Модернизированный барометр-гигрометр-термометр с батарейным питанием на базе микроконтроллера EFM8SB20F16 и E-ink дисплея

Алексей Кузьминов (г. Москва)

В статье приведены принципиальные схемы, программные средства, разводка плат, конструкция и результаты работы барометра-термометра-гигрометра на базе МЭМС-датчика BME280, микроконтроллера (МК) EFM8SB20F16 и E-ink дисплея WFT0000CZ04 (1,54″ epaper-B) с разрешением 200×200 пикселов с питанием от литиевой батарейки ER14335, позволяющей обеспечивать непрерывную работу прибора в течение как минимум 10 лет при обновлении показаний давления, температуры и влажности раз в 10 минут.

Введение

В последнее время в широкой продаже появились новые дисплеи, которые стали называть электронной бумагой (E-paper) или электронными чернилами (E-ink). Эти дисплеи обладают интересным свойством: они сохраняют информацию на экране как в так называемом sleep-режиме (или режиме сна), так и вообще при отключении питания. Потребление тока (около нескольких мА) у этих дисплеев происходит только во время обновления информации на экране (несколько секунд), а в sleep-режиме они практически ничего не потребляют (ток не более нескольких десятых мкА). Эти дисплеи имеют разрешение от полутора до нескольких сотен пикселов по вертикали и горизонтали, в связи с чем, например, при отображении текстовых символов (цифр и букв) достигается практически типографское качество изображения. Использование таких дисплеев в различных устройствах до последнего времени сдерживалось их относительно высокой ценой (от 1000 руб. и выше) и, кроме того, достаточно узким температурным диапазоном работы (0...+30°С). Однако в последние несколько лет цена их упала почти в 3 раза, а температурный диапазон стал заметно шире (–20...+50°С). Автор задался вопросом: нельзя ли в барометре-термометре-гигрометре [1] заменить ЖКИ на E-ink дисплей? Высокое разрешение E-ink дисплея, конечно, требует существенно бóльшую программную память, чем имеется в МК EFM8SB10F8, применённом в [1], поскольку обмен информацией МК с ЖКИ примитивен, и поэтому для него много памяти не нужно. В связи с этим автор применил более новый МК EFM8SB20F16 с удвоенной программной памятью (16 кБ) и существенно бóльшим объёмом оперативной памяти (4 кБ против 0,5 кБ у EFM8SB10F8, хотя это не потребовалось). Помимо этого, в МК EFM8SB20F16 имеется несколько новых опций, среди которых, например, два интерфейса SPI (SPI0 и SPI1), двукратное снижение потребляемого тока в sleep-режиме (0,3 мкА против 0,6 мкА у EFM8SB10F8) и др. Как оказалось впоследствии, программной памяти МК EFM8SB20F16 более чем достаточно даже при использовании дисплея с довольно приличным разрешением 200×200 пикселов, а использование двух независимых SPI существенно упростило разводку платы МК. Помимо дисплеев, в последнее время снизилась и цена BME280 – датчика давления, температуры и влажности (модуль с BME280 можно приобрести на Aliexpress по стоимости чуть менее 240 руб.). Причём, что интересно, модуль с BME280 стоит почти в 3 раза дешевле, чем сама микросхема BME280 в корпусе LGA-8.

Дальнейшее изложение построено следующим образом. Вначале приведены принципиальные схемы устройства и кратко описаны его программные средства (в основном касающиеся вывода информации из МК в дисплей, поскольку о связи МК с BME280 подробно написано в [1]), далее показана разводка и общий вид плат прибора. Затем рассмотрены его конструкция и результаты работы.

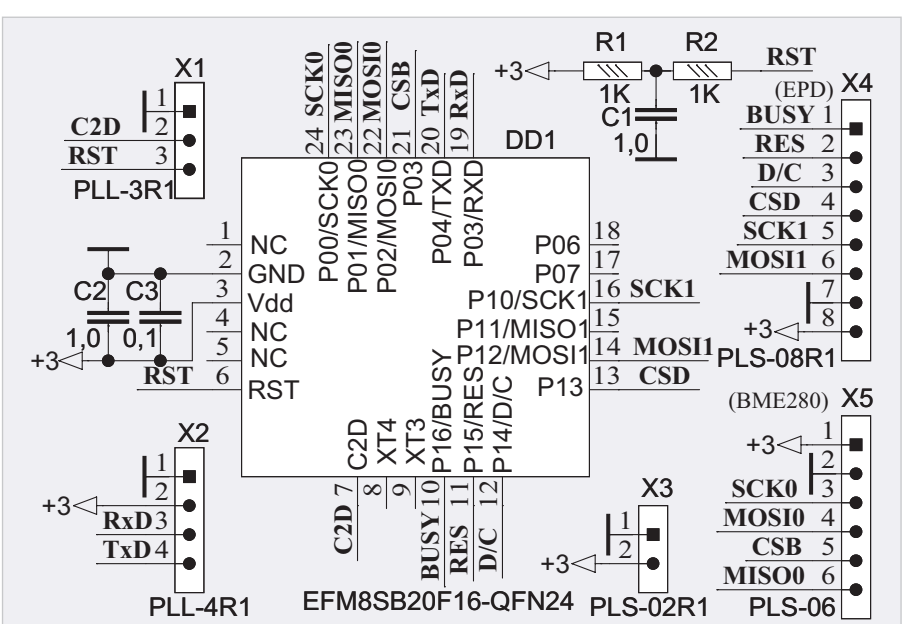


Рис. 1. Плата МК

Принципиальные схемы

В приборе используются три платы: плата МК (рис. 1), плата дисплея (рис. 2), которая непосредственно подключается к плате МК, и плата стабилизатора (рис. 3), которая также подключается к плате МК двухпроводным кабелем.

Как видно из рис. 1, плата МК очень проста. Основой платы служит МК

EFM8SB20F16-QFN24 (DD1) в корпусе QFN-24 размером 4×4 мм. Для программирования МК на плате предусмотрены две возможности. Первая – с помощью USB DEBUG адаптера по двухпроводному интерфейсу C2, вторая – по интерфейсу RS-232 через COM-порт компьютера.

Для программирования по интерфейсу C2 используется разъём X1, к которому подключается кабель от USB DEBUG адаптера с сигналами этого интерфейса (C2D и RST) и «земля», а сам адаптер подключается к USB-порту компьютера (подробности см. в [2]). Для программирования по интерфейсу RS-232 используется разъём X2, к которому подключается кабель с сигналами этого интерфейса TxD, RxD, «земля» и питание +3 В (подробности см. в [3]).

RC-цепочка R1R2C1 предназначена как для штатной работы МК при включении питания (она затягивает низкое состояние сигнала RST, требующееся по штату работы, на время заряда конденсатора C1), так и в режиме программирования по интерфейсу C2 с помощью сигналов RST и C2D (резистор R2 даёт возможность легко управлять сигналом RST от этого интерфейса).

Интерфейс сопряжения МК с BME280 и дисплеем один и тот же – SPI. Для сопряжения с BME280 используется штыревой разъём X5, к которому ответным гнездом подключается модуль с BME280. Для сопряжения с дисплеем используется угловой штыревой разъём X4, к которому ответным гнездом (X1, рис. 2) подключается плата дисплея (рис. 2).

Как было упомянуто ранее, МК EFM8SB20F16 оборудован двумя интерфейсами SPI: SPI0 и SPI1. Для сопряжения с BME280 используется интерфейс SPI0, а для сопряжения с дисплеем – SPI1. Если для сопряжения с BME280 используется полный (двунаправленный) интерфейс SPI (сигналы MOSI0, MISO0 и SCK0), т.е. в/из BME280 передаётся/принимается информация, то для сопряжения с дисплеем (точнее, с его платой – см. далее) используется однонаправленный интерфейс SPI (сигналы MOSI1 и SCK1), или, другими словами, в дисплей только передаётся информация. Скорость обмена по обоим интерфейсам SPI – 5 Мбод. Сигналы выбора кристалла CS для BME280 и дисплея разные: CSB и SCD соответственно. Обмен по SPI идёт с тем устройством, у которого состояние CS низкое (лог. 0). Помимо сигналов интерфейса SPI1

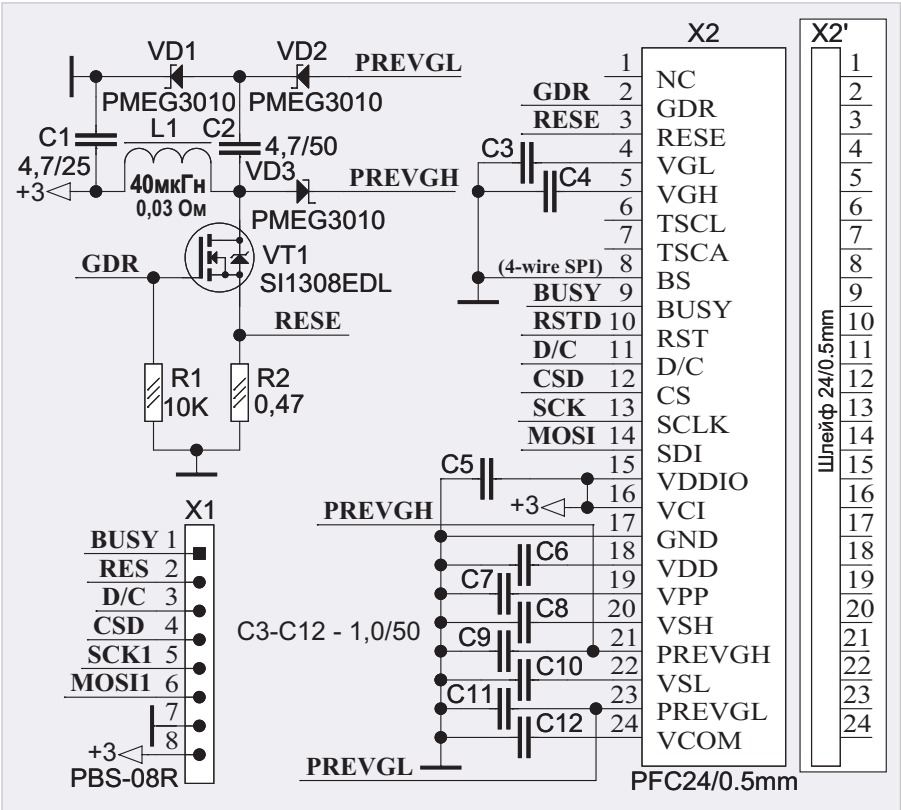


Рис. 2. Плата дисплея

(и CSD) в плату дисплея передаются сигналы D/C (Data/Command – данные/команда) и RES (сброс) и принимается сигнал BUSY (занято).

Питание на плату МК (+3 В и «земля») подаётся на угловой штыревой разъём X3; к нему ответным гнездом подключается кабель, вторая сторона которого подключается к стабилизатору напряжения +3 В, а он, в свою очередь, подключается к батарейке.

Плата дисплея (рис. 2) также не отличается особой сложностью. В ней используется повышающий DC-DC-преобразователь, построенный на транзисторе SI1308EDL (VT1), диодах Шоттки PMEG3010 (VD1–VD3) и конденсаторе C2. Назначение преобразователя – формирование двух относительно высоких напряжений: +20 В (PREVGH) и –20 В (PREVGL). Для работы преобразователя на затвор VT1 из контроллера дисплея подаётся меандр частотой около 1,8 МГц (сигнал GDR). VT1 усиливает по току этот меандр, диоды выпрямляют его, а конденсаторы C9 и C11 сглаживают пульсации двух выпрямленных напряжений. Резистор R2 ограничивает ток стока VT1, а резистор R1 выключает транзистор при отсутствии меандра (когда сигнал GDR находится в высокоимпедансном состоянии – при выключении питания в sleep-режиме). НЧ LC – фильтр L1C1 препятствует проникновению ВЧ составляющей напряжения в тракт питания дисплея и всего устройства. Дроссель L1 представляет собой катушку, намотанную на ферритовом кольце D3.1-d1.7-h2.15 размером (D/d/h) 3,1×1,7×2,15 мм тройным проводом ПЭЛ-0,22. При количестве витков 5 индуктивность дросселя составляет около 40 мкГн, а омическое сопротивление около 0,03 Ом. Материал, из которого изготовлено кольцо, к сожалению, неизвестен, но его магнитная проницаемость μ, по опыту автора, существенно превышает 2000. Кольцо выпускается с уже скруглёнными кромками. Вместо этого кольца можно использовать кольцо К4×2,5×1,7 размером (D/d/h) 4×2,5×1,7 мм из материала М2000НМ1 (μ = 2000), у которого необходимо скруглить острые кромки (см. далее). На этом кольце следует намотать 10 витков тройным проводом ПЭЛ-0,22. При этом индуктивность составит около 30 мкГн, а омическое сопротивление около 0,04 Ом. Автор проверял работу устройства на обоих кольцах. Эта проверка показала, что прибор прекрасно работает при использовании любого из двух колец.

Следует отметить один интересный момент. Как в описании самого дисплея (см. datasheet на дисплей 1,5″ epaper-B), так и в описании контроллера

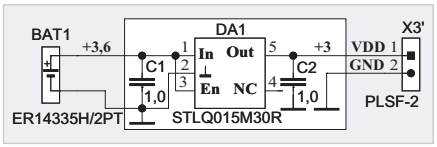


Рис. 3. Плата стабилизатора и кабель питания

UC8151, используемого в нём, в качестве дросселя (L1, рис. 2) применена катушка индуктивности для поверхностного монтажа, имеющая индуктивность 10 мкГн с максимальным током 1 А и максимальным током насыщения 0,5 А, при котором магнитопровод входит в насыщение, или, другими словами, перестаёт выполнять свои функции. Омическое сопротивление этой катушки 0,1 Ом. Выбор именно такой катушки индуктивности определяется отнюдь не её максимальным током насыщения, поскольку, как показали измерения (см. далее), максимальный ток при работе DC-DC-конвертора составляет всего 3 мА, и о насыщении магнитопровода катушки при таком токе вообще можно забыть. Дело совсем в другом, а именно – в омическом сопротивлении катушки (0,1 Ом). Действительно, с одной стороны, чтобы не пропускать ВЧ-составляющую напряжения DC-DC-конвертора в тракт питания, индуктивность катушки должна быть как можно больше (индуктивное сопротивление $X_L = \omega L$), с другой, для повышения эффективности работы DC-DC-конвертора омическое сопротивление катушки должно быть как можно меньше. С учётом того, что номинал резистора, подключённого к истоку транзистора (R2), составляет всего 0,47 Ом, омическое сопротивление катушки должно быть существенно меньше 0,47 Ом. Поэтому в описании контроллера и применена подобная катушка. Но в нашем случае при использовании вышеупомянутых двух катушек их индуктивность существенно больше 10 мкГн (40 мкГн и 20 мкГн), а омическое сопротивление существенно меньше 0,1 Ом (0,03 Ом и 0,04 Ом). Вот поэтому эти катушки идеально работают. Стоимость колец существенно различается: если кольцо К4×2,5×1,7 можно приобрести за 7–10 руб., то кольцо D3.1-d1.7-h2.15 стоит на порядок дороже – около 75 руб. Однако, поскольку оно выпускается с уже скруглёнными кромками, при его применении отсутствует «головная боль» со скруглением острых кромок, требующимся для кольца К4×2,5×1,7 (эта технология подробно описана далее).

Конденсаторы C3–C12 положены по штату работы дисплея. На плате расположен 24-контактный разъём PFC24/0.5mm (X2), к которому подключается шлейф дисплея. Разъём X1 подключается к ответному разъёму X4 платы МК (рис. 1).

Здесь следует сделать некоторое отступление по поводу причины, по которой сконструирована плата дисплея. Изначально автором был приобретён дисплей WFT0000CZ04 (1,54″ epaper-B) с разрешением 200×200 пикселов с уже готовой платой дисплея. Однако батарейка CR2477, которую автор изначально использовал, не проработав и двух дней, безнадёжно «села». В связи с этим были проведены измерения потребления тока всей платы, которые показали следующее. В активном режиме обновления информации на экране дисплея, когда работает DC-DC-преобразователь, потребление тока составило около 15 мА, а в отдельные моменты – до 18 мА. Время потребления тока в активном режиме составило около 8 секунд. В sleep-режиме потребление тока составило около 3 мА. Кроме того, напряжение батарейки CR2477 в активном режиме падало до 2,75 В, а в sleep-режиме (3 мА) – до 2,9 В. Такое положение вещей автора не устроило, поскольку, как указано в описании как дисплея, так и контроллера (UC8151), потребление тока в активном режиме составляет не более нескольких мА, а в sleep-режиме – 0,6 мкА. Это и послужило причиной конструирования платы дисплея по схеме рис. 2.

Диоды VD1–VD3 (в оригинальной схеме это MBR0530 в корпусе SOD-123 размером 1,6×3,7 мм) были заменены диодами PMEG3010 в более компактном корпусе SOD323 размером 1,3×2,5 мм. Кроме того, PMEG3010 по сравнению с MBR0530 имеют меньшее прямое падение напряжения и больший максимальный ток (1 А против 0,5 А у MBR0530). Катушка индуктивности L1 (в оригинальной схеме это катушка для поверхностного монтажа номиналом 30 мкГн и размером (D/h) 7×2,5 мм) была заменена катушкой, намотанной на вышеупомянутом кольце. С оригинальной платы дисплея были удалены все компоненты, кроме дисплея, который к ней приклеен, а плата дисплея по схеме рис. 2 была приклеена пористой лентой с двусторонним липким слоем к поверхности, очищенной от компонентов оригинальной платы (см. далее). В связи с этим автор рекомендует приобретать только один дисплей без платы дисплея. Измерения потребления тока платы по схеме рис. 2 показали следующее. В активном режиме потребление тока составило около 4 мА, а в sleep-режиме тестер показал нулевой ток (на диапазоне тестера в 30 мА ток 0,6 мкА покажет нулевое значение). Кроме того, время активного режима снизилось до 3 секунд. Это вполне устроило автора.

В связи с тем, что, как указано выше, напряжение батарейки CR2477 существенно падало в активном режиме работы дисплея, автор применил более ёмкую батарейку ER14335 (размером 2/3 AA) с напряжением 3,6 В ёмкостью 1,65 А·ч и микропотребляющий стабилизатор напряжения STLQ015M30R с выходным напряжением 3 В и падением напряжения не более 50 мВ при токе 150 мА (рис. 3). Потребление тока этого стабилизатора составляет, согласно описанию (datasheet), не более 1 мкА при токе 150 мА, а при меньшем токе, соответственно, ещё меньше. Плата стабилизатора (рис. 3) приклеена к батарейке пористой лентой с двусторонним липким слоем (см. далее). От стабилизатора отходит кабель питания, на второй стороне которого расположен разъём X3', который вставляется в ответный разъём X3 платы МК (рис. 1). Батарейка ER14335/2PT (BAT1) снабжена двумя приваренными контактами, на которые надеты 2 цанговых гнезда кабеля от платы стабилизатора.

Программные средства

Наиболее полную информацию о программировании E-ink (или E-paper) дисплеев можно найти на сайтах www.e-paper-display.com и www.good-display.com, поскольку из описания (datasheet) контроллера UC8151 понять, как программируются подобные дисплеи, достаточно проблематично. На этих сайтах приведены примеры программ для плат STM32 (на C), Arduino (на C), Raspberry Pi (на Phyton) и ESP8266 (на Phyton). Наличие подобных программ позволяет с их помощью легко запрограммировать E-ink дисплей в более простом, 8-разрядном МК, с программной памятью всего 16 кБ, каковым является EFM8SB20F16. Наиболее легко читаемы программы на Phyton'е, поскольку в них вся программа представлена всего одним текстовым *.py-файлом, в отличие от программ на C, где имеются бесчисленные дополнительные *.h-файлы,

включённые (#include <...>) в основную программу, и «лазить» по этим файлам, чтобы понять работу основной программы, – дело очень неприятное, долгое и неблагодарное.

В основном при программировании E-ink дисплея имеются две проблемы. Первая – это инициализация дисплея. В примерах программ такая инициализация заключается в выводе в дисплей порядка 30–35 команд и данных, определяющих разрешение дисплея, установку счётчиков строк и столбцов на начало, различные моменты установки напряжений DC-DC-конвертора, температурные параметры и т.п., и особой сложности (инициализация) не представляет. Для решения этой проблемы нужно просто очень внимательно, без ошибок переписать (из примеров) все эти команды и данные, передаваемые вслед за командами, в свою программу, и всё будет работать.

Иное дело – вывод информации в дисплей. Здесь имеется два варианта. Но прежде чем описывать эти два варианта, сделаем некоторое отступление относительно того, что и как требуется вывести в E-ink дисплей.

Исходя из разрешения дисплея 200×200 пикселов для отображения 3 параметров: давления, температуры и влажности – имеет смысл расположить показания этих параметров на дисплее в три ряда. В первом ряду – показания давления, во втором – температуры, в третьем – влажности. Если, например, давление равно 751 мм рт. ст., температура 25°C, а влажность 45%, то в первом ряду должно быть число «751» и два символа, отражающих размерность давления, например, «мм Hg» (достаточно часто используемая), поскольку символы «мм рт. ст.» занимают много места. Другими словами, по горизонтали должно располагаться 5 символов. Такое же количество символов должно располагаться во втором и третьем ряду. Например, для температуры это будет знак («+» или «–»), показания, например, «25», символы «°» и «C», а для влажности – показания, например, «45», размерность, «%», и какой-либо символ, отражающий само понятие влажности, например, капля с делениями, как часто его обозначают. Причём, поскольку перечисленных символов для влажности всего 4, перед показаниями должен присутствовать символ пробела, чтобы общее количество символов было 5, как для давления и температуры.

Дисплей в программном смысле представляет собой строки и столбцы. Каждая строка имеет ширину ровно 8 пикселов, которым соответствует 8 бит или 1 байт. Таким образом, в дисплей по вертикали поместится ровно 25 строк (25 × 8 = 200), начиная с нулевой и кончая 24-й. Кроме того, в дисплее имеется ровно 200 столбцов, начиная с нулевого и кончая 199-м. Если показания располагаются в 3 ряда, значит, каждый символ должен состоять из 8 строк (8 × 3 = 24), и для заполнения всего дисплея должна присутствовать ещё одна пустая строка, чтобы общее количество строк было 25. Таким образом, каждый символ по вертикали должен занимать 8 × 8 = 64 пикселя. Поскольку по горизонтали имеется 200 столбцов, а символов 5, то, разделив 200 на 5, получим 40. Но между символами, чтобы они не сливались, должны быть предусмотрены пробелы хотя бы в 3 пикселя. Если ширина символа будет, например, 37, то по горизонтали 5 символов займут 5 × 37 = 185 пикселов, и останется ещё 15 пробельных столбцов, что составляет по 3 пробела на символ. Таким образом, один символ должен занимать поле в 64×37 пикселов или 8 однобайтных строк по 37 столбцов, и для его вывода в дисплей понадобится 8 × 37 = 296 байт. О том, как сформировать эти 296 байт, будет рассказано далее. Вывод информации в дисплей осуществляется по строкам и столбцам.

Теперь, возвращаясь к прерванной последовательности изложения, можно уже пояснить, что это за два варианта вывода.

В первом варианте специальными командами, посылаемыми в дисплей, можно сформировать 3 окна высотой по 8 строк и шириной 200 столбцов. При этом нужно указать номер начальной и конечной строки окна и номер начального и конечного столбца. Например, для первого ряда это может быть 0-я и 7-я строки, или, пропустив одну пробельную строку сверху, 1-я и 8-я строки. А номера начального и конечного столбцов указать как 0-й и 199-й. В дисплее организованы счётчики строк и столбцов. При выводе символа в окно счётчики строк и столбцов могут инкрементироваться (увеличиваться на единицу) или декрементироваться (уменьшаться). Это зависит от команды, посылаемой в дисплей. Для простоты объяснения предположим, что они инкрементируются. Пусть начальные и конечные значения счётчиков строк и столбцов равны 1 и 8 и 0 и 199 соответственно. Тогда при выводе символа (296 байт) происходит следующее. При выводе восьми байт (с 0-го по 7-й из 296) в 0-й столбец счётчик строк инкрементируется, но после вывода 7-го байта (в строку номер 8) счётчик строк автоматически устанавливается на 1-ю строку, а счётчик столбцов инкрементируется, т.е. устанавливается с 0-го на 1-й столбец. Это позволяет не следить за счётчиками, в связи с чем все 296 байт каждого символа, если их выводить подряд, выведутся туда, куда нужно. Аналогично можно сформировать 2-е окно для 2-го ряда и 3-е окно для 3-го ряда символов, и таким же способом вывести всю необходимую информацию в дисплей. Кстати, такой же способ вывода применяется при выводе информации в OLED-дисплей. На первый взгляд кажется, что вывод информации в E-ink дисплей достаточно простой. Но здесь кроется один неприятный момент, или своеобразная ловушка (по времени). Дело в том, что, после того как информация записана в памяти контроллера, для того чтобы, она попала на экран дисплея, необходимо дать команду, которую в программах называют “update” (обновить) или “refresh” (освежить). В русском языке есть понятие «освежить в памяти». На самом деле такой «рефреш» при переносе изображения из записанного в памяти контроллера на экран дисплея занимает несколько секунд, в отличие от OLED-дисплея, у которого информация на экране появляется сразу же после записи в память контроллера. Например, «рефреш» для каждого подобного окна занимает около 5 секунд, а для всех 3 окон – более 15 секунд. Кроме того, при таком оконном выводе требуется предварительно ещё очистить весь дисплей, т.е. заполнить всю его память (а это 25 × 200 = 5000 байт) числами ffh, и в конце дать ещё одну команду «рефреш». Причём, как ни странно, очистка экрана занимает около 3 секунд. В результате подобный оконный вывод займёт около 22 секунд (это было проверено секундомером). Естественно, при таком оконном выводе дисплей работает по полной программе и потребляет значительный ток в течение 22 секунд. Даже при обновлении показаний давления, температуры и влажности один раз в 10 минут всё равно 22 секунды – непозволительно долго. Здесь уже никакой батарейки не хватит, или её придется достаточно часто менять. В связи с этим автор задался вопросом: а нельзя ли вывести всю информацию на экран дисплея за

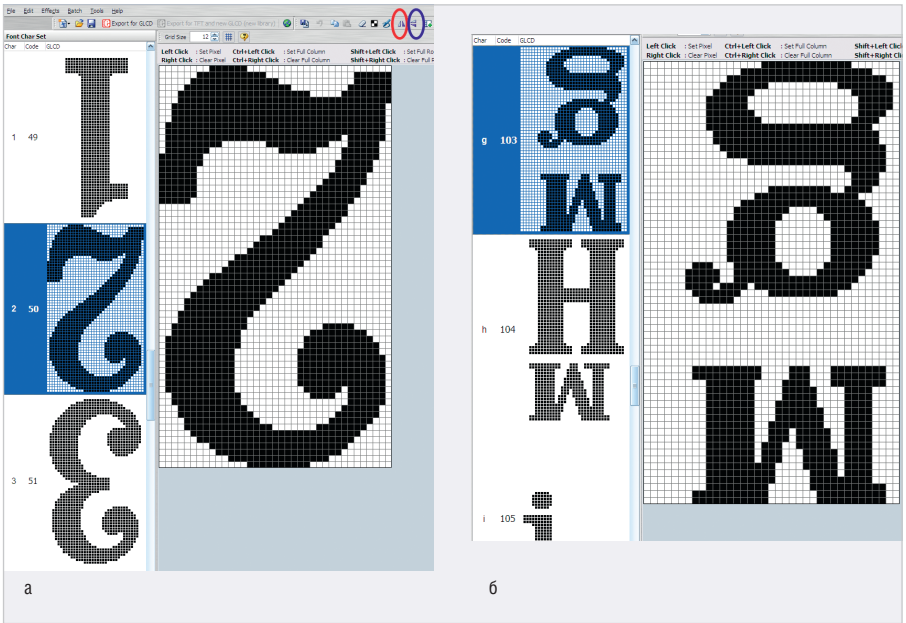


Рис. 4. Примеры отображения символов в программе GLCD Font Creator

один раз так, как это делается, например, при очистке экрана? Тогда появился ещё один вариант.

2-й вариант вывода. Пусть последние (пятые) символы в каждом из 3 рядов отражают размерности измеренных величин, например, «м/g» для давления, «С» для температуры и «капля» для влажности. Тогда, чтобы заполнить самый правый (199-й) столбец, можно вывести один верхний пробел шириной в одну строку (один пустой байт, равный ffh), далее вывести последние 8 байт (из 296) символа «м/g», затем вывести последние 8 байт символа «С» и далее вывести последние 8 байт символа «капля». В результате выведется ровно 25 строк (25 байт) в 199-й столбец, после чего счётчик столбцов автоматически декрементируется и установится на 198-й, а счётчик строк, инкрементируясь и достигнув 24, установится на нулевую строку. После этого опять пропускаем верхнюю строку и выводим, как и ранее, по 8, но уже предпоследних байт каждого символа. Таким же образом выводим и все остальные из 296 байт вышеуказанных символов. Далее аналогичным образом выводим и все остальные символы (уже числа) показаний давления, температуры и влажности. Далее даём одну команду «рефреш», и на этом весь вывод информации в дисплей заканчивается. Кроме того, при таком «безоконном» выводе очищать дисплей (с командой «рефреш») уже не потребуется. Запрограммировав подобный вывод в МК и запустив его, автор стал наблюдать за дисплеем. Экран замигал, и через 3 секунды и на дисплее полностью отразились все показания и их размерности. Такой вывод информации в дисплей автор и взял на вооружение. На первый взгляд кажется, что подобный вывод более сложен и потребует больших усилий с точки зрения программирования. Однако практика показала, что ничего сверхсложного в этом нет. В дополнительных материалах к статье приведён текст фрагмента программы с подобным выводом. Просмотрев этот фрагмент, можно убедиться, что всё довольно просто.

Теперь по поводу получения кодов символов. Для этого автор использовал программу GLCD Font Creator v. 1.2.0.0 (от компании MikroElektronika – www.mikroe.com). В ней можно выбрать практически любой шрифт, его параметры (например, размер, свойства – жирный, обычный, наклонный и т.п.). Программа сформирует все символы данного шрифта и выведет их на экран монитора компьютера. Далее необходимо убрать все пустые строки и столбцы сверху, снизу и справа, чтобы каждый символ вписался в окно определённого размера (в нашем случае это 64×37 пикселов). Это делается специальными опциями с пиктограммами, на которые необходимо навести курсор мыши и кликнуть. Для того чтобы цифры и символы выводились на дисплей справа налево и сверху вниз, каждый символ необходимо перевернуть вверх ногами (отразить по вертикали) и отразить по горизонтали. Это можно сделать всего двумя кликами мыши по соответствующим пиктограммам в меню программы (синий и красный овалы на рис. 4а). В этом случае, например, двойка будет выглядеть как на рис. 4а, а символ м/g – как на рис. 4б.

После этого, нажав пиктограмму “Export for GLCD” и в открывшемся окне – опцию “microC”, получим файл на С (его также следует назвать), в котором будут содержаться все 296 байт для каждого символа. Их остаётся только привести в приемлемый для программы вид и сформировать из них двумерный массив, например, MD[19][296], где первое измерение [19] отражает количество символов, а второе [296] – количество байт для каждого символа.

Для цифр был выбран шрифт Clarendon Condenced размера 65, жирный. Для него был сформирован файл на С с названием:

```
//GLCD FontName : Clarendon_Condensed37x64
//GLCD FontSize : 37 x 64
```

который автор и использовал для отображения цифр.

Шрифт Clarendon был выбран по следующим соображениям. Он не такой строгий, как, например, Arial или Courier New, но и не слишком вычурный. По сравнению с Times New Roman жирным Clarendon значительно «жирней», т.е. чёрные пиксели каждого символа занимают бóльшую площадь окна 64×37 пикселов, и, естественно, символ легче читается. Кроме того, на взгляд автора, шрифт Clarendon достаточно симпатичный.

Символы, которых нет в этом шрифте, были буквально нарисованы в программе GLCD Font Creator. Это следующие символы: «м/Н» (оба символа нарисованы), «м/g» («м» нарисована, «g» взята готовая из шрифта меньшего размера) – таким образом, 2 символа «м/Н» и «м/g» превращаются в надпись «мм/Hg»; «°» нарисован, «С» был немного сужен, «%» (нарисован – оригинальный «%» сужен), символ капли с делениями (нарисован), символы «+» и «–» нарисованы. Символ пробела взят готовый (там все нули). Таким образом, общее количество символов вместе с цифрами (их 10 – от «0» до «9») составляет 19, поэтому, как указано выше, двумерный массив получился размерностью MD[19][296]. Рисование в программе очень простое: наведя курсор мыши на белый пиксель и нажав левую кнопку, можно получить чёрный пиксель, а наведя на чёрный и нажав правую кнопку – белый.

И последнее, что следует добавить по поводу программных средств, – это изменения в инициализации устройств, поскольку МК EFM8SB20 отличается от МК EFM8SB10 [1].

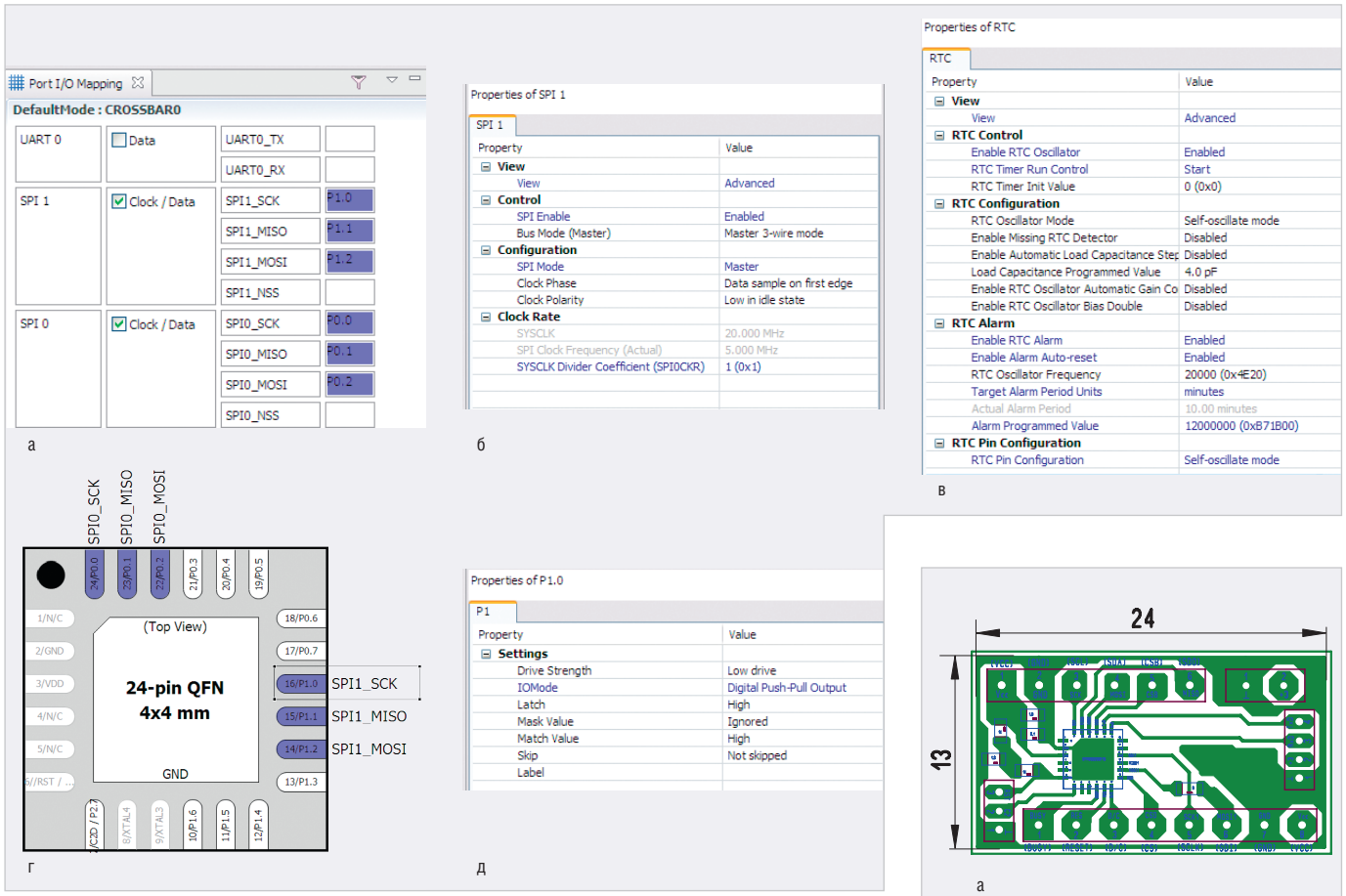


Рис. 5. Настройка устройств МК: а) разрешение SPI0 и SPI1, б) настройка SPI1, в) настройка RTC, г) настройка портов, д) пример настройки порта P1.0

Во-первых, необходимо разрешить оба SPI – SPI1 и SPI0 (рис. 5а) и настроить каждый из них, как показано на рис. 5б. Во-вторых, настроить RTC, как показано на рис. 5в (эта настройка отличается от настройки RTC в EFM8SB10 [1]). В-третьих, в связи с разными корпусами МК EFM8SB10 (QFN20) и EFM8SB20 (QFN24) настройка портов должна соответствовать рис. 5г, д. Все остальные настройки – те же самые, что и в [1].

После трансляции программы в специальном окне среды программирования Simplicity Studio v.4 (от Silicon Laboratories) отобразится результат этой трансляции (сообщение):

```
Program Size: data=117.1 xdata=0
const=0 code=11354
LX51 RUN COMPLETE.  0 WARNING(S),
0 ERROR(S)
Finished building target: EFM8SB-
20F16G-A-QFN24.omf
```

Из этого сообщения можно заключить, что в программе использована почти вся внутренняя оперативная память с прямой адресацией объёмом 128 байт (data=117.1), а внешняя оперативная память с косвенной адресацией объёмом 4 кБ не использована (xdata=0). Кодовая часть программы использует далеко не всю программную память объёмом 16 кБ, или 16 384 байта (code = 11 354). Остаток программной памяти составляет: 16 384 – –11 354 = 5030 байт → 5 кБ. Кроме того, при трансляции применена так называемая small-модель, в которой данные располагаются в области памяти с прямой адресацией (data). В этом случае, во-первых, существенно экономится программная память, а во-вторых, программа работает намного быстрее.

Разводка и внешний вид плат

Разводка плат (рис. 6–8) сделана автором с помощью программы SprintLayOut v.6. Файл разводки в формате *.lay6 приведён в дополнительных материалах к статье на сайте журнала.

Из рисунков разведённых плат и их внешнего вида можно заключить, что разводка плат очень проста, а сами платы небольшого размера. Здесь следует добавить, что если не предполагается программирование МК с помощью COM-порта компьютера, то разъём (X2, рис. 1) не нужен, поэтому припаивать его (и сверлить для него отверстия) совсем не обязательно (в данном случае он отсутствует). На рис. 6а в его

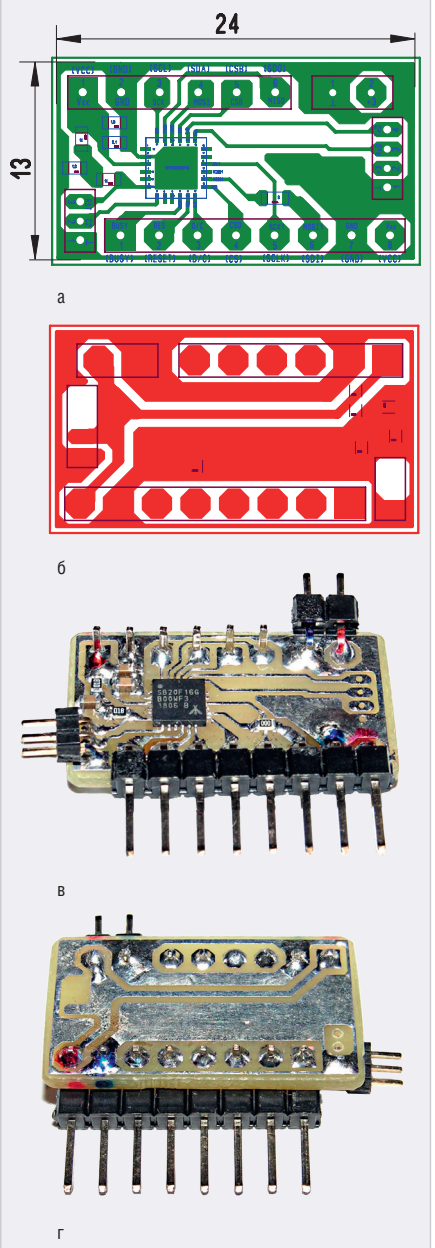


Рис. 6. Разводка и внешний вид платы МК: а, в – вид со стороны расположения компонентов; б, г – вид с обратной стороны

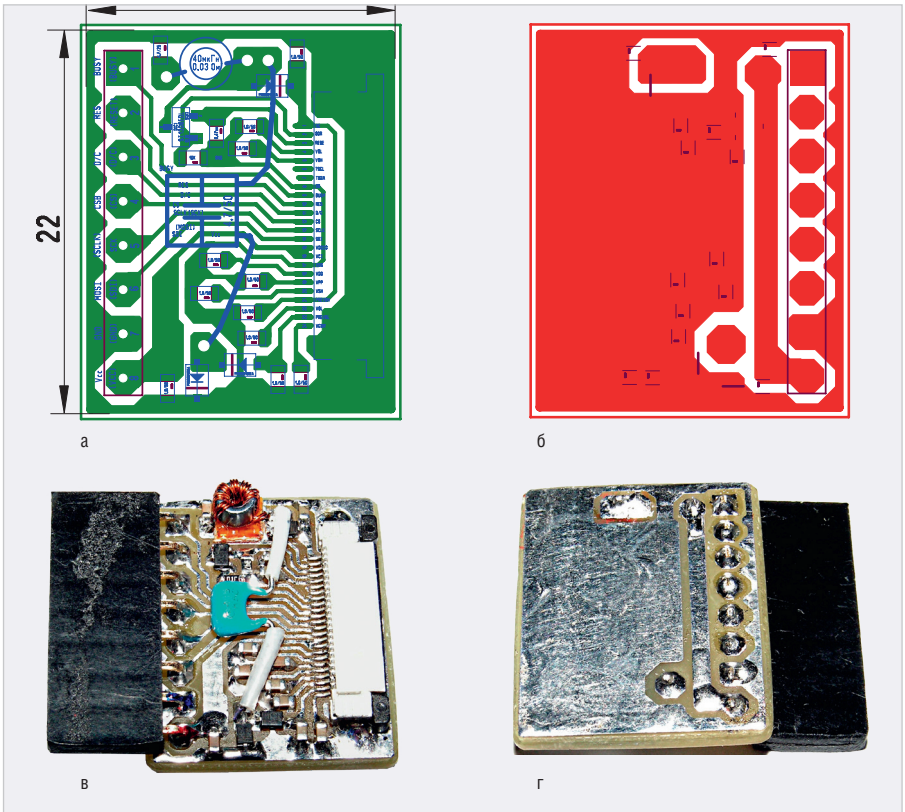


Рис. 7. Разводка и внешний вид платы дисплея: а, в – вид со стороны расположения компонентов, б, г – вид с обратной стороны

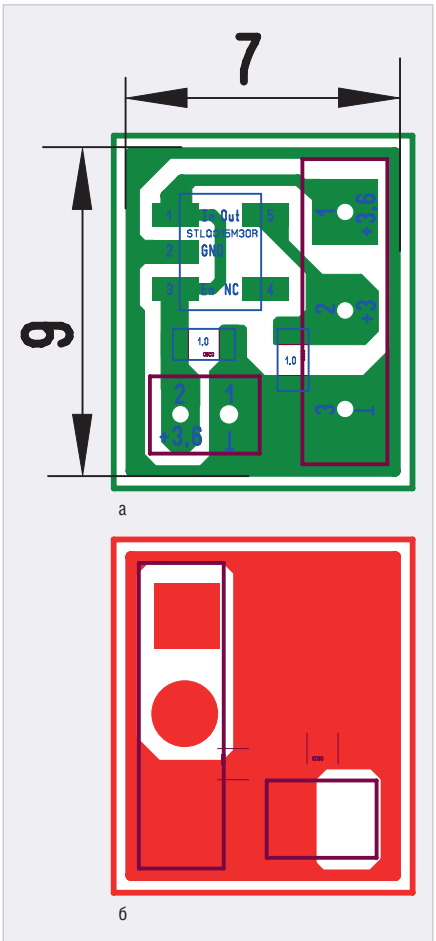


Рис. 8. Разводка платы стабилизатора: а – вид со стороны расположения компонентов, б – вид с обратной стороны

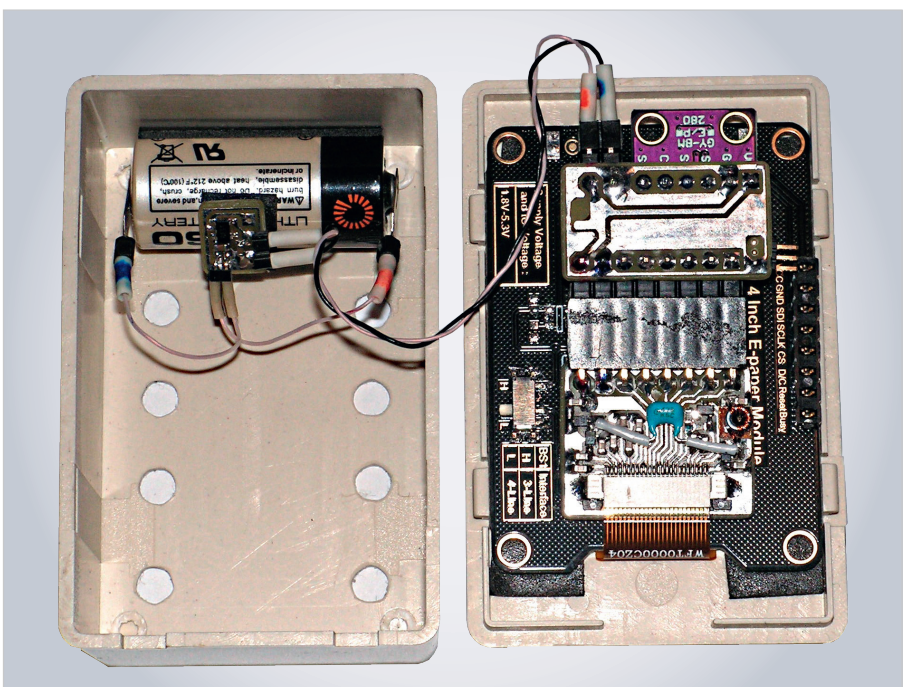
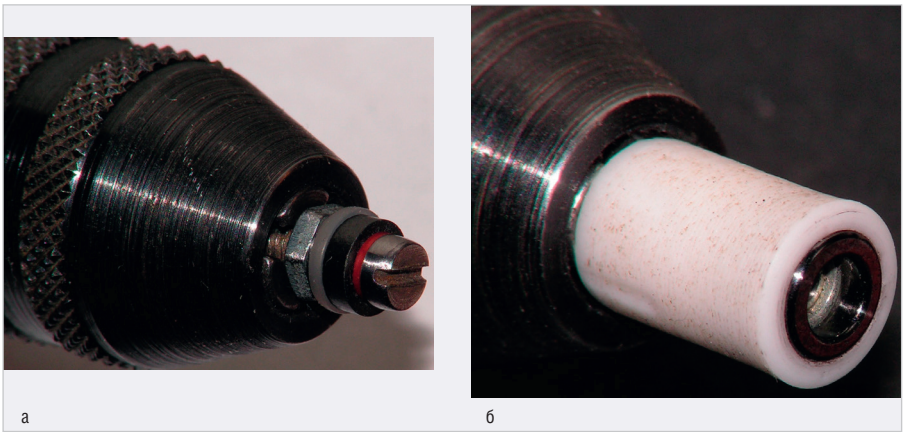


Рис. 9. Устройство в открытом корпусе

место справа в середине, на рис. 6б, г – слева в середине. Для того чтобы катушка индуктивности не касалась дорожек платы, под кольцо с намотанным проводом к плате приклеена пластина из тонкого (1 мм) текстолита (рис. 7в). Чтобы выводы единственного конденсатора, предназначенного для навесного монтажа (он голубого цвета на рис. 7в), не касались дорожек платы, на них надет фторопластовый кембрик (тефлоновая трубка). В связи с простотой на рис. 8 приведена только разводка платы стабилизатора; её можно заменить приклеенной к батарейке пористой лентой с двусторонним липким слоем (см. далее рис. 9). Через все переходные отверстия плат (со слоя на слой) проходят либо контакты разъёмов, либо выводы навесных компонентов, которые необходимо пропаять с двух сторон плат. Это позволило не использовать металлизацию отверстий, технология которой в домашних условиях весьма проблематична и поэтому неприемлема.

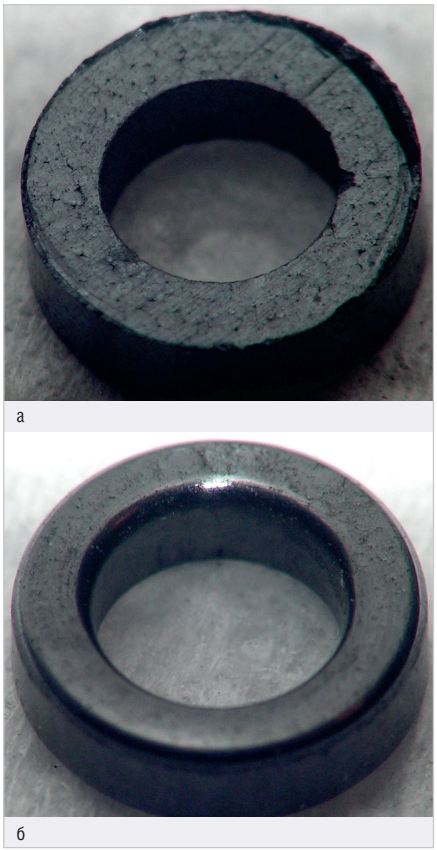
Конструкция и результаты работы устройства

Устройство расположено в корпусе (рис. 9) размером 70×45×28 мм (название корпуса «20-33» RUICHI) с защёлкивающейся крышкой (безвинтовое соединение). Для дисплея в крышке прорезано окно. Поскольку дисплей намертво приклеен к плате, с которой все компоненты были удалены (см. выше), эта плата приклеена к внутренней поверхности крышки пористой лентой с двусторонним липким слоем. К этой плате той же лентой приклеена плата дисплея, в которую вставлена плата МК, а в неё – плата с модулем BME280. Шлейф от дисплея перегнут через выемку и вставлен в плату дисплея. Батарейка приклеена той же лентой к внутренней поверхности второй половины корпуса,



а б

Рис. 10. Скругление кромок ферритового кольца: а – внешних, б – внутренних



а б

Рис. 11. Внешний вид ферритового кольца К4×2,5×1,7: а – необработанного, б – обработанного

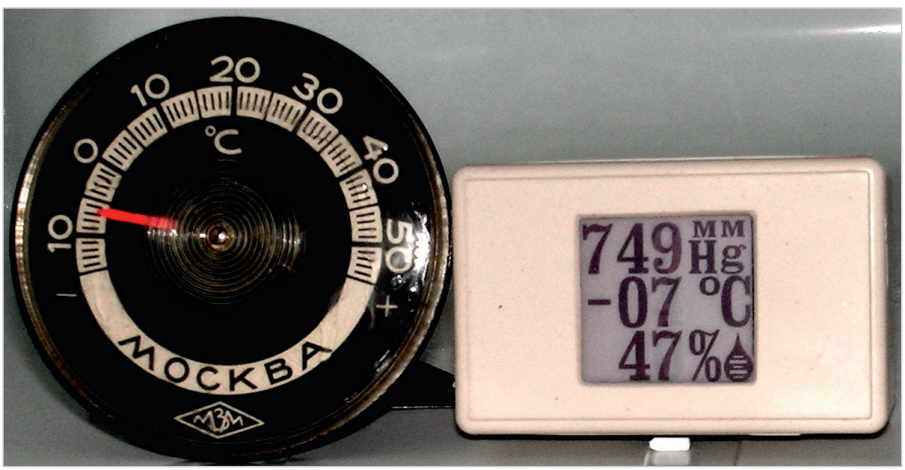


Рис. 13. Сравнение показаний уличного термометра и прибора, помещённых в морозилку холодильника

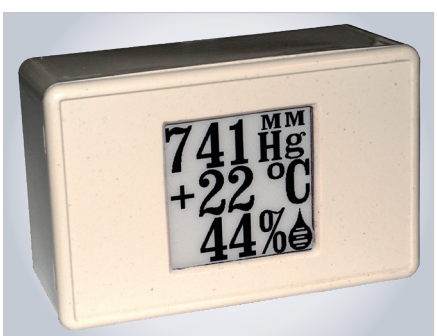


Рис. 12. Общий вид работающего прибора в сборе

а к ней – плата стабилизатора. Для доступа воздуха к устройству в этой половине корпуса просверлены 10 отверстий диаметром 4 мм.

Технология скругления острых кромок ферритового кольца очень проста. Для скругления внешних кромок кольцо зажимается винтом с гайкой и двумя пластиковыми шайбами. Винт вставляется в патрон дрели и, при её вращении острые кромки легко снимаются мелкой наждачной шкуркой в течение нескольких секунд (рис. 10а). Для скругления внутренних кромок кольцо с натягом вставляется во втулку (например, из фторопласта), которая также зажимается в патроне дрели (рис. 10б). Результат, как говорят, налицо (рис. 11). Конечно, не все кольца имеют дефекты, как на рис. 11а, однако автор всё же рекомендует стачивать острые кромки. Всей этой процедуры можно избежать, если использовать кольцо с уже скруглёнными кромками (рис. 7в).

После подключения кабеля питания от батарейки к плате МК прибор сразу начинает работать, и остаётся только защёлкнуть крышку (рис. 12).

Для проверки работоспособности прибора при отрицательных температурах он вместе с уличным термометром был помещён в морозилку холодильника примерно на 25 минут. Хотя показания дисплея при –6°С (рис. 13) выглядят несколько тусклее, чем при комнатной температуре (рис. 12), они вполне читаемы. Здесь следует заметить, что основное назначение прибора – это работа в комнатных условиях, а не для измерения параметров в морозилке холодильника.

Заключение

Применение E-ink дисплея с разрешением 200×200 пикселов совместно с малогабаритным микропотребляющим МК EFM8SB20F16 и готовым модулем с BME280 позволило сконструировать недорогой прибор небольшого размера, измеряющий атмосферное давление, температуру и влажность. По сравнению с подобным прибором с ЖКИ [1] описанный в статье прибор может работать от одной литиевой батарейки ER14335 до 10 лет и, кроме того, визуализация его показаний существенно улучшена за счёт практически типографского качества изображения символов.

Литература

1. *Кузьминов А.* Барометр-гигрометр-термометр с батарейным питанием на базе MEMS-датчика BME280, микроконтроллера EFM8SB10F8 и ЖКИ-модуля H1313 // Современная электроника. 2022. № 7, 8.
2. *Кузьминов А.Ю.* Связь между компьютером и микроконтроллером. Современные аппаратные и программные средства. М.: Перо, 2018.
3. *Кузьминов А.* Программирование микроконтроллеров EFM8 с помощью встроенного загрузчика программ // Радио. 2018. № 12.