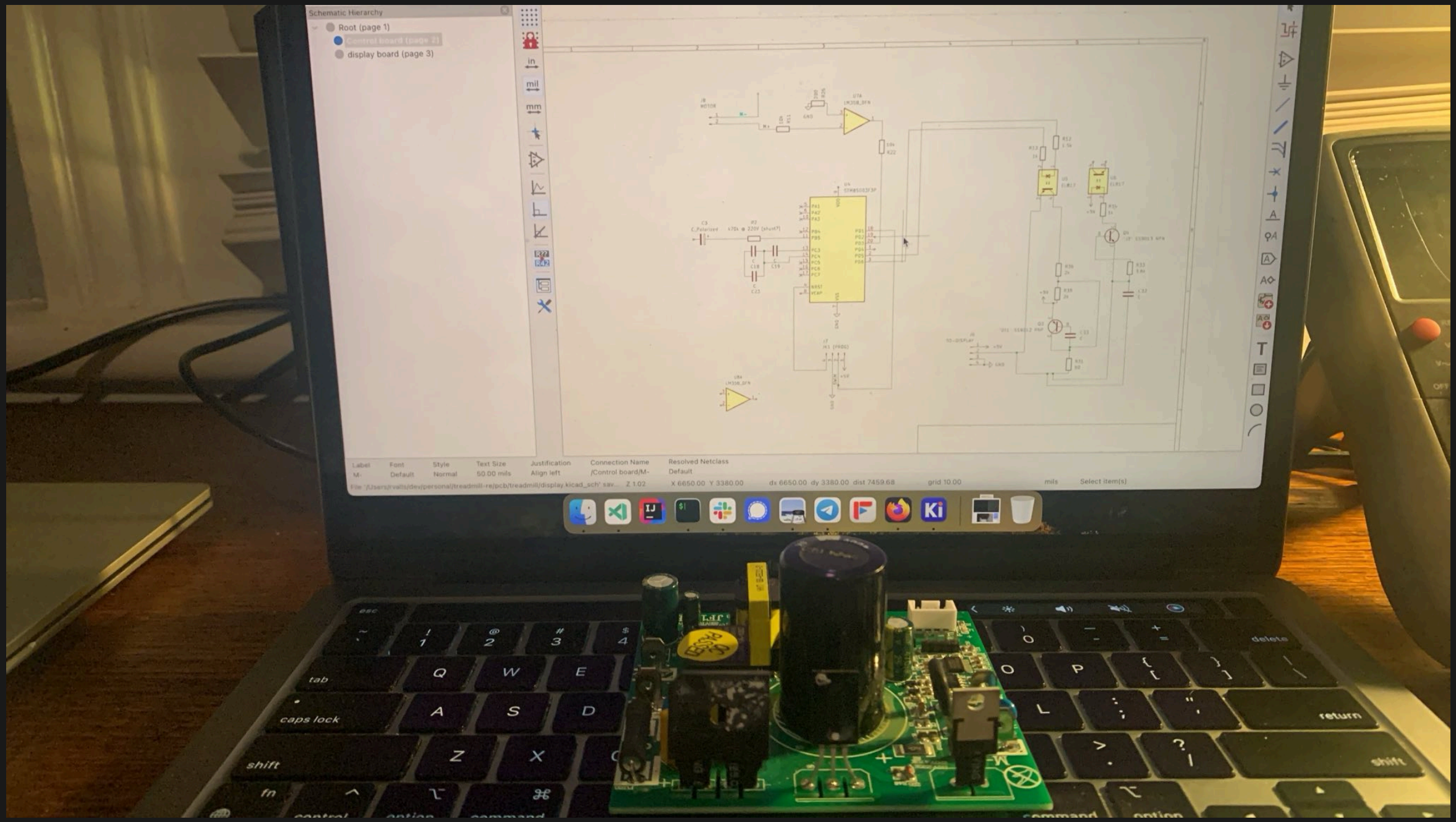


PCBs reversing: control

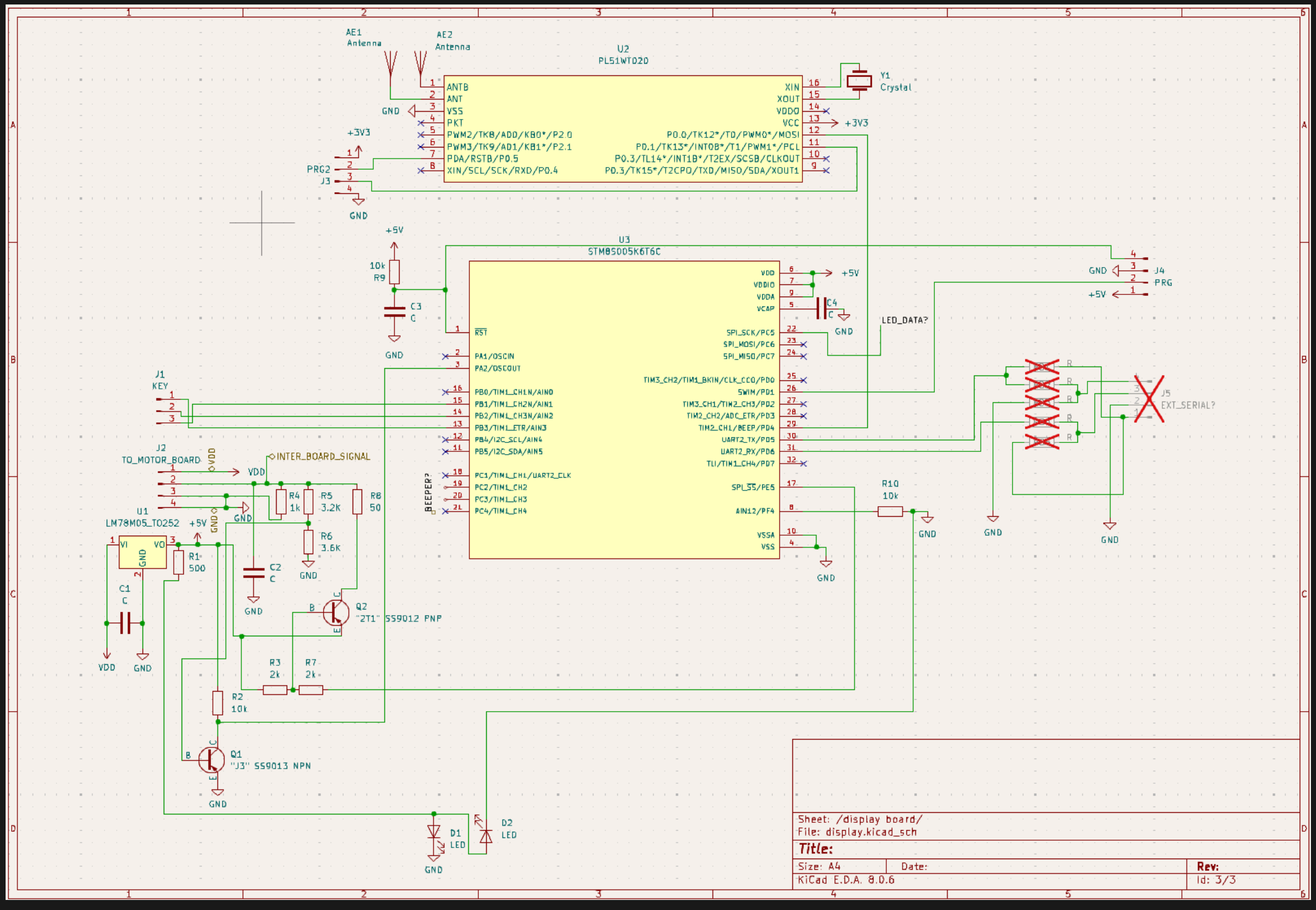
Control PCB components



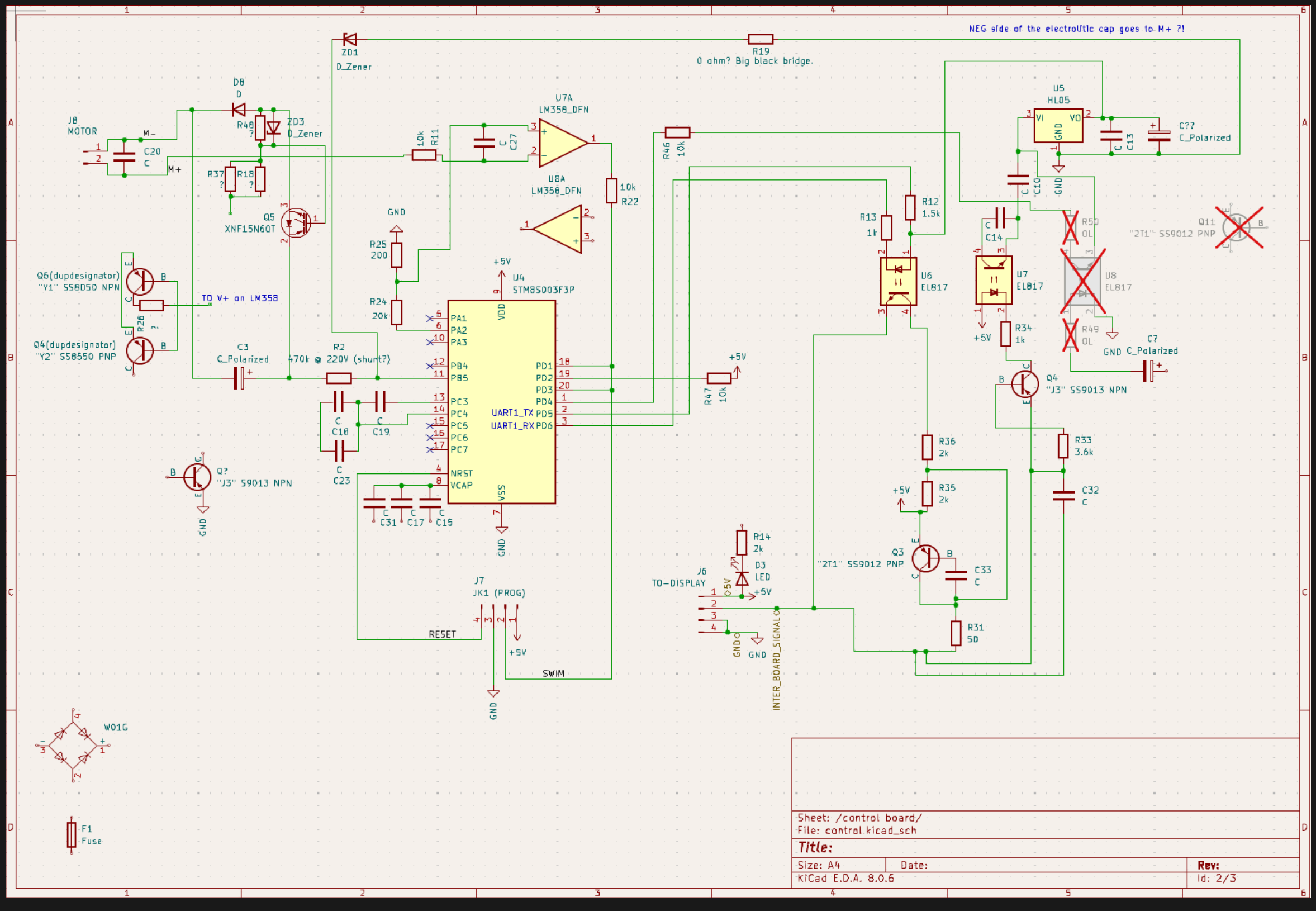
OSS PCB reversing with KiCad ...



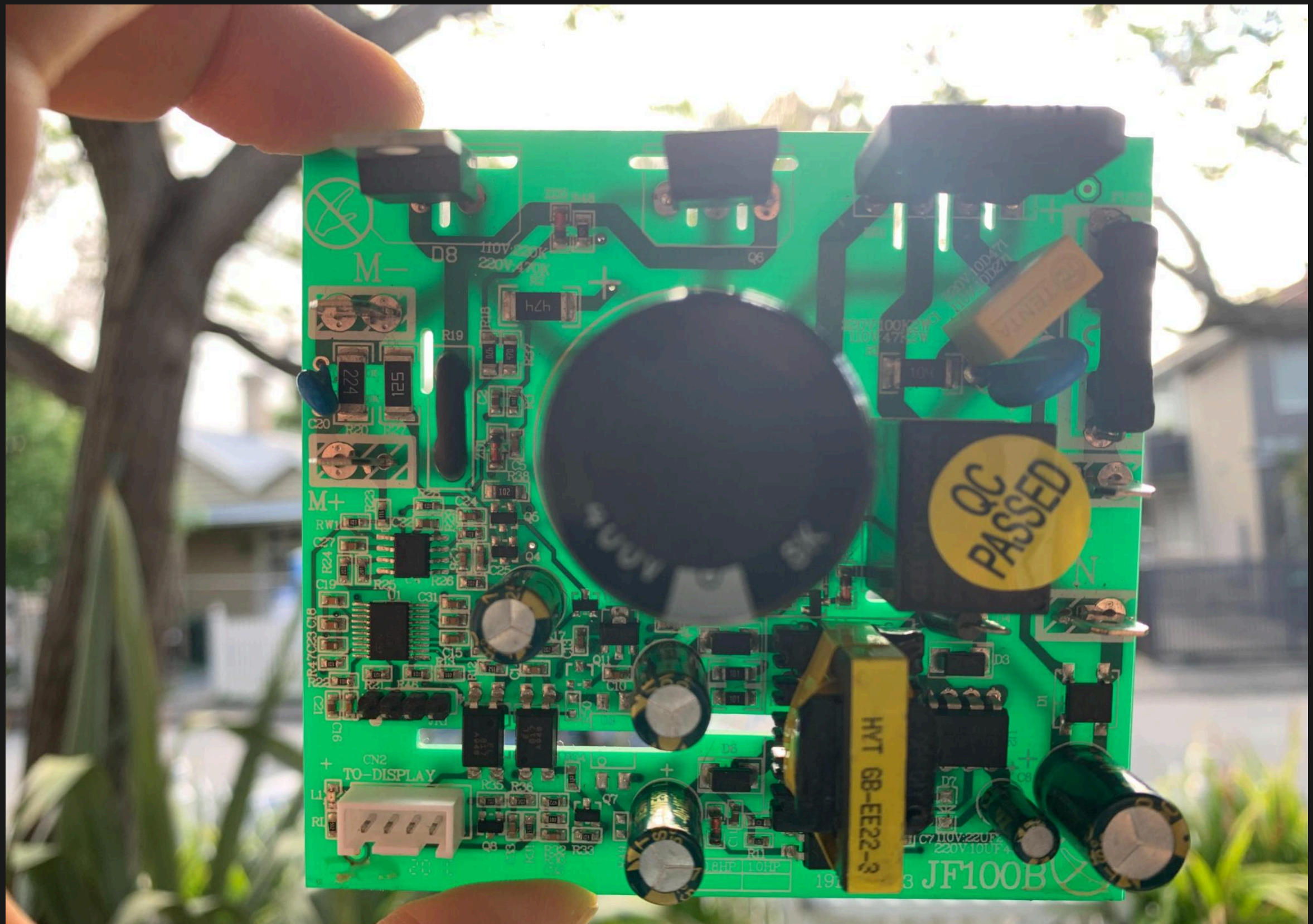
For the "display" board ...



... and the "control" board

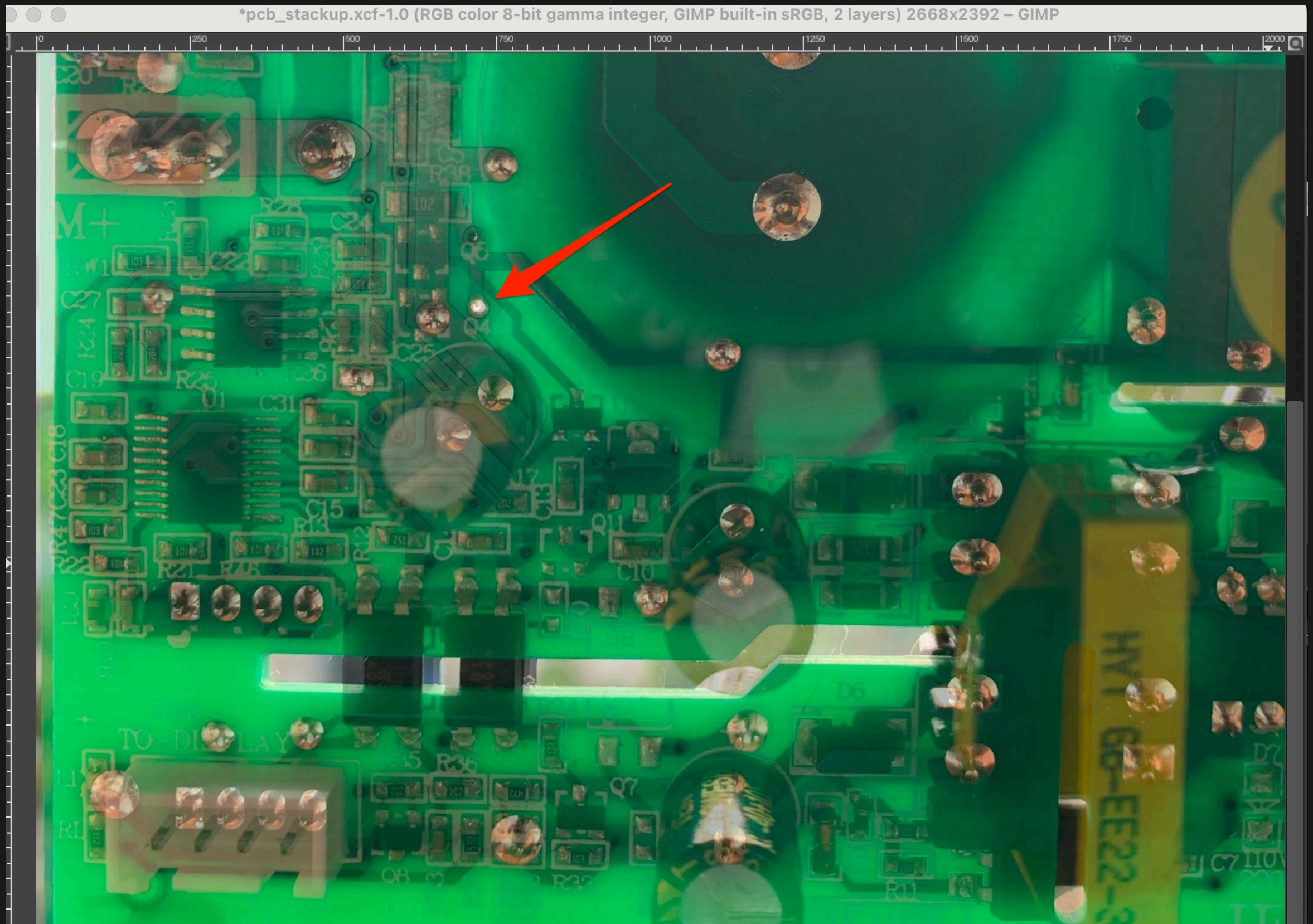


Boards against sunlight



Find landmarks and align...

… to meld them with GIMP



Add Python-Fu if you like

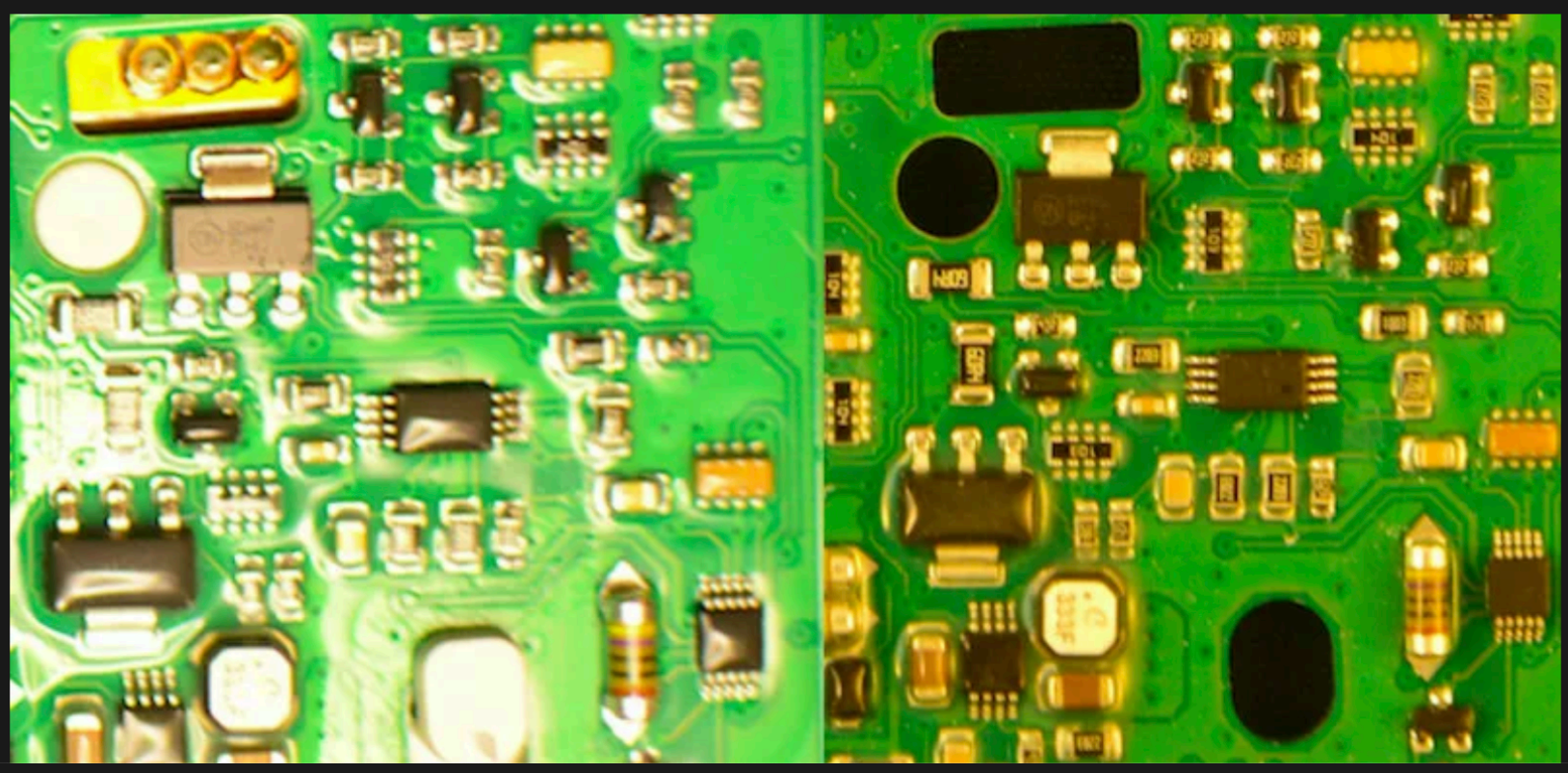
```python
#!/usr/bin/env python2

import time
from gimpfu import *

def opacity_slideshow(image, drawable):
  layer = image.layers[0]

  for opacity_pct in range(0, 101, 3):
    pdb.gimp_layer_set_opacity(layer, opacity_pct)
    time.sleep(0.1)
    gimp.displays_flush()

register(
    "python_fu_opacity_slideshow",
    "Opacity slideshow",
    "Turn opacity to see through between layers progressively",
    "brainstorm",
    "Open source (GPLv3)",
    "2024",
    "<Image>/Filters/PCB Reversing/Opacity slideshow",
    "*",
    [],
    [],
    opacity_slideshow)

main()
```

PERFECT PHOTOGRAPHS FOR PCB REVERSE ENGINEERING

by: Jenny List

19 Comments

September 1, 2020



https://hackaday.com/2020/09/01/perfect-

radare2 and LLMs: decai.r2js

```
const curlcmd = `curl -X POST -s https://api-inference.huggingface.co/models/${hfModel} \
    -H "Authorization: Bearer ${hfKey}" \
    -H "Content-Type: application/json" \
    -H "x-wait-for-model: true" \
    -d '${payload}'`.replace(/\n/g, "");

//if (decaiDebug) {
//     console.log(curlcmd);
//}

const res = r2.syscmds(curlcmd);
// Debug response instead of request
if (decaiDebug) {
    console.log(JSON.stringify(res, null, 4));
}

try {
    return JSON.parse(res).generated_text;
    //return JSON.stringify(res, null, 4);
} catch (e) {
    console.error(e);
    console.log(res);
}
return "error invalid response";
}
```

pdc: PseuDo C

```
[0x0000822d]> pdc
 // callconv: a reg (x, y, a);
int fn.big (int x, int y) {
    loc_0x0000822d:
        // anol @ 0x0000822d: int big(int x, int y) {
        // CALL XREF from fn.entry @ 0x9aa4(x)
        // FOCUS1 This function appears to handle UART communication by prepari
        // ng and sending data packets with various status flags and parameters,
        // while also managing state transitions based on multiple condition chec
        // ks.
        sp -= 0x0e
        a = [0x76]
        if (res == 0) goto 0x00008236 // unlikely
        goto loc_0x00008233;
    loc_0x00008236:
        // CODE XREF from fn.big @ 0x8231(x)
        a = [0x78]
        a --
        if (res == 0) goto 0x0000823e // unlikely
        goto loc_0x0000823b;
    loc_0x0000823e:
        // CODE XREF from fn.big @ 0x8239(x)
        [0x75] = a
        [0x77] = a
        [0x78] = a
        [0x0073] = 0x0a
        [0x0108] = 0x01
        a = [0x61]
        [0x3c] = a
        a --
        if (res != 0) goto 0x00008256 // unlikely
        goto loc_0x00008253;
    loc_0x00008256:
        // CODE XREF from fn.big @ 0x8251(x)
        a --
        if (res == 0) goto 0x0000825e // unlikely
        goto loc_0x00008259;
```

"Simple" decai -d

```
[0x0000822d]> decai -d
int big(int x, int y) {
    unsigned char uart_data = 0;
    unsigned char temp_value = 0;
    unsigned short energy_value = 0;
    unsigned char status_flags = 0;
    unsigned char counter = 0;
    int i;

    unsigned char sp[14];

    if (!mem[0x76]) {
        if (!--mem[0x78]) {
mem[0x75] = mem[0x77] = mem[0x78] = 0;
mem[0x73] = 0x0A;
mem[0x108] = 1;

temp_value = mem[0x61];
mem[0x3C] = temp_value;

if (--temp_value) {
    if (!--temp_value) {
        if (mem[0x62] & 0x01) {
            mem[0x8B] = 1;
        }
        if (mem[0x62] & 0x02) {
            mem[0x8B] = 2;
        }

        status_flags = mem[0x62] | 0x04;
        sp[0x0E] = status_flags;

        if (status_flags == 0x04) {
            mem[0x10A] = 1;
        } else if (mem[0x8C] > 1) {
            mem[0x10A] = 0;
        }
```

Improved decai -d

```
char big_function(void) {
    uint8_t tx_buffer[0x80]; // Buffer for transmit data

    // Check initial state
    if (g_state_byte_76 != 0) {
        // Initialize transmission
        g_state_byte_76 = 0;
        g_tx_buffer = 0xF7;
        g_uart_control = 0xF8;
        g_param_40 = 1;
        g_param_41 = 2;
        g_param_42 = g_mode_3c;
        g_param_43 = 0;

        // Handle different modes
        uint8_t mode = g_mode_3c - 1;
        if (mode == 0) {
            // Mode 0 handling
            g_packet_size = 0x10;
            g_tx_data[0] = g_value_190;
            g_tx_data[1] = 0x34;
            g_tx_data[2] = mode | 4;
            g_tx_data[3] = g_value_18f;
            g_tx_data[4] = g_value_18d;
            g_tx_data[5] = g_value_18e;
            g_tx_data[6] = g_value_192;
        }
        else if (mode == 1) {
            // Mode 1 handling - larger packet
            g_packet_size = 0x1E;

            // Fill packet with various status values
            g_tx_data[0] = g_status_a2;
            g_tx_data[1] = g_status_a3;
            // ... (remaining packet construction)
```

Calling conventions? decai -s

```
[0x0001941c]> decai -s
'afs const char* r_str_bool(int64_t arg1)
[0x0001941c]> 'afs const char* r_str_bool(int64_t arg1)
[0x0001941c]> decai -d
const char * str_bool (int64_t arg1) {
    int boolean_res = arg1 & 1;
    const char * false_str = "false";
    const char * true_str = "true";
    boolean_res &= boolean_res;
    const char * result = (boolean_res != 0)? true_str : false_str;
    return result;
}
[0x0001941c]>
```

STM8 support radare2 PRs

https://github.com/radareorg/radare2/pull/23552
https://github.com/radareorg/radare2/pull/23541
https://github.com/radareorg/radare2/pull/23526
https://github.com/radareorg/radare2/pull/23517
https://github.com/radareorg/radare2/pull/23305
https://github.com/radareorg/radare2/pull/23411
https://github.com/radareorg/radare2/pull/23251
https://github.com/radareorg/radare2/pull/23542
https://github.com/radareorg/radare2/pull/23030
https://github.com/radareorg/radare2/pull/23244
https://github.com/radareorg/radare2/pull/23486
https://github.com/radareorg/radare2/pull/23509

r2svd

```
1> UART|
 - 0x00005231  reg.UART1.DR
   0x00005232  reg.UART1.BRR1
   0x00005233  reg.UART1.BRR2
   0x00005234  reg.UART1.CR1
   0x00005235  reg.UART1.CR2
   0x00005236  reg.UART1.CR3
   0x00005237  reg.UART1.CR4
   0x00005238  reg.UART1.CR5
   0x00005239  reg.UART1.GTR
   0x0000523a  reg.UART1.PSCR
   0x00005230 UART1
   0x00005231 UART1.SR: UART1 status register
UART1.DR: UART1 data register
   0x00005232 UART1.BRR1: UART1 baud rate register 1
   0x00005233 UART1.BRR2: UART1 baud rate register 2
   0x00005234 UART1.CR1: UART1 control register 1
```

cmsis-svd maintainers 

VincentDary commented 3 weeks ago — Member

@posborne, can you add **@brainstorm** to the Pypi, since he contributes to the integration of cmsis-svd to radare2-extras, see radare2-extras issues #364, this is a high value for the visibility of this project.

❤️ 1 🚀 1

PRs against radare2-extras and r2ai

https://github.com/radareorg/r2ai/pull/66

https://github.com/radareorg/r2ai/pull/65

https://github.com/radareorg/radare2-extras/pull/370

https://github.com/radareorg/radare2-extras/pull/364

Conclusion

- r2 is top notch if you need to do a STM8 analysis.
- Decompilation via LLMs is surprisingly good.
- This is just the start, no RAG, r2 "zignatures", fine tuning...
- Other radare2 areas were improved too as a result of this work: XREFs, r2 projects, etc...
- Most important: Me and pancake had tons of fun with this :)

I have a dream...

- Dump (automate STM8 glitching too if protected).
- Generate r2 "zignatures" database of the ST's official stdlib, fine tuned with a local LLM model.
- Output a C project with all relevant files, peripherals, makefile included.
- Requiring minimal manual tweaking needed on the output code.
- Compile it all with SDCC and free your STM8 powered devices.

... and a competition!

Want to win r2con 2024 swag?

Submit an issue to the repo 👇 with a writeup detailing which secret commands move the treadmill motor.