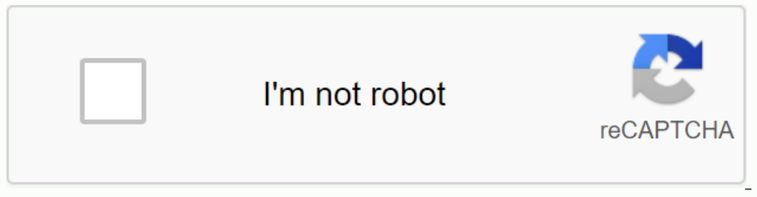

Continue

W1209 temperature controller pdf download

Community Library Challenges Groups Questions Tutorials Engineers Workbench Overview Features Comparison Store Overview Features Resource Comparison Blog Help Center © 2020 GrabCAD, STRATASYS Computer Assisted Design (CAD) Solution Files and all associated content posted on this site are created, uploaded, managed and owned by external users. Any CAD and any associated text, image or data is not funded or identified with any real-world company, organization or item, product or product, or goodness they may claim to display. Displays the current temperature: The thermostat displays the current temperature in oC by default. When in any other situation input for approximately 5 seconds will cause the thermostat to return to this default view. Trigger Temperature Setting: To set the trigger temperature, click the button marked 'SET'. The display of the seven sections will be abomination. You can now set trigger temperature (in oC) by using the '+' and '-' buttons at fixed levels of 0.1 degrees. If buttons are not pressed for approximately 2 seconds, the trigger temperature is stored and the display returns to the current temperature. Setting the parameters: To define a parameter first, press and hold the 'SET' button for at least 5 seconds. The view of the seven sections should now display 'P0'. This value represents the parameter P0. Clicking the '+' or '-' buttons will be repeated over a bike despite the different parameters (P0 through P6). Clicking the 'SET' button with each of the name parameters displayed will allow you to change the value for this parameter by using the '+' and '-' buttons (see below). When you are finished setting a parameter, click the Set button to exit this option. If buttons are not pressed for about 5 seconds, the thermostat exits the parameter options and returns to the default temperature view. Setting the cooling or heating parameter P0: The P0 parameter includes two settings, C and H. When set to C (default) the relay will energize when the temperature reaches. Use this setting if you are connecting to an air conditioning system. When set to H the relay will de-eneise when the temperature reaches. Use this setting if you control a heating device. Set the P1 hysterics parameter: This sets the temperature change number before the relay changes mode. For example, if the default 2oC and trigger temperature is set to 25oC, it does not tagging until the temperature falls back below 23oC. This histrisis setting helps stop the thermostat from continually triggering when the temperature drifts around the travel temperature. This project uses STM8 eForth to transform the W1209 shelf board into a data logging thermostat with a terminal interface and an embedded programming environment. It provides source code, ready-to-use firmware, and documentation. Features are: Heating thermostat, e.g. for chicken egg incubator no special tool installation necessary binary ready, and source New binary supplies can be built with the help of Travis-CI (or locally on your Linux computer) interactive programming in Forth, even when the thermostat task is running! Any serial terminal program, e.g. pycocom, hamdcom, or e4thcom data login with entry ring buffer 144, can be used. Interval of 0.1 hours to 10 hours, and access log through the L.dump column console The command prints the log through the tori console - the last line is the last L.wipe command deletes the lowest log memory records and the highest temperature records the number of relay cycles and the heating duty cycle detection of a disconnected sensor is easy to use in the Parameters menu for a set point , hysterics and travel delay this is a work in progress even though it has been completed in the feature. Please consider the beta software (please write a problem if you need support or want to give feedback :-)). Possible future features: A simple field bus for building a network of thermostat units more fail-safe features (e.g. parameter integrity, monitoring limitations) Please note that implementing new features may require using a more compact STM8 eForth, or removing existing features. The build modular eForth STM8 feature makes it easy. With minor modifications, you can use code for other generic thermostat boards that support STM8s eForth. You can add other panels using a modular structure. The code can also be used for a no-interactive thermostat, for example using the C0135 relay board. Note getting started: STM8 eForth only works on supported STM8 chips - W1209 panels with Nuvoton chip will need to change before they can be used with the code in this repository! 512 576 bytes from the registry store relies on an undocumented feature of STM8S003F3P6 that may not work in some strokes. Although there are no known instances of where the extended EEPROM did not work, using the STM8S103F3P6 chip or reducing the registry buffer to 64 bytes will always work. After programming binary firmware to the W1209 board, it should work as a regular thermostat. Parameters can be defined by using the clipboard keys (defined, +, -). The following are recommended for programming: W1209 ST-Link Programmer TTL-Serial-Interface Please refer to the STM8 eForth Wiki for instructions on W1209 programming using an ST-Link-compliant programmer. After programming, the view should display the temperature value (at °C) or .dEF. (Default) if no sensor is connected.) Before using the thermostat, please reset the parameter values by holding the + and keys up to the text rES. appears in the LED display (about 4s). Pressing the Offset key leads to the parameter menu. The menu returns to the temperature view when you have not pressed a key for more than 10s. 10.0 - 80.0 37.5 °C Heating Thermostat set point (turn off above) LoG. 0.0 - 10.0 10.0 hours Luger interval during dEL hours. 0.0 - 60.0 0 0.0 s thermostat heating trip delay -2.0 - 2.0°C thermhom offset (for corrections around desired coordinates) hYS. 0.1 - 2.0 0.5°C thermostat hysteria (the difference between lower travel points and top travel points) note that in most cases hanging the dEL parameter delay travel. There's no need for that. Using the data log the Data Loger feature uses the top 576 years of internal EEPROM as a 144 ring-buffe feed. You can set the logger interval (time between examples) between 6 minutes (0.1h) and 10 hours by the LoG menu item. The following items are recorded: the lowest temperature at the highest temperature heating cycle must DC = 100% * t.on/(t.on + t.off) The number of relay cycles can be accessed in the data log through the Forth terminal with the L.dump command. The log can be deleted with the L.wipe command. To use the Porat console, connect a serial interface adapter to the + and pins. The following chart illustrates the impact of isolation improvements, changing the hysteric parameter, and the impact of overnight heating temperature delay in my living room: such a chart can be created with the following steps: set the log interval by the required observation time to 0.1h for example control optimization. 3.5h for the 3 weeks it takes to hatch a chicken egg to let the thermostat run (no intervention required) connect TTL-RS232 interface to keys + (RX) and (TX) (pins alongside LED display) to access console with serial console program with 9600-N-8-1 settings for Linix e.g. e4thcom, minicom, picoterm, or miniterm.py for Windos e miniterm.py, PuTTY, HyperTerminal Press enter key - STM8 eForth needs to respond with L.dump certificate type to extract the data (note the last line is now) Make a note of the read time, and registry interval) copy and paste the data into a layout sheet program to use the known read time, and the log interval for the time on the x-axis of the x/y chart controller Thermostat control.fs implements a very simple 2-point controller with offset, and delay. There is no other reason for any of the parameters except that they can be used for noise relief, which measuring the sensor has been handling. In future versions it may be replaced by a simple PI controller, where the duty cycle of the relay is the control variable. Working with the code in this repository replicating this buffer, running makes the solution dependent on a dependency solution. This will download binary STM8 eForth, and add required folders, files, and symbols. The general workflow for this setting is this: Clone the pool to install stm8flash to connect Donegal ST-LINK-V2 to the W1209 session to reverse the defaults to delete the stock firmware warning: there is no known public source for the stock firmware, and after deleting it there is no going back! Run do stm8ef binary flash for interactive scripts install e4thcom optionally install the development version of ucSim (or use docker image) to take advantage of image creation not templates for binary W1209 programming Follow the instructions on the STM8EF Wiki page for board W1209 (if stm8flash is installed just turn on to do flash). Interactive scripting using the tori console is supported by the binary eForth STM8 root. Please refer to the instructions for a serial console. The recommended console emulator is e4thcom: it supports line editing, uploading source files with #include, and using a directory with #require. Type #i.fs to load the full source code. For continuous integration use cases, simload starting with ucSim to create an STM8 binary file that contains the full thermostat script, including the W1209-FD master image. The Docker tg9541/docker-sdcc image contains tool dependencies for continuous integration (see .travis.yml or use the Travis-CI character to review the execution log). About stm8 eForth master system code is based on STM8EF binary edition. Makefile uses the Modular Build method to automatically build binary for the W1209-FD tablet support folder. For more information, see the STM8EF Wiki. This is a community progea - it is driven by user donations! Please write a problem if you have any questions and post a comment on HACKADAY. Project IO, or contribute new documents, code, use cases, and requirements. Also consider writing about it in forums, blogs, on Twitter, or making a YouTube video in your native language so that others can find it (please use #W1209 #STM8EF hashtags). Commercial use can be used in code in this store for commercial applications in accordance with the license terms. Page 2 You cannot perform this operation at this time. You signed in with a tab or other window. Reload to refresh the session. You exited on another tab or window. Reload to refresh the session. We use optional third-party analytics cookies to understand how you use GitHub.com so we can build better products. Learn more. We use optional third-party analytics cookies to understand how you use GitHub.com so we can build better products. You can always update your selection by clicking Cookie Preferences at the bottom of the page. For more information, see our Privacy Statement. We use essential cookies to perform essential website functions, for example, they are used to sign in to you. Learn more we always actively use analytics cookies to understand how you use our websites so we can make them better, for example, they're used to collect information about the pages you visit and how many clicks you need to perform a task. For more details