ruthsarian / **stc_diywatch**

Firmware for STC15F204EA based DIY digital watch kit.

View license

2 stars 0 forks

Star Watch

Code Issues Pull requests Actions Projects Wiki Security

master

ruthsarian Added links to power branch. ... on Jun 25, 2017 31

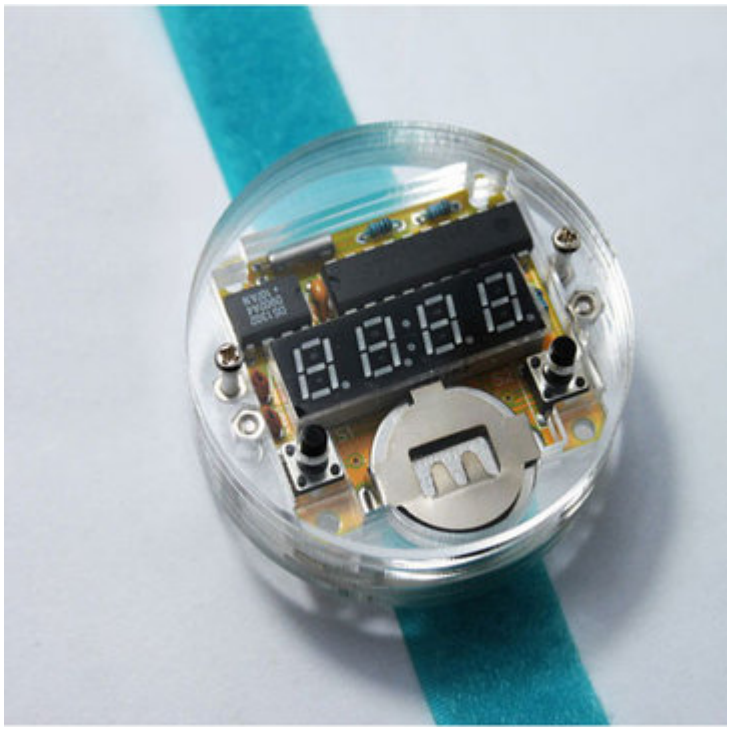
View code

README.md

STC DIY Watch Kit Firmware

NOTE: This project is in development and not fully tested. Expect bugs.

This is a firmware replacement for the STC15L204EA based DIY LED watch kit (available through Banggood and eBay). It uses sdcc to build and stcgal to flash the firmware.



The STC15L204EA is the 3 volt version of the STC15F204EA which runs on 5 volts.

Features

- Display for 5 seconds, then go into power down mode until the left button is pressed. This helps preserve battery power.
- Display time, month/day, year, and day of week.
- Set time, day, month, and year. Day of week is calculated automatically.
- Option to display time in 12 or 24 hour format.
- Day of week as letter abbreviation.
- Secret scrolling message.

Features To Add

- Improve power consumption.
- Create low battery alert of some kind.
- Change display brightness. (low priority; maybe store value in clock ram?)

How to Use the Watch

- A short press of left button cycles through the display modes (time, day/month, year, day of week)

- A long press of the left button will enter the change value mode and is indicated by blinking numbers.
- The right button is used to increment the value that is blinking. A short press increments by one. Press and hold the button to increment quickly.
- While displaying the current time, hold both buttons down to display the secret message.

Power Consumption

This watch operates off a 3 volt CR2032 coin cell battery. Due to this, and the fact that the case is a pain to remove to install a new battery, power consumption is a concern. The stock firmware appears to draw about 5 milliamps (mA) when the display is on and about 350 microamps (uA) when the display is off. This firmware currently draws about 8mA when the display is on and about 300uA when the display is off. Assuming a fresh CR2032 has about 200 milliamp hours (mAh), the battery, if the watch is left off, will last about a month. Display consumption can be reduced further by slowing the clock of the microcontroller (via the CLK_DIV register). This can yield about 2mA savings with the display on, but further work is needed.

Power Modification, Part 1 - Remove the 3 10k Resistors

To maximize battery life, power consumption with the display off needs to be reduced as much as possible. This watch kit includes 3 10k pull-up resistors connected to the DS1302. These three pins have 40k pull-down resistors to ground. This is where the majority of the 300uA is being consumed with the display off. To remove this constant power drain, remove (or don't install) the three 10k resistors. To compensate, the 3 DS1302 pins need to be changed to push-pull output in the firmware. The end result is power consumption with the display off is reduced to around 50uA. This should result in a battery life closer to 5 months! The code for this modification will be added to the power branch of this repository.

Power Modification, Part 2 - Cut Traces and Add a Diode

The DS1302 has a low-power mode that it will enter when power to VCC2 (pin 1) is removed. In this mode the DS1302 draws nanoamps. Unfortunately this kit has both VCC2 and VCC1 (pin 8) tied together, leaving the DS1302 into normal operating mode even while the display is off, drawing up to several hundred microamps (but in practice, measured to be around 50uA).

If VCC1 and VCC2 were separated, VCC1 powered from the battery, and VCC2 controled by the microcontroller, there's an opportunity to cut power to VCC2 when the display turns off, reducing power consumption to about 400 nanoAmps.

To achieve this, two traces must be cut. First is to cut the trace that comes from the capacitor next to the DS1302 and pin 1. This trace carries power from the battery and cutting it removes power to both VCC1 and VCC2. The next trace to cut is the one between pin 1 and pin 8 of the DS1302. Both traces are on the underside of the PCB. With the traces cut, a jumper wire should be added between pin 1 of the DS1302 (VCC2) and port 3.0 (pin 11) of the microcontroller.

To bring power back to VCC1, a diode should be used to connect the battery to pin 8 of the DS1302. The diode is used instead of jumper wire because VCC2 nees to be at a higher voltage than VCC1 for the DS1302 to switch to VCC2 power and return to normal operation. The diode will provide a 0.6v drop, putting VCC2 at a slightly higher voltage than VCC1 when VCC2 is powered.

The firmware then needs to be modified to put port 3.0 into push-pull output mode and then set it to logic 1. The firmware then needs to change it to a logic 0 just before going into power down mode. The code for this modification will be added to the power branch of this repository.

NOTE: The power modification is a work in progress and unproven.

Hardware

- DIY LED Digital Watch Kit, based on STC15L204EA and DS1302, e.g. Banggood SKU 206204
- Connected to PC via cheap USB-UART adapter, e.g. CP2102, CH340G. Banggood: CP2102 USB-UART adapter

Connection

P1 header	UART adapter
P3.1	RXD
P3.0	TXD
GND	GND
5V	5V

Requirements

- Windows, Linux, or Mac (untested on Linux or Mac; please comment with results on Linux or Mac)
- sdcc installed and in the path (recommend sdcc >= 3.5.0)
- sdcc (or optionally stc-isp). Note you can either do "git clone --recursive ..." when you check this repo out, or do "git submodule update --init --recursive" in order to fetch stcgal.

Usage

```
make clean
make
make flash
```

Options

- Override default serial port: `STCGALPORT=/dev/ttyUSB0 make flash`
- Add other options: `STCGALOPTS="-l 9600 -b 9600" make flash`

Use STC-ISP flash tool

Instead of stcgal, you could alternatively use the official stc-isp tool, e.g stc-isp-15xx-v6.85I.exe, to flash. A windows app, but also works fine for me under mac and linux with wine.

NOTE: Due to optimizations that make use of "eeprom" section for holding lookup tables, if you are using 4k flash model mcu AND if using stc-isp tool, you must flash main.hex (as code file) and eeprom.hex (as eeprom file). (Ignore stc-isp warning about exceeding space when loading code file.) To generate eeprom.hex, run:

```
make eeprom
```

Clock assumptions

Some of the code assumes 11.0592 MHz internal RC system clock (set by stc-isp or stcgal). For example, delay routines would need to be adjusted if this is different.

Differences from Original Work

This project is based on the stc_diyclock project by Jens Jensen. Without his original work this project would not exist. The modifications from Jens' original work include

- Some of the content of this README was altered to reflect the differences with this project.
- Code for features not supported by this kit (temperature and light sensors) was removed.
- LED display routines were modified to reflect this kit's hardware.
- Other code changes to reflect the different PIN configuration of this kit.
- Code formatting changes to fit my personal tastes.
- New code to support putting the MCU into power down mode to save battery life; similar to how the original firmware operates.

Disclaimers

This code is provided as-is, with NO guarantees or liabilities. As the original firmware loaded on an STC MCU cannot be downloaded or backed up, it cannot be restored. If you are not comfortable with experimenting, I suggest obtaining another blank STC MCU and using this to test, so that you can move back to original firmware, if desired.

References

- Jen Jensen's stc_diyclock project
- STC15L204EA datasheet
- Maxim DS1302 datasheet
- sdcc user guide
- stcgal

Releases

No releases published

Packages

No packages published

Languages

- **C** 97.6%
- **Makefile** 2.4%