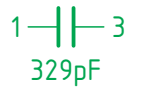
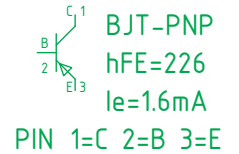
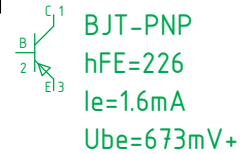
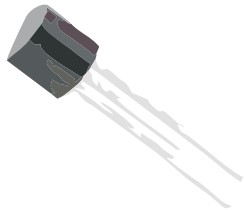


Описание Тестер ЭРЭ с AVR микроконтроллером и минимумом дополнительных элементов



версия 1.4-м

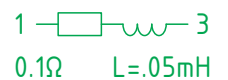


Markus Reschke

© 2012

madires@theca-tabellaria.de

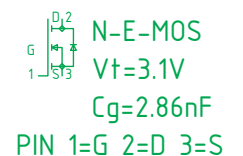
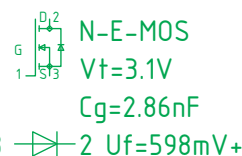
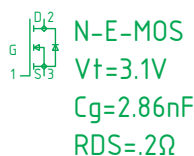
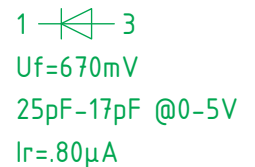
русский перевод
by indman



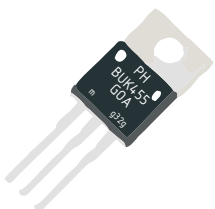
Составлено by bm-magic

Переведено с помощью www.DeepL.com/Translator (бесплатная версия)

6 августа 2021 г.



13.05.2019/MOR



Оглавление

1	Описание	7
1.1	О тестере	7
1.2	Рекомендации по безопасности	7
1.3	Лицензия	7
1.3.1	Дополнительная правовая оговорка	7
1.4	Отличия версий ПО	8
1.5	Исходный код	8
1.6	Поддерживаемое оборудование	9
1.7	Поддерживаемые варианты оборудования	9
1.7.1	пользовательский интерфейс	9
1.7.2	дополнительные возможности	9
1.7.3	дополнительные тесты и измерения	9
1.8	Создание прошивки	9
2	Оборудование	10
2.1	Аппаратные опции	10
2.2	Опции программного обеспечения	10
2.3	Доступные языки интерфейса	11
2.4	Конфигурация	11
2.5	Шины и интерфейсы	11
2.5.1	I2C/SPI	11
2.5.2	Последовательный TTL(RS-232)	12
2.5.3	1-Wire шина	12
2.6	Дисплеи	12
2.6.1	HD44780	13
2.6.2	ILI9163	14
2.6.3	ILI9341/ILI9342	14
2.6.4	ILI9481	15
2.6.5	ILI9486	15
2.6.6	ILI9488	15
2.6.7	PCD8544	16
2.6.8	PCF8814	16
2.6.9	SH1106	16
2.6.10	SSD1306	17
2.6.11	ST7036	17
2.6.12	ST7565R	18
2.6.13	ST7735	18
2.6.14	ST7920	18
2.6.15	STE2007/HX1230	19
2.6.16	VT100 терминал	19
2.7	Кнопки тестирования и управления	19
2.7.1	Кнопка Тест	19
2.7.2	Поворотный энкодер (аппаратная опция)	19

2.7.3	Кнопки управления +/- (аппаратная опция)	20
2.7.4	Сенсорный экран (аппаратная опция)	20
2.7.5	Пользовательский интерфейс	20
2.7.6	Соединение с компьютером	21
2.7.7	Последовательный выход	21
2.7.8	Автоматизация	21
2.7.9	VT100	21
3	Включение	22
3.0.1	Включить	22
3.0.2	Тестирование	22
3.0.3	Контроль источника питания	22
3.0.4	Выключение	23
3.0.5	Меню	23
3.1	Меню	24
3.1.1	ШИМ генератор	24
3.1.2	Простой интерфейс генератора ШИМ	24
3.1.3	Альтернативный интерфейс генератора ШИМ	24
3.1.4	Генератор прямоугольного сигнала	24
3.1.5	Тестирование стабилитронов (аппаратная опция)	24
3.1.6	Внутрисхемное измерение ESR	25
3.1.7	Проверка конденсатора на ток утечки	25
3.1.8	R/C/L - монитор	25
3.1.9	LC-измеритель(аппаратная опция)	25
3.1.10	Частотомер (аппаратная опция)	26
3.1.11	Базовый вариант	26
3.1.12	Расширенный вариант	26
3.1.13	Счетчик событий (аппаратная опция)	26
3.1.14	Тест поворотного энкодера	27
3.1.15	Контрастность дисплея	27
3.1.16	Приёмник ИК RC/Декодер	27
3.1.17	ИК RC Передатчик	29
3.1.18	Тестирование оптронов	30
3.1.19	Тестирование сервоприводов RC	30
3.1.20	Сканирование 1-Wire шины	31
3.1.21	Температурный датчик DS18B20	31
3.1.22	Датчики температуры и влажности DHTxx	32
3.1.23	Самотестирование	32
3.1.24	Автокорректировка	32
3.1.25	Сохранение/Загрузка	33
3.1.26	Информация о корректировке	34
3.1.27	Шрифт	34
3.1.28	Выключение	34
3.1.29	Выход	34
4	Детали измерения	35
4.1	Резисторы	35
4.2	Конденсаторы	35
4.3	Индуктивности	36
4.4	Разряд компонентов	36
4.5	ADC супердискретизация	36
4.6	Отображение результатов на дисплее	37
4.7	Дополнительные подсказки	37
4.8	Полезные ресурсы	39
4.9	Журнал изменений	39

5	Дистанционный контроль	40
5.1	Команды управления	40
5.1.1	ERR	40
5.1.2	OK	40
5.1.3	N/A	40
5.2	Основные команды:	40
5.2.1	VER	40
5.2.2	OFF	40
5.3	Команды тестирования:	40
5.3.1	PROBE	40
5.3.2	COMP	40
5.3.3	MSG	40
5.3.4	QTY	40
5.3.5	NEXT	40
5.3.6	TYPE	41
5.3.7	HINT	41
5.3.8	MHINT	41
5.3.9	PIN	41
5.3.10	R	42
5.3.11	C	42
5.3.12	L	42
5.3.13	ESR	42
5.3.14	I_l	42
5.3.15	V_F	42
5.3.16	V_F2	42
5.3.17	C_D	42
5.3.18	I_R	42
5.3.19	R_BE	42
5.3.20	h_FE	42
5.3.21	h_FE_r	42
5.3.22	I_C	42
5.3.23	I_E	43
5.3.24	V_BE	43
5.3.25	I_CEO	43
5.3.26	V_th	43
5.3.27	C_GS	43
5.3.28	R_DS	43
5.3.29	V_GS_off	43
5.3.30	I_DSS	43
5.3.31	C_GE	43
5.3.32	V_GT	43
5.3.33	V_T	43
5.3.34	R_BB	43
5.3.35	V_Z	43
6	Файл Code Makefile	44
6.1	Makefile	44
6.1.1	Тип MCU	44
6.1.2	Тактовая частота MCU	44
6.1.3	Тип осциллятора	44
6.1.4	Тип MCU Avrdude	45
6.1.5	Avrdude ISP программист	45
6.2	конфигурация config.h	46
6.2.1	Эксплуатация оборудования	46
6.2.2	Возможности программного обеспечения	48

6.2.3	Пользовательский интерфейс	51
6.2.4	Управление питанием	52
6.2.5	Настройки измерения и смещения	53
6.2.6	F & E - предназначен для разработчиков микропрограммного обеспечения	55
6.2.7	Шина	56
6.3	конфигурация Config_328.h	57
6.3.1	Модуль дисплея	57
6.3.2	Назначение портов и выводов	57
6.3.3	Шина	59
6.4	конфигурация Config_644.h	60
6.4.1	Модуль дисплея	60
6.4.2	Назначение портов и выводов	60
6.4.3	Шина	62
6.5	конфигурация Config_1280.h	63
6.5.1	Модуль дисплея	63
6.5.2	Назначение портов и выводов	63
6.5.3	Шина	65
7	Сбор настроек	66
7.1	Arduino Nano, Uno или Mega 2560	66
7.2	DIY Kit „AY-AT“ также GM328A	66
7.3	BSide ESR02 (DTU-1701)	68
7.4	Fish8840 TFT	69
7.5	GM328 !не GM328A!	69
7.6	Hiland M644	70
7.7	M12864 DIY Тестер транзисторов	71
7.8	МК-328	72
7.9	T3/T4	72
7.10	Многофункциональный тестер TC-1 и семейство (T7)	74
7.11	Разъем Arduino MEGA	76
8	Программирование тестера компонентов	77
8.1	Настройка тестера компонентов	77
8.2	программирование микроконтроллера	77
8.3	Операционная система Linux	77
8.4	Использование в Linux	78
8.5	установка пакетов программ	78
8.6	скачать источники	78
8.7	Использование интерфейсов	78
8.8	Членство в группе	79
8.9	рабочая среда	79
8.10	создание и передача прошивки	80
9	Предварительная версия	81
9.1	v1.44m 2021-08	81
9.2	v1.43m 2021-03	81
9.3	v1.42m 2020-12	81
9.4	v1.41m 2020-09	82
9.5	v1.40m 2020-06	83
9.6	v1.39m 2020-03	84
9.7	v1.38m 2019-12	84
9.8	v1.37m 2019-09	84
9.9	v1.36m 2019-05	85
9.10	v1.35m 2019-02	85

9.11	v1.34m	2018-10	86
9.12	v1.33m	2018-05	86
9.13	v1.32m	2018-02	86
9.14	v1.31m	2017-12	86
9.15	v1.30m	2017-10	87
9.16	v1.29m	2017-07	87
9.17	v1.28m	2017-04	87
9.18	v1.27m	2017-02	87
9.19	v1.26m	2016-12	88
9.20	v1.25m	2016-09	88
9.21	v1.24m	2016-08	88
9.22	v1.23m	2016-07	88
9.23	v1.22m	2016-03	89
9.24	v1.21m	2016-01	89
9.25	v1.20m	2015-12	89
9.26	v1.19m	2015-11	89
9.27	v1.18m	2015-07	89
9.28	v1.17m	2015-02	90
9.29	v1.16m	2014-09	90
9.30	v1.15m	2014-09	90
9.31	v1.14m	2014-08	90
9.32	v1.13m	2014-07	90
9.33	v1.12m	2014-03	90
9.34	v1.11m	2014-03	91
9.35	v1.10m	2013-10	91
9.36	v1.09m	2013-07	91
9.37	v1.08m	2013-07	91
9.38	v1.07m	2013-06	92
9.39	v1.06m	2013-03	92
9.40	v1.05m	2012-11	92
9.41	v1.04m	2012-11	92
9.42	v1.03m	2012-11	92
9.43	v1.02m	2012-11	92
9.44	v1.01m	2012-10	93
9.45	v1.00m	2012-09	93
9.46	v0.99m	2012-09	93

1.1. О тестере

Тестер компонентов создан на основе проекта Markus Frejek [1] и [2] и его преемника Karl-Heinz Kübbeler [3] и [4]. Это альтернативное программное обеспечение для текущей схемы Karl-Heinz с некоторыми изменениями в пользовательском интерфейсе и методами, используемыми для тестирования и измерения. Karl-Heinz выпускает официальные релизы, поддерживающие также более старые процессоры ATmega, данная же версия работает на процессорах ATmega с объемом 32кБ памяти и более.

Подсказка: запускайте режим автокорректировки при первом запуске, а так же, если вы сделали какую-либо модификацию, обновление прошивки или применили другие тестовые контакты (зажимы).

1.2. Рекомендации по безопасности

Тестер компонентов - **это не профессиональный цифровой измеритель!**

Это простой тестер для измерения некоторых деталей. Используйте его для демонтированных электронных компонентов. Тестовые контакты не имеют защиты и не допускают работу на напряжениях выше, чем 5В.

Не используйте тестер для цепей под высоким напряжением!

Если вы тестируете конденсатор, обязательно разрядите его прежде, чем подключить! Это не просто безопасность, ваша жизнь может находиться в опасности, если вы соедините тестовые контакты с цепью под напряжением или источником питания, особенно в электросети!

1.3. Лицензия

Автор исходного кода не предоставил информации о лицензии, которая ограничивает использование микропрограммного обеспечения. Он утверждал, что это открытый источник и любой коммерческий пользователь должен связаться с ним. К сожалению, мы (Karl-Heinz и я) не нашли способа связаться с ним. Поэтому я выбрал лицензию с открытым исходным кодом 01.01.2016.

Автор может связаться с нами, чтобы изъяснить свои пожелания по этому вопросу. Так как исходный код этой прошивки был обработан и переписан с массой новых функций, я думаю, что такой подход оправдан. Лицензия зарегистрирована под EUPL V.1.1.

1.3.1. Дополнительная правовая оговорка

Название продукта или название компаний – возможно торговые марки соответствующих владельцев.

1.4. Отличия версий ПО

Karl-Heinz написал действительно полную документацию на тестер [16], очень рекомендую прочитать её. Я объясню вам существенные отличия от его версии:

- пользовательский интерфейс
 - + никаких проблем! ;-)
 - + пользовательский интерфейс, никаких проблем
 - + Автоматизация (команды дистанционного управления)
 - функция разрядки подключенного компонента - функция разрядки подключенного компонента
 - + специальный метод тестирования сопротивлений менее 10Ω (вместо метода проверки ESR)
 - измерение сопротивления
 - + измерение ёмкости от 5pF ;
 - + дополнительный метод тестирования для конденсаторов от $4.7\mu\text{F}$ до $47\mu\text{F}$
 - + метод корректировки/компенсации
 - Выборка АЦП для измерений очень малой емкости или индуктивности не использовался
 - диоды
 - + автоопределение
 - биполярные плоскостные транзисторы (BJTs)
 - + V_f определяется для более подходящего (виртуального) I_b на основе hFE
 - + обнаружение германиевых биполярных плоскостных транзисторов с высоким током утечки (JFETs)
 - + обнаружение ВJT с барьером Шоттки
 - обнаружение JFETs с очень низким I_{DSS}
 - тиристоры (симисторы)
 - + обнаружение MT1 и MT2
 - приёмник ИК RC и декодер
 - ИК RC передатчик
 - проверка оптронов
 - тестирование сервоприводов RC
 - работа схемы возможна на тактовой частоте процессора 20 МГц
 - 1-Wire (DS18B20)
 - DНxx датчики
 - счетчик импульсов (событий)
 - структурированный исходный код
 - + ещё изменения, о которых я не могу вспомнить сейчас.
- Более подробное описание функций в разделах ниже.

1.5. Исходный код

Начальная прошивка m-версии была основана на исходном коде Карла-Хайнца. Далее была сделана глубокая оптимизация, добавлено больше комментариев и переменных, реструктурированы функции. Большие функции разделены на несколько меньших частей. Вскоре моя прошивка стала независимой версией. Например, были добавлены простые каркасы для поддержки различных дисплеев и интерфейсных шин. Я надеюсь, что сейчас код легко читается и поддерживается. Вы можете загрузить актуальное программное обеспечение с этих сайтов:

[5] <https://github.com/madires/Transistortester-Warehouse>

[6] <https://github.com/kubi48/TransistorTester-source/tree/master/Markus>

[7] <https://github.com/Mikrocontroller-net/transistortester/tree/master/Software/Markus>

1.6. Поддерживаемое оборудование

Прошивка работает на всех устройствах, которые совместимы со стандартной схемой, размещенной в документации Карла-Хайнца и которые используют один из следующих MCU:

- ATmega 328 config_328.h
- ATmega 324/644/1284 config_644.h
- ATmega 640/1280/2560 config_1280.h

Вы можете настроить назначение портов процессора при необходимости. Дисплей может быть символьным или графическим (монохромный или цветной). Пожалуйста, посмотрите раздел 2.6 «Дисплей», со страницы 12.

1.7. Поддерживаемые варианты оборудования

1.7.1. пользовательский интерфейс

- поворотный энкодер
- дополнительные кнопки (увеличение/уменьшение)
- сенсорный экран
- последовательный интерфейс (TTL, RS232, USB-посл. адаптер)

1.7.2. дополнительные возможности

- внешний 2.5-вольтовый источник опорного напряжения
- аппаратная калибровочная емкость
- защитное реле для разрядки конденсаторов

1.7.3. дополнительные тесты и измерения

- проверка стабилитронов / измерение внешнего напряжения <50В
- простой частотомер
- расширенный вариант частотомера с прескалером и кварцевыми генераторами для низких и высоких частот
- аппаратный ИК приёмник.

1.8. Создание прошивки

Прежде всего, вы должны отредактировать Makefile, см. главу 6.1 со страницы 44 здесь и далее. Модель MCU, частота, тип генератора и настройки адаптера программирования.

- В конфигурационном **config.h** см. главу 6.2 на стр.46, где устанавливаются опции управления и меню. Здесь Вы можете выбрать аппаратное и программное обеспечение, язык для работы и т.д.

- Наконец, в **config_<MCU>.h** глобальная конфигурация, как отображение и назначение пин-выводов, которые различаются в зависимости от используемого MCU. Подробнее см. в главе ?? на стр.??.

Применяются следующие файлы config<MCU>.h:

- ATmega 328 config_328.h
- ATmega 324/644/1284 config_644.h
- ATmega 640/1280/2560 config_1280.h

- В главе 7 «Клоны», начиная со страницы 66, вы найдете настройки для различных версий тестера или клонов. Если ваш тестер отсутствует в списке, пожалуйста, отправьте настройки по электронной почте автору [8], чтобы помочь другим пользователям.

Все настройки и значения объясняются в самих файлах.

Как скомпилировать прошивку вы увидите в разделе 8.10 на странице 80.

2.1. Аппаратные опции

- поворотный энкодер или кнопки управления
- 2.5-вольтовый источник опорного напряжения
- защитное реле (разрядка конденсаторов)
- измерение напряжения стабилизации для стабилитронов
- частотомер (базовый и расширенный варианты)
- счетчик событий
- LC-измеритель
- приёмник/декодер ИК для пультов дистанционного управления (аппаратный модуль приемника ИК)
- поддержка аппаратного калибровочного конденсатора (только с АТМega644/1284) для корректировки смещений напряжения
- SPI шина (программная и аппаратная поддержка)
- I2C шина (программная и аппаратная поддержка)
- TTL последовательный интерфейс(аппаратная и программная поддержка)
- 1-Wire шина (программная)

Внешний 2.5-вольтовый источник опорного напряжения необходимо использовать только, если он в 10 раз более точный, чем стабилизатор напряжения. Иначе это приведёт к худшим результатам. Если вы используете микросхему MCP1702 с типичным допуском 0.4% как стабилизатор напряжения, внешний ИОН в схеме не нужен!

2.2. Опции программного обеспечения

- ШИМ генератор (2 варианта)
- измерение индуктивности
- измерение ESR и внутрисхемное измерение ESR
- проверка поворотных энкодеров
- генератор прямоугольного сигнала (требуется доп. кнопки)
- приёмник/декодер ИК для пультов дистанционного управления (модуль приемника ИК должен быть подключён к схеме)
- ИК RC передатчик (ИК светодиод с транзистором управления)
- проверка оптронов
- проверка сервоприводов (требуется доп. кнопки и дисплей на более чем 2 строки)
- обнаружение UJTs
- проверка тока утечки конденсаторов
- тест температурного датчика DS18B20
- цветное отображение тестовых контактов (для цветного графического дисплея)
- вывод информации о найденных компонентах через последовательный TTL(RS-232), например, на PC
- команды управления для автоматизации с использованием последовательного TTL
- измерение обратного hFE для BJTs
- DHT11/22 датчик температуры и влажности.
- ...

Выберите необходимые опции внимательно, чтобы они соответствовали вашим потребностям, а также ресурсам процессора, т.е. размеру RAM, EEPROM и флэш-памяти. Если скомпилированная прошивка превышает размер флэш-памяти процессора, попытайтесь отключить некоторые опции, в которых вы не нуждаетесь.

2.3. Доступные языки интерфейса

- Английский (по умолчанию)
- Немецкий
- Чешский - (поддержка Кара) - шрифт на базе кодировки ISO8859-1
- Чешский 2 - (поддержка Bohu) - шрифт на базе кодировки ISO8859-2
- Датский - (поддержка glenndk@mikrocontroller.net)
- Итальянский - (поддержка Gino_09@EEVblog)
- Польский - (поддержка Szpila)
- Румынский - (поддержка Dumidan@EEVblog)
- Испанский - (поддержка pepe10000@EEVblog)
- Датский язык - (поддержка glenndk@mikrocontroller.net)
- незначительные изменения в шрифте
- Русский - (поддержка indman@EEVblog)
- шрифт с поддержкой кириллицы на базе кодировки Windows-1251
- Русский 2 - (поддержка hapless@EEVblog)
- шрифт с поддержкой кириллицы на базе кодировки Windows-1251
,альтернативный вариант

Десятичная часть измерений обозначена точкой, но вы можете изменить её на запятую, если сделать соответствующую настройку.

2.4. Конфигурация

Ваш MCU установлен в файле Makefile ... глава 6.1 со страницы 44.

Желаемые опции в config.h ... глава 6.2 со страницы 46.

. Для специфических для MCU настроек, таких как назначение контактов и индикация, вы устанавливаете в зависимости от MCU в файле config_MCU:

- ATmega 328 config_328.h
- ATmega 324/644/1284 config_644.h
- ATmega 640/1280/2560 config_1280.h

по шаблону ... глава ?? со страницы ??.

After, редактирование, сборка прошивки производится в соответствии с разделом 8.10 со страницы 80.

. Если прошивка становится слишком большой, попробуйте отключить опции, которые не важны для вас.

2.5. Шины и интерфейсы

2.5.1. I2C/SPI Некоторым дисплеям и другим аппаратным средствам необходимы протоколы I2C или SPI для соединения с процессором. Поэтому прошивка включает драйверы для обоих протоколов. Программный и аппаратный режимы могут обеспечить назначения различных контактов для разных тестеров и поддержку драйверов шины. Программный режим может использовать любые выводы IO на одном и том же порту, в то время как аппаратный режим использует специальные контакты шины процессора. Недостаток программного режима - низкая скорость. Аппаратный режим намного быстрее. Вы можете легко определить различие в скорости при подключении жидкокристаллических цветных дисплеев. LCD модули с интерфейсом SPI могут управляться аппаратным SPI или с использованием программной эмуляции. Тестер на ATmega328 поддерживает только программную эмуляцию SPI из-за особенности схемы. ATmega324/644/1284 с большим количеством портов ввода-вывода позволяет использовать программную эмуляцию SPI а также намного более быстрый аппаратный SPI. Драйверы SPI или I2C включены в LCD модули и могут быть настроены в разделе конфигурации дисплея - config-<MCU>.h. Альтернативно, вы можете также включить I2C и SPI в config.h, настроить порты и конфигурацию в специальных разделах - config-<mcu>.h ??(смотрите I2C PORT или SPIPORT)56. Если вы выберете программный SPI и включите режим чтения (SPI_RW), убедитесь, что установлен также SPI_PIN и SPI_MISO. См. раздел SPI в config-<MCU>.h для примера. Страница 65.

2.5.2. Последовательный TTL(RS-232) Тестер может поддерживать вывод информации через последовательный интерфейс TTL. Если он используется для связи с РС, у вас должен быть также USB - TTL преобразователь или классический драйвер RS-232. Встроенное микропрограммное обеспечение использует аппаратный или программный UART. Последовательный интерфейс TTL включается и настраивается в файле config.h (смотрите секцию "Busses"), контакты порта определены в config <mcu>.h (смотрите SERIAL_PORT). Страница 65

У программного UART есть недостаток - режим TX не имеет постоянно высокий уровень, если он не активен. Это происходит из-за способа, которым управляется вывод порта МК. Чтобы исправить недостаток, управление портом должно быть изменено, что вызовет увеличение объема встроенного микропрограммного обеспечения. Но эта проблема не доставляет неприятностей с большинством USB-TTL последовательных преобразователей. Если проблема всё же есть, вы можете добавить подтягивающий резистор (10-100k) на выход TX, чтобы поддерживать сигнал на высоком уровне в режиме ожидания. Настройка по умолчанию для последовательного TTL 9600 8N1:

- 9600 бит/с, 8 бит данных, нечетный, 1 стоп бит, без контроля потока

2.5.3. 1-Wire шина Другая поддерживаемая шина – 1-Wire, которая может использовать любые щупы/тестовые контакты(ONEWIRE_PROBES) или специальный контакт ввода-вывода (ONEWIRE_IO_PIN) страница 56. Драйвер разработан для стандартной частоты шины и устройств с внешним питанием. Назначение контактов для подключения:

Тестовый контакт No1: Земля(Gnd)

Тестовый контакт No2: DQ (данные)

Тестовый контакт No3: Питание Vcc (ток ограничен резистором 680 Ω)

Требуется внешний резистор подтяжки 4.7 к Ω между контактами DQ и Vcc!

Для устройства, требующего подключения к шине только одного клиента, можно при желании узнать его код ПЗУ (ONEWIRE_READ_ROM). В случае CRC ошибки или если подключено несколько клиентов, на дисплее будет символ '?-?'. Если код ПЗУ равен нулю - есть проблема чтения. При удачном считывании первая часть кода ПЗУ показывает семейство устройства, вторая часть – его серийный номер.

Для устройства, требующего подключения к шине только одного клиента, можно при желании узнать его код ПЗУ (ONEWIRE_READ_ROM). В случае CRC ошибки или если подключено несколько клиентов, на дисплее будет символ '?-?'. Если код ПЗУ равен нулю - есть проблема чтения. При удачном считывании первая часть кода ПЗУ показывает семейство устройства, вторая часть – его серийный номер.

2.6. Дисплеи

На данный момент поддерживаются следующие контроллеры дисплея:

- HD44780	(символьный дисплей, 2-4 строки на 16-20 символов)	Стр.13
- ILI9163	(цветной графический дисплей 128x160)	Стр.14
- ILI9341/ILI9342	(цветной графический дисплей 240x320 или 320x240)	Стр.14
- ILI9481	(цветной графический дисплей 320x480, частично проверен)	Стр.15
- ILI9486	(цветной графический дисплей 320x480, частично проверен)	Стр.15
- ILI9488	(цветной графический дисплей 320x480, частично проверен)	Стр.15
- PCD8544	(графический дисплей 84x48)	Стр.16
- PCF8814	(графический дисплей 96x65)	Стр.16
- SH1106	(графический дисплей 128x64)	Стр.16
- SSD1306	(графический дисплей 128x64)	Стр.17
- ST7036	(символьный дисплей, 3 строки на 16 символов, не проверен)	Стр.17
- ST7565R	(графический дисплей 128x64)	Стр.18
- ST7735	(цветной графический дисплей 128x160)	Стр.18
- ST7920	(графический дисплей 128x64)	Стр.18
- STE2007/HX1230	(графический дисплей 96x68)	Стр.19
- VT100 терминал	(через последовательный интерфейс)	Стр.19

Заботьтесь о правильном напряжении питания LCD дисплея и допустимых логических

уров- нях на его шинах! Простой переключатель уровня с последовательно включенными резисто- рами, основанный на внутренних ограничительных диодах контроллера дисплея, может ра- ботать, но только на низкой скорости шины программного SPI. Поэтому я реко- мендую ис- пользовать соответствующий преобразователь уровня на микросхемах.

Если дисплей ничего не показывает, проверьте правильность подключения и попро- буйте различные настройки контрастности в файле(config<mcu>.h) со страницы ??.

Для большинства дис-плеев можно аппаратно подключить выводы /CS и /RES через под- тягивающие резисторы к GND/Vcc и закомментировать соответствующие настройки для портов IO, если присутствует только один дисплей на шине.

Большинство графических дисплеев имеют настройки для изменения ориентации изоб- ражения, например, для поворота изображения на 90° и зеркального отражения по гори- зонтали или вертикали. Таким образом, изображение можно настроить для каждого дис- плея по мере необходимости. Для цветных графических дисплеев доступны дополни- тельныенастройки. В стандартном цветовом режиме тестер использует разные цвета, кото- рые могут быть изменены путем редактирования файла colors.h . Если закомментировать LCD_COLOR, будет включен двухцветный режим и цвет шрифта будет COLOR_PEN, в то время как цвет фона - COLOR_BACKGROUND. В случае, если базовые цвета RGB красный и синий поме- нялись местами, включите LCD_BGR для реверса красного и си- него цветов. У некоторых дисплеев есть обратные субпиксели RGB и контроллер дисплея не знает об этом.

Подсказка для ATМega328: если вы подключаете поворотный энкодер к PD2/PD3, пожа- луйста, соедините вывод /CS модуля с PD5 и задействуйте настройку LCD_CS в config_328.h (применяется к графическим дисплеям). Иначе импульсы с поворотного энкоде- ра, вме- шавшись в шину данных, приведут к ошибкам в работе дисплея.

2.6.1. HD44780 использует 4-х битный режим. Назначение контактов.

Дисплей	config-<MCU>.h	328 75,6%	644 37,9%	1280 13,1%	По умолчанию
DB4	LCD_DB4	PD0	PB4	PB0	
DB5	LCD_DB5	PD1	PB5	PB1	
DB6	LCD_DB6	PD2	PB6	PB2	
DB7	LCD_DB7	PD3	PB7	PB3	
RS	LCD_RS	PD4	PB2	PB4	
R/W	LCD_RW	Gnd	Gnd	Gnd	
E	LCD_EN1	PD5	PB3	PB5	

Таблица 2.1. Пинаут для параллельной шины HD44780.

Можно также управлять данным ЖК-дисплеем через подключаемый I2C модуль PCF8574. Адрес I2C также должен быть настроен. Назначение контактов для дисплея с подключен- ным модулем PCF8574:

Дисплей	config-<MCU>.h	328 76,5%	644 38,2%	1280 9,8%	По умолчанию
DB4	LCD_DB4		PCF8574_P4		
DB5	LCD_DB5		PCF8574_P5		
DB6	LCD_DB6		PCF8574_P6		
DB7	LCD_DB7		PCF8574_P7		
RS	LCD_RS		PCF8574_P0		
R/W	LCD_RW		PCF8574_P1		
E	LCD_EN1		PCF8574_P2		
LED	LCD_LED		PCF8574_P3		

Таблица 2.2. Пинаут для ЖК-Дисплей с PCF8574.

2.6.2. ILI9163 использует 4-х проводной SPI режим. Назначение контактов:

Дисплей	config-<MCU>.h	328 87,4%	644 43,8%	1280 11,2%	По умолчанию
/RESX	LCD_RES	PD4	PB2	PB4	опционально
/CSK	LCD_CS	PD5	PB4	PB5	опционально
D/CX	LCD_DC	PD3	PB3	PB7	
SCL	LCD_SCL	PD2	PB7	PB1	SPI Clock
SDIO	LCD_SDA	PD1	PB5	PB2	SPI MOSI

Таблица 2.3. Назначение выводов для ILI9163.

Можно управлять настройками “x/y” - поворот, чтобы добиться корректной ориентации изображения для вашего дисплея. При необходимости можно настроить x-смещение. Если активировать параметр LCD_LATE_ON, тестер запускается с очищенным дис- плеем, вызывающим небольшую задержку при включении питания. Иначе возможно появление при старте некоторых случайных пикселей (шума) на экране.

2.6.3. ILI9341/ILI9342 использует 4-х строчный SPI или 8-битный параллельный режимы.

Дисплей	config-<MCU>.h	328 87,4%	644 51,7%	1280 13,2%	По умолчанию
RES	LCD_RES	PD4	PB2	PB4	опционально
CS	LCD_CS	PD5	PB4	PB5	опционально
DC	LCD_DC	PD3	PB3	PB7	
SCK	LCD_SCK	PD2	PB7	PB1	SPI Clock
SDI	LCD_SDI	PD1	PB5	PB2	SPI MOSI
SDO	LCD_SDO	-	PB6	PB3	nur ILI9341*
* неиспользованный					

Таблица 2.4. 4-строчный SPI-пинаут для ILI9341/ILI9342.

Дисплей	config-<MCU>.h	328	644	1280 13,1%	По умолчанию
	LCD_PORT			PORTB	
/RES X	LCD_RES			PB4	опционально
/CSX	LCD_CS			PB5	опционально
D/CX	LCD_DC			PB7	
WRX	LCD_WR			PB0	
RDX	LCD_RD			PB6	
	LCD_PORT2			PORTL	
D0	LCD_DB0			PL0	LCD_PORT2 Pin #0
D1	LCD_DB1			PL1	LCD_PORT2 Pin #1
D2	LCD_DB2			PL2	LCD_PORT2 Pin #2
D3	LCD_DB3			PL3	LCD_PORT2 Pin #3
D4	LCD_DB4			PL4	LCD_PORT2 Pin #4
D5	LCD_DB5			PL5	LCD_PORT2 Pin #5
D6	LCD_DB6			PL6	LCD_PORT2 Pin #6
D7	LCD_DB7			PL7	LCD_PORT2 Pin #7

Таблица 2.5. 8-битное параллельное назначение выводов для ILI9341/ILI9342.

Можно управлять настройками “x/y” - поворот и “x” - смещение, чтобы добиться корректной ориентации изображения для вашего дисплея. Не забывайте устанавливать координаты по “y” и “x” для контроллера (ILI9341 240x320 и ILI9342, 320x240). В некоторых модулях дисплея отключен расширенный набор команд ILI9341 (вывод EXTC подключен к Gnd). В этом случае вы можете увидеть размытое или малоконтрастное изображение, которое можно исправить, включив LCD_EXT_CMD_OFF. Ввиду относительно большого числа пикселей, работа дисплея несколько медленная. Весь экран полностью обновляется с частотой приблизительно 3 секунды с кварцем на 8 МГц и при использовании программной эмуляции SPI. Лучше использовать аппаратный SPI или параллельную шину.

2.6.4. ILI9481 (частично проверен)

использует 8-битную или 16-битную параллельные шины или 4-х строчный SPI.

Дисплей	config-<MCU>.h	328	644 51,6%	1280 13,2%	По умолчанию
	LCD_PORT			PORTB	
/RESX	LCD_RES			PB4	опционально
/CSX	LCD_CS			PB5	опционально
D/CX	LCD_DC			PB7	
WRX	LCD_WR			PB0	
RDX	LCD_RD			PB6	опционально
	LCD_PORT2			PORTL	
D0	LCD_DB0		PB0	PL0	LCD_PORT2 Pin #0
D1	LCD_DB1		PB1	PL1	LCD_PORT2 Pin #1
D2	LCD_DB2		PB2	PL2	LCD_PORT2 Pin #2
D3	LCD_DB3		PB3	PL3	LCD_PORT2 Pin #3
D4	LCD_DB4		PB4	PL4	LCD_PORT2 Pin #4
D5	LCD_DB5		PB5	PL5	LCD_PORT2 Pin #5
D6	LCD_DB6		PB6	PL6	LCD_PORT2 Pin #6
D7	LCD_DB7		PB7	PL7	LCD_PORT2 Pin #7

Таблица 2.6. Назначение выводов для 8-битного параллельного ILI9481/ILI9486.

Дисплей	config-<MCU>.h	328	644	1280	По умолчанию
D8	LCD_DB8			PC0	LCD_PORT3 Pin #0
D9	LCD_DB9			PC1	LCD_PORT3 Pin #1
D10	LCD_DB10			PC2	LCD_PORT3 Pin #2
D11	LCD_DB11			PC3	LCD_PORT3 Pin #3
D12	LCD_DB12			PC4	LCD_PORT3 Pin #4
D13	LCD_DB13			PC5	LCD_PORT3 Pin #5
D14	LCD_DB14			PC6	LCD_PORT3 Pin #6
D15	LCD_DB15			PC7	LCD_PORT3 Pin #7

Таблица 2.7. Дополнительное назначение выводов для 16-битного параллельного.

Дисплей	config-<MCU>.h	328 87,4%	644 43,8%	1280 11,2%	По умолчанию
RES	LCD_RES	PD4	PB2	PB4	опционально
CS	LCD_CS	PD5	PB5	PB5	опционально
D/C	LCD_DC	PD3	PB3	PB7	
SCL	LCD_SCL	PD2	PB7	PB1	SPI Clock
SDA	LCD_SDA	PD1	PB5	PB2	SPI MOSI

Таблица 2.8. Не рекомендуется 4-проводная пинаутная схема SPI.

Из-за высокого разрешения дисплея и цветовой схемы RGB666 (3 байта на пиксель) SPI работает довольно медленно, даже для аппаратного SPI и тактовой частоты MCU 16МГц. Поэтому я бы не рекомендовал использовать интерфейс SPI. Можно управлять настройкой (LCD_ROTATE) чтобы повернуть изображение. При необходимости вы также можете управлять настройками поворота “x” и / или “y”.

2.6.5. ILI9486 (частично протестировано)

использует 8-битную или 16-битную параллельные шины или 4-х строчный SPI. Назначение контактов такое же, как у ILI9481.

2.6.6. ILI9488 (частично протестировано)

использует 8-битную или 16-битную параллельные шины или 4-х строчный SPI. Назначение контактов такое же, как у ILI9481.

2.6.7. PCD8544 использует 3-х проводной SPI режим. Назначение контактов:

Дисплей	config-<MCU>.h	328 82,5%	644 39,1%	1280 10,0%	По умолчанию
/RES	LCD_RES	PD4	PB2	PB4	опционально
/SCE	LCD_SCE	PD5	PB4	PB5	опционально
D/C	LCD_DC	PD3	PB3	PB7	
SCLK	LCD_SCLK	PD2	PB7	PB1	SPI Clock
SDIN	LCD_SDIN	PD1	PB5	PB2	SPI MOSI

Таблица 2.9. Назначение выводов для PCD8544.

Так как у дисплея всего 84 пикселя в горизонтальной развертке, вы получите только 14 символов в строке при использовании шрифта 6x8. Таким образом, 2 символа не могут быть выведены на дисплей. Смягчить этот недостаток можно, если попробовать сократить некоторый текст в variables.h.

2.6.8. PCF8814 обычно использует 3-х проводной SPI режим. Назначение контактов (только программный режим):

Дисплей	config-<MCU>.h	328 82,6%	644 41,3%	1280 10,6%	По умолчанию
/RES	LCD_RES	PD4	PB2	PB4	
/CS	LCD_CS	PD5	PB4	PB5	опционально
SCLK	LCD_SCLK	PD2	PB7	PB1	SPI Clock
SDIN	LCD_SDIN	PD1	PB5	PB2	SPI MOSI

Таблица 2.10. Назначение выводов для PCD8814.

Можно управлять настройками “y” - поворот, чтобы добиться корректной ориентации изображения для вашего дисплея и “x” - поворот вверх-вниз с помощью MX контакта.

2.6.9. SH1106 (проверен не полностью) использует 3-х проводной, 4-х проводной SPI режим или I2C. 3-х проводной SPI требует программный режим и SPI_9 для включения.

Дисплей	config-<MCU>.h	328 83,0%	644 41,6%	1280 10,7%	По умолчанию
/RES	LCD_RESET	PD4	PB2	PB4	опционально
/CS	LCD_CS	PD5	PB4	PB5	опционально
A0	LCD_A0	PD3	PB3	PB7	
SCL (D0)	LCD_SCL	PD2	PB7	PB1	SPI Clock
SI (D1)	LCD_SI	PD1	PB5	PB2	SPI MOSI

Таблица 2.11. Пинаут для SH1106 4-х проводной SPI.

Дисплей	config-<MCU>.h	328 83,2%	644 41,6%	1280 10,7%	По умолчанию
/RES	LCD_RESET	PD4	PB2	PB4	опционально
/CS	LCD_CS	PD5	PB4	PB5	опционально
A0		Gnd			
SCL (D0)	LCD_SCL	PD2	PB7	PB1	SPI Clock
SI (D1)	LCD_SI	PD1	PB5	PB2	SPI MOSI

Таблица 2.12. Пинаут для SH1106 4-х проводной SPI (только бит-бах).

Дисплей	config-<MCU>.h	328 84,0%	644 41,9%	1280 10,7%	По умолчанию
/RES	LCD_RESET	PD4	PB2	PB0	опционально
/CS		Gnd			
SCL (D0)	I2C_SCL	PD1	PC0	PD0	
SDA (D1)	I2C_SDA	PD0	PC1	PD1	
SA0	Gnd (0x3c)				3.3V (0x3d)

Таблица 2.13. Назначение выводов для SH1106 I2C.

Используя настройки поворота по оси “x/y”, вы можете изменить ориентацию вывода, если необходимо. Для многих дисплейных модулей на базе SH1106 необходимо, чтобы смещение по x было равно 2.

2.6.10. SSD1306 использует 3-х или 4-х проводной SPI режим а также I2C. 3-х проводной SPI поддерживается только программным способом, SPI_9 должен быть включен.

Дисплей	config-<MCU>.h	328 83,1%	644 41,6%	1280 10,7%	По умолчанию
/RES	LCD_RES	PD4	PB2	PB4	опционально
/CS	LCD_CS	PD5	PB4	PB5	опционально
DC	LCD_DC	PD3	PB3	PB7	
SCLK (D0)	LCD_SCLK	PD2	PB7	PB1	SPI Clock
SDIN (D1)	LCD_SDIN	PD1	PB5	PB2	SPI MOSI

Таблица 2.14. Назначение контактов для 4-х проводного SPI.

Дисплей	config-<MCU>.h	328 83,2%	644 41,6%	1280 10,7%	По умолчанию
/RES	LCD_RES	PD4	PB2	PB4	опционально
/CS	LCD_CS	PD5	PB4	PB5	опционально
SCLK (D0)	LCD_SCLK	PD2	PB7	PB1	SPI Clock
SDIN (D1)	LCD_SDIN	PD1	PB5	PB2	SPI MOSI

Таблица 2.15. Для 3-х проводного SPI(только программный режим).

Дисплей	config-<MCU>.h	328 84,1%	644 42,0%	1280 10,7%	По умолчанию
/RES	LCD_RES	PD4	PB2	PB0	опционально
/SCL (D0)	I2C_SCL	PD1	PC0	PD0	опционально
SDA (D1&2)	I2C_SDA	PD1	PC1	PB0	
SA0 (D/C)	Gnd (0x3c)				3.3V (0x3d)

Таблица 2.16. Назначение выводов для SSD1306 I2C.

Можно управлять настройками “x/y” - поворот, чтобы добиться корректной ориентации изображения для вашего дисплея.

2.6.11. ST7036 (не проверен) ST7036 использует 4-х битный параллельный интерфейс или 4-х проводной SPI.

Дисплей	config-<MCU>.h	328 76,2%	644 38,1%	1280 9,8%	По умолчанию
DB4	LCD_DB4	PD0	PB4	PB0	
DB5	LCD_DB5	PD1	PB5	PB1	
DB6	LCD_DB6	PD2	PB6	PB2	
DB7	LCD_DB7	PD3	PB7	PB3	
RS	LCD_RS	PD4	PB2	PB4	
R/W	LCD_RW	Gnd	???	???	опционально
E	LCD_EN	PD5	PB3	PB5	
XRESET		Vcc	???	???	опционально

Таблица 2.17. Назначение контактов для параллельного 4-х битного интерфейса.

Дисплей	config-<MCU>.h	328 76,1%	644 38,1%	1280 9,8%	По умолчанию
XRESET	LCD_RESET	PD4	PB2	PB4	опционально
CSB	LCD_CS	PD5	PB4	PB5	опционально
RS	LCD_RS	PD3	PB3	PB7	
SCL (DB6)	LCD_SCL	PD2	PB7	PB1	SPI Clock
SI (DB7)	LCD_SI	PD1	PB5	PB2	SPI MOSI

Таблица 2.18. Для 4-х проводного SPI.

ST7036i может использовать интерфейс I2C, но он пока не поддерживается. Отдельная функция ST7036 является специальным контактом для включения расширенной системы команд (пин EXT), которая обычно используется. В том случае, если настройки LCD_EXTENDED_CMD и LCD_CONTRAST отключены, вывод должен быть закомментирован.

2.6.12. ST7565R использует 4/5 строчный SPI режим.

Дисплей	config-<MCU>.h	328 83,2%	644 41,7%	1280 10,7%	По умолчанию
/RES	LCD_RESET	PD0	PB2	PB4	опционально
/CS1	LCD_CS	PD5	PB4	PB5	опционально
A0	LCD_A0	PD1	PB3	PB7	
SCL (DB6)	LCD_SCL	PD2	PB7	PB1	SPI Clock
SI (DB7)	LCD_SI	PD3	PB5	PB2	SPI MOSI

Таблица 2.19. Пинаут для 4/5-линейного SPI ST7565R.

Можно управлять настройками “x/y” - поворот и “x” - смещение, чтобы добиться корректной ориентации изображения для вашего дисплея.

2.6.13. ST7735 использует 4-х проводной SPI режим.

Дисплей	config-<MCU>.h	328 95,7%	644 47,9%	1280 12,2%	По умолчанию
/RESX	LCD_RES	PD4	PB2	PB4	опционально
/CSK	LCD_CS	PD5	PB4	PB5	опционально
D/CX	LCD_DC	PD3	PB3	PB7	
SCL	LCD_SCL	PD2	PB7	PB1	SPI Clock
SDIO	LCD_SDA	PD1	PB5	PB2	SPI MOSI

Таблица 2.20. Назначение контактов для ST7735.

Можно управлять настройками “x/y” - поворот, чтобы добиться корректной ориентации изображения для вашего дисплея. Если активировать параметр LCD_LATE_ON, тестер запускается с очищенным дисплеем, вызывающим небольшую задержку при включении питания. Иначе вы можете наблюдать при старте некоторые случайные пиксели (шум) на экране. Недавно появился полу-совместимый вариант ЖК-дисплея, который не будет работать со стандартным драйвером. Решение состоит в том, чтобы использовать модифицированный драйвер (lcd_semi_st7735 вместо lcd_st7735) с одинаковыми настройками.

2.6.14. ST7920 использует параллельный 4-х битный режим или SPI.

Дисплей	config-<MCU>.h	328 83,5%	644 xx,x%	1280 xx,x%	По умолчанию
/XRESET	LCD_RES	PD4	PB2	PB4	опционально
/CS	LCD_CS	PD5	PB4	PB5	опционально
SCLK	LCD_SCLK	PD2	PB7	PB1	SPI Clock
SDIN	LCD_SDIN	PD1	PB5	PB2	SPI MOSI

Таблица 2.21. Назначение контактов для ST7920 SPI.

Дисплей	config-<MCU>.h	328 83,8%	644 41,8%	1280 10,7%	По умолчанию
/XRESET	LCD_RESET	Vcc	Vcc	Vcc	
E	LCD_EN	PD5	PB3	PB5	
RS	LCD_RS	PD4	PB2	PB4	
RW	LCD_RW	Gnd	Gnd	Gnd	
D4	LCD_DB4	PD0	PB4	PB0	
D5	LCD_DB5	PD1	PB5	PB1	
D6	LCD_DB6	PD2	PB6	PB2	
D7	LCD_DB67	PD3	PB7	PB3	

Таблица 2.22. Назначение контактов для параллельного 4-х битного интерфейса.

Из-за неудачной адресации для ST7920 могут применяться только шрифты шириной 8 пикселей. Чтобы справиться с горизонтальной 16-ти битной адресацией, я должен был добавить экранный буфер для символов.

2.6.15. STE2007/HX1230 использует 3-х проводный SPI режим.

Дисплей	config-<MCU>.h	328 82,5%	644 41,3%	1280 10,6%	По умолчанию
/RES	LCD_RES	PD4	PB2	PB4	опционально
/CS	LCD_CS	PD5	PB4	PB5	опционально
SCLK	LCD_SCLK	PD2	PB7	PB1	
SDIN	LCD_SDIN	PD1	PB5	PB2	

Таблица 2.23. Назначение контактов (только программная поддержка).

Можно управлять настройками “x/y” - поворот, чтобы добиться корректной ориентации изображения для вашего дисплея.

2.6.16. VT100 терминал Драйвер VT100 заменяет ЖК-дисплей и выводит информацию посредством VT100 терминала. Раздел конфигурации для VT100 уже включает активацию последовательного интерфейса TTL. Помните, что драйвер VT100 отключит другие опции, которые имели отношение к последовательному интерфейсу и могли бы исказить информацию.

2.7. Кнопки тестирования и управления

Основное управление тестером – кнопка “Тест”, но дополнительные опции управления также поддерживаются для более удобной работы.

2.7.1. Кнопка Тест Кнопка "Тест" запускает устройство, а также управляет пользовательским интерфейсом. Для этого тестер различает кратковременные и длительные нажатия кнопки (> 0.3 с). Быстрое нажатие обычно используется для продолжения работы или выбора пункта меню. Долгое нажатие выполняет зависящее от контекста действие. Если тестер будет ожидать, пока вы нажмете кнопку, то он даст подсказку и выведет на экран курсор на последней позиции во второй строке или в нижний правый угол LCD дисплея. Немигающий курсор сигнализирует о том, что на экран будет выведена дополнительная информация, а мигающий означает, что тестер возобновит циклическое тестирование. Курсор не задействован в меню и некоторых опциях, потому что это необходимо для правильного нажатия кнопки. Возможно включение подсказок действий, если у вашего прибора есть дополнительные кнопки и возможность отображать на дисплее достаточное количество строк текста (см. UI_KEY_HINTS в config.h). Подсказка отображается вместо курсора, если он доступен. На данный момент есть только одна такая подсказка для доп. кнопок (Меню / Тест).

2.7.2. Поворотный энкодер (аппаратная опция) С поворотным энкодером вы получите некоторую дополнительную функциональность в пользовательском интерфейсе, но это зависит от контекста. Некоторые режимы работы используют изменение скорости вращения энкодера, чтобы управлять дополнительными настройками или регулировкой значений. Алгоритм для поддержки работы энкодера основан на подсчете кода Грэя за один импульсный шаг или остановку (ENCODER_PULSES). Большинство энкодеров выдают 2 или 4 кодовых импульса за один шаг. Количество шагов или остановок за полный оборот на 360 угловых ступеней также приняты во внимание (ENCODER_STEPS). Вы можете использовать это значение для точной настройки угловой скорости и оптимизировать обратную связь. Более высокое значение замедляет скорость, более низкое значение – увеличивает её. В случае если необходимо изменить направление вращения на обратное, просто поменяйте назначение выводов процессора для контактов A и B в файле конфигурации config_<mcu>.h. Изменение угловой скорости измеряет время для двух шагов. Вы должны повернуть энкодер, по крайней мере, на два шага для средней скорости. Для более высокой скорости - на три шага. Одиночный шаг приводит к самой низкой скорости.

2.7.3. Кнопки управления +/- (аппаратная опция) Если вы предпочитаете кнопки вместо энкодера, то есть возможность добавить две альтернативные. Кнопки имеют такое же подключение как и энкодер (резисторы подтяжки, низкий уровень). Длительное нажатие кнопки увеличит "скорость если продолжать её удерживание.

2.7.4. Сенсорный экран (аппаратная опция) Альтернативная опция управления - сенсорный экран. Обратите внимание на то, что экран должен быть достаточно большой, с поддержкой приблизительно 8 текстовых строк по 16 символов в каждой. Чтобы оставить драгоценное информационное пространство на дисплее, пользовательский интерфейс не отображает иконки сенсора управления. Для этого есть невидимые сенсорные зоны слева и справа (каждая шириной в 3 символа), а также в верхней и нижней части дисплея (высотой в 2 строки) и один в центральной области. Левая и верхняя зоны служат для уменьшения значения или перемещения вверх по меню, в то время как нижняя и правая зоны - для увеличения или перемещения вниз по меню. На самом деле они делают то же, что и энкодер. Более длительное касание сенсорной зоны увеличивают скорость изменения, если это поддерживается функцией или инструментом (аналогично быстрому вращению энкодера). Центральная зона действует аналогично кнопке "Тест" но она не активирует, например, опцию проверки стабилитронов. Сенсорному экрану нужна настройка и калибровка для правильного функционирования. Калибровка автоматически запускается после включения тестера, если нет сохраненных данных в EEPROM. Вы можете также выполнить настройку через главное меню. Процедура простая. Если вы видите звездочку (желтую * на цветном дисплее), просто коснитесь её. После этого тестер удаляет звездочку и отображает исходную позицию x/y. Первая точка регулировки находится сверху справа, вторая точка - внизу слева. Вы можете использовать регулировку до трех раз. Можно пропустить процедуру в любое время если нажать тестовую кнопку. Если есть проблемы с калибровкой в виде странных "x/y" позиций, проверьте ориентацию сенсорного экрана относительно дисплея. У драйвера есть опции зеркального отражения или изменения ориентации. Предполагается что верхний левый угол дисплея - это нулевая позиция. Некоторые подсказки о необходимых настройках для конкретных значений x и y:

* первая точка регулировки: сверху справа.		
x	y	настройки
низ	низ	ТTOUCH_поворот_X
низ	верх	TOUCH_поворот_X & TOUCH_поворот_Y
hoch	низ	ни один
верх	верх	TOUCH_поворот_Y
Не забывайте сохранять данные после успешной настройки. . . (главное меню: сохранить).		

Таблица 2.24. Настройки в зависимости от значений x/y.

Поддерживаемые контроллеры сенсорного экрана:

- ADS7843/XPT2046

Вы найдете параметры конфигурации дисплея в файле config-<mcu>.h(в настоящее время это config_644.h и config_1280.h из-за отсутствия свободных портов в ATmega328).

2.7.5. Пользовательский интерфейс Есть несколько способов для того, чтобы сделать вывод результатов на дисплей более удобным и легко читаемым.

Для дисплеев с более чем 2 текстовыми строками:

- показать кнопки подсказки вместо курсора, если включена настройка (UI_KEY_HINTS)

Для графических дисплеев:

- все параметры перечисленные ранее

- иконки с распиновкой для 3-х выводных полупроводников (SW_SYMBOLS)

неприменимо для дисплеев с низким разрешением

Для цветных графических дисплеев:

- все параметры перечисленные ранее




- цветовое обозначение для тестовых контактов (UI_PROBE_COLORS)

- настраиваемый цвет для заголовков (UI_COLORED_TITLES)

- настраиваемый цвет для курсора и подсказок для кнопок (UI_COLORED_CURSOR)
- цветовой код для резисторов (SW_R_E24_5_CC, SW_R_E24_1_CC и SW_R_E96_CC)

2.7.6. Соединение с компьютером Тестер может поддерживать последовательный интерфейс TTL для связи с ПК. Это может быть соединение только по TX для вывода информации о найденных компонентах или двунаправленный интерфейс для автоматизации. В обоих случаях необходим последовательный интерфейс TTL, который настраивается в файле config.h (см. раздел «Busses») на странице 56. Специальные символы заменяются стандартными, например, Ω (Ом) становится простой R. Для меню и инструментов нет последовательного вывода, кроме результатов проверки оптопары.

Таблица преобразования:

	> <
	
	
Ω	R
μ	u

Подсказки:

- 9600 8N1;
- новая строка <CR> <NL>

2.7.7. Последовательный выход Тестер определяет компоненты и передает информацию на ПК посредством простой терминальной программы, если эта функция включена (см. UI_SERIAL_COPY в разделе «misc settings» в файле config.h) на странице 51. Последовательный вывод следует за выводом информации на ЖК-дисплее, но только для найденных компонентов.

2.7.8. Автоматизация Функция автоматизации позволяет управлять тестером с помощью удаленных команд через двунаправленное последовательное соединение. Для включения этой функции см. UI_SERIAL_COMMANDS в разделе «misc settings» в файле config.h на странице 51. По умолчанию поведение тестера немного изменится. Автоматизация обеспечивает режим ожидания и прибор **не будет** автоматически проверять компонент после включения. Командный интерфейс довольно прост. Вы отправляете команду - тестер отвечает. Связь основана на текстовых строках ASCII и командах, чувствительных к регистру. Каждая командная строка должна заканчиваться символом <CR> <NL> или <NL>. Имейте в виду, что тестер будет принимать команды только в режиме ожидания обратной связи от пользователя после включения питания, отображения компонента или выполнения функции меню. Строки отклика заканчиваются символом <CR> <NL>. См. раздел «Команды управления» 5 на странице 40 для объяснения списка команд.

2.7.9. VT100 Тестер может выводить информацию на терминал VT100 вместо жидкокристаллического дисплея (см. VT100 в разделе "Дисплеи") 2.6 на странице 19). Для сохранения корректной информации все другие опции для последовательного интерфейса отключены.

3.0.1. Включить Долгое нажатие кнопки при запуске прибора выбирает режим автоудержания показаний. В этом режиме устройство ожидает быстрого нажатия кнопки после отображения результата прежде, чем продолжить работу. Иначе тестер выбирает непрерывный (циклический) режим работы по умолчанию. Вы можете инвертировать выбор режима работы в `config.h` (`UI_AUTOHOLD`). После включения будет кратко показана микропрограммная версия. Более длительное нажатие кнопки (> 2 с) сбросит тестер к микропрограммным значениям по умолчанию. Это может быть полезным, если вы установили недопустимую контрастность дисплея. Если тестер обнаружит проблему с сохраненными значениями корректировок, то он выведет на экран ошибку контрольной суммы. Эта ошибка указывает на поврежденный EEPROM и устройство будет использовать микропрограммные значения по умолчанию. Для тестера с ручным выключателем питания вместо программного, пожалуйста, включите опцию `POWER_SWITCH_MANUAL` в `config.h`. В этом режиме прибор не сможет отключиться автоматически!

3.0.2. Тестирование После запуска прибор входит в режим тестирования и определяет подключенный компонент. В непрерывном (циклическом) режиме это действие автоматически повторится после короткой паузы. Если несколько раз компонент не будет определён, то тестер выключится. В режиме автоудержания – “autohold” (подсказка курсором) тестер выполняет один запуск и ожидает нажатия кнопки или поворота вправо энкодера. Задержка цикла и автоматическое выключение питания для непрерывного режима могут быть скорректированы настройками `CYCLE_DELAY` страница 48 и `CYCLE_MAX` страница 51 в `config.h`. Существует дополнительное автоматическое выключение питания для режима “autohold” (`POWER_OFF_TIMEOUT`) страница 52, которое активно только во время циклического тестирования. В обоих режимах можно вывести меню с дополнительными функциями или выключить устройство. Для получения дополнительной информации читайте ниже.

3.0.3. Контроль источника питания Каждый цикл тестирования начинается с определения напряжения батареи и вывода на дисплей её состояния (норма, слабая, разряжена). Тестер выключится при достижении нижнего порогового напряжения. Батарея контролируется регулярно во время работы. Конфигурация по умолчанию для контроля питания установлена для 9-ти вольтовой батареи, но она может быть изменена для большинства других источников питания. Посмотрите секцию "power management" в `config.h` на странице 52 для всех вариантов настройки. Контроль питания может быть полностью отключен опцией `BAT_NONE`, установлен на проверку постоянного напряжения для источников питания менее 5 В опцией `BAT_DIRECT` или установлен для контроля через делитель напряжения - опции `BAT_R1`, `BAT_R2` и `BAT_DIVIDER`. Некоторые тестеры поддерживают дополнительный внешний источник питания, но не позволяют его контролировать. В этом случае предусмотрена опция `BAT_EXT_UNMONITORED`, которая позволяет предотвратить проблемы с автоматическим выключением питания при низком уровне заряда. В этом режиме состояние батареи будет отображаться на дисплее как "ext"(внешняя). Для слабого и низкого уровня заряда предусмотрены настройки `BAT_WEAK` и `BAT_LOW`, а опция `BAT_OFFSET` учитывает падение напряжения, например, если в схеме установлен защитный диод и транзистор управления питанием структуры `ppr`.

3.0.4. Выключение При отображении результата последнего теста долгое нажатие кнопки “Тест” приводит к выключению прибора. Он покажет сообщение “bye!”(до свидания) а затем выключится. При более коротком нажатии кнопки тестер останется включенным. Это вызвано особенностью схемы управления питанием.

3.0.5. Меню Вы можете войти в меню двумя короткими нажатиями кнопки “Тест” после отображения последнего найденного компонента или выполненной функции (может потребоваться некоторая практика). Если поддержка поворотного энкодера включена, поворот влево также позволит войти в меню. Старый метод входа в меню путем короткого замыкания всех трех тестовых зондов также может быть включен через опцию (см. UI_SHORT_CIRCUIT_MENU). Если вызвано меню, то короткое нажатие кнопки выбирает следующий пункт, а долгое нажатие выполняет его. На дисплее с 2 строками можно видеть навигационную подсказку в нижнем правом углу. Символ “>”, если есть следующий пункт, или “<” для самого последнего пункта (возврат к первому пункту). В дисплеях с более чем 2 строками выбранный пункт отмечен символом “*” с левой стороны. С поворотным энкодером можно перемещаться по пунктам вверх или вниз, изменяя направление вращения и коротким нажатием кнопки запускать выбранный пункт. Возврат к первому пункту также возможен. Некоторые функции подсказывают вам схему расположения тестовых контактов, которые необходимо использовать при тестировании. Эта информация будет выведена на экран в течение нескольких секунд, но может быть пропущена коротким нажатием кнопки "Тест". Для функций, генерирующих сигнал на тестовый контакт No 2 - он используется в качестве вывода по умолчанию. В этом случае контакты No 1 и No 3 связаны с общим проводом (GND). Если тестер будет сконфигурирован для вывода специального сигнала на порт (OC1B), тестовые контакты не используются и их схема не будет выведена на экран.

3.1. Меню

3.1.1. ШИМ генератор Это та функция, которую ждали:-) Прежде, чем скомпилировать прошивку, выберите генератор ШИМ с простым пользовательским интерфейсом или с альтернативным интерфейсом для тестеров с поворотным энкодером и крупным дисплеем. Возможно отображение длительности импульса, если включить в конфигурации PWM_Show_duration.

Конфигурация выхода:

Тестовый контакт No2	выход ШИМ сигнала (с резистором на 680 Ω для ограничения тока)
Тестовый контакт No1 и No3	общий GND(земля)

3.1.2. Простой интерфейс генератора ШИМ Сначала вы должны выбрать желаемую частоту ШИМ в простом меню. Короткое нажатие кнопки - выбор нужной частоты, длинное - запускает вывод ШИМ для выбранной частоты. Коэффициент ШИМ регулировки устанавливается по умолчанию на 50%. Короткое нажатие кнопки "Тест" увеличивает скважность на 5%, долгое нажатие - уменьшает на 5%. Выход из функции - нажатие кнопки "Тест" дважды быстро. Если подключён энкодер, можете использовать его, чтобы выбрать частоту в меню и сделать ШИМ регулировку с шагом в 1%.

3.1.3. Альтернативный интерфейс генератора ШИМ Переключение между частотой и длительностью осуществляется нажатием кнопки "Тест". Выбранный параметр отмечается звёздочкой. Поверните энкодер по часовой стрелке, чтобы увеличить значение или против часовой чтобы уменьшить его. Более быстрый поворот энкодера увеличивает размер шага настройки. Долгое нажатие кнопки устанавливает значение по умолчанию (частота - 1кГц, скважность - 50%). Два коротких нажатия кнопки "Тест" выход из режима генератора.

3.1.4. Генератор прямоугольного сигнала Сигнальный генератор выдаёт меандр с переменной частотой до 1/4 от тактовой частоты процессора (2МГц для кварца на 8МГц). Частота по умолчанию 1000Гц, можно изменять её, вращая энкодер. Скорость вращения определяет изменение частоты, т.е. медленное вращение - малый шаг изменения, быстрое вращение - больший шаг изменения частоты. Так как генерация сигналов основана на тактовом режиме работы процессора, возможна только пошаговая установка частоты. Для низких частот шаги довольно мелкие, но для высоких частот они становятся больше и больше. Долгое нажатие кнопки "Тест" сбрасывает частоту по умолчанию до 1кГц, а два коротких нажатия кнопки – выход из режима генерации сигналов.

Конфигурация выхода:

Тестовый контакт No2	вывод (с ограничительным резистором на 680 Ω)
Тестовый контакт No1 и No3	общий GND(земля)

Подсказка: для работы в этом режиме необходим энкодер!

3.1.5. Тестирование стабилитронов (аппаратная опция) Если в устройстве присутствует встроенный преобразователь DC-DC, который создает высокое испытательное напряжение для измерения напряжения пробоя стабилитрона, то он должен быть соединён со специальными тестовыми контактами. Если кнопка "Тест" нажата и работает преобразователь, то на дисплее тестера будет отображаться текущее напряжение. Если кнопка удерживалась нажатой достаточно долго для стабильного теста напряжения, то после её отпускания кнопки будет показано минимальное измеренное напряжение. Можно повторять этот тест, пока вам нравится. Для выхода из режима дважды быстро нажмите кнопку "Тест". Если в приборе используется делитель напряжения 10:1 без преобразователя для измерения внешнего напряжения или повышающий DC-DC работает все время, вы можете выбрать альтернативный режим (ZENER_UNSWITCHED), в котором напряжение измеряется периодически без нажатия кнопки "Тест". Когда вы видите курсор справа внизу, то можно выйти из режима двумя быстрыми нажатиями тестовой кнопки. Есть еще

один вариант, что- бы запустить проверку стабилитронов во время автоматического тестирования (HW_PROBE_ZENER). Если ни один из компонентов не будет определен на стандартных тестовых контактах, Тестер проверит напряжение на контактах ZENER(K-A).

Если ваш тестер имеет нестандартный делитель напряжения (не 10:1), можно включить конфигурацию ZENER_DIVIDER_CUSTOM и указать номиналы резисторов (ZENER_R1 и ZENER_R2).

Как подключить стабилитрон:

Контакт +:	катод
Контакт - :	анод

3.1.6. Внутрисхемное измерение ESR Конденсаторы тестируются внутрисхемно и на экран выводится значение ёмкости и ESR, если измерение обнаруживает допустимый конденсатор.

Убедитесь, что конденсатор разряжен прежде, чем подключить к нему тестер!

Значения могут отличаться от стандартных измерений (вне схемы), потому что любой компонент в схеме, расположенный параллельно с тестируемым конденсатором, будет влиять на результат. Для начала измерения нажмите кнопку "Тест". Два быстрых коротких нажатия кнопки – выход из этого режима.

Подключение конденсатора:

Тестовый контакт No1:	плюс конденсатора
Тестовый контакт No3:	минус конденсатора GND(земля)

3.1.7. Проверка конденсатора на ток утечки Тест утечки в конденсаторе заряжает его и выводит на экран ток и напряжение через известное сопротивление. Первоначально, заряд конденсатора осуществляется через R1 (680Ω) а затем через резистор Rh (470kΩ), когда ток оказывается ниже определенного порога. Каждый цикл начинается с напоминания схемы контактов. После подключения конденсатора нажмите кнопку "Тест" (или поверните ручку энкодера вправо), чтобы запустить процесс зарядки. Затем, после окончания заряда, нажмите кнопку снова, тестер разрядит конденсатор и будет отображать напряжение на нём до тех пор, пока разряд не достигнет нижнего порога. Чтобы выйти из проверки, нажмите кнопку "Тест" дважды.

Подсказка: обращайте внимание на полярность электролитических конденсаторов!

Как подсоединить конденсатор:

Тестовый контакт No1:	положительный вывод
Тестовый контакт No3:	отрицательный вывод Gnd(земля)

3.1.8. R/C/L - монитор Монитор непрерывно измеряет пассивный компонент, подключенный к контактам No1 и 3. После запуска режима прибор на несколько секунд отображает подсказку распиновки контактов, которую можно пропустить, нажав кнопку тестирования. Между измерениями есть задержка в одну или две секунды, обозначенная значком курсора в правом нижнем углу, во время которой можно выйти из мониторинга двумя короткими нажатиями тестовой кнопки.

Доступные варианты мониторов:

- R-Monitor (сопротивление)
- C-Monitor (ёмкость + опционально ESR)
- L-Monitor (индуктивность)
- R/C/L-Monitor (R + опционально L или C + опционально ESR)
- R/L-Monitor (сопротивление + индуктивность).

3.1.9. LC-измеритель(аппаратная опция) Аппаратная опция LC-измерителя основана на простой схеме генератора LC, используемой в недорогих измерителях на PIC контроллере. Обычная схема (82мкГн и 1нФ) имеет базовую частоту около 595кГц, а подключение дополнительного конденсатора или индуктивности уменьшает эту частоту. С помощью эталонного конденсатора с известным значением, измеренными частотами и математической формулы неизвестный конденсатор или индуктивность могут быть вычислены. Измерители PIC LC обычно имеют диапазоны измерений 10нГн - 100мГн и 0,1пФ

- 900нФ. Они используют время стробирования 100мс для частотомера. ПО использует автоматическое переключение диапазона со временем стробирования 100мс и 1000мс чтобы улучшить разрешение для компонентов с малым значением. Таким образом, диапазоны начинаются примерно с 1нГн и 10fF(0.01пФ). Максимальная измеряемая индуктивность составляет примерно 150мГн.

Что касается верхнего предела емкости, то у меня возникла проблема с печатной платой. При емкости около 33нФ на выходном сигнале начинали появляться выбросы на переднем и заднем фронтах, искажая работу частотомера, который видит больше импульсов, чем есть на самом деле. Это известная проблема с цепями генератора на LM311. После испытания нескольких модификаций без видимого успеха, компаратор на основе триггера Шмидта кажется лучшим выбором для очистки выходного сигнала генератора. CMOS 4093 работает нормально в этом качестве. С чистым выходным сигналом максимальная измеряемая емкость составляет около 120нF (выше генератор LC становится нестабильным). Тестер обеспечивает более низкую границу частоты чем 10 кГц, то есть возможны теоретически максимальные значения 250пН или 3,5 мкФ, если генератор LC будет работать стабильно.

При запуске LC-метра тестер выполнит самонастройку, указанную сообщением «настройка ...». После этого можно подключить измеряемый конденсатор или индуктивность. Короткое нажатие кнопки переключает C и L режимы измерения (режим по умолчанию - C). Частота LC-генератора дрейфует со временем (до 100Гц) и требует повторной настройки. Если вы увидите увеличение нулевого значения или “-” без подключенных компонентов, запустите самонастройку долгим нажатием кнопки. Если есть проблема с самонастройкой или вы отмените ее, нажав какую-либо кнопку, тестер выйдет из режима измерений и сообщит об ошибке. Два коротких нажатия кнопки – выход из режима LC-измеритель.

Подсказки: в качестве эталонного нужно использовать высококачественный пленочный конденсатор емкостью 1нФ (с хорошей стабильностью). Вы также можете использовать обычный пленочный конденсатор на 1нФ, просто измерьте его емкость с помощью надежного LCR-измерителя и обновите значение LC_METER_C_REF. Если вас интересует частота LC-генератора и ее дрейф, то включите опцию LC_METER_SHOW_FREQ.

3.1.10. Частотомер (аппаратная опция) Есть два варианта частотомера.

Базовый – простой пассивный вход на контакте T0 MCU. У расширенного варианта есть входной буфер, два генератора для тестирования кварцев (с низкой и высокой частотами) а также дополнительный делитель частоты. Принципиальные схемы для обоих вариантов изображены в документации от Karl-Heinz [15].

3.1.11. Базовый вариант Частотомером можно измерить частоты приблизительно от 10Гц до 1/4 тактовой частоты процессора, с разрешением 1Гц для частот ниже 10кГц. Частота измеряется и выводится на экран дисплея постоянно, пока вы не закончите измерение двойным нажатием кнопки. Алгоритм автоматически выбирает диапазон длительности между 10мс и 1000мс на основе частоты. Вывод T0 может использоваться совместно с дисплеем.

3.1.12. Расширенный вариант Расширенный вариант частотомера имеет дополнительный делитель и позволяет измерять более высокие частоты. Теоретический верхний предел - 1/4 от тактовой частоты процессора, умноженная на делитель (16:1 или 32:1). Настройки определены в файле config_<mcu>.h. Так же не забывайте устанавливать корректное значение делителя частоты в config.h. Тестовый канал частотомера (буферизованный вход, кварцевый генератор для низкой и высокой частоты) настраивается нажатием кнопки “Тест” или вращением энкодера. Как обычно, два коротких нажатия кнопки “Тест” – выход из режима частотомера.

3.1.13. Счетчик событий (аппаратная опция) Счетчик событий использует вывод T0 в качестве выделенного входа и срабатывает от переднего фронта сигнала. Вывод T0 нельзя использовать совместно с дисплеем. Рекомендуется простой метод подсчета. Режим счетчика выбирается в небольшом меню, которое также отображает счетчик значений. Пункты меню выбираются коротким нажатием, настройки меняются поворотным

энкодером или дополнительными кнопками.

- Счетчик подсчет времени и событий
- Время (n) подсчет событий за заданный период времени
- События(t) подсчет времени для заданного количества событий

Первый пункт - режим счетчика. Второй пункт «n» - это количество событий. В режиме событий он покажет порог срабатывания, который можно изменить. Длительное нажатие кнопки сбрасывает порог к значению по умолчанию (100). В других режимах подсчета этот пункт заблокирован. Следующий пункт «t» - это период времени в секундах. Тот же алгоритм и для режима времени (значение по умолчанию 60с). Последний элемент запускается или останавливается после долгого нажатия кнопки. Когда счетчик запускает подсчитанные события, прошедшее время обновляется каждую секунду и после остановки отображается результат. Ограничение на период времени составляет 43200с (12 часов), а для событий $4 * 10^9$. Если какой-либо из этих пределов превышен, подсчет автоматически останавливается. Предел или порог события (в режиме событий) проверяется каждые 200мс, поэтому может быть некоторое превышение в случае более 5 событий/сек.

Выход триггера Дополнительно можно включить выход триггера (EVENT_COUNTER_TRIGGER_OUT), чтобы управлять каким-нибудь другим устройством. Выход триггера установлен на высокий уровень при подсчете, т. е. на подъем фронта при старте и спаде при остановке.

Назначение контактов выход триггера:

- Тестовый контакт No1: GND(земля)
- Тестовый контакт No2: выход (с резистором 680 Ω для ограничения тока)
- Тестовый контакт No3: GND(земля)

3.1.14. Тест поворотного энкодера Этот тест проверяет поворотные энкодеры для определения схемы его контактов. Вам необходимо подсоединить контакты А, В и общий контакт энкодера к тестовым контактам и повернуть энкодер на несколько шагов по часовой стрелке. Для алгоритма тестирования нужен поворот энкодера минимум на четыре шага, чтобы определить надлежащую функцию и схему контактов. Изменение направления вращения важно для правильного определения неизвестных А и В контактов. Когда поворотный энкодер будет обнаружен, прибор выведет на экран схему контактов и будет ожидать нажатия кнопки (или момент для циклического режима) прежде, чем продолжить тестирование. Чтобы выйти из теста энкодера нажмите кнопку "Тест" один раз во время тестирования.

3.1.15. Контрастность дисплея Вы можете скорректировать уровень контрастности для некоторых графических LCD модулей. Короткое нажатие кнопки увеличивает уровень, долгое - уменьшает его. Два коротких нажатия кнопки - выход из настройки. Поворотный энкодер также позволяет скорректировать уровень контрастности.

3.1.16. Приёмник ИК RC/Декодер Эта функция обнаруживает и декодирует сигналы от пультов дистанционного управления IR и требует подключения модуля приемника IR, например серии TSOP. При компиляции прошивки вы можете выбирать между двумя вариантами - приёмником TSOP, который подключен к тестеру и аппаратным встроенным модулем приёмника ИК. Первый требует соединения модуля ИК со стандартными тестовыми контактами. Второй - аппаратный модуль ИК, соединенный со специальным портом микропроцессора. На дисплее тестера будет отображен обнаруженный протокол, а также адрес (если доступен), команда и, в некоторых случаях, дополнительные данные в шестнадцатеричном виде. Формат:

<протокол> <зона данных>

Для ошибочного или неизвестного пакета и протокола будет показан символ "?". Для неизвестного протокола тестер выводит на экран число пауз и импульсов, продолжительность первого импульса и первой паузы в модулях 50 μ s:

? <импульсы>:<первый импульс>-<первая пауза>

Если число импульсов одинаково для различных кнопок RC - наиболее вероятно PDM или ШИМ модуляция. Изменяющееся число импульсов указывает на двухфазную модуляцию. Чтобы выйти из этого режима нажмите кнопку "Тест".

Поддерживаемые протоколы и их массив данных:

- JVC
<адрес >:<команда>
- Kaseikyō (японский код, 48 бит)
<код производителя >:<система>-<устройство>:<функция>
- Matsushita(Panasonic MN6014, C6D6 / 12 бит)
<выборочный код>:<кодовые данные>
- Motorola
<команда>
- NEC (стандарт и расширенный)
<адрес>:<команда>
- R для повтора цикла - Proton / Mitsubishi (M50560)
<адрес>:<команда>
- RC 5 (стандарт)
<адрес>:<команда>
- RC 6 (стандарт)
<адрес>:<команда>
- Samsung / Toshiba (32 бита)
<выборочный код>:<кодовые данные>
- Sharp / Denon
<адрес>:<команда>
- Sony SIRC (12, 15 и 20 бит)
12 15: <команда >:<адрес>
20: <команда >:<адрес>:<дополнительно>

Дополнительные протоколы:

- IR60 (SDA2008/MC14497)
<команда>
- Matsushita (Panasonic MN6014, C5D6 / 11 бит)
<выборочный код>:<кодовые данные>
- NEC μ PD1986C
<кодовые данные>
- RECS80 (standard & extended)
<адрес>:<команда>
- Sanyo (LC7461)
<выборочный код >:<ключ>
- RCA
<адрес>:<команда>
- Thomson
<устройство>:<функция>

Несущая частота модуля приемника TSOP не обязательно должна точно соответствовать частоте RC. Несоответствие уменьшает возможный диапазон, но это не имеет большого значения для этого приложения.

Модуль приёмника ИК, соединенный с тест контактами

Пожалуйста, подключайте модуль приемника ИК только после входа в режим!

Как соединить модуль TSOP:

- Тестовый контакт No1: общий GND(земля)
- Тестовый контакт No2: Vs – питание (с ограничительным резистором на 680 Ω)
- Тестовый контакт No3: выход/данные

Подсказка: токоограничивающий резистор по Vs подразумевает применение модуля приемника ИК с диапазоном напряжения питания от 2.5 до 5 В. Если вы используете только 5В модуль, то можно отключить ограничительный резистор в config.h файле (на ваш собственный риск). **Любое короткое замыкание в схеме может повредить процессор!**

Аппаратный модуль приёмника ИК

Для аппаратного модуля приёмника ИК используемый порт процессора назначается и на-

страивается в файле конфигурации config_<MCU>.h.

3.1.17. ИК RC Передатчик Передатчик ИК RC отправляет ИК-коды, которые вы ввели и предназначен для проверки ИК RC- ресиверов или устройств с дистанционным управлением. Этот инструмент требует для работы дополнительные кнопки либо поворотный энкодер, дисплей с более чем 4 строками и несложную схему для управления ИК-светодиодом. На дисплее отображается протокол, несущая частота, скважность и нескольких полей для ввода данных. Коротким нажатием кнопки тестирования вы можете переключаться между пунктами. Выбранный элемент обозначается символом *. Используйте энкодер (или другой режим ввода), чтобы изменить настройку/значение пункта. Длительное нажатие кнопки "Тест" отправляет ИК-код до тех пор, пока вы держите кнопку нажатой. Как обычно, два коротких нажатия кнопки - выход из функции. При изменении протокола несущая частота и скважность устанавливаются в значения по умолчанию для протокола. Но вы можете изменить их, если хотите. Несущая частота может быть установлена от 30 до 56кГц, а скважность 1/2 (50%), 1/3 (33%) или 1/4 (25%). Поля данных представляют собой настраиваемые пользователем части ИК-кода и поясняются ниже. В большинстве случаев это всего лишь адрес и команда.

Поддерживаемые протоколы и поля данных:

- JVC
<адрес:8> <команда:8>
- Kaseikyo (Japanese Code)
< код производителя:16> <система:4> <продукт:8> <функция:8>
- Matsushita (Panasonic, MN6014 12 бит)
< пользовательский код:6> <данные:6>
- Motorola
<команда:9>
- Стандарт NEC
<адрес: 8> <команда: 8>
- NEC Extended
<адрес: 16)> <команда: 8>
- Proton / Mitsubishi (M50560)
<адрес:8> <команда:8>
- RC-5 Standard
<адрес:5> <команда:6>
- RC-6 Standard, Mode 0
<адрес:8> <команда:8>
- Samsung / Toshiba (32 бит)
<пользовательский код: 8> < данные: 8>
- Sharp / Denon
<адрес:5> <команда:8> <маска:1>
- Sony SIRC-12
<команда: 7> <адрес: 5>
- Sony SIRC-15
<команда: 7> <адрес: 8>
- Sony SIRC-20
<команда: 7> <адрес: 5> <дополнительный: 8>

Дополнительные протоколы (SW_IR_TX_EXTRA):

- Thomson
<устройство:4> <функция:7>

Поля данных разделяются пробелами, а их синтаксис:

- <наименование поля>: < битность>

Конфигурация выхода:

- Тестовый контакт No2:: выходной сигнал (с сопротивлением 680Ω для ограничения тока)

Тестовый контакт No1 и No3: GND(земля)

Выход сигнала (контакт No 2) имеет резистор ограничения тока и может управлять ИК-светодиодом с током не более 5mA, что недостаточно для ИК-светодиодов с типичным потреблением 100mA. Поэтому вам нужна простая схема 3.1 управления на основе переключающего транзистора, ИК-светодиода и резистора для ограничения тока. Пример схемы управления ИК-светодиодом (V_f 1,5 В, при токе 100 мА) с потреблением тока примерно 50mA:

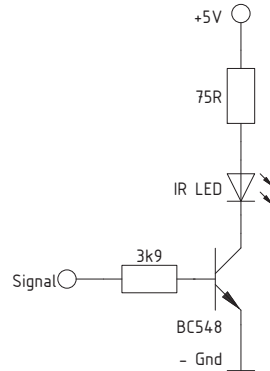


Рис. 3.1. пример 50mA ИК драйвер с (V_f 1.5В, I_f 100mA)

Подсказка: если длительность импульса/паузы является ошибочной, активируйте альтернативную циклическую задержку опцией SW_IR_TX_ALTDELAY. Она может потребоваться, если компилятор Си оптимизирует стандартный цикл задержки, несмотря на определенные команды для хранения ассемблерного кода.

3.1.18. Тестирование оптронов Тестирование оптронов Эта функция проверяет оптроны и показывает вам значения V_f LED, CTR (также I_f), задержки ton/toff (BJT тип). Она поддерживает стандартные оптроны с BJT транзисторами NPN структуры, NPN структуры Дарлингтона и оптосимисторы. Для измерения параметра "CTR" порт ввода-вывода процессора испытывает перегрузку, приблизительно, в течение 3 мс. Спецификация определяет максимальный выходной ток в 20mA, но мы перегружаем порт ввода-вывода приблизительно до 100mA. Поэтому, максимальное значение CTR ограничено и любой опtron со значением CTR выше 2000% нужно тестировать с осторожностью. Максимальный управляющий ток для LED, который нужен для определения оптосимистора - 5mA. В элементах с задержкой (MOSFET back to back) определение BJT и CTR будет бессмысленно. Тестирование оптронов со встречно включенными светодиодами проигнорировано. Для тестирования вам нужен простой адаптер со следующими 3 контрольными точками:

Тип – оптотранзистор BJT::

- Анод LED
- Катод LED и эмиттер BJT соединить вместе
- Коллектор BJT

Тип – оптосимистор:

- Анод LED
- Катод LED и MT1 оптосимистора соединить вместе
- MT2 оптосимистора

Можно подключить адаптер к любым 3 тестовым контактам прибора. Тестер определит расположение контактов автоматически. После входа в режим подключите адаптер и нажмите кратковременно кнопку "Тест" для начала тестирования оптрона. Если он будет найден, на дисплее появится тип оптрона и дополнительные параметры. Если компонент не обнаружен, на дисплее будет надпись "не определен". Мигающий курсор указывает на то, что вы должны нажать тестовую кнопку (или повернуть энкодер) для старта нового тестирования. Два коротких нажатия кнопки "Тест" выведут из режима, как обычно.

3.1.19. Тестирование сервоприводов RC Эта функция формирует ШИМ сигнал для сервоприводов RC, которые управляются 1- 2мс импульсами. Поддерживаются

фиксированные частоты ШИМ 50, 125, 250 и 333Гц, а длительность импульса может быть между 0.5 и 2.5мс. Есть также свип-режим с изменяемой длительностью импульса между 1 и 2мс с регулируемой скоростью перестройки. Вы можете скорректировать длительность импульса энкодером. По часовой стрелке для установки более длинного импульса, против часовой стрелки - для более короткого импульса. Длинное нажатие кнопки сбрасывает импульс к 1.5мс. Можно переключаться между режимами выбора длительности импульса и выбора частоты коротким нажатием кнопки "Тест" (отмечается звездочкой). В частотном режиме используется энкодер для выбора частоты ШИМ. Длинное нажатие кнопки "Тест" включает или отключает возможность перестройки (появляется символ «->» после частоты). Если этот режим включен - энкодер позволяет изменять период. Как обычно, два коротких нажатия кнопки "Тест" выход из функции.

Конфигурация выхода:

Тестовый контакт No2:: выход ШИМ (с резистором на 680Ω, ограничивающим ток) для ограничения тока)

Тестовый контакт No1 и No3: GND(земля)

Подсказка: вы должны обеспечить дополнительный источник питания для сервопривода. Некоторые схемы распиновки типичных 3-х контактных соединителей сервопривода:

Производитель	контакт 1	контакт 2	контакт 3
Эйртроникс	ШИМ белый/черный	Gnd черный	Vcc красный
Футаба	ШИМ белый	Vcc красный	Gnd черный
hitec	ШИМ желтый	Vcc красный	Gnd черный
Рапии JR	ШИМ оранжевый	Vcc красный	Gnd коричневый

Таблица 3.1. Назначение контактов для типичных 3-контактных серворазъемов

3.1.20. Сканирование 1-Wire шины 1-Wire сканирование определяет коды ПЗУ всех подключенных устройств. Пожалуйста, см. раздел «Шины и интерфейсы» для настройки шины 1-Wire. При использовании щупов тестер сообщит о назначении контактов и будет ждать, пока не обнаружит внешний подтягивающий резистор. Можете пропустить это нажатием кнопки “Тест”. Каждый раз, когда вы нажимаете кнопку тестирования, прибор будет сканировать текущее (следующее) подключенное устройство и отображать его ПЗУ код (в шестнадцатеричном формате). Первая часть кода – информация о производителе, вторая часть - его серийный номер. CRC опущен. Код производителя > = 0x80 (установлен бит 7) указывает на уникальный код клиента, а первые три (слева) шестнадцатеричные цифры серийного номера - это идентификатор клиента. Тестер сообщит о том, что найдено последнее устройство, а также выдаст информацию об ошибках CRC и шины. В случае завершения сканирования или ошибки шины вы можете начать новый процесс сканирования, нажав кнопку тестирования. [Для подключения смотрите страницу 12](#). И, как обычно, два коротких нажатия кнопки “Тест” - выход из режима.

3.1.21. Температурный датчик DS18B20 Эта функция проверяет температурный датчик 1-Wire DS18B20 и выводит на экран температуру. См. раздел "Шины и интерфейсы" для настройки шины 1-Wire. После запуска тестер сообщит вам назначение контактов и ожидает, что будет обнаружен внешний подтягивающий резистор. Можно пропустить это нажатием кнопки “Тест”. После подключения DS18B20 к тестовым контактам нажмите кнопку “Тест” ещё раз - датчик будет считан (это может занять почти секунду). Чтобы выйти из этого режима нажмите кнопку “Тест” дважды быстро. Длительным нажатием кнопки вы можете выбрать автоматический режим (автоматическое обновление), который будет обозначен знаком „*“ после наименования датчика.

Назначение контактов для датчиков:

Тестовый контакт No1: GND(земля)

Тестовый контакт No2: DQ (данные)

Тестовый контакт No3: Vcc (ток ограничен резистором 680Ω)

Требуется внешний подтягивающий резистор 4,7kΩ между DQ и Vcc!

3.1.22. Датчики температуры и влажности DHTxx Эта функция для тестирования DHT11, DHT22 и совместимых датчиков температуры и влажности. Сначала тестер подсказывает назначение контактов, а затем ожидает внешнего подтягивающего резистора. После обнаружения резистора, на дисплее отображается выбранный тип датчика (по умолчанию: DHT11) и короткое нажатие тестовой кнопки считывает его показания. При успешном прочтении прибор выводит измеренные значения, при любой ошибке результат будет равен $-$. Длительное нажатие кнопки изменяет тип датчика, а два коротких нажатия кнопки - выход из опции. При смене типа датчика у вас также есть возможность активировать его автоматическое считывание (каждую секунду), которое обозначается знаком "*" после типа датчика.

Поддерживаемые типы датчиков:

DHT11: DHT11, RHT01
DHT22: DHT22, RHT03, AM2302
DHT21, RHT02, AM2301, HM2301
DHT33, RHT04, AM2303
DHT44, RHT05

Назначение контактов для тестирования:

Тестовый контакт No1: GND(земля)
Тестовый контакт No2: Данные
Тестовый контакт No3: Питание Vdd (ток не ограничен)

Необходим внешний резистор подтяжки $4,7k\Omega$ между контактами No3 и No2!

Некоторые датчики уже включают в себя подтягивающий резистор $10k\Omega$, который работает также хорошо с короткими кабелями.

Подсказка: из-за требований к питанию датчика измерительный резистор 680Ω не может быть использован для ограничения тока.

Имейте в виду, что любое короткое замыкание может повредить MCU!

3.1.23. Самотестирование Если вы вошли в режим самотестирования, необходимо закоротить все три тестовых контакта. Тестер ожидает, пока это будет сделано. В случае любой проблемы можно прервать процедуру нажатием кнопки "Тест". Этим действием вы пропустите полную проверку. Функция самопроверки запускает каждый тест по 5 раз. Можно пропустить любой тест коротким нажатием кнопки или пропустить весь долгим нажатием. В тесте T4 нужно удалить "закоротку" тестовых контактов. Тестер будет ожидать, пока вы действительно не удалите короткое замыкание.

Описание тестов:

- T1 - проверка напряжения внутреннего ИОНа (в мВ)
- T2 - сравнение резисторов Rl (смещение в мВ)
- T3 - сравнение резисторов Rh (смещение в мВ)
- T4 - **удалить короткое замыкание тестовых контактов!**
- T5 - проверка контактов на утечку в режиме "pull-down" (напряжение в мВ)
- T6 - проверка контактов на утечку в режиме "pull-up" (напряжение в мВ)

3.1.24. Автокорректировка Автокорректировка измеряет сопротивление и ёмкость тестовых контактов или щупов, т.е. плата, печатные проводники и контакты образуют суммарное значение для создания нулевого смещения. Эта функция также измеряет внутреннее сопротивление тестового порта процессора в режиме ввода-вывода. Если эти тесты пропущены, или будут странные значения измеренных значений, то по умолчанию будут использованы параметры из config.h. Если все прошло нормально, тестер будет отображать и использовать новые значения, полученные при автокорректировке (они не будут автоматически сохранены в EEPROM, смотрите пункт "Сохранение/Загрузка" ниже). Смещение напряжения аналогового компаратора автоматически скорректировано при измерении ёмкости (в нормальном режиме тестирования без учёта автокорректировки), если конденсатор находится в диапазоне $100nF$ до $3.3\mu F$. Смещение для встроенного ИОНа определено таким же образом. Прежде, чем сделать автокорректировку в первый раз, измерьте калибровочный конденсатор со значением ёмкости между $100nF$ и $3.3\mu F$, по крайней мере

три раза, чтобы тестер смог скорректировать упомянутые выше смещения. Обычно, первое измерение приведет к пониженному значению ёмкости, второе – к повышенному, а третье будет наиболее точным. Обе величины смещения будут выведены на экран в конце процедуры. С аппаратным калибровочным конденсатором (только с ATmega644/1284) автоматическая обработка смещения при измерении емкости заменена специальной функцией, выполняемой во время процедуры корректировки. Таким образом, вы не должны замерять пленочный конденсатор дополнительно. Если смещения емкости различаются между парами тестовых контактов, можно включить проверку смещения пары в файле config.h. Раздел 6.2.5 начинается на странице 53 (CAP_MULTIOFFSET). Такая же поправка возможна для смещения величины сопротивления (R_MULTIOFFSET). Автокорректировка очень похожа на самотестирование по процедуре и пользовательскому интерфейсу.

Описание этапов корректировки:

- A1 - определение смещения для ИОНа и аналогового компаратора (только для опции с аппаратным конденсатором);
- A2 - определение сопротивления щупов/зажимов (от 10mΩ);
- A3 - **удалить короткое замыкание тестовых контактов/щупов!**;
- A4 - внутреннее сопротивление портов процессора по отношению к земле Gnd (напряжение RiL);
- A5 - внутреннее сопротивление портов процессора по отношению к питанию Vcc (напряжение RiH);
- A6 - определение ёмкости щупов/зажимов (в pF);

Ограничения:

- сопротивление тестовых контактов < 1,50 Ω для двух соединённых контактов;
- ёмкость тестовых контактов < 100 pF.
- Внутреннее сопротивление порта ввода-вывода в режиме (RiL) < 25 Ω
- Внутреннее сопротивление порта ввода-вывода в режиме (RiH) < 29 Ω

Подсказка: если сопротивление для тестовых щупов/зажимов изменяется слишком сильно - может быть проблема в контактах!

Помните: корректировка - это не калибровка! Калибровка - это процедура сравнения результатов измерений с известным измерительным стандартом и сохранения этого различия. Цель состоит в том, чтобы контролировать дрейф в течение долгого времени. Корректировка – это процедура для настройки устройства согласно определенным задачам.

3.1.25. Сохранение/Загрузка После выполнения корректировки вы можете обновить константы настроек в EEPROM по умолчанию, используя режим “Save”(Сохранение). В следующий раз, когда вы включите прибор, обновленные значения (профиль # 1) будут загружены и использованы автоматически. Для удобства можно сохранить и загрузить два профиля, например, если используется два набора различных тестовых контактов или зажимов. Идея режима сохранения состоит в том, чтобы предотвратить автоматическую запись данных корректировки. Если нужно использовать другие зажимы для некоторых измерений, тестер будет скорректирован для работы с этими временными зажимами. Если вернуться назад к стандартным контактам, прибор не должен приспосабливаться, потому что старые значения всё еще сохранены. Просто выключите и включите питание тестера. Есть опция (UI_CHOOSE_PROFILE) для автоматического входа в меню загрузки профиля после включения тестера.

3.1.26. Информация о корректировке Выводит на экран текущие значения корректировок и используемые смещения. Ис- пользование внешнего 2.5-вольтового источника опорного напряжения обозначено символом „*“ после значения Vcc.

3.1.27. Шрифт Отображение шрифта в тестовых целях.

3.1.28. Выключение Эта функция выключит тестер, если активна настройка SW_POWER_OFF на странице 52.

3.1.29. Выход Если вы вошли в меню по ошибке, можно выйти из него этой командой.

4.1. Резисторы

Резисторы измеряются дважды (в обоих направлениях) и значения сравниваются. Если значения отличаются намного, тестер предполагает, что есть два резистора вместо одного. В этом случае прибор выводит на экран результат как два резистора с номерами контактов "1 - 2 - 1 и отличающимися двумя значениями сопротивления. Для резисторов с сопротивлением ниже, чем 10 Ом выполняется дополнительное измерение с более высоким разрешением. В некоторых, редких случаях, прибор не может обнаружить очень малое сопротивление. Тогда повторно выполните тест.

Если включена дополнительная проверка значений норм ряда E (SW_R_E *), тестер берет следующее более низкое и следующее более высокое значение нормы и сравнивает их с измеренным сопротивлением и с учетом допусков компонентов. Есть два режима вывода. В текстовом режиме прибор отображает ряд E и применяемый допуск, сопровождаемый соответствующими значениями нормы. Символ «-» означает, что нет соответствия значению нормы. В режиме вывода цветового кода тестер отображает ряд E и цветовой код резистора, включая цветовую полосу для допуска. Помните, что цвета могут отличаться в зависимости от модуля дисплея и используемого цветового режима.

Если какой-либо цвет отключен, просто отрегулируйте значение цвета (COLOR_CODE_*) в файле colors.h. Поиск в Интернете по запросу "RGB565 tool" покажет много онлайн инструментов для создания / выбора значений цвета RGB565.

4.2. Конденсаторы

Для измерения ёмкости используется три метода. Конденсаторы ёмкостью $> 47\mu\text{F}$ измеряются методом зарядки импульсами длительностью 10 мс. Конденсаторы в диапазоне от $4.7\mu\text{F}$ и $47\mu\text{F}$ тестируются таким же образом, но с зарядным импульсом длительностью 1 мс. Конденсаторы с малой ёмкостью тестируются методом аналогового компаратора. Таким образом, точность измерения различных конденсаторов оптимизирована.

Большие ёмкости требуют коррекции. Без исправления измеренные значения слишком большие. По моему скромному мнению, это вызвано методом измерения. Начиная с преобразования ADC после зарядки импульсом, требуется некоторое время для того, чтобы конденсатор потерял некоторый заряд из-за внутренних резистивных потерь. Также самому преобразователю ADC требуется некоторый заряд для работы. Из-за этого процесса необходимо больше времени для зарядки электролита и у конденсатора измеряется большее значение ёмкости. Измерение разряда затем пытается компенсировать этот эффект, но оно может сделать это только частично. Поправочные коэффициенты (CAP_FACTOR_SMALL, CAP_FACTOR_MID и CAP_FACTOR_LARGE в файле config.h) подобраны для работы с большинством моделей тестеров. В некоторых случаях вам, возможно, придется их изменить. Была добавлена логика при измерении больших ёмкостей, которые могут быть определены как резисторы. Резисторы $< 10\Omega$ дополнительно тестируются на предмет того, не являются ли они большими ёмкостями. Нижний диапазон измерения ёмкостей начинается от $5\mu\text{F}$ (включая нулевое смещение), и эти значения допустимы. Более низкие значения слишком сомнительны и могут быть вызваны расположением щупов (тестовых контактов) и будут немного отличаться.

Тестер пытается измерить ESR для конденсаторов с номиналом от 10 нФ и выше. Альтернативно можно включить старый метода замера ESR конденсаторов с номиналом от 180 нФ и выше. Но так как измерение ESR не сделано сигналом переменного тока с определённой частотой, не ожидайте стабильных результатов. Используемый метод может быть сопоставим с тестом на частоте 1 кГц. Так или иначе, результаты достаточно хороши при проверке

электролитических конденсаторов. Для небольших емкостей вы можете получить различные результаты в зависимости от тактовой частоты процессора. Я предположил бы, что г-н Фурье в состоянии объяснить это. Другие проведенные измерения определяют ток утечки (саморазряда) для конденсаторов с ёмкостью более чем $4.7\mu\text{F}$. Это дает подсказку о состоянии электролитического конденсатора. В моих тестах типичное значение саморазряда для исправных электролитов составило:

- 10-220 μF	1-3 μA
- 330-470 μF	4-5 μA
- 470-820 μF	4-7 μA
- > 1000 μF	5-7 μA на каждые 1000 μF

Дополнительная проверка значений норм ряда E также доступна для конденсаторов (SW_C_E*), но только в текстовом режиме, потому что существует слишком много разных вариантов цветовых кодов.

4.3. Индуктивности

Измерения индуктивности не очень точны, так как частота работы процессора, конструкция печатной платы тестера оказывают влияние на результаты. Базовый метод основан на измерении времени между подачей электрического тока и достижением его определенного уровня. Для больших индуктивностей есть проверка небольшим током, для малых индуктивностей нужна проверка большим током, который превышает нагрузочный предел порта MCU в течение очень короткого времени (приблизительно до 25 микросекунд). При исследовании эффекта изменения частоты работы MCU и других вещей я нашел образец отклонений, который может использоваться для компенсации. Поэтому добавлена возможность пользовательской тонкой настройки.

В файле `inductor.c` в функцию "MeasureInductor" введена переменная "Offset" для компенсации. Эта переменная - смещение для опорного напряжения. Положительное смещение будет уменьшать значение индуктивности, отрицательное – увеличит его. Компенсация при проверке большим током зависит от частоты работы MCU и разделена на три отдельных диапазона, каждый со специальным смещением. При тесте небольшим током - только простая компенсация в данный момент, необходимы дополнительные тесты. Если вы видите какие-либо большие отклонения результатов измерений при сравнении с образцовым LCR - измерителем, можно скорректировать смещения, чтобы получить более точные параметры. Если вы хотите проверить значения норм ряда E, включите в `config.h` опцию (SW_L_E*) (только текстовый режим).

Подсказка: при получении неожиданных результатов повторно запустите тест. Измерение индуктивности выполняется только если её активное сопротивление менее $2k\Omega$.

4.4. Разряд компонентов

Тестер пытается разрядить любой подключенный компонент до и во время измерения. Когда он не может разрядить компонент ниже указанного порога (CAP_DISCHARGED), то выдаст ошибку, отображающую номер т. контакта и оставшееся напряжение.

Режим разряда основан не на фиксированной паузе, он адаптирует себя к разрядному уровню. Таким образом, батарея будет идентифицирована быстрее (около 2с), а конденсаторы большой ёмкости будут разряжаться дольше. Если такой конденсатор будет определён как батарея, пожалуйста, повторите проверку. При помехе или ошибке, возможно, понадобится скорректировать параметр CAP_DISCHARGED приблизительно до 3 мВ. Индикация остаточного напряжения поможет вам выбрать нужное значение.

4.5. ADC супердискретизация

Функция ADC изменена, чтобы поддерживать супердискретизацию переменной (1-255 отсчётов). Значение по умолчанию – 25 выборки. Можно попытаться улучшить точность измерения, увеличивая число выборок. Обратите внимание на то, что большое число выборок будет занимать больше времени, приводя к более медленным измерениям.

4.6. Отображение результатов на дисплее

Изменены некоторые обозначения и применены сокращения. Отображение информации может быть разделено на несколько частей и использовать многостраничный режим для того, чтобы поддерживать дисплей всего с несколькими строками. Для одного диода низкое значение действующего V_f (измеренное током $10\mu\text{A}$) показано в фигурных скобках, если напряжение ниже 250mV . Это должно подсказать вам, что это, возможно, германиевый диод. Большинство таблиц спецификации для германиевых диодов определяет V_f при токе в 0.1mA , который тестер не поддерживает. При более высоком действующем V_f , как правило, будет приблизительно 0.7V и трудно отличить германий от кремниевых диодов. Ток утечки I_R для одного диода или I_{CEO} для ВJT будет выведен на экран, если он превышает 50nA . Германиевые транзисторы ВJTs имеют ток утечки от нескольких μA до, приблизительно, $500\mu\text{A}$. Германиевые диоды обычно имеют ток утечки в несколько μA . Для некоторых элементов также отображается ёмкость. В случае если ёмкость ниже 5pF или измерение приводит к сбою по некоторым причинам, выведенное на экран значение будет 0pF . Если обнаружен FET транзистор со встроенным каналом, симметричными стоком и истоком, например, JFET, схема контактов показывает 'x' вместо 'D' или 'S', потому что оба вывода могут быть функционально идентичными. Посмотрите спецификацию на FETs, если вам нужно больше информации о цоколёвке. Цоколёвка для симистора показана как IDs 'G', '1' и '2'. '1' - это вывод MT1, '2' - MT2. Для однопереходных UJT транзисторов, в случае, если их детектирование включено, '1' - база B1, '2' - база B2 и 'E' - эмиттер. Если включена нестандартная опция определения цоколёвки (выбран SW_SYMBOLS в config.h), будет показан соответствующий символ с соответствующей распиновкой для 3-х контактных полупроводников. Если на дисплее недостаточно пространства для символа, его вывод будет пропущен.

4.7. Дополнительные подсказки

ВJTс Строчная буква после значения hFE указывает тип тестовой цепи, которая используется для измерения hFE:

- e: схема с общим эмиттером
- c: схема с общим коллектором

Если вы включили вывод тестового тока hFE (SW_HFE_CURRENT), то тестер отобразит I_C для схемы с общим эмиттером и I_E для схемы с общим коллектором.

При проверке диодов V_f измеряется с R_l (высокий испытательный ток) и R_h (низкий испытательный ток), и оба напряжения сохраняются. Функция вывода для ВJT ищет соответствующий диод для V_{BE} и интерполирует два измерения V_f на основе транзисторов hFE для виртуального тестового тока. Так мы получим более подходящие результаты для различных типов транзисторов, так как V_{BE} для слаботочного ВJT не измеряется таким же испытательным током, что и для мощного ВJT.

Для ВJTс с внутренним резистором (база-эмиттер) тестер отобразит на экране этот резистор и пропустит измерение hFE. Если у ВJTс есть встроенный защитный диод, такой транзистор может быть обнаружен как ВJT или как два диода вместе с резистором база-эмиттер (низкоомный резистор - 2 диода). Тогда тестер покажет эти два диода и резистор с намеком на возможную NPN - структуру или PNP ВJT. Проблема состоит в том, что низкоомный встроенный резистор мешает нормальной идентификации ВJT.

Другой особый случай - ВJT с интегрированным защитным диодом той же структуры что и ВJT. Этот диод создает эффект "паразитного" транзистора. У ВJT NPN-структуры будет "паразитный" PNP и наоборот. Такой транзистор будет помечен символом + после типа ВJT. Для транзистора Шоттки (ВJT с барьером Шоттки) ограничивающий диод между базой и коллектором и V_f определяются, если детектирование включено (SW_SCHOTTKY_VJT) См. страницу 49. Обратите внимание, что транзистор Шоттки имеет повышенный I_{CEO} .

Симисторы могут использоваться в трех или четырех различных режимах работы, также известных как квадрант. Обычно некоторые параметры будут отличаться для каждого квадранта, например, ток срабатывания затвора (I_{GT}). В некоторых случаях возможно, что тестовый ток достаточен для срабатывания затвора в одном квадранте, но не в другом. Так как два тестовых прогона необходимы, чтобы вычислить контакты для MT1 и MT2 тестер не сможет определить различие между ними, т.е. контакты могли поменяться местами. Вы так же можете найти симисторы, которые имеют слишком высокий ток удержания (I_H), препятствующий их правильному обнаружению. Если симистор имеет слишком высокий ток срабатывания затвора - тестер обычно обнаруживает только резистор.

Диоды CLD Диодная проверка идентифицирует CLD (диод со стабилизацией тока) как стандартный диод и отображает I_F как ток утечки. Обратите внимание на то, что анод и катод у CLD инвертированы по сравнению со стандартными диодами. Специальная проверка на CLD сложна в реализации, начиная с определения тока утечки для германиевого или высоковольтного диода Шоттки, находящегося в диапазоне I_F ($\geq 33\mu A$). Если у диода есть необычное прямое падение напряжения, довольно низкий V_f при небольшом тестовом токе (2-е значение в фигурных скобках) и емкость совсем не измерена, тогда, скорее всего, это CLD.

Неподдерживаемые компоненты Любой полупроводник, требующий высокого напряжения или большого тока переключения, не может правильно определиться, так как тестер обеспечивает максимальный ток до 7mA и максимальное напряжение 5В. Таким образом, такие компоненты как DIAC с V_{BO} 20-200V невозможно проверить. То же самое для тиристорov и симисторov с большим током запуска.

Известные проблемы и способы их решения Если у вашего тестера есть одна из следующих проблем, вы можете попробовать включить обходной путь:

Проблема:

- hFE слишком высокое.

При использовании схемы с общим коллектором с R1 в качестве базового резистора, базовое напряжение по неизвестной причине измеряется как очень малое. Так появляется базовый ток, который вызывает слишком высокое значение hFE. Тестеры, у которых замечена эта проблема - Hiland M644!

Обходной путь - включите настройку "NO_HFE_C_RL" в config.h См. страницу 50!

- Конденсаторы (серия Panasonic NF) могут быть обнаружены как диод или два встречных диода. Измерение емкости также не даёт определить приемлемое значение.

- При использовании SMPS или преобразователя DC-DC в качестве источника питания, тестер иногда обнаруживает конденсатор емкостью 50µF, даже если не подключено никакого компонента.

- ESR у конденсаторов небольшой емкости может меняться в зависимости от частоты работы процессора.

- Проблема с измерением ESR у твердотельных LowESR электролитических конденсаторов.

4.8. Полезные ресурсы

[10] - Немецкий форум

<https://www.mikrocontroller.net/topic/248078>

Основной язык форума - немецкий, но и английский также возможен.

[11] - Английский форум

[https://www.eevblog.com/forum/testgear/\\$20-lcr-esr-transistor-checker-project/](https://www.eevblog.com/forum/testgear/$20-lcr-esr-transistor-checker-project/)

Основной язык форума - только английский.

[12] - Российский форум

<https://vrtp.ru/index.php?showtopic=16451&ST=0>

<https://vrtp.ru/index.php?showtopic=25020&st=0>

- Информация о различных клонах, например, фотографии, схемы и прошивки

[13] - Российская сторона (von indman@EEVblog)

<https://disk.yandex.ru/d/yW8xa5NJgUo5z>

- Информация о различных клонах, например, фотографии, схемы и прошивки

[14] - Руководство по компиляции, файлы для установки Winavr (indman@eevblog)

https://drive.google.com/file/d/1-IJA8uTcsCA_6SYHEuMydjfS2vNgmwdH/edit

[15] Актуальное k-firmware

<https://github.com/kubi48/TransistorTester-source>

[16] Актуальная документация для k-firmware

<https://github.com/kubi48/TransistorTester-documentation>

4.9. Журнал изменений

Вы можете найти его в главе 9 со страницы 81.

5.1. Команды управления

Когда тестер принимает команды управления, ответ может содержать следующие текстовые строки, кроме командных ответов, содержащих данные:

- 5.1.1. ERR** - неизвестная команда;
- команда не поддерживается в контексте компонента;
- переполнение буфера.

- 5.1.2. ОК** - команда выполнена
(некоторым командам может потребоваться некоторое время для обработки).

- 5.1.3. N/A** - информация / значение не определено.

Ответ с данными никогда не отображается ни с одной из стандартных текстовых строк выше, чтобы предотвратить любую возможную путаницу.

5.2. Основные команды:

- 5.2.1. VER** - определяет версию прошивки;
- проверить подключение и определить набор команд на основе версии;
- пример ответа: «1.33m».

- 5.2.2. OFF** - выключает тестер;
- тестер отвечает «ОК» перед отключением питания;
- пример ответа: «ОК» < тестер отключен>.

5.3. Команды тестирования:

- 5.3.1. PROBE** - тестирует т.контакты и пропускает любые паузы, ожидающие обратной связи с пользователем;
- тестер отвечает «ОК» после завершения проверки;
- пример ответа: < истекает время для тестирования> «ОК».

- 5.3.2. COMP** - определяет идентификатор типа компонента
(см. COMP_* в файле common.h для идентификаторов);
- пример ответа для ВJT: «30».

- 5.3.3. MSG** - возвращает сообщение об ошибке;
- применяется только при возникновении ошибки (COMP: 1);
- ответ может отличаться в зависимости от языка пользовательского интерфейса;
- пример ответа: «Батарея? 1:1500 мВ».

- 5.3.4. QTY** - определяет количество компонентов;
- пример ответа для ВJT: «1»

- 5.3.5. NEXT** - выбор следующего компонента;
- применяется, если найдены два компонента (QTY: 2);
- пример ответа: «ОК».

5.3.6. TYPE

- определяет более конкретный тип компонента
- применяется к BJT, FET и IGBT
- доступные типы:
 - NPN NPN (BJT)
 - PNP PNP (BJT)
 - JFET JFET (FET)
 - МОП -транзистор (полевой транзистор)
 - N-канальный n-канал (FET, IGBT)
 - P-ch p-канал (FET, IGBT)
 - enh. - режим обогащения(FET, IGBT);
 - dep. - режим истощения (FET, IGBT)
- пример ответа для BJT: "NPN"
- пример ответа для FET (MOSFET): «MOSFET n-ch enh»

5.3.7. HINT

- отображает подсказки по специальным функциям;
- применяется для диодов, BJT, FET и IGBT;
- доступные подсказки:
 - NPN возможно NPN BJT (диод)
 - PNP возможно PNP BJT (диод)
 - резистор в базе -эмиттер R_BE (диод, BJT)
 - BJT+ встроенный flyback диод на той же подложке создающий 2-й "паразитный"BJT (BJT)
 - D_FB встроенный / flyback диод (BJT, FET, IGBT)
 - D_CLAMP с барьером Шоттки (BJT)
требуется включения обнаружения транзистора Шоттки.
 - SYM симметричный сток и исток (FET)
- пример ответа для BJT: «D_FB R_BE»;
- пример ответа для FET (MOSFET): «D_FB»

5.3.8. MHINT

- возвращает подсказки по измерениям;
- относится к BJT;
 - h_FE_e - измерение h_FE выполняется с помощью схемы с общим эмиттером (BJT)
 - h_FE_c - измерение h_FE выполняется по схеме с общим коллектором (BJT)
- пример ответа для BJT: "h_FE_e"

5.3.9. PIN

- определяет цоколёвку компонента
- используемые идентификаторы:

- резистор	x = подключен,		- = не подключен
- конденсатор	x = подключен,		- = не подключен
- диод	A = анод,	C = катод,	- = nicht verbunden
- BJT	B = база,	C = коллектор,	E = эмиттер
- FET	G = затвор,	S = исток,	D =сток, x = сток/ исток
- IGBT	G = затвор,	C = коллектор,	E = эмиттер
- SCR	G = затвор,	A = анод,	C = катод
- TRIAC	G = затвор,	2 = MT2,	1 = MT1
- PUT	G = затвор,	A = анод,	C = катод
- UJT	E = эмиттер,	2 = B2,	1 = B1
- формат ответа:
<идентификатор зонда No 1> <идентификатор зонда No 2> <идентификатор зонда No 3>
- пример ответа для резистора: "xx-";
- пример ответа для диода: «C-A»;
- пример ответа для BJT: «EBC».

- 5.3.10. R** - определяет значение сопротивления;
- применяется к резистору (содержит индуктивность);
- пример ответа: «122R».
- 5.3.11. C** - определяет значение емкости;
- применяется к конденсатору;
- пример ответов: "98nF462uF".
- 5.3.12. L** - определяет значение индуктивности;
- применяется к резистору (содержит индуктивность);
- пример ответа: «115uH».
- 5.3.13. ESR** - определяет значение ESR (эквивалентное последовательное сопротивление);
- требует включения измерения ESR;
- применяется к конденсатору;
- пример ответа: "0.21R".
- 5.3.14. I_I** - определяет значение I_{leak} (эквивалентный ток утечки собственного разряда);
- применяется к конденсатору;
- пример ответа: «3.25uA».
- 5.3.15. V_F** - определяет значение V_F (прямое напряжение);
- применяется к диоду и PUT;
- также применяется к встроенному диоду MOSFET и обратноточному диоду ВJT или IGBT;
- пример ответа: «654mV».
- 5.3.16. V_{F2}** - определяет значение V_F слаботоочного измерения (прямое напряжение);
- применяется к диоду;
- пример ответа: «387mV».
- 5.3.17. C_D** - определяет значение C_D (емкость диода);
- применяется к диоду;
- пример ответа: "8pF".
- 5.3.18. I_R** - определяет значение I_R (обратный ток);
- применяется к диоду;
- пример ответа: «4.89uA».
- 5.3.19. R_{BE}** - определяет значение R_{BE} (резистор база-эмиттер);
- применяется к диоду и ВJT;
- пример ответов: "38.2R5171R".
- 5.3.20. h_{FE}** - определяет значение h_{FE} (усиление постоянного тока);
- соответствует ВJT;
- пример ответа: «234».
- 5.3.21. h_{FE_r}** - определяет обратное значение h_{FE} (коллектор и эмиттер перевернуты);
- соответствует ВJT;
- пример ответа: «23».
- 5.3.22. I_C** - определяет тестовый ток I_C для измерения hFE;
- требует вывода тестового тока для включения измерения hFE;
- измерение hFE для схемы с общим эмиттером;
- соответствует ВJT;
- пример ответа: «3245uA».

- 5.3.23. I_E** - определяет тестовый ток I_E для измерения hFE;
- требует вывода тестового тока для включения измерения hFE;
- измерение hFE для схемы с общим коллектором;
- соответствует ВJT;
- пример ответа: «3245uA».
- 5.3.24. V_BE** - определяет значение V_BE (напряжение база-эмиттер);
- соответствует ВJT;
- пример ответа: «657мВ».
- 5.3.25. I_CEO** - определяет значение I_CEO
(ток коллектор-эмиттер, открытая база);
- соответствует ВJT;
- пример ответа: «460.0uA».
- 5.3.26. V_th** - определяет значение V_th (пороговое напряжение);
- применяется к FET (MOSFET) и IGBT;
- пример ответа: «2959мВ».
- 5.3.27. C_GS** - определяет значение C_GS (емкость затвора);
- применяется к FET (MOSFET);
- пример ответа: «3200pF».
- 5.3.28. R_DS** - определяет значение R_DS_on
(сопротивление сток-исток);
- применяется к FET (MOSFET);
- пример ответа: «1.20R»
- 5.3.29. V_GS_off** - возвращает значение V_GS (off)
(напряжение отсечки);
- относится к FET (режим истощения);
- пример ответа: «-3072мВ».
- 5.3.30. I_DSS** - определяет значение I_DSS
(ток истока-стока, нулевое смещение / закороченный затвор);
- применяется к FET (режим истощения);
- пример ответа: «6430uA».
- 5.3.31. C_GE** - определяет значение C_GE (емкость затвор-эмиттер);
- соответствует IGBT;
- пример ответа: "724pF".
- 5.3.32. V_GT** - определяет значение V_GT (напряжение триггера затвора);
- применяется к SCR и TRIAC;
- пример ответа: «865мВ».
- 5.3.33. V_T** - определяет значение V_T (напряжение смещения);
- применяется к PUT;
- пример ответа: «699мВ».
- 5.3.34. R_BB** - определяет значение R_BB (сопротивление между базами);
- требует включения UJT для определения;
- применяется к UJT;
- пример ответа: «4758R».
- 5.3.35. V_Z** - возвращает значение V_Z (Zener или внешнее напряжение);
- требуется проверка ZENER во время обычного тестирования для включения;
- относится к Zener Diode или внешнему напряжению (опция оборудования);
- пример ответа: «6750 мВ»

Как уже упоминалось, встроенное программное обеспечение может быть адаптировано для различных тестеров и дополнительных функций. Для этого есть некоторые настройки в Makefile, в config.h и config_<MCU>.h. В этой главе объясняются настройки. Makefile управляет компиляцией исходного кода и содержит такие основные вещи, как содержит основные сведения, такие как тип MCU и программатор ISP. В файле config.h содержит общие настройки для работы и функций. И config_<MCU>.h отвечает за аппаратный уровень, т.е. за модули LCD и назначение выводов.

6.1. Makefile

В Makefile настройки выполняются путем установки определенных переменных. Для регулировки просто измените значение или строку после переменной. Для некоторых переменных существуют несколько предложений, которые комментируются символом #. Там, пожалуйста, прокомментируйте желаемое настройки (удалить #), и при необходимости закомментируйте настройки по умолчанию (вставить #).

6.1.1. Тип MCU

```
# avr-gcc: MCU model
# - ATmega 328/328P      : atmega328
# - ATmega 328PB       : atmega328pb
# - ATmega 324P/324PA  : atmega324p
# - ATmega 640         : atmega640
# - ATmega 644/644P/644PA : atmega644
# - ATmega 1280        : atmega1280
# - ATmega 1284/1284P  : atmega1284
# - ATmega 2560        : atmega2560
MCU = atmega328
```

Листинг 6.1. Выбранное слово - atmega328

6.1.2. Тактовая частота MCU

```
# MCU frequency:
# - 1MHz : 1
# - 8MHz : 8
# - 16MHz : 16
# - 20MHz : 20
FREQ = 8
```

Листинг 6.2. Выбранное слово - 8 МГц

6.1.3. Тип осциллятора

```
# oscillator type
# - internal RC oscillator      : RC
# - external full swing crystal : Crystal
# - external low power crystal : LowPower
OSCILLATOR = Crystal
```

Листинг 6.3. Слово - Кристалл

6.1.4. Тип MCU Avrdude

```
# avrdude: part number of MCU
# - ATmega 328 : m328
# - ATmega 328P : m328p
# - ATmega 328PB : m328pb
# - ATmega 324P : m324p
# - ATmega 324PA : m324pa
# - ATmega 640 : m640
# - ATmega 644 : m644
# - ATmega 644P : m644p
# - ATmega 644PA : m644p
# - ATmega 1280 : m1280
# - ATmega 1284 : m1284
# - ATmega 1284P : m1284p
# - ATmega 2560 : m2560
PARTNO = m328p
```

Листинг 6.4. Выбранное слово - m328p

6.1.5. Avrdude ISP программист

С программатором необходимы:

Имя; BitClock и Порт.

```
# Arduino as ISP
#PROGRAMMER = stk500v1
#PORT = /dev/ttyACM0
#OPTIONS = -b 19200

# Bus Pirate
#PROGRAMMER = buspirate
#PORT = /dev/bus_pirate
#OPTIONS = -B 10.0

# Diamex ALL-AVR/AVR-Prog
PROGRAMMER = avrispmkII
PORT = usb
OPTIONS = -B 1.0

# Pololu USB AVR Programmer
#PROGRAMMER = stk500v2
#PORT = /dev/ttyACM0
#OPTIONS = -B 1.0

# USBasp
#PROGRAMMER = usbasp
#PORT = usb
#OPTIONS = -B 20

# USBtinyISP
#PROGRAMMER = usbtiny
#PORT = usb
#OPTIONS = -B 5.0

# Arduino Uno bootloader via serial/USB
#PROGRAMMER = arduino
#PORT = /dev/ttyACM0
#OPTIONS = -D -b 115200

# Arduino Mega2560 bootloader via serial/USB
#PROGRAMMER = wiring
#PORT = /dev/ttyACM0
#OPTIONS = -D -b 115200
```

Листинг 6.5. Выбранный это Диамекс

Если вашего программатора нет в списке, введите соответствующие значения. Для получения дополнительной информации см. руководство Avrdude или онлайн-документацию [9].

6.2. конфигурация config.h

Этот файл используется для настройки работы и функций. Поскольку это обычный заголовочный файл языка Си, хорошо известные используются известные правила комментирования для Си. Чтобы активировать что-либо, удалите "//" в начале строки. На деактивировать в начало строки вставляется "//". Некоторые настройки требуют числового значения, которое при необходимости нужно отрегулировать. должны быть отрегулированы.

6.2.1. Эксплуатация оборудования

поворотный регулятор для работы

- Стандартные контакты: PD2 & PD3 (ATmega 328)
- может быть параллельно с модулем LCD
- см. ENCODER_PORT для выводов порта (config_<MCU>.h)
#define HW_ENCODER - комментарий для деактивации

количество импульсов серого кода на ступеньку или фиксатор

- Импульс поворотного энкодера - это полная последовательность из 4 импульсов кодов Грея.
- типичные значения: 2 или 4, редко 1
#define ENCODER_PULSES ... 4 - Настройте с вашим поворотным энкодером

количество фиксаторов или ступеней

- используется для измерения скорости вращения вращающегося датчика
- не обязательно должны точно совпадать и позволяют вам точно настроить (большее значение: медленнее, меньшее значение: быстрее)
- типичные значения: 20, 24 или 30
#define ENCODER_STEPS ... 24 - Настройте с вашим поворотным энкодером

больше/меньше клавиш для управления

- альтернатива поворотному энкодеру
- см. KEY_PORT для выводов порта (config_<MCU>.h)
//#define HW_INCDEC_KEYS - прокомментировать, чтобы активировать

опорное напряжение 2,5 В для проверки Vcc

- Стандартный вывод: PC4 (ATmega 328)
- должен быть как минимум в 10 раз точнее, чем регулятор напряжения.
- см. TP_REF для вывода порта (config_<MCU>.h)
- если необходимо, отрегулируйте UREF_25 ниже для вашего опорного напряжения.
//#define HW_REF25 - прокомментировать, чтобы активировать

Типичное напряжение опорного напряжения 2,5 В (в мВ)

- см. спецификацию справочника напряжений
- или возьмите $\geq 5,5$ разрядный DMM для измерения напряжения.
#define UREF_25 ... 2495 / -Измените значение, если необходимо

Защитное реле для разрядки конденсаторов

- Стандартный вывод: PC4 (ATmega 328)
- Низкий сигнал: короткое замыкание тестовых контактов.
Высокий сигнал через внешнее опорное устройство: короткое замыкание.
//#define HW_DISCHARGE_RELAY - прокомментировать, чтобы активировать

Измерение напряжения до 50 В постоянного тока / тест диодов Зенера

- Стандартный контакт: PC3 (ATmega 328)
- Делитель напряжения 10:1
- для диодов Зенера
- DC-DC повышающий преобразователь, управляемый тестовой кнопкой
- См. TP_ZENER для вывода порта в config_<MCU>.h
//#define HW_ZENER - прокомментировать, чтобы активировать

нестандартный делитель напряжения для теста Зенера

- Стандартный делитель напряжения 10:1
- ZENER_R1: верхний резистор в Ω
- ZENER_R2: нижний резистор в Ω
- //#define ZENER_DIVIDER_CUSTOM - прокомментировать, чтобы активировать
- #define ZENER_R1 180000 -Измените значение, если необходимо
- #define ZENER_R2 20000 -Измените значение, если необходимо

альтернативный режим проверки Зенера: не включать повышающий преобразователь

- если повышающий преобразователь DC-DC работает постоянно
- при измерении внешнего напряжения (схема без повышающего преобразователя)
- //#define ZENER_UNSWITCHED - прокомментировать, чтобы активировать

Проверка генератора при нормальном отборе проб

- требует, чтобы повышающий преобразователь работал постоянно (ZENER_UNSWITCHED)
- Напряжения min/max предназначены для определения действующего напряжения Зенера. Минимальное напряжение должно быть выше уровня шума, а максимальное напряжение должно быть ниже выходного напряжения. напряжение должно быть ниже, чем выходное напряжение повышающего преобразователя.
- //#define HW_PROBE_ZENER - прокомментировать, чтобы активировать
- #define ZENER_VOLTAGE_MIN 1000 /* min. voltage in mV */ - изменение
- #define ZENER_VOLTAGE_MAX 30000 /* max. voltage in mV */ - изменение

Выделенный выход сигнала

- если вывод OC1B MCU подключен как выделенный выход сигнала вместо Резистор R1 от тестового контакта 2.
- //#define HW_FIXED_SIGNAL_OUTPUT - прокомментировать, чтобы активировать

Простой счетчик частоты

- Стандартный вывод: T0 (PD4 ATmega 328) непосредственно как вход частоты
- отсчитывает до 1/4 тактовой частоты микроконтроллера.
- Возможно, параллельно с ЖК-модулем.
- //#define HW_FREQ_COUNTER_BASIC - прокомментировать, чтобы активировать

Расширенная частота счетчика

- Кристаллические генераторы низкой и высокой частоты и буферизованный ввод частоты
- Прескалер 1:1 и 16:1 (32:1)
- см. COUNTER_PORT для выводов порта (config_<MCU>.h)
- требуется дисплей с более чем 2 текстовыми строками.
- установить настройку прескалера схемы: 16:1 или 32:1.
- #define FREQ_COUNTER_PRESCALER ... 16 / 16:1 / - выбранный
- //#define FREQ_COUNTER_PRESCALER ... 32 / 32:1 /

счетчик событий

- Стандартный контакт: T0 (PD4 ATmega 328)
- использует T0 непосредственно как вход события/импульса (нарастающий фронт)
- общая работа с дисплеями для T0 невозможна.
- требует дополнительных клавиш (например, поворотный энкодер) и дисплея более 5 строк
- только для тактовой частоты MCU 8, 16 или 20 МГц.

`/// define HW_EVENT_COUNTER - прокомментировать, чтобы активировать`

выход счетчика событий триггера

- Использует пин # 2 как выход триггера, пины # 1 и # 3 - Gnd
- устанавливает высокий уровень на выходе триггера во время счета.

`/// define EVENT_COUNTER_TRIGGER_OUT - Включить`

ИК-детектор/декодер (через специальный контакт MCU)

- требуется модуль ИК-приемника, например, серии TSOP.
- модуль подключен к выделенному контакту ввода/вывода.
- см. IR_PORT для вывода порта (`config_<MCU>.h`)
- для дополнительных протоколов включите SW_IR_RX_EXTRA

`/// define HW_IR_RECEIVER - прокомментировать, чтобы активировать`

Фиксированный конденсатор для самонастройки

- См. TP_CAP и ADJUST_PORT для выводов порта (`config_<MCU>.h`)

`/// define HW_ADJUST_CAP - прокомментировать, чтобы активировать`

Счетчик L/C

- Использует T0 непосредственно в качестве ввода частоты
- см. LC_CTRL_PORT в `config_<MCU>.h` для пинов порта

`/// define HW_LC_METER - прокомментировать, чтобы активировать`

L/C Meter значение опорного конденсатора C_p (в 0,1 пФ)

- должно быть около 1000пФ

`#define LC_METER_C_REF 10000` - комментарий для деактивации

L/C Meter также отображает частоту LC генератора.

- помогает обнаружить дрейф частоты осциллятора
- требует отображения более двух текстовых строк

`/// define LC_METER_SHOW_FREQ - прокомментировать, чтобы активировать`

Реле для параллельного конденсатора (выборка АЦП)

`/// define HW_CAP_RELAY - прокомментировать, чтобы активировать`

6.2.2. Возможности программного обеспечения

ШИМ-генератор с простым управлением

- Выход через OC1B

`#define SW_PWM_SIMPLE` - комментарий для деактивации

PWM-генератор с расширенными возможностями

- Выход через OC1B
- требует дополнительных кнопок и дисплея с более чем 2 строками текста

`/// define SW_PWM_PLUS - прокомментировать, чтобы активировать`

измерение индуктивности

`#define SW_INDUCTOR` - комментарий для деактивации

измерение ESR

- Тактовая частота MCU >= 8 МГц требуется

`#define SW_ESR` - комментарий для деактивации

- Выберите SW_OLD_ESR для старого метода измерения, начиная с 180nF

`/// define SW_OLD_ESR - прокомментировать, чтобы активировать`

ESR Tool Измерение ESR в цепи

- требует активации SW_ESR или SW_OLD_ESR

`/// define SW_ESR_TOOL - прокомментировать, чтобы активировать`

тест поворотного энкодера


```

// #define SW_ENCODER - прокомментировать, чтобы активировать
генератор сигналов квадратной волны
- требует дополнительных клавиш или поворотного энкодера
#define SW_SQUAREWAVE- комментарий для деактивации
ИК-детектор/декодер (через тестовые контакты)
- требуется модуль ИК-приемника, например, серии TSOP.
- Модуль подключается к тестовым контактам.
#define SW_IR_RECEIVER - комментарий для деактивации
токоограничивающий резистор для модуля ИК-приемника
- только для модулей 5 В
- Внимание: Любое короткое замыкание может разрушить MCU.
// #define SW_IR_DISABLE_RESISTOR - прокомментировать, чтобы активировать
Дополнительные протоколы для ИК-детектора/декодера
- более редкие протоколы, которые увеличивают использование флэш-памяти;)
// #define SW_IR_RX_EXTRA - прокомментировать, чтобы активировать
ИК передатчик дистанционного управления
- Выход через OC1B
- требует дополнительных кнопок и дисплея с более чем 4 строками текста
- также требуется ИК-светодиод с простым драйвером
// #define SW_IR_TRANSMITTER - прокомментировать, чтобы активировать
альтернативная петля задержки для ИК пульта дистанционного передатчика
- на случай, если компилятор C испортит стандартный цикл задержки
приводит к неправильному времени импульса/паузы.
// #define SW_IR_TX_ALTDELAY - прокомментировать, чтобы активировать
Дополнительные протоколы для дистанционного ИК-передатчика
- более редкие протоколы, которые увеличивают использование флэш-памяти;)
// #define SW_IR_TX_EXTRA - прокомментировать, чтобы активировать
optocoupler test
#define SW_OPTO_COUPLER - комментарий для деактивации
испытание однопереходных транзисторов
#define SW_UJT - комментарий для деактивации
Проверить транзистор Шоттки (BJT с зажимом Шоттки)
#define SW_SCHOTTKY_BJT - комментарий для деактивации
Тестирование сервоприводов RC
- Выход через OC1B
- требует дополнительных кнопок и дисплея с более чем 2 строками текста
// #define SW_SERVO - прокомментировать, чтобы активировать
измерение температуры с помощью DS18B20
- также активировать ONEWIRE_PROBES или ONEWIRE_IO_PIN (см. 'Шины')
// #define SW_DS18B20 - прокомментировать, чтобы активировать
Считать код ПЗУ OneWire и отобразить его.
- Опция для инструментов связанных с OneWire.
- требуется дисплей с более чем двумя линиями подсветки.
// #define SW_ONEWIRE_READ_ROM - прокомментировать, чтобы активировать
Сканирование 1-Wire шины отображает коды ПЗУ всех подключенных участников ши-
ны.
- требуется дисплей с более чем двумя строками.

```

- также включить ONEWIRE_PROBES или ONEWIRE_IO_PIN (см. 'Шины')
SW_ONEWIRE_SCAN // #define - прокомментировать, чтобы активировать

тест тока утечки конденсатора

- требуется дисплей с более чем двумя линиями подсветки.

// #define SW_CAP_LEAKAGE - прокомментировать, чтобы активировать

отображение обратного hFE для BJTs

- hFE, обмениваемые для коллектора и эмиттера

#define SW_REVERSE_HFE - комментарий для деактивации

Display I_C/I_E Тестовый ток для измерения hFE

- I_C для цепи общего эмиттера

- I_E для цепи общего коллектора

// #define SW_HFE_CURRENT - прокомментировать, чтобы активировать

мониторы R/C/L

- Мониторинг пассивных компонентов, подключенных к зондам #1 и #3

- Мониторы для L требуют включения SW_INDUCTOR

- для ISR, либо SW_ESR, либо SW_OLD_ESR должны быть включены

// #define SW_MONITOR_R /* Только R */

// #define SW_MONITOR_C /* Только C плюс ISR */

// #define SW_MONITOR_L /* Только L */

// #define SW_MONITOR_RCL /* R плюс L, или C плюс ESR */

// #define SW_MONITOR_RL /* R плюс L */ - (один||несколько)

DHT11, DHT22 и совместимые датчики влажности и температуры

// #define SW_DHTXX - прокомментировать, чтобы активировать

Тест шрифтов диплома

#define SW_FONT_TEST - комментарий для деактивации

проверка сопротивления на соответствие стандартному значению серии E

- требуется дисплей с более чем 2 текстовыми строками

- Для работы в режиме цветового кода требуется цветной графический дисплей

// #define SW_R_E24_5_T E24 5% допуск, текст

// #define SW_R_E24_5_CC E24 5% допуск, цветовой код

// #define SW_R_E24_1_T E24 1% допуск, текст

// #define SW_R_E24_1_CC E24 1% допуск, цветовой код

// #define SW_R_E96_T E96 1% допуск, текст

// #define SW_R_E96_CC E96 1% допуск, цветовой код

// #define SW_R_E96_EIA96 E96 1% допуск, EIA96 код - (один||несколько)

проверить конденсатор на соответствие стандартному значению серии E

- требуется дисплей с более чем 2 текстовыми строками

// #define SW_C_E6_T E6 20% допуск, текст

// #define SW_C_E12_T E12 10% допуск, текст - (один||несколько)

проверить индуктивность на соответствие стандартному значению серии E

- требуется дисплей с более чем 2 текстовыми строками

// #define SW_L_E6_T E6 20% допуск, текст

// #define SW_L_E12_T E12 10% допуск, текст - (один||несколько)

Для некоторых тестеров отключение измерения hFE с цепью общего коллектора и Rl в качестве базового резистора

- Проблема:

hFE-значения слишком высокие, потому что базовое напряжение измеряется слишком низко

- пострадавшие испытатели:

Niland M664 (в процессе изготовления)

// #define NO_HFE_C_RL - прокомментировать, чтобы активировать

циклы запуска осциллятора (после пробуждения из режима энергосбережения):

- типичные значения
- Внутренний RC-осциллятор:6
- Кристалл:... 16384 (также 256 или 1024 в зависимости от настроек предохранителя)
- Резонатор:... 16384 (также 256 или 1024, в зависимости от настроек предохранителя)

```
#ifndef OSC_STARTUP
#define OSC_STARTUP 16384
#endif
```

Листинг 6.6. Измените значение, если оно НЕ соответствует тестеру!

6.2.3. Пользовательский интерфейс

выбор языка стандартом является ISO 8859 -1.

Префикс `_2` эквивалентен ISO 8859 -2.

- русский всегда Windows -1251.

```
#define UI_ENGLISH
// #define UI_CZECH
// #define UI_CZECH_2
// #define UI_DANISH
// #define UI_GERMAN
// #define UI_ITALIAN
// #define UI_POLISH
// #define UI_POLISH_2
// #define UI_ROMANIAN
// #define UI_RUSSIAN
// #define UI_RUSSIAN_2
// #define UI_SPANISH
```

Листинг 6.7. Выберите нужный язык, здесь предварительно выбран английский.

запятая вместо точки для указания десятичной дроби.

`// #define UI_COMMA` - прокомментировать, чтобы активировать

Отображение температуры в градусах Фаренгейта вместо Цельсия.

`// #define UI_FAHRENHEIT` - прокомментировать, чтобы активировать

показывать шестнадцатеричные значения в верхнем регистре вместо строчных букв

`// #define UI_HEX_UPPERCASE` - прокомментировать, чтобы активировать

стандартный режим работы Auto-Hold - вместо непрерывного режима

`#define UI_AUTOHOLD` - комментарий для деактивации

вызов меню путем короткого замыкания всех трех тестовых контактов.

- старое поведение по умолчанию

`// #define UI_SHORT_CIRCUIT_MENU` - прокомментировать, чтобы активировать

подсказки вместо курсора, если доступно.

- в настоящее время только "Меню / Тест"

- требует дополнительных клавиш и дисплея с достаточным количеством

строк текста (рекомендуется: > = 8 строк)

`// #define UI_KEY_HINTS` - прокомментировать, чтобы активировать

профиль настройки - выбирается после включения питания.

- Для тестеров с дополнительными, постоянно установленными измерительными кабелями.

`// #define UI_CHOOSE_PROFILE` - прокомментировать, чтобы активировать

выход через последовательный интерфейс TTL

- также активируйте `SERIAL_BITBANG` или `SERIAL_HARDWARE` (см. раздел 'Шины')

`// #define UI_SERIAL_COPY` - прокомментировать, чтобы активировать

управление тестером через последовательный TTL интерфейс

- также активируйте `SERIAL_BITBANG` или `SERIAL_HARDWARE` и `SERIAL_RW`

`// #define UI_SERIAL_COMMANDS` - прокомментировать, чтобы активировать

Максимальное время ожидания после тестирования (в мс)

- действительна только для непрерывного режима.
- Время между выводом результата и началом нового тестового цикла.

```
#define CYCLE_DELAY ... 3000 - При необходимости измените время
```

Максимальное количество тестовых запусков без компонентов

- относится только к непрерывному режиму.
- При достижении этого числа тестер выключается.

```
#define CYCLE_MAX ... 5 - При необходимости измените номер
```

Автоматическое выключение , если в течение некоторого времени (в сек) не нажимается ни одна клавиша.

- применяется только в режиме автоудержания.

```
// #define POWER_OFF_TIMEOUT ... 60 - прокомментировать, чтобы активировать
```

символы компонентов для причудливой распиновки

- для 3-контактных полупроводников
- требует графического отображения и растрового изображения символов

```
#define SW_SYMBOLS - комментарий для деактивации
```

цветовая кодировка для контрольных контактов

- требуется цветной графический дисплей
- отредактируйте файл colors.h, чтобы выбрать правильные цвета зондов.

(COLOR_PROBE_1, COLOR_PROBE_2 и COLOR_PROBE_3)

```
#define UI_PROBE_COLORS - комментарий для деактивации
```

color title

- требуется цветной графический дисплей
- отредактируйте файл colors.h для выбора предпочтительного цвета (COLOR_TITLE)

```
// #define UI_COLORED_TITLES - прокомментировать, чтобы активировать
```

цветной курсор и подсказки клавиш

- требуется цветной графический дисплей
- отредактируйте файл colors.h, чтобы выбрать предпочтительный цвет (COLOR_CURSOR)

```
// #define UI_COLORED_CURSOR - прокомментировать, чтобы активировать
```

пункт меню выключить

```
// #define SW_POWER_OFF - прокомментировать, чтобы активировать
```

значения округления для DS18B20

- DS18B20 (0.1 °C/F)

```
// #define UI_ROUND_DS18B20 - прокомментировать, чтобы активировать
```

Хранение данных встроенного ПО (тексты, таблицы и т.д.)

- Данные самонастройки всегда хранятся в EEPROM
- Шрифты и символы всегда хранятся во Flash

```
#define DATA_EEPROM /* хранить данные в EEPROM */
```

```
// #define DATA_FLASH /* хранить данные во Flash */
```

6.2.4. Управление питанием

тип автоматического выключателя

- выключатель питания с мягкой фиксацией (стандарт)
как в опорной цепи тестера.

тестер способен самостоятельно отключиться

- ручной выключатель

Тестер не может выключиться

```
// #define POWER_SWITCH_SOFT
```

```
#define POWER_SWITCH_MANUAL -выбранный
```

режим мониторинга батареи

- BAT_NONE полностью деактивирует контроль батареи
- BAT_DIREKTE прямое измерение напряжения батареи (<5В)
- BAT_DIVIDER измерение через делитель напряжения

```
//#define BAT_NONE
//#define BAT_DIRECT
#define BAT_DIVIDER
```

-выбранный

Дополнительный внешний источник питания без контроля

Некоторые тестеры поддерживают дополнительный внешний источник питания, но из-за схемотехники он не позволяет измерять напряжение.

Это может привести к отключению тестера из-за низкого напряжения батареи.

Переключатель в нижней части предотвращает выключение, если измеренное напряжение ниже 0,9 В (вызвано током утечки диода).

```
//#define BAT_EXT_UNMONITORED - прокомментировать, чтобы активировать
```

делитель напряжения для мониторинга батарей

- BAT_R1: верхний резистор в Ω
- BAT_R2: нижний резистор в Ω
- Значения по умолчанию: R1=10k, R2=3,3k

```
#define BAT_R1 ... 10000
#define BAT_R2 ... 3300
```

- изменение
- изменение

Падение напряжения из-за диода защиты от обратной полярности и транзистора управления питанием (в мВ)

- или любой другой цепи в источнике питания.
- Возьмите DMM и измерьте падение напряжения!
- Диод Шоттки около 200 мВ / PNP BJT около 100 мВ.

```
#define BAT_OFFSET ... 290
```

- изменение

низкое напряжение батареи (в мВ)

- Тестер предупреждает, когда достигается значение BAT_WEAK.
- Падение напряжения BAT_OFFSET учитывается при расчете.

```
#define BAT_WEAK dots 7400
```

- изменени

напряжение для разряженной батареи (в мВ)

- Тестер выключается при достижении значения BAT_LOW.
- Падение напряжения BAT_OFFSET учитывается при расчете.

```
#define BAT_LOW ... 6400
```

- изменени

спящий режим для снижения энергопотребления.

```
#define SAVE_POWER
```

- комментарий для деактивации

6.2.5. Настройки измерения и смещения

Опорное напряжение АЦП на основе Vcc (в мВ)

```
#define UREF_VCC dots 5001
```

- изменени

Смещение для внутреннего опорного напряжения (в мВ): от -100 до 100

- Для компенсации отклонений между реальным и измеренным значением.
- АЦП имеет разрешение около 4,88 мВ для $V_{ref} = 5 \text{ В (Vcc)}$ и 1,07 мВ для $V_{ref} = 1,1 \text{ В (зазор)}$.
- Добавляется к измеренному напряжению опорной полосы.

```
#define UREF_OFFSET ... 0
```

- изменени

Точные значения тестовых резисторов

- Значение по умолчанию для Rl равно 680 Ω
- Значение по умолчанию для Rh равно 470 Ω

```
/ Rl в  $\Omega$  /
```

```
#define R_LOW ... 680
```

- изменени

```
/ Rh in  $\Omega$  /
```

```
#define R_HIGH ... 470000
```

- изменени

Смещение для систематических ошибок измерения сопротивления с Rh (470k) в Ω

- Если резисторы $>20k$ слишком высокие или слишком низкие, отрегулируйте смещение соответствующим образом.
 - Смещение по умолчанию составляет 350Ω
- ```
#define RH_OFFSET ... 350
```

- изменени

### сопротивление тестовых контактов/кабелей (в $0.01 \Omega$ )

- стандартное смещение для трасс и тестовых кабелей
  - Сопротивление двух испытательных контактов, соединенных последовательно
  - предполагая, что все тестовые контакты имеют одинаковое/подобное сопротивление
  - обновляется в процессе самокоррекции
- ```
#define R_ZERO ... 20
```

- изменени

Смещение сопротивления для отдельных пар тестовых кабелей

- если они сильно отличаются
- будет обновляться в самонастройке
- ```
//#define R_MULTIOFFSET
```

- изменени

### емкость тестовых контактов/кабелей (в пФ)

- смещение по умолчанию для MCU, платы и тестового кабеля
- обновляется в процессе самонастройки
- Примеры емкостей для различных длин кабеля:
  - 3 пФ            около    10 см
  - 9 пФ            около    30 см
  - 15 пФ           около    50 см
- Максимальное значение 100 пФ

```
#define C_ZERO ... 43
```

- изменени

### удельная емкость пары тестового кабеля

- вместо среднего значения для всех пар тестового кабеля
- обновляется при самонастройке
- ```
//#define CAP_MULTIOFFSET
```

- прокомментировать, чтобы активировать

Максимальное конечное напряжение заряда для конденсаторов (в мВ)

- ниже которого мы считаем конденсатор разряженным

```
#define CAP_DISCHARGED ... 2
```

- изменени

поправочные коэффициенты для конденсаторов (в 0,1%)

- положительный фактор увеличивает значение емкости
- отрицательный фактор уменьшает значение емкости

```
CAP_FACTOR_SMALL для конденсаторов < 4,7  $\mu$ F
CAP_FACTOR_MID для конденсаторов 4,7 - 47  $\mu$ F
CAP_FACTOR_LARGE для конденсаторов > 47  $\mu$ F
#define CAP_FACTOR_SMALL ... 0 без коррекции
#define CAP_FACTOR_MID ... -40 -4.0%
#define CAP_FACTOR_LARGE ... -90 -9.0%
```

- изменени

- изменени

- изменени

количество раундов АЦП для каждого измерения

- Допустимые значения находятся в диапазоне от 1 до 255

```
#define ADC_SAMPLES ... 25
```

- изменени

100nF AREF буферный конденсатор

- используется некоторыми картами MCU
- увеличит время измерения
- Рекомендация: заменить на конденсатор 1nF.

```
#define ADC_LARGE_BUFFER_CAP
```

- комментарий для деактивации

6.2.6. F & E - предназначен для разработчиков микропрограммного обеспечения

Включить функции чтения для модуля дисплея

- Драйвер дисплея и настройки интерфейса должны поддерживать this
- ```
//#define LCD_READ - прокомментировать, чтобы активировать
```

### прочитать ID контроллера дисплея

- ID отображается на экране приветствия (по версии прошивки)
  - требует функции чтения дисплея (LCD\_READ)
  - рекомендуется: последовательный выход (UI\_SERIAL)
- ```
//#define SW_DISPLAY_ID - прокомментировать, чтобы активировать
```

Считывает регистры контроллера дисплея

- и выводит их последовательно через TTL.
 - требует функций чтения дисплея (LCD_READ)
 - и последовательного вывода (UI_SERIAL)
- ```
//#define SW_DISPLAY_REG - прокомментировать, чтобы активировать
```

## 6.2.7. Шина

**Шина I2C** может потребоваться для определенного оборудования.

- может быть уже включена настройкой дисплея (config\_<MCU>.h)
- для битового порта и выводов смотрите I2C\_PORT (config\_<MCU>.h)
- Аппаратный I2C (TWI) автоматически использует правильный пин MCU.
- Закомментируйте I2C\_BITBANG или I2C\_HARDWARE, чтобы активировать it
- прокомментировать одно из автобусных часов

```
//#define I2C_BITBANG bit-bang I2C
//#define I2C_HARDWARE MCU Аппаратные средства TWI
//#define I2C_STANDARD_MODE 100kHz Тактовая частота шины
//#define I2C_FAST_MODE 400kHz Тактовая частота шины
//#define I2C_RW Включить поддержку чтения (не проверено)
```

**Шина SPI** может потребоваться определенному оборудованию.

- может быть уже включена настройкой дисплея (config\_<MCU>.h)
- для битового порта и выводов см. SPI\_PORT (config\_<MCU>.h)
- Аппаратный SPI автоматически использует правильные пины MCU.
- Закомментируйте SPI\_BITBANG или SPI\_HARDWARE для включения.

```
//#define SPI_BITBANG бит-банг SPI
//#define SPI_HARDWARE Аппаратный SPI
//#define SPI_RW включить поддержку чтения
```

### Последовательный TTL порт

- может быть уже включен через настройки дисплея (config\_<MCU>.h)
- для битового порта и выводов смотрите SERIAL\_PORT (config\_<MCU>.h).
  - Аппаратная последовательность автоматически использует правильные пины для MCU.
  - Закомментируйте SERIAL\_BITBANG или SERIAL\_HARDWARE для включения

```
//#define SERIAL_BITBANG Последовательный бит-банг
//#define SERIAL_HARDWARE Серийное оборудование
//#define SERIAL_RW включить поддержку чтения
```

### 1-Wire шины

- Информацию о выделенном контакте ввода/вывода вы найдёте на сайте ONEWIRE\_PORT (config\_<MCU>.h).
- Закомментируйте ONEWIRE\_PROBES или ONEWIRE\_IO\_PIN для включения.

```
//#define ONEWIRE_PROBES через Тестконтакт
//#define ONEWIRE_IO_PIN через выделенный контакт ввода/вывода
```



## 6.3. конфигурация Config\_328.h

содержит настройки аппаратного обеспечения для дисплеев, кнопки и так далее. Когда микропрограмма скомпилирована, соответствующий файл является в соответствии с MCU будет включена автоматически. Это снова Заголовочный файл C, т.е. применяются правила комментирования для C. Кроме “//” для отдельных строк Блочные комментарии также делаются с помощью “#if 0 ... #endif” используются. Чтобы прокомментировать блок просто добавьте “//” перед соответствующими “#if 0” и “#endif” ; чтобы прокомментировать обратное. Вы также можете прокомментировать блок удалив строки с “#if 0” и “#endif” .

### 6.3.1. Модуль дисплея

Дисплеи и их настройки обсуждаются в главе 2.6. начиная со страницы 12. Примером может служить модуль ST7565R, в котором #if 0 и #endif были закоментированы “//”.

#### Modul ST7565R

```
//#if 0
#define LCD_ST7565R /* display controller ST7565R */
```

Листинг 6.8. По // #if 0 и #define LCD\_ST7565R ... выбран правильный модуль

```
//#endif
```

Листинг 6.9. Конец цикла также закоментирован

### 6.3.2. Назначение портов и выводов

#### тестовые контакты/тестовые щупы

- Первые 3 контакта аналогового порта должны быть использованы для тестовых контактов.
- Пожалуйста, не изменяйте определения TP1, TP2 и TP3!
- Не используйте этот порт совместно с POWER\_CTRL или TEST\_BUTTON!

```
#define TP1 PC0 /* test pin / probe #1 */
#define TP2 PC1 /* test pin / probe #2 */
#define TP3 PC2 /* test pin / probe #3 */
```

Листинг 6.10. Не меняйте контакты

#### Фиксированные опции HW

- Возможные варианты.

```
#define TP_ZENER PC3 /* test pin for 10:1 voltage divider */
#define TP_REF PC4 /* test pin for 2.5V reference and relay */
#define TP_BAT PC5 /* test pin for battery (4:1 voltage divider) */
```

Листинг 6.11. Не меняйте контакты

#### тестовые резисторы

- Для вывода ШИМ / квадратной волны через тестовый контакт 2 R\_RL\_2 должен быть PB2/OC1B.
- Не используйте этот порт совместно с POWER\_CTRL или TEST\_BUTTON!

```
#define R_RL_1 PB0 /* Rl (680R) for test pin #1 */
#define R_RH_1 PB1 /* Rh (470k) for test pin #1 */
#define R_RL_2 PB2 /* Rl (680R) for test pin #2 */
#define R_RH_2 PB3 /* Rh (470k) for test pin #2 */
#define R_RL_3 PB4 /* Rl (680R) for test pin #3 */
#define R_RH_3 PB5 /* Rh (470k) for test pin #3 */
```

Листинг 6.12. Не меняйте контакты

**выделенный выход сигнала через OC1B** - пожалуйста, не изменяйте!

```
#define SIGNAL_OUT PB2 /* MCU's OC1B pin */
```

Листинг 6.13. не изменяйте!

**выключатель питания** - Не может быть тем же портом, что и ADC\_PORT или R\_PORT.

```
#define POWER_CTRL PD6 /* control pin (1: on / 0: off) */
```

Листинг 6.14. адаптировать при необходимости

**тестовая кнопка** - Не может быть тем же портом, что и ADC\_PORT или R\_PORT.

```
#define TEST_BUTTON PD7 /* test/start push button (low active) */
```

Листинг 6.15. адаптировать при необходимости

**Поворотный энкодер**

```
#define ENCODER_A PD3 /* rotary encoder A signal */
#define ENCODER_B PD2 /* rotary encoder B signal */
```

Листинг 6.16. адаптировать при необходимости

**Кнопки вверх/вниз**

```
#define KEY_INC PD2 /* increase push button (low active) */
#define KEY_DEC PD3 /* decrease push button (low active) */
```

Листинг 6.17. адаптировать при необходимости

**Счетчик частоты**

- Простая и расширенная версия
- Входом должен быть вывод PD4/T0

```
#define COUNTER_IN PD4 /* signal input T0 */
```

Листинг 6.18. не изменяйте!

**ИК-детектор/декодер**

- фиксированный модуль, подключенный к выделенному контакту ввода/вывода

```
#define IR_DATA PC6 /* data signal */
```

Листинг 6.19. адаптировать при необходимости

### 6.3.3. Шина

**SPI** -SCK PB5, MOSI PB3, MISO PB4 и /SS PB2

- Бит-Банг-SPI уже может быть установлен в области отображения

```
/* for bit-bang and hardware SPI */
#ifndef SPI_PORT
#define SPI_PORT PORTB /* port data register */
#define SPI_DDR DDRB /* port data direction register */
#define SPI_PIN PINB /* port input pins register */
#define SPI_SCK PB5 /* pin for SCK */
#define SPI_MOSI PB3 /* pin for MOSI */
#define SPI_MISO PB4 /* pin for MISO */
#define SPI_SS PB2 /* pin for /SS */
```

Листинг 6.20. ggf. anpassen

**I2C** - Аппаратный I2C (TWI) использует SDA PC4 и SCL PC5

- Возможно, в области отображения уже установлен бит-банг I2C.

```
/* for bit-bang I2C */
#ifndef I2C_PORT
#define I2C_PORT PORTC /* port data register */
#define I2C_DDR DDRC /* port data direction register */
#define I2C_PIN PINC /* port input pins register */
#define I2C_SDA PC4 /* pin for SDA */
#define I2C_SCL PC5 /* pin for SCL */
```

Листинг 6.21. адаптировать при необходимости

**TTL последовательный интерфейс**

- аппаратный USART0 использует Rx PD0 & Tx PD1

```
/* for hardware RS232 */
#define SERIAL_USART0 /* use USART0 */
/* for bit-bang RS232 */
#define SERIAL_PORT PORTD /* port data register */
#define SERIAL_DDR DDRD /* port data direction register */
#define SERIAL_PIN PIND /* port input pins register */
#define SERIAL_TX PD1 /* pin for Tx (transmit) */
#define SERIAL_RX PD0 /* pin for Rx (receive) */
#define SERIAL_PCINT 16 /* PCINT# for Rx pin */
```

Листинг 6.22. адаптировать при необходимости

**1-Wire шины**

- выделенный вывод ввода/вывода

```
#define ONEWIRE_PORT PORTC /* port data register */
#define ONEWIRE_DDR DDRC /* port data direction register */
#define ONEWIRE_PIN PINC /* port input pins register */
#define ONEWIRE_DQ PC6 /* DQ (data line) */
```

Листинг 6.23. адаптировать при необходимости

## 6.4. конфигурация Config\_644.h

содержит настройки, связанные с оборудованием, для дисплеев, кнопок и так далее. Когда микропрограмма скомпилирована, соответствующий файл является в соответствии с MCU будет включена автоматически. Это снова Заголовочный файл C, т.е. применяются правила комментирования для C. Кроме “//” для отдельных строк Блочные комментарии также делаются с помощью “#if 0 ... #endif” используются. Чтобы прокомментировать блок просто добавьте “//” перед соответствующими “#if 0” и “#endif” ; чтобы прокомментировать обратное. Вы также можете прокомментировать блок удалив строки с “#if 0” и “#endif” .

### 6.4.1. Модуль дисплея

Дисплеи и их настройки обсуждаются в главе 2.6. начиная со страницы 12. Примером может служить модуль ILI9341, в котором #if 0 и #endif были закомментированы “//” .

#### Модуль ILI9341

```
//#if 0
#define LCD_ILI9341 /* display controller ILI9341/ILI9342 */
```

Листинг 6.24. По // #if 0 и #define ILI9341 ... выбран правильный модуль.

```
//#endif
```

Листинг 6.25. Конец цикла также закомментирован

### 6.4.2. Назначение портов и выводов

#### тестовые контакты/тестовые щупы

- Первые 3 контакта аналогового порта должны быть использованы для тестовых контактов.
- Пожалуйста, не изменяйте определения TP1, TP2 и TP3!
- Не используйте этот порт совместно с POWER\_CTRL или TEST\_BUTTON!

```
#define TP1 PA0 /* test pin / probe #1 */
#define TP2 PA1 /* test pin / probe #2 */
#define TP3 PA2 /* test pin / probe #3 */
```

Листинг 6.26. Не меняйте контакты

#### Фиксированные опции HW

```
#define TP_ZENER PA3 /* test pin for 10:1 voltage divider */
#define TP_REF PA4 /* test pin for 2.5V reference and relay */
#define TP_BAT PA5 /* test pin for battery (4:1 voltage divider) */
#define TP_CAP PA7 /* test pin for self-adjustment cap */
```

Листинг 6.27. Не меняйте контакты

#### тестовые резисторы

- Для вывода ШИМ / квадратной волны через тестовый контакт 2 R\_RL\_2 должен быть RB2/OC1B.
- Не используйте этот порт совместно с POWER\_CTRL или TEST\_BUTTON!

```
#define R_RL_1 PD2 /* Rl (680R) for test pin #1 */
#define R_RH_1 PD3 /* Rh (470k) for test pin #1 */
#define R_RL_2 PD4 /* Rl (680R) for test pin #2 */
#define R_RH_2 PD5 /* Rh (470k) for test pin #2 */
#define R_RL_3 PD6 /* Rl (680R) for test pin #3 */
#define R_RH_3 PD7 /* Rh (470k) for test pin #3 */
```

Листинг 6.28. Не меняйте контакты

**выделенный выход сигнала через OC1B** - пожалуйста, не изменяйте!

```
#define SIGNAL_OUT PD4 /* MCU's OC1B pin */
```

Листинг 6.29. не изменяйте!

**выключатель питания** - Не может быть тем же портом, что и ADC\_PORT или R\_PORT.

```
#define POWER_CTRL PC6 /* control pin (1: on / 0: off) */
```

Листинг 6.30. адаптировать при необходимости

**тестовая кнопка** - Не может быть тем же портом, что и ADC\_PORT или R\_PORT.

```
#define TEST_BUTTON PC7 /* test/start push button (low active) */
```

Листинг 6.31. адаптировать при необходимости

### Поворотный энкодер

```
#define ENCODER_A PC3 /* rotary encoder A signal */
#define ENCODER_B PC4 /* rotary encoder B signal */
```

Листинг 6.32. адаптировать при необходимости

### Кнопки вверх/вниз

```
#define KEY_INC PC4 /* increase push button (low active) */
#define KEY_DEC PC3 /* decrease push button (low active) */
```

Листинг 6.33. адаптировать при необходимости

### Счетчик частоты

- Простая и расширенная версия
- Входом должен быть вывод PB0/T0

```
#define COUNTER_IN PB0 /* signal input T0 */
```

Листинг 6.34. не изменяйте!

### Управление расширенным счетчиком частоты

```
#define COUNTER_CTRL_DIV PC0 /* prescaler (low 1:1, high x:1) */
#define COUNTER_CTRL_CH0 PC1 /* channel addr #0 */
#define COUNTER_CTRL_CH1 PC2 /* channel addr #1 */
```

Листинг 6.35. адаптировать при необходимости

### Счетчик L/C

- Вход частоты, подключенный к контакту PB0/T0

```
#define LC_CTRL_CP PC0 /* reference cap (low: on / high: off) */
#define LC_CTRL_LC PC1 /* L/C selection (low: C / high: L) */
```

Листинг 6.36. не изменяйте!

### ИК-детектор/декодер

- фиксированный модуль, подключенный к выделенному контакту ввода/вывода

```
#define IR_DATA PC2 /* data signal */
```

Листинг 6.37. адаптировать при необходимости

**Встроенный конденсатор для самобалансировки** - Вывод АЦП - это TP\_CAP сверху

- настройки действительны для резистора 470 kΩ
- должен быть пленочный конденсатор от 100 нФ до 1000 нФ

```
#define ADJUST_RH PC5 /* Rh (470k) for fixed cap */
```

Листинг 6.38. адаптировать при необходимости

### Реле для параллельного конденсатора (Выборка АЦП)

- между TP1 и TP3
- конденсатор должен иметь емкость от 10 нФ до 27 нФ

```
#define CAP_RELAY_CTRL PC2 /* control pin */
```

Листинг 6.39. адаптировать при необходимости

### 6.4.3. Шина

**SPI** -SCK PB7, MOSI PB5, MISO PB6 и /SS PB4

- Бит-Банг-SPI уже может быть установлен в области отображения

```
/* for bit-bang and hardware SPI */
#ifndef SPI_PORT
#define SPI_PORT PORTB /* port data register */
#define SPI_DDR DDRB /* port data direction register */
#define SPI_PIN PINB /* port input pins register */
#define SPI_SCK PB7 /* pin for SCK */
#define SPI_MOSI PB5 /* pin for MOSI */
#define SPI_MISO PB6 /* pin for MISO */
#define SPI_SS PB4 /* pin for /SS */
```

Листинг 6.40. адаптировать при необходимости

**I2C** - Аппаратный I2C (TWI) использует SDA PC1 и SCL PC0

- Возможно, в области отображения уже установлен бит-банг I2C.

```
/* for bit-bang I2C */
#ifndef I2C_PORT
#define I2C_PORT PORTC /* port data register */
#define I2C_DDR DDRC /* port data direction register */
#define I2C_PIN PINC /* port input pins register */
#define I2C_SDA PC1 /* pin for SDA */
#define I2C_SCL PC0 /* pin for SCL */
```

Листинг 6.41. адаптировать при необходимости

### Последовательный интерфейс TTL

- используется аппаратный USART

USART0: Rx PD0 и Tx PD1

USART2: Rx PD2 и Tx PD3

```
/* for hardware TTL serial */
#define SERIAL_USART 0 /* use USART0 */
/* for bit-bang TTL serial */
#define SERIAL_PORT PORTD /* port data register */
#define SERIAL_DDR DDRD /* port data direction register */
#define SERIAL_PIN PIND /* port input pins register */
#define SERIAL_TX PD1 /* pin for Tx (transmit) */
#define SERIAL_RX PD0 /* pin for Rx (receive) */
#define SERIAL_PCINT 24 /* PCINT# for Rx pin */
```

Листинг 6.42. адаптировать при необходимости

**1-Wire** шины - выделенный вывод ввода/вывода

```
#define ONEWIRE_PORT PORTC /* port data register */
#define ONEWIRE_DDR DDRC /* port data direction register */
#define ONEWIRE_PIN PINC /* port input pins register */
#define ONEWIRE_DQ PC2 /* DQ (data line) */
```

Листинг 6.43. адаптировать при необходимости

## 6.5. конфигурация Config\_1280.h

содержит настройки, связанные с оборудованием, для дисплеев, кнопок и так далее. Когда микропрограмма скомпилирована, соответствующий файл является в соответствии с MCU будет включена автоматически. Это снова Заголовочный файл C, т.е. применяются правила комментирования для C. Кроме “//” для отдельных строк Блочные комментарии также делаются с помощью “#if 0 ... #endif” используются. Чтобы прокомментировать блок просто добавьте “//” перед соответствующими “#if 0” и “#endif” ; чтобы прокомментировать обратное. Вы также можете прокомментировать блок удалив строки с “#if 0” и “#endif” .

### 6.5.1. Модуль дисплея

Дисплеи и их настройки обсуждаются в главе 2.6. начиная со страницы 12. Примером может служить модуль ILI9486, в котором #if 0 и #endif были закомментированы “//” .

**Модуль ILI9486:**

```
//#if 0
//#define LCD_ILI9481 /* display controller ILI9481 */
#define LCD_ILI9486 /* display controller ILI9486 */
```

Листинг 6.44. По // #if 0 и #define ILI9486 ... выбран правильный модуль.

```
//#endif
```

Листинг 6.45. Конец цикла также закомментирован

### 6.5.2. Назначение портов и выводов

**тестовые контакты/тестовые щупы**

- Первые 3 контакта аналогового порта должны быть использованы для тестовых контактов.
- Пожалуйста, не изменяйте определения TP1, TP2 и TP3!
- Не используйте этот порт совместно с POWER\_CTRL или TEST\_BUTTON!

```
#define TP1 PF0 /* test pin / probe #1 */
#define TP2 PF1 /* test pin / probe #2 */
#define TP3 PF2 /* test pin / probe #3 */
```

Листинг 6.46. Не меняйте контакты

**Фиксированные опции HW**

```
#define TP_ZENER PF3 /* test pin for 10:1 voltage divider */
#define TP_REF PF4 /* test pin for 2.5V reference and relay */
#define TP_BAT PF5 /* test pin for battery (4:1 voltage divider) */
#define TP_CAP PF7 /* test pin for self-adjustment cap */
```

Листинг 6.47. Не меняйте контакты!

**тестовые резисторы**

- Для вывода ШИМ / квадратной волны через тестовый контакт 2 R\_RL\_2 должен быть PB2/OC1B.
- Не используйте этот порт совместно с POWER\_CTRL или TEST\_BUTTON!

```
#define R_RL_1 PK0 /* Rl (680R) for test pin #1 */
#define R_RH_1 PK1 /* Rh (470k) for test pin #1 */
#define R_RL_2 PK2 /* Rl (680R) for test pin #2 */
#define R_RH_2 PK3 /* Rh (470k) for test pin #2 */
#define R_RL_3 PK4 /* Rl (680R) for test pin #3 */
#define R_RH_3 PK5 /* Rh (470k) for test pin #3 */
```

Листинг 6.48. Не меняйте контакты!

**выделенный выход сигнала через OC1B** - пожалуйста, не изменяйте!

```
#define SIGNAL_OUT PB6 /* MCU's OC1B pin */
```

Листинг 6.49. Не меняйте контакты!

**выключатель питания** - Не может быть тем же портом, что и ADC\_PORT или R\_PORT.

```
#define POWER_CTRL PA6 /* control pin (1: on / 0: off) */
```

Листинг 6.50. адаптировать при необходимости

**тестовая кнопка** - Не может быть тем же портом, что и ADC\_PORT или R\_PORT.

```
#define TEST_BUTTON PA7 /* test/start push button (low active) */
```

Листинг 6.51. адаптировать при необходимости

### Поворотный энкодер

```
#define ENCODER_A PA3 /* rotary encoder A signal */
#define ENCODER_B PA1 /* rotary encoder B signal */
```

Листинг 6.52. адаптировать при необходимости

### Кнопки вверх/вниз

```
#define KEY_INC PA3 /* increase push button (low active) */
#define KEY_DEC PA1 /* decrease push button (low active) */
```

Листинг 6.53. адаптировать при необходимости

### Счетчик частоты

- Простая и расширенная версия
- Входом должен быть вывод PB0/T0

```
#define COUNTER_IN PD7 /* signal input T0 */
```

Листинг 6.54. не изменяйте!

### Управление расширенным счетчиком частоты

```
#define COUNTER_CTRL_DIV PD4 /* prescaler (low 1:1, high x:1) */
#define COUNTER_CTRL_CH0 PD5 /* channel addr #0 */
#define COUNTER_CTRL_CH1 PD6 /* channel addr #1 */
```

Листинг 6.55. адаптировать при необходимости

### Счетчик L/C

- Вход частоты, подключенный к контакту PB0/T0

```
#define LC_CTRL_CP PC4 /* reference cap (low: on / high: off) */
#define LC_CTRL_LC PC5 /* L/C selection (low: C / high: L) */
```

Листинг 6.56. не изменяйте!

### ИК-детектор/декодер

- фиксированный модуль, подключенный к выделенному контакту ввода/вывода

```
#define IR_DATA PA0 /* data signal */
```

Листинг 6.57. адаптировать при необходимости

**Встроенный конденсатор для самобалансировки** - Вывод АЦП - это TP\_CAP сверху

- настройки действительны для резистора 470 kΩ
- должен быть пленочный конденсатор от 100 нФ до 1000 нФ

```
#define ADJUST_RH PA5 /* Rh (470k) for fixed cap */
```

Листинг 6.58. адаптировать при необходимости

### Реле для параллельного конденсатора (Выборка АЦП)

- между TP1 и TP3
- конденсатор должен иметь емкость от 10 нФ до 27 нФ

```
#define CAP_RELAY_CTRL PA2 /* control pin */
```

Листинг 6.59. адаптировать при необходимости



### 6.5.3. Шина

**SPI** -SCK PB1, MOSI PB2, MISO PB3 и /SS PB0

- Бит-Банг-SPI уже может быть установлен в области отображения

```
/* for bit-bang and hardware SPI */
#ifndef SPI_PORT
#define SPI_PORT PORTB /* port data register */
#define SPI_DDR DDRB /* port data direction register */
#define SPI_PIN PINB /* port input pins register */
#define SPI_SCK PB1 /* pin for SCK */
#define SPI_MOSI PB2 /* pin for MOSI */
#define SPI_MISO PB3 /* pin for MISO */
#define SPI_SS PB0 /* pin for /SS */
```

Листинг 6.60. адаптировать при необходимости

**I2C** - Аппаратный I2C (TWI) использует SDA PD1 и SCL PD0

- Возможно, в области отображения уже установлен бит-банг I2C.

```
#ifndef I2C_PORT
#define I2C_PORT PORTD /* port data register */
#define I2C_DDR DDRD /* port data direction register */
#define I2C_PIN PIND /* port input pins register */
#define I2C_SDA PD1 /* pin for SDA */
#define I2C_SCL PD0 /* pin for SCL */
```

Листинг 6.61. адаптировать при необходимости

### Последовательный интерфейс TTL

- используется аппаратный USART

USART0: Rx PE0 и Tx PE1

USART2: Rx PH0 и Tx PH1

USART1: Rx PD2 и Tx PD3

USART3: Rx PJ0 и Tx PJ1

```
/* for hardware TTL serial */
#define SERIAL_USART 0 /* use USART0 */
/* for bit-bang TTL serial */
#define SERIAL_PORT PORTE /* port data register */
#define SERIAL_DDR DDRE /* port data direction register */
#define SERIAL_PIN PINE /* port input pins register */
#define SERIAL_TX PE1 /* pin for Tx (transmit) */
#define SERIAL_RX PE0 /* pin for Rx (receive) */
#define SERIAL_PCINT 8 /* PCINT# for Rx pin */
```

Листинг 6.62. адаптировать при необходимости

**1-Wire** шины - выделенный вывод ввода/вывода

```
#define ONEWIRE_PORT PORTA /* port data register */
#define ONEWIRE_DDR DDRA /* port data direction register */
#define ONEWIRE_PIN PINA /* port input pins register */
#define ONEWIRE_DQ PA4 /* DQ (data line) */
```

Листинг 6.63. адаптировать при необходимости

для различных моделей тестеров.

Если у вас есть тестер, который не указан в этом списке тестера, пожалуйста, пришлите краткое описание тестера и его настроек автору по электронной почте автору [8], чтобы помочь другим пользователям.

## 7.1. Arduino Nano, Uno или Mega 2560

- Nano/Uno: ATmega 328, ATmega 2560, тактовая частота 16 МГц
- Скачайте схему Arduino и схему распиновки, чтобы увидеть соответствие между Arduino и назначением его контактов. В качестве примера смотрите MEGA на странице 76.
- Встроенный адаптер USB2serial упрощает последовательную связь. (включите SERIAL\_HARDWARE и соответствующие опции).
- **Не используйте** контакты TXD/RXD ни для чего другого.
- Если вы хотите использовать вывод D13 Arduino, удалите светодиодную цепь, которая будет мешать.
- Здесь перечислены только основные настройки

```
#define HW_FIXED_SIGNAL_OUTPUT
```

Листинг 7.1. Параметры оборудования

```
#define UI_AUTOHOLD
```

Листинг 7.2. Пользовательский интерфейс

```
#define POWER_SWITCH_MANUAL
#define BAT_NONE
```

Листинг 7.3. управление производительностью

```
#define ADC_LARGE_BUFFER_CAP /* 100nF cap at AREF */
```

Листинг 7.4. Настройки измерения и смещения

## 7.2. DIY Kit „AY-AT“ также GM328A

- ATmega328
- Цветной LCD модуль ST7735 (бит-банг-SPI)
- Поворотный энкодер (PD1 и PD3, параллельно дисплею)
- Внешнее опорное напряжение 2,5 В (TL431)
- Одночастотный счетчик с выделенным входом (PD4)
- Измерение внешнего напряжения до 45 В (PC3)
- Настройки взяты из flywheelz@EEVBlog
- Настройки для полусовместимого дисплея ST7735, предоставленные b0hoon4@gmail.com.

```
#define HW_ENCODER
#define ENCODER_PULSES 4 /* usually 4 pulses per step */
#define ENCODER_STEPS 20 /* usually 20 detents */
#define HW_REF25
#define HW_ZENER
#define HW_FREQ_COUNTER_BASIC
```

Листинг 7.5. Параметры оборудования

```
#define ENCODER_A PD3 /* rotary encoder A signal */
#define ENCODER_B PD1 /* rotary encoder B signal */
```

### Листинг 7.6. Поворотный энкодер

Вход для счетчика частоты - PD4 (T0)

```
#define LCD_ST7735
#define LCD_GRAPHIC /* graphic display */
#define LCD_COLOR /* color display */
#define LCD_SPI /* SPI interface */
#define LCD_PORT PORTD /* port data register */
#define LCD_DDR DDRD /* port data direction register */
#define LCD_RES PD0 /* port pin used for /RESX */
#define LCD_CS PD5 /* port pin used for /CSX (optional) */
#define LCD_DC PD1 /* port pin used for D/CX */
#define LCD_SCL PD2 /* port pin used for SCL */
#define LCD_SDA PD3 /* port pin used for SDA */
#define LCD_DOTS_X 128 /* number of horizontal dots */
#define LCD_DOTS_Y 160 /* number of vertical dots */
// #define LCD_OFFSET_X 4 /* enable x offset of 2 or 4 dots */
// #define LCD_OFFSET_Y 2 /* enable y offset of 1 or 2 dots */
#define LCD_FLIP_X /* enable horizontal flip */
// #define LCD_FLIP_Y /* enable vertical flip */
#define LCD_ROTATE /* switch X and Y (rotate by 90Grad) */
// #define LCD_LATE_ON /* turn on LCD after clearing it */
#define FONT_10X16_HF /* 10x16 font */
#define SYMBOLS_24X24_HF /* 24x24 symbols */
#define SPI_BITBANG /* bit-bang SPI */
#define SPI_PORT LCD_PORT /* SPI port data register */
#define SPI_DDR LCD_DDR /* SPI port data direction register */
#define SPI_SCK LCD_SCL /* port pin used for SCK */
#define SPI_MOSI LCD_SDA /* port pin used for MOSI */
```

### Листинг 7.7. Цветной LCD модуль

Если вы хотите, чтобы тестер запускался с пустым дисплеем, удалите знак комментария перед LCD\_LATE\_ON.

```
#define LCD_DOTS_X 160 /* number of horizontal dots */
#define LCD_DOTS_Y 128 /* number of vertical dots */
#define LCD_LATE_ON /* turn on LCD after clearing it */
#define FONT_8X8_HF /* 8x8 font */
#define SYMBOLS_30X32_HF /* 30x32 symbols */
```

### Листинг 7.8. полусовместимый дисплей ST7735

Смещение компенсации индуктивности для 20МГц модели

- предоставлено indman@EEVBlog

- Раздел для режима высокого тока в функции MeasureInductor() в inductor.c

```
#if CPU_FREQ == 20000000
/* 20 MHz */
if (Temp < 1500) /* < 1.5us / < 100uH */
{
Offset = -10;
}
else if (Temp < 5000) /* 1.5-5us / 100-330uH */
{
Offset = -10;
}
else /* > 5us / > 330uH */
{
Offset = -30;
}
}
#endif
```

### Листинг 7.9. Смещение компенсации индуктивности для модели 20 МГц

### 7.3. BSide ESR02 (DTU-1701)

- ATmega 328, тактовая частота 8 МГц
- Дисплей ST7565 (бит-банг-SPI)
- Внешнее опорное напряжение 2,5 В (TL431)
- предоставлено indman@EEVBlog

```
#define HW_REF25
```

Листинг 7.10. Параметры оборудования

```
#define BAT_DIVIDER
#define BAT_EXT_UNMONITORED
#define BAT_R1 47000
#define BAT_R2 47000
#define BAT_OFFSET 420
```

Листинг 7.11. управление производительностью

```
#define LCD_ST7565R /* display controller ST7565R */
#define LCD_GRAPHIC /* graphic display */
#define LCD_SPI /* SPI interface */
#define LCD_PORT PORTD /* port data register */
#define LCD_DDR DDRD /* port data direction register */
#define LCD_RESET PD0 /* port pin used for /RES (optional) */
#define LCD_A0 PD1 /* port pin used for A0 */
#define LCD_SCL PD2 /* port pin used for SCL */
#define LCD_SI PD3 /* port pin used for SI (LCD's data input) */
#define LCD_DOTS_X 128 /* number of horizontal dots */
#define LCD_DOTS_Y 64 /* number of vertical dots */
#define LCD_OFFSET_X /* enable x offset of 4 dots */
#define LCD_FLIP_X /* enable horizontal flip */
#define LCD_FLIP_Y /* enable vertical flip */
#define LCD_START_Y 0 /* start line (0-63) */
#define LCD_CONTRAST 15 /* default contrast (0-63) */
#define FONT_8X8_VF /* 8x8 font */
#define SYMBOLS_24X24_VFP /* 24x24 symbols */
#define SPI_BITBANG /* bit-bang SPI */
#define SPI_PORT LCD_PORT /* SPI port data register */
#define SPI_DDR LCD_DDR /* SPI port data direction register */
#define SPI_SCK LCD_SCL /* port pin used for SCK */
#define SPI_MOSI LCD_SI /* port pin used for MOSI */
```

Листинг 7.12. ЖК-модуль

## 7.4. Fish8840 TFT

- ATmega328, тактовая частота 8 МГц; цветной дисплей ST7565 (бит-банг-SPI)
- Внешнее опорное напряжение 2,5 В (TL431)
- Настройки от indman@EEVBlog/bdk100@vrtp.ru

```
#define LCD_ST7735
#define LCD_GRAPHIC /* graphic display */
#define LCD_COLOR /* color display */
#define LCD_SPI /* SPI interface */
#define LCD_PORT PORTD /* port data register */
#define LCD_DDR DDRD /* port data direction register */
#define LCD_RES PD3 /* port pin used for /RESX (optional) */
// #define LCD_CS PD5 /* port pin used for /CSX (optional) */
#define LCD_DC PD2 /* port pin used for D/CX */
#define LCD_SCL PD0 /* port pin used for SCL */
#define LCD_SDA PD1 /* port pin used for SDA */
#define LCD_DOTS_X 128 /* number of horizontal dots */
#define LCD_DOTS_Y 156 /* number of vertical dots */
#define LCD_OFFSET_X /* enable x offset of 4 dots */
#define LCD_OFFSET_Y /* enable y offset of 2 dots */
#define LCD_FLIP_X /* enable horizontal flip */
// #define LCD_FLIP_Y /* enable vertical flip */
#define LCD_ROTATE /* switch X and Y (rotate by 90Grad) */
#define LCD_LATE_ON /* turn on LCD after clearing it */
#define FONT_10X16_HF /* 10x16 font */
#define SYMBOLS_30X32_HF /* 30x32 symbols */
#define SPI_BITBANG /* bit-bang SPI */
#define SPI_PORT LCD_PORT /* SPI port data register */
#define SPI_DDR LCD_DDR /* SPI port data direction register */
#define SPI_SCK LCD_SCL /* port pin used for SCK */
#define SPI_MOSI LCD_SDA /* port pin used for MOSI */
```

Листинг 7.13. цветной дисплей

## 7.5. GM328 **!не GM328A!**

... которая описана выше как 7.2 „АУ-АТ“.

- ATmega328, тактовая частота 8 МГц; дисплей ST7565 (бит-банг-SPI)
- Параметры предоставлены rddube@EEVblog

```
#define LCD_ST7565R
#define LCD_GRAPHIC /* graphic display */
#define LCD_SPI /* SPI interface */
#define LCD_PORT PORTD /* port data register */
#define LCD_DDR DDRD /* port data direction register */
#define LCD_RESET PD0 /* port pin used for /RES (optional) */
#define LCD_A0 PD1 /* port pin used for A0 */
#define LCD_SCL PD2 /* port pin used for SCL */
#define LCD_SI PD3 /* port pin used for SI (LCD's data input) */
#define LCD_CS PD5 /* port pin used for /CS1 (optional) */
#define LCD_DOTS_X 128 /* number of horizontal dots */
#define LCD_DOTS_Y 64 /* number of vertical dots */
#define LCD_START_Y 0 /* start line (0-63) */
#define LCD_CONTRAST 11 /* default contrast (0-63) */
#define FONT_8X8_VF /* 8x8 font */
#define SYMBOLS_24X24_VFP /* 24x24 symbols */
#define SPI_BITBANG /* bit-bang SPI */
#define SPI_PORT LCD_PORT /* SPI port data register */
#define SPI_DDR LCD_DDR /* SPI port data direction register */
#define SPI_SCK LCD_SCL /* port pin used for SCK */
#define SPI_MOSI LCD_SI /* port pin used for MOSI */
```

Листинг 7.14. цветной дисплей

## 7.6. Hiland M644

- ATmega 644, тактовая частота 8 МГц; дисплей ST7565 (бит-банг-SPI)
- Поворотный энкодер (PB7 и PB5, параллельно дисплею)
- Внешнее опорное напряжение 2,5 В (TL431)
- Буст-конвертер для Zenertest
- Расширенный счетчик частоты; (но нет входного буферного каскада для прямого измерения частоты)
- Фиксированный конденсатор для самонастройки; (в случае проблем замените MLCC на фольгированный конденсатор 220nF)
- Настройки Хорста О. (obelix2007@mikrocontroller.net)

```
#define HW_ENCODER
#define ENCODER_PULSES 4 /* 4 */
#define HW_REF25
#define HW_ZENER
#define HW_FREQ_COUNTER_EXT
#define FREQ_COUNTER_PRESCALER 16 /* 16:1 */
#define HW_ADJUST_CAP
```

Листинг 7.15. Параметры оборудования

```
#define NO_HFE_C_RL /* if hFE values too high */
```

Листинг 7.16. Обходные пути! Установите NO HFE C RL!

```
#define LCD_ST7565R
#define LCD_GRAPHIC /* graphic display */
#define LCD_SPI /* SPI interface */
#define LCD_PORT PORTB /* port data register */
#define LCD_DDR DDRB /* port data direction register */
#define LCD_RESET PB4 /* port pin used for /RES (optional) */
// #define LCD_CS PB2 /* port pin used for /CS1 (optional) */
#define LCD_A0 PB5 /* port pin used for A0 */
#define LCD_SCL PB6 /* port pin used for SCL */
#define LCD_SI PB7 /* port pin used for SI (LCD's data input) */
#define LCD_DOTS_X 128 /* number of horizontal dots */
#define LCD_DOTS_Y 64 /* number of vertical dots */
// #define LCD_OFFSET_X /* enable x offset of 4 dots */
// #define LCD_FLIP_X /* enable horizontal flip */
#define LCD_FLIP_Y /* enable vertical flip */
#define LCD_START_Y 0 /* start line (0-63) */
#define LCD_CONTRAST 3 /* default contrast (0-63) */
#define FONT_8X8_VF /* 8x8 font */
#define SYMBOLS_24X24_VFP /* 24x24 symbols */
#define SPI_BITBANG /* bit-bang SPI */
#define SPI_PORT LCD_PORT /* SPI port data register */
#define SPI_DDR LCD_DDR /* SPI port data direction register */
#define SPI_SCK LCD_SCL /* port pin used for SCK */
#define SPI_MOSI LCD_SI /* port pin used for MOSI */
```

Листинг 7.17. дисплей

```
#define POWER_CTRL PB1 /* controls power (1: on / 0: off) */
```

Листинг 7.18. управление производительностью

```
#define ENCODER_A PB5 /* rotary encoder A signal */
#define ENCODER_B PB7 /* rotary encoder B signal */
```

Листинг 7.19. Поворотный энкодер

```
#define KEY_INC PB5 /* increase push button (low active) */
#define KEY_DEC PB7 /* decrease push button (low active) */
```

Листинг 7.20. Назначение выводов для направления вверх/вниз

```
#define ADJUST_RH PC6 /* Rh (470k) for fixed cap */
```

Листинг 7.21. Назначение выводов для фиксированного конденсатора для самонастройки

## 7.7. M12864 DIY Тестер транзисторов

- ATmega328, тактовая частота 8 МГц
- Дисплей ST7565 (бит-банг-SPI)
- Поворотный энкодер (PD1 и PD3, параллельно дисплею)
- Внешнее опорное напряжение 2,5 В (TL431)

```
#define HW_ENCODER
#define ENCODER_PULSES 4 /* not confirmed yet, could be also 2 */
#define ENCODER_STEPS 24 /* not confirmed yet */
#define HW_REF25
```

Листинг 7.22. Параметры оборудования

```
#define LCD_ST7565R
#define LCD_GRAPHIC /* graphic display */
#define LCD_SPI /* SPI interface */
#define LCD_PORT PORTD /* port data register */
#define LCD_DDR DDRD /* port data direction register */
#define LCD_RESET PD0 /* port pin used for /RES */
#define LCD_A0 PD1 /* port pin used for A0 */
#define LCD_SCL PD2 /* port pin used for SCL */
#define LCD_SI PD3 /* port pin used for SI (LCD's data input) */
#define LCD_DOTS_X 128 /* number of horizontal dots */
#define LCD_DOTS_Y 64 /* number of vertical dots */
// #define LCD_OFFSET_X /* enable x offset of 4 dots */
#define LCD_FLIP_Y /* enable vertical flip */
#define LCD_START_Y 0 /* start line (0-63) */
#define LCD_CONTRAST 11 /* default contrast (0-63) */
#define FONT_8X8_VF /* 8x8 font */
#define SYMBOLS_24X24_VFP /* 24x24 symbols */
#define SPI_BITBANG /* bit-bang SPI */
#define SPI_PORT LCD_PORT /* SPI port data register */
#define SPI_DDR LCD_DDR /* SPI port data direction register */
#define SPI_SCK LCD_SCL /* port pin used for SCK */
#define SPI_MOSI LCD_SI /* port pin used for MOSI */
```

Листинг 7.23. LCD-Modul

```
#define ENCODER_PORT PORTD /* port data register */
#define ENCODER_DDR DDRD /* port data direction register */
#define ENCODER_PIN PIND /* port input pins register */
#define ENCODER_A PD3 /* rotary encoder A signal */
#define ENCODER_B PD1 /* rotary encoder B signal */
```

Листинг 7.24. Поворотный энкодер

```
#define KEY_PORT PORTD /* port data register */
#define KEY_DDR DDRD /* port data direction register */
#define KEY_PIN PIND /* port input pins register */
#define KEY_INC PD3 /* increase push button (low active) */
#define KEY_DEC PD1 /* decrease push button (low active) */
```

Листинг 7.25. Назначение выводов для направления вверх/вниз



## 7.8. МК-328

- ATmega328, тактовая частота 8 МГц; цветной дисплей ST7565 (бит-банг-SPI)
- Внешнее опорное напряжение 2,5 В (TL431)
- Настройки от brunosso@EEVblog и подтверждено Peeps@EEVblog

```
#define HW_REF25
```

Листинг 7.26. Параметры оборудования

```
#define LCD_ST7565R
#define LCD_GRAPHIC /* graphic display */
#define LCD_SPI /* SPI interface */
#define LCD_PORT PORTD /* port data register */
#define LCD_DDR DDRD /* port data direction register */
#define LCD_RESET PD0 /* port pin used for /RES (optional) */
#define LCD_CS PD5 /* port pin used for /CS1 (optional) */
#define LCD_A0 PD1 /* port pin used for A0 */
#define LCD_SCL PD2 /* port pin used for SCL */
#define LCD_SI PD3 /* port pin used for SI (LCD's data input) */
#define LCD_DOTS_X 128 /* number of horizontal dots */
#define LCD_DOTS_Y 64 /* number of vertical dots */
#define LCD_START_Y 0 /* start line (0-63) */
#define LCD_CONTRAST 11 /* default contrast (0-63) */
#define FONT_8X8_VF /* 8x8 font */
#define SYMBOLS_24X24_VFP /* 24x24 symbols */
#define SPI_BITBANG /* bit-bang SPI */
#define SPI_PORT LCD_PORT /* SPI port data register */
#define SPI_DDR LCD_DDR /* SPI port data direction register */
#define SPI_SCK LCD_SCL /* port pin used for SCK */
#define SPI_MOSI LCD_SI /* port pin used for MOSI */
```

## 7.9. Т3/Т4

Листинг 7.27. цветной дисплей

- ATmega328, тактовая частота 8 МГц; Дисплей ST7565 (бит-банг SPI)
- Параметры предоставлены tom666@EEVblog

```
#define LCD_ST7565R
#define LCD_GRAPHIC /* graphic display */
#define LCD_SPI /* SPI interface */
#define LCD_PORT PORTD /* port data register */
#define LCD_DDR DDRD /* port data direction register */
#define LCD_RESET PD4 /* port pin used for /RES */
#define LCD_A0 PD3 /* port pin used for A0 */
#define LCD_SCL PD2 /* port pin used for SCL */
#define LCD_SI PD1 /* port pin used for SI (LCD's data input) */
#define LCD_CS PD5 /* port pin used for /CS1 (optional) */
#define LCD_DOTS_X 128 /* number of horizontal dots */
#define LCD_DOTS_Y 64 /* number of vertical dots */
#define LCD_START_Y 0 /* start line (0-63) */
#define LCD_CONTRAST 11 /* default contrast (0-63) */
#define FONT_8X8_VF /* 8x8 font */
#define SYMBOLS_24X24_VFP /* 24x24 symbols */
#define SPI_BITBANG /* bit-bang SPI */
#define SPI_PORT LCD_PORT /* SPI port data register */
#define SPI_DDR LCD_DDR /* SPI port data direction register */
#define SPI_SCK LCD_SCL /* port pin used for SCK */
#define SPI_MOSI LCD_SI /* port pin used for MOSI */
```

Листинг 7.28. LCD-модуль

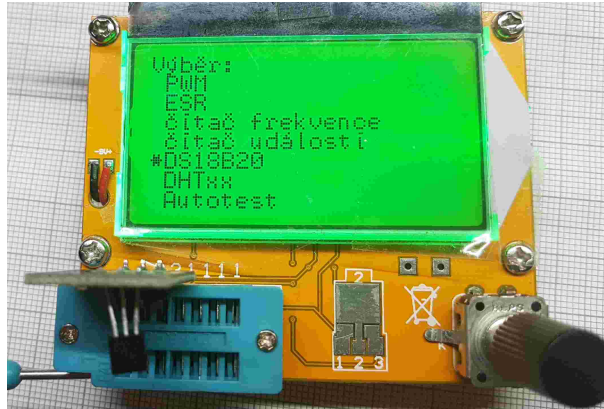
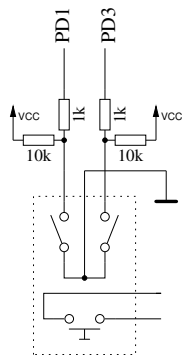
Существуют варианты Т4 с разным расположением выводов:

```
#define LCD_RESET PD0 /* port pin used for /RES */
#define LCD_A0 PD1 /* port pin used for A0 */
#define LCD_SCL PD2 /* port pin used for SCL */
#define LCD_SI PD3 /* port pin used for SI (LCD's data input) */
#define LCD_CS PD5 /* port pin used for /CS1 (optional) */
```

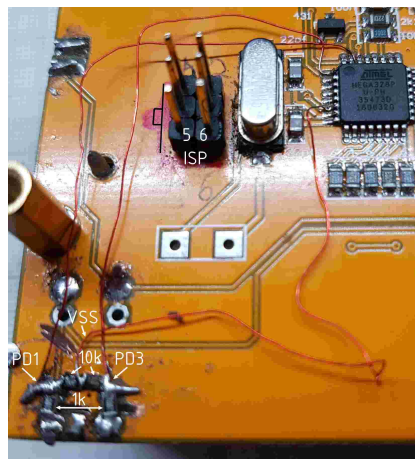
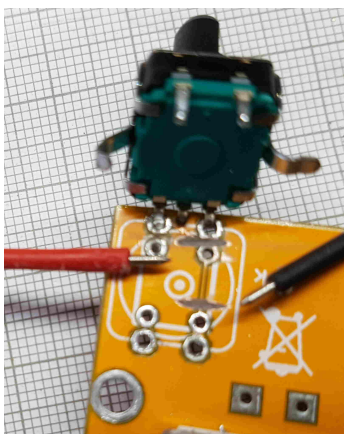
Листинг 7.29. ЖК-модуль с разным расположением выводов



**И перестроенный вариант** с поворотным кругом, в честь Karl-Heinz Kübbeler [4] которая предлагает гораздо больше возможностей, чем версия с “только” одной кнопкой:



(a) Схема из книги Кхк (b) Конвертированный T4 со шрифтом 6x8 ISO8859\_2 CZ.  
Рис. 7.1. T4 с поворотным датчиком Эксплуатация.



(a) Прерывания трассировки Передняя сторона

(b) и на спине

Рис. 7.2. Конверсия T4. Поворотный энкодер и ISP

Рабочий пример конфигурации:

```
#define HW_ENCODER
#define ENCODER_PULSES 4
#define ENCODER_STEPS 24
#define HW_REF25
#define UREF_25 2495
#define HW_FREQ_COUNTER_BASIC
#define HW_EVENT_COUNTER
#define EVENT_COUNTER_TRIGGER_OUT
#define SW_PWM_SIMPLE
#define SW_INDUCTOR
#define SW_ESR
#define SW_DS18B20
#define SW_REVERSE_HFE
#define SW_DHTXX
#define UI_COMMA
#define UI_AUTOHOLD
#define UI_KEY_HINTS
#define POWER_OFF_TIMEOUT 30
#define SW_POWER_OFF
#define UI_ROUND_DS18B20
#define ONEWIRE_PROBES
```

Листинг 7.30. Используемые параметры в config.h

## 7.10. Многофункциональный тестер ТС-1 и семейство (Т7)

- ATmega324 (очень плохая распиновка); тактовая частота 16 МГц;
- цветной дисплей ST7735 (бит-банг SPI)
- внешнее опорное напряжение 2,5 В (TL431)
- модуль фиксированного ИК-приемника; повышающий преобразователь для Zenertest (работает постоянно, не стандартный делитель напряжения 100 кΩ/12 кΩ)
- Фиксированный конденсатор для самонастройки (в случае проблем замените MLCC на пленочный конденсатор 220 нФ)
- Питание от литий-ионного аккумулятора 3.7 В
- тестовые копии от jellytot@EEVblog и joystick@EEVblog
- первая информация от indman@EEVblog

**Примечания:** Управляющий MCU U4 (STC15L104W) должен быть заменен простой двухтранзисторной схемой.

-(ТС1-Mod, см. исходный репозиторий для аппаратного обеспечения/Markus/ TC1-Mod.kicad.tgz, резервный 5uA ток в целом) или перепрограммированы с измененной прошивкой (см. <https://github.com/atar-axis/tc1-u4>).

Идентификатор также может быть U3 (T7 Plus) или U5 для более новых плат.

- Установите байт расширенного предохранителя на 0xfd (обнаружение коричневого отключения).
- Если D2 (выпрямительный диод для испытательного напряжения Зенера) нагревается, замените его на другой. Диод Шоттки, рассчитанный на обратное напряжение 80 В или выше, например, SS18.
- Замените C11 и C12 (фильтрующие колпачки фильтра испытательного напряжения Zener) на 10 или 22 мкФ. Электролитический колпачок с низким ESR, рассчитанный на напряжение 100 В или выше, благодаря постоянному току MLCC. Проблема снижения мощности входной емкости.
- В зависимости от используемого вами ЖК-модуля, вам может потребоваться вместо этого LCD\_FLIP\_X установить LCD\_FLIP\_Y.
- ТС-1 не может обеспечить выход сигнала (ШИМ / квадратная волна / и т.д.) К зонду №2. Используйте PD4 (OC1B) в качестве специального сигнального выхода (добавьте резистор для ограничения тока) и Включите HW\_FIXED\_SIGNAL\_OUTPUT в config.h.
- Если вы хотите добавить поворотный регулятор или кнопки включения/выключения, пожалуйста, используйте PB5 (дисплей D/C) и PB6 (дисплей SDA).
- Вы также можете получить счетчик частоты, используя PB0 (T0) в качестве входа и добавив простой входной каскад.
- Перемычка PD0 (не используется микропрограммой m)

```
#define HW_REF25
#define HW_ZENER
#define ZENER_DIVIDER_CUSTOM
#define ZENER_R1 100000
#define ZENER_R2 12000
#define ZENER_UNSWITCHED
#define HW_IR_RECEIVER
#define HW_ADJUST_CAP
```

Листинг 7.31. Параметры оборудования

```
#define HW_PROBE_ZENER
#define ZENER_VOLTAGE_MIN 1000 /* min. voltage in mV */
#define ZENER_VOLTAGE_MAX 40000 /* max. voltage in mV */
```

Листинг 7.32. Различные настройки

```
#define IR_DATA PD3 /* data signal */
```

Листинг 7.33. ИК-детектор/декодер

```
#define ADJUST_RH PC6 /* Rh (470k) for fixed cap */
```

Листинг 7.34. Фиксированный конденсатор для самонастройки

```
#define TP_ZENER PA4 /* test pin with 10:1 voltage divider */
#define TP_REF PA3 /* test pin with 2.5V reference */
#define TP_BAT PA5 /* test pin with 4:1 voltage divider */
#define TP_CAP PA7 /* test pin for self-adjustment cap */
```

Листинг 7.35. Назначение клемм для испытательных щупов

```
#define R_RL_1 PC0 /* Rl (680R) for test pin #1 */
#define R_RH_1 PC1 /* Rh (470k) for test pin #1 */
#define R_RL_2 PC2 /* Rl (680R) for test pin #2 */
#define R_RH_2 PC3 /* Rh (470k) for test pin #2 */
#define R_RL_3 PC4 /* Rl (680R) for test pin #3 */
#define R_RH_3 PC5 /* Rh (470k) for test pin #3 */
```

Листинг 7.36. Назначение выводов для резисторов датчика

```
#define POWER_CTRL PD2 /* controls power (1: on / 0: off) */
```

Листинг 7.37. управление производительностью

```
#define TEST_BUTTON PD1 /* test/start push button (low active) */
```

Листинг 7.38. Назначение выводов для кнопки тестирования

```
#define LCD_ST7735
#define LCD_COLOR /* color graphic display */
#define LCD_SPI /* SPI interface */
#define LCD_PORT PORTB /* port data register */
#define LCD_DDR DDRB /* port data direction register */
#define LCD_RES PB4 /* port pin used for /RESX (optional) */
// #define LCD_CS PB? /* port pin used for /CSX (optional) */
#define LCD_DC PB5 /* port pin used for D/CX */
#define LCD_SCL PB7 /* port pin used for SCL */
#define LCD_SDA PB6 /* port pin used for SDA */
#define LCD_DOTS_X 128 /* number of horizontal dots */
#define LCD_DOTS_Y 160 /* number of vertical dots */
#define LCD_OFFSET_X 2 /* enable x offset of 2 or 4 dots */
#define LCD_OFFSET_Y 1 /* enable y offset of 1 or 2 dots */
// #define LCD_FLIP_X /* enable horizontal flip */
#define LCD_FLIP_Y /* enable vertical flip */
#define LCD_ROTATE /* switch X and Y (rotate by 90Grad) */
#define LCD_LATE_ON /* turn on LCD after clearing it */
#define FONT_10X16_HF /* 10x16 font */
#define SYMBOLS_30X32_HF /* 30x32 symbols */
#define SPI_BITBANG /* bit-bang SPI */
#define SPI_PORT LCD_PORT /* SPI port data register */
#define SPI_DDR LCD_DDR /* SPI port data direction register */
#define SPI_SCK LCD_SCL /* port pin used for SCK */
#define SPI_MOSI LCD_SDA /* port pin used for MOSI */
```

Листинг 7.39. цветной дисплей

## 7.11. Разъем Arduino MEGA

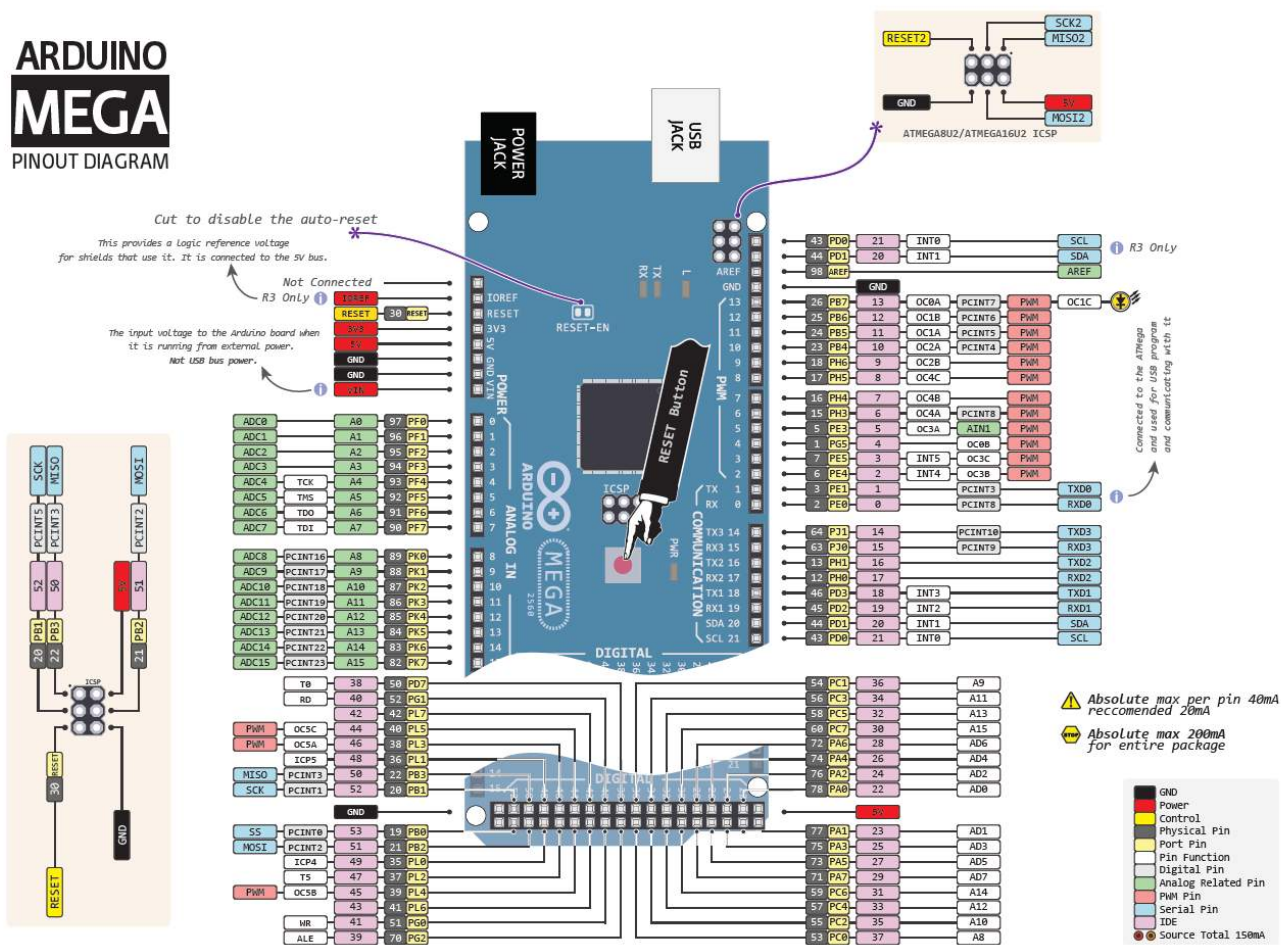


Рис. 7.3. Arduino Mega

Это изображение скопировано с сайта: <https://duino4projects.com/arduino-mega-pinout-diagram/>

Чтобы избавить других коллег от отчаяния и "бессонных ночей" которые пережил автор этой главы после того, как приобрел клон-тестер без опыта работы с AVR и захотел "научить" его немецкому языку, была написана эта глава.

Опыт, приобретенный здесь, должен помочь другим "желающим не имеющим опыта, чтобы УСПЕШНО запрограммировать свой тестер.

Эта возможность используется для того, чтобы поблагодарить автора и разработчика Транзистортестера Карл-Хайнц Кюббелер [3] за его приверженность и терпение, потому что следующие страницы никогда не были бы написаны без его помощи.

Чтобы перевод прошивки и запись в MCU прошли успешно и в то же время . . .

"Колесо не придется изобретать заново часть следующих страниц была взята из описание тестера транзисторов Карла-Хайнца Кюббелера см. на сайте [3].

Так что еще раз . . . **Огромное спасибо.**

### 8.1. Настройка тестера компонентов

Для этого, пожалуйста, прочитайте главу 1.8 на странице 9.

### 8.2. программирование микроконтроллера

Программирование тестера контролируется файлом Makefile.

Makefile гарантирует, что программное обеспечение будет скомпилировано в соответствии с опциями, установленными ранее в Makefile. в соответствии с опциями, установленными ранее в Makefile.

Результат перевода имеет расширение файла .hex и .eep.

Обычно файлы называются ComponentTester.hex и ComponentTester.eep.

Файл .hex содержит данные для памяти программ (флэш-памяти) процессора ATmega.

Файл .eep содержит данные для EEPROM устройства ATmega. Оба файла должны быть загружены в нужную память.

Кроме того, ATmega должна быть правильно сконфигурирована с предохранителями.

Если вы используете Makefile вместе с программой avrdude [9], вам не нужно вам не нужно знать детали предохранителей.

Вам нужно вызвать "make fuses только если вы не используете кристалл, или вызвать "make fuses-crystal". необходимо вызвать команду "make fuses-crystal если на плате установлен кристалл 8MHz.

Если вы не уверены в предохранителях, оставьте их такими, какими они были установлены на заводе, и с завода и запустить тестер в этом состоянии. Может оказаться, что программа работает слишком медленно, если вы используете данные программы, сгенерированные для 8-операции, но это операцию, но это можно исправить позже!

Но неправильно установленные предохранители могут помешать программированию ISP в дальнейшем.

### 8.3. Операционная система Linux

Программирование под Linux дает много преимуществ, поскольку эта ОС была разработана специалистами, ориентирующимися на пожелания пользователей. Кроме того, среда предоставляется бесплатно и прекрасно поддерживается.

Еще одним преимуществом является безопасность самой ОС, а также при использовании Интернета. Текущие редакции намного проще в использовании, чем конкурирующие ОС. Это руководство должно побудить всех "не"пользователей Linux протестировать его СЕЙЧАС, запрограммировав на нем свой тестер.



В качестве примера мы используем Linux Mint в текущей версии, которая доступна в интернете. Установка возможна различными способами.

## 8.4. Использование в Linux

на вновь установленной ОС.

Для тех, кто не любит писать, Linux предлагает более простой способ.

Скопируйте данное руководство на USB-накопитель и откройте его в Linux. Затем подведите мышь к названию документа, нажмите здесь левую кнопку мыши и перетащите документ к левому краю экрана, пока не появится возможная рамка. Теперь отпустите мышь.

Теперь документ занимает левую половину экрана.

Следующим шагом будет одновременное нажатие **Ctrl**+**Alt**+**t**, чтобы открыть окно команд. Теперь он перемещен таким же образом к правому краю экрана.

## 8.5. установка пакетов программ

Теперь нам нужен доступ в Интернет.

Чтобы запрограммировать тестер, нам сначала нужно установить пакеты программ:

'binutils-avr', 'avrdude', 'avr-libc' и 'gcc-avr'. Далее система контроля версий 'git'.

На данный момент перейдите на эту страницу в этом документе, до этого текста:

```
sudo apt-get install avrdude avr-libc binutils-avr gcc-avr git
```

Выделите текст при нажатой левой кнопке мыши, подведите курсор мыши к правой командное окно и вставьте **более сокращенно** **БС**, нажав среднюю кнопку мыши (колесо прокрутки).

После подтверждения **↵**, 'sudo' запросит пароль пользователя.

Это позволит загрузить и установить все пакеты программного обеспечения через 'apt'.

Вам может понадобиться подтвердить вопросы между ними с помощью **Я**.

Пожалуйста, обратите внимание, что Linux чувствителен к регистру.

Согда не отвечаешь на вопрос **я**, отвечай на вопрос **Я**!

Согда не отвечаешь на вопрос **Я**. Вы можете проверить, была ли система контроля версий 'git' успешно установлена, с помощью команды:

```
git version
```

командование. В ответ программа должна вывести номер своей версии.

## 8.6. скачать источники

и документации из архива git достигается с помощью заявления:

```
git clone https://github.com/Mikrocontroller-net/transistortester
```

Теперь файлы находятся в [личной папке] Linux на (/home/„user“) под именем „transistortester“.

Проверка наличия. Откройте окно терминала,

```
ls
```

 и подтвердите клавишей **↵**.

Для загрузки новых обновлений достаточно в будущем:

```
cd ~/tra
```

затем **tab** и введите **↵**, и теперь в этой директории

```
git pull
```

затем **↵**.

### Примечание:

*В случае возникновения проблем или сообщений об ошибках, самый простой способ - удалить или переименовать существующую папку 'transistortester', а затем скачать ее снова, используя вышеуказанную инструкцию.*

## 8.7. Использование интерфейсов

подготовить для пользователя.

USB-устройства можно обнаружить, введя 'lsusb' в командном окне. Введите 'lsusb' сначала без, а затем с подключенным USB-программатором.

Сравнение результатов позволит определить местонахождение USB-программатора.

Результат работы lsusb может выглядеть следующим образом:

```
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 003: ID 046d:c050 Logitech, Inc. RX 250 Optical Mouse
Bus 002 Device 058: ID 03eb:2104 Atmel Corp. AVR ISP mkII
Bus 002 Device 059: ID 2341:0042 Arduino SA Mega 2560 R3 (CDC ACM)
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub}
```

Здесь AVR ISP mkII был обнаружен как устройство 58 (DIAMEX ALL-AVR).

Идентификатор 03eb является идентификатором производителя, а идентификатор 2104 - идентификатором продукта.

Эти два идентификатора требуются для файла `/etc/udev/rules.d/90-atmel.rules` и создаются с помощью:

```
sudo xed /etc/udev/rules.d/90-atmel.rules
```

В данном примере файл `90-atmel.rules` состоит из одной строки:

```
SUBSYSTEM=="usb", ATTRS{idVendor}=="03eb", ATTRS{idProduct}=="2104", MODE="0660",
GROUP="plugdev"
```

Эта запись разрешает доступ к устройству членам группы `"plugdev"`.

Чтобы иметь возможность использовать известных программистов, рекомендуется следующий текст в `90-atmel.rules`:

```
Copy this file to /etc/udev/rules.d/90-atmel.rules
AVR ISP mkII - DIAMEX ALL-AVR
SUBSYSTEM=="usb", ATTRS {idVendor}=="03eb", ATTS {idProduct}=="2104", MODE="0660",
 GROUP = "plugdev",
USB ISP-programmer für Atmel AVR
SUBSYSTEM=="usb", ENV {DEVTYPE}=="usb_device", SYSFS {idVendor}=="16c0", MODE="0666",
 SYSFS {idProduct} == "05dc",
USB asp programmer
ATTRS {idVendor}=="16c0", ATTRS {idProduct}=="05dc", GROUP="plugdev", MODE="0660"
USBtiny programmer
ATTRS {idVendor}=="1781", ATTRS {idProduct}=="0c9f", GROUP="plugdev", MODE="0660"
Pololu programmer
SUBSYSTEM=="usb", ATTRS {idVendor}=="1ffb", MODE="0666"
```

После создания файла можно проверить его создание и содержание:

```
less /etc/udev/rules.d/90-atmel.rules
```

USB-устройство Arduino SA Mega 2560 System, также определяемое как Устройство 59, создает доступ к последовательному устройству „`/dev/ttyACM0`” для членов группы `'dialout'`.

## 8.8. Членство в группе

Поэтому ваш идентификатор пользователя должен быть членом как группы `'plugdev'`, так и группы группы `'dialout'`. Команда:

```
sudo usermod -a -G dialout,plugdev $USER
```

должны убедиться, что вы принадлежите к этой группе. Теперь вы должны иметь доступ к обоим устройствам с помощью `avrdude`. Вы можете проверить это с помощью команды `'id'`.

Если у вас возникли проблемы, вы также можете проверить членство с:

`меню` `» управление системой` `» пользователи и группы` `» ?пароль`, теперь появится окно с двумя вкладками.

Если вы нажмете на свое имя во вкладке `Пользователи`, вы увидите свой профиль и членство в группах справа. С помощью кнопки `Add` теперь можно добавлять новые группы.

## 8.9. рабочая среда

Подготовка. Сначала перейдите к `/transistortester/Software/Markus/` на панели задач с зеленым значком папки (Nemo), щелкните правой кнопкой мыши на `ComponentTester-1.(наивысший номер)m.tgz`


щелкните, в выборе `<декомпрессировать здесь>` папку распаковать и закрыть Nemo.

Для сохранения оригинальности и потому, что окно терминала всегда открывается в `../home/“user“`, рекомендуется поместить туда его рабочий каталог с именем **Mytester**.

Используя уже известный метод, выберите следующий каталог и вставьте его в окно терминала с помощью средней клавиши.




```
cd ~/transistortester/Software/Markus/
```

После подтверждения и команды: 'ls' вы увидите все упакованные папки (с окончанием .tgz) и только одну папку, где это окончание отсутствует -> таким образом, наша (ранее распакованная) папка .

Для следующих двух команд сначала вставьте **только, без** : !

```
cp -r 'MyT' Mytester/
```

Отметьте каталог нужной модели с помощью мыши.

Теперь установите мигающий курсор на последний символ текста 'MyT' с помощью  и удалите эти символы. После удаления последнего символа нажмите клавишу  мыши. Только теперь используйте . Теперь создается рабочая среда. Проверить наличие и содержание можно с помощью:







```
diff 'MyT' Mytester/
```

где 'MyT', как и ранее, должен быть заменен на каталог "требуемой модели тестера".

С последним заявлением:



```
ln -s ~/transistortester/software/Markus/Mytester ~/Mytester
```

будет создана ссылка на рабочий каталог. С этого момента вы можете легко попасть в этот каталог:

```
++, cd  My  
```

и вы окажетесь в нужной вам директории. С помощью 'ls' вы можете увидеть содержимое.

Продолжайте редактировать Makefile с уже известным утверждением:

```
xed Ma  
```

**Здесь самое главное - зарегистрировать имеющийся USB-программатор.**

См. главу 6.1.5, на странице 45, тема программатор.

## 8.10. создание и передача прошивки

После редактирования Makefile, config.h и config-<MCU>.h выполните команду „make“ или вашей IDE для компиляции прошивки. В результате будут созданы два файла:

- прошивка ComponentTester.hex в формате Intel hex
- ComponentTester.eep Данные EEPROM в формате Intel Hex.

Прошивка записывается во флэш-память, а данные EEPROM записываются в EEPROM. Данные содержат два набора стандартных значений корректировки, тексты и таблицы. Если вы хотите сохранить старые значения настройки в EEPROM при обновлении микропрограммы, вы можете использовать опцию вы можете использовать переключатель DATA\_FLASH в config.h, чтобы перенести тексты и таблицы в прошивку. тексты и таблицы в микропрограмму.

В этом случае во флэш-память необходимо записать только микропрограмму; EEPROM остается неизменным.

В Makefile предусмотрены следующие цели:

|              |                                                   |
|--------------|---------------------------------------------------|
| make clean   | удалить все файлы объектов                        |
| make         | для компиляции программы                          |
| make fuses   | установить биты предохранителей (через avrdude)   |
| make upload  | записать прошивку и данные EEPROM (через avrdude) |
| make prog_fw | записать только прошивку (через avrdude)          |
| make prog_ee | записать только данные EEPROM (через avrdude)     |

Теперь осталась только радость от достигнутого успеха.



### 9.1. v1.44m 2021-08

- Вывод кодов EIA-96 для стандартных значений сопротивления E96 SW\_E96\_EIA96, (предложение Chris02@EEVblog).
- Добавлена поддержка 4-проводного SPI в драйверах для ILI9481 и ILI9486.
- Исправлена проблема тайм-аута в драйвере ST7920. (по сообщению ralleiner@mikrocontroller.net).
- Изменение функции разряда: В случае возникновения проблемы разряда, напряжение без нагрузки на напряжение без нагрузки ИУ измеряется и отображается сообщением об ошибке разряда например, напряжение батареи без нагрузки.
- Символ витого диода для второго диода при антипараллельном выводе диоды (по сообщению hogo@EEVblog).
- Опция для малоактивной подсветки в драйвере HD44780 для адаптера PCF8574-I2C. добавлено LCD\_BACKLIGHT\_LOW, (предложено Andreas Deimel).
- Неправильные символы для р-канальных МОП-транзисторов в растровых картах символов 24x24 исправлено (по сообщению Peeps@EEVblog).

### 9.2. v1.43m 2021-03

- R&D драйвер дисплея для идентификации контроллера.
- Управляющий сигнал LCD\_RD теперь необязателен для ILI9341, ILI9481, ILI9486 и ILI9488.
- Модифицированный драйвер для совместимых с ST7735 дисплеев (предложенный b0hoon4@gmail.com).
- Опция запуска теста Zener во время обычного поиска деталей ( HW\_PROBE\_ZENER, предложение от indman@EEVblog). Требуется ZENER\_UNSWITCHED. И добавлена соответствующая дистанционная команда (V\_Z).
- Опция делителя напряжения для Zenertest для конкретного пользователя. (ZENER\_DIVIDER\_CUSTOM, ZENER\_R1 и ZENER\_R2).
- опцию для низкого разрешения при тестировании зенера, а также удалил опцию переключатель конфигурации ZENER\_HIGH\_RES.
- Обновленные русские тексты (спасибо indman@EEVblog).
- Дополнительное отображение длительности импульса для генераторов ШИМ (PWM\_SHOW\_DURATION, предложение от гостя „хамбургер“@mikrocontroller.net).
- архив исходных текстов теперь включает каталог „dep чтобы сделать некоторые IDE счастливым (предложение от DAIRVINE@EEVblog).
- Переименовать конфигурационный переключатель SW\_PROBE\_COLORS в UI\_PROBE\_COLORS.
- Заголовки могут отображаться другим цветом (UI\_COLORED\_TITLES). Также курсоры и подсказки по работе (UI\_COLORED\_CURSOR).
- Обновлены румынские тексты (спасибо Dumidan@EEVblog).
- Обновлены альтернативные польские тексты (спасибо Jacon@EEVblog).
- Испанские тексты обновлены (спасибо pepe10000@EEVblog).

### 9.3. v1.42m 2020-12

- Обновленный набор символов 6x8 (спасибо Bohu).
- Новый набор символов 6x8 в соответствии с ISO8859-2 (FONT\_6X8\_ISO8859\_2\_HF, спасибо Bohu).

- Новая аппаратная опция: LC-метр (HW\_LC\_METER). На основе простого схема с LC осциллятором, аналогичная дешевым наборам LC измерителей. комплекты.
  - Display\_Value() теперь также поддерживает Fempto (f).
  - Если CYCLE\_MAX установлен на 255, автоматическое выключение отключено и тестер будет работать до тех пор, пока его не выключат вручную.
  - Проблема с отображением странных частот в расширенном счетчике частот в определенном созвездии (по сообщению Szybkijjanek@EEVblog).
- Также индикация отсутствия сигнала или слишком низкой частоты изменилась с „0Hz“ на „-“.
- Простой счетчик частоты теперь показывает „-“ вместо „0Hz“ если нет сигнала или частота слишком низкая. присутствует сигнал или частота слишком низкая.
  - Опция для вывода испытательного тока I\_C или I\_E измерения hFE ( SW\_HFE\_CURRENT, предложено Obelix2007@EEVblog),
- и добавлены соответствующие удаленные команды (I\_C и I\_E).
- Романтические тексты (спасибо Dumidan@EEVblog).
  - Переключатель конфигурации для инструмента ISR (SW\_ESR\_TOOL, предложение от indman@EEVblog).
  - Текстовый вывод значений E-стандарта теперь также выводит единицы измерения (предложение от indman@EEVblog).
  - Обновлено альтернативные польские тексты (спасибо Jacon@EEVblog).
  - Последняя строка текста на дисплее просто перезаписывалась, если UI\_KEY\_HINTS is включен (сообщено Obelix2007@EEVblog).
  - Добавлена функция ожидания пользователя перед удалением последней строки.
  - Обновлено тексты на испанском языке (спасибо pere10000@EEVblog).
  - Исправлена ошибка в „#define“ для стандартных значений E96 (сообщено Obelix2007@EEVblog).
  - Драйвер для дисплея с ILI9488 8/16-битным параллельным и 4-проводным SPI, (спасибо Bohu для пробной копии).

#### 9.4. v1.41m 2020-09

- Новая команда дистанционного управления „MHINT“ для заметок об измерениях (в настоящее время только тип испытательной цепи для h\_FE).
- Добавлен авторежим (автоматическое обновление) для DS18B20. Предложение от (Obelix2007@EEVblog).
- Дополнительная функция конечной нагрузки для устранения проблемы ISR на некоторых Тестерклоны (по сообщению indman@EEVblog). Значение ESR было слишком высоким для Elko  $\geq 470$  мкФ на тестовых выводах #1 и #2, в ESR инструменте, C мониторе и RCL мониторе. А для электролитического конденсатора  $\geq 4700$  мкФ во всех режимах. Затронутые тестеры - Hiland M644 и Варианты TC-1.
- Обновлено знаки степеней в шрифте\_16x26\_hf.h и шрифте\_16x26\_iso8859-2\_hf.h (изменения внесены Bohu).
- Драйвер для контроллера OLED SH1106 (3-Wire SPI, 4-Wire SPI и I2C; благодаря Old-Papa для тестового показа).
- Отсутствие инициализации сигнала /RES для I2C в драйвере SSD1306 добавлено.
- Вывод типа тестовой схемы для hFE.
- Изменен вывод шестнадцатеричных значений в нижний регистр и добавлен переключатель для включения верхнего регистра (UI\_HEX\_UPPERCASE).
- Опция для функций OneWire для дополнительного считывания и отображения данных Код ПЗУ (ONEWIRE\_READ\_ROM, предложенный indman@EEVblog).
- Контрольные контакты DQ и Vcc для шины OneWire, соответствующие распиновке датчиков поменяны местами (предложение indman@EEVblog).
- Обнаружение транзистора Шоттки (SW\_SCHOTTKY\_BJT).
- Улучшение обнаружения типов истока FET с помощью фильтров для Транзисторы Шоттки.
- Оптимизированное управление текстовыми строками для очистки дисплея.
- Специальный драйвер дисплея для идентификации контроллеров дисплея.

- Монитор R/C/L (SW\_MONITOR\_RCL, предложено indman@EEVblog).
- Проблема с изменением значений индуктивности в мониторе L и R/L решена ( сообщает indman@EEVblog).
- Измерение V\_GS(off) для FET типа с обеднением (предложено joshto@EEVblog). Добавлена команда "V\_GS\_off" к командам дистанционного управления.
- Драйвер I19341 расширен для поддержки 8-битной параллельной шины. Также переключатель конфигурации для I19341 с деактивированным расширенным набор команд отключен (LCD\_EXT\_CMD\_OFF, спасибо Bohu за тестовую копию).
- Набор символов кириллицы 16x26 (FONT\_16X26\_WIN1251\_HF, благодаря Юрий\_K@VRTP.RU).
- Проблема с отсутствием  $\mu$  (микро) в нескольких наборах символов ISO8859-2 исправлено (сообщено indman@EEVblog и Obelix2007@EEVblog).
- Исправлена ошибка с числом для ATmega 2560 в Makefile (сообщено Bohu).
- Цветное состояние батареи для цветных дисплеев (предложение indman@EEVblog).
- Альтернативный набор символов 8x16 Win1251 с обновленной версией indman@EEVblog.
- Проблема с проверкой малых резисторов в CheckResistor() исправлена (сообщено indman@EEVblog).
- SmallResistor(), MeasureInductance() и GetGateThreshold() для поддержки обновлено для поддержки ADC\_LARGE\_BUFFER\_CAP.
- Отсутствующее обновление опорного источника АЦП в MeasureInductance() и GetGateThreshold() добавлено.
- Возможность выбора профиля настройки после включения питания UI\_CHOOSE\_PROFILE, (предложение от Боху).
- Специальная функция мониторинга резисторов и индукторов SW\_MONITOR\_R, SW\_MONITOR\_L, (предложение indman@EEVblog).
- Испанские тексты обновлены (спасибо pepe10000@EEVblog).

## 9.5. v1.40m 2020-06

- Драйвер для дисплеев с I19481 или I19486.
- Исправлена логическая ошибка для цветных дисплеев с отключенной функцией цвета.
- Добавление переключателя конфигурации для цветных дисплеев, чтобы поменять местами красный и синий цвета цветовые каналы для красного и синего (LCD-\_BGR).
- Улучшен монитор R/L для уменьшения разброса показаний. Помогает не во всех случаях. (сообщено indman@EEVblog).
- Проблема с "#ifdef" для функций Display\_HexByte() и Display\_HexDigit() исправлено (сообщено AlcidePiR2@EEVBlog).
- Поддержка ATmega 640/1280/2560.
- Определение направления вращения в ReadEncoder() было изменено на противоположное. Изменено на правильное направление и обновленные настройки в config\_<MCU>.h и клонах. обновлено.
- Исправлена проблема инициализации с аппаратным SPI.
- Альтернативный режим для проверки зенера, когда повышающий преобразователь работает постоянно или является не доступен (ZENER\_UNSWITCHED, (предложение от indman@EEVblog).
- переключатель конфигурации для буферного конденсатора AREF 100нФ вместо 1нФ (ADC\_LARGE\_BUFFER\_CAP). Требуется для некоторых плат MCU.
- Альтернативные кириллические шрифты 8x16 (FONT\_8X16ALT\_WIN1251\_HF) and 8x8 (FONT\_8X8ALT\_WIN1251\_VF, спасибо indman@EEVblog).
- Обновлены русские тексты (спасибо indman@EEVblog).
- Возможность использования простого переключателя вкл/выкл вместо стандартного стандартный выключатель с мягкой защелкой (POWER\_SWITCH\_MANUAL).
- Обнаружение двух коротких касаний центра в ReadTouchScreen() добавлено.
- ошибка алгоритма, касающаяся TOUCH\_FLIP\_X/TOUCH\_FLIP\_Y в Touch\_CharPos() исправлено (по сообщению Боху).

- Несколько наборов символов ISO8859-2 заменены обновленными версиями Bohu заменен.
- Опция проверки резисторов на соответствие стандартным значениям E (SW\_R\_E\*). Для конденсаторов (SW\_C\_E\*) и индукторов (SW\_L\_E\*).

## 9.6. v1.39m 2020-03

- Польские тексты в соответствии с ISO 8859-2 (спасибо Jason).c
- Проблема с макросом препроцессора в SPI.c и синтаксическая ошибка в ADS7843.c исправлено (по сообщению Боху).
- Опция для хранения данных прошивки во флэш-памяти вместо EEPROM DATA\_FLASH, (предложено Виталием).
- Кириллические кодировки переименованы в "win1251 а чешские кодировки заменены на шрифты заменены на шрифты ISO8859-2 (спасибо Bohu).
- Функция для вывода набора символов в целях тестирования (SW\_FONT\_TEST).
- Сканирование OneWire для вывода кодов ПЗУ устройств шины (SW\_ONEWIRE\_SCAN).
- Опция для специфического смещения сопротивления для каждой пары тестовых контактов (R\_MULTIOFFSET, предложенный Виталием).

## 9.7. v1.38m 2019-12

- Необязательное округление температуры для DS18B20 (UI\_ROUND\_DS18B20, предложение by Obelix2007@EEVblog)
- Поддержка DHT11, DHT22 и совместимых датчиков (SW\_DHTXX. Спасибо indman@EEVblog и Obelix2007@EEVblog за тестирование).
- Добавлены два тонких набора кириллических символов (спасибо Андрею@EEVblog).
- Изменен выход биполярных транзисторов, чтобы показывать V\_BE и hFE даже в том случае, если в случае резистора В-Е. Также команды дистанционного управления адаптированы соответствующим образом.
- Тексты на чешском языке плюс несколько шрифтов с чешскими символами (спасибо Bohu). персонажей (благодаря Боху).
- Функции для наблюдения за R/C/L (SW\_MONITOR\_RL и SW\_MONITOR\_C, предложено indman@EEVblog).
- Триггерный выход для счетчика событий (предложение Bohu).
- Обновлены тексты на чешском языке (спасибо Bohu).
- Измерение hFE при схеме с общим коллектором и Rl в качестве базового резистора может быть отключен (NO\_HFE\_C\_RL), чтобы некоторые тестеры не давали чрезмерные результаты (по сообщению Obelix2007@EEVblog).
- Опция вывода напряжения Зенера в высоком разрешении (ZENER\_HIGH\_RES, предложено Andbro@EEVblog).
- OneWire\_Probes() улучшена для минимизации ложных обнаружений.
- Обновлены русские тексты (спасибо indman@EEVblog).
- Обновленные испанские тексты (спасибо pepe10000@EEVblog).

## 9.8. v1.37m 2019-09

- Исправлена ошибка в DS18B20\_Tool(), когда ONEWIRE\_IO\_PIN включен (сообщено bm-magic).
- Решена проблема с отображением сообщения об ошибке сторожевого таймера на цветных дисплеях.
- Новая функция: счетчик событий (HW\_EVENT\_COUNTER, предложено bm-magic).
- Простой счетчик частоты теперь использует TestKey() для ввода данных пользователем. На дважды для выхода (раньше это было одно нажатие клавиши).
- Опция отображения инвертированного значения hFE транзисторов (SW\_REVERSE\_HFE, предложение towe96@EEVblog). Также удаленные команды расширяется командой "h\_FE\_r".
- Настройка битклока (BITCLOCK) для avrdude в Makefile (предложение от bm-magic).
- Проблема с обнаружением TRIAC в случае слишком высокого I\_GT в Q3 или слишком высокий I\_H исправлен (о проблеме I\_GT сообщил petroid).
- тексты Tester\_str, PWM\_str, Hertz\_str и CTR\_str перенесены в раздел, посвященный языку заголовочные файлы (предложение indman@EEVblog).

- Изменен вывод значений частоты (Герц) на фиксированную строку (ранее "H"). как единица для DisplayValue() плюс дополнительное "z".
- Опция отображения справки пользовательского интерфейса (UI\_KEY\_HINTS). В настоящее время только "Menu/Test" (предложение от carrascoso@EEVblog).
- Обновлено польские тексты (C szpila@EEVblog).
- русские тексты (спасибо indman@EEVblog).
- тексты на испанском языке (спасибо pere10000@EEVblog).

## 9.9. v1.36m 2019-05

- В драйвер ST7565R добавлен дополнительный набор символов 6x8.
- Дополнительный пункт меню для выключения тестера (SW\_POWER\_OFF).
- Добавлен контроль батареи в TestKey() и Zener\_Tool().
- В TestKey() добавлено обнаружение двух коротких нажатий тестовой клавиши, и удалены дублирующие функции в нескольких функциях для уменьшения размера прошивки. размер прошивки.
- Драйвер для дисплеев с ST7036 (4-битный параллельный & 4-проводной SPI, не тестировался).
- Пользовательские функции питания и контроля батареи для лучшей интеграции с другими устройствами. интеграция с другими функциями.
- Драйвер для дисплеев с PCF8814 (3-линейный SPI; спасибо Mahmoud Laouar для тестирования).
- Драйвер для дисплеев с STE2007/HX1230 (3-линейный SPI).
- Исправлена ошибка в функции LCD\_Clear в драйвере PCD8544.
- Отсутствующий кириллический шрифт добавлен в драйвер ST7565R (по сообщению Andrey@EEVblog).
- Обновлено шрифт\_8x16\_cyrillic\_vfp.h (спасибо Andrey@EEVblog).
- Проблема с неправильным символом в шрифте\_HD44780\_cyr.h решена.

## 9.10. v1.35m 2019-02

- Для смещения емкости теперь можно использовать смещение для каждого тестового вывода вместо прежнего среднего значения для всех.  
Test-Pins вы теперь можете использовать специфические смещения тестовых выводов (CAP\_MULTIOFFSET\_).
- Определение выводов для ST7920 в 4-битном параллельном режиме исправлено в config\_644.h (по сообщению jakeisprobably@EEVblog).
- Добавлена поддержка 3-линейного SPI в драйвере SSD1306.
- Драйвер SPI теперь может посылать 9-битные слова (только bitbang).
- Проблема с увеличением отклонения для резисторов от 7к5 до 19к5 Ом в CheckResistor() решена (сообщено Виталием).
- Добавлен альтернативный цикл задержки в IR\_Send\_Pulse(), который может быть установлен через SW\_IR\_TX\_ALTDELAY (спасибо Виталию).
- Переключатель конфигурации для дополнительных ИК-протоколов SW\_IR\_EXTRA был заменен на SW\_IR\_RX\_EXTRA для приемника/декодера и SW\_IR\_TX\_EXTRA для IR передатчик.
- Проблема с отсутствием новой строки для команд дистанционного управления в Display\_NextLine() фиксированный.
- Изменен вывод для SIRC на IR\_Decode(), чтобы быть ближе к протоколу. (предложение Виталия).
- Исправлена ошибка в IR\_Send\_Code() для SIRC-20 (сообщено Виталием).
- Обновление var\_russian.h (спасибо indman@EEVblog).
- Автоматическое отключение для режима автоматического удержания (POWER\_OFF\_TIMEOUT).
- Конфигурация выводов для тестовой кнопки и управления питанием разделена. (CONTROL\_PORT -> POWER\_PORT и BUTTON\_PORT).

- Несколько небольших улучшений.

### **9.11. v1.34m 2018-10**

- Испытание конденсаторов на ток утечки.
- Значение по умолчанию для RH\_OFFSET изменено на 350 Ом.
- Исправлена проблема с отсутствием пункта меню для фиксированного модуля ИК-приемника.
- Польский текст (спасибо Szpila).
- Драйвер дисплея для вывода на терминал VT100.
- Поддержка датчика температуры DS18B20.
- Драйвер для шины OneWire.

### **9.12. v1.33m 2018-05**

- Исправлена ориентация символа TRIAC в символах \_32x32\_hf.h.
- Команды дистанционного управления для автоматизации (через последовательный порт TTL последовательный интерфейс).
- Смещение X & Y для драйвера ST7735 теперь может быть изменено.
- Вызов меню путем замыкания тестовых контактов теперь возможен (UI\_SHORT\_CIRCUIT\_MENU).
- Проблема с разгрузочным реле в сочетании с поворотным датчиком устранена.
- Добавлен переключатель конфигурации для отключения спящих режимов MCU.
- Прием данных для последовательного TTL интерфейса (битовый bang & аппаратный USART).
- Ошибка в серийном выводе текста исправлена, и серийный вывод для результатов Opto-coupler test.
- Датский текст (с сайта glennndk@mikrocontroller.net).
- Настройки поправочных коэффициентов для конденсаторов.

### **9.13. v1.32m 2018-02**

- Вывод найденных компонентов дополнительно через последовательный интерфейс.
  - Драйвер для последовательного TTL интерфейса (аппаратный & bit-bang).
  - Обновление var\_russian.h (спасибо indman@EEVblog).
  - Поддержка смещений X&Y в драйвере ST7735.
  - Изменены настройки мониторинга батареи.
- Переключатель для отключения контроля батареи и для добавления неконтролируемого внешнего источника питания.
- Подключение питания не контролируется.
- Добавлен конфигурационный переключатель для выбора альтернативного режима работы при запуске стартап (UI\_AUTOHOLD).
  - Фильтр для германиевых транзисторов с высоким током утечки в функции обнаружения.
- Усовершенствованы выводные транзисторы с обедненным затвором.
- В драйвер для PCD8544 добавлена графическая распиновка.
  - Исправлена ошибка в функции LCD\_CharPos() для повернутого вывода в драйвере PCD8544.
  - Функции для графической распиновки улучшены и частично перенесены в display.c.
- При необходимости отделите выход от распиновки.
- Индикатор при использовании внешнего опорного напряжения (отображаемые значения).
  - Улучшен ИК декодер и добавлены дополнительные протоколы.
  - Дополнительные протоколы для ИК-пульта дистанционного управления.

### **9.14. v1.31m 2017-12**

- ИК-пульт дистанционного управления (передатчик).
- Поддержка вывода фиксированного сигнала через OC1B, когда OC1B не используется для тестового резистора от тестового контакта #2.
- Изменены настройки мониторинга батареи для поддержки других вариантов питания для питания.
- Драйвер для OLED-модулей на базе SSD1306.
- Поддержка цвета при выборе пунктов меню или параметров.

- Драйвер для ЖК-модулей на базе ILI9163.
- Исправлена проблема в генераторе квадратных волн.
- LCD драйвер для PCD8544, расширенный за счет повернутого на 180° выхода.
- Исправлена ошибка редактирования в Servo\_Check().

### 9.15. v1.30m 2017-10

- Опция запятой вместо точки для десятичных знаков.
- Поддержка расширенного счетчика частоты с входным буфером, НЧ и ВЧ кристаллический осциллятор.
- Незначительные улучшения простого счетчика частоты.
- Проблема с временем затворения в частотном счетчике для частот ниже 10 кГц при тактовой частоте MCU. 20 МГц фиксированный.
- Измерение ESR модифицировано для более коротких импульсов заряда, т.е. теперь ESR может быть измерено для конденсаторов  $f$  от 10nF. Если вы предпочитаете старый метод измерения вы можете активировать его альтернативно.
- Исправлена ошибка в обнаружении короткого замыкания на тестовых контактах.
- Драйвер ЖК-дисплея для ST7920 расширен за счет повернутого на 180° выхода.

### 9.16. v1.29m 2017-07

- Поддержка сенсорных экранов и драйверы для совместимости с ADS7843 Контроллер.
- Исправлена ошибка в настройке контрастности для PCD8544.
- Исправлена глупая ошибка в CheckSum().
- Драйвер для ЖК-модулей на базе ST7920 с разрешением 64x128 пикселей.
- SmallResistor() оптимизирован и улучшена логика обнаружения в CheckResistor(), чтобы позволить очень маленькие резисторы лучше обнаруживаются в сочетании с контактными резисторами тестового провода.
- Изменена логика управления и порог для транзисторов Дарлингстона в Get\_hFE\_C(),
- Исправление проблемы с некоторыми типами NPN.
- Центральный драйвер SPI. Драйвер и конфигурация ЖК-модулей соответственно.
- Итальянский текст Джино\_09@EEVblog.
- Поддержка HD44780 с набором кириллических символов путем hapless@EEVblog.

### 9.17. v1.28m 2017-04

- Больше/меньше клавиш как альтернатива поворотному энкодеру (HW\_INCDEC\_KEYS).
- Добавлен сброс к частоте по умолчанию в генераторе прямоугольников.
- Дальнейшие улучшения в определении скорости вращения поворотного энкодера. кодер (ENCODER\_STEPS). Изменения в функциях, использующих вращение скорость.
- Добавлен сброс к значениям по умолчанию в альтернативном генераторе ШИМ.
- русский текст от indman@EEVblog (только набор символов 8x16 по горизонтали выровненный)
- Добавлена поддержка фиксированного пленочного конденсатора для самонастройки смещения напряжения.
- Потенциальная ошибка в обработке смещения V\_ref в SmallCap() фиксированный.
- Возможность конфигурирования ЖК-модулей с ST7735 для запуска с очищенным дисплеем (без случайных точек).

### 9.18. v1.27m 2017-02

- GetLeakageCurrent() расширен измерением большого тока для CLD. Благодаря texasrugo@EEVblog для некоторых тестовых диодов.
- Исправлена ошибка в функции MilliSleep().
- Устранена проблема с большой индуктивностью при обнаружении диодов.
- Компенсация для измерения индуктивности в диапазоне мН.
- Поддержка ЖК-адаптеров на базе PCF8574 в драйвере для HD44780.
- Драйвер для битового и аппаратного I2C.

- Исправлена ошибка в обработке переменной распиновки для HD44780. Исправлены модули ЖК-дисплея.
- Цветная распиновка для нескольких функций меню.
- Функция тестирования для модельных сервоприводов.
- Альтернативный ШИМ-генератор с переменной частотой и шириной импульса. Требуется Поворотный энкодер и увеличенный дисплей.
- выход R\_DS для МОП-транзисторов и Vf собственного диода.
- Поддержка фиксированного модуля ИК-приемника в ИК-детекторе/декодере.
- Издание удалено в названии, так как Классическое издание теперь устарело.

### 9.19. v1.26m 2016-12

- Встроенная компенсация для измерения индуктивности (требует доработки).
- Адаптация FrequencyCounter() для поддержки ATmega 324/644/1284.
- Исправлена проблема в логике измерения индуктивности. Примечание от indman@EEVblog.
- Ошибка в работе с опорными напряжениями для ATmega 324/644/1284 решен.
- Определение скорости вращения поворотных энкодеров улучшено, чтобы позволить для лучшей обработки различных значений для импульса/шага и импульса/блокировки. лучше.
- Добавлен аппаратный SPI во все драйверы для ЖК-модулей на базе SPI.

### 9.20. v1.25m 2016-09

- Много изменений для поддержки ATmega 324/644/1284.
- Управление тестовыми резисторами изменено на переменные контакты порта.
- Программная опция для цветового кодирования тестовых контактов.
- Централизованное управление цветом.
- Файл с перечнем настроек для различных версий тестера или клонов.
- Исправлена небольшая проблема с символами 24x24 VP в config.h.  
Примечание от lordstein@EEVblog и hapless@EEVblog.

### 9.21. v1.24m 2016-08

- Измерение тока утечки саморазряда конденсаторов более 4,7 мкФ.
- Тип обнаружения биполярных транзисторов с диодом на одной подложке.
- Измерение тока утечки для токов до нА расширен диапазон. Для диодов и биполярных транзисторов, токи утечки превышают 50 нА.
- Отображение диодов свободного хода для транзисторов теперь проверяется на правильность диод (контакты и полярность).
- Исправлена ошибка в отображении диодов свободного хода для биполярных транзисторов.
- Добавлена функция поиска конкретного диода и соответствующим образом настроены некоторые другие функции. функции отрегулированы соответствующим образом.
- Улучшенное обнаружение диодов для включения германиевых диодов с очень низким Vf при низких токах. низкий Vf при малых токах.
- Проблема с LCD\_ClearLine(0) для ILI9341 и ST7735 решена.
- Улучшенное обнаружение обедненных FET. Германиевые транзисторы с высокой Ток утечки отфильтровывается. FET с низким I\_DSS теперь также являются I\_DSS обнаружены. Измерение I\_DSS.

### 9.22. v1.23m 2016-07

- Поддержка совместимых ЖК-модулей PCD8544 и ST7735.  
Благодаря hansibull@EEVblog для дисплея PCD8544.
- добавлена функция ожидания для 20 МГц тактовой частоты MCU.
- MeasureESR() теперь поддерживает тактовые частоты АЦП, отличные от 125 кГц.
- Обнаружение программируемых однопереходных транзисторов (PUT) и Добавлено UJT's (Unijunction Transistor).  
Спасибо edavid@EEVblog за прислать мне несколько UJT для тестирования.
- Незначительные оптимизации для ILI9341 и ST7565R.
- Снова исправлена проблема с символами размером более 8x8 для ST7565R.



- Контакт порта /RES для ILI9341 был проигнорирован. Ошибка исправлена и также исправлена некорректная задержка при аппаратном сбросе.
  - Поддержка отдельных линий передачи данных для устройств на базе HD44780 ЖК-модули.
  - Определяемый пользователем делитель напряжения для напряжения батареи.
  - Выход If для добавленных оптопар.
  - Изменены тестовые контакты из инструмента ESR на 1-3 для совместимости с к-прошивкой. с прошивкой k.
  - Глобальные настройки, специфичные для MCU, перенесены в отдельные заголовочные файлы. заголовочные файлы.
- Несколько незначительных изменений для поддержки ATmega664/1284 также.
- Обновление чешских текстов. Спасибо Капе.

### 9.23. v1.22m 2016-03

- Тест для оптопары с выходом V\_f светодиода, CTR и t\_on или t\_off. раз (типы с транзисторным выходом). Спасибо all\_repair@EEVblog за оптопары для тестирования.

### 9.24. v1.21m 2016-01

- Лицензировано под EUPL V.1.1
  - Загрузка и сохранение значений настройки оптимизированы и поддерживаются для Добавлены два профиля настройки.
  - Добавлен RC-6 к ИК детектору. Ключевая проблема преждевременного удаления IR модуль приемника снят.
- Конфигурационный переключатель для отключения резистора Vs для ограничения тока. для модулей 5В ИК-приемников.

### 9.25. v1.20m 2015-12

- Функция реализована для обнаружения и декодирования ИК-пультов дистанционного управления.
- Требуется модуль ИК-приемника TSOP.
- MainMenu() изменен для уменьшения использования оперативной памяти.

### 9.26. v1.19m 2015-11

- Графическая распиновка для 3-контактных полупроводниковых приборов. Показывает символ плюс тестовых контактов.
  - Встроенная поддержка цвета.
  - Прямой вывод количества диодов в ShowDiode(), если их больше 3. были найдены (больше не через Show\_Fail()). Заметка от незадачливого @EEVblog
  - LCD\_ClearLine() была расширена во всех драйверах ЖК-модулей, так что теперь только остаток строки для ускорения очистки, особенно для графических ЖК-дисплеев. для графических ЖК-дисплеев.
- Идея заключается в том, чтобы сначала вывести текст, а затем очистить его. вместо того, чтобы сначала стереть всю строку, а затем вывести текст. текст.
- Драйвер, написанный для LCD модулей на базе ILI9341/ILI9342.
- Спасибо На форуме Overtuner@EEVblog два LCD модуля для тестирования.
- Решена проблема с  $\mu$ /микро символами в файлах charset.
  - Исправлена ошибка с символами больше 8x8 в LCD\_Char() для ST7565R.
  - Обновлено тексты на чешском языке (спасибо Кара).
  - Небольшая ошибка в MenuTool() при переходе от последнего к первому пункту фиксированный.

### 9.27. v1.18m 2015-07

- MenuTool() улучшена таким образом, что обновляется только измененный список.
- В противном случае обновляется только индикатор выбора.
- Исправлена ошибка управления переменными в config.h.
- Возможность сброса настроек прошивки по умолчанию при включении.
- Функции сохранения/чтения значений настройки оптимизированы.

- Драйвер для графических модулей ST7565R.
- Простая среда, предназначенная для добавления новых LCD контроллеров. Общие функции отображения перенесены в display.c. Каждый контроллер получает свой собственный исходный и заголовочный файл. Старый драйвер для HD44780 была адаптирована к новым условиям.
- Пользовательский интерфейс перестроен для гибкой работы с многострочными ЖК-модулями. перестроен.
- Удалены исходные зависимости для ATmega168 (слишком маленький ;)).
- Оптимизирована логика работы в MenuTool().
- Начата новая редакция прошивки, которая также поддерживает графические LCD модули.
- Эта версия называется 'Trendy Edition'. Старая версия прошивки теперь называется "Classic Edition".

### 9.28. v1.17m 2015-02

- Улучшение функции CheckDiode(). Измеренное Vcc учитывается в сопротивлении проверить.
- Также проблема обнаружения резисторов около 2 кΩ с дополнительным DC-DC преобразователем м (HW\_ZENER) решено.
- Исправлены ошибочные комментарии.
- Очистка целочисленных типов данных.

### 9.29. v1.16m 2014-09

- Тест для поворотного энкодера.
- Улучшено несколько мелочей в MeasureInductance() для повышения точности. точность.
- Добавлена функция ShowAdjust() для отображения абсолютных значений Vcc и внутреннего параметра опорное напряжение (предложение Властимила Валуша).
- Несколько небольших улучшений.

### 9.30. v1.15m 2014-09

- Расширение функции TestKey() путем определения динамической скорости вращения диска из опциональный поворотный энкодер.
- Генератор сигналов квадратной волны с переменной частотой.
- Изменено MeasureInductance() для возврата времени в нс и изменено вычисление в MeasureInductance() адаптирован (спасибо Властимилу Валушу).

### 9.31. v1.14m 2014-08

- Адаптированный пользовательский интерфейс для поворотных датчиков.
- Исправлено предупреждение компилятора о R\_Pin2 в ShowDiode() (благодаря Милан Петко).
- Резисторы от 1,5 кΩ до 3 кΩ были признаны двойными диодами. несаце).
- ShortCircuit() изменена таким образом, чтобы можно было избежать создания преднамеренного короткое замыкание в случае проблемы.
- Добавлен счетчик частоты (аппаратная опция).

### 9.32. v1.13m 2014-07

- Чешские тексты (спасибо Кара).
- Прямое измерение ESR и вывод ШИМ-генератора с использованием тестовых выводов.
- Обработка инструкций прекомпилятора для оптимизированных опций.
- Поддержка поворотных датчиков для работы (аппаратная опция).

### 9.33. v1.12m 2014-03

- Проблема умлаута решена для немецких текстов (спасибо Андреасу Хёбелю).
- измерение ESR для конденсаторов >0,18 мкФ.
- Вывод ЖК-модуля оптимизирован для экономии нескольких байт флэш-памяти.

### 9.34. v1.11m 2014-03

- Улучшено распознавание выводов симисторов (G и MT1). Выходные данные показывают

MT1 и MT2.

- Выделенная функция вывода для распиновки полупроводников. Выход на формат 123= для лучшей читабельности.
- Оптимизировано несколько выходных функций.
- Улучшенное тестирование биполярных транзисторов, включающее транзисторы с защитным диодом на том же подложка (создает паразитный второй транзистор). транзистор).
- Вывод транзистора отмечает этот особый случай знаком „+,, символ „+,, после спецификации типа.
- Диодный выход расширен индикацией возможного биполярного транзистора с защитным диодом и резистором база-эмиттер. и резистор база-эмиттер. Это признано двойным диодом как двойной диод.
- Вывод резистора база-эмиттер сигнализирует об этом особом случае. особый случай.
- Выход биполярных транзисторов с индикацией база-эмиттер-резистор добавлено.
- Когда найден резистор база-эмиттер, выходной сигнал hFE и V\_BE пропускаются, так как оба значения не могут быть правильными,
- Интегрированное обнаружение диодов в обедненных FET в диодных испытаниях улучшено.
- Проблема обнаружения стока и истока в FET с истощением устранена.
- Симметричное обнаружение стока и истока для обедненных FET.
- Vth теперь отрицательный для P-канальных FET.
- Измерение V\_GT для тиристоры и симисторы.
- В связи с увеличением размера прошивки, генератор ШИМ теперь доступен только для модели ATmega328.

### 9.35. v1.10m 2013-10

- Встроенная поддержка внешнего опорного напряжения 2,5 В (аппаратная опция).
- Встроенная поддержка реле защиты (разряд конденсатора) ( аппаратный вариант).
- Текст прощания при выключении питания заменен на текст приветствия при включении. изменился,
- для облегчения обнаружения низкого напряжения питания и для уменьшения падения напряжения, вызванного DC-DC преобразователем при включении. преобразователя при включении питания.
- Реализован тест диодов Зенера (аппаратная опция).
- В главном меню есть опция выхода из меню по ошибке при вызове. случайно.
- Поддержка частоты 16 МГц MCU.

### 9.36. v1.09m 2013-07

- Обнаружение встроенных IGBT.
- Дополнительная проверка встроенных МОП-транзисторов.
- Измерение hFE для биполярных транзисторов учитывает ток утечки при измерении.
- в цепи эмиттера.
- Для МОП-транзисторов указано направление встроенного диода.
- Проблема решена с переключенными выводами стока и истока на улучшенных МОП-транзисторах.
- Решение проблем некоторых IDE с Makefile. Важные ценности или параметры также могут быть установлены в config.h.

### 9.37. v1.08m 2013-07

- Так как функция SmallResistor() не может обеспечить правильное значение сопротивления постоянному току для определенных индуктивностей, то значение сопротивления для определенных индуктивностей, функция CheckResistor() была расширена функцией обнаружения проблемных случаев, чтобы сохранить стандартные значения измерений.
- Добавлено измерение индуктивности (только для ATmega328/P)
- Незначительные улучшения в отображении диодов и биполярных транзисторов.
- Реализовано измерение тока утечки.

- Решена проблема с германиевыми транзисторами с высоким током утечки. Обнаружено как P-канальный JFET.

- Переименовал несколько функций и добавил или переформулировал комментарии.

### **9.38. v1.07m 2013-06**

- Оптимизирован выход диода и добавлена индикация Vf для малых токов.

- Улучшено обнаружение диодов. Конденсаторы и резисторы исключены гораздо лучше. Обнаружение конденсатора пропускается, когда диод обнаруженный диод, чтобы сократить процесс поиска.

- Исправлена ошибка переполнения массива в CheckResistor().

- Улучшена логика отображения курсора для указания наличия дополнительной информации или для указания поиска новой детали.

- Улучшена работа генератора ШИМ, чтобы избежать случайного завершения (теперь требуется два коротких нажатия клавиш).

- Функция общего меню реализована и все меню изменены на нее (изменилась планировка!).

- TestKey() теперь выводит красивый мигающий курсор.

### **9.39. v1.06m 2013-03**

- Несколько небольших улучшений и немного почистили.

- TestKey() расширен, чтобы пользователь мог быть проинформирован об ожидаемом вводе. о предполагаемых входных данных.

- Функция TestKey() улучшена в отношении коротких нажатий клавиш.

- ШИМ-генератор для генерации широтно-импульсных модулированных сигналов с различными

- Реализованы частоты и свободно выбираемый рабочий цикл.

- Реализация функции сна для снижения энергопотребления тестера.

Таким образом, среднее энергопотребление снижается примерно вдвое.

(подсветка исключена).

- Улучшение функции окончания заряда. Если разрядка не удастся, пораженные контакты выводятся информация о затронутых контактах и остаточном напряжении. Это должно помочь, для обнаружения низкого значения CAP\_DISCHARGED.

- Добавлена возможность устанавливать типы неисправностей.

### **9.40. v1.05m 2012-11**

- LargeCap\_table[] и SmallCap\_table[] перенесены из EEPROM во флэш-память, для уменьшения использования EEPROM. Прошивка с немецкими текстами использовано больше, чем 512 байт из ATmega168.

### **9.41. v1.04m 2012-11**

- Простая логика, встроенная в выход диода, устраняет необходимость измерения емкости для антипараллельных диодов. измерение емкости опущено.

### **9.42. v1.03m 2012-11**

- Решена проблема обнаружения силовых диодов. Диоды с высоким током утечки были обнаружены как резистор.

- Исправлены предупреждения компилятора о неинициализированных переменных.

Увеличение размера прошивки на 44 байта :-)

### **9.43. v1.02m 2012-11**

- Верхний предел для сопротивления испытательных проводов 1,00 Ω в Встроенная самонастройка.

- Функции самодиагностики и самобалансировки выполняют проверку на короткое замыкание и обеспечивают обратную связь. и предоставить обратную связь.

- Главное меню предоставляет обратную связь об успехе/неудаче выбранного действия.

### **9.44. v1.01m 2012-10**

- Контрольная сумма для сохраненных значений настройки плюс проверка.

- Функция измерения малых сопротивлений (разрешение: 0,01  $\Omega$ ).
- Самонастройка, расширенная за счет смещения нуля для сопротивления измерительных кабелей.
- CheckResistor() выполняет дополнительное измерение для маленьких резисторов (<10  $\Omega$ ). выполнен.
- Добавлена функция сравнения масштабированных значений.
- Несколько функций, приспособленных для переменного масштабирования значений.

#### **9.45. v1.00m 2012-09**

- Простое меню для выбора самодиагностики,
- Самопроверка,
- сохранение калибровочных значений в EEPROM.
- Отображение значений настройки.
- hFE изменен с 16 на 32 бита (больше нет ограничения в 65 к $\Omega$ ).

#### **9.46. v0.99m 2012-09**

- Первая опубликованная версия, основанная Карлом-Хайнцем.

---

## *Литература*

---

- [1] <http://www.mikrocontroller.net/articles/AVR-Transistortester>  
*Онлайн документация по Транзистортестеру,*
- [2] Markus Frejek <http://www.mikrocontroller.net/topic/131804>  
*Нить Маркуса Фрежека, Форум, 2009 г..*
- [3] [http://www.mikrocontroller.net/articles/AVR\\_Transistortester](http://www.mikrocontroller.net/articles/AVR_Transistortester)  
*Онлайн документация по Транзистортестеру*
- [4] <http://www.mikrocontroller.net/topic/248078>  
*Нить от Karl-Heinz K., 2012*
- [5] <https://github.com/madires/Transistortester-Warehouse>  
*Полная коллекция программного обеспечения*
- [6] <https://github.com/kubi48/TransistorTester-source/tree/master/Markus>  
*Полная коллекция программного обеспечения*
- [7] <https://github.com/Mikrocontroller-net/transistortester/tree/master/Software/Markus>  
*Полная коллекция программного обеспечения*
- [8] [madires@theca-tabellaria.de](mailto:madires@theca-tabellaria.de)  
*Авторская почта*
- [9] <http://www.mikrocontroller.net/articles/AVRDUDE>  
*Онлайн-документация интерфейса программиста avrdude,*
- [10] <https://www.mikrocontroller.net/topic/248078>  
*Основной язык - немецкий, но английский тоже подходит.*
- [11] [https://www.eevblog.com/forum/testgear/\(\\$20-lcr-esr-transistor-checker-project/](https://www.eevblog.com/forum/testgear/($20-lcr-esr-transistor-checker-project/)  
*Только английский.*
- [12] <https://vrtp.ru/index.php?showtopic=16451> - Российский форум  
*Информация о различных клонах, например, изображения, схемы и прошивки*
- [13] <https://disk.yandex.ru/d/yW8xa5NjgUo5z> - (by indman@EEVblog)  
*Информация о различных клонах, например, изображения, схемы и прошивки*
- [14] [https://drive.google.com/file/d/1-IJA8uTcsCA\\_6SYHEuMydjfS2vNgmwdH/edit](https://drive.google.com/file/d/1-IJA8uTcsCA_6SYHEuMydjfS2vNgmwdH/edit)  
*Инструкция и файлы для WinAVR (by indman@EEVblog)*
- [15] <https://github.com/kubi48/TransistorTester-source>  
*Онлайн Транзистортестеру*
- [16] <https://github.com/kubi48/TransistorTester-documentation>  
*Онлайн документация по Транзистортестеру*