VSR Mini Mega Alternator Regulator

Communications and Programming Guide

V1.110621

Table of Contents

ABOUT THIS GUIDE

The guide is based on the guide for the VSR series of product offerings developed (or contemplated) by William A. (“Al”) Thomason. The suite of communications features has be reduced to align with the features of the VSR Mini Mega Alternator Regulator (VSMMAR) and does not apply to any other devices, current or contemplated.

This guide is documents communications into and out of the VSMMAR on the serial USB port and messages which may be sent out of one of the other two serial ports (Serial1 and Serial2) which may be configured to support the Remote TFT Display feature.

You will soon see that setting parameters using serial commands is very powerful, but quite tedious. Using the Dashboard tool (once it becomes available for the VSMMAR) gives you all this power without the tedium. Nonetheless, refer to this guide to understand the parameters in detail.

ASCII COMMUNICATIONS

The Mini Mega processor daughter card of the VSMMAR features a built in micro-USB port to allow for advanced monitoring, diagnostics, configuration, and firmware updates. Simply connect a USB cable (making sure it is a proper data USB cable, and not a charger-only cable) between your computer and the USB micro port and utilize a terminal in Visual Studio Code/PlatformIO, the monitor in the Arduino IDE, or a serial terminal app such as Putty (see below).

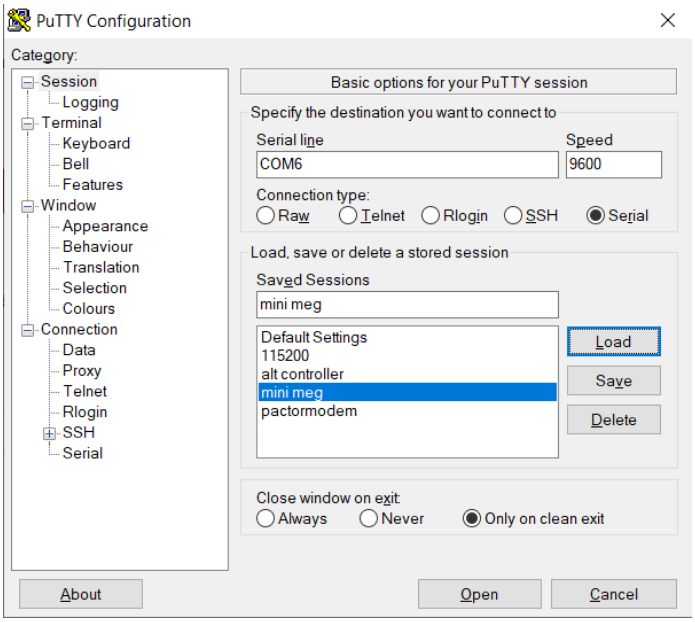
TERMINAL PROGRAMS

Many operating systems have built-in terminal programs, as does the Arduino IDE. After connecting the regulator to the computer you can use one of these to open a communications window to the VSMMAR.

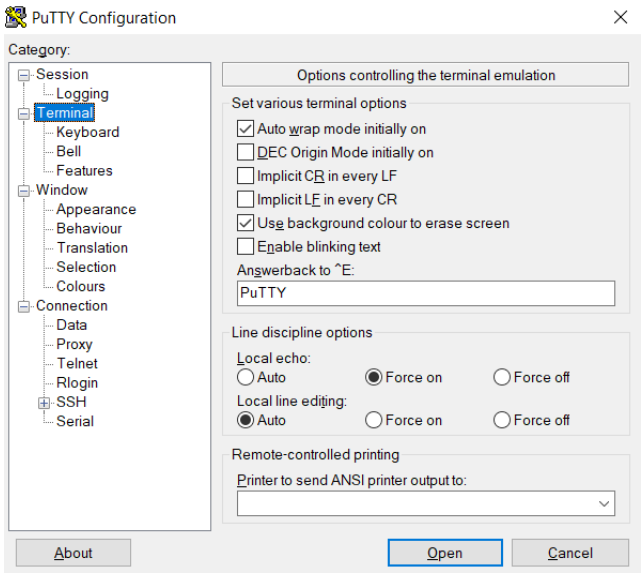
NOTE: Some terminal programs do not send a complete end-of-line terminator (CR+LF - specifically, Arduino's Serial Monitor). To support these environments, the VSMMAR will recognize the character '@' as an alternative EOL.

Remember that any configuration changes you make to the regulator via ASCII commands may not take effect until the regulator is rebooted. When you have finished making changes, make sure to issue the $RBT: command to not only assure changes are saved to the regulators' non-volatile memory, but that the changes are then utilized by the regulator. (Refer to $RBT: - ReBooT system on page 32) After rebooting the VSMMAR verify the changes you sent were indeed recognized by the regulator by inspecting the various status strings.

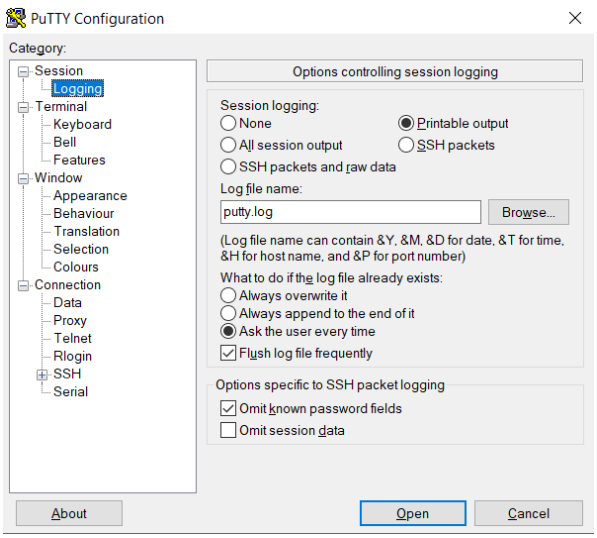
PuTTY: Another versatile option is the free 'PuTTY' program (www.putty.org). It supports a wide range of operating systems and includes a very nice logging function. To use, connect up the USB cable to the PC and start Putty. Configure as shown here – selecting the Serial COM port which your USB serial port is associated with, and setting the speed to 9600 and clicking the Connection Type: Serial button.



Next click on the ”Terminal” in the Category tree on the left side of the PuTTY window and duplicate the settings shown here:

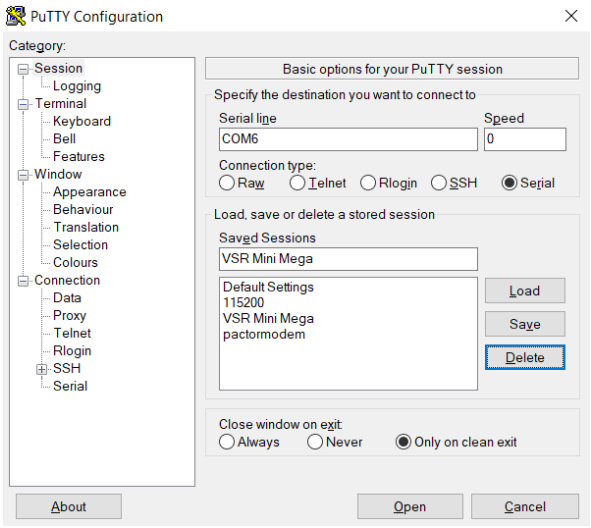


If you wish to keep a log file of the session, click on “Logging”, under ”Session”, in the Category tree on the left side of the PuTTY window, enter a file name and mimic the settings shown here:



Logging sessions is very helpful for debugging your installations, the files are comma separated and easily import into Excel using the import wizard specifying commas (,) as the separator. Refer to *Appendix A: Receiving data FROM the VSMMAR*: for details on the output.

Finally, after making all these settings, consider returning to the “Session” item in the Category tree and saving the PuTTY configurations for later by typing a new name in the “Saved Sessions” space and then clicking “Save”. The new session name will appear in the list of saved sessions. Next time you open Putty, you can click on the name of that saved session and then click “Load” to reload all of the setting.



Once you have completed the configuration, or when you reload the settings, click the “Open” button to start the terminal session.

Bench-top Configuration: When a USB cable is connected to the VSMMAR and PC, power is supplied to the logic portion of the hardware. This allows you to do bench-top configuration before completing the installation and connecting the actual alternator, battery, shunts, etc.

Again… remember to be sure to send a $RBT: command as your last step, to assure changes are recorded. (And add an “@” to the end of each command line if needed to get the VSMMAR to respond to the commands). After the regulator reboots make sure to verify your changes before installing the regulator in a live installation.

OVERVIEW OF ASCII COMMUNICATIONS WITH THE VSMMAR

- The VSMMAR outputs status strings on a continuous basis.
- The automatically output status strings are divided into two categories: major and minor
- The minor strings are output once for every ten times the major strings are output. So, for example, ten $AST strings will be followed by one $SST string.
- ASCII message strings can be sent to the VSMMAR to either
    a. request a particular message string be output by the VSMMAR (for example: $CPE:0 asks the VSMMAR to return the current CPE settings), or
    b. set parameters of the VSMMAR by sending the appropriate string name and variables.
- The syntax of commands sent to the VSMMAR must be exactly correct to be accepted. Confirm all intended setting changes by sending a request command to output the new settings and reviewing those settings before committing the changes via the $RBT: command.

- NOTE: While all this capability exists, “programming” the VSMMAR via ASCII commands can be tedious. Once the “Dashboard” tool supports the VSMMAR, consider using that tool instead to make system changes and to

monitor the VSMMAR. The Dashboard tool, written by Rick Bell, hides all of this complexity behind a very easy to use graphical interface.

APPENDIX A: RECEIVING DATA FROM THE VSR SYSTEM DEVICES:

Messages which will be output by the VSMMAR:

AST; -- ALTERNATOR STATUS

CPE; -- CHARGE PROFILE ENTRY

SCV; -- SYSTEM CONFIGURATION

SST; -- SYSTEM STATUS

FLT; -- FAULTED

AOK; -- ACKNOWLEDGE

DBG; -- DEBUG

RST; -- RESET

All status outputs are suspended during the receiving and processing of a command string. In this way, a command which expects a response (for example: $RSC:) can be assured the next string sent back by the regulator is the response to the request command (though one should still do error checking and validation, as the simple regulator will often just ignore commands which have syntax errors) .

Formats are all in clear, if not long, ASCII strings using comma-separated fields. Note the presence of double commas (separated by a space) between major 'sections'; this is to simplify manual reading of the strings. Each string is delivered as one continuous line, terminated by a CR/LF.

Additional details of each output string may be discovered by examine the command string for changing those parameters.

AST; ALTERNATOR STATUS

AST: "AST;, Hours, , BatVolts, AltAmps, BatAmps, SystemWatts, ,TargetVolts, TargetAmps, TargetWatts, AltState, ,BTemp, ATemp, ,RPMs, , AltVolts, FTemp, FAmps, FLD%"

Hours:	Time regulator has been powered up, in hours and fraction (to 2 digits) of hours.
BatVolts:	Derived Battery Volts, in volts and fractions of volts (to 1mV resolution). Used to decide change mode changes.
AltAmps:	Measured Alternator Amps, in Amps and fraction of Amps (to 1/10th of an Amp)
BatAmps:	Measured Battery Amps (you are using both shunts right?) being used to decide charge modes. Voltage and current readings made by the VSMMAR are directly reported as *AltVolts* and *AltAmps*.
SystemWatts:	Current measured System Watts being delivered.
TargetVolts:	Volts the regulator is attempting to bring the battery to. This value is the *actual* voltage value being driven to and reflects the adjusted charge profile entry (CPE) and the sysVolts index value.
TargetAmps:	Amps the regulator will limit the alternator to. This value is the *actual* amperage being driven to and reflects the derating, half power mode adjustments, and or SCUBA-mode adjustments. *Note: If the VSMMAR is configured with no limits for alternator amps and/or system watts a self-imposed limit of 1,000A / 15,000W is set as the maximum values. AST; will report these working values. To use on larger systems, you will need to modify the source code and proceed at your own risk.*
TargetWatts*:	Watts the regulator is actually working to limit the system to.
ChargingState:	Current state of the Alternator, encoded as follows: 0 = unknown, 1 = disabled, 2 = faulted, 3 = faulted, reduced load, 4 = sleeping, 10 = warm-up, 11 = ramping, 12 = determining alternator capacity, 20 = bulk charge, 21 = acceptance charge, 22 = overcharge 30 = float charge, 31 = forced float charge, 32 = LiFePO forced shutdown, 36 = post float, 38 = equalize
BTemp:	Measured temperature of NTC sensor attached to bat temp port, in degrees. -99 indicates temperature has not been measured, NTC sender has failed or is not attached.
ATemp:	Measured temperature of NTC sensor attached to alt temp port, in degrees C. -99 indicate temperature has not been measured, or NTC sender has failed. -100 indicates the Alternator temp NTC probe is shorted (to select half-power mode)

RPMs:	Measured RPMs of engine (Derived from alternator RPMs and the engine/alternator drive ratio)
AltVolts:	Measured alternator volts, in volts and fractions of volts (to 1mV resolution)
FET Temp:	The temperate of the FETs in degrees C. -99 indicated FET temperate cannot be measured.
Field Amps:	This is a measurement of the current (amperage) being delivered to the field.
	-99 indicated field current is not being measured.
Field %:	% (0..100%) field is being driven.

CPE; CHARGE PROFILE ENTRY

In response to RCP: command, the CPE string displays the current values of a charge profile entry (CPE). Special note on charge profile entries: All voltage and current values in charge profile tables are displayed in their normalized '12-volt' values. See Defining Charging Voltages and Amps for additional information.

"CPE;, n, ACPT_BAT_V_SETPOINT, EXIT_ACPT_DURATION, acptEXIT, reserved_1, , LIMIT_OC_AMPS, EXIT_OC_DURATION, EXIT_OC_VOLTS, reserved_2, ,FLOAT_BAT_V_SETPOINT, LIMIT_FLOAT_AMPS, EXIT_FLOAT_DURATION, FLOAT_TO_BULK_AMPS, FLOAT_TO_BULK_AHS, FLOAT_TO_BULK_VOLTS, , EXIT_PF_DURATION, PF_TO_BULK_VOLTS, PF_TO_BULK_AHS, ,EQUAL_BAT_V_SETPOINT, LIMIT_EQUAL_AMPS, EXIT_EQUAL_DURATION, EXIT_EQUAL_AMPS, , BAT_TEMP_1C_COMP, MIN_TEMP_COMP_LIMIT, BAT_MIN_CHARGE_TEMP, MaxCharge"

n:	Charge Profile number is being displayed/returned (1..8)
ACPT_BAT_V_SETPOINT:	Target battery voltage during bulk and acceptance phases
EXIT_ACPT_DURATION:	Time limit to stay in acceptance phase – in minutes.
EXIT_ACPT_AMPS:	Amp limit to trigger exiting acceptance mode
reserved_1:	Reserved for future use. Nothing implemented yet. Displays 0.
LIMIT_OC_AMPS:	Max Amps which will be supplied by during OVERCHARGE mode.
EXIT_OC_DURATION:	Time limit to stay in overcharge phase – in minutes.
EXIT_OC_VOLTS:	Target battery voltage during overcharge phase

reserved_2: Reserved for future use. Nothing implemented yet. Displays 0.

FLOAT_BAT_V_SETPOINT: Target battery voltage during float phase

LIMIT_FLOAT_AMPS: Max amps which will be supplied by during float phase.

EXIT_FLOAT_DURATION: Time limit to stay in float mode – in minutes.

FLOAT_TO_BULK_AMPS: Amp limit to trigger resumption of bulk charge mode

FLOAT_TO_BULK_AHS: Amp hours withdrawn after entering float to trigger resumption of bulk charge

FLOAT_TO_BULK_VOLTS: Volt limit to trigger resumption of bulk charge mode

Note: If the regulator is in forced_float mode via a FEATURE-IN port, then none of the above checks to exit float mode (for example, EXIT_FLOAT_DURATION) will be performed. However, regulation will still occur to FLOAT_BAT_V_SETPOINT and LIMIT_FLOAT_AMPS.

EXIT_PF_DURATION: Time limit to stay in post float mode – in minutes, before resuming float charge

PF_TO_BULK_VOLTS: Battery Voltage that will trigger resumption of float charge mode

PF_TO_BULK_AHS: Amp hours withdrawn after entering post float to trigger resumption directly to bulk charge mode

EQUAL_BAT_V_SETPOINT: Target battery voltage during equalize phase

LIMIT_EQUAL_AMPS: Current limit of alternator while in equalize mode

EXIT_EQUAL_DURATION: Time limit to stay in equalize mode – in minutes.

EXIT_EQUAL_AMPS: Amp limit to trigger exiting equalize mode

BAT_TEMP_1C_COMP: Temperature compensations value per 1-degree C (normalized to ’12-volt’ battery)

MIN_TEMP_COMP_LIMIT: Minimum temperate at which temperature compensation will still be applied. In degrees C

BAT_MIN_CHARGE_TEMP:	Minimum temperature at which charging will be allowed. Below this temperature, regulator will be forced into float mode.
BAT_MAX_CHARGE_TEMP:	Maximum temperate at which charging will be allowed. Above this temperature, regulator will be forced into float mode.

SCV; SYSTEM CONFIGURATION

"SCV;, CONFIG_LOCKOUT, REVERSED_BAT_SHUNT, REVERSED_ALT_SHUNT, SV_OVERRIDE, BC_MULT_OVERRIDE, CP_INDEX_OVERRIDE, , ALT_TEMP_SETPOINT, ALT_AMP_DERATE_NORMAL, ALT_AMP_DERATE_SMALL_MODE, ALT_AMP_DERATE_HALF_POWER, ALT_PULLBACK_FACTOR, , ALT_AMPS_LIMIT, ALT_WATTS_LIMIT, , ALTERNATOR_POLES, ENGINE_ALT_DRIVE_RATIO, BAT_AMP_SHUNT_RATIO, ALT_AMP_SHUNT_RATIO, , ALT_IDLE_RPM, MIN_FIELD_TACH_PWM, ENGINE_WARMUP_DURATION, REQURED_SENSORS);

CONFIG_LOCKOUT:	Current lockout level. (0..2), see $SCO: command.
Reserved:	Always = 0; *(Was Favor 32v system detection over 24/48v?)*
REVERSED_BAT_SHUNT:	0 or 1: Reverse polarity of battery shunt readings? (1 = yes)
REVERSED_ALT_SHUNT:	0 or 1: Reverse polarity of alternator shunt readings? (1 = yes)
SV_OVERRIDE:	System voltage auto-detect (=0), or force (1.0x .. 2.0x → 12v .. 24v)
BC_MULT_OVERRIDE:	Override battery Capacity DIP switches (Dip 4,5). (0.00 = No)
CP_INDEX_OVERRIDE:	Override charge profile DIP Switches (Dip 1,2,3) (0 = No)
ALT_TEMP_SETPOINT:	Target max running temperature for alternator, in degrees C
ALT_AMP_DERATE_NORMAL:	Normal amp reduction (de-rating) fraction
ALT_AMP_DERATE_SMALL_MODE:	Amp reduction (de-rating) fraction when in small alternator mode
ALT_AMP_DERATE_HALF_POWER:	Amp reduction (de-rating) fraction when in half-power mode.
ALT_PULLBACK_FACTOR:	Pull-back factor, for reducing field drive at lower RPMs.

ALT_AMPS_LIMIT: Defined alternator size, or -1 to enable auto-sizing. Set this = 0 for installations where alternator sizing is not to be regulated (for example in battery-focused installations).
Note: During startup, and unless defined, this value will present: 1,000

ALT_WATTS_LIMIT: Defined system size, or -1 to enable auto-sizing. Set this = 0 for installations where watts loading is not to be regulated (for example in a battery-focused installation).
Note: During startup, and unless defined, this value will present: 15,000

ALTERNATOR_POLES: Number of poles on alternator

ENGINE_ALT_DRIVE_RATIO: Ratio of engine and alternator drive pulley

BAT_AMP_SHUNT_RATIO: Battery Shunt ratio in amps / mV

ALT_AMP_SHUNT_RATIO: Alternator shunt ratio in amps / mV

ALT_IDLE_RPM: Idle RPM value used as basis for Field Drive Reduction at lower RPMs.

MIN_FIELD_TACH_PWM: Minimum % of field drive that will be applied if tach mode is enabled.

ENGINE_WARMUP_DURATION: Number of seconds after power on before entering ramping mode.

REQURED_SENSORS: Key indicating critical sensors which have been configured via the $SCA: command

SST; SYSTEM STATUS

"SST; , firmwareVersion, , smallAltMode, tachMode, , cpIndex, systemAmpMult, systemVoltMult, , altCapAmps, altCapRPMs, , Accumulated Amp Hours, Accumulated Watt Hours, , Forced Tach Mode"

firmwareVersion: Firmware revision identifier. Will have format of "AREG" followed w/o a space by the version number. E.g., "AREG1.3M"

SmallAltMode: 0 or 1, has the user selected Small Alternator Mode? (1 = yes)

tachMode: 0 or 1, has user selected Tach Mode? (1 = yes)

cpIndex: Which charge profile (1..8) is currently being used?

systemAmpMult: What adjustment factor for Battery Amp Hour Capacity (1-10x) is currently being used?
Fractional values may also be used to fine tune the system to a given battery size. This needs to be entered via the $SCO: command.

systemVoltMult: Detected system voltage. Adjusts target charge profile volts per the following table:

SysVolt	Detected System Voltage	Charge Profile VOLTAGE Adjustment Factor
1	12v	1x
2	24v	2x

altCapAmps: If regulator is configured to auto-determine the capacity of the alternator, this will be the current high-water mark noted.

altCapRPMs: RPMs at which that capacity was noted.

Accumulated Amp Hours: The number of amp-hours that have been produced in the current charge cycle.

Accumulated Watt Hours: The number of Watt-Hours produced in the current charge cycle.

Forced Tach Mode: 0 or 1, has user forced tach-mode on via the $SCT: command.

FLT; FAULTED

FLT: "FLT;, FaultCode, RequiredSensorStatus "

System has Faulted, fault code number (See source code for details.)

RequiredSensorStatus is a combined number following the pattern as defined by the $SCA command and detailed in the Required Sensor section on page number 26

Following this, the AST, SST and SCV will be printed, as well as the currently active CPE.

AOK; ACKNOWLEDGE

Sent after a successfully receiving change command ($CPx, or $SCx), or $EDB

DBG; DEBUG

Special string with extra internal parameters. See Source code for details of Debug String

RST; RESET

Regulator has been requested to reset. This can take up to 10 seconds to complete.

APPENDIX B: SENDING DATA *TO* THE REGULATOR:

All commands begin with the character `$', contain 3 letters (CAPS) followed by a `:' and then parameters relevant to the command. All must end with a CR (or CR/LF) or may optional be terminated with the character `@' (this is to accommodate a bug in the Arduino IDE that does not send CR nor LF at the end of entered strings). A complete 'string' must be received within 60 seconds from the `$' to the ending `@'/CR/LF, or the regulator will abort the capture of that string command and begin looking for a new `$' starting character. (See *#define IB_BUFF_FILL_TIMEOUT 60000UL* in OSEnergy_Serial.h).

Sending communications TO the regulator:

$CPA:n - Change ACCEPT parameters in CPE user entry n (n = 7 or 8)

$CPO:n - Change OVERCHARGE parameters in CPE user entry n (n = 7 or 8)

$CPF:n - Change FLOAT parameters in CPE user entry n (n = 7 or 8)

$CPP:n - Change POST-FLOAT parameters in CPE user entry n (n = 7 or 8)

$CPE:n - Change EQUALIZE parameters in CPE user entry n (n = 7 or 8)

$CPB:n - Change BATTERY parameters in CPE user entry n (n = 7 or 8)

$CPR:n - RESTORES Charge Profile 'n' to default values

$RAS: - Request All Status back

$RCP:n - Request to send back CPE entry #N (n=1..8) **

$RCP:0 - Request to send back current selected CPE**

$SCA: - Changes ALTERNATOR parameters in System Configuration table

$SCT: - Changes TACHOMETER parameters in System Configuration table

$SCO: - Override features

$SCR: - RESTORES System Configuration table (+ Bluetooth) to default

$MSR: - RESTORE all parameters (to as defined at program compile time).

$EDB: - Enable DeBug serial strings

$RBT: - ReBooT system

$FRM: - Force Regulator Mode

Defining Charging Voltages and Amps – All volts and amps are represented for a normalized 12-volt 500Ah battery and are automatically scaled depending on the sampled battery voltage at startup and the setting of the battery capacity DIP switches.

Note that the MAXIMUM length of a string is fixed at 70 characters, including the line termination character(s). When assembling a command to send make sure not to exceed this length, remove any extra spaces if present to assure total length is under 70 characters.

****Note on Requesting Status commands.** All 'Request' commands will reply via the USB Serial port.

$CPA:n Change Acceptance parameters in CPE user entry n (n = 7 or 8)

This command (with its parameters) will cause the acceptance (and bulk) portion of a charge profile entry (CPE) to be updated. Parameters must be in the following order and include comma "," separators where indicated. Extra spaces before and/or after the parameters are allowed.

$CPA:n <ACPT_V_BAT_SETPOINT>, <EXIT_ACPT_DURATION>, <EXIT_ACPT_AMPS>,<Reserved>

---

n: (7 → 8) *'n'* is the charge profile table entry that will be modified. Use range 7 to 8.

ACPT_V_BAT_SETPOINT: <FLOATING POINT NUMBER (0.0 → 20.0) > Voltage the Regulator will use during acceptance phase. When this voltage has been reached, the regulator will transition from bulk to acceptance phase. This value is a floating-point number and entered for a normalized 12-volt system (See note: Defining Charging Voltages and Amps)

EXIT_ACPT_DURATION: <WHOLE NUMBER (minutes: 0 → 600 (10 hours)) > After entering acceptance phase, a timer will be started. After 'EXIT_ACPT_DURATION' minutes have expired acceptance mode will exit and the regulator will move to overcharge mode, if overcharge mode is defined for the charge profile, otherwise the regulator will move to float mode. Setting 'EXIT_ACPT_DURATION' = 0 will disable time based exiting of acceptance mode and only amp-based monitoring will be used.

EXIT_ACPT_AMPS: <WHOLE NUMBER (-1 → 200)> After entering acceptance phase, delivered amps will be monitored and if they fall to (or below) *'*EXIT_ACPT_AMPS' acceptance mode will exit and the regulator will move to overcharge mode, if overcharge mode is defined for the charge profile, otherwise the regulator will move to float mode. This is provided that the battery voltage is at the target VBat Set Point above (to prevent early exiting from low amps being delivered as a result of the engine slowing down to say very slow idle).

Setting *'*EXIT_ACPT_AMPS' = 0 will disable amp-based exiting of acceptance mode and only time monitoring to '*ExitDuration'* will be used as the exit criteria.

Setting *'*EXIT_ACPT_AMPS' = -1 will disable amp-based exiting of acceptance mode and time monitoring of 'EXIT_ACPT_AMPS'will be used as above. In addition, when the time spent in acceptance mode has exceed 5x the duration spent in bulk mode, the regulator will also trigger an exit. This is known as "adaptive acceptance").

Note: If you set BOTH 'EXIT_ACPT_AMPS*' and 'EXIT_ACPT_AMPS'* = 0, the regulator will bypass acceptance mode.

Reserved: <0> Placeholder for future. No action defined at this time. Must be set to 0.

Examples:

$CPA:7 14.5, 200, 40, 0 @ #7: 14.5VOLTS, EXIT AFTER 200 MINUTES OR UNDER 40AMPS

$CPA:8 12.4, 0, 20, 0@ #8: 12.4 VOLTS, EXIT ONLY ON AMPS UNDER 20

-

$CPA:810.4,0,20,0@ #8: 10.4 VOLTS, EXIT ONLY ON AMPS UNDER 20
(SHOWN WITH OPTIONAL '@' FOR ARDUINO IDE TERMINAL SUPPORT)

$CPO:n Change Overcharge parameters in CPE user entry n (n = 7 or 8)

This command (with its parameters) will cause the overcharge portion of a Charge Profile Entry (CPE) to be updated. Parameters must be in the following order and include comma "," separators where indicated. Extra spaces before and/or after the parameters are allowed.

$CPO:n <LIMIT_OC_AMPS>, <EXIT_OC_DURATION>, <EXIT_OC_VOLTS>, <Reserved>

---

n: (7 → 8) '*n'* is the charge profile table entry that will be modified. Use range 7 to 8.

LIMIT_OC_AMPS: <WHOLE NUMBER (0 → 50)> After entering overcharge phase, delivered amps will be monitored and regulated to 'LIMIT_OC_AMPS'. Setting LIMIT_OC_AMPS' = 0 will disable overcharge mode.

EXIT_OC_DURATION: <WHOLE NUMBER (minutes: 0 → 600 (10 hours)) > After entering overcharge phase, a timer will be started. After 'EXIT_OC_DURATION' minutes have expired, overcharge mode will exit and the regulator will move to float mode. Setting 'EXIT_OC_DURATION'= 0 will disable overcharge mode.

EXIT_OC_VOLTS: <FLOATING POINT NUMBER (0.0 → 20.0) > Once battery voltage reached 'EXIT_OC_VOLTS', overchage phase will be exited. This value is a floating-point number and entered for a normalized 12-volt system (See: Defining Charging Voltages and Amps). Setting 'EXIT_OC_VOLTS' = 0 will disable OVERCHAGE mode.

Reserved: <0> Place holder for future. No function at this time. Must be set to 0.

$CPF:n Change float parameters in CPE user entry n (n = 7 or 8)

This command (with its parameters) will cause the float portion of a charge profile entry (CPE) to be updated. Parameters must be in the following order and include comma "," separators where indicated. Extra spaces before and/or after the parameters are allowed.

$CPF:n <FLOAT_BAT_V_SETPOINT>, <LIMIT_FLOAT_AMPS>, <EXIT_FLOAT_DURATION>, <FLOAT_TO_BULK_AMPS>, <FLOAT_TO_BULK_AHS>, < FLOAT_TO_BULK_VOLTS >

---

n: (7 → 8) '*n'* is the charge profile table entry that will be modified. Use range 7 to 8.

FLOAT_BAT_V_SETPOINT: <FLOATING POINT NUMBER (0.0 → 20.0) > Voltage the regulator will use during float phase. This value is a floating-point number and entered for a normalized 12-volt system (See note: Defining Charging Voltages and Amps)

LIMIT_FLOAT_AMPS: <WHOLE NUMBER (-1 → 50)> While in float phase delivered amps will be monitored and regulated to 'LIMIT_FLOAT_AMPS *s'* . Setting 'LIMIT_FLOAT_AMPS' = -1 will disable this feature and only 'FLOAT_BAT_V_SETPOINT' will be regulated. 1

EXIT_FLOAT_DURATION: <WHOLE NUMBER (0 → 30000 (500 hours)) > After entering float phase, a timer will be started. After 'EXIT_FLOAT_DURATION' minutes have expired float mode will exit and the regulator will move to postfloat mode. Setting 'EXIT_FLOAT_DURATION' = 0 will cause the regulator to remain in float mode unless triggered by another exit criteria.

FLOAT_TO_BULK_AMPS: <WHOLE NUMBER (-300 → 0)> While in float mode, if *'FLOAT_TO_BULK_AMPS'* are exceeded it is an indication that a large load has been placed on the battery and current is being withdrawn, the regulator will restart a charge cycle, looping back to bulk mode. Setting *'FLOAT_TO_BULK_AMPS'* = 0 will disable this feature.

FLOAT_TO_BULK_AMPS is most useful in the case where there is an amp shunt is placed on the battery, as when the amp draw from the battery exceeds *FLOAT_TO_BULK_AMPS*, it is a clear indication energy is being drawn from the battery. In this case, set *FLOAT_TO_BULK_AMPS* equal to the number of amps being drawn from the battery that should be used to trigger a return to bulk mode.

In cases where the amp shunt is installed on the Alternator, this can also be of use by sizing *FLOAT_TO_BULK_AMPS* to a value slightly above expected house load values. However, perhaps a better indication is to set this to =0, and use FLOAT_TO_BULK_VOLTS.

FLOAT_TO_BULK_AHS: <WHOLE NUMBER (-250 → 0)> After entering float mode if the accumulated number of amp hours removed from the battery exceeded *'FLOAT_TO_BULK_AHS'* the regulator will re-start a charge cycle, looping back to bulk mode. Setting '*FLOAT_TO_BULK_AHS'* = 0 will disable this feature

This is another way to indicate the need to restart charging of the battery, and perhaps a better approach than using raw 'FLOAT_TO_BULK_AMPS', but it is only usable with an amp shunt placed on the battery.

FLOAT_TO_BULK_VOLTS: <WHOLE NUMBER (0.0 → 20.0)> While in float mode, if battery voltage drops below 'FLOAT_TO_BULK_VOLTS' we assume this indicates a large load has been placed on the system and the regulator will restart a charge cycle, looping back to bulk mode. Setting 'FLOAT_TO_BULK_VOLTS' = 0 will disable this feature.

In determining to exit float mode, a rolling average value for measured amps and volts is used. This way short term events (e.g., a surge of a refrigerator starting up and before the alternator can respond) will not pull the regulator out of float mode. Note also that the revert amp –hours is a negative value and measures the number of amp hours removed from the battery after entering float mode.

$CPP:n Change Postfloat parameters in CPE user entry n (n = 7 or 8)

This command (with its parameters) will cause the postfloat portion of a charge profile entry (CPE) to be updated. Parameters must be in the following order and include comma "," separators where indicated. Extra spaces before and/or after the parameters are allowed.

$CPP:n <EXIT_PF_DURATION>, <PF_TO_BULK_VOLTS>, <PF_TO_BULK_AHS>

---

n: (7 → 8) '*n*' is the charge profile table entry that will be modified. Use range 7 to 8.

EXIT_PF_DURATION: <WHOLE NUMBER (minutes: 0 → 30000 (500 hours)) > After entering postfloat phase, a timer will be started. After 'EXIT_PF_DURATION' minutes have expired v mode will exit and the regulator will revert to float phase. Setting 'EXIT_PF_DURATION' = 0 will disable postfloat mode revering to float charge immediately.

PF_TO_BULK_VOLTS: <<FLOATING POINT NUMBER (0.0 → 20.0)> While in postfloat mode, if the system battery voltage drops below 'PF_TO_BULK_VOLTS ' it is an indication that a large load has been placed on the system and the regulator will re-start a charge cycle, looping back to bulk mode. Setting 'PF_TO_BULK_VOLTS' = 0 will disable this feature.

PF_TO_BULK_AHS: <WHOLE NUMBER (-250 → 0)> After entering postfloat mode if the accumulated number of amp hours removed from the battery exceeded 'PF_TO_BULK_AHS' the regulator will re-start a charge cycle, looping back to bulk mode. Note that this trigger goes directly to bulk, as opposed to back to
float mode. Setting 'PF_TO_BULK_AHS'= 0 will disable this feature

$CPE:n Change Equalize parameters in CPE user entry n (n = 7 or 8)

This command (with its parameters) will cause the Equalize portion of a charge profile entry (CPE) to be updated. Parameters must be in the following order and include comma "," separators where indicated. Extra spaces before and/or after the parameters are allowed.

$CPE:n < EQUAL_V_BAT_SETPOINT >, < LIMIT_EQUAL_AMPS' >, < EXIT_EQUAL_DURATION >, < EXIT_EQUAL_AMPS >

---

n: (7 → 8) '*n*' is the charge profile table entry that will be modified. Use range 7 to 8.

EQUAL_V_BAT_SETPOINT: <FLOATING POINT NUMBER (0.0 → 25.0) > Voltage the regulator will use during equalize mode. This value is a floating-point number and entered for a normalized 12-voly system (See note: Defining Charging Voltages and Amps). Setting 'EQUAL_V_BAT_SETPOINT' = 0 will disable equalize mode.

LIMIT_EQUAL_AMPS: <WHOLE NUMBER (0 → 50)> Optional additional current limit while in equalize phase; the regulator will cap delivered amps to 'LIMIT_EQUAL_AMPS' . Setting 'LIMIT_EQUAL_AMPS' = 0 will disable this amperage capping.

EXIT_EQUAL_DURATION: <WHOLE NUMBER (minutes: 0 → 240 (4 hours)) > After starting an equalize phase, a timer will be started. After 'EXIT_EQUAL_DURATION' minutes have expired, equalize will terminate and the regulator will enter float mode. Setting 'EXIT_EQUAL_DURATION' = 0 will disable equalize mode.

EXIT_EQUAL_AMPS: <WHOLE NUMBER (0 → 50)> During equalize mode delivered amps will be monitored and if it falls to (or below) 'EXIT_EQUAL_AMPS' , equalization will be terminated and the regulator will move to float mode. Note that as a precaution, battery voltage is not checked when sampling Equalization Exit Amps (as it is in acceptance and overcharge modes). It is up to the operator to keep the engine speed up and allow for a full equalization session to occur. Setting 'EXIT_EQUAL_AMPS' = 0 will disable amp-based exiting of equalize mode and only time monitoring will be used.

$CPB:n Change Battery parameters in CPE user entry n (n = 7 or 8)

This command (with its parameters) will cause the remaining portion of a charge profile entry (CPE) to be updated. Parameters must be in the following order and include comma "," separators where indicated. Extra spaces before and/or after the parameters are allowed.

$CPB:n <VBat Comp per 1^{O}C>, < Min Comp Temp >, <Min Charge Temp>, <Max Charge Temp>

---

n: (7 → 8) '*n*' is the charge profile table entry that will be modified. Use range 7 to 8.

VBat Comp: <FLOATING POINT NUMBER (0.0 → 0.1)> This is used to adjust all target VBat voltages based on the current battery temperature in 1 degree C increments. This value is a floating-point number and entered for a normalized 12-volt system at 25c. (See note: Defining Charging Voltages and Amps). Set = 0.0 to disable temperature-based voltage compensation.

Min Comp Temp: <WHOLE NUMBER (-30 → 40)> Additional compensation to battery target voltages will be stopped when the battery is at or below this temperature in degrees C.

Min Charge Temp: <WHOLE NUMBER (-50 → 10) > If the battery drops below this temperature, the system will be forced into float mode to protect it.

Max Charge Temp: <WHOLE NUMBER (20 → 95) > If the battery reaches this temperature, the system will be forced into float mode to protect it. If the battery temperature continues to raise, the system may eventually fault based on the value of #define FAULT_BAT_TEMP in the source code (60c (140F) by default).

$CPR:n - RESTORES Charge Profile 'n' to default values

Restores to default (values at compile time) charge profile entry (CPE) '*n*'. After entry '*n*' is restore, the regulator will be restarted automatically.

$CPR:n

---

n: (7 → 8) '*n*' is the charge profile table entry that will be restored. Use range 7 to 8.

$RAS: Request All Status back

This command will instruct the VSMMAR to send out, via the USB erial port, a copy of the all known status strings. It is useful for external applications to capture the current status without needing to await the arrival of each string (which could take several minutes or more, depending on how status strings are paced out).

$RAS:

After all status strings have been sent, "AOK" will be sent to indicate the end.

$RCP:n Request to send back CPE entry #N (n=1..8) **

This command will instruct the VSMMAR to send out, via the USB Serial port, the saved contents of the CPE entry N, where N is a number from 1 to 8. See "CPE: Charge Profile Entry" for description of resulting transmission.

$RCP:n

$RCP:0 - Request to send back current selected CPE**

Special version of request for CPE, this will send back the currently selected (via the DIP switches) CPE.

$RCP:0

Note, the CPE entry sent back in response to a $RCP: command will reflect the current values contained in flash memory which may not match what the regulator is currently working with. If a CPE has been modified and

saved to flash, those modifications will be reflected. However, until the VSMMAR is rebooted it will not utilize those values. For current active targets being used, look at AST; and SCV; status strings.

N**ote on Change Requests to Charge Profiles:** The source code currently will allow only charge profile 7 or 8 (the two customizable entries) to be modified via an ASCII command. This is to reduce the potential for major errors in the regulator. If you wish to modify other charge profiles, the source code will need to be revised to either change the default tables, or alter the trap to allow changes via ASCII beyond entry 7 or 8.

Also, take great care in setting these values, especially the exiting time and amp thresholds. Some of these thresholds can be disabled by setting to 0, disabling that threshold test. If both amps & time values are disabled, it is possible for the regulator to stay in a full charge state indefinitely, likely causing damage to the battery. As the VSMMAR may be deployed with either a battery shunt or alternator shunt, of with both a battery and alternator shunt (which is the recommended configuration), the CPE entries will behave differently depending on which deployment model is used. And some entries might have no meaning. Great Flexibility results in Great Responsibility…

All change profile commands will reply with "AOK;" if the command was processed successfully. However, to assure the changes are stored and used, the regulator must be reset using the $RBT: command after all changes have been made.

$SCA: Changes Alternator parameters in System Configuration table

Used to update the system configuration table entries associated with the alternator.

$SCA: <reserved>, < Alt Target Temp >, <Alt Derate (norm) >,<Alt Derate (small) >, <Alt Derate (half) >, <PBF>, <Alt Amp Cap >, <System Watt Cap. >, < BAT_AMP_SHUNT_RATIO >, < BAT_AMP_SHUNT_RATIO>, < REVERSED_BAT_SHUNT>, < REVERSED_ALT_SHUNT: >, <Idle RPMs>,<Warmup Delay>,<RequiredSensors>

Reserved: <WHOLE NUMBER = 0> Not used. Must be set to 0.

Alt Target Temp: <WHOLE NUMBER (15 → 120)> Operating temperature under which the regulator should attempt to keep the alternator. If the alternator temperature exceeds this value, the regulator will reduce field current to allow the alternator to cool off. If the Alternator temperature continues to

rise and exceeds this temperature by 10% (as defined by #define FAULT_ALT_TEMP) the regulator will fault out and stop all power production.

Alt Derate(norm), Alt Derate(small),
Alt Derate(half): <FLOATING POINT NUMBER (0.10 → 1.00) > These derating values are used to limit the alternator's maximum current output to some percentage (10% to 100%) of its demonstrated capability (see *Alt Amp Cap*). The three values correspond to the following alternator modes:

- Normal - Condition when neither of the other modes is selected.
- Small Alternator Mode – selected via DIP switch 6 (or the override via $SCT command)
- Half-Power Mode – Selected by shorting the alternator NTC temperature sensor wires.

In operation, de-rating values are applied to both the Alt Amp Cap as well as the internal maximum field PWM drive. In this way, a smaller alternator is protected, even if an alternator shunt is not connected.

PBF: < INTEGER (-1 → 10)> Pullback factor for reducing Field Drive at lower RPMs. If the VSMMAR is able to determine RPMs (via the stator wire), the Alternator Field Drive will be reduced when the regulator detects the engine is at Idle. At idle the max PWM will be capped at around 1/4 of full field, which should result in some current bring produced. As RPMs are increased, this 'Field Drive Capping' will slowly be removed. PBF determines how quickly this pull-back is scaled off.

Set = 0 to disable this feature.

Set = -1 (DEFAULT) to cause Field Drive to be reduced to a maximum of 70% drive in the case where the VSMMAR is no longer able to measure RPMs via the stator-in signal. This might be, for example, where an engine is operating at extremely low RPMs, below the cut-in point for the alternator. Or where the engine is no longer running. The 70% limit will only be enabled if at one time during operation the regulator was able to measure RPMs successfully.

For many engine / alternator combinations the default value of 1 should result in good operations. However, if you have installed a large alternator on a rather modest sized engine, you might notice the engine struggles when trying to increase RPMs from idle. In that case, increase the PBF value. A factor of 8x or so might be needed in the case of a small sailboat engine with a large 150-amp or greater alternator (consider also using the Alt Amp Cap and/or System Watts Cap capabilities as well to restrict maximum engine loading at higher RPMs).

If the engine has a large capacity relative to the alternator size, consider reducing the PBF to 1. Doing so will allow a greater production of amps while at idle, while at the same time preventing the alternator from being driven at Full Field during low RPMs (and hence low cooling)

Finally, if your system matches an engine with great capability, and the alternator has good cooling / heat management – you can set the PBF factor = 0 to disable any capping of field drive while the engine is at idle. This will allow for maximum alternator output at idle, however if the regulator is enabled but the engine is not actually running, field drive will increase to Full Field until a fault check causes the regulator to reset. Do not leave the 'ignition' in the on position, without the engine actually running to prevent

this situation. It would be advisable to assure there is a temperate sensor attached to the alternator in this case – to prevent unintended overheating during prolonged idle periods.

Note: Field Pullback is dependent upon the stator sensing wire being connected to the alternator. If the regulator is unable to reliability sense RPMs, all idle pullback features will be disabled. Note also that one should make sure to configure the tachometer via the *$SCT:* command.

Alt Amp Cap: <WHOLE NUMBER (-1→ 500) > This regulator will limit the amperage output of the alternator to this value, after applying the '*Alt Derate xxx*' factors. There is no adjustment made to this value based on system voltage or selection of system battery size – the values declared will be used directly. A special feature is enabled by setting this = -1: the regulator will drive the alternator as hard as it can for a short period of time when first entering bulk phase and in this way will auto-sample the alternator size based on its capabilities.

Note: ***Alt Amp Cap*** is a feature intended to be used in alternator-centric deployments (refer to the VSMMAR Reference Manual) and **should *only* be defined if there is a current shunt located *at the alternator*.** During operation the regulator will limit measured amps to *Alt Amp Cap* value. During reduced power modes (i.e., – Half-Power mode) the amperage allowed will be further limited by the appropriate scaling factor.

For battery-Centric deployments this value should be set = 0, thereby disabling measured amperage limits of the alternator. With amperage limits disabled, reduced power modes will apply the scaling factor to the PWM duty cycle. It should be noted that there may not be a direct relation between reductions in PWM duty cycles and delivered amps – care should be used when setting up the system. (Default = 0, disabled)

System Watts Cap: <WHOLE NUMBER (-1→ 20000)> The regulator will limit the system wattage to this value. Its primary use is to protect the driving engine and/or belts – by limiting the maximum amount of work the engine is asked to do on behalf of the alternator. (Work being a function of both volts and amps, hence watts). It may also be used to limit the total amount of power being delivered into the battery by all charging sources. There is no derating or adjustment made to this value based on system voltage or selection of system battery size. System Watts Capacity is used to after applying the '*Alt Derate xxx*' factors. It is used to protect the alternator from over-current usage. A special feature is enabled by setting this = -1, the regulator will drive the alternator as hard as it can for a short period of time when first entering bulk phase. This will then be used to define the amp Limit of the aternator.

> Note on Alt Amps and System Watts: You may set either of these parameters =-1 to allow the regulator to automatically calculate limits based on the sampled capability of the alternator, or set them = 0 to disable that feature. Though these two are interlaced, they are indeed separately monitored and adjustments to the Field PWM are made independently for each.

(Default = 0, disabled)

BAT_AMP_SHUNT_RATIO:
ALT_AMP_SHUNT_RATIO: <WHOLE NUMBER 500 → 20000)> Enter the ratio of your amp measurement shunts in terms of amps / mVolts. e.g., if you have a 250A / 75mV shunt, you would enter

3333 (250/0.075). And you may adjust the number to allow for fine tuning of the amp shunt. e.g., if your shunt has a 3% error, you could enter 3433

Caution: Shunt voltage is limited to +/-80mV. Do NOT exceed this value!

REVERSED_BAT_SHUNT:
REVERSED_ALT_SHUNT: <WHOLE NUMBER (0, or 1) > Allows software correction if an amp shunt was wired backwards. Set = 1 and amp readings will have their polarity changed.

Idle RPMs: < INTEGER (0 → 1500)> Used in conjunction with PBF to manage Field Drive at lower RPMs. As
RPMs rise above Idle RPMs, field drive will be increased at a rate determined by PBF. During normal operation, Idle RPMs can be detected automatically by the VSMMAR. However, in more sensitive installations where the management of the alternator Field Drive at low RPMs is critical, additional system reliability can be achieved by defining the IDLE RPMs value to be used in all calculations. In extreme installations (for example, a very small engine with large efficient alternator), Idle RPMs may be defined artificially high; doing so will cause the regulator to increases its pullback of Field Drive during low RPM operations.

Set = 0 (DEFAULT) to enable 'auto' determination of Idle RPM.

Warmup Delay: <WHOLE NUMBER (seconds: 15 → 600) > Hold-off period when regulator is first powered on before it will begin to apply a load to the engine. This is the number in seconds of delay the regulator will remain in pre-ramp mode before moving into ramping mode. Default = 15 seconds.

Required Sensors: <WHOLE NUMBER (0 → 255) > Many capabilities depend on the presence of sensors. Battery compensation requires the presence of a battery temperature sensor; Alternator temperature regulation requires the presence of an alternator temperature sensor. If one or more of these sensors are not installed, or fail during operation, results could be less then desired. As a precaution against this, *Required Sensors* allows the identification of critical sensors for your installation, and if any of them are missing or fail the regulator will take action to reduce demands placed on the system.

Required Sensors allows the identification of critical sensors. It is a number created by summing up the value associated with each potential critical sensor. For example: if you wished to indicate the alternator and battery temperature sensors are critical, you would enter 3 (1+2). The value of 0 disables critical Required Sensor checks and the regulator will utilize other existing fallback modes.

Sensor	**Value**	**Default Action of missing sensor**
Alternator Temperature Sensor	1	Enable Half-Power mode
Battery Temperature Sensor	2	Force to FLOAT mode
Battery Current Shunt	4	FAULT regulator (See note**)
Alternator Current Shunt	8	FAULT regulator (See note**)

Force FAULT override	128	Overrides 'Default' action and forces regulator into FAULT mode.

Table 1 - Required Sensor Encoding

If at any time one of the Required Sensors are identified as failed or missing the LED will flash its normal patterns, but in RED. In addition, if a Feature_out port is configured to drive a dash-lamp it will turn on the lamp fulltime indicating a fault.

The VSMMAR may also be configured to cause a non-recoverable FAULT condition, overriding the default actions listed in Table 1 by adding 128 to the summed number. In the prior example of battery and alternator temperatures sensors being critical, sending 131 instead of 3 will cause the regulator to FAULT if either is noted as missing or fails.

Note** It is difficult to determine if an amp shunt has failed vs. if are truly reading zero current. Because of this, the VSMMAR will delay checking for the presence of a working current shunt until after bulk mode has been completed. If at any time during bulk mode a current of greater than 5A was noted, it will be flagged as the shunt being present and working. Once this determination is made no additional checks will be made – as a valid operating condition for the regulator is a true zero current (example, when actively regulating current to zero amps in float mode).

$SCA: will reply with "AOK;" if the command was processed successfully. However, to assure the changes are stored and used the regulator must be reset using the $RBT: command after all changes have been made.

Example: Large alternator powered by large main engine:

- Disable Idle adaptive pullback (alternator has massive cooling capability, and engine has sufficient reserve)
- Adjust amp shunt to 250A/75mV (as appropriate for your shunt)
- Define engine idle @ 550 RPMs
- Must have alternator temperature sensor present, else cause regulator to FAULT
- Leave other values as default (See *Appendix D:)*

Command Syntax: (color-coded to make it easier to map values to their names)
$SCA: <reserved>, < Alt Target Temp >, <Alt Derate (norm) >,<Alt Derate (small) >, <Alt Derate (half) >, <PBF>, <Alt Amp Cap >, <System Watt Cap. >, < BAT_AMP_SHUNT_RATIO>,< BAT_AMP_SHUNT_RATIO>, < REVERSED_BAT_SHUNT>, < REVERSED_ALT_SHUNT >, <Idle RPMs>,<Warmup Delay>,<RequiredSensors>

```
$SCA:0,95,1.0,0.75,0.50,0,0,0,3333,3333,0,0,550,6
0,129@ $RBT:@
```

Notes: Some versions of the Arduino IDE do not send the correct CR/LF termination – in these cases the '@' symbol (as shown above) may be placed at the end of a line to communicate end-of-command.

Example: Detailed configuration. Large alternator powered by large main engine, 1500AH industrial FLA battery:

- Acceptance @ 14.4v until acceptance current is less than 1% of capacity, 8hr max.
    - *8hr Time limit is set as fallback in case of battery current sensing failure*.

    13.2v float - revert back if 2% of capacity is removed

    15.3v Equalize, 3Hr duration.
- No overcharge nor postfloat phases.
- Allow alternator to operate up to 105c
- 30mV per degree C temperature comp

Command Syntax: (color-coded to make it easier to map values to their names)
$SCA: <reserved>, < Alt Target Temp >, <Alt Derate (norm) >,<Alt Derate (small) >, <Alt Derate (half) >, <PBF>, <Alt Amp Cap >, <System Watt Cap. >, < BAT_AMP_SHUNT_RAT>, < BAT_AMP_SHUNT_RATIO>, < REVERSED_BAT_SHUNT>, < REVERSED_ALT_SHUNT >, <Idle RPMs>,<Warmup Delay>,<RequiredSensors>

```
$SCA:0,105,1.0,0.75,0.50,0,0,0,3333,3333, 0, 0,550,30,0@
$CPA:7 14.4,480,5,0@
$CPO:7 0,0,0,0@
$CPF:7 13.2,-1,0,0,-10,12.7@
$CPP:7 0,0,0@
$CPE:7 15.3,0,180,0@
$CPB:7 0.030,15,-50,125@
$SCO:7,3,1,0@
$RBT:@
```

Example: Change warm-up delay to 30 seconds, leaving the rest of the configuration at the default values:

Command Syntax: (color-coded to make it easier to map values to their names)
$SCA: <reserved>, < Alt Target Temp >, <Alt Derate (norm) >,<Alt Derate (small) >, <Alt Derate (half) >, <PBF>, <Alt Amp Cap >, <System Watt Cap. >, < BAT_AMP_SHUNT_RATIO >, < BAT_AMP_SHUNT_RATIO>, < REVERSED_BAT_SHUNT>, < REVERSED_ALT_SHUNT >, <Idle RPMs>,<Warmup Delay>,<RequiredSensors>

```
$SCA:0,90,1.0,0.75,0.50,-1,0,0, 3333, 3333,0,0,0,30@
```

Note: Any values in the command syntax after the ones you intend to change, can be left out of the command. So here, we did not enter a value for *RequiredSensors*.

Note: As you can see, setting parameters in this way is very powerful, but quite tedious. Using the Dashboard tool gives you all this power without the tedium. Nonetheless, refer to this guide to understand the parameters in detail.

$SCT: Changes TACHOMETER parameters in System Configuration table

Update calibration ratios and parameters associated with alternator-driven tachometers. It should be noted the regulator will function correctly without changing any of these parameters; you need only change them if you wish to estimate the RPMs of your engine to be reported by the VSMMAR.

$SCT: <Alt Poles>, < Eng/Alt drive ratio >, <Tach Min Field>, <ForceTM>

Alt Poles: <WHOLE NUMBER (2 → 25)> Number of poles in the alternator.

Eng/Alt Drive Ratio: <FLOATING POINT NUMBER (0. 5 → 50)> Enter the ratio of your engine drive pulley diameter vs. the alternator drive diameter. Example, if your engine has a 7" drive pulley, and the alternator has a 2.6" drive pulley, then enter 2.6923 which is 7.0 / 2.3

Tach Min Field: <WHOLE NUMBER (-1 → 30)> This is the percentage value the PWM will be kept at as the minimum drive when the DIP switch has selected Tach Mode. (DIP switch 7) *Be very careful* with this value as it will set the floor at which the alternator is driven. If that floor is too high, it will prevent the regulator from 'regulating', which can result in burning out the battery. This is the actual PWM value sent to the field drive; though it is capped at 30% the full hardware PWM.

Set *Tach Min Field* = -1 to enable auto-determination. The VSMMAR will monitor the stator signal and when it becomes stable will use that PWM drive value as the floor. Alternatively, set this = 0 to disable any tach field drive *even if* the DIP switch is turned on.

If a *Tach Min Field* value is set (any value greater than 0), and Tach Mode is enabled, the regulator will use this value as a minimal field drive percentage, even during the warm-up period. You may have to experiment with this value to get one which matches your system, taking care not to make it too great as that could cause issues with overcharging of your battery.

If a fixed *Tech Min Field* value is set, the regulator will also hold in the warmup / idle phase until it is able to stably see RPMs, indicating the engine is running. This will prevent the field from being driven any harder in the case of the regulator is powered on via the Enable pin, but the actual engine is not running. Do take note though: if the stator wire is not connected (or has failed), and/or the *Tach Min Field %* drive value is too low, the regulator will remain in warm-up mode, 'appearing' to have failed when in fact it was been instructed to wait until it can see a stator signal.

ForceTM: <0, or 1> Setting the value to 1 will force on Tach Mode, independent of the Tach Mode DIP switch (DIP switch 7). 0 will cause the regulator follow the DIP-Switch. (DEFAULT = 0)

$SCT: will reply with “AOK;” if the command was processed successfully. However, to assure the changes are stored and used the regulator must be reset using the $RBT: command after all changes have been made.

$SCT: will not be recognized if system has been locked-out via the $SCO: command.

$SCO: Override features

Overrides the DIP Switches for charge profile (1,2,3) and battery capacity (4,5) selection. This command also allows the selection of auto detect for system voltage (12v, 24v), or forcing a fixed defined target system voltage.

$SCO: <CP_Index>, <BC_Index >, <SV_Override>, <Lockout>

CP Index: <WHOLE NUMBER (0 → 8)> Which charge profile entry should be used? (1..8). Set = 0 to use DIP switches for selection.

BC Index: < FLOATING POINT NUMBER (0.0 → 10.0)> Which battery capacity multiplier entry should be used against normalized 500Ah battery? (1..4). Set = 0.00 to restore selection to DIP Switch value.

SV Override: < FLOATING POINT NUMBER (0.0 → 4.0)> Enable (by setting = 0.0) or override the auto system voltage detection feature by defining the SV multiplier to be used. Though auto detect is a nice feature, being able to fix the system voltage can improve reliability and allow support for battery voltages which are not a whole number multiple of the ‘12v’ normalized battery used in the CPE tables. The following table shows some common values which may be used:

SV_Override value	Forced System Voltage	Charge Profile VOLTAGE Adjustment Factor
0	Auto	Auto
1	12v	1x
2	24v	2x

(Set SV Over-ride = 0 to restore auto-selection of 12-volt or 24-volt system voltages)

Lockout: <WHOLE NUMBER (0 → 2)> Security feature: Restricts ability to perform changes and/or provide input to the regulator which can impact how the alternator charges the battery. ***BE CAREFUL:*** Once lockout is enabled (value other then 0), it can ***ONLY*** be cleared by doing a hardware based master

reset (refer to Reference Manual), or re-flashing the firmware with changes to the EEPROM keys. No other command, not even $MSR: will be able to clear a non-zero lockout. ***Be especially careful with lockout=2, as the ONLY way to recover the regulator from this state will require special programming hardware probes.***

0 = No locking out.

1 = Prevent any configuration changes.

2 = Prevent any configuration changes – may NOT be cleared via Feature-in method.

DO NOT SET THE LOCKOUT UNLESS YOU HAVE A VERY GOOD REASON TO AND UNDERSTAND ALL OF THE IMPLICATIONS.

$SCO: will reply with "AOK;" if the command was processed successfully. However, to assure the changes are stored and used the regulator must be reset using the $RBT: command after all changes have been made.

$SCO: will no longer be recognized once it has been locked-out. See section "Restore to As-Compiled (default) Status"

Example: Configure to override DIP switches and positively define system voltage and battery capacity:

- Use CPE#3 (FLA#2 – large batteries)
- 1500AH battery (BC Index = 3)
- 12v system (SV Override = 1)
- Lockout NOT enabled (Allows continued changes)

```
$SCO:3,3,1,0@
$RBT:@
```

$SCR: RESTORES System Configuration table to default

Restores system configuration values to original as-compiled (default).

$SCR: will reply with "AOK;" if the command was processed successfully. However, to assure the changes are stored and used the regulator must be reset using the $RBT: command after all changes have been made.

$SCR: will not be recognized if system has been locked-out via the $SCO: command.

$MSR: RESTORE all parameters (to as defined at program compile time

Restores all configurable parameters to the as-compiled (default) values. This is a combination of the $SCR: , $CCR commands, the $CPR:n commands for all charge profile entry (CPE) tables, and $RBT; command. Plus, any calibrations done are cleared. It performs the same function as holding Feature-In

restore function (Refer to the Reference Manual *"RESTORE TO ASCOMPILED (DEFAULT) STATUS"*). Alternator will RESET after this command is completed.

$MSR:

Note $MSR is disabled if the regulator has been locked out via the $SCO command. In this case, you will need to do to do a full system reset via the Feature_in connector or re=flash the firmware. If successful, the regulator will reply with "AOK;" and then reboot.

$MSR: will not be recognized if system has been locked-out via the $SCO: command.

$EDB: Enable DeBug serial strings

Will cause regulator to start sending $DBG; strings via ASCII Serial USB communication port. This has the like effect to adding *"#define DEBUG*" to the source code, except $EDB: will ONLY enable the serial strings. Other debug features will not be changed. This command is also only effective for the time the regulator is running. If it is powered down, or reset (for example by a fault, or by receiving a command string that causes a reset), the regulator will restore to its default handling of $DBG: strings. See Appendix G for details on the output of the debug string returned in response to the $EDB: request.

$EDB:

$RBT: ReBooT system

Will cause regulator to reset. This is useful to load any changes from saved Flash memory into the regulator for its use.

$RBT:

$RBT: will not be recognized if system has been locked-out via the $SCO: command.

It is strongly suggested that you use the $RBT command after you have finished making changes to the alternator's configuration via other ASCII commands. This will restart the regulators, allowing those changes to

be recognized – but more importantly, some hardware needs the $RBT: command as a signal to actually save requested changes in non-volatile memory.

$FRM: Force Regulator Mode

This command (with its parameters) will force the regulator to change its current mode to the one indicated. Once forced into a mode the regulator will continue to manage the system accordingly, even if this means the regulator immediately exits the forced mode. For example, if you force the regulator into float mode, but the amps being taken from the battery exceed the exit_float criteria, the regulator will return to the bulk phase.

$FRM:<Mode>

Mode: <Character> The ASCII character ***immediately*** following the ':' will be used to force the alternator mode. Character must match EXACTLY the following (including case), must be IMMEDIATELY after the ':', but may be followed by any number of additional characters.

- B = Force into BULK mode.
- A = Force into ACCEPTANCE mode.
- O = Force into OVER-CHARGE mode.
- F = Force into FLOAT mode.
- P = Force into POST-FLOAT mode.
- E = Force into EQUALIZE mode.

Any other character will be ignored and no change will be made. If the active Charge Profile has 'disabled' a given phase, the regulator will immediately exit that phase – even if it is 'forced' into it, and switch to the next appropriate phase. Also note that if the exit criteria of a forced-mode phase is met, the regulator will again exit that phase quickly.
In such cases the mode may be changed before the next $AST string is sent.

Examples:

$FRM:B	→ Forces regulator into BULK mode
$FRM:Bulk	→ Forces regulator into BULK mode
$FRM:Bob@	→ Forces regulator into BULK mode (Note use of '@' as needed with Arduino IDE terminal)
$FRM:b	→ Ignored (lower case 'b')

No pre-existing condition check is made when receiving these mode change commands. For example, normally you would be able to enter equalize mode only if the regulator was already in float or post-float mode. However, the $FRM: command can force the regulator into equalize mode directly from any state, including bulk or ramping. (Do remember: as noted above - if conditions are such, it may not stay in equalize very long.)

APPENDIX D: DETAILS OF CPE (CHARGE PROFILE ENTRIES)

The following are excerpt from the CPE.H source code file to give more details on how each parameter impacts battery charging.

```
//----- This structure defines a 'profile' for battery charging.  Each stage consist of 'modes', primarily:  Bulk, Acceptance,
//      Overcharge, and Float.  Each mode has a max voltage set point, and criteria for exiting that phase (Exceeding a time limit,
//      or Amps dropping below a given value).   Of special note is the entry Float and Post Float, which have additional criteria
//      resuming charging.
//

#define MAX_CPES 8    //  There are 8 different Charge profile Entries
#define CUSTOM_CPES 2 //  The last two of which are set aside as 'customizable' and are changeable via the ASCII string commands.

// create human-readable names for charge profiles
// *** BE CERTAIN THESE CORRESPOND TO ANY CHANGES MADE IN CPE.cpp ***
enum ChargeProfiles {LIFELINE, STD_FLA, HD_FLA, AGM_2, GEL, FIREFLY, FOUR_STAGE_FLA, LIFEPO4};

typedef struct
{ // Charging Profile Structure
   const char *BATTERY_TYPE;    // String for the battery type
   float ACPT_BAT_V_SETPOINT;   // Set point for Ramp, Bulk and Acceptance battery voltage.
                                // Alternator will transition from BULK mode into Accept Mode when this voltage is reached,
                                // and then start the Accept Duration counter.

   uint32_t EXIT_ACPT_DURATION; // Stay in Accept mode no longer then duration in mS
                                //(Set = 0 to disable Acceptance phase and move directly to OC or Float mode)

   int EXIT_ACPT_AMPS;          // If Amps being delivered falls to this level or below, exit Accept mode and go to next
                                //      Set ExitAcptAmps =  0 to disable Amps based transition and only rely on EXIT_ACPT_DURATION
timeout.
                                //      Set ExitAcptAmps = -1 to disable Amps based transition and rely on EXIT_ACPT_DURATION timeout
                                //                           or ADPT_ACPT_TIME_FACTOR adaptive duration.
                                //      Set ExitAcptAmps = Same value used for LIMIT_OC_AMPS if Overcharge mode is to be used.
                                // Note:  If both Time and Amps are set = 0, Acceptance will be bypassed.
                                // FUTURE:  EXIT_ACPT_DVDT  Add dV/dt exit criteria for Acceptance mode, need to decide what it is :-)

   // Overcharge mode is sometimes used with AGM batteries and occurs between Acceptance and Float phase.
   int LIMIT_OC_AMPS;         // During Overcharge phase, Amps are capped at this low value.  (Set this = 0 to disable OC mode.)
   float EXIT_OC_VOLTS;       // Overcharge will continue until the battery voltage reaches this level.
   uint32_t EXIT_OC_DURATION; // Over Charge mode duration in mS.
                              // ( as a safety step, setting OC_VOLTS or DURATION = 0 will also disable OC mode..)
                              // FUTURE:  EXIT_OC_DVDT  Add dV/dt exit criteria for Overcharge mode, need to decide what it is :-)

   float FLOAT_BAT_V_SETPOINT;   // Set point for Float battery voltage, do not exceed this voltage.
   int LIMIT_FLOAT_AMPS;         // During Float, manage system to keep Amps into Battery at or under this value.
```

```
                                //   May = 0, set = -1 to disable limit.
   uint32_t EXIT_FLOAT_DURATION; // Alternator will stay in Float mode this int32_t (in mS) before entering Post-Float (no charging)
                                //   mode.  Set = 0UL disable transition to Post-float mode.
   int FLOAT_TO_BULK_AMPS;       // If Amps being delivered exceeds this value, we will assume a LARGE load has been placed on the
battery and we need to re-enter
                                // BULK phase.   Set this = 0 to disable re-entering BULK phase feature
   int FLOAT_TO_BULK_AHS;        // If the number of Ahs removed from the battery after 1st entering Float mode exceed this value, revert
back to BULK.
                                // Note this will ONLY be usable if the Amp shunt is at the battery.  Set = 0 to disable this feature.
   float FLOAT_TO_BULK_VOLTS;    // As with Amps, if the voltage drops below this threshold we will revert to Bulk.  Set = 0 to disable.

   uint32_t EXIT_PF_DURATION; // Only stay in Post_float mode (no charging) this amount of time.
                              //   Set = 0UL to disable times based Post-float exiting and exit only on Voltage.
   float PF_TO_BULK_VOLTS;    // If during Post-Float mode VBat drops below this voltage, re-enter FLOAT mode.
                              // Set = 0.0 to disable exiting of post-float mode based on voltage.
                              // Config note:  IF you configure the system to enter post-float mode from float-mode
                              //  (by setting a time value EXIT_FLOAT_DURATION), AND you
                              // set both EXIT_PT_DURATION and PF_TO_BULK_VOLTS = 0,
                              // the regulator will in effect turn off the alternator once charging is completed
                              //   and not restart a charge cycle until powered down and up again.
                              //   This can be useful if you truly want a one-time only charge.
                              //You could also config the FEATURE-OUT port to indicate the complete charge cycle has finished,
                              //  to say power-off the driving engine
   int PF_TO_BULK_AHS;        // If the number of Ahs removed from the battery after 1st entering Post Float mode exceed this value,
revert back to BULK.
                              // Note this will ONLY be usable if the Amp shunt is at the battery.
                              //   Set = 0 to disable this feature.

   float EQUAL_BAT_V_SETPOINT;   // If Equalize mode is selected, this is the target voltage.  Set = 0 to prevent user from entering
Equalization mode.
   int LIMIT_EQUAL_AMPS;         // During equalization, system will limit Amps to this value.
                                 //Set = 0 to disable amp limits during Equalization Mode.
   uint32_t EXIT_EQUAL_DURATION; // Regulator will not stay in Equalization any longer then this (in mS).  If set = 0, then Equalization
mode will be disabled.
   int EXIT_EQUAL_AMPS;          // If Amps fall below this value during Equalization while at VBat setpoint -- exit equalization.
                                 //  Set = 0 to disable exit by Amps and use only time.

   float BAT_TEMP_1C_COMP;  // Battery Temperature is compensated by this factor for every 1C temp change.
                            //   Note this is based off of BAT_TEMP_NOMINAL (25c)
   int MIN_TEMP_COMP_LIMIT; // If battery temperature falls below this value (in deg-c), limit temp compensation voltage
                            //   rise to prevent overvoltage in very very cold places.
   int BAT_MIN_CHARGE_TEMP; // If Battery is below this temp (in deg-c), stop charging and force into Float Mode to protect
                            //   it from under-temperature damage.
   int BAT_MAX_CHARGE_TEMP; // If Battery exceeds this temp (in deg-c),  stop charging and force into Float Mode to protect
                            //   it from over-temperature damage.
   uint8_t CPSPLACEHOLDER[8]; // Room for future expansion
} tCPS;
```

```
#define BAT_TEMP_NOMINAL 25    // Nominal temp which .BAT_TEMP_1C_COMP is based around (in deg-C).
#define PID_VOLTAGE_SENS 0.020 // When looking at charger mode transitions, if we come within 20mV of the target voltage (for rep 12v
battery), consider we have 'met' that voltage condtion.
                               //  (This is to help with smaller charging sources driving a LARGE battery, where the charging source
                               //    is not really able to push VBat slightly over the limit..)
```

Actual content of the CPE tables. Remember, all references are against a 'normalized' 12-volt / 500Ah battery.

const tCPS PROGMEM defaultCPS[MAX_CPES] = {

// eliminated decimal places to fix conversion narrowing problem. Note that, for example, 6 hours uses 60 here and 4.5 hours uses 45

// Name	BULK/ACCEPTANCE			OVERCHARGE			FLOAT			FLOAT TO BULK			POST-FLOAT TO BULK			EQUALIZATION				Min	Min Temp	Bat Min Chrg	Bat Max Chrg	
	Target	Exit Accpt		Limit	Exit	Exit		Limit	Exit							Equal	Limit	Exit	Exit	Temp	Temp	Chrg	Chrg	
	Volts	Duration	Amps	Amps	Volts	Duration	Volts	Amps	Duration	Amps	Ahrs	Volts	Duration	Volts	Ahrs	Volts	Amps	Duration	Amps	Comp	Limit	Temp	Temp	
{"LIFELINE"	14.3	60 * 360000UL	15	0	0	0 * 360000UL	13.3	-1	0 * 360000UL	-10	0	12.8	0 * 360000UL	0	0	0	0	0 * 360000UL	0	0.0234	-9	-45	45}	// LIFELINE BATTERY #1 Default (safe) profile & AGM #1 (Low Voltage AGM).
{"STD FLA"	14.8	30 * 360000UL	5	0	0	0 * 360000UL	13.5	-1	0 * 360000UL	-10	0	12.8	0 * 360000UL	0	0	0	0	0 * 360000UL	0	0.005 * 6	-9	-45	45}	// #2 Standard FLA (e.g. Starter Battery small storage)
{"HD FLA"	14.6	45 * 360000UL	5	0	0	0 * 360000UL	13.2	-1	0 * 360000UL	-10	0	12.8	0 * 360000UL	0	0	15.3	25	30 * 360000UL	0	0.005 * 6	-9	-45	45}	// #3 HD FLA (GC, L16, larger)
{"AGM #2"	14.7	45 * 360000UL	3	0	0	0 * 360000UL	13.4	-1	0 * 360000UL	-10	0	12.8	0 * 360000UL	0	0	0	0	0 * 360000UL	0	0.004 * 6	-9	-45	45}	// #4 AGM #2 (Higher Voltage AGM)
{"GEL"	14.1	60 * 360000UL	5	0	0	0 * 360000UL	13.5	-1	0 * 360000UL	-10	0	12.8	0 * 360000UL	0	0	0	0	0 * 360000UL	0	0.005 * 6	-9	-45	45}	// #5 GEL
{"FIREFLY"	14.4	60 * 360000UL	7	0	0	0 * 360000UL	13.4	-1	0 * 360000UL	-20	0	12.0	0 * 360000UL	0	0	14.4	0	30 * 360000UL	3	0.024	-20	-20	50}	// #6 Firefly (Carbon Foam)
{"CUSTOM "	14.4	60 * 360000UL	15	15	5.3	30 * 360000UL	13.1	-1	0 * 360000UL	-10	0	12.8	0 * 360000UL	0	0	15.3	25	30 * 360000UL	0	0.005 * 6	-9	-45	45}	// #7 4-stage HD FLA (& Custom #1 changeable profile)
{"LiFePO4 "	13.8	10 * 360000UL	15	0	0	0 * 360000UL	13.6	0	0 * 360000UL	0	-50	13.3	0 * 360000UL	0	0	0	0	0 * 360000UL	0	0.000 * 6	0	0	40}	// #8 LiFeP04 (& Custom #2 changeable profile

APPENDIX E: DEFAULT SYSTEM CONFIGURATION

The following documents default "system" values (as compiled) for the VSMMAR's system configuration. It is configured assuming both and alternator and battery shunts are installed. Note that a 300A / 75mV shunt is being used on the alternator and a 500A / 50mV shunt is being used on the battery. (The latter is the shunt used in the Link-10 battery meter as well as others).

Settings in System.cpp:

```
tSCS systemConfig = {
    false, // .REVERSED_BAT_SHUNT          --> Assume shunt is not reversed.
    false, // .REVERSED_ALT_SHUNT          --> Assume shunt is not reversed.
    90,    // .ALT_TEMP_SETPOINT           --> Default Alternator temp - 90c  (Approx 195f)
    1.00,  // .ALT_AMP_DERATE_NORMAL       --> Normal cap Alternator at 100% of demonstrated max Amp capability,
    0.75,  // .ALT_AMP_DERATE_SMALL_MODE   --> Unless user has selected Small Alt Mode via DIP switch, then do 75% of its
capability
    0.50,  // .ALT_AMP_DERATE_HALF_POWER   --> User has shorted out the Alternator Temp NTC probe, indicating they want
           //                                    to do 1/2 power mode.
    -1,    // .ALT_PULLBACK_FACTOR         --> Used to pull-back Field Drive as we move towards Idle.
    0,     // .ALT_IDLE_RPM                --> Used to pull-back Field Drive as we move towards idle.
    //                                         Set = 0 causes RPMs to be determined automatically during operation.
   //!! NOTE THAT I SET THIS TO 125 AMPS instead of 0 for my genset
    125,   // .ALT_AMPS_LIMIT --> The regulator may OPTIONALLY be configured to limit the size of the alternator output
    //                              Set = 0 to disable Amps capping.
    //                              Set = -1 to auto-size Alternator during Ramp. (required Shunt on Alt, not Bat)
    0,     // .ALT_WATTS_LIMIT  --> The regulator may OPTIONALLY be configured to limit the load placed on the engine via the
Alternator.
    //                                Set = 0 to disable, -1 to use auto-calc based on Alternator size. (Required Shunt on Alt)
    12,    // .ALTERNATOR_POLES            --> # of poles on alternator (Leece Neville 4800/4900 series are 12 pole alts)
    ((6.7 / 2.8) * 1.00),        // .ENGINE_ALT_DRIVE_RATIO      --> Engine pulley diameter / alternator diameter &  fine tuning
calibration ratio
    (int)((500 / 0.050) * 1.00), // .BAT_AMP_SHUNT_RATIO **Spec of amp shunt,  500A / 50mV shunt (Link10 default) and %
calibrating error
    //                                CAUTION:  Do NOT exceed 80mV on the AMP Shunt input
    (int)((300 / 0.075) * 1.00), // .ALT_AMP_SHUNT_RATIO  -->
    //                                DUBLER 061720 CHANGED TO 300A/75mV for alternator shunt
```

```
//                                CAUTION:  Do NOT exceed 80mV on the AMP Shunt input


-1, // .FIELD_TACH_PWM  --> If user has selected Tach Mode, use this for MIN Field PWM.
//                      Set = -1 to 'auto determine' the this value during RAMP phase
//                      Set =  0 to in effect 'disable' tach mode, independent of the DIP switch.
0, // .FORCED_TM   --> User can FORCE tach mode independent of DIP switch using $SCT command.
//                      0=DIP/off, 1=Force-on

0,   // .CP_INDEX_OVERRIDE  --> Use the DIP switch selected indexes
0.0, // .BC_MULT_OVERRIDE   --> Use the DIP switch selected multiplier
0.0, // .SV_OVERRIDE        --> Enable Auto System voltage detection
0,   // .CONFIG_LOCKOUT     --> No lockouts at this time.
#ifdef BENCHTEST
2,   // .ENGINE_WARMUP_DURATION  **Shortened for bench testing**
#else
15,  // .ENGINE_WARMUP_DURATION  --> Allow engine X seconds to start and 'warm up' before placing a load on it.
//                                  DUBLER MOD 080320 changed from 60
#endif
0};  // .REQURED_SENSORS  --> Force check and fault if some sensors are not present (eg alt temp sensor)
```

APPENDIX F: ERROR CODES AND MEANING

The following is a description of error codes as reported via the ASCII status and/or the LED blinking pattern. Most errors are hard-faults, indicating a condition which the VSMMAR is unable to decipher or correct itself and as such will shut down until corrected, in order to prevent any potential systems or battery damage. A few errors will attempt to auto-restart to see if the failing condition clears (for example, error low battery voltage).

These error codes may change and be expanded, refer to System.h for latest list. Many error codes are related to internal logic checks, if those are received look for a firmware upgrade. However, some errors codes occur during installation errors and / or system issues. A prime example is the Alternator overheating errors – which typically indicate a need to either increase cooling and/or enable SMALL-ALT-MODE or increase its pullback. These alternator overheating issues are very common when using small frame alternators (anything under 30lbs) combined with large battery banks or any Li-based battery bank.

FET over temperature (error #40) is an indication of a hardware issue with the regulator its self, or perhaps a short in the alternator field.

```
//----    Error codes. If there is a FAULTED status, the variable errorCode will contain one of these...
//              Note at this time, only one error code is retained.  Multi-faults will only show the last one in the checking tree.
//              Errors with + 0x8000 on them will cause the regulator to re-start, others will freeze the regulator.
//              (Note combinations like 10, and 11 are not used.  Because one cannot flash out 0's, and kind of hard to
//               tell if 11 is a 1+1, or a real slow 2+0)
#define FC_LOOP_BAT_TEMP               12             // Battery temp exceeded limit
#define FC_LOOP_BAT_VOLTS              13             // Battery Volts exceeded upper limit (measured via INA226)
#define FC_LOOP_BAT_LOWV               14  + 0x8000U  // Battery Volts exceeded lower limit, either damaged or sensing wire missing.
                                                      //   (or engine not started!)
#define FC_LOOP_ALT_RPMs               22             // Alternator seems to be spinning way to fast!
#define FC_LOOP_ALT_TEMP_RAMP          24             // Alternator temp reached / exceeded while ramping - this can NOT be right,
                                                      //   to reach target while ramping means way too risky.
#define FC_LOG_ALT_STATE               31             // Global Variable chargingState has some unsupported value in check_for_faults()
#define FC_LOG_ALT_STATE1              32             // Global Variable chargingState has some unsupported value in  manage_ALT()
#define FC_LOG_CPI_STATE               33             // Global Variable cpIndex has some unsupported value in caculate_ALT_Targets()
#define FC_LOG_CPINDEX                 34             // Global Variable cpIndex has some unsupported value in check_for_faults()
#define FC_LOG_SYSAMPMULT              35             // Global Variable systemAmpMult has some unsupported value in check_for_faults()
#define FC_LOG_BAT_STATE1              36             // Global Variable chargingState has some unsupported value in manage_BAT()

#define FC_SYS_FET_TEMP                41             // Internal Field FET temperature exceed limit.
#define FC_SYS_REQIRED_SENSOR          42             // A 'Required' sensor is missing, and we are configured to FAULT out.
#define FC_ADC_READ_ERROR              71             // Internal error - unable to use ADC subsystem

#define FC_BAT_INA226_READ_ERROR       100 + 0x8000U  // Returned I2C error code is added to this, see I2C lib for error codes.
#define FC_ALT_INA226_READ_ERROR       200 + 0x8000U  // Returned I2C error code is added to this, see I2C lib for error codes.
```

APPENDIX G: Debug String Contents

The $EDB: request will return the following data:

Timestamp in to the 1/1000 of a second
Charging State
fieldPWMvalue,

Feature-in port 1 state
Feature-in port 2 state
Feature-in port 3 state

PWMErrorV, PWMErrorA, PWMErrorW, PWMErrorAT, PWMError,

errorAT,

'B' followed by
measuredBatVolts to three decimal places
measuredBatAmps to one decimal place

'A' followed by
measuredAltVolts to three decimal places
measuredAltAmps to one decimal place

thresholdPWMvalue
fieldPWMLimit,
otPullbackFactor to two decimal places

measuredAltTemp, measuredBatTemp, in degrees C

the results of checkStampStack() which reflect how much of the stack has been used