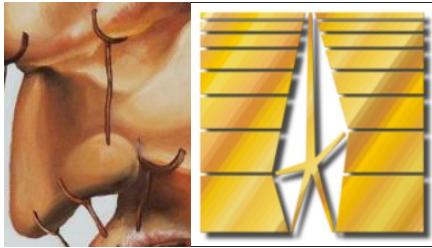


eNoze V2

Второй релиз аппаратной (HW) и программной (SW/FW) части
проекта графен – электронный нос

20-May-2024

Igor Bocharov



Цель.....	1
Доработка нагрева.....	2
Периферия и датчики.....	4
Датчик положения двигателя.....	5
Консольный клиент.....	6
Алгоритм.....	6
Интерфейсы.....	7
Модель температуры от сопротивления.....	10
Статический накал V3.....	11
Сделано на V2.....	12
Планы на V3.....	13
EI Noze - Modbus –V2.....	14
Modbus function codes.....	15
EI Noze - Modbus – V1.....	16
Сборка MCU проекта.....	18
Выборочно схема платы V2.....	19
PCB V2 с доработками.....	20

Цель

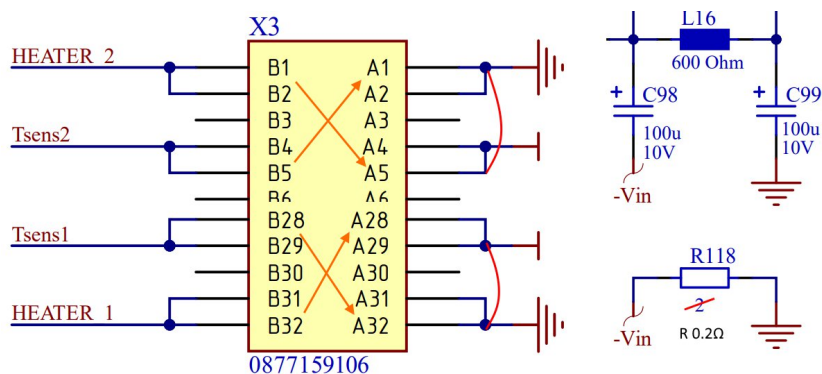
- 1. Изготовление HW и соответствующей прошивки FW, **совместимой** с первой версией V1 платы и верхнего ПО
- 2. Добавление периферии: датчиков температуры, 3х ходовой клапан, мотор, датчик положения мотора
- 3. Софт верхнего уровня

Проблемы V1

- 1. Нагрев - дисбаланс каналов нагревателя, неясна точность нагрева и температура на кристалле
- 2. Нет сенсоров температуры и влажности, выхода на мотор
- 3. Верхний софт не убедителен

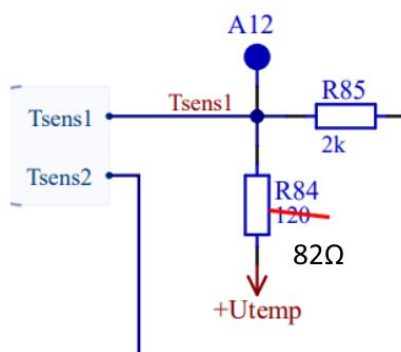
Доработка нагрева

Перепутаны (перекрещены) земли накала и сенсора, перекус земель достаточно для ощутимой разницы в mV в напряжении на термо сенсоре и неправильного измерения сопротивления единиц ом. В новой ревизии поменять земли или на плате, но логичнее на плате датчика. Также сильнее соединить земли - занизить R118 до 0.2 Ом.



Очень маленькое и шумное напряжение на термо сенсоре датчика, 20-50mV. Поставить R84/R103 на **82Ω**. Это при +Utemp 1.090V даст: 50mV/13ma@4Ω/25°C и 250mV/10mA@24Ω@100°C. При Vref=2.5V это как раз полный размах оцифровки для Ki=10 предусилителя.

На V3 схему усиления Rsens в целом надо проинспектировать и пересмотреть, сейчас точность сомнительна. Все задающие резисторы поставить 1% точности, и так уже в софте подогнаны коэффициенты. Изготовителям датчика надо улучшать серийность, такой разброс сопротивлений накала вынуждает использовать шим на 5В.



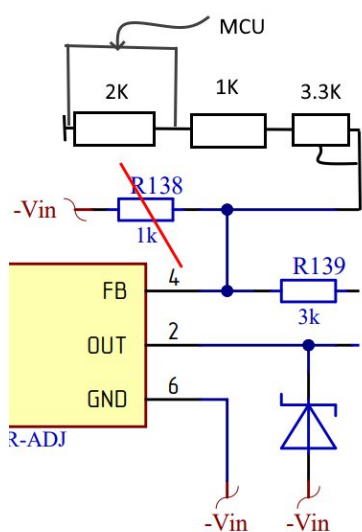
Ноль-перемычки на разъёме и резисторы подачи напряжения на Rsens



Программная реализация ШИМ на накале излишня и вредна, PWM уже есть в DC/DC. Из-за лёгкой подложки редкий ШИМ нагрева модулирует сигнал. В V2 накал занижен до 3В через подбор резисторов FB DC/DC и сделан обход нижнего резистора закорачиванием через mosfet дискретным сигналом с MCU PC14. Это даёт два напряжения 1.8-2.4 и 2.1-4.9В, можно регулировать подстроечником, что должно давать примерно 45 и 95 на графене. Нижнее плечо выбрано что бы случайно при обрыве не подать Vin на графен. Можно заменить R119 для LED2 на 300Ом, будет ярче.

Алгоритм. По умолчанию – старый алгоритм, в котором при $T_s > 60$ включается V2_Cmd_Heat_Boost. Установка V2_Cmd_Heat_Still > 0 *выключает* софт алгоритм и на накале будет постоянное напряжение от DC/DC, которое вручную можно ставить больше/меньше через V2_Cmd_Heat_Boost. Катушка V2_Cmd_HEAT также запускает 301h старого алгоритма, равно как V2_Cmd_START запускает измерение 300h. Тест режим всё выключает.

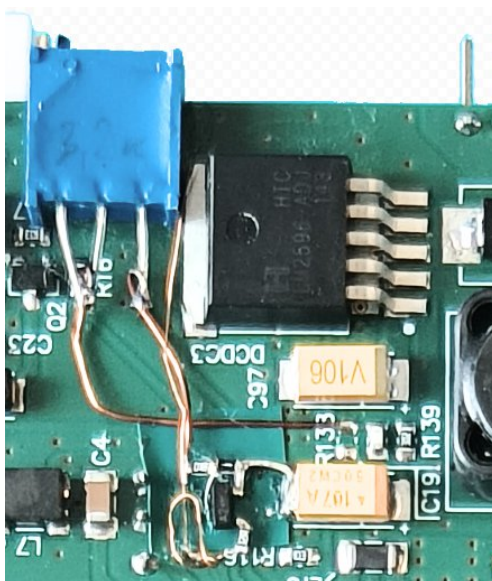
На V3 переделать накал на DC/DC управляемый с DAC с MCU через FB с учётом низких напряжений порядка 1-3В. Сделать по одному DC/DC на каждый канал т.к. они сильно разные (см прил.2).



$$V_{out} = V_{ref} \left(1.0 + \frac{R2}{R1} \right)$$

$$R2 = R1 \left(\frac{V_{out}}{V_{ref}} - 1.0 \right)$$

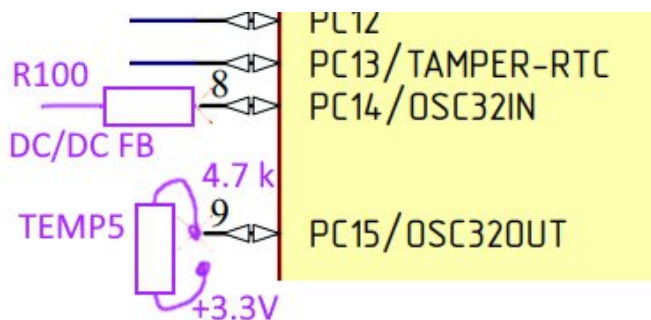
Where $V_{ref} = 1.23 \text{ V}$, $R1$ between 1.0 k and 5.0 k



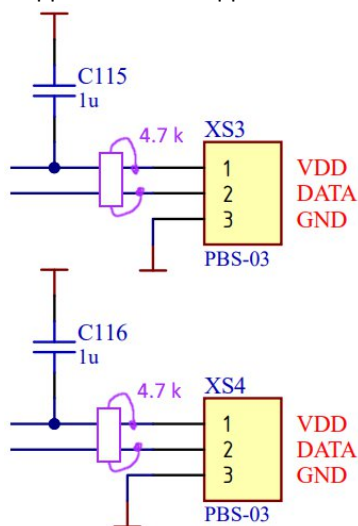
Подпайка к свободным ногам контроллера

PC14 – активация boost_mode для DC/DC через 100R

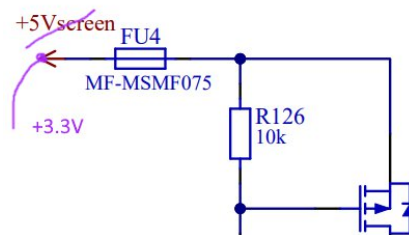
PC15 – однопроводной интерфейс датчика 5 с подтяжкой на 3V3s, разъём впаять отдельно



Подтяжка линии данных вверх для датчиков HMDT



Занижение питания датчика BME для линии RX/TX Screen, переход в режим i2c и подключение 2х датчиков.



Плата датчиков BME может быть толерантна к 5V и тогда занижение не обязательно. Инициализация этих датчиков при старте и при выходе из Test.

Периферия и датчики

Всего реализовано девять каналов измерения погоды в виде температура/влажность/давление. Дискретные входы для датчиков положения вала. Дискретные выходы на мотор и клапан 24V.

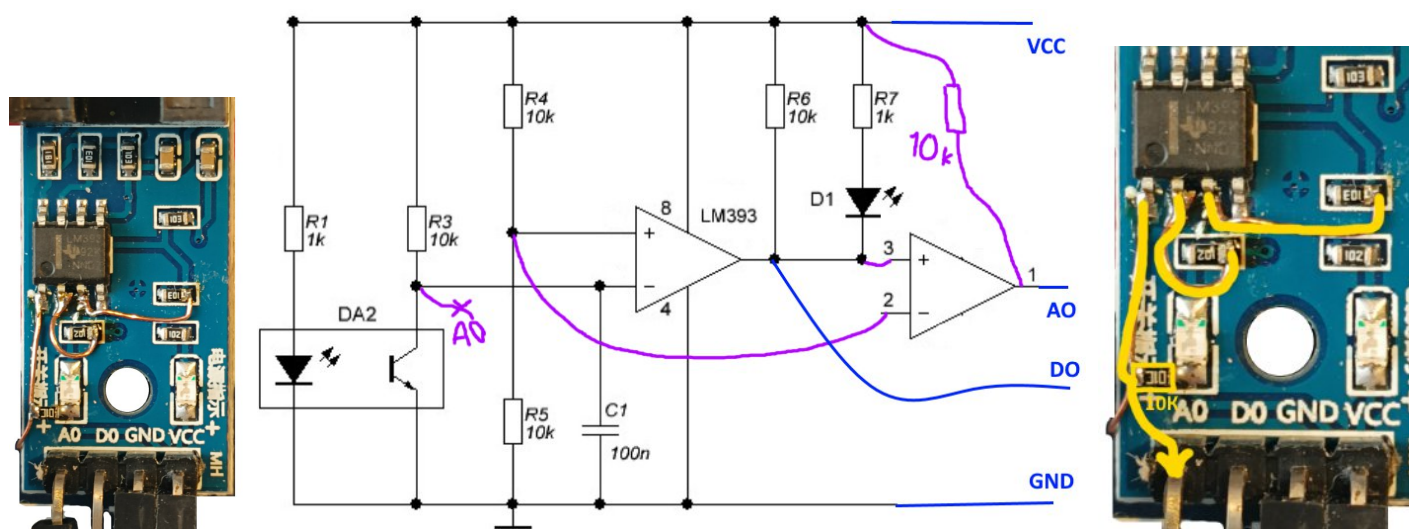
- Temp0 – [V2_PCB_Temp0](#), аналоговый наплатный датчик LM60, низкой точности. В SW вручную откорректирован на -2C.
- TempR1/R2 – [V2_Heat_T1/T2](#). Rsens от графена уже обсуждены выше. Датчики в виде золотой напылённой полосы на графене, подача измерительного напряжения, отдельный буфер и усилитель. (SW – сделать в основном алгоритме поддержку R1/R2 **двух** номиналов сопротивлений для двух каналов, иначе не нагреть ровно. Уточнить параметры полинома для преобразования R в T для золота.)
- Temp1/2 – [V2_1Wire_Sens1/2](#) однопроводные цифровые датчики, два типа: чисто температура DS18B20 и датчик температуры-влажности DHT, автоопределение типа, 0xFFFF если нет датчика.
- Temp3/4 – [V2_1Wire_Sens3/4](#) однопроводные цифровые датчики аналогично предыдущим, отмечены как HMDT1/2 на плате. (Сделать подтяжку линии данных на + через 4.5K, включение питания через PWR_HMDT)
- Temp5 – [V2_1Wire_Sens5](#) доп канал однопроводного датчика для красивого выравнивания регистров modbus на 0430h, впаян отдельно, включен на PC15 (как вариант подклеен на КТП8 на фторопластовый задник графена). Сюда же подключается датчик вращения, который активируется при регистре Rotat>0.
- Temp6/7 – [V2_I2C_Sens1/2](#) датчик BME280 вставлен на место TX0/RX0 в режиме I2C. При этом FU4 перекинут с 5V на 3.3V если будет прямое подключение без LDO на плате BME. На один физический i2c можно адресовать два BME280 перемычкой на нём. В коде сделана поддержка нескольких датчиков с разным i2c адресом. Исходя из сложной логики загрузки калибровки, эти датчики инициализируются при старте платы или выходе из Test.
- Motor – X2, 12V, [V2_Cmd_Motor](#)
- Valve – X5, 24V, [V2_Cmd_Valve](#)

Датчик положения двигателя

Двигатель обычный, не шаговый, оборотов много, положение точно выставить нельзя, поэтому реализован *счётный алгоритм* останова по количеству оборотов.

- По умолчанию значение регистра Rot=0 и доп разъём Temp5 работает в режиме датчика температуры, при этом катушка V2_Cmd_Motor вручную запускает и останавливает двигатель обычным способом на постоянное вращение.
- При установке в регистр Rot значения отличного от нуля, после запуска двигателя отсчитывается установленное значение *срабатывания* датчиков положения и двигатель останавливается сбросом регистра V2_Cmd_Motor. При этом схемотехнически датчики положения можно параллелировать (open drain) и увеличивать точность позиционирования по количеству датчиков на один оборот оси двигателя.

Готовый датчик имеет включённый режим при открытом опто зазоре, при этом выходной транзистор открыт, что не даёт включать параллельно несколько датчиков (но схема подсчёта работоспособна). Для параллельного включения сразу нескольких датчиков надо инвертировать сигнал, при этом плата датчика дорабатывается следующим образом, используя свободный второй компаратор.



2DO – т.к. не было свободных ног MCU, задействована нога PC15 OSC в которой нет ALT режима связи с TIMER и подсчёт идёт вручную опросом с периодом 50ms. Потом надо навесить на нормальную ногу и делать подсчёт аппаратным таймером. И вообще поставить шаговый двигатель.

Консольный клиент

Для отладки обмена с платой написан консольный клиент на питоне. Функции:
















- Получение и парсинг регистров V1, с учётом swar байтов
 - 1-8h номера каналов и состояние термо сенсора
 - 300-301h управление
 - 302-312h установки термостата
 - 30h измеренная матрица
- Управление V2
 - Однобитные Coil, по всем трём функциям-командам modbus 03-05-10
 - 16 битные регистры записи, установка командой 06h значений Ro для двух каналов, V1Ts и Rot
 - 16 битные регистры чтения, регулярное чтение и парсинг регистров сенсоров

```
PS D:\dev\enose> py .\v2enose.py
Ioffe inst. Graphene eNose V2 client, modbus-COM16, z/q-quit, h-help
Ver b'20052024' Ts:66.00 Ro:8.80 Rot:7 CoilsON: h-boost
Help: z,q-exit,
get: p-toggle poll regs (+/- speed), c,a-coils, s-sets, v-V1 regs,
set: 0...9?-coils, w-Ro, t-Ts, r-Rot
0=start, 1=heat, 2=h-boost, 3=h-still, 4=motor, 5=valve, 15=test,
Ts:66.00 Ro:8.80 Rot:7
V1 MeasN.Ch.dT:45.10.940 Tr:21.66/37.08 Rt:9.39/11.03 Vref:3.310
V1 Start=1, Heat=0
V1 Ts:66.00 Rs:10.60 Ro:8.80 A:0.0031 B:0 Kp:0.30 Ki:0.01 M1:0xffff0xffff
0x030(R) resp error
Regs: HeartB:5711 Rt:9.3/11.0 Tr:21.2/37.0 Tb: 27.9
Sens: 1:27.5 2:empty 3:27.6/40.4 4:30.2 5:rot
Sen2: 1:25.34/37.78/1009.76 2:empty
Regs: HeartB:5712 Rt:9.3/11.0 Tr:21.0/36.7 Tb: 27.8
Sens: 1:27.5 2:empty 3:27.6/40.4 4:30.2 5:rot
Sen2: 1:25.36/38.04/1009.76 2:empty
```

Алгоритм

1. Подача питания, инициализация периферии, загрузка параметров Ts Ro A Msk ORv2 из flash
2. Постоянно измерение девяти датчиков температуры, 50ms канал, полный цикл примерно 1s.
3. Постоянный обмен по modbus по прерыванию USART
4. Если Start – то запуск измерения каналов, 100ms канал
5. Если Heat – то включение Heat DC/DC примерно 1s цикл термостат, если Still – постоянно нагрев
6. Если Motor – то подача 12V на X2 "Vent ", если Rot>0 - то счёт оборотов по разъёму Temp5
7. Если Valve – то включение 24V DC/DC и подача напряжения на X5
8. Если Test – то останов измерения каналов, термодатчиков, двигателя и генерация Rand

Индикаторы

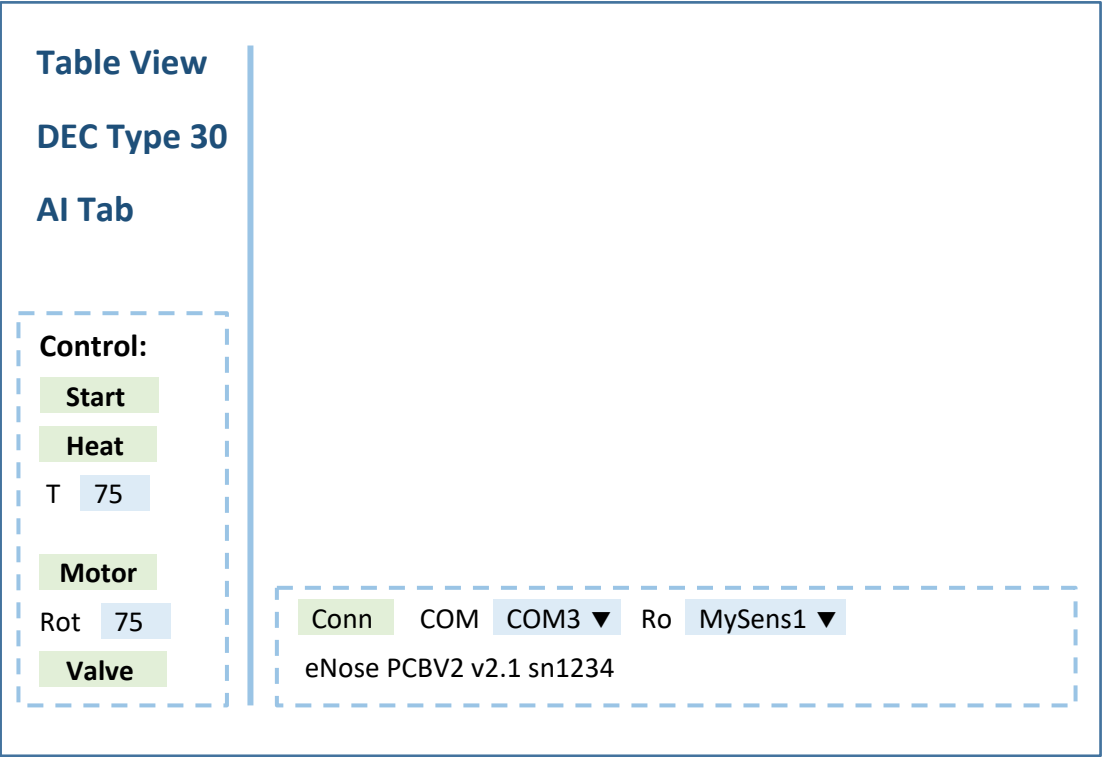
LED12   HMDT ON	Питание 1wire датчиков Temp3/4; включено всегда, кроме Test
 R135  Reserv	Система, мигает раз в 2 секунды и при modbus обмене
R125 LED4   Heater2 ON	1й подогрев графена, решает алгоритм термостата, или всегда вкл если Still
 R124  Heater1 ON	2й подогрев графена, решает алгоритм термостата, или всегда вкл если Still
R119 LED2  Heater PWR	Включение Heat DC/DC, команда V2_Cmd_HEAT
 LED6 R11  Screen PWR	Питание i2c датчиков Temp6/; включено всегда, кроме Test (может по i2c запитаться)
Valve ON  	Напряжение на клапан, вместе с +24V (у мотора индикатора нет!)
+24V  	Включение 24V DC/DC, команда V2_Cmd_Valve

Интерфейсы

Предлагаемый прототип интерфейса

Общая часть

Слева – закладки и управление процессом. Все остальные параметры или прописаны в плату, но доступны по modbus, или вынесены в *конфиг*. Сейчас понятны два интерфейса - 2Д графики с таблицей, и лепестковая диаграмма. Следующие интерфейсы будут добавляться в закладки.



Слева Control: запуск измерения, запуск нагревателя с установкой температуры, запуск мотора с установкой оборотов, включение клапана. В середине снизу Conn: выбор ком порта, выбор Ro датчика *из conf файла*, вывод релиза платы.

- Алгоритм термостата реализован в плате, достаточно установленной температуры.
- Сопротивления термосенсоров чипа вынести в конфиг файл, тут выбирать *чип* из списка по названию.
- Всё остальное что есть конфигурационного в плате в этом интерфейсе не реализовывать

Конфигурация платы и текущее значение сенсоров
(не делать, приведено для справки)

Подумать где разместить Connect и BoardName, отдельной вкладки недостойно. Остальные параметры скрыты т.к. известные константы, доступны через modbus редактор или скрипт. Точнее:

- RL RR выбирается по названию чипа из вкладки Control и хранится в конфиге под каждый чип,
- параметр A для золота фиксирован и прописан в плату, параметр B не нужен в диапазоне до 100 градусов,
- маску ставят через лепестковую диаграмму
- список и значения датчиков температуры есть на вкладке с таблицей и 2Д графике.

Остаётся: выбор порта, (опционально - слейв адрес,) кнопка "connect", выбор Rol Ror датчика по имени, и вывод версии платы (опционально), это можно в Control вкладку на основной экран.

4.5

— поле ввода

Set

— кнопка

[410h] RoL

4.5

Ω

<1.0..25.0>

Регистр modbus, название регистра, контрол ввода значения, единица измерения, валидация, тултип-подсказка.

Left R sens at 0 deg °C

BOARD

Connect:

Serial

COM2

Slave

1

Conn

Board name:

[100h/8] Name

eNose PCB V2

[200h/4] Ver

v2.1

[204h/4] Serial

2002

Board config:

[410h] RoL

4.5

Ω<1.0..25.0>

Left R sens at 0 deg °C

[411h] RoR

5.1

Ω<1.0..25.0>

Right R sens at 0 deg °C

[308h] A

3.9083

10⁻³ °C⁻¹

<1.0000..9.0000>

A-polynomial Rt=Ro(1+A*t+B*t^2)

[308h] B

-5.775

10⁻⁷ °C⁻²

<1.0000..9.0000>

B-polynomial Rt=Ro(1+A*t+B*t^2)

Set

[310h] Mask01-16

1111 1111 0011 1100

<16>

[311h] Mask17-32

0001 1111 0011 1111

<16>

[312h] Mask33-38

1111 11

<6>

38 channels measure mask

Set

Sensors:

[423h] T_{RL}

85.1

°C

[424h] T_{RR}

80.8

°C

Sens L/R

[425h] T_{BOARD}

24.3

°C

Board sens

[426h] Sens1

25.6

°C %

[428h] Sens2

23.1/487

°C %

[42Ah] Sens3

empty

°C %

[42Ch] Sens4

25.1

°C %

[42Eh] Sens5

rotate

°C %

Ext 1wire sens T H

[430h] Sens6

23.1/449/1023

°C % kPa

[433h] Sens7

22.8/390/1118

°C % kPa

Ext i2c sens T H P

Измерение в виде плоских графиков и таблицы.

Термодатчики опрашиваются софтом перед стартом, на всё измерение фиксируется их количество и измеряемые ими величины (T, TH, THP) . Привязку названий датчиков к номерам прописать в конфиг. Датчики *появляются* в графике и таблице под своими названиями.

Новая точка в графике и строка в таблице появляются каждое полное измерение примерно 5-20 секунд. В каждой строке таблицы проставляются: время с начала измерения hh.mm.ss (DateTime убрать), текстовая метка (плохое решение, убрать?), показания термодатчиков, значения сопротивлений активных каналов датчика. Кнопка Export выкидывает таблицу в CSV. Для графиков R и лепестковой решить задачу отображаемого диапазона.

Table Chart

Control:

Start [400h]
Start measurement

Motor [404h]
Start motor

Test [40Fh]
Simulating mode

Thermostat:

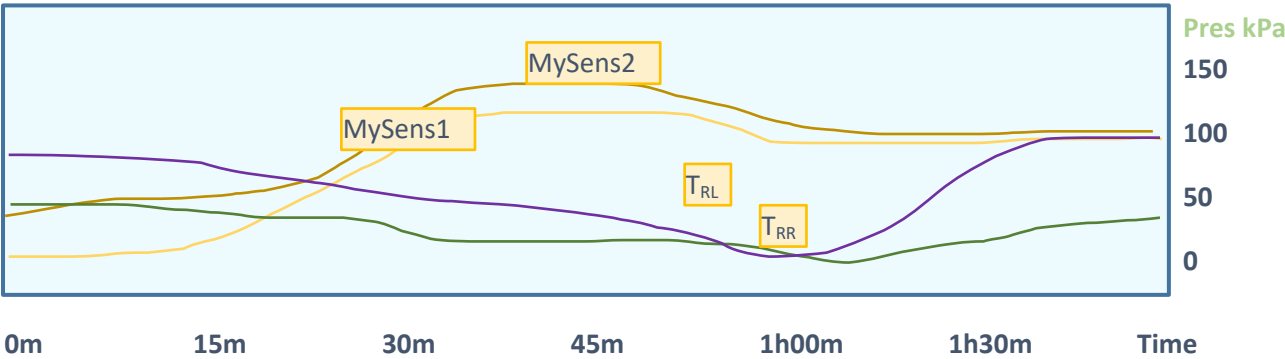
T 45 deg °C <20..100>
Sensor temp deg °C [302h]

Heat [401h]
Start heating

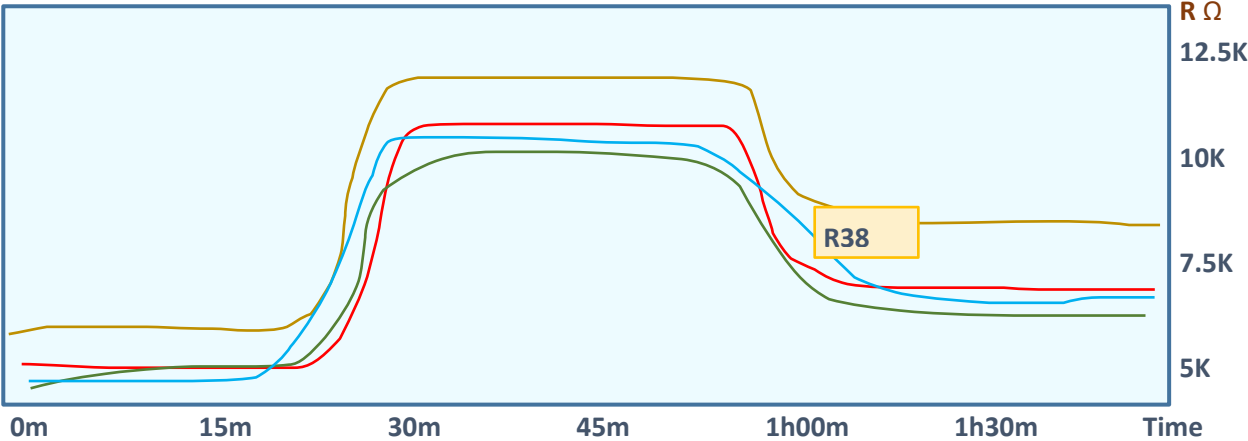
Boost [402h]
Boost heating

Still [403h]
Fix heating

THP sensors



R sensors



Tab

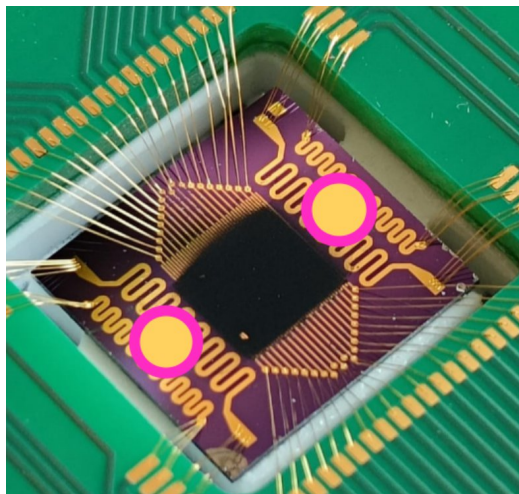
Time	Label	S1T	S2T	S3T	S3H	S7P ...	S8T	R1	R2	...	R37	R38
08:12:35	MyLab	84.1	86.3	24.7	354	1006 ..	27.4	1.23K	2.12K	...	0.92G	1.01G

Модель температуры от сопротивления

Задача коэффициентов полинома термосопротивления на чипе графена. Чтобы понимать, что измерение температуры графен чипа примерно верное, надо иметь модель зависимости сопротивление золотых напылённых дорожек от температуры.

Для этого необходимо составить таблицу и график сопротивлений левый/правый для температур: 0(лёд), 22(комната), 50(термофен, режим измерения), 100(термофен, очистка чипа) градусов,

- взять все чипы что есть, т.к. большой разброс номиналов,
- накали подавать через два лабораторных БП одновременно на обе полосы, пропорционально их сопротивлениям, выравнивая температуру на каналах накала так, что бы в среднем по подложке вышел градиент не видимый тестером, это $\pm 0.5^{\circ}\text{C}$
- в табличку занести оба напряжения накала, так удобнее будет для платы
- как термопоинт (точка измерения температуры) назначить точку в середине каждой накальной полосы (см картинку); если не жалко - ткнуть в центр графена, понять градиент по подложке
- точность измерения и установки температуры принять например 5°C ($\pm 2.5^{\circ}\text{C}$), на шкале 20-100 $^{\circ}\text{C}$
- измерять термопарой тестера, например через каплю термопасты КТП8
- далее по табличке составить график температура от сопротивления, парами каналов для каждого чипа
- попробовать привязать к ГОСТ 6651, к полиному для платины и никеля
- вычислить и усреднить коэффициенты А и В из полинома термопар металлов платина-никель (R_0 - сопротивление на нуле градусов мы измеряли в начале), возможно коэффициент В будет лишний в нашем диапазоне измерений
- записать их в регистры платы и запустить старый алгоритм поддержания температуры с контролем термопарой



Сделать выводы:

- если мы убираемся в 5% точности *измерения* температуры по сопротивлению, то принять что мы победили, можно временно оставить алгоритм регулирования термостата на плате V2, но затем на V3 DC/DC переделать на статику (см ниже)
- если очень большой технологический разброс, низкая точность, то признать что измеряем что-то при непонятной температуре, предусмотреть на плате чипа термодатчик и снимать примерную температуру с него, можно реализовать жестко закодированную таблицу температур под каждый датчик

Выводы: для термосенсора графен чипов актуальна формула $R_t = R_0 \cdot (1 + A \cdot t)$, где **A = 0,0031**

Статический накал V3

Убрать регулировку термостата через алгоритм ПО и оставить одну регулировку через DC/DC: он точнее, быстрее, плавнее и без программного кода

- учесть большой диапазон разброса сопротивлений накала 5-20 Ω
- предусмотреть контроль обрыва и КЗ накала
- сделать *два* DC/DC, по одному на каждый накал накала
- выбрать DC/DC у которого нормальный режим вход 12V, а на выход $\sim 1.5V$ 0.5A, т.е. впрыток под его Vref;
- прикинуть теплоотвод решения, возможно будет греться на таких Vin/Vout
- для установки напряжения в цепь ОС DC/DC поставить DAC со стороны MCU, 8-12 bit хватит
- без работы DAC и кода MCU должно быть приемлемое напряжение накала
- завести *измерение напряжения* накала на два ADC MCU
- схему с ОУ измерения сопротивления накала оставить
- подумать над схемой измерения накала, сейчас шумит, и кажется не очень грамотная схема, двуполярная, сначала буфер, непонятная, потом K_u усилитель.
- поднять ток сенсора температуры до 2`500mV сигнала в полном размахе для полного нагрева для самого высокоомного датчика
- я бы поставил на плату чипа любой полупроводниковый термосенсор, цифровой однопроводной или аналоговый - без разницы, или сразу погодную станцию BME280 впаять
- как супер-вариант, посмотреть как бы на подложку с каждого бока разварить и опустить на подложку каноническую термопару
- на плате чипа можно оставить место на два светодиода, твердотельщики любят облучать всё

Сделано на V2

всей командой инженеров

HW

- изучение темы, анализ старой схемы, платы, софта контроллера и верхнего софта
- выработка совместного решения по доработке железа
- разработка механической части пробоотборника
- разработка V2 платы полностью *совместимой* с ПО V1
- добавление датчиков температуры/влажности/давления
- переделка схемы для линейного нагрева на два порога для 40 и 100

FW

- глубокий рефакторинг кода с полной совместимостью с V1
- переделка и доработка Modbus протокола
 - согласование нового поля регистров 0x400 для новой верхней софты
 - проработка маппинга старых регистров на новые
 - добавление команд для новых регистров
 - добавление поддержки дискретных modbus команд 05h 06h
 - проблема переворота сетевой последовательности битов Msb-Lsb
- подключение датчиков аппаратно и написание драйвера
 - термо LM60C
 - термо DS18B20
 - термо/влажность DHT22-AM2302
 - термо/влажность/давление BME280
 - hot-swap для однопроводных датчиков с автоопределением типа на лету
 - датчик положения вала двигателя, счётный режим
- тест режим, для удобства программиста верхнего софта, генерация тестовых данных

SW

- полностью новый верхний софт на qt
- питон скрипт теста modbus V2

Вопросы к V1 и V2

- Точность измерения сопротивления каналов сенсора,
- Влияние большого количества ключей на входе на измерения в статике и динамике
- Необходимость параллельного измерения каналов
- Накал, точность измерения R и через него T на графене
- Точность и линейность поддержания T графена с алгоритмом термостата
- Вход для датчика положения оси и его алгоритм

Планы на V3

Концепция – понять что делаем

а – исследовательское устройство в рамках науки и для отчётов, *это версия что сейчас*

б – штучный лабораторный дорогой прибор, надо понять кому и что надо

в – серийный дешёвый датчик газа, это к графенщикам сначала, пусть дадут дешёвый простой датчик

Мажорные вопросы

- У нас уже вышел графен с нужными характеристиками?
- Сколько надо каналов на датчике (38 нормально для исследований)?
- Нужно ли их в параллель цифровать и зачем, для чего скорость?
- Измерение ёмкости, временные характеристики графена
- Освещение датчика

Технические миноры

- Оставить блочную компоновку Analogue-MCU-PC
- Аналоговую часть переработать
 - Аккуратное усиление сигнала с каналов датчика в широком диапазоне токов сопротивления
 - Форм фактор датчика, поставить усилитель, коммутацию и ADC на плату датчика
 - Режим измерения ёмкости
 - Линейный накал с управлением через DAC
 - Свободные дискретные входы-выходы для внешней периферии
- MCU поставить в виде готовой платы [STM Nucleo](#)
 - Соединение с хостом можно USB можно Eth
 - Протокол можно modbus можно json подобный
- Верхний софт можно сделать консольным на питоне для удобства работы экспериментатора
- Красивый GUI софт для инвесторов

El Noze - Modbus –V2

20-May-24

Addr	FuncCmd	Dev Cmd	Desc	Remark
Write coil 400h Func 05h – write single individual coil, addr 400-405h ... 40Fh Func 10h – also write as <i>multiply</i> registers start from 400h Func 03h – read multi start from 400h				
400	03h 05h 10h	V2_Cmd_START	Start measurements	Sets 300h
401		V2_Cmd_HEAT	Start heating	Sets 301h
402		V2_Cmd_Heat_Boost	Boost heater voltage	
403		V2_Cmd_Heat_Still	Linear heater	
404		V2_Cmd_Motor	Start motor rotation	
405		V2_Cmd_Valve	Turn valve on (?)	
406-40E		Reserved		
40F		V2_Cmd_Test	Test data generation	
Write registers 410h Func 06h – write single reg, addr 410h, 411h, 412h Func 10h – write as <i>multiply</i> registers start from 410h Func 03h - read multi reg start from 410h				
410	03h 06h 10h	V2_Heat_OR0_OHM	Heater sens R0 and R1 values in Ω at 0 deg. °C	Set value = (uint16_t) 67 as 6.7 Ω Sets 306h
411		V2_Heat_OR1_OHM		
412		V2_Rotat	Set rotation count	If 0, then TEMP5 is T sens
413-41F		Reserved		
Read only registers 420h Func 03h – read multi start from 420h				
420	03h	V2_HeartBeat	1 sec continuously incrementing watchdog timer	To check if device alive, and uptime in sec (uint16)
421		V2_Heat_R1	Heater sens R value, Smeasured in Ohms	Resistance 123 = 12.3 Ω Same as 03-04h
422		V2_Heat_R2		
423	TEMP1	V2_Heat_Temp1	Heater T value in deg. °C calculated from Ohms value	Temp 567 = 56.7 °C Same as 05-06h
424	TEMP2	V2_Heat_Temp2		
425	TEMP0	V2_PCB_Temp0	PCB semi analogue T sensor (LM60, accuracy ±2.0°C)	Temp 234 = 23.4 °C
426-427	TEMP3	V2_1Wire_Sens1	Single wire temperature and humidity sensors (AM2303), or only temperature (DS1820), type autodetect	Temp 234 = 23.4 °C 0xFFFF – empty sens 0xFFFFE – rotat sens Humidity 789 = 78.9% 0xFFFF – sens w/o humi
428-429	TEMP4	V2_1Wire_Sens2		
42A-42B	HMDT1	V2_1Wire_Sens3		
42C-42D	HMDT2	V2_1Wire_Sens4		
42E-42F	TEMP5	V2_1Wire_Sens5		
430-432	RX/TX0	V2_I2C_Sens1	I2C temperature, humidity and pressure sensor (BME280)	Temp 2345 = 23.45 °C 0xFFFF – empty sens Humidity 4567= 45.67% Press. 56789+50000 = 1067.89 hPa
433-435	RX/TX0	V2_I2C_Sens2		

Modbus function codes

Code	Hex	DataType	In/Out	Semantic	Modicon name	Payload
01	0x01	1byte *X	DO	Read discrete output	Read Coil Status	> CoilsQuantity[16] < CoilsData[8*x%q]
02	0x02	1byte *X	DI	Read discrete input	Read Input Status	> CoilsQuantity[16] < CoilsData[8*x%q]
03	0x03	16bytes *X	AO	Read analogue output	Read Holding Registers	> RegsQuantity[16] < RegsData[16*x]
04	0x04	16bytes	AI	Read single analogue input	Read Input Register	> RegQuantity[16] < RegData[16]
05	0x05	1byte	DO	Write single discrete output	Force Single Coil	> CoilData[16, 0 FF] < CoilData[16]
06	0x06	16bytes	AO	Write single analogue output	Preset Single Register	> RegData[16] < RegData[16]
15	0x0F	1byte *X	DOx	Write multiply discrete output	Force Multiple Coils	> CoilsQuantity[16] BytesCount[8] CoilsData[16*x%q] < CoilsQuantity[16]
16	0x10	16bytes *X	AOx	Write multiply analogue output	Preset Multiple Registers	> RegsQuantity[16] BytesCount[8] RegsData[16*x] < RegsQuantity[16]

Request: SlaveAddr[8b] – FuncCmd[8b] - Coil/Reg Addr[16b] – {payload} – CRC16[16b]

Response: SlaveAddr[8b] – FuncCmd[8b] - BytesCount[8b] – {payload} – CRC16[16b]

coil payload: Rd: 01010101 110000 ... Wr: 00FF=setOn

reg payload: Rd: Hi-Lo Hi-Lo Hi-Lo ... Wr: Hi-Lo Hi-Lo Hi-Lo ...

Example Func03

(Hex)	Field name	(Hex)	Field name
11	Device address	11	Device address
03	Functional code	03	Functional code
00	Address of the first register Hi bytes	06	Number of bytes more
6B	Address of the first register Lo bytes	AE	Register value Hi #40108
00	Number of registers Hi bytes	41	Register value Lo #40108
03	Number of registers Lo bytes	56	Register value Hi #40109
76	Checksum CRC	52	Register value Lo #40109
87	Checksum CRC	43	Register value Hi #40110
		40	Register value Lo #40110
		49	Checksum CRC
		AD	Checksum CRC

El Noze - Modbus – V1

Reg Dec	Reg Hex	Desc	Read	Write	Value range	Rem.
0001	00h 01	Номер измерения	03h	-	0000h...FFFFh	При каждом измерении всех датчиков - инкремент See V2 5Dh
0002	00h 02	Время измерения	03h	-	0000h...FFFFh	Время затраченное на измерение всех датчиков See V2 5Fh
0003	00h 03	Температура датчика T1	03h	-	0000h...FFFFh	коэф. x100, °C V2 421h...
0004	00h 04	Температура датчика T2	03h	-	0000h...FFFFh	коэф. x100, °C
0005	00h 05	Сопротивление датчика T1	03h	-	0000h...FFFFh	коэф. x100, Ом V2 423h...
0006	00h 06	Сопротивление датчика T2	03h	-	0000h...FFFFh	коэф. x100, Ом
0007	00h 07	Опорное напряжение	03h	-	0000h...FFFFh	коэф. x1000, мВ
0008	00h 08	Номер текущего канала	03h	-	0000h...FFFFh	See V2 5Eh
00768	03h 00	Команды управления – старт/стоп измерения	03h	10h	0000h...0001h	0 – стоп V2 400h 1 – старт
00769	03h 01	Команды управления – старт/стоп подготовка	03h	10h	0000h...0001h	0 – стоп V2 401h... 1 – старт 0100h=256
00770	03h 02	Задание для нагрева нагревателей, t	03h	10h	0000h...FFFFh	float, °C
00771	03h 03				0000h...FFFFh	
00772	03h 04	Задание для нагрева нагревателей, Rt	03h	10h	0000h...FFFFh	float, Ом
00773	03h 05				0000h...FFFFh	
00774	03h 06	Сопротивление датчика при нулевой темп-ре, R0	03h	10h	0000h...FFFFh	float, Ом. 6.66 def
00775	03h 07				0000h...FFFFh	V2 421h
00776	03h 08	Кэффициент A	03h	10h	0000h...FFFFh	float
00777	03h 09				0000h...FFFFh	
00778	03h 0A	Кэффициент B	03h	10h	0000h...FFFFh	float
00779	03h 0B				0000h...FFFFh	
00780	03h 0C	Кэффициент Kp	03h	10h	0000h...FFFFh	float
00781	03h 0D				0000h...FFFFh	
00782	03h 0E	Кэффициент Ki	03h	10h	0000h...FFFFh	float
00783	03h 0F				0000h...FFFFh	
00784	03h 10	Маска измеряемых каналов 1...16	03h	10h	0000h...FFFFh	измерение 1 – производится, 0 – не производится
00785	03h 11	Маска измеряемых каналов 17...32	03h	10h	0000h...FFFFh	
00786	03h 12	Маска измеряемых каналов 33...38	03h	10h	0000h...FFFFh	

Все данные имеют перепутанные (swap) октеты в ushort слове modbus. При этом сетевая последовательность полуслов верная, lsb. Это означает что:

IEEE754-float 6.66= 0x msb(40D5) lsb(1EB8) -> saved as D540 B81E

uint16_t SetCoil_ON = msb(00) lsb(FF) -> saved as FF 00 (65 280)

00256	01h 00h	Имя модуля	03h	-	0000h...FFFFh	Чтение имени модуля осуществлять начиная с регистра 01h 00h. Значение закодировано согласно ASCII (см. п.5).
00257	01h 01h		03h	-	0000h...FFFFh	
00258	01h 02h		03h	-	0000h...FFFFh	
00259	01h 03h		03h	-	0000h...FFFFh	
00260	01h 04h		03h	-	0000h...FFFFh	
00261	01h 05h		03h	-	0000h...FFFFh	
00262	01h 06h		03h	-	0000h...FFFFh	
00263	01h 07h		03h	-	0000h...FFFFh	
00512	02h 00	Версия ПО	03h	-	0000h...FFFFh	Чтение версии осуществлять начиная с регистра 02h 00h. Значение закодировано согласно ASCII (см. п.5).
00513	02h 01		03h	-	0000h...FFFFh	
00514	02h 02		03h	-	0000h...FFFFh	
00515	02h 03		03h	-	0000h...FFFFh	
00516	02h 04	Серийный номер	03h	-	0000h...FFFFh	Чтение версии осуществлять начиная с регистра 02h 05h. Значение закодировано согласно ASCII (см. п.5).
00517	02h 05		03h	-	0000h...FFFFh	
00518	02h 06		03h	-	0000h...FFFFh	
00519	02h 07		03h	-	0000h...FFFFh	

1E	00h 30h	Сопротивление R1	38 пар регистров для 38 каналов сопротивлений Начало с 30h и 56h (R20) по 0x30(48) регистров (96 байт) Значение в swap float, Омы			
1F	00h 31h					
91	00h 5Bh	Сопротивление R38	Map to 0001h, Номер измерения			
92	00h 5Ch					
93	00 5Dh	V2_REG_MEAS_NUM_COPY	Map to 0008h, Номер тек. канала			
94	00 5Eh	V2_REG_MEAS_CH_COPY	Map to 0002h, Время измерения			
95	00 6Fh	V2_REG_MEAS_TIME_COPY				

Err codes	Desc.
0x01	Устройство не поддерживает запрашиваемую функцию
0x02	Ошибка запроса регистра(ов)
0x03	Записываемые данные находятся вне диапазона допустимых значений для данного регистра
0x07	Попытка записи нового адреса или скорости при отсутствии перемычки между клеммами «INIT» и «GND»
0x15	Ошибка проверочного кода CRC-16

Режимы считывания и записи

- 1.Регистры 0x0001..0x0008 считывать одним запросом начиная с 0x0001
- 2.Регистры 0x0030..0x004B считывать двумя запросами начиная с 0x0030h и 0x0056h по 0x30(48) регистров
- 3.Регистры 0x0100..0x0107 считывать одним запросом начиная с 0x0100
- 4.Регистры 0x0200..0x0207 считывать одним запросом начиная с 0x0200
- 5.Регистры 0x0300..0x0301 считывать одним запросом начиная с 0x0300
- 6.Регистры 0x0302..0x0312 считывать одним запросом начиная с 0x0302
7. Регистры 0x0300..0x0301 записывать одним запросом начиная с 0x0300
8. Регистры 0x0302..0x030B записывать одним запросом начиная с 0x0302
9. Регистры 0x030C..0x030F записывать одним запросом начиная с 0x030C
8. Регистры 0x0310..0x0312 записывать одним запросом начиная с 0x0310

Сборка MCU проекта

Плата версии EINos_rev2_Scheme-23Feb24

- Процессор GD32F103VBT корпус LQFP100
- Среда Keil5 uVision v5.38.0.0 2022, проект в папке KEIL5_PRJ
- Компилятор ARMCC v5.06 upd 7 build 960
- GD32 SPL V2.2.0 2020-09-30, каталог GD32F10x_SPL включая it и libopt файлы
- Отключить ссылки на встроенные пакеты Target options: C/C++ -> No auto includes
- CMSIS Cortex-M3 Core V3.30 17.Feb.2014, каталог CMSIS
- Debug = ST-LinkV2
- F7-compile, F8-upload

Проект собирается без бубна, и даже без warning

Project: eNoze

GD32F10X_CL

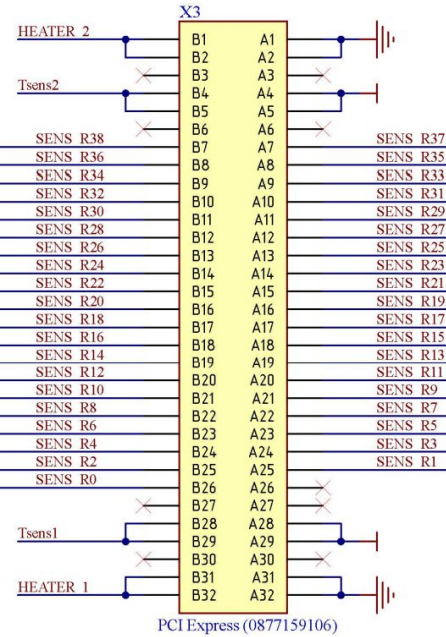
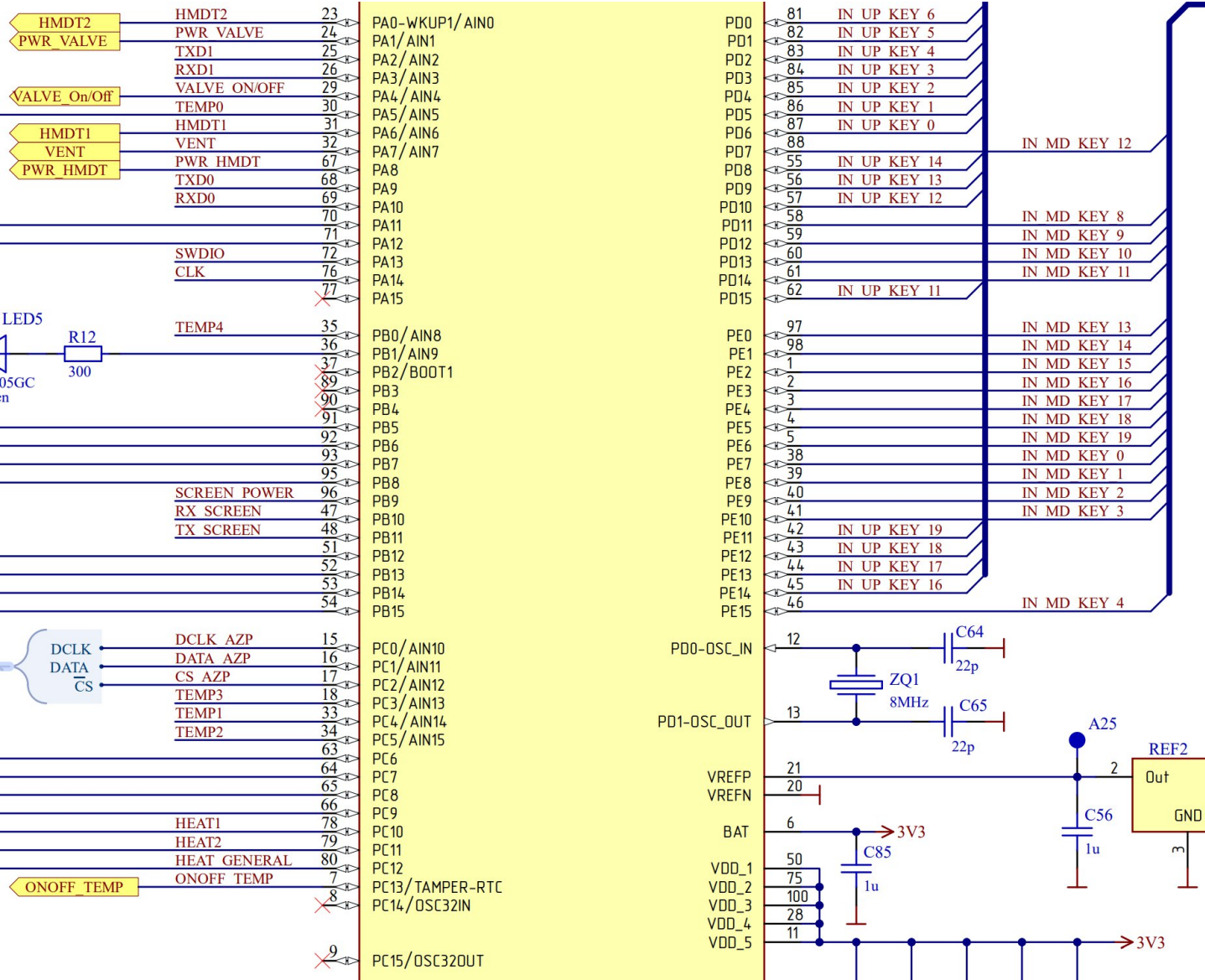
- app
 - main.c
 - rtos_v1.c
 - timer.c
 - crc.c
 - data.c
 - meas.c
 - flash.c
 - modbus.c
 - usart.c
 - temper.c
 - ds18b20.c
 - dht22.c
 - bme280.c
 - config.c
- CMSIS
 - startup_gd32f10x_md.s
 - system_gd32f10x.c
- GD32F10x_SPL
 - gd32f10x_adc.c
 - gd32f10x_exti.c
 - gd32f10x_gpio.c
 - gd32f10x_rcu.c
 - gd32f10x_timer.c
 - gd32f10x_usart.c
 - gd32f10x_misc.c
 - gd32f10x_fmcc.c
 - gd32f10x_i2c.c

Build Output

```
Rebuild started: Project: eNoze
*** Using Compiler 'V5.06 update 7 (build 960)', folder: 'D:\dev
Rebuild target 'GD32F10X_CL'
assembling startup_gd32f10x_md.s...
compiling main.c...
compiling crc.c...
compiling rtos_v1.c...
compiling timer.c...
compiling dht22.c...
compiling modbus.c...
compiling config.c...
compiling gd32f10x_i2c.c...
compiling data.c...
compiling ds18b20.c...
compiling gd32f10x_misc.c...
compiling gd32f10x_exti.c...
compiling system_gd32f10x.c...
compiling gd32f10x_rcu.c...
compiling gd32f10x_gpio.c...
compiling flash.c...
compiling usart.c...
compiling gd32f10x_usart.c...
compiling gd32f10x_adc.c...
compiling gd32f10x_fmcc.c...
compiling gd32f10x_timer.c...
compiling temper.c...
compiling bme280.c...
compiling meas.c...
linking...
Program Size: Code=33420 RO-data=1612 RW-data=160 ZI-data=3928
FromELF: creating hex file...
".\output\Project.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed: 00:00:02
```

- CMSIS
- GD32F10x_SPL
- KEIL5_PRJ
- bme280.c
- config.c
- config.h
- crc.c
- data.c
- data.h
- dht22.c
- ds18b20.c
- flash.c
- flash.h
- main.c
- meas.c
- meas.h
- modbus.c
- modbus.h
- rtos_v1.c
- rtos_v1.h
- temper.c
- temper.h
- timer.c
- timer.h
- usart.c
- usart.h

Выборочно схема платы V2



PCB V2 с доработками

