WAKESPEED™

CHARGE SMARTER

COMMUNICATIONS AND CONFIGURATION GUIDE



Member

RV INDUSTRY ASSOCIATION PROUD MEMBER

NMEA

ABYC Boating Safety... Built In

TABLE OF CONTENTS

About this guide

The Wakespeed® Offshore series of product offerings features a common set of configuration and communications capability. This guide is used to document both Serial (ASCII) communications and CAN (Control Area Network) capabilities. Refer to the individual devices user's guide for additional details. Using the information in this guide one can access advanced configuration capabilities of the WS500 Alternator Regulator, as well as create a vertical stacked solution with tight integration between the regulator, BMS devices, Displays and engines. More so, this guide will be helpful in the development of supporting applications to aid in the configuration and use of the WS500 Alternator Regulator.

A note of edits and revisions

As this document is revised, new additions or capability will be placed in **RED** text. It is hoped this will aid in the quick assessment of new features. This release is intended for use with Firmware v2.5.1 or above.

2.5.1 Notes:

Release 2.5.1 primarily addresses an unintended impact of twin engine installs and non-Li based batteries, as well as introduced a requested feature.

- Error Notification #91 will now only be issued if the lost BMS connection was to a CAN device with 'Priority' equal to or above 120 (Suggested minimum priority for BMS / SOC device in RV-C)
- New Feature-Out modifier: Solid lamp activation only during FAULT vs. blinking out of fault number as well as suppression of engine startup lamp activation.
- Corrected documentation on MaxEngineLoading (CNG:) Floating-point number vs. integer (WS500 changed in 2.5.0 – but not correctly documented in prior release of this document)
- Added Required Sensor status indicator to SST: string.
- Extended SST; string adding additional indicators, e.g.: Half-Power, Feature-in status, White-Space active...
- Increased maximum Amp Shunt Ratio from 20,000 to 60,000 (See $SCA:)
- SAE - J1939 ACL Manufactured ID changed to 1305, Dragonfly Energy. Any devices checking the CAN ID need to recognize this change from TJC's ID (963)
- Support for MG Energy SmartLink Battery Controller.
- Added support for NMEA2000/J1939 ACL based 'Device Instance' field. Allows changing of the Charger Instance (See $CCN command) via NMEA2000/J1939 CAN message PGN 126208. (Example: Victron Cerbo adjusting 'Device Instance'). Value is capped at 12 (as reported by NMEA2000 devices) see: $CCN command for details.
- Corrected under-reporting of FLD % in AST and SCV (Min Tach Drive). Ala, 89% vs. 90%
- Corrected conflicts with remote sensing of Battery Current on system using Victron Lynx BMS + other Victron equipment (Skylla, some MPPT, etc).
- Added support for Victron Smart Shunts to supply remote Battery current and Temperature. ($CCN:)

A WORD OF CAUTION WITH TACH MODE AND WS500 FIRMWARE RELEASE 2.5.1+

Installs which use the Alternators Stator Wire to drive the engine Tachometers have a well know problem , Tach 'Dropping Out', particularly during the transition between Acceptance and Float stages. Many Alternator Regulators allows the user to define a Minimum Field Drive in order to allow some level of Alternator output at all times thereby keeping the tachometer active. The WS500 offers this capability via the $SCT: command. (See page 59).

However, there is risk associated with such an approach, and especially with Li battery installs of causing an over-charge situation and resulting BMS initiated disconnect. With this in mind, up to Firmware Release 2.5.0, if at any time the measured battery voltage or current was significantly above the present goal, the WS500 would set field output to 0%, even if the user had specified a *Tach Min Field %* as set in the $SCT: command. Safer for the system, but did cause the Tech to drop out at times.

Based on field feedback, beginning with Firmware release 2.5.1, the WS500 will respect the user defined *Tach Min Field %* value at all times, unless the WS500 is in Idle/Standby mode, or a faulted condition.

Users are cautioned to be aware of the significant change and assure their system performs not only as expected but in a safe way if they are using TachMode and Tach Min Field % settings.

ASCII COMMUNICATIONS

The WS500 Alternator Regulator utilizes ASCII communications as a machine-independent M2M (Machine to Machine) method for communicating between the WS500 and outside devices allowing for configuration, advanced monitoring / logging and diagnostics.

This manual is largely dedicated to documenting the ASCII protocol used by Wakespeed products; and though by their nature they are readable by Humans – and many interact with the Wakespeed Alternator Regulator directly via ASCII, it was primarily intended to communications to other tools – such as the Wakespeed application and 3rd party solutions (OPE Tether, OGSS Configuration tool, Ruby scripts, etc).

Wakespeed Application

One example of how these ASCII messages are used is with the Wakespeed Application. The applications allows for the creation of fully customized configurations based on selecting the alternator, battery, (optionally) BMS, as well as other system level features. Available for both iPhone and Android the application will create custom ASCII configuration files which can be used to configured the WS500. More so, when used with an Android one can use an OTG cable to allow for direct connection to the WS500 USB port allowing for sending the configuration to the WS500 as well as monitoring, logging and firmware update without the need for a Windows PC.

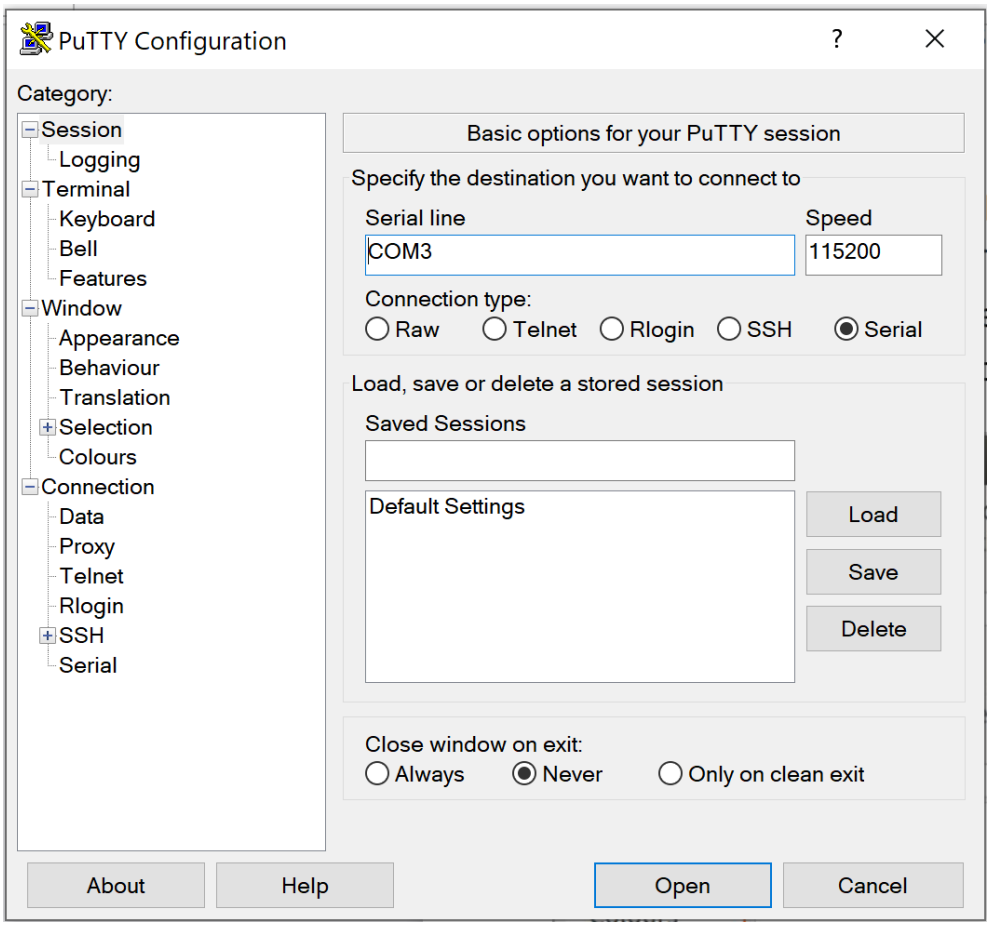
Hint: If you press and hold the 'Next' button at the bottom of the Configuration screens, a small raw ASCII window will open allowing you to see raw ASCII traffic from the regulator.

Terminal Programs

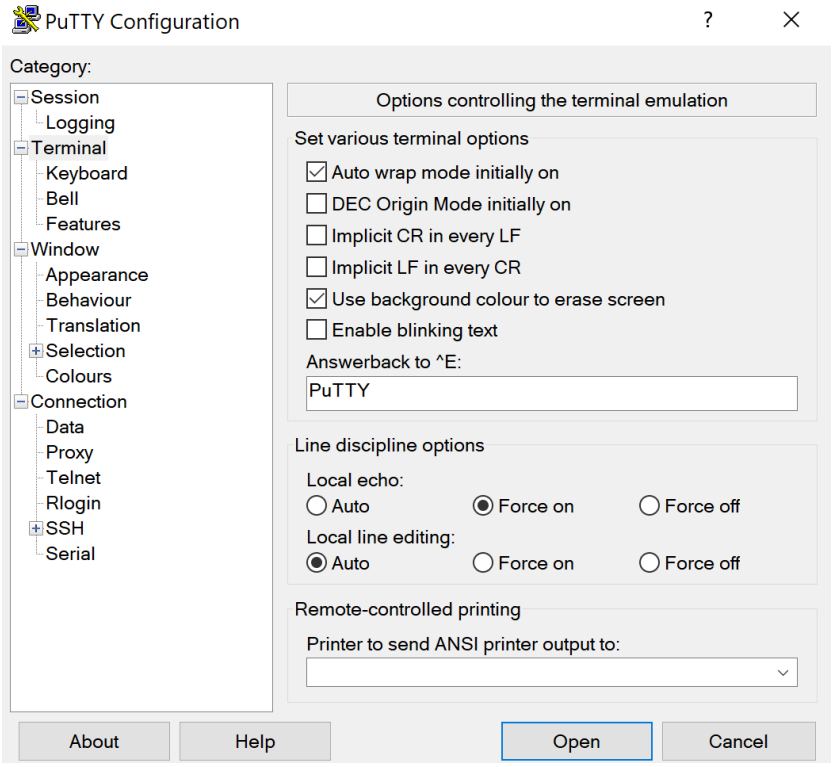
If one wishes, the raw ASCII data may be accessed directly using serial terminal programs and a computer. Many operating systems have built in terminal programs, or 3rd part programs such as PuTTY may be used to directly access the raw ASCII communication capability of the WS500 via its USB port. After connecting the regulator to the computer you can use one of these to open a communications window to the WS500 Alternator Regulator. A special note, some terminal programs do not send a complete end-of-line terminator (CR+LF). To support these environments, the WS500 Alternator Regulator will recognize the character '@' as an alternative EOL indicator.

Remember that any configuration changes you make to the regulator may not take effect until the regulator is restarted. When you have finished, make sure to issue the $RBT: command to not only assure changes are saved to the regulator's non-volatile memory, but that the changes are then utilized by the regulator. (refer to $RBT: - ReBooT system on page 78). After rebooting the WS500 Alternator Regulator verify the changes you sent were recognized by the regulator by inspecting the various status strings.

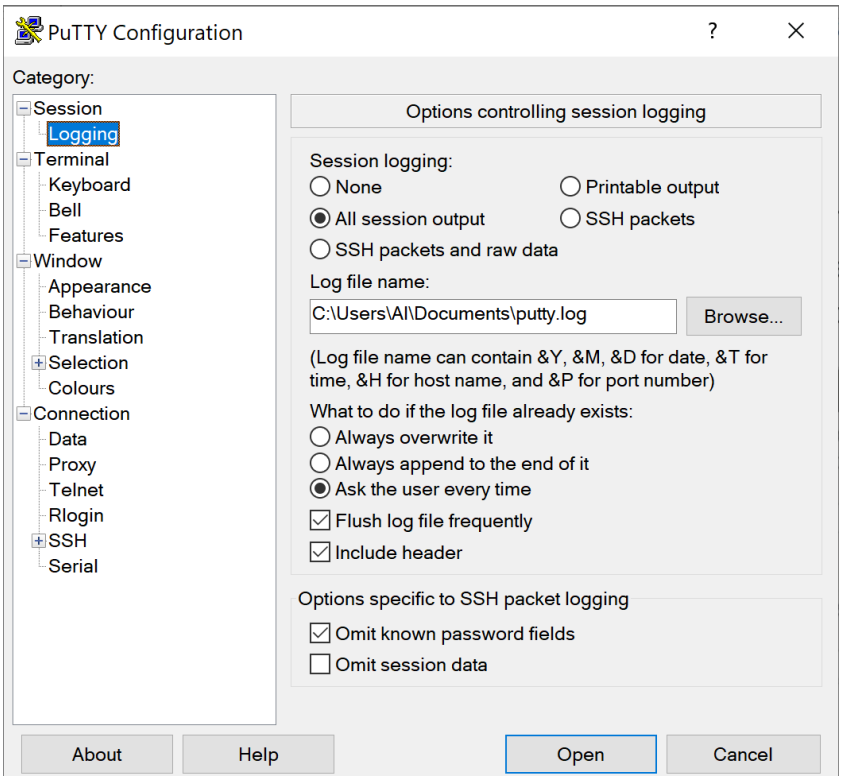
Putty: A versatile option is the free 'Putty' program (www.putty.org). It supports a wide range of OSs and includes a very nice logging function. To use connect up the USB cable and start Putty. Configure as shown here – selecting the Serial Line which your USB serial port is associated with, and setting the speed to 115200 and clicking the Serial Connection Type redial button.



Next click on the Terminal category and click the Forced-on button for local echo (as indicated on the following page), this will assure you can see what you are typing.



Finally, if you wish to keep a logfile of the session click on the Logging category, enter a file name and select the 'Printable Output' button as 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

Receiving data FROM the regulator: for details on the output.

Once you have done your configuration press the Open button to start the terminal session. A hint for Wondows users: If you have a need to Paste anything into Putty, know it does not use the normal CTRL-V command. Instead, position the cursor in the black display windows and click the RIGHT button, which will cause the clipboard to be pasted and sent to the WS500.

Bench-top Configuration:

When a USB cable is connected to the WS500 Alternator Regulator power is supplied to the logic portion of the hardware. This allows you to do bench-top configuration before completing the installation on the actual alternator. Make sure to do a $RBT: command as your last step. After the regulator reboots make sure to verify your changes before installing the regulator in a live installation.

CAN (Control Area Network) Communications

The WS500 Alternator Regulator features CAN (Control Area Network) ports. Developed in the 1980's by Bosch and targeted towards the transportation sector, CAN is now one of the most widely deployed communications standards covering not only the Transportation sector but also used in Industrial, Heavy Industry, Farming, Medical, Consumer, and more. Over a billon CAN nodes have been deployed, with modern automobiles contained upwards of 100+ individual nodes each! It is a proven reliable and robust communications standard with many feature to assure deterministic and prioritized communication resulting in a solid and proven reliable communications backbone.

Utilizing CAN devices are able to integrate into a 'System' where each works in cooperation with the others. Further, the CAN allow simple and reliable way to connect computers or displays for ongoing monitoring and easy configuration. The WS500 Alternator Regulator utilizes standards covering physical wiring, message content and other communications standards. These include:

- CAN Specification 2.0b / ISO-11898
- CiA 303
- SAE J1939
- RV-C
- NMEA-2000
- OSEnergy (Open Systems Energy - derived from the RV-C standard)

OSEnergy (Open Systems Energy) is an architectural specification who's aim is to provide a framework for the design, deployment, and operation of charging sources associated with a DC battery. Allowing them to work together in a 'systems' approach while meeting the full requirements of an associated battery as well as concurrently supplying house power needs in a consistent and efficient way. You can learn more here: https://github.com/OSEnergy/OSEnergy

Through the application of these standards the WS500 Alternator Regulator is able deliver several key benefits, including:

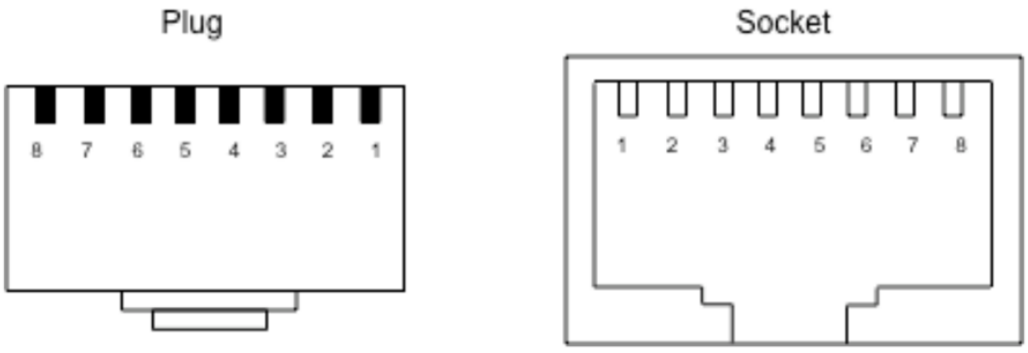
- Coordination of charging goals and objectives; all devices work towards the SAME goal vs. fighting each other.
- Tight BMS integration. CAN communications allows for the WS500 Alternator Regulator to fully integrate with the needs and directions of a BMS at levels unattainable using simple 'Charge Enable' wires.
- Prioritization of charging sources, e.g.: Utilization of Solar to its maximum capability while filling in the remaining energy needs from an engine driven alternator - thereby saving fuel.
- Remote sensing / Port Expander: The WS500 Alternator Regulator is able to take advantage of the CAN communications capability to transfer real-time battery status: voltage, amperage, and temperature as well as operational status (e.g., off-line in the case of a LiFeP04). By using this capability wiring and installations may be simplified.
- Self healing / fail over: Ability to self-recover from a failed, removed, or turned off device. The system continuously monitors all devices and adjusts as needed.
- 'Get-Home' total system failure mode: In the event of a catastrophic total system communications failures, the WS500 Alternator Regulator will fail-to-safe and operate in a stand-alone mode. Allowing for continued charging, but perhaps with less optimization and longer times needed.

CAN wiring

Use good quality CAT-5, CAT-5e, or CAT-6 cable to connect between devices in a daisy-chained fashion plugging into one of the two RJ45 connectors on the regulator. At *each end* of the daisy-chain install a terminator plug into the open RJ45 connector. It is important that the CAN bus be a single end-to-end chain with termination at each end. DO NOT connect an extra CAT-5 cable between the end devices making a loop – instead make sure each end point has one open RJ45 connector and then plug in the terminators.

The total length of the CAT-5 daisy-chain should be kept under 100M (300') with no more than 100x nodes total for best reliability.

The RJ45 connectors follow the CiA-303 standard, as shown here:



RJ45 connector

Pinning for RJ45 connector

Pin	Signal	Description
1	CAN_H	CAN_H bus line (dominant high)
2	CAN_L	CAN_L bus line (dominant low)
3	CAN_GND	Ground / 0 V / V-
4	(SFTY_STOP)	Optional - Machine Enable / Emergy Stop signal (Extension of CiA-303 spec)
5	-	Reserved
6	(CAN_SHLD)	Optional CAN Shield
7	(GND)	Optional ground
8	(CAN_V+)	Optional CAN external positive supply (dedicated for supply of transceiver and optocouplers, if galvanic isolation of the bus node applies) NOTE For recommended range of external power supply see clause 5.4

Figure 1 – CiA 303 CAN RJ45 connector specification

Connect the CAN_H and CAN_L signals. If supported, CAN_SHLD may also be optional connected as needed. It is generally NOT recommended to connect the CAN_GND to anything, as this may create a ground loop between it and ALT-

NMEA-2000® Support

The WS500 Advanced Alternator Regulator CAN protocol shares the same foundation and electrical specifications as NEMA2000 and produces NMEA2000 compliant messages. It needs to be noted the WS500 also by default transmits RV-C CAN messages, as well as J1939, which are not fully in line with the NMEA2000 specification. In most cases this does not caused problems when connecting the WS500 to a NMEA2000 network. However, if you do have issues with either NMEA2000 or OSEnergy communications you might try using a NMEA2000 certified CAN bridge such as the Maretron USB100 Gateway, or Yacht Devices YDNB-07 Bridge. In both cases it might be helpful to configure the bridges to filter out non-NMEA2000 messages (See Appendix C: Sample Yacht Devices CAN-CAN bridge scrip for a sample YDNB-07 script to accomplish this).

Wiring:

To connect into an existing NMEA-2000 network you will need to make up a patch cable. The simplest way to order a M12 Can adapter to connect the RV-C compliant CAN connection to a NMEA2000 backbone. Another approach would be to cut one end off a common CAT-5 cable and use a field attach NMEA2000 connector. Referring to "Figure 1 – CiA 303 CAN RJ45 connector specification" above as well as "Figure 2 - NMEA2000 connector pinout" below only the *CAN_H* and *CAN-_L* wires need to be connected. With either approach take care not to exceed the maximum Drop-cable length of 6 meters, take care also to properly terminate the CAN network. For reference here is a wiring guide:

Signal	CAT-5 Cable			NMEA2000 Cable	
	POSITION	COLOR		POSITION	COLOR
CAN_H	1	White/Green --OR-- White/Orange		4	White
CAN_L	2	Green --OR-- Orange		5	Blue

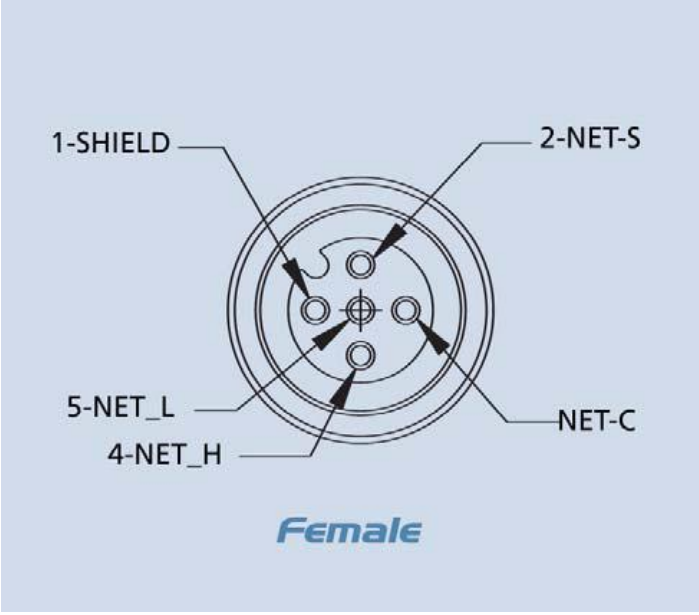


Figure 2 - NMEA2000 connector pinout

There is no need to connect the CAN-GND, and in fact doing so may cause reliability issues due to ground-loops.

A number of NMEA2000™ status output messages are supported. These messages may be useful when the WS500 is connected to a NMEA2000 network and will allow the operational status of the alternator and battery to be displayed. Refer to Appendix B: CAN messages for a list of supported messages. Note that the WS500 will send two groups of some of the NMEA2000 messages, one set for the Battery and one set for the Regulators operation. Refer to the details of each PGN.

Be sure to properly configure the regulator when using these messages, specifically the `Engine ID' and 'Charger Instance' using the $CCN: command, and note also that the Battery Instance number transmitted in NMEA2000 messages is reduced by 1 from the WS500 Battery Instance (NMEA2000 uses 0 oriented numbering, while RV-C uses 1 oriented numbering).

A NOTE ON FEATURE-IN PORT

The Feature-in port allows a wide range of optional capabilities in the WS500 to be selected in real time. Throughout this configuration guide there are a number of places where the Feature-In port may be enabled to be looked at to select a given capability. There are also options to modify the behavior of the Feature-In port from its normal behavior. Care should be exercised when configuring the WS500 to assure that the desired behavior of the Feature-in port is selected, and that some other expected use has not inadvertently been disabled. Example, by utilizing the Feature-In port to allow to selection of an alternative DC-DC converter set point, the default behaviors of forcing the regulator mode to Float with CPE #8 will be overridden; this could cause unexpected behaviors if the system design expects to use the Feature-in port for a legacy BMS integration.

Feature-In can be used as

1. Force Equalize (when using CPE #7)
2. Force Float (when using CPE #8)
3. Force whitespace (setting RMP to positive value in CNG when in CPE #8. Negative value of RPM in CNG forces whitespace on all the time ignoring Feature-In. Pos/New values of RPM in CNG are ignored when in CPE 7.)
4. Force regulator power (setting Half-power to 0% in SCA)
5. Feature-In polarity change for any of the above.

A NOTE ON HIGH ENERGY SYSTEM INSTALLS

The WS500 is unique in its ability to operate with a large range of system voltages duo in part to the use of run-time voltage scalars (See $SCO command). This, combined with its other wide capabilities makes it a very flexible product for use when designing and deploying 'High Energy' DC systems. However, such systems require great care in their overall design and deployment. It is incumbent upon the user of the WS500 to make sure they fully understand all aspects of their system, how components are installed and interact. The reader is encouraged to seek out Industry Standards for reliability, as well as industry Best Practices from reliable sources. How the system behaves with multiple charging sources, how extreme stress conditions are handled (both prevention, and reaction), all are needed to be well thought through. Wakespeed provides on our website a wide range of BMS and Battery systems which we have proofed in the lab – integration between the BMS and the WS500 for example; but such documents cannot cover all the considerations needed for more rich system installs which go well beyond the simple alternator/battery combo.

The following are some other considerations that might be helpful, especially with regards to 48v deployments. One is cautioned though, these are only some hints, it is fully up to the system design to assure a well functioning system in all cases and points of operations – including warnings and fault conditions. Significant damage can occur without careful though and design of the system.

48V VS. 52V VS. 56V

In the field there are a number of different batteries that are used for '48v' deployments. Some are 15 cell LiFeP04 which at times sold as 48v batteries while 16 cell ones are sold as '52v' ones. Others use a different chemistry and are marketed at a higher 'nominal' voltage – perhaps showing 56v. The WS500 is able to accommodate all these variations, and for this note the term '48v' should be construed as applying to all as well.

THE WS500 AND FIELD DRIVE

At 12v and 234v voltage it is common for the Field to be specified to operate at the same voltage as the connected battery. 12v for a 12v battery, 24v field for a 24v battery, etc. This is not always the case with 48v alternators – in many (but not all) alternators, the field is actually specified at 12v. In such cases there are a couple of ways the WS500 may be deployed with a hybrid alternator (48v output, 12v field)

1) Connect the ALT+ (Red) wire to a 12v source

> With this approach, the field power source is supplier from a 12v source by connect ALT+ (Red wire) to a 12v (perhaps chassis) battery. VBAT+ (Red/Yellow wire) still needs to be connected to the battery for proper voltage sensing, btu the WS500 does not require that both the target battery and the power source for the field be the same, not the same voltage.
>
> Such an mix of voltages is a bit more complex with regards to wiring, but the WS500 functions well in the type of design. Do note that some harness (ala, the Van hardness) combine ALT+ and VBAT+ into a single wire to simplify installs. In this case such a deployment will not work.

2) Apply a derate values ($SCA command)

The $SCA command has three 'Derate' values which can be defined: Normal, Small Alt, and Half-power. These are typically used to reduce the output of an alternator to account for system cooling concerns (ala, alternator overheating), and/or to reduce the load on the driving engine. But they can also be used to in effect reduce the 'Field Voltage' from 48v to allow direct driving of a 12v field while the ALT+ wire is attached to the 48v battery. To so this simply start with a 'Normal' derate value of 0.25 (aka, 25%), and that will reduce the average field voltage to an acceptable operation range. Most 48v deployments use this technique.

Change in Default Operation of the WS500

Due to the prevalence of 12v fields in 48v alternators, beginning with version 2.5.0 of the firmware, the WS500 will by *DEFAULT* automatically apply the 25% derate values if those have not been explicitly defined by a $SCO command: In other words, if you take a factory fresh WS500 (or one that has been issued a $MSR command to restore to Factory Fresh condition) and deploy it in a 48v system, field drive will be capped at 25%. If your alternator has a true 48v field (some do), you will want to explicitly issue a $SCO command to restore the 100% field drive. (Or max field drive is otherwise appropriate)

Wakespeed has taken this step as the large majority of 48v alternators seem to have 12v fields, so we are adjusting the DEFAULT behavior to accommodate them. As always the regulator may be configured to better match your actual system, with the $SCO command in this case.

Surge Protection (aka, the dreaded Load-Dump)

How to prevent and protect the system for a high energy voltage spikes. This is a very critical topic when designing a Lithium based battery system, with even more care needed in a 48v system. First a bit of background.

Alternators are by design electro-mechanical devices (vs. a pure electron device such as AC and Solar chargers) and as such there is a significant lag time for alternators to respond to changes in field drive. During normal operation the control look of the Alternator, Battery, and WS500 take into account this nature, but if that loop is broken, a new situation needs to be addressed.

With the increased use of Li based batteries, and their accompanying BMS – the situation where the BMS goes into a protection mode and disconnects the Charge Bus of the battery creates what is known in the industry as a Load-Dump. SAE /ISO define these as a high energy even with voltages approaching a spike of 120v – in a 12v system. Double that for a 24v system. Testing in the Wakespeed lab has indicated that uncontrolled disconnects in a 48v system can, and often do, result in voltage spikes as high as 380-390v.

Clearly such high voltage spikes can (and have shown to) cause significant damage if not contained.

The Transportation sector (think cars, trucks, etc) long ago settled on a Best Practice of suppression for 12v/24v system through the use of special surge suppressing diodes know as Avalanche Diodes as the way to address these high voltage events. Relevant specifications one might reference include ISO 16750-2

By using Avalanche Diodes in the alternators rectifier block (vs. lower cost standard diodes) the system is protected from these events as the Avalanche Diodes suppress any voltage spike to perhaps 30v (in a 12v system) which are non-damaging to other devices. Most quality 12v and 24v alternators come with Avalanche Diodes and Wakespeed strongly recommends using such alternators, as they are a very critical part of an overall safe system design.

The special consideration in 48v systems is that Avalanche Diode Technology is not available in 48v alternators – the transportation sector instead focuses on well designed system which prevent load-dumpy through well designed systems. (Ref ISO 21780) As a result, 48v system designs *must* assure prevention of uncontrolled disconnect events (aka, load dump).

As such careful system design is critical for safe deployments in a 48v environment.

This is most critical during a disconnect even, there must be sufficient time to safely shut down an alternator before the contactors/FETs open, else what is known as a Load-Dump situation is crated that results in system damaging voltage spikes (upwards of 400v for a 48v deployment). It is perhaps the single largest limitation of which batteries we (Wakespeed) can quality as providing a safe system with couples with the WS500 (or any alternator based charging solution for that matter). What we need to do is receive advanced notice (Via CAN or some other communications method) of a pending disconnect before the actual disconnect. Though alternates take perhaps 400-500mS to fully shut down, we design our systems around a 2-second (min) forewarning.

RECEIVING DATA FROM THE REGULATOR:

 = Features supported by Alternator Regulators (e.g., WS500)

 = Features supported by DCDC Converters (e.g., WS3000)

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

Formats are in clear ASCII 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 with a CR/LF termination.

Additional details of each status 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: Derived Battery Amps being used to decide charge modes.

Voltage and current readings made by the WS500 Alternator Regulator are directly reported as *AltVolts* and *AltAmps*. Unless overridden by an external source (example via a Remote Battery Sensor) those same values will be assumed to be *BatVolts* and *BatAmps* and used by the regulator to make charge state decisions.

SystemWatts: Current measured System Watts being delivered.

TargetVolts: Volts the regulator is attempting to bring the battery to. Either from the internal Charge Profile entry, or if the regulator is locked onto a BMS, from the goals being send to it.

TargetAmps*: Battery limit amperage the regulator will limit current to.

TargetWatts*: Watts the regulator is actually working to limit the system to.

AltState: Current state of the Alternator, per the following table:

Value	
0,1,4	- Alternator Off
2,3	- Alternator FAULTED (See Fault Code)
5	- In special DC-Disconnected CV mode.
6	- Alternator in Idle (Ala, Half-power Mode with active DCDC converter)
9	- In CONFIGURATION mode
10	- Alternator Standby mode, or in delay mode while engine warms up.
11,15	- Ramping towards BULK mode.
12,20	- In BULK mode
21	- In ACCEPTANCE mode
22	- In OVER CHARGE mode
30	- In FLOAT mode
31	- In FORCED_FLOAT mode (via Feature_in pin and CPE = #8, or missing required sensor)
36	- In OFF (Post Float) mode
38	- In EQUALIZE mode

39	- In CVCC mode (only available in system under direction of CAN master)

BTemp: Measured temperature of NTC sensor attached to B-port in degrees C or battery temperature received via external CAN sensor. -99 indicates temperature has not been measured, NTC sender has failed, not attached, and there is no remote temperature information available via the CAN connection.

ATemp: Measured temperature of NTC sensor attached to A-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 ½ power mode)

RPMs: Measured RPMs of engine (Derived from Alternator RPMs and the Engine/Alternator drive ratio)

(The following additions are available with Firmware version 1.0.0 and above)

AltVolts: Measured Alternator Volts, in volts and fractions of volts (to 1mV resolution)

FTemp: If equipped, this is the temperate of the FETs in degrees C.
-99 indicated FET temperate cannot be measured.

FAmps: If equipped, this is a measurement of the current (amperage) being delivered to the field.
-99 indicated field current is not being measured.

FLD %: % (0..100%) field is being driven.

*Note: * If the WS500 Alternator Regulator is configured with no limits for Battery Amps and/or System Watts a self-impose limits of 1,000A / 15,000W as max values. AST; will report these working values.*

DST; -- DC-DC ENERGY TRANSFER STATUS 

"DST;, DCDC_State, Volts, ,SecondBatVolts, SecondBatCurrent, PrimBatCurrent, DCDC_Temp "

DCDC_State: Operational status of optional DC-DC converter. See AltState table above for additional details.

Value	
0	- Converter Off (or not present)
1,4,6	- Converter in Standby or Idle mode
2,3	- FAULTED (See Fault Code below)
10..40	- Charge Modes: (Reference AST Table above)
71	- 2nd Battery augment mode.

Volts: Presently active target Voltage goal/trigger. (See $CDD on page 72 for details)

SecondBatVolts: Measured Secondary battery voltage as reported by the DC-DC converter. Note this may be remotely senses battery voltage, or it may simply be the voltage at the DC DC Converters Secondary Battery terminals.

SecondBatCurrent: If DC-DC Converter is capable, this is the measured current of the secondary battery.

PrimBatCurrent: If DC-DC Converter is capable, this is the measured current of the primary battery.

Note: In both the Primary and Secondary battery current, the values reflect current supplied/consumed by the DC-DC Converter only, not the entire battery. A Positive value indicated energy transfer out of the DC-DC Converter and INTO the respective battery

DCDC_Temp: Internal temperature of the DC_DC Converter, in degrees C

Note: DST: will only be sent if the optional DC-DC converter has been configured and enabled (See $CDD: command on page 72). If enabled, it will alternated with the AST status string.

DCV; -- DC-DC ENERGY TRANSFER CONFIGURATION

"DCV;, Model, Mode, , Volts, Volts_HalfPower, ,Co-Charge Limit Amps, ,Aug Limit Amps, Aug Limit Volts, Aug Limit SOC"

Model: Model of DC-DC Converter configured to support (See $CDD: command)

Mode: Operational mode of optional DC-DC converter bridging the Primary (Target, or High Side) and Secondary (Low Side, e.g.: Chassis) battery.

Voltage: 2nd battery Target voltage, above this charge assist mode is enabled.

Volts HalfPower: 2nd battery Target voltage when regulator is in Half-power mode

Co-Charge Limit Amps: Max current while in primary battery Charge Assist mode.

Aug Limit Amps: Max current while in 2nd battery augmentation mode.

Aug Limit Volts: Primary battery voltage cutoff, below this 2nd battery augmentation mode is disabled.

Aug Limit SOC: Primary battery SOC cutoff, below this 2nd battery augmentation mode is disabled.

Note: DCV: will only be sent if the optional DC-DC converter has been configured, ala the Model is not set = None. (See $CDD: command on page 72)

ENG; -- ENGINE CONFIGURATION

"ENG;, J1939MaxLoad, ,RFM_RPM, RFM1, RFM2, RFM3, RFM4, RFM5, RFM6, RFM7, RFM8”

J1939MaxLoad: Maximum Engine Load allowed as reported via J1939

RFM_RPM: Maximum expected engine RPMS for tachometer based engine loading management

RFM(n): Max field drive % modifier to be applied at each RPM bucket from 0..RFM_RPM

Reference $CNG: command on page 37

CST; , -- CAN STATUS 

"CST;, BatteryID, IDOverride, Instance, Priority, ,Enable NMEA2000?, Enable OSE?, ,AllowRBM?,IsRBM, ShuntAtBat?, ,RBM ID, IgnoringRBM?,Enable_ALT_CAN, ,CAN_ID, ,EngineID, BitRate, Aggregate BMS"

BatteryID: Battery number (or Instance) the regulator is associated with. 1..100
The following 'convention' is suggested – but not required:

1. Main House Battery
2. Primary Engine Starter battery (port engine)
3. Secondary House Battery
4. Secondary Engine Starter battery (starboard engine)
5. Generator Starter Battery
6. Forward Thruster battery
7. Aft Thruster Battery

IDOverride: Battery number (or Instance) is set via the DIP switches, however it is possible to 'override' the DIP switches using the $CCN: command. (0= no override)

Instance: Charger Instance (1..13). Set with $CCN: command (Default = 1)

Priority: Device priority, used to decide which devices should provide charging current, as well as who will be potential 'master' device. Set with $CCN: command (Default = 70)

Enable NMEA2000?: 0 or 1, Is regulator configured to send NMEA-2000 messages? (1 = Yes)

Enable OSE?: 0 or 1, Is regulator configured to send OSEnergy type messages? (1 = Yes)

By using the $CCN: command, the user may disable portions of the CAN message stack. One would do this in cases where conflicts exist with existing devices on a shared CAN bus. An example might be the regulator is installed into an existing NMEA-2000 system, and it is desired to have NMEA-2000-like status be sent out; however some of the OSE messages cause issues with existing NMEA-2000 instruments. In this case the user may choose to disable OSE messages.

CAUTION: If OSE messages are disabled, all CAN-based value-add capabilities of the regulator will also be disabled. Including remote instrumentation, common charging goal, and charging device prioritization. *DISABLE OSEnergy MESSAGING WITH CAREFUL CONSIDERATION* and perhaps consider setting up isolated networks instead with a CAN bridge to forward the NEMA2000 messages to the proper NMEA2000 bus.

AllowRBM?: 0..2: Is regulator configured to attempt to act as the Remote Battery Master? (1 = RBM, 2 = 'Virtual RBM')

IsRBM?: 0..2: Does regulator currently think it is the Remote Battery Master? (1 = RBM, 2 = 'Virtual RBM')

ShuntAtBat?: 0 or 1, Does regulator currently think its shunt is directly connected to the battery? (1 = Yes, 0 = No)

The WS500 Alternator Regulator is able to assume the role of the Remote Battery Master, thereby acting as the central coordinator for all charging sources. In practice, using the regulator as the RBM typically would occur only with small installations, twin engines installations are a common example. However, one is also able to configure a more extensive system where the WS500 Alternator Regulator is configured as a backup device. Set this via the $CCN: command

RBM ID: Remote Battery Master ID: ID number of remote device which is currently recognized as the Remote Battery Master. 0 = WS500 Alternator Regulator has not associated itself with RBM.

IgnoringRBM?: 0 or 1, Is the regulator ignoring the Remote Battery Master? (1 = Yes)
If the Remote Battery Master sends information which seems unbelievable, this flag will be set and the regulator will ignore it. ***Such a condition indicates something is wrong in the overall system and that should be investigated and resolved.*** Conditions which will cause this fault include:

- Indicated Battery Voltage too high, or too low. (8..18v for normalized 12v battery)
- Indicated Battery Current too high (> +/- 2,000A)
- Voltage difference between battery and alternators > 1.5v (indicating issue with alternator wiring)
-

Enable_ALT_CAN: Bit-mask enabling alternative CAN based remote sensing and/or RBMs. See $CCN: on page 67 for details.

CAN_ID: This is the current CAN Node ID, or node address which the WS500 Alternator Regulator has been assigned.

EngineID: Engine ID (or number) the WS500 Alternator Regulator with the engine it is mounted on. Used for monitoring J1939 RPMs messages as well as sending NMEA2000 RPMs back out. Note by convention EngineID = 0 is the default value.

BitRate Data communication rate CAN is set for.

Aggregate BMS: Indicates optional aggregation of BMS / RBM's.

CPE; -- CHARGE PROFILE ENTRY

In response to RCP: command, this displays the current values of a Charge Profile Entry. Special note on Charge Profile Entries: All voltage and current values in Charge Profile tables are displayed in their normalized '12v' values. See Defining Charging Voltages and Amps for additional information.

"CPE;, n, acptVBAT, acptTIME, acptEXIT, res1, , ocAMPS, ocTIME, ocVBAT, ocAEXIT, , floatVBAT, floatAMPS, floatTIME, floatRESUMEA, floatRESUMEAH , floatRESUMEV, ,pfTIME, pfRESUME, pfRESUMEAH, , equalVBAT, equalAMPS, equalTIME, equalEXIT, , BatComp, CompMin, MinCharge, MaxCharge, ,RdcVolts, RdcLowTemp, RdcHighTemp, RdcAmps, ,floatSOC, ,MaxAmps, ,pfVBAT, MaxBatVolts"

n: Charge Profile 'n' is being displayed/returned (1..8)

acptVBAT: Target battery voltage during BULK and ACCEPT phase

acptTIME: Time limit to stay in ACCEPT mode – in Minutes.

acptEXIT: Amp limit to trigger exiting ACCEPT mode

res1: Reserved for future use (dV/dT exit criteria, currently = 0, disabled)

ocAMPS: Max Amps which will be supplied by during OVERCHARGE mode.

ocTIME: Time limit to stay in OVERCHARGE mode – in Minutes.

ocVBAT: Target battery voltage during OVERCHARGE phase

ocAEXIT: Amp limit to trigger exiting OVERCHARGE mode

floatVBAT: Target battery voltage during FLOAT phase

floatAMPS: Max Amps which will be supplied by during FLOAT mode.

floatTIME: Time limit to stay in FLOAT mode – in Minutes.

floatRESUMEA: Amp limit to trigger resumption of BULK charge mode

floatRESUMEAH: Amp Hours withdrawn after entering Float to trigger resumption of BULK charge mode

floatRESUMEV: Volt limit to trigger resumption of BULK charge mode

Note: If the regulator is in FORCED_FLOAT mode via the FEATURE-IN pin, then none of the above checks to exit float mode (e.g., floatTIME) will be performed. However, regulation will still occur to *floatVBAT* and *floatAMPS*.

pfTIME: Time limit to stay in POSTFLOAT mode – in Minutes, before resuming FLOAT charge mode.

pfRESUME: Battery Voltage that will trigger resumption of FLOAT charge mode

pfRESUMEAH: Amp Hours withdrawn after entering Post Float to trigger resumption directly to BULK charge mode

equalVBAT: Target battery voltage during EQUALIZE phase

equalAMPS: Current limit of Alternator while in EQUALIZE mode

equalTIME: Time limit to stay in EQUALIZE mode – in Minutes.

equalEXIT: Amp limit to trigger exiting EQUALIZE mode

BatComp: Temperature Compensations value per 1-degree C (normalized to '12v' battery)

CompMin: Minimum temperate to apply compensation at. In degrees C

MinCharge: Minimum temperate to charge the battery at, below this will force into a standby mode.

MaxCharge: Maximum temperate to charge the battery at, above this will force into a standby mode.
Note: Prior editions INCORRECTLY indicated regulator would enter FLOAT mode..

RdcVolts: Battery low Volts trigger for Reduced Charging

RdcLowTemp: Battery low Temperature trigger for Reduced Charging.

RdcHighTemp: Battery high Temperature trigger for Reduced Charging.

RdcAmps: Current limit while in Reduced Charging state.

flaotSOC: SOC level to trigger resumption of BULK charge mode .

MaxAmps: Maximum battery charge current during any charge phase.

pfVBAT: Target battery voltage during POST-FLOAT phase

MaxBatVolts: Optional max battery voltage fault threshold

NPC; -- NAME & PASSWORD CONFIGURATION

"NPC;, <reserved>, Name, Password, , DeviceID"

<reserved>	Set = 1
Name:	Name of Regulator (Used for CAN device ID) (ASCII up to 18 characters)
Password:	Password (ASCII up to 18 characters). Note that if the password has a leading dot (.) character, it is 'hidden' and "****" will be returned for its value.
DeviceID	Semi-Unique device ID of WS500 Alternator Regulator

SCV; -- SYSTEM CONFIGURATION 

"SCV;, Lockout, *BTS2ATS?*, RevAmp, SvOvr, BcOvr, CpOvr, ,AltTempSet, drtNORM, drtSMALL, drtHALF, PBF, ,Amp Limit, Watt Limit, , Alt Poles, Drive Ratio, Shunt Ratio, ,IdleRPM, TachMinField, Warmup Delay, Required Sensor, DC_Disconnected_VBat, FeatureIN_mod, TriggerHalfPowerRPM, Ignore Sensor, Feature-OUT mod, ,BMS Amp Cap, Promiscuous Mode"

Lockout: Current lockout level. (0..2), see $SCO: command.

BTS2ATS?: 0 or 1: Is BTS being used as 2nd Alternator Temp Sensor? (1 = yes)

RevAmp: 0 or 1: Reverse polarity of Amp Shunt readings? (1 = yes)

SvOvr: System Voltage auto-detect (=0), or force (1.0x .. 4.0x → 12v .. 48v)

BcOvr: Override Battery Capacity DIP switches (Dip 5/6). (0.00 = No)

CpOvr: Override Charge Profile DIP Switches (Dip 2..4) (0 = No)

AltTempSet: Target max running setpoint for Alternator, in degrees C

drtNORM: Normal Amp reduction (de-rating) fraction

drtSMALL: Amp reduction (de-rating) fraction when in SMALL - MODE

drtHALF: Amp reduction (de-rating) fraction when in half-power mode.

PBF: Pull-back factor, for reducing Field Drive at lower RPMs.

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

Watt Limit: Defined System size, or -1 to enable auto-sizing. Set this = 0 for installations where Watts loading is not to be regulated (ala, battery focused installations).
Note: During startup, and unless defined, this value will present: 15,000

Alt Poles: Number of poles on Alternator

Drive Ratio: Ratio of engine and alternator drive pulley

Shunt Ratio: Amp Shunt ratio in Amps / mV

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

TachMinField: Minimum % of field drive that will be applied if TACH MODE is enabled.

Warmup Delay: Number of seconds after power on before entering RAMP mode.

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

DC_ Disconnected _VBat: Indicated how charging device should behave when it received a DC-Disconnect message via the CAN.

FeatureIN_mod: Is the behavior of Feature-In being modified from default?

TriggerHalfPowerRPM: RPM value under which Half-power mode will be selected.

Ignore Sensor: Which sensors should be ignored? (Prevents unwanted noise from causes false readings)

FeatureOUT_mod: Is the behavior of Feature-Out port being modified from default?

BMS Amp Cap: Has a BMS Amperage Limit been defined?

Promiscuous Mode: Has the device been configured to auto-restart on most hard faults?

SST; -- SYSTEM STATUS

"SST;, Version , ,Derate Mode, System Options, , CP index, BC Mult, SysVolts, ,AltCap, CapRPMs, , Ahs, Whs, ,ForcedTM?, RequredSensorFlag"

Version: Firmware revision identifier. Will have format of "AREG" followed w/o a space by the version number. E.g., "AREG1.0.1"

Derate Mode: The following table indicates which of the Derate Factors are being used at the moment:

Value	Derate Mode
0	Derate Normal
1	Derate Small Alt Mode
2	Derate Half Power

Note that prior to 2.5.1, only Normal and Small Alt Mode were indicated

System Options: Bit field indicating various modes / features status per the following table.

Value	Indicator
+1	Is Tach Mode enabled? (+1 = Yes)
+2	Is the Feature-IN port activated, ala, has Voltage been applied to the Feature-in wire? (+2 = Yes)
+4	Is White Space activated at this time? (+4 = Yes)

This is a Summed tabled, in that each indicator is added into a final number as shown here. Example: A *System Mode* value indicted that Tach Mode has been enabled, and that the device is presently operating in Half-Power mode.

Note that prior to 2.5.1, only Tach Mode indicator was implemented.

CP Index: Which Charge profile (1..8) is currently being used?

BC Mult: 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.

SysVolt: 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
2.67	32v	2.667x
4	48v	4x

Fractional values may also be used to support battery voltages such as 8v, 32v, 42v. Those values will need to be selected manually via the $SCO: command.

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

CapRPMs: And this will be the RPMs at which that capacity was noted at.

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

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

ForcedTM?: 0 or 1, has user forced Tach-Mode on via the $SCT: command.

RequiredSensorFlag: Is a Required Sensor missing? 0 indicates no missing Required Sensor has been detected, for non-zero values reference $SCA: command on page 51 for details.

FLT; -- FAULTED

FLT: "FLT;, FaultCode, RequriedSensorStatus "

System has Faulted, fault code number.

RequiredSensorStatus is a combined number following the pattern as defined by the $SCA command and detailed in `Table 1 - Required Sensor Encoding ' on page number 55

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

AOK; -- ACKNOWLEDGE

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

NAK; -- NEGATIVE ACKNOWLEDGE

Sent to indicate the rejection of a received change command ($CPx, or $SCx), $MSW, or $EDB. NAK is a new capability added in Firmware version 2.3.x and may be used in addition to AOK timeouts to indicate a miss formed command.

DBG; -- DEBUG

Special string with extra internal parameters for use in debugging internal operations.

RST; -- RESET

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

SENDING DATA TO THE REGULATOR:

All commands begin with the character `$', contain 3 letters(CAPS) followed by a `:' and then parameters as requested by the command. All must end with a CR (or CR/LF) or may optional be terminated with the character `@'. A complete 'string' must be received within 60 seconds from the `$' to the ending `@'/CR/LF, else the regulator will abort the capture of that string and begin looking for a new `$' starting character.

Defining Charging Voltages and Amps – All volts and amps used in the Charge Profile Entries are represented by a normalized 12v 500Ah battery and are automatically scaled depending on the sampled battery voltage at startup and the setting of the Battery Capacity DIP switches.

Take care 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 serial port. In addition, IF the request arrived via the J1939 CAN 'Terminal' DGN (17E00h), a copy of the reply will also be returned to the requesting CAN node. This is useful to gain access to the advanced ASCII setup parameters of the WS500 Alternator Regulator via the CAN.

$CNG: - Changes eNGine Configuration table  

Use to set parameters associated with Engines that are not already defined elsewhere

---

$CNG: <J1939MaxLoad>, <RFM_RPM>, <RFM1>, <RFM2>, <RFM3>, <RFM4>, <RFM5>, <RFM6>, <RFM7>, <RFM8>

J1939MaxLoad: **< FLOATING POINT NUMBER (0.0 → 1.20)>** Indicated the maximum percentage engine load that should be allowed as reported via J1939 PGN Number 61443. Field Drive of the Alternator will be reduced as needed to lower total Engine Load to at or below *J1939MaxLoad* % target.

Another way *J1939MaxLoad* may be used is if PGN 61443 does not contain the Engine Load % field, but does contain the Throttle Position field – if the reported throttle position exceeds *J1939MaxLoad*, the regulator will then be placed into half-power mode.

Set = 0 to disable monitoring of J1939 based engine load. (Default)

RFM_RPM: <WHOLE NUMBER (-10,000 → 10,000)> RPM based Field Modifier. The RFM capability allows for an RPM based curve of 'White Space' engine power which may be directed to the Alternator. This is for use to support white-space engine load capping via a non J1939 based approach. *RFM_RPM* defines the upper expected limit of Engine RPMs. From there 8x engine RPM ranges are defined which are then associated with the RFMx caps. Each RMP range will be defined on a 100x RPM boundary based on *RFM_PRM* / 8

A non-zero value for *RFM_RPM* enables the RPM Field Modifiers, which may be triggered in one of two ways. 1st: If RFM_RPM is a positive value, FEATURE-IN will be re-purpose FEATURE-IN to enable RFM mode when active; overriding the default use of FEATURE-IN. Care must be made if the installation requires access to Forced-Float mode (CPE #8), or Equalize Mode (CPE# 1-7) via external hardware switches as this re-purpose of FEATURE-IN overrides any other use, even if the Feature-In modifier has been used (See $SCO: command). Typically, one should expect to use CAN based communications to the regulator for access to those functions when FEATURE-IN has been repurposed to RFM. (Setting = 0 will restore FEATURE-IN to behave in its normal fashion)

The 2nd way to activate RPM Field Modifiers is to use a negative value for RPM_PRM. In this case, the Modifies will be active at all times, independent of the FEATURE-IN; the function of which will not be altered.

Set = 0 to disable RPM based White Space definitions. (default = 0)

RFMx: <FLOATING POINT NUMBER (0.0 → 1.0)> Used in conjunction with *RFM_RPM* above. In each case when active via FEATURE-IN the *RFMx* is used to further modify the active 'De-Rate' value (drtNORM, drtSMALL, drtHALF – reference $SCA command). In this way loading may be adjusted to an Engine RPM dependent limit when the engine is servicing another non-alternator load (example, a marine engine driving a propeller – feature-in would be connected to the Transmission interlock switch).

The RFMx factors provide an *additional* modification to the maximum allowed field drive. As an example of the relationship between RFM_RPM and RFMx, the following commands are issued:

$CNG: 0.0,3200, 0.1, 0.2, 0.4, 1.0, 1.0, 1.0, 0.95, 0.5@

Will result in the following max PWM field drive limits when FEATURE-ON is active:

RPMs	Normal	Half	Small Alt
0-400	10%	5%	7.5%
400-800	20%	10%	15%
800-1,200	40%	20%	30%
1,200-1,600	100%	50%	75%
1,600-2,000	100%	50%	75%
2,000-2,400	100%	50%	75%
2,400-2,800	95%	42%	67%
2,800 and above	50%	25%	71%

(Tabled built using default $SCA: derate values of 100%, 50% , & 75%):

RFMx only modify the upper limit for field drive, all other functions of the regulator remain in place and at any given point in time the actual field drive % may well be below the upper limit. Note also that care should be used in configuring RFMx modifiers, especially if RPMs are measured using the stator, as setting the field drive cap to low could prevent accurate RPM readings thereby preventing the regulator from ever starting any alternator field drive.

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

This command will cause the ACCEPT (and BULK) portion of a Charge Profile Entry 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 <VBat Set Point>, <Exit Duration>, <Exit Amps>,<Reserved>

---

n: (7 → 8) '*n*' is the Charge Profile Table Entry that will be modified. Use range 7 to 8.

VBat Set Point: <FLOATING POINT NUMBER (0.0 → 16.5) > Voltage the Regulator will use during ACCEPT phase. When this voltage has been reached, the regulator will transition from BULK to ACCEPT phase. This value is a floating-point number and entered for a normalized 12v system (See note: Defining Charging Voltages and Amps)

Exit Duration: <WHOLE NUMBER (0 → 600 (10 hours)) > After entering ACCEPT phase, a timer will be started. After '*ExitDuration*' minutes have expired ACCEPT mode will exit and the regulator will move to OVER-CHARGE mode. Setting '*ExitDuration*' = 0 will disable time based exiting of ACCEPT mode and only AMP based monitoring will be used.

Exit Amps: <WHOLE NUMBER (-1 → 200)> After entering ACCEPT phase, delivered amps will be monitored and if they fall to (or below) *'ExitAmps'* ACCEPT mode will exit and the regulator will move to OVER-CHARGE mode. This is providing 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 *'ExitAmps'* = 0 will disable amp-based exiting of ACCEPT mode and only time monitoring to '*ExitDuration*' will be used.

Setting *'ExitAmps'* = -1 will disable amp-based exiting of ACCEPT mode and time monitoring of '*ExitDuration*' will be used as above. In addition, when the time spend in Acceptance mode has exceed 5x the duration spent in Bulk mode, the regulator will also trigger an exit. (Adaptive Acceptance).

Note: If you set BOTH '*ExitDuration*' & *'ExitAmps'* = 0, the regulator will bypass ACCEPT mode.

Reserved: <0> Must be 0.

Example:

$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 OVER-CHARGE (Sometimes referred to as FINISH phase) portion of a Charge Profile Entry 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 Amps>, <Exit Duration>, <Exit VBat>, < Exit Amps >

---

n: (7 → 8) '*n*' is the Charge Profile Table Entry that will be modified. Use range 7 to 8.

Limit Amps: <WHOLE NUMBER (-5 → 50)> After entering OVER-CHARGE phase, delivered amps will be monitored and regulated to *'LimitAmps'* . Setting *'LimitAmps'* = 0 will disable OVER-CHARGE mode.

Exit Duration: <WHOLE NUMBER (0 → 600 (10 hours)) > After entering OVER-CHARGE phase, a timer will be started. After '*ExitDuration*' minutes have expired OVER-CHARGE mode will exit and the regulator will move to FLOAT mode. Setting '*ExitDuration*' = 0 will disable OVER-CHARGE mode.

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

Exit Amps: <WHOLE NUMBER (0 → 50)> If defined, *'ExitAmps'* will keep the regulator in OVER-CHARGE phase until the battery acceptance current has fallen to (or below) this value. During this time voltage will remain at *'ExitVBat'* . *'ExitAmps'* will NOT override the overall '*ExitDuration*' transition criteria. Set = 0 to disable this option.

In all applications of OverCharge phase, Battery Voltage will be limited to the MAXIMUM of either '*Exit Vbat*' as defined in $CPO, or '*VBat Setpoint*' as defined in the $CPA command above.

During operation, if Vbat falls notably below the batteries Bulk/Acceptance target as defined by CPA, the device will revert back to Bulk/Acceptance phase. If this occurs repeatedly, review your amperage transition values as it may be they are too low. Or it simply may be a one-off condition where a large load was placed in the system and the charging device is not able to support, requiring energy to be withdrawn from the battery.

In another special application of OverCharge phase (available in firmware v2.4.x and above), the *limit amps* may be set to -1. In this case 'Exit Vbat' is often set to match that of the Float target voltage and the above mentioned revert to Bulk/Acceptance is not checked for. This usage of Overchage can be helpful to manage the transition from Acceptance phase to Float in cases where it is desired to keep an external tachometer active.

$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 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 <VBat Set Point>, <Limit Amps>, <Exit Duration>, <Revert Amps>, <Revert Amp-hours>, <Revert Volts>, <Revert SOC>

---

n: (7 → 8) '*n'* is the Charge Profile Table Entry that will be modified. Use range 7 to 8.

VBat Set Point: <FLOATING POINT NUMBER (0.0 → 16.5) > Voltage the Regulator will use during FLOAT phase. This value is a floating-point number and entered for a normalized 12v system. Setting this value to 0.0v will cause all charging to stop during 'float' mode. (See note: Defining Charging Voltages and Amps)

Limit Amps: <WHOLE NUMBER (-1 → 50)> While in FLOAT phase delivered Amps will be monitored and regulated to *'LimitAmps'* . Setting *'LimitAmps'* = -1 will disable this feature and only '*VBatSetPoint'* will be regulated.

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

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

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

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

Revert Amps-hours: <WHOLE NUMBER (-250 → 0)> After entering FLOAT mode if the accumulated number of Amp Hours removed from the battery exceeded *'RevertAmp-hours'* the regulator will re-start a charge cycle, looping back to BULK mode. Setting *'RevertAmp-hours'* = 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 raw RevertAmps, but it is only usable if the amp shunt is placed on the battery.

Revert Volts: <FLOATING NUMBER (0.0 → 16.5)> While in FLOAT mode, if battery voltage drops below *'RevertVolts'* we assume this indicates a large load has been placed on the system and the regulator will re-start a charge cycle, looping back to BULK mode. Setting *'RevertVolts'* = 0 will disable this feature.

Revert SOC: <WHOLE NUMBER (0 → 100)> While in FLOAT mode, if the batteries SOC (State Of Charge) as reported by a CAN attached BMS or other RBM drops below *'RevertSOC'* the regulator will re-start a charge cycle, looping back to BULK mode. Setting *'RevertSOC'* = 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 are a negative value, and measure the number of AHs removed from the battery after entering Float mode.

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

This command (with its parameters) will cause the POST- FLOAT portion of a Charge Profile Entry 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 Duration>, <Revert VBat>, <Revert Amp-hours>, <VBat Set Point>

---

n: (7 → 8) '*n*' is the Charge Profile Table Entry that will be modified. Use range 7 to 8.

Exit Duration: <WHOLE NUMBER (0 → 30000 (500 hours)) > After entering POST-FLOAT phase, a timer will be started. After '*ExitDuration*' minutes have expired POST-FLOAT mode will exit and the regulator will revert to BULK phase. Setting '*ExitDuration*' = 0 will cause the regulator to remain in POST-FLOAT mode unless triggered by another exit criteria.

Revert VBat: <<FLOATING POINT NUMBER (0.0 → 16.5)> While in POST-FLOAT mode, if the system battery voltage drops below *'RevertVBat''* 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 *'RevertVBat'* = 0 will disable this feature.

Revert Amps-hours: <WHOLE NUMBER (-250 → 0)> After entering POST-FLOAT mode if the accumulated number of Amp Hours removed from the battery exceeded *'RevertAmp-hours'* 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 *'RevertAmp-hours'* = 0 will disable this feature

VBat Set Point: <FLOATING POINT NUMBER (0.0 → 16.5) > Voltage the Regulator will use during POST-FLOAT phase. This value is a floating-point number and entered for a normalized 12v system. If set = 0.0 (default), charging will be disabled during post-float mode.

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

This command (with its parameters) will cause the EQUALIZATION portion of a Charge Profile Entry 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 <VBat Set Point>, < Max Amps >, <Exit Duration>, <Exit Amps>

---

n: (7 → 8) '*n*' is the Charge Profile Table Entry that will be modified. Use range 7 to 8.

VBat Set Point: <FLOATING POINT NUMBER (0.0 → 20.0) > Voltage the Regulator will use during EQUALIZE mode. This value is a floating-point number and entered for a normalized 12v system (See note: Defining Charging Voltages and Amps). Setting *'VBat' = 0 will* disable EQUALIZE mode.

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

Exit Duration: <WHOLE NUMBER (0 → 600 (10 hours)) > After starting an EQUALIZE phase, a timer will be started. After '*ExitDuration'* minutes have expired EQUALIZE will emanate and the regulator will enter FLOAT mode. Setting '*ExitDuration'* = 0 will disable EQUALIZE mode.

Exit Amps: <WHOLE NUMBER (0 → 50)> During EQUALIZE mode delivered Amps will be monitored and if it falls to (or below) *'ExitAmps'* 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). It is up to the operator to keep the engine speed up and allow for a full equalization session to occur. Setting *'ExitAmps'* = 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 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^{0}C>, < Min Comp Temp >, <Min Charge Temp>, <Max Charge Temp>, < Rdc Volts>, < Rdc Low Temp>, < Rdc High Temp>, < Rdc Amps>, <MaxBatAmps>, <MaxBatVolts>

---

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 12v system at 25c. (See note: Defining Charging Voltages and Amps). Set = 0.0 to disable temperature based voltage compensation.

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

Min Charge Temp: <WHOLE NUMBER (-50 → 10) > If the battery drops below this temperature, charging will be disabled. Once temperature rises above *Min Charge Temp*, a new charge cycle will start.

Max Charge Temp: <WHOLE NUMBER (20 → 95) > If the battery reaches this temperature, the system will be disabled to protect it. If the battery temperature continues to rise, the system will eventually FAULT when battery temperature exceeds *Max Charge Temp* by 20%. Note that Min and Max Temp limits will override any outside direction the WS500 regulator received; even if a CAN connected BMS is asking for charging, if either of these hard stop limits are exceed the regulator will enter Disabled mode.

Rdc Volts: <FLOATING POINT NUMBER (0.0 → 12.0) > If the Battery Voltage is at or below this value, the regulator will work in Reduced Charge mode, where the maximum amps delivered to the battery will be limited to *Rdc Amps*. Set *Rdc Volts* = 0.0 to disable checking of battery voltage for Reduced Charging mode.

Rdc Low Temp : <WHOLE NUMBER (-99 → 20) > If the battery temperature falls to or below this value in degrees C, the regulator will enter Reduced Charge mode. Set = -99 to disable this test

Rdc High Temp : <WHOLE NUMBER (-99 → 95) > > If the battery temperature rises to or above this value in degrees C, the regulator will enter Reduced Charge mode. Set = -99 to disable this test

Rdc Amps : <WHOLE NUMBER (0 → 100) > When the regulator is in Reduced Charge mode, battery acceptance current will be limited to this value. Rdc Amps is defined relative to the normalized 500Ah battery and is adjusted by the system battery capacity. As with *Max/Min Charge Temp*, the any of the

RDC triggers (Temp, voltage) will also override any directions arriving from the BMS. Set = 0 to disable Reduced Charge mode.

Max Bat Amps : <WHOLE NUMBER (0 → 2000) > If set, this will be the maximum amperage allowed to flow into the battery. Remember, this value is set relative to the 12v/500A 'nominal' battery and it is also adjusted by the Battery Capacity Multiplier; so a value of 250 would cause the regulator to limit charging current to a 0.5c rate. Set = 0 (default) to disable Max Battery Amps

Max Bat Volts: <FLOATING POINT NUMBER (0.0 → 20.0)> *Max Bat Volts* is used to provide an additional fault check for the maximum allowed battery voltage under all conditions. If at any time Max Bat Volts is exceeded, the device will enter a FAULT state. This check is in addition to existing built-in high voltage limits, and allows for a more restrictive value to be defined by the installer. Set = 0.0 (default) to disable this extra check

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

Restores to default (values at compile time) Charge Profile Entry '*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.

$DEP: - DEPendency

Dependency is a way to allow the device to confirm both the correct device type, as well as a minimum firmware version required to utilize this configuration. In most cases sending in configuration strings with additional parameters (ala, new capabilities) that the present level of firmware, the device will simply ignore the extra parameters; and this might cause undesirable behavior. Beginning with firmware version 2.4.2, an optional $DEP may be used to allow the device to check to see if the firmware is able to support all configuration commands and parameters.

$DEP: <Version>, <Version> …

Version: (String) '*Version*' is in the same form as that reported out by $SST: and contains both the device type as well as a minimum release of firmware needed. Take care when declaring *<version>* as the format is well defined: *mmmmxx.yy.zz*

mmmm: a 4 character string containing the device type. Supported devices types are:

- AREG: WS500 Alternator Regulator; OEM branding is also an AREG device.
- DCDC: WS3000r DCDC Converter. OEM branding is also a DCDC device.

xx, yy, zz: Numeric values separated by two '.' xx, yy, zz must be complete and numeric (No wide cards allowed). The device firmware must be equal to or greater than the defined xx, yy, xx

Examples of $DEP are:

AREG2.4.2 → WS500 Alternator Regulator, firmware must be 2.4.2 or above
DCDC2.4.3 → WS3000r DCDC Converter, firmware must be 2.4.3 or above.

$DEP: DCDC2.4.3, AREG2.4.2 @

$RAS: - Request All Status back

This command will instruct the WS500 Alternator Regulator to send out via the Serial port a copy of the all known status strings. It is useful for external applications to capture the current status w/o needing to await the arrival of each string (which could take several minutes or more, depending on how status strings are spaced 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 WS500 Alternator Regulator to send out via the 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 currently selected CPE

Special version of Request for CPE, this will send back the currently selected (via the DIP switched) 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 WS500 Alternator Regulator 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 only Charge Profile 7 or 8 (the two customizable entries) may be modified via an ASCII command. This is to reduce the potential for major errors in the regulator.

Also, take great care in setting these values, esp 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 WS500 Alternator Regulator may be deployed with the amp shunt at the Alternator, or at the battery, some of 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.

$RLF: - Request Last Fault

This command will instruct the WS500 Alternator Regulator to send a copy of the Last Known Fault information, including the Fault Number and a copy of the AST and CST strings at the time of the fault – preceded by two dots (..). After the last string is sent out, $AOK will be sent.

If no recorded Fault exists, only the $AOK will be sent. $MSR will clear any saved last-fault information (along with all other configuration information!).

$RLF:

After all status strings have been sent, "AOK" will be sent to indicate the end. Example of Low Voltage Fault:

```
$RLF:@

..FLT;,14,0
..AST;,0.00, ,0.06,-3.8,-3.8,0, ,13.10,100,15000,2, ,-99,-99, ,0, ,0.06,19,-99,0
..CST;,1,0,2,70,  ,1,1,  ,1,0,1, ,0,0,0,  ,129, ,0,0,0
AOK;
```

$SCA: - Changes ALTERNATOR (and System) parameters  

Used to update the system configuration entries associated with the Alternator and System.

$SCA: <BTS2ATS?>, < Alt Target Temp >, <Alt Derate (norm) >,<Alt Derate (small) >,<Alt Derate (half) >, <PBF>, <Alt Amp Cap>, <System Watt Cap.>, <Amp Shunt Ratio>, <Shunt Reversed?>, <Idle RPMs>, <Warmup Delay>, <Required Sensors>, <Ignore Sensors>, *<BMS Amp Cap>*

BTS2ATS?: <WHOLE NUMBER (0, or 1) > If *BTS2ATS?* is set = 1, the Battery Temperature Sensor will be treated as a 2nd Alternator Temperature Sender. The warmer of the two sensors will be used as Alternator Temperature for both reporting and regulation. Useful if two alternators are being driven from one regulator using a common field. Make sure the Battery temperature is being supplied via the CAN.

Using the BTS for a 2nd ATS is also useful with the DC-DC converter option; it allows two alternators to be monitored.

(Default = 0)

Alt Target Temp: <WHOLE NUMBER (15 → 150)> Operating temperature the regulator should attempt to keep the Alternator under in degrees C. If the Alternator temperature exceeds this value, the regulator will reduce field current to allow the alternator to return to a safe operating temperature. If the Alternator temperature continues to rise and exceeds this temperature by 10% the regulator will fault out and stop producing power.

Alt Target is used by the DC-DC converter to reduce power transfer as alternators reach their target temperature.

Alt Derate(norm),
Alt Derate(small),
Alt Derate(half): <FLOATING POINT NUMBER (0.0 → 1.00) > These derating values are used to limit the alternator's maximum current output to some % (10% to 100%) of its demonstrated capability (see *Alt Amp Cap*). The three values correspond to the mode the Alternator:

- Normal - Condition when either of the other modes are not selected.
- Small Alternator Mode – selected via DIP switch 8.

- Half Power Mode – Selected by shorting the Alternator NTC temperature sensor wires. Half-power mode may also be selected when engine RPMs fall below the threshold set via *<Trigger Half-power RPMS>* in the $SCT: commend below, or via the Feature-In (See $SCO: command below).

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 the amp shunt is not connected.

If a given *Alt Derate* values is set = 0.0 (0%), this will cause the alternator to be shut down and the regulator to enter a ready-idle state. Such a condition can be useful if deploying a DC-DC converter (See $CDD command on 72) by always powering the regulator and using half-power to select between when the engine is running or turned off, but in all cases allowing the DC-DC converter to be managed. When released from read-idle mode, the alternator will change to pre-ramp mode and do its normal startup process.

Note: When setting the 'Alt Derate(x)' values, make sure the 'Normal' derate value is as large or larger than any of the other two, else an internal error/hard fault may occur.

Note: Beginning with release 2.5.0, if Derate values have not been explicitly set by sending a $SCA: configuration command to the device (meaning, the device is running using factory default configuration values for $SCA parameters), and the system deployed is a 48v system, the derate values noted above will be reduced automatically to approximately a quarter of the default values. This is done to account for the large number of 48v alternators which utilize a 12v field. It is STRONGLY recommended that with 48v installs (and all installs for that matter) that the device be fully configured to match the system design, including appropriate derate values for the alternator.

PBF: < INTEGER (-1 → 10)> Pull-back factor for reducing Field Drive at lower RPMs. If the WS500 Alternator Regulator 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 (DEFAULT) to disable this feature.

Set = -1 to cause Field Drive to be reduced to a maximum of 70% drive in the case where the WS500 Alternator Regulator 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.

If PBF is used, for many engine / alternator combinations a value of 1 will 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 sail-boat engine with a large 150A 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 temperature sensor attached to the alternator in this case – to prevent unintended overheating during prolonged idle periods.

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

Note: It is suggested that White Space feature be used instead of PBF, as they are simpler to define and have additional capability. See $CNG: on page 37

Alt Amp Cap: <WHOLE NUMBER (-1→ 500) > Alternator Amperage Capacity (<Alt *Amp Cap*>) is an uncommon factor and primarily used when the Current Shunt located at the alternator (vs. the battery). When defined the regulator will limit the Amperage output of the alternator to this value, after applying the '*Alt Derate xxx*' factors. (Reference Derate factors above) 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 1st entering Bulk phase and in this way will auto-sample the alternator size based on its capabilities.

(Default = 0, disabled)

Do note with this option, there may be some interactions between the field % pullbacks and an attempt to reduce what is perceived as 'alternator current' pullback, for example during half-power mode.

With Alternator Amperage Cap 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 Amperages – care should be used when setting up the system. (Default: 1.00, 0.75, 0.50 respectively)

System Watts Cap: <WHOLE NUMBER (-1→ 20000)> This 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 in 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 1st entering Bulk phase. This will then be used to define the Amp Limit of the Alternator.

> 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)

Amp Shunt Ratio: <WHOLE NUMBER 500 → 60000)> Enter the ratio of your Amp measurement shunt 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!

Shunt reversed?: <WHOLE NUMBER (0, or 1) > Allows software correction if the Amp Shunt was wired backwards. Set = 1 and Amp readings will have their polarity changed.

Idle RPMs: < INTEGER (0 → 2500)> 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 WS500 Alternator Regulator. 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 (very small engine with large efficient alternator), Idle RPMs may be defined artificially high; doing so will cause the regulator to increases its pull back of Field Drive during low RPM operations.

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

Warmup Delay: <WHOLE NUMBER (-600 → -15 15 → 600) > Hold-off period when regulator is 1st 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 RAMP mode. Default = 30 seconds. Beginning with v2.4.1 firmware, use of a negative number for Warmup Delay will cause the Fast Ramp feature to be disabled, resulting in a slow ramping up of alternator at all times. This is more consistent with how v2.2.0 firmware always operated.

If a negative value is used for Warmup Delay, then Fast Ramping is disabled and all ramps will be at a slower pace. One might use this in cases where the underlying engine is not able to adjust quickly, perhaps an older mechanical injection diesel, or if the installer wishes to provide slower changes for some other reason – belt wear, etc. Using a negative value for Warmup Delay will slow down the pace of all load increasing changed. Do not, this will NOT reduce the ability to pull back when needed, example a limit is reached. This only impacts how quickly loads are applied.

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, 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 fall-back 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 **	4	Force to FLOAT mode (See note**)
CAN RBM	8	Disable charging, Feature-in may be used to Force to Float when CPE #8 is selected
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 the feature-out port / dash-lamp it will also flash out error codes during a fault.

If the WS500 Alternator Regulator has been configured to repurpose the BTS for a 2nd alternator, a failure in either ATS or the repurposed BTS (now known as ATS2) will trigger the Alternator Temperature Required Sensor check.

The WS500 Alternator Regulator 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 Bat and Alt sensors being critical, sending 131 instead of 3 will cause the regulator to FAULT if either is noted as missing or fails.

Battery current / Battery Temperature values arriving via the CAN from a remote battery master or BMS may be used to satisfy the Bat Temp and or Bat Current checks in *Required Sensors.* In addition, by selecting *CAN RBM* a check for the entire presence of a remote battery master or BMS may be required as well.

Note** When not receiving Battery Current via the CAN, and utilizing the shunt for measurements, it is difficult to determine if an Amp Shunt has failed vs. a true reading 0A of current. Because of this, the WS500 Alternator Regulator will delay check for the presence of a working Current Shunt until after Bulk has been completed. A current reading of greater then 5A at any time during BULK phase will be used as an indicator that the current shunt is present and working, as will the arrival of battery current measurements via the CAN. Once this determination is made no additional checks will be made – as a valid operation condition for the regulator may at times have a battery current of 0A (example, when actively regulating current to 0A in FLOAT mode).

Ignore Sensors: <WHOLE NUMBER (0 → 255) > A companion to *Required Sensors*, this bit-flag parameter will cause the regulator to ignore measurements on the flagged sensors. Typically this is only used where sensing is done via Remote Sensing, example: If Battery Current is supplied via CAN from the BMS, and the local current shunt is not used - by setting the *Ignore Sensor* bit, random noise is prevented from causing miss readings. The stator is another key example, if not used one can ground that signal wire, or set the *Ignore Sensor* bit for the Stator.

Ignore Sensors is a number created by summing up the value associated with each potential critical sensor. For example: if you wished to ignore the Alternator temperature and current shunt readings, you would enter 5 (1+4).

Sensor	**Value**
Alternator Temperature Sensor	1
Battery Temperature Sensor	2
Local Current Shunt	4
Stator	8
Feature In (always treated as Inactive)	16

Table 2 – Ignore Sensor Encoding

Be careful with the relationship between *Ignore* and *Required* Sensors, an ignored senor will cause a positive trap if it is 'required' and the value is not supplied elsewhere. Also be careful with the Feature-In signal wire, especially as it is at times used with BMS's in a legacy configuration to signal that charging should be stopped.

BMS Amp Cap: <WHOLE NUMBER (0 → 2500) > BMS Amperage Capacity (<BMS *Amp Cap*>) issued to define a global system limit in a like way that <*System Watts Capacity*>. BMS *Amp Cap* defined the maximum charge amperage the BMS is defined to support. If the battery $CPB MaxAmps (aka, C rate) allows current to exceed the maximum allowed BMS current, the regulator will restrict current so as to not over-load the BMS rating. *BMS Amp Capacity* is clipped to 10A increments, so a defined value of 258A will be saved as 250A

If BMS Aggregation is enabled, the total BMS current allowed will be increased to match that of the total recognized online and active BMS's.

If a RBM is controlling the charge cycle, sending in goal battery current limits, BMS *Amp Cap* will be ignored.

Set = 0 (Default) to disable BMS Current Limit.

$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:

- If not already done, set regulator's name
- Disable Idle adaptive pullback (alternator has massive cooling capability, and engine has sufficient reserve)
- Adjust amp shunt to 250A/75mV
- Define engine idle @ 550 RPMs
- Must have Alternator Temperature sensor present, else cause regulator to FAULT
- Leave other values as default (See *Appendix D:)*

```
$SCN:0,MainsAlt,5555@
$SCA:0,95,1.0,0.75,0.50,0,0,0,3333,0,550,60,129@
$RBT:@
```

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

- Set name to MainsAlt
- Acceptance @ 14.4v until acceptance current is less than 1% of capacity, 8hr max.
    - *8hr Time limit is set as fall-back 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 post-float phases.
- Allow Alternator to operate up to 105c
- 30mV temp comp

```
$SCN:0,MainsAlt,4449@
$SCA:0,105,1.0,0.75,0.50,0,0,0,3333,0,550,30,0@
$CPA:7 14.4,480,5,0@
$CPO:7 0,0,0,0@
$CPF:7 13.2,-1,0,0,-10,12.7,0@
```

```
$CPP:7 0,0,0,0.0@
$CPE:7 15.3,0,180,0@
$CPB:7 0.030,0,-45,45,0,-99,-99,0,0@
$SCO:7,3,1,0,0@
$RBT:@
```

Example: Shorted warm-up delay to 15 seconds, leaving the rest of the configuration at the default values:

```
$SCA:0,90,1.0,0.75,0.50,-1,0,0, 10000,0,0,15,0@
```

$SCT: - Changes TACHOMETER parameters 

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 WS500 Alternator Regulator.

$SCT: <Alt Poles>, < Eng/Alt drive ratio >, <Tach Min Field>, <ForceTM>, <Trigger Half-Power RPM>

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 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 (7.0 / 2.3)

Tach Min Field: <WHOLE NUMBER (0 → 30)> This is the % value the PWM will be kept at as the minimum drive when *ForceTM* (Below) has been enabled. *BE VERY CAREFUL* with this value as it will set the floor in which the alternator is driven. If that floor is too high, it will prevent the regulator from 'regulating', burning out the battery. This is the actual PWM value sent to the field drive; though it is capped at 30%.

~~Set *Tach Min Field* = -1 to enable auto-determination.~~ (Note: Disabled, installer must set value manually. -1 will be treated as Disabled (0))

If a *Tach Min Field* value is set (any value greater then 0), and TACH-MODE is enabled, the regulator will use this value as a minimal field drive %. 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 *Tach 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 ENABLED, 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.~~ (Note: This hold check was removed beginning with Firmware 2.5.1)

Set = 0 (default) to not inject any artificial PWM % min value.

And a special note on Li based installs be very careful with *Tach Min Field* as incorrect setting will prevent the regulator from fully stopping charging when the battery has reached it target SOC.

ForceTM: <0, or 1> Setting the value to 1 will enable Tach-mode. (0 = DEFAULT)

Trigger Half-Power Mode RPM: <WHOLE NUMBER (0 → 10,000)> If the measured engine RPMs is below this value, the regulator will be forced into Half Power mode (See $SCA command above)– in the same way as if the ATS sensor lines are shorted. This feature is most usable of the regulator is CAN connected to the engines J1939 bus and therefore able to reliably pick up engine RPMs (as opposed to using Stator sensing directly).

Set = 0 (DEFAULT) to prevent RPM triggered Half-Power Mode.

Example: $SCT: 12,2.83,0,0,500@

$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 and Battery Capacity selection, as well as other capabilities of the device. This command also allows the selection of auto detect for system voltage (12v, 24v, 48v), or forcing a fixed defined target system voltage.

$SCO: <CP_Index>, <BC_Index >, <SV_Override>, <Lockout>, <Feature-IN>, <Feature-OUT>, <Promiscuous Mode>

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. It is possible to also receive the battery capacity via CAN. In normal cases the *BC Index* will be overridden by a CAN received value. However, if you do not wish the *BC Index* to be overridden by CAN received values; configure *BC Index* with a negative value. The multiplier will have the same effect, the sign of the value only indicates if the *BC Index* should also override any CAN received value or not. Positive = yes, allow a CAN supplied value to override; negative = no, ignore both the DIP switch and any value is supplied via the CAN.

SV Override: < FLOATING POINT NUMBER (0.0 → 4.5)> 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
2.67	32v	2.67x
3	36v	3x
3.5	42v	3.5x
4	48v	4x

(Set SV Over-ride = 0 to restore auto-selection of 12v, 24v, or 48v 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 master reset, or returning the device to the factory for resetting. No other command, not even $MSR: (unless given the password) 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 returning the device to the factory.***~~

0 = No locking out.

1 = Prevent any configuration changes.

~~2 = Prevent any configuration changes – may NOT be cleared via $MSR: method~~.
(Redacted in v2.4.3, option 2 behaves the same as option 1.)

To clear a Lockout level 1, issue the $MSR: command with the devices password (See $SCN command). A level 2 lockout is not clearable and the regulator will need to be returned to the factor for servicing.

Feature-IN: <WHOLE NUMBER (0 → 2)> Modifies the behaviors of the Feature-in port according to the following table. Applying a modifier will override any other default behavior.

Feature-IN Override	Option	Modified behavior of Feature-IN port
0	Default	
1	CPE #8 Force To Float (Inverted)	Function of Feature-IN will be inverted. Active will allow normal operation; while inactive will force the regulator into FLOAT mode. For use only with CPE #8, otherwise this option has no effect.
2	Select Half-power (Inverted)	If Feature-in is not active, regulator will be placed into Half-power mode (See $SCA command above). When selected, activating Feature-in will release regulator from half-power mode.

Feature-OUT: <WHOLE NUMBER (0 → 3)> Modifies the behaviors of the Feature-out port according to the following table. Applying a modifier will override any other default behavior.

Feature-IN Override	Option	Modified behavior of Feature-OUT port
0	Default	
1	Activate in post Float	Feature-out will remain inactive until the Post-Float phase is reached. Often used to signal generator-stop when a battery if fully charged. The device entering a HARD FAULT will also activate the Feature-out with this option.
2	Battery Heater Active	Caused Feature-out to become active when the battery heater is indicated as being active via CAN. Useful to drive enable pin on external power source for battery heater as BMS indicates a need.
3	Fault indicator	Feature-out will become active ONLY if the WS500 is in a faulted condition. In this case both the initial 'engine warm up check' and the blinking out of the fault code number is suppressed and instead the Feature-out will be active (grounded) only when a true fault occurs.

Promiscuous Mode: <0, or 1> Setting the value to 1 will allow a number of hard faults to self reset as opposed to requiring power cycling to clear. Reference "

Appendix E: Error codes and meaning" with items marked with the modifier of 0x4000. (0 = DEFAULT)

$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.

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

- If not already done, set regulators name
- Use CPE#3 (FLA#2 – large batteries)
- 1500AH battery (BC Index = 3)
- 12v system (SV Override = 1)
- Lockout NOT enabled (Allows continued changes)

```
$SCN:0,MainsAlt,5555@
$SCO:3,3,1,0,0@
$RBT:@
```

$SCN: - Changes NAME (and PASSWORD)

Update Name and Password configuration. The name is used by the CAN subsystem to identify *this* regulator.

$SCN: <RESERVED>, < Reg Name >, <Reg Password>

RESERVED <WHOLE NUMBER (0, or 1)> Set = 0;

Reg Name: <STRING (up to 18 characters, no spaces, comma, or '@') > Name used for CAN ID. If you have twin engines, you might wish to set these to descriptive names.

Reg Password: <STRING (up to 18 characters, no spaces, comma, or '@') > If the *Reg Password* string starts with a dot (.) character, this will be considered 'Hidden' and not reported out in the NPC status string. Do not that in such cases the '.' is still considered part of the password and will need to be sent in anytime it is needed (For example, to release a lockout level)

$SCN: 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.

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

$SCR: - RESTORES System Configuration table to default

Restores System Configuration values to original factory default values.

$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.

$CCN: - Change parameters in the CAN Configuration table

Configure the CAN Configuration Table for this regulator

$CCN: <Battery Instance Override>, <Device Instance >, <Device Priority>, <AllowRMB?>, <ShuntAtBat?>, <Enable-OSE?>, <Enable-NMEA2000?>, <Enable_ALT_CAN?>, <Engine ID>,<BitRate>,< DC_ Disconnected _VBat>, <Aggregate_BMS>

Battery Instance Override: <WHOLE NUMBER (0 → 100)> What battery instance is this device associated with? (1..100). Set = 0 to use DIP switches for selection.

Device Instance: <WHOLE NUMBER (1 → 13)> Which instance of charging devices is this? Allows unique identification of charging sources. Device Instance is mostly used by external displays. It is considered good practice to give each regulator a unique *Device Instance* number, however if the installer fails to do that the WS500 will automatically adjust the *Device Instance* number in case of a conflict. (Default = 1)

Device Priority: <WHOLE NUMBER (1 →250)> What is the relative priority of this charging device?

A key value of the OSEnergy protocol is the ability to prioritize charging sources. This value is what is used to decide a given charging sources priority. If the needs of the associated battery (and any additional loads) can be met by higher priority charging sources the regulator will reduce its output to 0A. However, if the battery/load needs cannot be meet the regulator will deliver current to its limits as needed. If there are two or more charging sources with the same priority, battery /load needs will be split between them. (Useful in dual engine installations to balance loads between both engines). (Alternators Default=70)

Device priority is also used to decide who should act as the Remote Battery Master, or the overall coordinator in the system to assure all charging devices are working towards the same goal. If *AllowRBM?* Is enabled, the WS500 Alternator Regulator will assume the RBM role if no other higher device exists. This can be useful in simple installations where no Battery Monitor is installed or as a fall-back for a failed battery monitor.

Allow RBM?: <WHOLE NUMBER (0..2)> Should the WS500 Alternator Regulator attempt to act as the Remote Battery master?

0 = Do not allow the regulator to assume the RBM role.
1 = Allow the regulator to potentially assume the RBM role. (Default)
2 = Allow the regulator to potentially assume the RBM role as a 'Virtual' device.
Used in combination with *<enable_ALT_CAN?>, a Virtual RBM,* is a mode where the "alternative supported CAN" data is re-transmitted as a RV-C compliant RBM. Note that in this mode, though the WS500 will represent the alternative CAN supported BMS as an RV-C compliant device, if another device with higher priority is present the WS500 will following RV-C spec and stop transmitting as an RBM.

Also, in the lack of an active alternative CAN device, Virtual RBM mode will defer back to a standard RBM mode.

Shunt At Bat?: <WHOLE NUMBER (0, or 1)> Is the shunt connected to the Battery? Used during RBM mode to know if we are seeing alternator or battery current. 0=no, 1=yes(default)

Enable OSE?: <WHOLE NUMBER (0, or 1)> Should the WS500 Alternator Regulator send and receive OSEnergy (RC-V) status and coordination messaged via the CAN bus? 0=no, 1=yes(default)

There may be some simple installations where one wishes to use the WS500 Alternator Regulator to only broadcast status to NMEA2000 devices, and the OSEnergy messaging (RV-C standard) causes issues with some existing NMEA2000 devices. Do note that disabling OSEnergy mode will remove many of the systems benefits such as coordinated / prioritized charging, simplified remote instrumentation, and more.

Enable NMEA2000?: <WHOLE NUMBER (0, or 1)> Should the WS500 Alternator Regulator send out NMEA-2000 like status messages via the CAN bus? 0=no, 1=yes(default)

Enable_ALT_CAN: <WHOLE NUMBER (0 → 255) > Though RBMs supporting RV-C/OSEnergy protocols provide the best level of integration, there are other CAN based BMS and remote sensors which may be utilized. *Enable_ALT_CAN* is a way to signal which of those additional protocols to support. Note that some protocols are able to provide full RBM direction, while others only offer a limited amount of remote sensing. It is up to the installer to assure which messages are supported by the attached device and if the resulting system integrations meet expectations and needs. A value of 0 (default) will disable any additional CAN protocol support, and it should also be noted that a given device may require other configuration changes such as adjusting CAN baud rate, and even disabling some optional protocols supported (e.g., NMEA2000)

Protocol	Value	Features
NMEA2000 RAT	1	Battery current and temperature via NMEA2000 battery monitor.
SMA	2	BMS devices supporting 11-bit 'Sunny' SMA type messages. Aka, 'Victron BMS' protocol
SMA w/'Zero Output Technology" enabled.	3	Enabled active regulation of battery current to 0A (and passed voltage goal) instead of disabling alternator with so signaled from BMS.
MG Energy Systems	4	Support for MG Energy Systems BMS.
MG Energy Systems w/'Zero Output Technology" enabled.	5	MG Energy Systems BMS with active regulation of battery current to 0A.
Discover	6	Support for Discover Battery CAN messages (AEBus Protocol)
Discover w/'Zero Output Technology" enabled.	7	Support for Discover Battery AEBus with active regulation of battery current to 0A when SOC >= 100%
Lithium Werks	11	Support for Lithium Werks BMS using proprietary communications protocol.
Lithium Werks w/'Zero Output Technology" enabled.	12	Utilizes active regulation of battery current to 0A when 'do not charge' state is requested from BMS.
LUX	13	'LUX' 11-bit BMS protocol.
LUX w/'Zero Output Technology" enabled.	14	Enabled active regulation of battery current to 0A (and passed voltage goal) instead of disabling alternator with so signaled from BMS.
Victron LYNX BMS	16	Support for Victron LYNX BMS
Victron LYNX BMS w/'Zero Output Technology" enabled.	17	LYNX BMS with active regulation of battery current to 0A.
Victron SmartShunt	18	Receive Battery Amps, and optional Temperature, via Cerbo / SmartShunt combo.

Table 3 – Alternative CAN protocols support

Note: Prior to Firmware v 2.1.0, Enable_ALT_CAN was a single value of 0 or 1 and used to specify only NMEA200_RAT support. Refer to

Appendix A: CAN enabled BMS for additional details.

Engine ID: <WHOLE NUMBER (0 → 250)> Used to associate the regulator with the engine it is mounted on. Specifically with regards to RPMs. The regulator monitors for a matching J1939 engine RPM (PGN: 61444) and will use it instead of measured stator RPMs. Also, if NMEA2000 messages are enabled and the regulator is able to measure RPMs – NMEA2000 PGN: Engine parameters rapid (#127488) will be sent with the RPMs indicated as being associated with this Engine ID. Default ID = 0.

BitRate: <WHOLE NUMBER (0 → 4)> If supported by hardware, allows CAN communications rate to be changed from the default 250Kbps.

Value	Rate
0	Default (250Kbps)
1	100Kbps
2	125Kbps
3	250Kbps
4	500Kbps

DC_ Disconnected _VBat: <FLOATING POINT NUMBER (0.0 → 20.0)> If a DC-Disconnect command is received via the CAN bus, the charging source will normally be placed into a Disabled mode. However, it may also be optional placed into a CV state using this voltage.

A value of 0.0 (default) causes transition to DISABLED mode or FAULT stat.

Aggregate BMS: <WHOLE NUMBER (0 → 10)> Some installs utilize more than one BMS in a given battery bank. This may be done to support slightly different locations (ala, port and starboard side of a vessel due to space constraints and/or weight distribution), or more commonly to increase maximum available current by paralleling BMSes. In doing so, not all BMS devices will perform aggregation, instead they will present themselves as different Battery Instance ID's, despite being connected to the same physical DC bus. *Aggregate BMS* allow support of this type of install by monitoring for any BMS or potential RBM with not only the configured *Battery Instance* number (See above), but also up to 10 Battery Instances. From the base *Battery Instance* up to *Battery Instance* + *Aggregate BMS.* Example, if *Battery Instance* is set for 2 (either via the DIP switches, or *Battery Instance Override above)* and *Aggregate BMS* is set for 5, then ANY battery Instance from 2 to 7 will be consider all part of the same 'battery' and aggregated by the WS500.

Aggregation will sum up the battery current, and capacity; will monitor for any given battery device coming on line or going offline. The request charge state will be the least aggressive (e.g., if ONE online battery asked for Float, then the WS500 will enter Float mode even if the remaining batteries are asking for Bulk). The battery with the most extreme temperature (Hot or cold) will be used to make temperature related decisions, and if a given battery goes off line the WS500 will continue to

monitor and respond to the remaining batteries. Once if ALL batteries go offline the WS500 will enter a fault state.

Aggregate BMS is an advanced capability, and it is up to the installer to test and assure the overall system responds in an expected way. Note that if enabled, Remote Voltage Sensing is disabled and the WS500 MUST be installed with the voltage sensing wires attached to an appropriate point in the battery bank. At present, the following BMS's have been proofed with aggregated:

- MG Energy BMS
- Lithionics NeverDie™ BMS

A value of 0 (default) will disable any battery aggregation.

$CCN: 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.

$CDD: - Change parameters in the DC-DC Configuration table 

Configure the DC-DC converter for this regulator

$CDD: <Model>, <Mode>, <Volts>, <Volts_HalfPower>, <CoCharge_Amps_limit>, <Aug_Amps_limit>, <Aug_limitV>, <Aug_limitSOC>,

---

Model: <WHOLE NUMBER (0 → 5)> Which DC-DC converter is attached?

Value	Brand	Supported Modes
0	Not Present (default)	n/a
1	WS3000r APS12BI58-3000	0..3
2	WS3000r-24	0..3
3	WS1500	0..3
5	Wakespeed WS48-12X	0..3

Default = 0 (None)

Mode: <WHOLE NUMBER (0 → 3)> How should optional DC-DC converter behave?

Value	Mode
0	Off, Disable, Not Present
1	Primary Battery Charging only
2	2nd Battery Augment only
3	Dynamic

The WS500 Advanced Alternator Regulator has the ability to manage an optional external DC-DC converter and control the transfer of power between two separate DC busses:

- *Primary Battery:* The Battery the WS500 is attached to and managing. Typically a higher voltage House Battery (ala, 48v Lithium house battery). Note that there may or may not be an alternator connected to the Primary Battery and controlled by the WS500.

- *2nd Battery:* A separate battery system the other side of the DCDC convert is attached to. Typically a lower voltage battery, example, the 12v chassis starter battery with its associated factory 12v alternator. In this case the WS500 is *NOT* directly managing the factory 12v alternator.

In *Primary Battery Charging* mode the converter is used to transfer power from a 2nd battery/alternator to help meet the charging needs of the primary battery the WS500 is managing. Note that if there is also a Primary Alternator the WS500 is controlling, both the alternator and the DCDC converter will be managed to meet the battery charging needs. In *2nd Battery Augment* mode, power is taken from the primary battery/alternator and directed to the 2nd battery. *Dynamic* mode intelligently manages between *Charge Assist* and *2nd Battery Augment* mode, directing power as needed and available.

In a typical application the WS500 is attached to the house battery, with our without an associated alternator also controlled by the WS500; the 2nd battery being the chassis battery with the cassis alternator. Here the DC-DC converter may be configured for either charging of the house battery using excess power from the Chassis Alternator, or transfer power from the house battery/alternator to the chassis battery – either to augment the existing chassis alternator during high current demands, or fully emulating a 12v alternator serving the needs of the chassis battery with power supplied by the house alternator. An example of this would be to replace the OEM alternator with a 48v alternator directed to the House battery, the DC-DC converter may be configured in *2nd Battery Augment* mode to emulate the removed 12v alternator and provide power to the chassis system, legacy 12v loads, and the starter battery.

Another option would be to leave the OEM 12v alternator in place untouched, configure the DC-DC converter for *Charge Only or Dynamic* mode allowing for the charging of a 48v house battery without the addition of a 48v alternator. Even though the WS500 is not controlling a 48v physical alternator, it is still managing the charge cycle to the 48v battery, including the complete CPE profiles as well as any CAN based BMS integration.

Dynamic is helpful in installs where the 2nd battery is at times charged, and at times not. A great example is an RV where the 2nd battery is the chassis battery. By configuring the WS500 to be powered at all times (Brown wire always powered) and connect the Ignition wire to the Feature-in (Selecting Half-Power), the WS500 will be able to recognize when the RV engine is running and help charge the 48v house battery. Then when the engine is tuned off the Alternator is disabled (via setting the Half-Power derate value of 0.00) and the DCDC converter is turned around to support ongoing 12v legacy house loads directly connected to the starter battery without fear of discharging it (Hint, install a safety low-voltage cutout between the starter battery and legacy 12v loads just in case!)

See note of how MODE can be overridden while in the Half_power state depending on the value of *Volts_HalfPower* . See more below.

Default = 0 (Off/Disabled)

Volts: <FLOATING POINT NUMBER (0.0 → 33.0**)> This value is used both when transferring power to the 2nd battery, as well as while pulling power from the 2nd battery. If the DC-DC converter is configured in 2nd Battery Assist mode the DC-DC converter will deliver this fixed voltage. A typical value is 13.8v to allow charging of the Starter battery. While in Dynamic Mode, if the 2nd battery voltage drops below this threshold value, the DC-DC converter will begin transferring energy from the House battery to the 2nd battery.

And finally, while in Charge mode, if the 2nd battery falls below this value, power transfer will be stopped.

If set = 0.0 (default), the DC-DC converter will not provide power to the 2nd battery if so configured, but will provide the max DC-DC converter energy to the primary battery. Use 0.0 with care.

Volts_HalfPower: <FLOATING POINT NUMBER (-33.0 → 33.0**)> Optional target voltage which is selected if the regulator is running in Half-Power mode. (See $SCA command on page 51 and $SCT command on page 59). Typically this is used in conjunction with some indication for when the engine is not running to have the DC-DC converter operate with a lower 2nd battery voltage.

When the system is operating a Half Power state (ala, the *Volts_HalfPower* value is used as the setpoint value), using a negative value for *Volts_HalfPower,* the Mode of the DCDC converter will be overridden and forced into *Mode* = 2 (*2nd Battery Augment).* Care is needed as this override is absolute, and will occur even if the configured *Mode* is set for any other value (other than Disabled). But in this way the installer is able to have more control over if the DCD Converter depending on if the engine is running or not.

Set = 0.0 (default) will cause the DC-DC converter to be fully disabled in Half-power mode.

CoCharge_Amps_limit: <WHOLE NUMBER (-500 → 500**)> While charging the Primary Battery, the DC-DC converter will limit the amount of power transferred between batteries to the maximum of its either the maximum capability of the DC-DC converter, or *CoCharge_Amps_limit*. This current limit is typically measured on the 2nd battery side of the DC-DC converter (Check the Supported DC-DC converter table above for any notes). A value of 0 (default) will cause the DC-DC converter to run at its

maximum capability. Set *CoCharge_Amps_limit* to match the DCDC converter installation wiring as well as limits to the 2nd battery alternator capability.

A negative value will cause *CoCharge_Amps_limit* to be reduced by 50% when HalfPower mode is selected.

AUG_amps_limit: <WHOLE NUMBER (0 → 500**)> While augmenting the 2nd battery with power from the primary battery, the DC-DC converter will limit the amount of amps transferred between batteries to the maximum of its capability, or *AUG_amps_limit*. This current limit is typically measured on the 2nd battery side of the DC-DC converter, unless noted above below in the Supported DC-DC converter table. A value of 0 (default) will cause the DC-DC converter to run at its maximum capability. Set this to match the DC-DC converter installation wiring as well as the maximum discharge limit of the primary battery.

> *** Note: All the DC-DC related limits will be validated against the selected Device Type. If a value outside the allowed range of the device has been specified (Min and/or max), a 'Miss-configured' fault will be issued as there is something wrong with the system design. Make sure to validate all these limit values against the specific device specification sheets.*

AUG_LimitV: <FLOATING POINT NUMBER (0.0 → 20.0)> Discharge limit for the Primary battery, If the Primary battery voltage drops below this value, *2nd battery augmentation* mode is disabled. Setting *AUG_limitV* provides for protection of over-discharging the primary battery.

NOTE: The value here is normalized to the 12v/500Ah battery concept in line with all other CPE values. In this way the system will automatically adjust for a 24v or 48v install without needing to modify this value.

Set = 0.0 (default) to disable this check

AUG_LimitSOC: <WHOLE NUMBER (0 → 100)> Discharge limit for the Primary battery. If the SOC reported by an externally CAN connected BMS reports the SOC (State of Charge) of the Primary battery falling below this limit, *2nd battery augmentation* mode is disabled.

Set = 0 (default) to disable this check

When supporting a DCDC converter, there are additional capabilities of the WS500 which may be utilized. Example, consider also the application of 'Required Sensors' (ref $SCA: command), and specifically the requirement of a CAN connected BMS/RBM. By setting Required Sensors, one will be able to disable the DC-

DC converter when there is no CAN connected battery present. An example might be in a case where the house battery is physically located in a trailer which can be disconnected from the tow vehicle. By using Required Sensors with RBM, no power will be presented to the trailer wiring unless a CAN communication link has been established beforehand with the BMS.

Consider also the placement of an ATS (Alternator Temperature Sensor) on the 12v alternator. Doing so will allow the WS500 to control the power demands of the DC-DC converter taking into account the 12v alternator temperature, reducing those demands if the temperature rises above the Alternator Temp Setpoint in the $SCA command. By utilizing CAN based BMS connection, and remote instrumentation of the battery temperature, the BTS can be repurposed for a 2nd ATS allowing both the 48v and the 12v alternator to be monitored.

Consult details in the selected DC-DC Converter for additional voltage and/or current limits, which may likely be lower than the ASCII command limits shown here.

$CDD: 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.

$CCR: - RESTORES CAN Configuration to default

Restores CAN Configuration values to original as-compiled (default).

$CCR: 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.

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

$MSR: - RESTORE all parameters to factory default.

Restores all configurable parameters to the factory default values. This is a combination of the $SCR: , $CCR commands, the $CPR:n commands for all Charge Profile Entry tables, and $RBT; command. Alternator will RESET after this command is completed.

$MSR: <Password>

Note $MSR is disabled if the Regulator has been locked out via the $SCO command. If successful, the regulator will reply with “AOK;” and then reboot.

Password: <STRING (up to 18 characters, no spaces, comma, or ‘@’) > Optional *Password* may be supplied to clear a level 1 lockout. (Take note of the space between the ‘:’ and the <password>, example: *$MSR: 1234@)*

$MSR: will not be recognized if system has been locked-out (level 1) via the $SCO: command unless the correct devices password is supplied. Note that a lockout level 2 is not clearable, even if the correct password is supplied.

Note: Some versions of firmware require a space between the : and the password, example:

$MST: 1234@

As opposed to:

$MSR:1234@

$EDB: - Enable DeBug serial strings

Will cause regulator to start sending $DBG; strings via ASCII communication ports. If it is powered down, or reset (ala a Fault, or by receiving a command string that causes a reset), the regulator will restore to its default handling of $DBG: strings.

$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 Alternators configuration via other ASCII commands. This will restart the regulator, allowing those changes to be recognized – but more important some hardware needs the $RBT: command as a signal to actually save requested changes in non-volatile memory. If your device using what is known as EEPROM-Emulation, any changes you make will not be saved until you issue the $RBT: command.

$RBT: will respond with the string *RST;* if it is accepted and the device will then restart (reboot). Note that this is different than the more common $AOK; response.

$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.
- C = Force into CONFIGURATION 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 meet, the regulator will again exit that phase quickly. In such cases the mode may be chanced before the next $AST string is sent.

Configuration Mode is a special mode where the regulator will listen and process all CAN and ASCII communications, but it will not actively drive the alternator nor doing error checking. It can be useful if you wish to do extensive configuration of the regulator without worrying about overheating the field, or have faults reported due to not yet in place sensors. As well as allowing simpler bench-top configuration w/battery under-voltage faults occurring. Once entered, the only way to exit CONFIGURATION mode is to $RBT: or power-cycle the regulator.

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 A: CAN ENABLED BMS

The ability for the WS500 Alternator Regulator to communicate with BMS devices using the CAN (Control Area Network) provides for a higher level of integration and reliability than is possible using only legacy methods (e.g., Charge Enable wire), as well as installation simplification via remote instrumentation of battery current, voltage, and/or temperature (BMS support dependent). The WS500 primarily utilized RV-C CAN protocol (ref: http://www.rv-c.com/), and providing a battery/BMS is also compliant with this standard the WS500 will operate seamlessly without any additional configuration needed.

In addition, other CAN protocols may be supported. This appendix provides details of optional CAN protocols supported by the WS500 regulator. Refer to the command $CCN: on page 67 and how the '*Enable_ALT_CAN'* field is defined.

> ***WARNING: Not all BMS devices provide advanced notification of pending DC disconnect. It is up to each installer/designer to verify the details of the BMS and its integration with the Wakespeed WS500 Advanced Alternator Regulator to assure a reliable system. It is suggested that at minimum a 2 second warning be issued by the BMS before any actual physical disconnect occurs.***

CAN based BMS integration can be a challenging concept at first, though the end result does allow for a simpler and more robust overall system install. If the installer has any questions about what is contained in here, or the suitability of a given BMS, send an Email to: *support@TJCMicro.com*

NMEA2000_RAT: Should the WS500 Alternator Regulator look for a NMEA2000 device to supply remote sensing of battery Amperage and Temperature via PGN: 127508 and PGN: 127506 for SOC and SOH. To reduce confusion in the NMEA2000 network, if *Enable_NMEA2000_RAT* is set = yes, and the sending of PGN 127508 messages for the Battery Instance will be suppressed. When using NMEA2000_RAT mode, great care is needed in the overall setup with regards to the Battery Instance. Some systems ignore this field and set all devices to Instance=0, which will cause great confusion.

Example product(s) include:

- Maretron DCM100
- Victron Smart Shunts

SMA: The 'Sunny' SMA CAN protocol is an 11-bit message commonly used in the Solar and Electric Vehicle industry. It allows for a BMS to tell charging sources to start, stop, and which goals to utilize. Often know as the SMA protocol, it is also reflective of the Victron 11-bit BMS protocol, though at a different CAN speed. Some representative BMS's which use this protocol includes:

- MG Energy (https://www.mgenergysystems.eu/) *
- REC BMS (https://www.rec-bms.com/ , https://rec-bms-na.com/) *
    - *MUST* use 'Victron' variant of this BMS
- Orion Jr (https://www.orionbms.com/)
    - Select 'Victron Inverter'
- SIMP BMS (https://github.com/tomdebree/SimpBMS)

* Note: These SMA BMS's have been tested in the lab and/or in field installations. Others may have only been confirmed via specifications. Contact support@wakespeed.com if you have any questions on a specific BMS.

The 11-bit CAN messages will be 'converted' into an OSEnergy type device with a fixed CAN-ID of 69 and a priority of 120. Aggregation is not supported. The following CAN messages are utilized. (Refer to your BMS guide for details of these messages)

- 0x351: Battery Charge Voltage, Charge Current Limitation
- 0x356: Battery Current, Battery Temperature
- 0x35A: Alarm/Warning

Both 0x351 and 0x356 must be broadcasted at a minimum rate of 1,000mS to allow the WS500 to recognize a device and begin following its directions. Once doing so, if *Battery Charge Voltage* and *Charge Current Limitation* (Message 0x351) are both non-zero the WS500 will enter CV/CC mode using those goals. A zero value for one or both of those will cause the WS500 to enter standby. Optionally, if *SMA w/'Zero Output Technology"* is selected a zero goal value for current with a non-zero value for voltage will cause the WS500 to actively regulate battery current to 0A, while leaving the alternator able to supply energy to ongoing house loads.

Message 0x356 may be used to remotely supply *Battery Current* readings to the WS500, reducing the need to install a physical current shunt. In addition, *Battery Temperature* measurements will be used in combination with the presently active CPE to enforce hard charging limits based on high and/or low temperatures, even if the BMS continues to request charging.

0x35A: The Alarm/Warning message will be monitored for any active indication. Upon receiving a Warning indicators, the WS500 will stop charging and perform a recoverable fault (regulator will restart). However if a Fault flag is received, the WS500 will enter a non-recoverable hard fault.

Different BMS devices behave differently, but the best practice is if the BMS sends out a warning and/or fault 2 seconds before doing a physical disconnect to prevent voltage spikes in the system from an uncoordinated disconnect.

MG Energy System: The MG Energy Systems line of BMSs and batteries supports both an 11-bit SMA like interface and a 29-bit based messaging protocol. It is preferred to utilize the 29-bit messages. When selected the 29-bit protocol is a J1939 compatible messaging system utilizing a series of public and private messages that will then 'converted' into an OSEnergy type device using the original CAN node address and with a fixed priority of 120. The MG Energy System BMS issues a proper warning before disconnects, assuring safe shutting down of the alternator.

MG Energy System support allows for remote instrumentation of battery current and temperature, the sending of battery charge goals and limits (Voltage/current), Dynamic C-Rate management based on installed battery capacity, as well as Aggregation of up to 10x

As with the SMA protocol, when *Battery Charge Voltage* and *Charge Current Limitation* are both non-zero the WS500 will enter CV/CC mode with those goals. A zero value for one or both of those will cause the WS500 to enter standby. Optionally, if *"w/Zero Output Technology"* is selected a zero goal value for current with a non-zero value for voltage will cause the WS500 to actively regulate battery current to 0A, while leaving the alternator active to supply energy to ongoing house loads.

Beginning with 2.5.1, support for the MG Energy SmartLink has been added. During running if a SmartLink controller is detected the Wakespeed WS500 will lock onto that while ignoring the individual BMS's. If a SmartLink is not present the WS500 will monitor each BMS directly to perform aggregation.

Victron LYNX BMS: The Victron line of LYNX BMSs are supported using a proprietary 29-bit protocol which is compatible with J1939. When selected the LYNX messages are 'converted' into an OSEnergy type device using the original CAN node address with a fixed priority of 120.

LYNX BMS support allows for remote instrumentation of battery current and temperature, the sending of battery charge goals and limits (Voltage/current), and Dynamic C-Rate management based on installed battery capacity.

Regulators disable or 'Zero Output Technology' options may be selected to adjust how the WS500 behaves when the BMS request no additional battery charging. . The LYNX BMS will also issues a proper warning before disconnects, assuring safe shutting down of the alternator.

Lithium Werks:** Support for the Lithium Work BMS is provided via a proprietary CAN communications protocol. When enabled, this protocol will allow for the Lithium Werks BMS to send charge goals and limits (Voltage and current) to the WS500 Alternator Regulator. Battery current and Temperature are also provided as well as BMS warning and alarming states.

Both normal and Zero-output technology is supported, and the communications protocol may allow for stable active current regulation using the Remote shunt, however if instability is noted a physical shunt may need to be installed. The Lithium Werks BMS does not provide a 2-seconds advanced notice of impending disconnect, as such some level of mitigation steps will need to be put into place in the overall system design.

****Note: As of June 2021, this protocol has not been proofed in the field. Contact support@wakepseed.com to enquire about the latest status.**

LUX: The LUX protocol is a slight variant of the SAM protocol. Also an 11-bit message, it is common in China sourced BMS's. As with the SMA handling, LUX 11-bit CAN messages will be 'converted' into an OSEnergy type device with a fixed CAN-ID of 69 and a priority of 120. Aggregation is not supported. The following CAN messages are utilized. (Refer to your BMS guide for details of these messages)

- 0x351: Battery Charge Voltage, Charge Current Limitation
- 0x356: Battery Current, Battery Temperature
- 0x359: Alarm Only (Warning ignored)
- 0x35C: Charge Enable Bit

Details for message 0x351 and 0x356 are the same as with SMA, detail for message 0x359 is the same as the SMA 0x35A but a different address. Message 0x35C includes byte-0, bit 8 to incident if charging is allowed. It (along with a goal current of 0A in message 0x351) will be used to direct the WS500.

Batteries which have been proofed with the LUX protocol include:

- Dyness: B4850 (https://www.dyness-tech.com.cn/product/26.html)
    - Requires use of 'Wakespeed' BMS firmware, contact Dyness.
- PylonTech US3000C
    - Requires use of 'Wakespeed' BMS firmware, contact PylonTech

Discover Battery: Support for the Discover Battery BMS is provided via a proprietary CAN communications protocol. When enabled, this protocol will allow for the Discover Battery BMS to send charge goals and limits (Voltage and current) to the WS500 Alternator Regulator. Battery current and Temperature are also provided as well as BMS warning and alarming states.

Both normal and Zero-output technology is supported, and the communications protocol may allow for stable active current regulation using the Remote instrumentation, however if instability is noted a physical shunt may need to be installed. The Discover Battery BMS does not provide a 2-seconds advanced notice of impending disconnect, as such some level of mitigation steps will need to be put into place in the overall system design.

APPENDIX B: CAN MESSAGES

The WS500 Alternator Regulator contains a Control Area Network (CAN) subsystem. The purpose of this network is to allow communications of status, configuration, and coordination of charging in a systems view.

The WS500 Alternator Regulator utilizes a mixture of open source standards including:

- OSEnergy – (https://github.com/OSEnergy/OSEnergy) Open Systems Energy initiative: Overriding specification defining communication hardware and protocols allowing for coordination of charging devices.
- J1939 – SAE standard providing basic coordination of nodes and communications of messages
- NMEA-2000 – Marina orientated status messages. Proprietary specification built upon J1939.
- RV-C -- (RV-C.com) True open source specification targeting primary Recreation Vehicle industry also based on J1939. Extended to include many needed communications to support the OSEnergy initiative.
- Others: A few selected 2.0A (11-bit addressing) messages are supported as documented in $CCN on page 67.

CAN messages summary

The following summarizes the pgns sent and/or received via the WS500 Alternator Regulator over the CAN bus. Items in ***BOLD*** are utilized by the WS500 (Sent or received) – though the actual messages may be impacted if a subsystem is enabled as well as if the WS500 is operating as the RBM for the system.

Instances:

A key concept for all the CAN based communications is Instances. Instances allow for the identification of unique devices and the messages associated with such devices. The WS500 utilizes 3 distinct 'Instances':

- Battery Instance
- Charger Instance (Alternator)
- Charger Instance(s) (DC-DC Converter)
- 2nd Battery (Low Side) instance (DC-DC Converter)
- Engine Instance

When parsing CAN based messages, it is important to keep the concept of Instance in mind. A detail of NMEA2000 messages: NMEA2000 defines the 1st occurrence of a Battery Instance as 0, while RV-C uses 1 (0 is invalid). For NMEA2000 messages the WS500 will use 'BatteryID - 1'.

Charger Instances will use the OSEnergy standard (RV-C compliant) representation of instances. Alternator based messages will be represented with (Charger Instance + 0x30); 0x30 being assorted with 'Engine Driving' charging sources. As an example: The 1st 'charger' instance will be 49 (for WS500 instance #1), and 50 for a 2nd WS500 if installed.

The user is able to configure any of these Instance numbers (Battery, or Charger) – and details are up to the installer.

DC-DC Converter:

If the optional DC-DC converter is installed and configured additional status messages will be sent. The Instance number used by the DC-DC converter as associated with the primary battery will be:

- DC-DC Converter Charger Instance = $CCN <Device Instance > + 0x70
- 2nd (Low Side) battery Instance :
    - $CCN<Battery Instance Override> + 1

- – OR –
- $CCN <Battery Instance> + $CCN <Aggregate BMS>

The 2nd battery 'Instance' number is set automatically as n+1, where n is the primary (High Side) Battery Instance number. Example, if the main battery instance number is 1, then the 2nd battery instance will be 2. However, if BMS Aggergation has been enabled via the $CCN <Aggregate BMS> value, then 'n' become the entire set of instance numbers aggregated. So, if the main battery instance is 1, but aggregation is set to 10x, then instances 1..10 are aggregated and the 2nd battery becomes instance 11

When the DC_DC converter is present and enabled, the following CAN status messages may be transmitted:

- Charger Status - 1FFC7h
- Charger Status2 - 1FEA3h

Note that with the addition of a DC-DC Converter up to 3x sets of 'charger status' messaged may broadcasted :

- Primary Battery Alternator based status
- Primary Battery DC-DC Converter based status (If enabled)
- 2nd battery DC-DC Converter based status.

'Housekeeping' Messages:

Each of the supported CAN protocols has a number of house keeping messages, these are fundamental to the protocol and the user is refer to the respective standards for details. Some examples of 'Housekeeping' messages supported by the WS500 include:

- 0xEA00 – J1939 Address Request
- 0xEE00 - J1939 Address Clamed
- 0xEA00 - J1939 DGN/PGN Request

- 0xFEEB - J1939 Product identification message
- 0xFEDA – J1939 Software Identification message
- 0xE800 - J1939 Acknowledgment
- 126208 – NMEA2000 Request Group Function
- 126464 – NMEA2000 PGN Tx and Rx list
- 126996 - NMEA2000 Product Information

```
/*****************************************************************************
// NMEA2000-DC Detailed Status  - PGN127506
// Tx:
//  - SID                   Sequence ID. If your device is e.g. boat speed and heading at same time, you can set
//                          same SID for different messages to indicate that they are measured at same time.
//  - DCInstance            DC instance.
//  - DCType                Defines type of DC source. See definition of tN2kDCType
//  - StateOfCharge         % of charge  (If provided by BMS)
//  - StateOfHealth         % of heath
//  - TimeRemaining         Time remaining in minutes
//  - RippleVoltage         DC output voltage ripple in V
```

Two copies of 127506 will be sent out:

- One identifying a 'Battery' DCType and "BatteryID – 1" for the instance.
- One identifying as 'Alternator' DCtype and using "Charger Instance" + 0x30 for the instance.

If the DC-DC Converter is active, three additional messages will be sent out:

- One identifying a 'Battery' DCType and "2nd (Low Side) battery Instance" as defined above for the instance.
- Two identified as 'Converter' DCType and using "Charger Instance" + 0x70 (HS) or 0x71(LS) for the instance.

```
/*****************************************************************************
// NMEA2000-Charger Status  - PGN127507
// Tx:
//  - Instance                    ChargerInstance.
//  - BatteryInstance             BatteryInstance.
//  - Operating State             see. tN2kChargeState
//  - Charger Mode                see. tN2kChargerMode
//  - Charger Enable/Disable      boolean
//  - Equalization Pending        boolean
//  - Equalization Time Remaining  double seconds
```

```
/*****************************************************************************
// NMEA2000-Battery Status  - PGN127508
//   This PGN will not be sent if ENABLE_NMEA2000_RAT is set = YES.
// Tx:
//  - BatteryInstance       BatteryInstance.
//  - BatteryVoltage        Battery voltage in V
//  - BatteryCurrent        Current in A
//  - BatteryTemperature    Battery temperature in Ã‚Â°K. Use function CToKelvin, if you want to use Ã‚Â°C.
//  - SID                   Sequence ID.
```

If EnableAltCan = NMEA2000_RAT is selected (See $CCN: command), the 'Battery' copy of this PGN will be suppressed, to reduce the potential for confusion in a multi-WS500 install.

```
/*****************************************************************************
// NMEA2000-Battery Status  - PGN127508
//   This is a 2nd copy of PGN127508 which is sent containing the Alternator specific details
// Tx:
//  - AlternatorInstance       ChargerInstance**  (0x30 + Charger Instance from $CCN:)
//  - AlternatorVoltage        Alternator voltage in V
//  - AlternatorCurrent        Alternator Current in A
//  - AlternatorTemperature    Alternator temperature in Ã‚Â°K. Use function CToKelvin, if you want to use Ã‚Â°C.
//  - SID                      Sequence ID.
```

Notes: The use of PGN127508 is more accurately describes as a DC Source message, and not restricted to 'Batteries'. This sent instance is transmitted using the Alternator specific measured values. As an example, if the device is configured to NOT have the 'Shunt At Bat', it can be assumed the shunt is located on the alternator itself (with Battery current being delivered via CAN), and this message is then able to be used to see that detail via NMEA2000.

VAlt+ measurements have proven to be too unreliable; all 'VAlt' measurements are based on the value sensed by the Vbat+ line. The Charger Instance number follows the OSEnergy/RV-C standard in that it contains both the Charger Instance number as well as an Identifier that this is an Alternator. Charger Instance consists of 0x30 being added to the defined Charger Instance number via $CCN:

```
/*****************************************************************************
// Charger Configuration Status     127510
// Note this has not yet confirmed to be right.
// Tx:
//  - ChargerInsance              ChargerIntance.
//  - BatteryInstance             BatteryInstance.
//  - Charger Enable/Disable      tN2kOnOff
//  - ChargeCurrentLimit          CurrentLimit
//  - CharginAlgorithm
//  - ChargerMode
//  - BatteryTemperature
//  - Equalization Enable/Disable
//  - Over Charge Enable/Disable
//  - Equalization Time Remaining seconds


 /*****************************************************************************
 // NMEA2000-Battery Configuration Status   -- 127513
 // Tx:
 //  - BatteryInstance       BatteryInstance.
 //  - BatType               Type of battery. See definition of tN2kBatType
 //  - SupportsEqual         Supports equalization. See definition of tN2kBatEqSupport
 //  - BatNominalVoltage     Battery nominal voltage. See definition of tN2kBatNomVolt
 //  - BatChemistry          Battery See definition of tN2kBatChem
 //  - BatCapacity           Battery capacity in Coulombs.
 //  - BatTemperatureCoeff   Battery temperature coefficient in %
 //  - PeukertExponent       Peukert Exponent
 //  - ChargeEfficiencyFactor Charge efficiency factor
```

```
//*************************************************************************
// Converter (Inverter/Charger) Status 127750
// Tx:
//  - 1 Sequence ID
//  - 2 Connection Number
//  - 3 Operating State
//  - 4 Temperature State
//  - 5 Overload State
//  - 6 Low DC Voltage State
//  - 7 Ripple State
//
```

Note that 127750 does not describe a mode for fixed voltage/current charge (CVCC mode), and as such 'Bulk' will be used will be used in those cases.

```
 /*************************************************************************
 // Engine parameters rapid  - 127488
 // Rx:
 //  - EngineInstance        Engine instance.
 //  - EngineSpeed           RPM (Revolutions Per Minute)   (Both send and receiving of N2K RPMs supported)
 //  - EngineBoostPressure   in Pascal
 //  - EngineTiltTrim
```

```
//*************************************************************************
// Engine parameters dynamic     -- 127489
// Rx:
//  - EngineInstance        Engine instance.
//  - EngineOilPress        in Pascal
//  - EngineOilTemp         in Kelvin
//  - EngineCoolantTemp     in Kelvin
//  - AltenatorVoltage      in Voltage
//  - FuelRate              in litres/hour
//  - EngineHours           in seconds
//  - EngineCoolantPress    in Pascal
//  - EngineFuelPress       in Pascal
//  - EngineLoad            in %  (Receive only)
//  - EngineTorque          in %
```

```
/*****************************************************************************
// Engine parameters rapid  - 127488
// Tx:
//  - EngineInstance        Engine instance.
//  - EngineSpeed           RPM (Revolutions Per Minute)   (Both send and receiving of N2K RPMs supported)
//  - EngineBoostPressure   in Pascal
//  - EngineTiltTrim
```

Note: Tx rate and Priority of Engine Parameters Rapid has been reduced from 100mS/3, to 500mS/7 to better match exiected usage (Human display) and reduce potential for CAN message blocking of other critical messages.

```
 /*************************************************************************
 // DC Source Status 1
 // Input:
 //  - Instance              DC Instance (bus) ID.
 //  - Device Priority       Relative ranking of DC Source
 //  - DC Voltage            0..3212.5v, in 50mV steps
 //  - DC Current            -2M..+2MA, in 1mA steps (0x77359400 = 0A)


 /*************************************************************************
 // DC Source Status 2
 // Input:
 //  - Instance              DC Instance (bus) ID.
 //  - Device Priority       Relative ranking of DC Source
 //  - Source Temperature    -273 to 1735 Deg-C  in 0.03125c steps
 //  - State of Charge       Batteries: % SOC;  DC Charging sources:  Current % output.
 //  - Time Remaining        Estimated number of minutes until SOC reaches 0% or 100%
 //  - Time Remaining Interpretation     Time to Full Charge, or time to Full Discharge?

/*************************************************************************
// DC Source Status 3 - 1FFFBh
// Input:
//  - Instance              DC Instance (bus) ID.
//  - Device Priority       Relative ranking of DC Source
//  - State of Health       % expected remaining lifetime
//  - Capacity Remaining    Current capacity / capability of battery in Ah
//  - Relative Capacity     % of current capacity vs. design specified capacity.
//  - AC RMS Ripple         in mV
*/

 /*************************************************************************
 // DC Source Status 4
 // Input:
 //  - Instance              DC Instance (bus) ID.
 //  - Device Priority       Relative ranking of DC Source
 //  - Desired Charge Mode   Charging mode / state being requested.
 //  - Desired DC Voltage    Target voltage for chargers to deliver  0..3212.5v, in 50mV steps
 //  - Desired DC Current    Target current for all chargers to deliver combined  -1600A..1612.5A, in 50mA steps (0x7D00 = 0A)
 //  - Battery Type
```

```
/*****************************************************************************
 // DC Source Status 5
 // Input:
 //  - Instance              DC Instance (bus) ID.
 //  - Device Priority       Relative ranking of DC Source
 //  - DC Voltage            High precision value in 1mV.  Useful for remote instrumentation
 //  - VDC ROC               Rate-of-change (dV/dT) in mV/s  -- 32000 = 0 mV/s


//*****************************************************************************
// DC Source Status 6 - 1FEC7h
// Input:
//  - Instance              DC Instance (bus) ID.
//  - Device Priority       Relative ranking of DC Source
//  - HV Limit Status       Reached upper operational voltage range?
//  - HV Limit Disconnect   Safety disconnect?
//  - LV Limit Status       Reached lower operational voltage range?
//  - LV Limit Disconnect   Safety disconnect?
```

Note: DC Source Status 6 is only received, not transmitted.

```
//*****************************************************************************
// DC Source Status 11 - 1FEA5h
// Input:
//  - Instance              DC Instance (bus) ID.
//  - Device Priority       Relative ranking of DC Source
//  - PwrOnOff              Status of main Battery Switch / Contactor (TRUE = connected)
//  - ChrgOnOff             Status of Charge Bus switch (TRUE = connected)
//  - ChrgDet               Has a charge source been detected?
//  - ResvStat              Is battery running on its 'reserved capacity'?
//  - BatAHCap              Battery capacity in Ah's
//  - DCPower               Watts being received or delivered to/from battery  (as opposed to AMPs in DC_STATUS_1)
```

```
/**************************************************************************
// Charger Status - 1FFC7h
// Input:
//  - Instance
//  - Charge Voltage                0..3212.5v, in 50mV steps
//  - Charge Current                -1600..+1512.5 in 50mA steps (0x7D00 = 0A)
//  - % max current
//  - Operating State               (Bulk, float, etc)
//  - Default PO state
//  - Auto Recharge
//  - Force Charged
*/

/**************************************************************************
// Charger Status2 - 1FEA3
// Input:
//  - Instance              Instance of charger
//  - DC Source Instance    DC Instance (bus) ID associated with
//  - Device Priority       Relative ranking of DC charging Source
//  - DC Voltage            0..3212.5v, in 50mV steps
//  - DC Current            1600..+1512.5 in 50mA steps (0x7D00 = 0A)
//  - Temperature           -40..210 in deg-C, in 1C steps



/**************************************************************************
// Charger Configuration Status - 1FFC6h
// Input:
//  - Instance
//  - Charging Algorithm
//  - Controller Mode
//  - Battery Sensor Present
//  - Charger AC Line          Line 1 or 2 (AC Chargers only)
//  - Battery Type
//  - Battery Bank Size         0..65,530 Ah, 1Ah increments
//  - Maximum charging current  0..250, 1A increments
*/
```

```
/**************************************************************************
// Charger Configuration Status2 - 1FF96h
// Input:
//  - Instance
//  - Max Charge Current %
//  - Max AC current %          Of attached line      (AC Chargers only)
//  - Shore Breaker Size        0..250, 1A increments (AC Chargers only)
//  - Default Batt Temp
//  - Recharge Voltage           0..3212.5v, in 50mV steps
*/



/**************************************************************************
// Charger Configuration Status3 - 1FECCh
// Input:
//  - Instance
//  - Bulk Voltage           0..3212.5v, in 50mV steps
//  - Absorption Voltage     0..3212.5v, in 50mV steps
//  - Float Voltage          0..3212.5v, in 50mV steps
//  - Temp Comp              mV/K
*/


/**************************************************************************
// Charger Configuration Status4 - 1FEBFh
// Input:
//  - Instance
//  - Bulk Time           0..65,530min in 1min steps
//  - Absorption Time     0..65,530min in 1min steps
//  - Float Time          0..65,530min in 1min steps
*/

/**************************************************************************
// Charger Equalization Status - 1FF99h
// Input:
//  - Instance
//  - Time Remaining           0..65,530min in 1min steps
//  - Pre-Charging
*/
```

```
/*******************************************************************************
// Charger Equalization Configuration Status - 1FF98h
// Input:
//  - Instance
//  - Equalization Voltage      0..3212.5v, in 50mV steps
//  - Equalization Time         0..65,530min in 1min steps
*/


/*******************************************************************************
// Terminal - 17E00h
// Input:
//  - Source / Destination
//  - Count                     0..8
//  - Characters                Buffer with up to 8 characters
*/



/*******************************************************************************
// RV-C (ISO) Diagnostics message - 1FECAh
// Input:
//  - On / Off
//  - Active / Standby
//  - DSA                       Default Source Address (RV-C defined DSA)
//  - SPN                       Service Point Number (Fault code appended with 0x7F000)
//  - FMI                       Failure Mode Identifier
//  - Occurrence Count
//  - DSA Extension             Required Sensor Flags
//  - Bank Select
*/
```

J1939 messages:

```
/*****************************************************************************
// J1939 Engine Load - 61443
// Rx:
//  - Engine Load

/*****************************************************************************
// J1939 Engine Speed  - 61444
// Rx:
//  - Engine Speed



/*****************************************************************************
// J1939 Alternator Information message- FED5h (65237)
// Tx:
//  - Alternator Speed
//  - Alternator 1 Status
//  - Alternator 2 Status
//  - Alternator 3 Status
//  - Alternator 4 Status
*/
```

APPENDIX C: SAMPLE YACHT DEVICES CAN-CAN BRIDGE SCRIP

The following scrip has been developed by Yacht Devices to use in their YDNB-07 to perform filtering of CAN messages and only allow the appropriate NMEA2000 messages to pass. Refer to the YDNB-07 uses guide for how to install this script on the YDNB-07.

```
#*******************************************************************************
# P R O G R A M   G E N E R A L   I N F O R M A T I O N
#
# Advanced NMEA 2000 filter
#
# Program Version: 1.01 @ 31.07.2020
#
# (C) Yacht Devices LTD, 2020
# Author: Korolev Alexey
#
#*******************************************************************************
#
# Version control section
#
# 1.01 @ 31.07.2020
# cleanup comments
#
# 1.00 @ 31.07.2020
# initial tests passed
#
# 1.00b @ 28.07.2020
# initial release
#
#*******************************************************************************


#*******************************************************************************
# P R O G R A M   D E S C R I P T I O N
#
# Bridges CAN1 (J1939, RVC, N2K) <---> CAN2 (pure N2K)
#
# need to have the addresses on CAN1 be viewable on CAN2.
#
# CAN1 -> CAN2 PGNs to be forwarded:
#
# 127488, 127501, 127506, 127507, 127508, 127510, 127513, 127750, 127751
#
# CAN2 -> CAN1 PGNs to be forwarded:
#
```

```
# ISO 059904, 127488, 127506, 127508
#
# everything else needs to be blocked.
#
#*****************************************************************************

#*****************************************************************************
# L I C E N S E   I N F O R M A T I O N
#
# This program is free software: you can redistribute it and/or modify
#    it under the terms of the GNU General Public License as published by
#    the Free Software Foundation, either version 3 of the License, or
#    (at your option) any later version.
#
#    This program is distributed in the hope that it will be useful,
#    but WITHOUT ANY WARRANTY; without even the implied warranty of
#    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
#    GNU General Public License for more details.
#
#    You should have received a copy of the GNU General Public License
#    along with this program.  If not, see <https://www.gnu.org/licenses/>.
#
#*****************************************************************************


#*****************************************************************************
# H A R D W A R E   I N F O
#
#----------------------------------------------------------------------------
# Y D N B - 0 7
#
# Target hardware: Yacht Devices YDNB-07 NMEA 2000 bridge
# YDNB-07 hardware revision: any
# YDNB-07 firmware: any
#
# CAN1 speed 250 kbps
# CAN1 connected to NMEA 2000 network segment with J1939, RVC, N2K PGNs
# CAN2 connected to NMEA 2000 network segment with N2K PGNs only
#
#*****************************************************************************



#==================   P R O G R A M   S T A R T   ============================

# D E B U G  S E T T I N G S  S E C T I O N
#
# uncomment the line below to get an MCU assembly code in YDNBSAVE.CFG file
#DECOMPILER=1

# uncomment the line below to record a diagnostics file YDNBLOG.TXT
#DIAGNOSTICS=30

# uncomment the line below to record a diagnostics file in binary format (CAN file)
#LOG_FORMAT=BINARY
```

```
# B R I D G E  H A R D W A R E  S E T T I N G S  S E C T I O N
#
# Disable forwarding on both interfaces
FW_CAN1_TO_CAN2=OFF
FW_CAN2_TO_CAN1=OFF


# I N I T I A L I Z A T I O N
#

#placeholder for init()
init()
{
# --- user-configurable options ------------------------------------------------

# --- user-configurable options ends -------------------------------------------
}

# CAN1 -> CAN2
# 127488
match(CAN1, 0x1F20000, 0x01FFFF00)
{
    send()
}
# 127501
match(CAN1, 0x1F20D00, 0x01FFFF00)
{
    send()
}
# 127506
match(CAN1, 0x1F21200, 0x01FFFF00)
{
    send()
}
# 127507
match(CAN1, 0x1F21300, 0x01FFFF00)
{
    send()
}
# 127508
match(CAN1, 0x1F21400, 0x01FFFF00)
{
    send()
}
# 127510
match(CAN1, 0x1F21600, 0x01FFFF00)
{
    send()
}
# 127513
match(CAN1, 0x1F21900, 0x01FFFF00)
{
    send()
```

```
}
# 127750
match(CAN1, 0x1F30600, 0x01FFFF00)
{
    send()
}
# 127751
match(CAN1, 0x1F30700, 0x01FFFF00)
{
    send()
}

# CAN2 -> CAN1 PGNs to be forwarded:
# 127488
match(CAN2, 0x1F20000, 0x01FFFF00)
{
    send()
}
# 127506
match(CAN2, 0x1F21200, 0x01FFFF00)
{
    send()
}
# 127508
match(CAN2, 0x1F21400, 0x01FFFF00)
{
    send()
}

# ISO 059904
match (CAN2, 0xEA0000,0xFF0000) {
       send()
}

#====================   P R O G R A M   E N D S   ============================
# EOF
```

APPENDIX D: DETAILS OF CPE (CHARGE PROFILE ENTRIES)

The following will give more details on how each parameter impacts battery charging.

```
Typedef struct {                                    // Charging Profile Structure  -- how will we CHARGE a battery?

    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.
    int         PH_AI;                              // Place holder.  FUTURE:  EXIT_ACPT_DVDT  Add dV/dt exit criteria for Acceptance mode,
                                                        need to decide what it is :-)
    float       PH_AF;                              // Place holder.


                                                    // 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.
    int         EXIT_OC_AMPS;                       // Will remain in Overcharge mode holding LIMIT_OC_VOLTS until Amps being delivered falls
                                                        to this level or below.
                                                    //   (Set = 0 to disable this exit check - and exit OC mode imidieatly upon reaching
                                                        EXIT_OC_VOLTS)
    uint32_t    EXIT_OC_DURATION;                   // Over Charge mode duration in mS. Do not exceed this duration in total OC time.   (Set =
                                                        0 to disable max time allowed)
                                                    // ( as a safety step, setting OC_VOLTS or DURATION = 0 will also disable OC mode..)
    int         PH_OI;                              // Place holder.   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.
int         FLOAT_TO_BULK_SOC;                   // If the batteries SOC (as reported by a CAN attached BMS or RBM) drops below this value,
                                                    reg 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.

                                                    // Some batteries allow extended temperature operations but require a lower max energy
                                                        transfer during those times.
                                                    //  Note that these will NOT over-ride the BAT_MIN_CHARGE_TEMP or BAT_MAX_CHARGE_TEMP
                                                        values above, but will allow pull-back as we approach them.
    float         BAT_LOW_RC_VOLTS;                 //  If battery voltage is below this level, cap charging current to BAT_CAP_AMPS value.
                                                        Set = 0.0 to disable low-volts check
    int           BAT_LOW_RC_TEMP;                  // If battery is below this temp (in deg-c), cap charging current to BAT_CAP_AMPS value.
                                                        Set = -99 to disable lower boundary check
    int           BAT_HIGH_RC_TEMP;                 // If battery is above this temp (in deg-c), cap charging current to BAT_CAP_AMPS value.
                                                        Set = -99 to disable upper boundary check
    int           BAT_RC_AMPS;                      // If one of the Reduce Charging triggers above are tripped, cap battery acceptance
                                                        current to this value.   (If no amp shunt, est PWM pullback)  Set = 0 to disable.
                                                    // Note, for safety, this is the ONE condition where the Regulator will over-ride an RBMs
                                                        request battery acceptance current.  Normally we follow the RBM,
                                                    // but if the RBM is not configured correctly - or is just being lazy and not asking for
                                                        reduced current in the case of extreme battery temps, we
                                                    // we will act locally.
  } tCPS;


#define BAT_TEMP_NOMINAL      25                  // Nominal temp which .BAT_TEMP_1C_COMP is based around (in deg-C).
#define BAT_AMPHR_NOMINAL    500                  // CPE's are based on a 500Ah 'nominal' battery size
#define BAT_VOLTS_NOMINAL     12                  //   which is also a '12v' battery
```

Actual content of the CPE tables. Remember, all references are against a ‘normalized’ 12v / 500Ah battery.

```
onst tCPS PROGMEM defaultCPS[MAX_CPES] = {
        //      Bulk/Accpt                              Overcharge                                      Float                                                   Post Float                              Equalization
                                                                                                    Temp Comp       Extended Range
        {14.1f, 6.0*3600000UL, 15, 0,0.0,        0,  0.0 , 0,   0*3600000UL, 0,0.0,       13.4f,  -1, 0*3600000UL, -10,  0, 12.8f, 50,0.0,    0*3600000UL, 0.0, 0, 0,0.0,     0.0f,  0,    0*3600000UL, 0,
                                                                                          0.004f*6, -9, -45, 45,     0.0, -99, -99,  0, 100,      0.0},  // #1 Default (safe) profile & AGM #1 (Low VOltage
                                                                                          AGM).
        {14.8f, 3.0*3600000UL,  5, 0,0.0,        0,  0.0 , 0,   0*3600000UL, 0,0.0,       13.5f,  -1, 0*3600000UL, -10,  0, 12.8f, 50,0.0,    0*3600000UL, 0.0, 0, 0,0.0,     0.0f,  0,    0*3600000UL, 0,
                                                                                          0.005f*6, -9, -45, 45,     0.0, -99, -99,  0, 100,      0.0},  // #2 Standard FLA (e.g. Starter Battery, small
                                                                                          storage)
        {14.6f, 4.5*3600000UL,  5, 0,0.0,        0,  0.0 , 0,   0*3600000UL, 0,0.0,       13.2f,  -1, 0*3600000UL, -10,  0, 12.8f, 50,0.0,    0*3600000UL, 0.0, 0, 0,0.0,    15.3f,  25, 3.0*3600000UL, 0,
                                                                                          0.005f*6, -9, -45, 45,    0.0, -99, -99,  0, 100,     0.0},  // #3 HD FLA (GC, L16, larger)
        {14.7f, 4.5*3600000UL,  5, 0,0.0,        0,  0.0 , 0,   0*3600000UL, 0,0.0,       13.4f,  -1, 0*3600000UL, -10,  0, 12.8f, 50,0.0,    0*3600000UL, 0.0, 0, 0,0.0,     0.0f,  0,    0*3600000UL, 0,
                                                                                          0.004f*6, -9, -45, 45,    0.0, -99, -99,  0, 500,     0.0},  // #4 AGM #2 (Higher Voltage AGM)
        {14.1f, 6.0*3600000UL,  5, 0,0.0,        0,  0.0 , 0,   0*3600000UL, 0,0.0,       13.5f,  -1, 0*3600000UL, -10,  0, 12.8f, 50,0.0,    0*3600000UL, 0.0, 0, 0,0.0,     0.0f,  0,    0*3600000UL, 0,
                                                                                          0.005f*6, -9, -45, 45,    0.0, -99, -99,  0, 100,     0.0},  // #5 GEL
        {14.2f, 0.5*3600000UL, 25, 0,0.0,       30, 14.4f,15, 0.5*3600000UL, 0,0.0,       13.4f,   0, 0*3600000UL,    0,-50, 13.0f,  0,0.0,    0*3600000UL, 0.0, 0, 0,0.0,     0.0f,  0,    0*3600000UL, 0,
                                                                                          0.000f*6,  0,   5, 45,    0.0,   7,  42, 25, 250,     0.0},  // #6 Battleborn
        {14.4f, 6.0*3600000UL, 15, 0,0.0,       15, 15.3f, 0, 3.0*3600000UL, 0,0.0,       13.1f,  -1, 0*3600000UL, -10,  0, 12.8f, 50,0.0,    0*3600000UL, 0.0, 0, 0,0.0,    15.3f,  25, 3.0*3600000UL, 0,
                                                                                          0.005f*6, -9, -45, 45,    0.0, -99, -99,  0, 100,     0.0},  // #7 4-stage HD LFA (& Custom #1 changeable profile)
        {14.2f,             0,  0, 0,0.0,        0,  0.0 , 0,   0*3600000UL, 0,0.0,        0.0 ,   0, 0*3600000UL,    0,-50, 13.0f, 70,0.0,    0*3600000UL, 0.0, 0, 0,0.0,     0.0f,  0,    0*3600000UL, 0,
                                                                                          0.000f*6,  0,   0, 50,    0.0,   5,  45, 25, 200,     0.0}   // #8 LiFeP04 (& Custom #2 changeable profile)
```

```
                // #1 Default (safe) profile & AGM #1 (Low VOltage AGM).
                // #2 Standard FLA (e.g. Starter Battery, small storage)
                // #3 HD FLA (GC, L16, larger)
                // #4 AGM #2 (Higher Voltage AGM)
                // #5 GEL
                // #6 Battle Born
                // #7 4-stage HD LFA (+ Custom #1 changeable profile)
                // #8 LiFeP04        (+ Custom #2 changeable profile)
        };
```

Appendix E: 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 WS500 Alternator Regulator is unable to decipher 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 (example, error low battery voltage).

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.

Other error codes are related to the overall system operation. Examples are the Battery Disconnected faults, where the regulator receives notification that the BMS has (or will soon) disconnect the battery from charging sources to protect it. Any overvoltage or disconnect error must be investigated carefully to determine the cause of the overvoltage condition and corrected. (Likely miss-configuration, though perhaps incorrect auto-detect of system voltage)

FET over temperature (error #41) 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)
```

```
              //----    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)
              //              Errors with a +0x4000 will be optionally treated as a 'promiscuous' hard fault.
              //               If the regulator is configured in 'promiscuous Mode'
              //               these faults will be treated with an auto-restart, just like  0x8000
```

```
#define FC_LOOP_BAT_TEMP                12  + 0x4000U   // Battery Temperature greatly exceeded configured upper limit.
#define FC_LOOP_BAT_HIGHV               13  + 0x4000U   // Battery Voltage greatly exceeded upper limit, measured by VBat+
#define FC_LOOP_BAT_LOWV                14  + 0x8000U   // Battery Voltage too low to operate as measured on VBat+
                                                        // Damaged or missing sensing wire or fuse? (or engine not started!)
#define FC_LOOP_BAT_MAXV                15  + 0x4000U   // Voltage at Vbat+ exceeded Max Bat Volts as defined by $CPB:
#define FC_LOOP_SHORTED_BAT_TEMP        16  + 0x4000U   // Battery Temperature is shorted (Defective)


#define FC_LOOP_ALT_TEMP                21  + 0x4000U   // Alternator Temperature greatly exceeded configured upper limit.
#define FC_LOOP_ALT_TEMP_RAMP           24  + 0x4000U   // Alternator Temperature greatly exceeded configured upper limit.
                                                        // (2nd temp reached / exceeded while ramping - this can NOT be right, to reach
                                                           target while ramping means way too risky.)

#define INTERNAL ERROR                  31..39


#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_NO_VALT_VOLTAGE              43 + 0x8000U    // No voltage has been sensed on the VAlt+ line, blown fuse?
#define FC_EXCESSIVE_VALT_OFFSET        44 + 0x4000U    // There is excessive voltage offset between VAlt+ and VBat+ sense lines - 2.5v.
                                                           (Not checked in 'Split' voltage systems)
#define FC_LOOP_VALT_MAXV               45  + 0x4000U   // Voltage at VAlt+ exceeded Max Bat Volts (Plus additional allowance for IR
                                                           drop) as defined by $CPB:
#define FC_LOOP_ALT_HIGHV               46 + 0x4000U    // Voltage greatly exceeded expected upper limit battery limit as measured at
                                                           VAlt+

#define FC_CAN_BATTERY_DISCONNECTED     51 + 0x4000U    // Received a generic CAN message that the battery charging bus has been
                                                           disconnected.
#define FC_CAN_BATTERY_HVL_DISCONNECTED 52 + 0x4000U    // A CAN command has been received asking for the battery bus to be disconnected
                                                           due to High Voltage.
                                                        // (Note that depending on the BMS, other alarms may trigger this same fault,
                                                           ala, high charge current)
#define FC_LOG_BATTINST                 53              // Battery Instance number is out of range (needs to be from 1..100)
#define FC_TOO_MANY_AGGERGATION         54              // Too many different BMS's are asking to be aggregated.
#define FC_CAN_AEBUS_FAULTED            55 + 0x8000U    // AEBus device (Discovery battery) has send a warning or fault status.
                                                        // As there is no fore-warning of a disconnect,
                                                        // treat all warnings as a pending disconnect and fault.  But then do auto-
                                                           restart to see if it clears.
#define FC_TOO_MANY_VEREG_DEVICE        56              // Too many VEreg (Victron) devices present to track
#define FC_CAN_BATTERY_LVL_DISCONNECTED 57 + 0x4000U    // A CAN command has been received asking for the battery bus to be disconnected
                                                           due to Low Voltage.
```

```
#define FC_CAN_BATTERY_HC_DISCONNECTED  58 + 0x4000U    // A CAN command has been received asking for the battery bus to be disconnected
                                                           due to High Current.
#define FC_CAN_BATTERY_HT_DISCONNECTED  59 + 0x4000U    // A CAN command has been received asking for the battery bus to be disconnected
                                                           due to High Battery Temperature.


#define FC_CAN_BATTERY_LT_DISCONNECTED  61 + 0x4000U    // A CAN command has been received asking for the battery bus to be disconnected
                                                           due to Low Battery Temperature.
#define FC_CAN_BATTERY_HVL_LIMIT        62 + 0x8000U    // A CAN status has been received that the battery has reached its upper limit,
                                                           but not yet disconnecting.  Charging should stop.

#define INTERNAL ERROR                  71..79
#define INTERNAL_DCDC_ERRORS            81..89


//--  The 9x codes are special ones, they do not cause a true FAULT, but indicate some altered condition and mode of operation.
//      Mostly these are used to support DM_RV (aka, ISO Diag) and CAN connected monitors such as the Victron Cerbo.
//      At present these codes will only be used over ISO_DIAG (CAN), will NOT cause a Regulator hard fault, and will only be issued
//      if the attached RBM is listed as a BMS or higher priority per RV-C (120 or above).
#define FC_CAN_BMS_SYNC_LOST            91              // Used to signal that an existing BMS sync has been lost and we are in an alt
                                                           mode (ala, Gethome)
#define FC_FORCED_TO_IDLE               92              // Use to indicate that the reg has been forced into Idle via perhaps the
                                                           Feature-in line, or some RPM based trigger.



#define INTERNAL ERROR                  1xx


#define FC_DCDC_HS_OVP                  212  + 0x4000U   // Primary Battery (HS) of DC-DC converter Over-voltage trip
#define FC_DCDC_HS_UVP                  213  + 0x8000U   // Primary Battery (HS) of DC-DC converter Under-voltage trip
#define FC_DCDC_LS_OVP                  214  + 0x4000U   // Secondary Battery (LS) of DC-DC converter Over-voltage trip
#define FC_DCDC_LS_UVP                  215  + 0x8000U   // Secondary Battery (LS) of DC-DC converter Under-voltage trip
#define FC_DCDC_OVER_TEMP               216  + 0x8000U   // DCDC Convert too hot.
#define FC_DCDC_MISCOFIG                217              // A configuration value has exceeded the selected DC-DC converter limit
#define FC_DCDC_MULTICONTROLLER         218              // More than one device seems to be trying to control the DCDC converter.
```