

Pinboard II Быстрый старт. AVR.

Содержание.

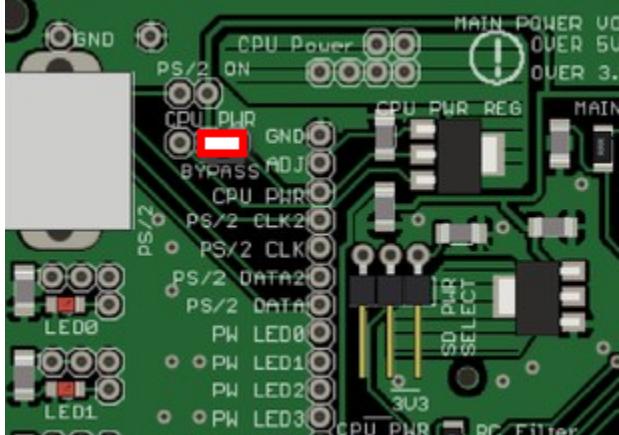
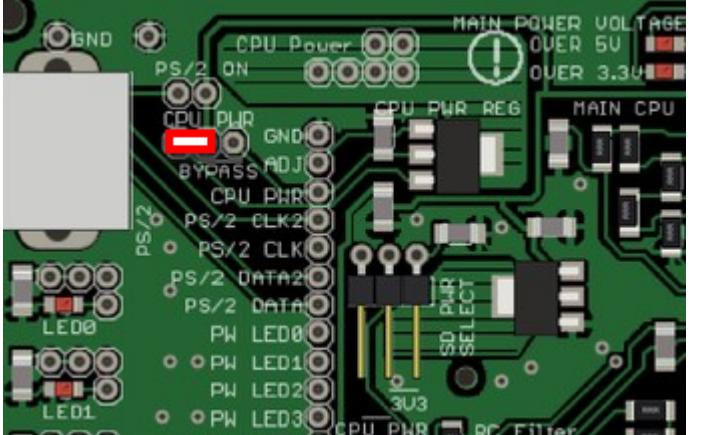
1. Конфигурация питания процессорного модуля.
2. Конфигурация линий связи.
3. Подготовка проводов и соединение джамперов.
4. Запуск и наблюдение за демопрограммой.
5. Активация бутлоадера и прошивка.
6. Подключение дисплея и кнопок
7. Тесты расширенной демопрограммы.

1. Конфигурация питания процессорного модуля.

Важнейший этап необходимый при КАЖДОЙ смене процессорного модуля. Дело в том, что у разных контроллеров разное питающее напряжение. Например, у AVR оно может достигать 5 вольт, тогда как у STM32 3.3 вольта уже предел.

Чтобы обезопасить контроллер, схема выбора напряжения сделана таким образом, чтобы контроллерный модуль сам определял свое питающее напряжение. Т.е. конфигурационные резисторы установлены на контроллерном модуле.

Но есть нюанс! Дело в том, что стабилизатору шины CPU POWER требуется около 2 вольт разницы между напряжениями входа и выхода, иначе он не сможет удержать необходимое напряжение. Так что если шина MAIN POWER настроена на 5 вольт и нам нужно получить 5 вольт на CPU POWER, то необходимо одеть перемычку BYPASS, которая исключит стабилизатор из схемы питания, пустив напряжение в обход. Соединив шины Main Power и CPU Power.

	
Стабилизатор заблокирован. BYPASS режим. Шины CPU Power и MAIN Power соединены накоротко.	Исходное положение. Стабилизатор активен. Напряжение на шине CPU Power задается платой контроллерного блока.

ВАЖНО!!!

При смене контроллерного модуля в ОБЯЗАТЕЛЬНОМ порядке возвращайте перемычку BYPASS в исходное положение, чтобы питание шло через стабилизатор. Тем самым, вы исключите риск сжечь контроллер неподходящим напряжением.

2. Конфигурация линии связи

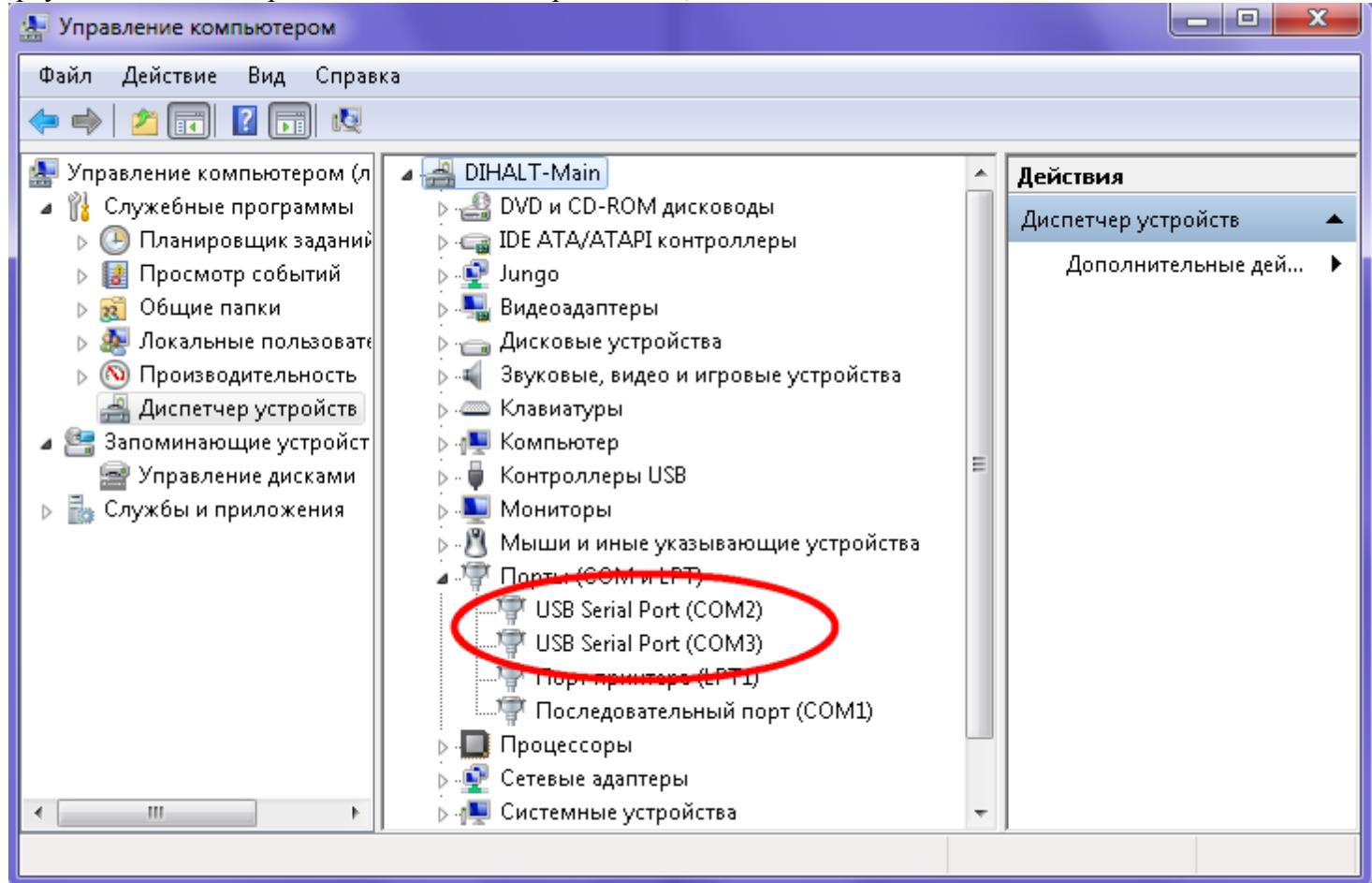
Для безопасности и удобства контроллер лучше прошивать через BootLoader. Бут это небольшая программка, которая сидит в верхних адресах памяти контроллера (в AVR модулях я прошиваю ее перед продажей платы сам. Во многих других контроллерах бут бывает прошит с завода) и слушает коммуникационный порт. В данном случае это UART. При старте первым делом контроллер выполняет

бутсектор и в течении нескольких секунд ждет кодовый сигнал с порта. Если сигнала нет, то происходит переход к основной программе контроллера.

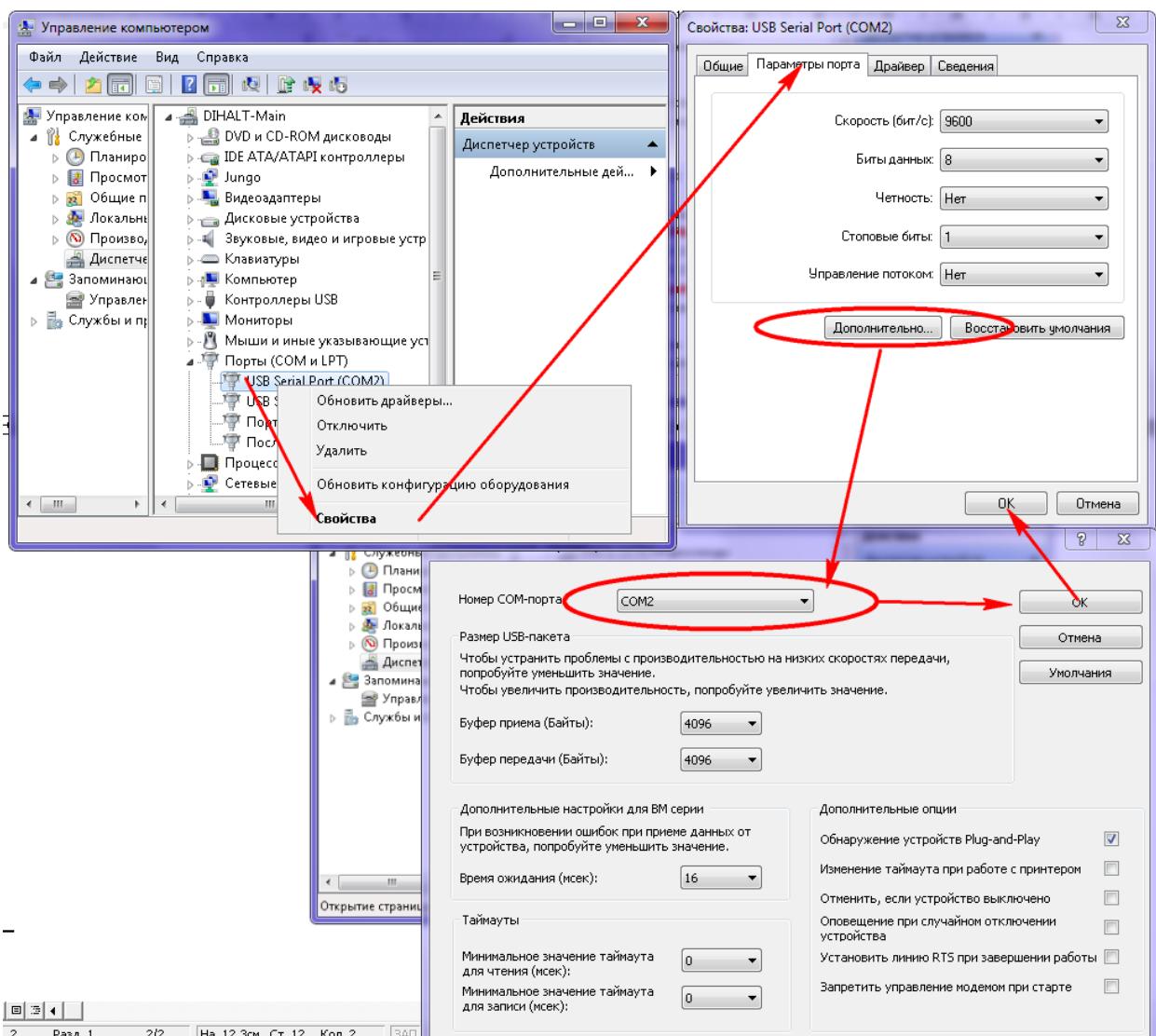
А вот если сигнал есть, то бут на него отвечает и готов принять и прошить в основную память прошивку.

Так что если мы хотим общаться с бутлоадером, то надо наладить линию связи, подключить UART к компьютеру, к виртуальному COM порту, чтобы программка AVRprog могла достучаться до бутлоадера.

Но тут есть ряд особенностей. AVRProg опрашивает только первые четыре COM порта. Поэтому наш виртуальный COM порт должен иметь номер от 1 до 4, не выше.

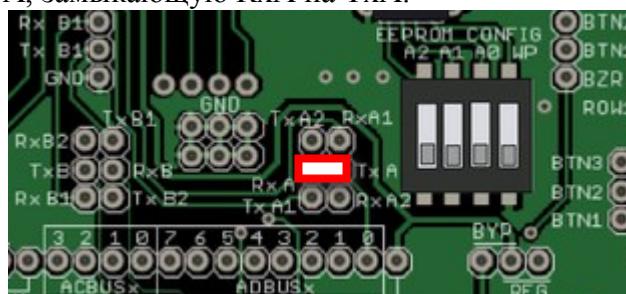


Если они иные, то нужно зайти в свойства какого либо из портов и выставить нужный порт принудительно:

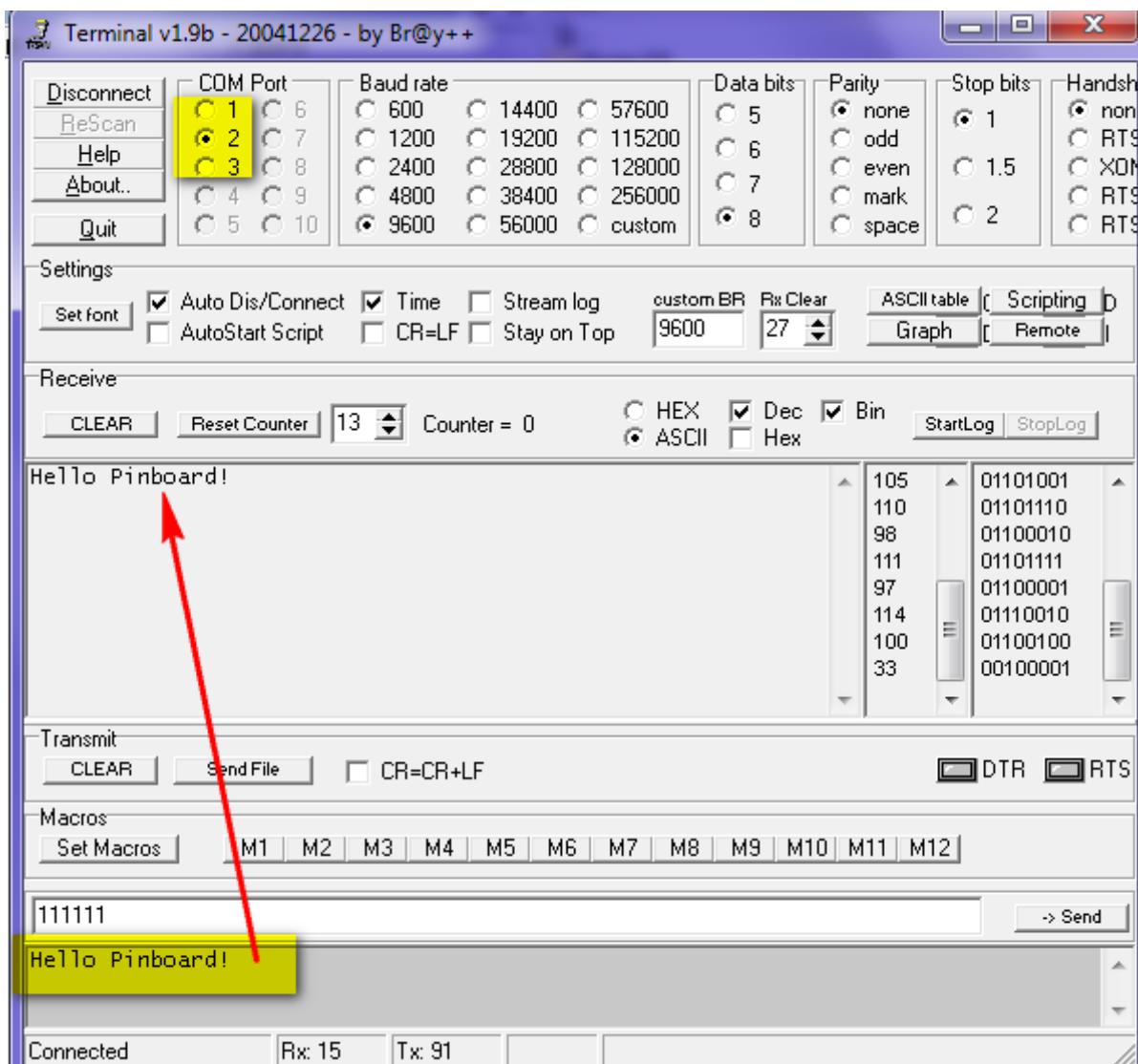


Правда при этом может отпасть что либо другое, что использовало ранее виртуальный COM порт. Например, блогуз передатчик для связи с сотовым. Но это мелочи ;)

После чего надо найти за какой канал какой COM порт отвечает. Запускаем программу **Terminal 1.9b** и замыкаем перемычку на канале A, замыкающую RxA на TxA:



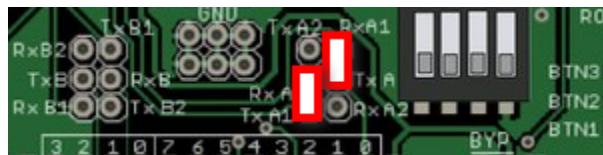
В терминале цепляемся на свободный порт:



И шлем что нибудь. Если символы вернулись, значит выбранный СОМ порт (в нашем случае СОМ2) соответствует каналу А. Не вернулось эхо? Значит не тот СОМ, ищем на каком у нас СОМ порту висит канал А.

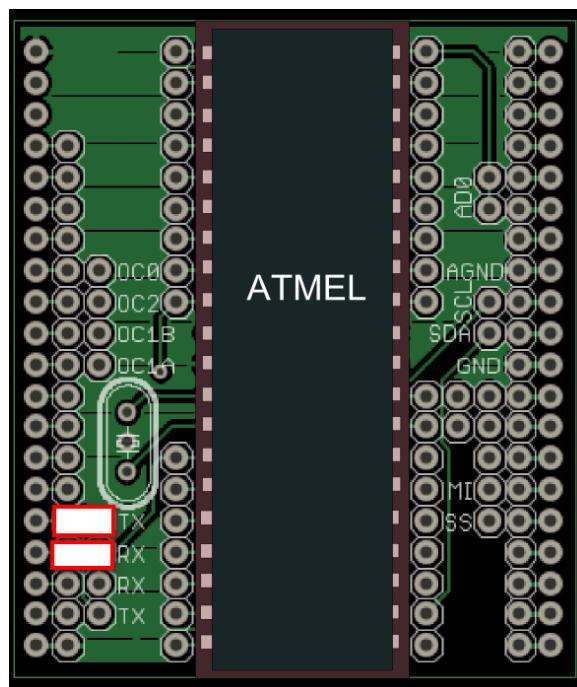
Потом замыкаем перемычку на канал В и таким же образом определяем порт канала В.

Следующим шагом нужно совместить любой канал, пусть это будет канал А с UART контроллера. Для этого надо одеть джамперы на коммутационной панели так, чтобы соединились соответствующие линии процессорного блока (он идет под номером 1, под номером 2 идет разъем модуля расширения) с соответствующим каналом FTDI. Покажу на примере подключения канала А с процессорным блоком.



Мы соединили джамперами TxA с RxA1 и RxA с TxA1. Канал В в данном случае никуда не подключен.

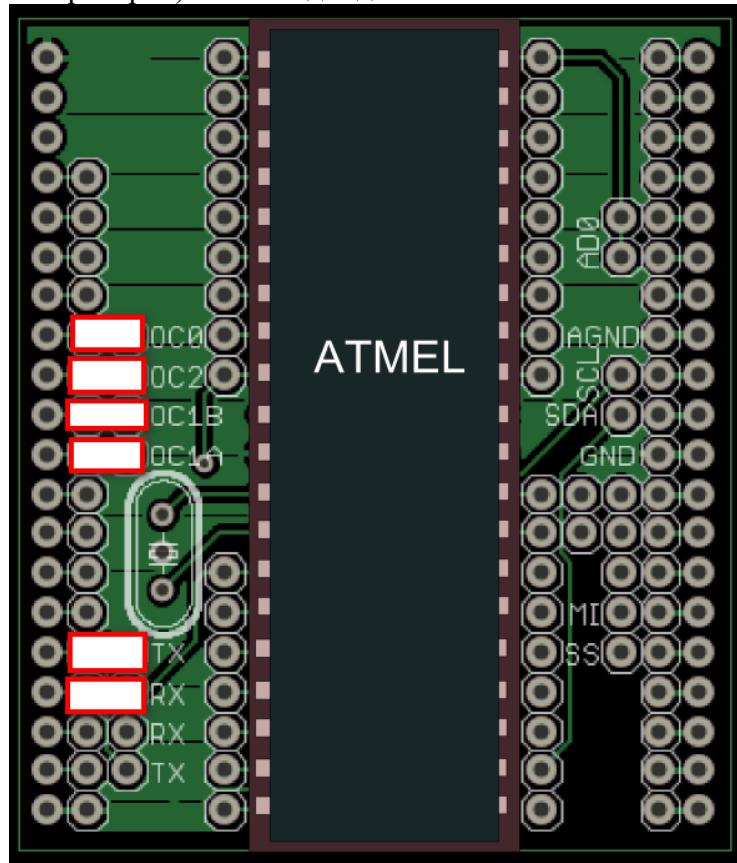
Теперь надо подключить джамперы UART на процессорном блоке, чтобы соединить линии связи от демоплаты, до Rx-Tx собственно микроконтроллера:



Обратите внимание, что на блоке один UART микроконтроллера дублирован на канал А и на канал В, для удобства. Если у микроконтроллера было бы два канала UART, то один был бы разведен на А, другой на В. Мы же подключили джамперы, соединив канал А с UART микроконтроллера.

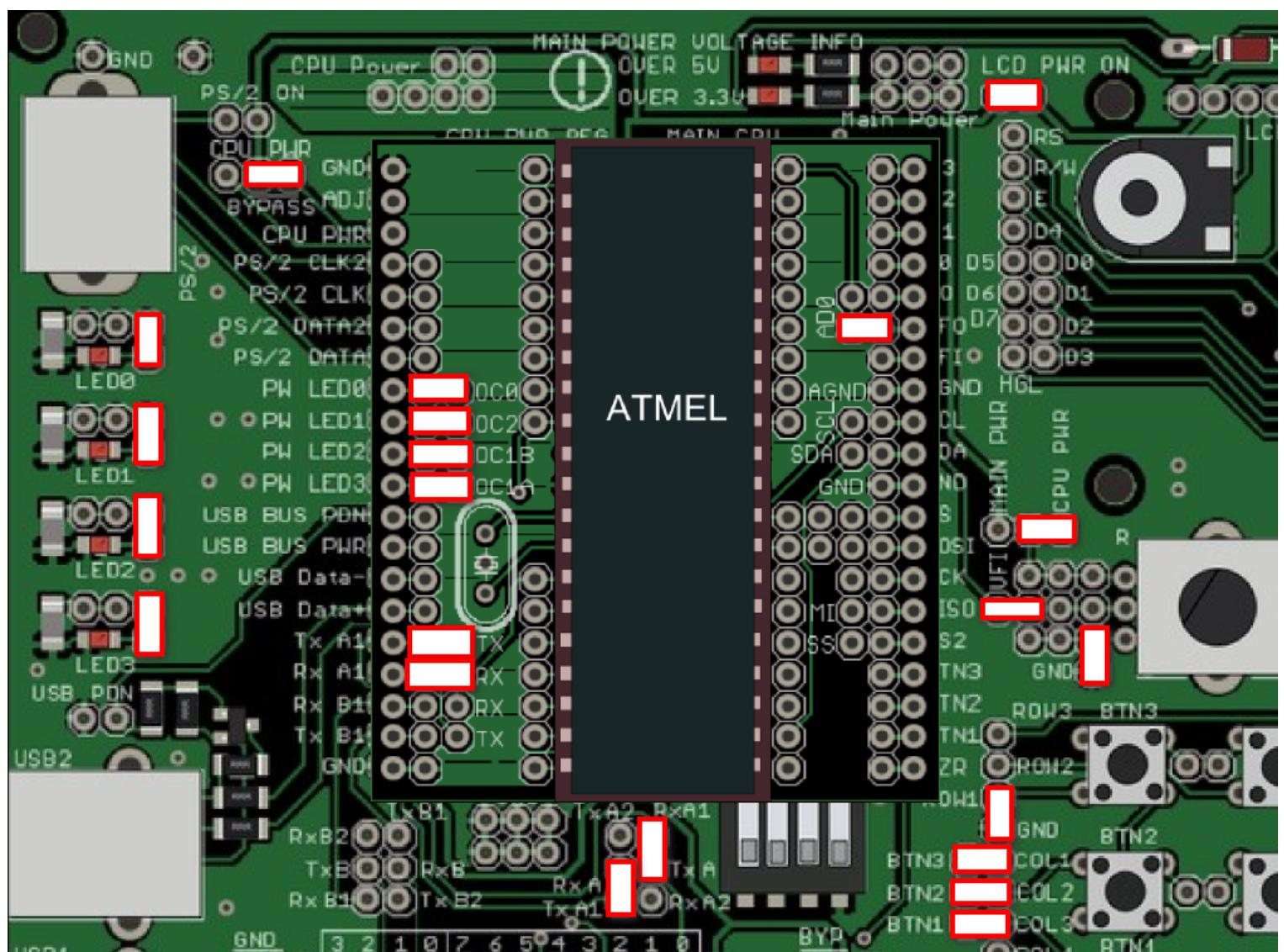
3. Подготовка проводов и подключение джамперов

Также не помешает подключить светодиоды, соединив джамперами выводы OCxx (вывод ШИМ генераторов) на светодиоды:

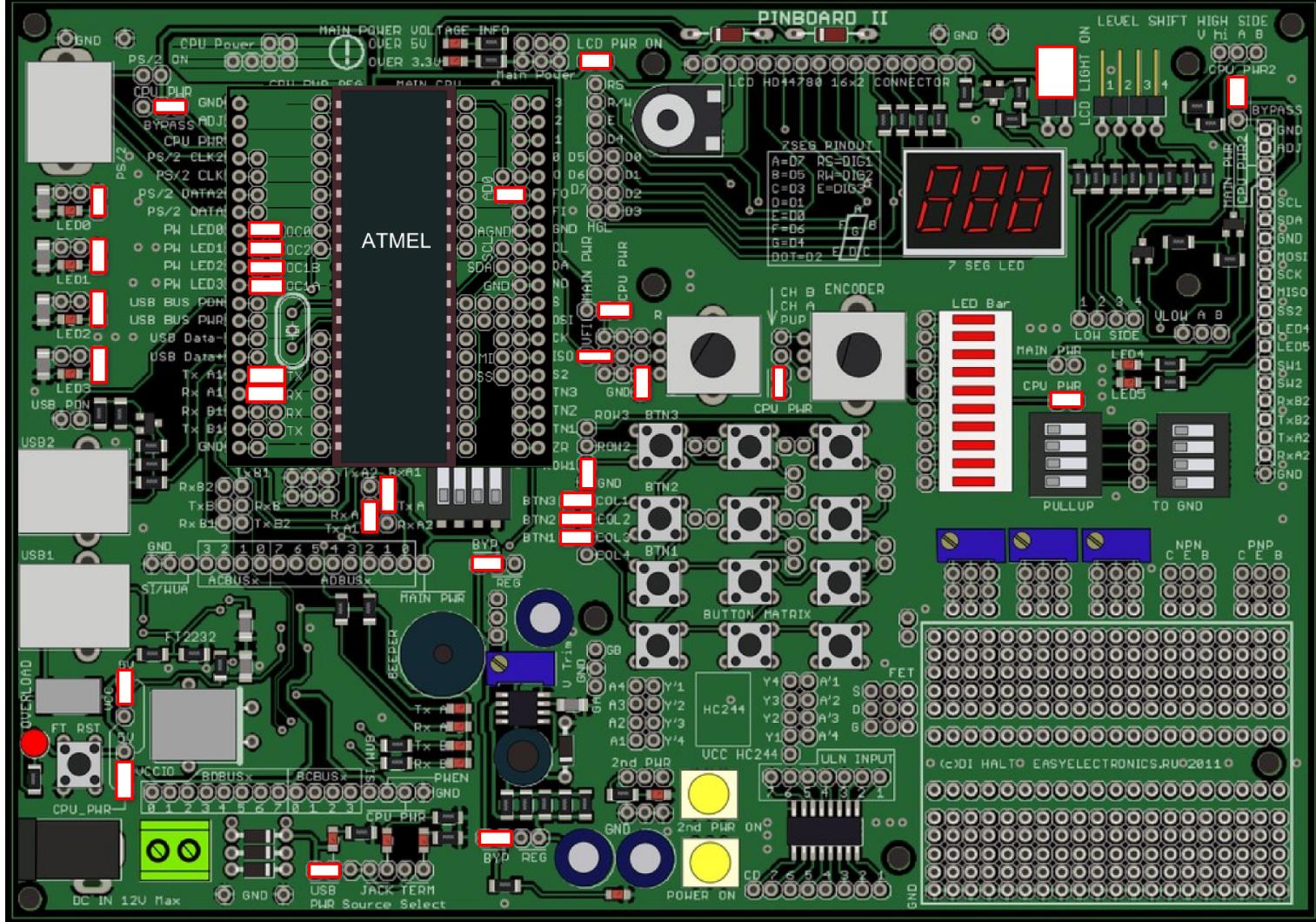


А на самой плате надо подключить джамперы, чтобы к линиями OCxx были подключены именно светодиоды, а не RC цепочки. Заодно не помешает подключить АЦП – правый джампер на процессорном блоке. И три джампера возле крутилки переменного резистора. Ну и, раз уж взялись за джамперы,

подготовить кнопки. Четыре джампера в правом нижнем углу. При этом мы кнопки BTN1..3 одним концом подключаем к земле (ROW1 на землю, вертикальный джампер), а колонки, отвечающие за кнопки, к выводам BTN1..3.



Общая карта джамперов для моего случая (Питание от USB, везде 5 вольт, поэтому все стабилизаторы закорочены перемычками BYPASS) выглядит так:



Еще потребуется спаять семижильный кабель. Он пригодится для подключения дисплея. Вот такой:



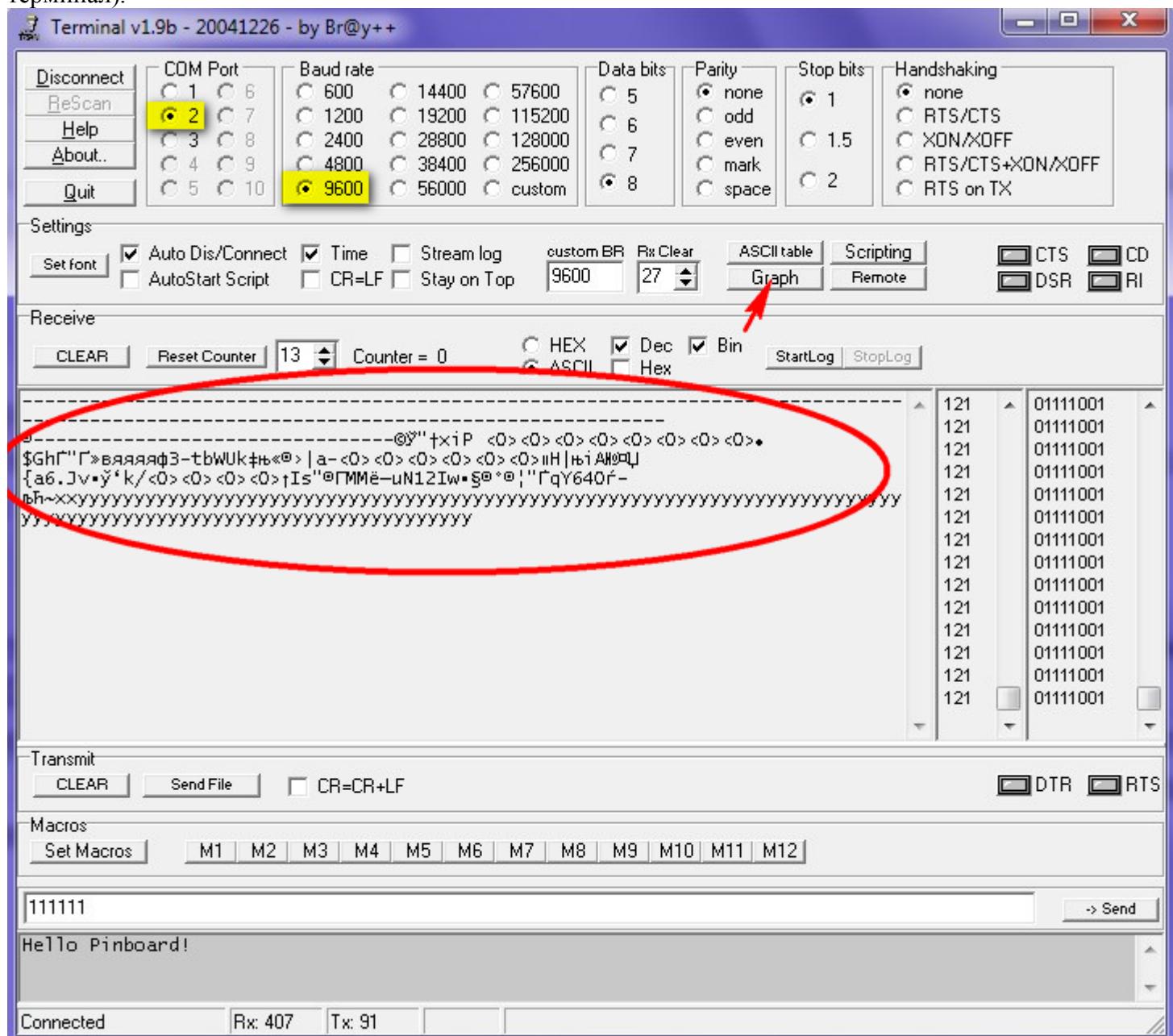
Четыре линии на шину данных дисплея. Три на управление. И желательно еще иметь четыре одиночных проводка с напаянными на них контактами, ими мы подключим кнопки к контроллеру.

4. Запуск и наблюдение за демкой.

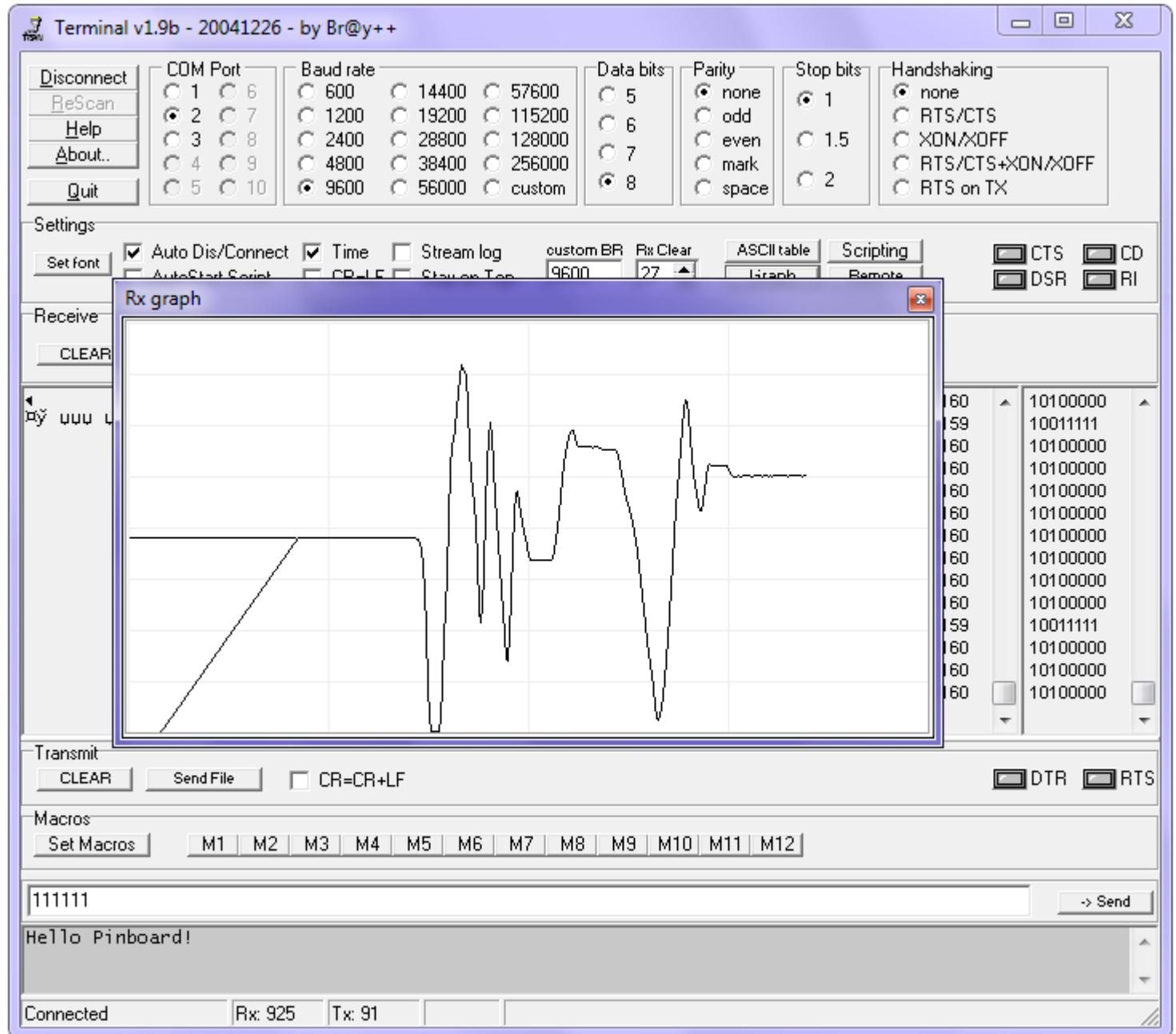
USB подключен, СОМ порты определились в системе. Джамперы все установлены. Включаем плату, кнопкой Power ON. Изначально, в ходе проверки, контроллер AVR прошит загрузчиком и небольшой демопрограммкой.

При включении должен загореться LED3 -- это стартовал Bootloader и просигнализировал о своем запуске. Через примерно 3-5 секунд LED3 погаснет и начнет выполняться основная программа.

В данном случае, заранее прошитая демка. Она ничего особо полезного не делает, только берет данные с АЦП и отправляет их в UART. При этом, если запустить программу Terminal и выбрать скорость 9600 и наш COM порт, настроенный на канал A, то увидим бегущие значения. При вращении ручки переменного резистора выдаваемые значения будут меняться от 0 до 255 (и также будет меняться символы выходящие в терминал).



А при нажатии кнопки Graph можно будет увидеть зависимость кривой от угла вращения:

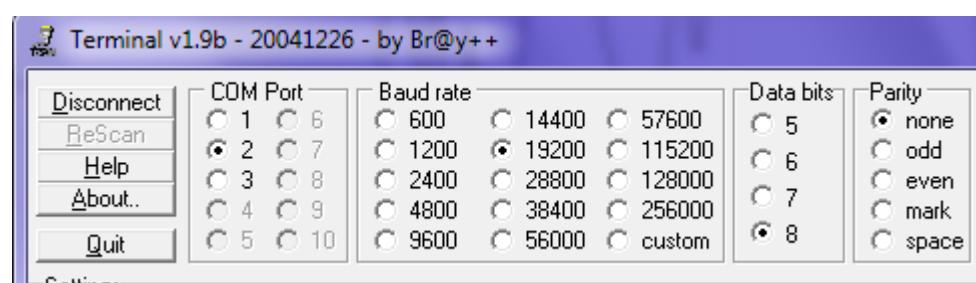


Если все получилось, значит микроконтроллер запущен, интерфейс работает и можно попробовать что-либо залить в него с помощью бутлоадера.

5. Активация бутлоадера

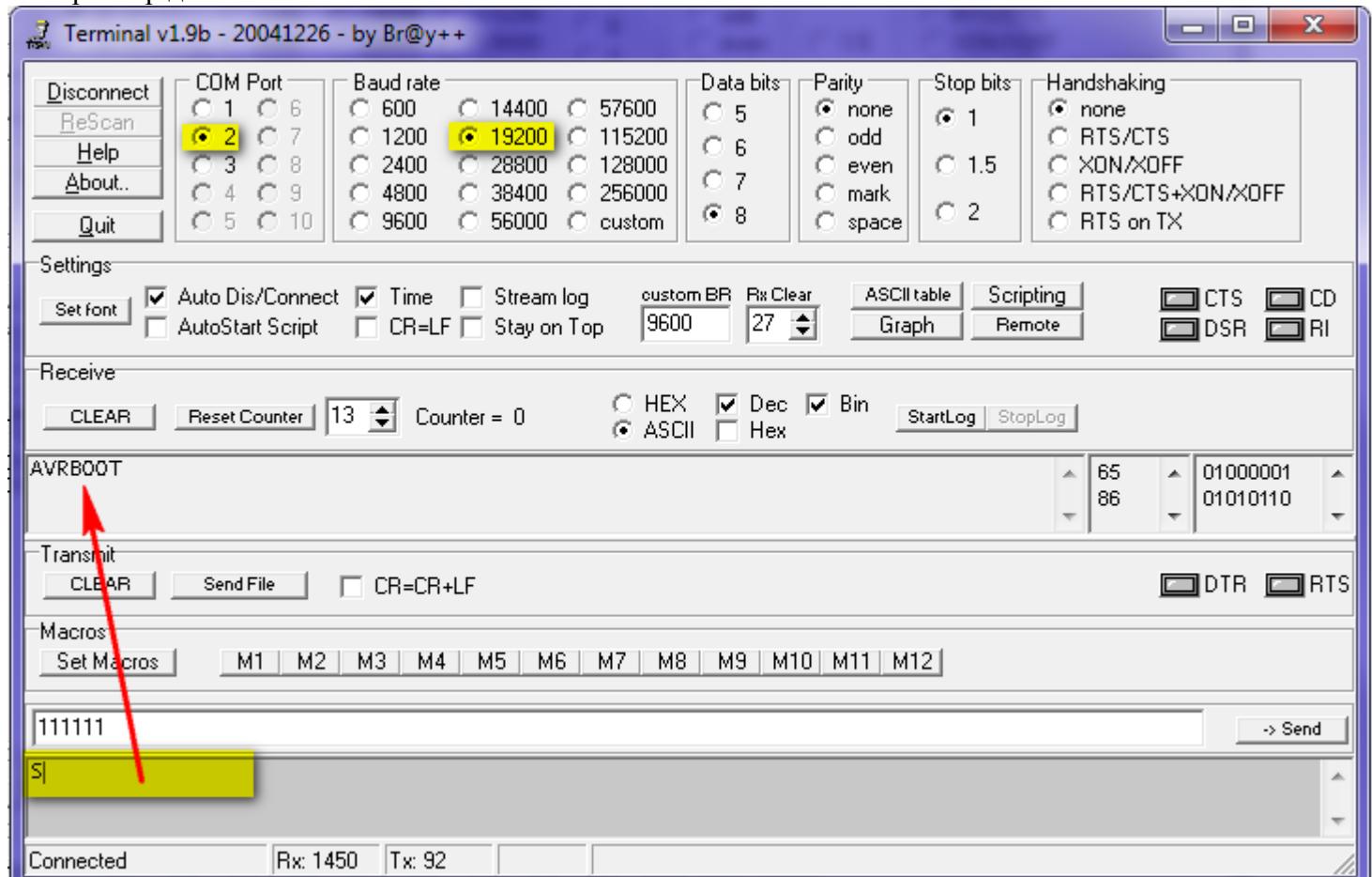
Проверим, вообще есть ли у нас бутлоадер и как он отвечает. Для этого

- Переключаем скорость в терминалке на 19200. И подключаемся к нашему порту отвечающему за канал А (в моем случае это COM2). Т.е. туда же куда и в прошлый раз.



- Нажимаем кнопку сброса на микроконтроллере. Она находится с торца процессорного модуля, под платой.

- И пока горит светодиод LED3 надо успеть послать в терминалку букву S. На скорости 19200. Контроллер должен ответить AVRBOOT.

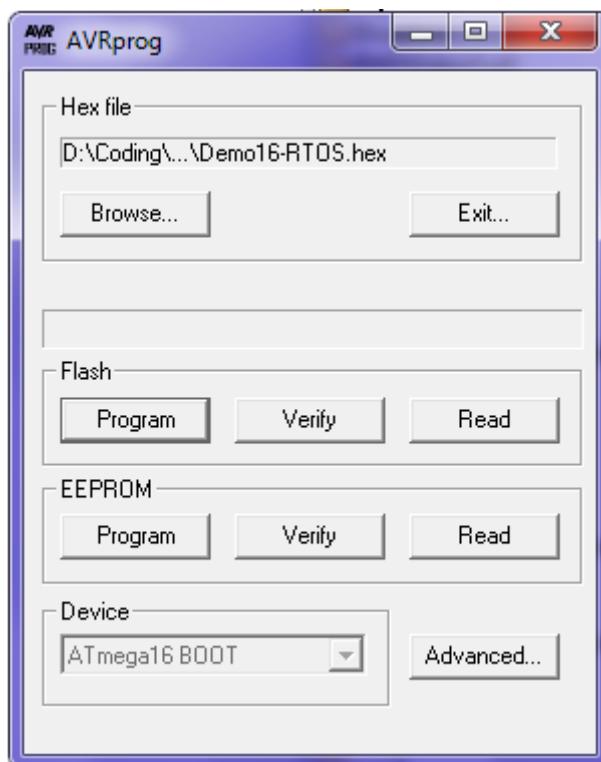


Ответ AVRBOOT означает, что бутлоадер на месте и готов принимать команды. Раз он ответил нам, то должен и ответить программе AVRProg.

- **Нажимаем Disconnect в терминальной программе и закрываем программу Terminal!**
- Опять жмем RESET и на этот раз запускаем AVRProg. Делать все нужно быстро. Не успеете запустить AVRProg пока горит диод – она выдаст, что No Supported board.

Учтите, что программе надо вначале запуститься, потом просканировать СОМ порты и попытаться из них получить по отклику. Так что на медленных или сильно загаженных левым софтом компьютерах AVRProg может не успеть запуститься пока работает бут. В этом случае сначала тыкаем запуск AVRProg, а потом RESET контроллера.

Если все хорошо, то вы увидите такую картину:



Тут все просто. В графе **Hex file** мы выбираем какой файл прошивки заливать в контроллер. Сгодится любой hex написанный под наш контроллер (в частности Mega16). Можете попробовать позагружать скомпилированные программы из учебного курса AVR. Лишь бы она занимала не больше чем 16кб-512б памяти флеша (для контроллера ATMega16). 512байт занимает загрузчик. На будущее -- я себе AVRProg забил на хоткеи внутри системы, это обычный экзешник, лежит в дебрях AVR Studio.

В блоке Flash – находится кнопки записи выбранного hex файла в флеш. А также верификации (чтение и сравнение) и просто чтения.

Блок EEPROM отвечает за загрузку hex файла в EEPROM память. Разумеется выбирать для этого надо будет hex сгенерированный для EEPROM (компилятор сам его генерит когда подключаешь в исходниках EEPROM секцию)

Под кнопкой Advanced прячутся кое какие доп настройки, вроде Lock битов. FUSE биты там тоже есть, но бутлоадер их не дает изменять. Так что не бойтесь, заблокировать контроллер через бутлоадер у вас не получится при всем желании.

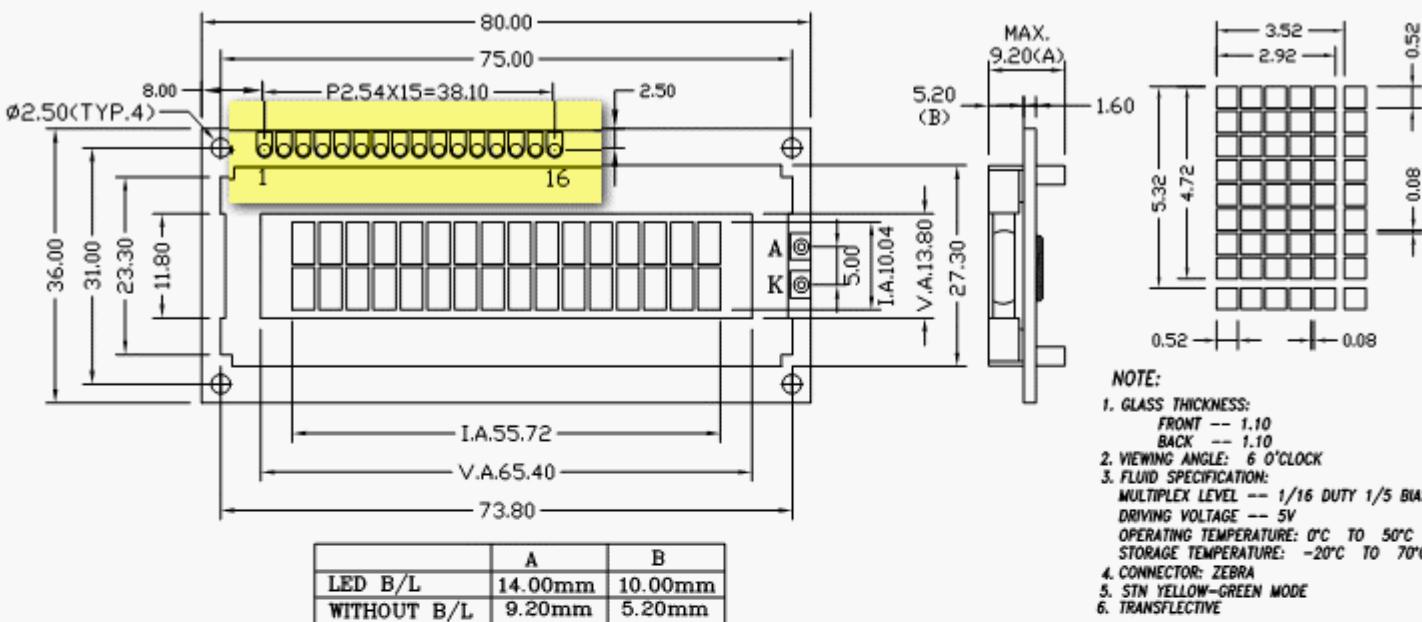
- Следующим шагом предлагаю залить мою расширенную демопрограммку. Выбирайте файл **Demo16-RTOS.hex** и жмите кнопку Program в группе Flash программы AVRProg. Пробежит прогресс бар и программа зальется в память.
- Нажмите Exit в AVRProg и закройте программу. Все, контроллер прошит и начал выполнять залитую в него **Demo16-RTOS**

6. Подключение дисплея и кнопок

Для полноценной работы демки нужно подключить к контроллеру дисплей, кнопки, а также пищалку.

Начнем с дисплея. В демоплате используется стандартный и очень популярный дисплей WH1602B на стандартном контроллере HD44780. Он использует параллельную шину данных и три линии управления. Шина данных может работать как в режиме 8ми бит, так и в режиме 4x бит. Обычно используется режим 4x бит, он может и сложней в реализации, зато требует меньше выводов.

Цоколевка у дисплея проста:



② MECHANICAL DATA

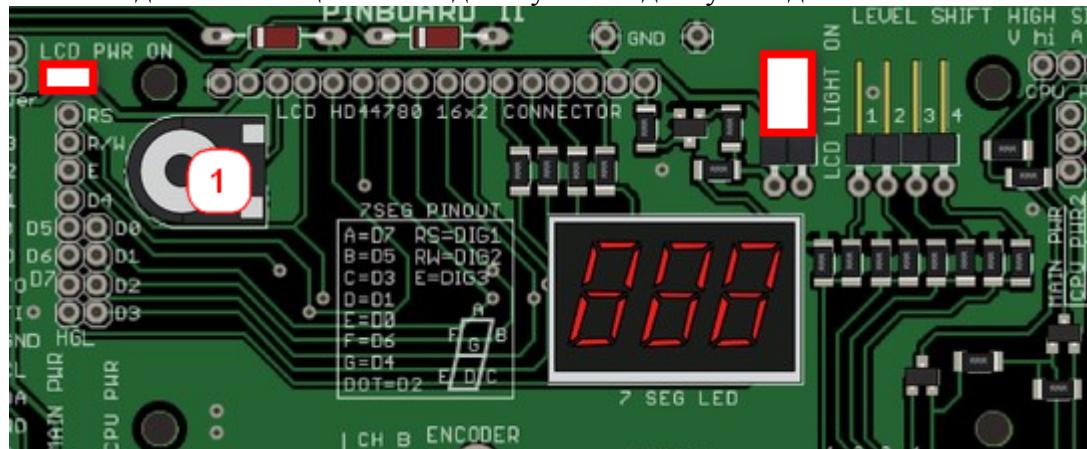
ITEM	SPECIFICATION	UNIT
Module Size (W X H X T)	80.0 x 36.0 x 9.2	mm
Viewing Area (W X H)	65.4 x 13.8	mm
Number of Character	16 Character x 2 Line	Char
Dot Pitch (W X H)	0.6 x 0.6	mm
Dot Size (W X H)	0.52 x 0.52	mm
Character Pitch (W X H)	3.52 x 5.32	mm
Character Size (W X H)	2.92 x 4.72	mm

④ BLOCK DIAGRAM

③ PIN CONFIGURATION

ITEM	SYMBOL	LEVEL	DESCRIPTION
1	VSS	L	Power GND
2	VDD	H	Power Positive
3	VEE	L	Voltage For LCD Panel Drive
4	RS	H/L	Register Select
5	R/W	H/L	ReadWrite Select
6	E	H/L	ReadWrite Enable
7	DB0	H/L	Data Bus
8	DB1	H/L	
9	DB2	H/L	
10	DB3	H/L	
11	DB4	H/L	
12	DB5	H/L	
13	DB6	H/L	
14	DB7	H/L	
15	A	H	LED BACKLIGHT +
16	K	L	LED BACKLIGHT -

Все выводы его вытащены на отдельную колодочку и подписаны:



D0...D7 – линии данных, в вырезке из документации зовутся DB0...DB7
 RS,R/W,E – линии управления.

Выход HGL управляет подсветкой. Т.е. подача на него логической 1 включает подсветку. Впрочем, подсветку можно включить и на постоянную работу, для этого надо одеть джампер LCD LIGHT ON (на картинке одет, он плашмя) расположенный.

Также необходимо одеть джампер питания дисплея LCD PWR ON иначе дисплей работать не будет.

ВАЖНО!!!

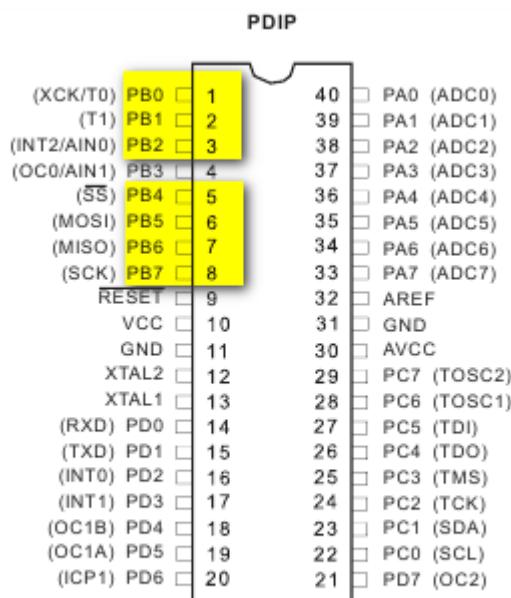
Питание дисплея не должно быть выше 5 вольт!!! Если у вас используется в шине MAIN POWER напряжение выше 5 вольт, то дисплей нужно снимать либо убирать джампер LCD POWER ON

Цифровой, семи сегментный, трехразрядный индикатор использует ту же шину для управления что и LCD и не может (не, при желании можно, но это будет просто форменный изврат) использоваться одновременно с ним. Потому и расположен под LCD дисплеем.

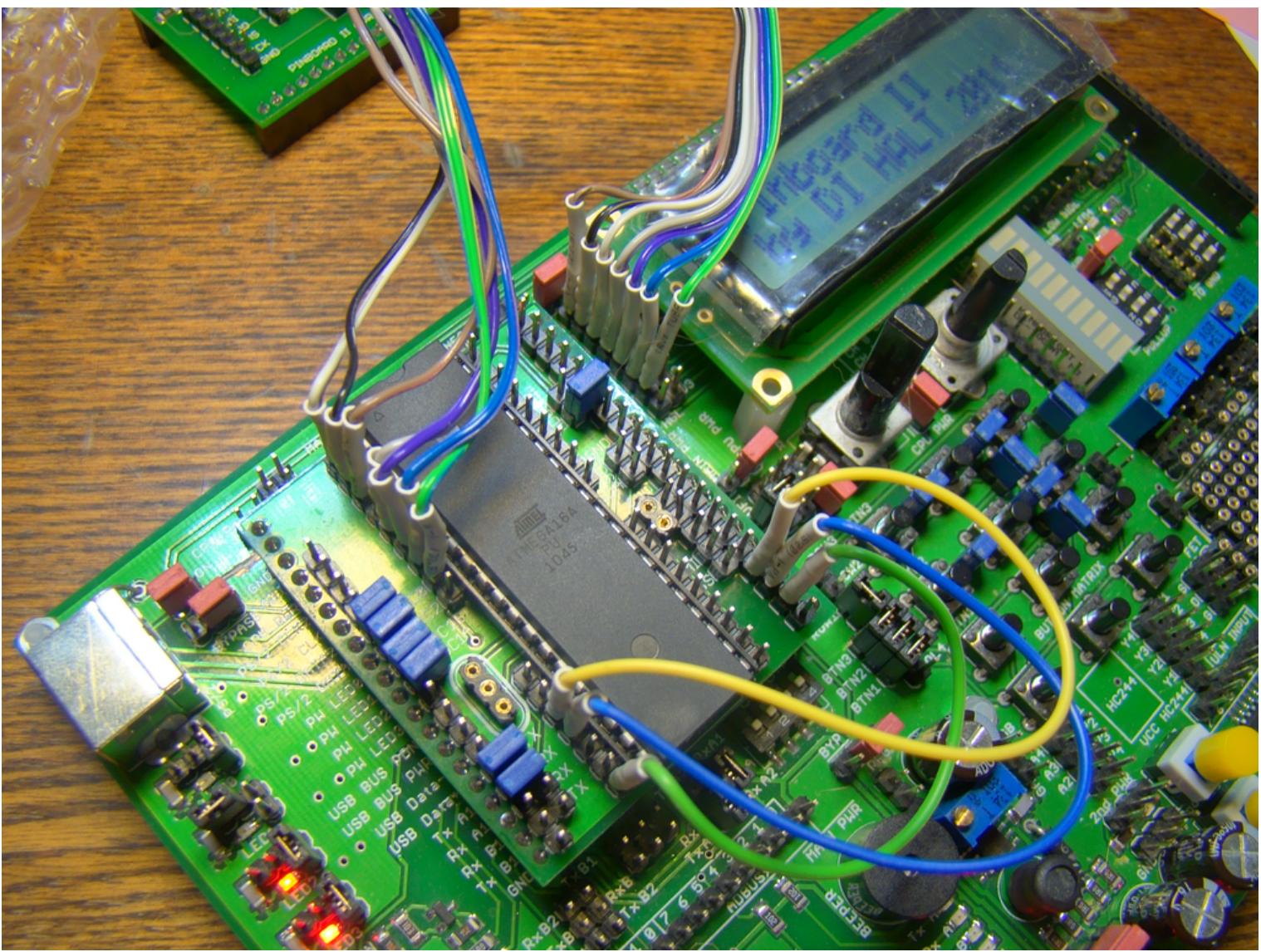
Соединяем выводы контроллера с колодкой дисплея. Для этого нам потребуется семижильный провод.

- PB0 - E
- PB1 - RW
- PB2 - RS
- PB4 - D4
- PB5 - D5
- PB6 - D6
- PB7 - D7

Выводы портов микроконтроллера AVR ATMega16 можно подглядеть в даташите. Вот из него вырезка:



Эти выводы соединяем нашим проводом с выводами дисплея, чтобы получилось как на фото:



Тремя проводками подключаем кнопки.

7. Тесты расширенной демопрограммы.

После запуска программы на дисплее зажжется надпись Pinboard II by DI HALT 2011. При нажатии кнопок будет выводиться что то вида «нажата кнопка – 2». Если поворачивать рукоятку АЦП, то будет выводиться значение с АЦП от 0 до 255 с небольшим округлением. Это же значение будет высылаться в терминал. А входящие символы, посылаемые в терминал с компа будут отображаться на дисплее демоплаты.

Также два светодиода LED2 и LED3 будут мигать, плавно меняя яркость от максимума к минимуму и обратно.

Вот ссылка на видео работы демки:
<http://youtu.be/4Bouy4pbVYk>

Небольшой FAQ

В: Почему подключаемся именно к этим портам? Это какие то специализированные порты?
О: Нет, просто я так захотел и мне так было удобно. Прелесть платы в том, что тут нет четкой привязки что к чему подключать. Можешь взять и перекинуть дисплей на порт A, например. Но разумеется надо будет внести изменения в программу.

В: А почему все проводками? Это что их каждый раз делать? Почему нельзя было сразу плату развести чтобы все было уже подключено.

О: Дело в том, что я хотел сделать полностью универсальную плату, без каких либо привязок на аппаратном уровне. Сделать готовые соединения всего со всем не представляется возможным. Поэтому весь основной межблочный монтаж делается навесной лапшой. Однако в этом нет ничего страшного, соединений требуется все же не так много, вряд ли тебе потребуется больше 20-30 жил провода, а это работы на пол часа-час по заготовке соединительной лапши. Да и монтажно-паяльный опыт никому не будет лишним.

В: Все подключил правильно, но на дисплее так ничего и не зажглось.

О: Скорей всего стоит покрутить ручку контраста яркости дисплея.

В: А что с Fuse битами делать? Я слышал их надо зашивать программатором.

О: Сейчас прошивка ведется через бутлоадер. Им нельзя изменить Fuse биты. Если же так сильно хочется, то придется подключить к плате 2FTBB программатор и прошить контроллер традиционным способом.

В: Я прошил плату обычным ISP программатором, а потом попытался прошить через загрузчик, но у меня ничего не вышло.

О: При прошивке через ISP если не принять дополнительных мер (включение загрузчика в прошивку) то загрузчик затирается и его надо залить заново. Файл загрузчика в архиве документации.