GENERALIZED ELECTRIC VEHICLE CONTROL UNIT GEVCU SPECIFICATION

Jack Rickard jack@evtv.me

BACKGROUND

For 40 years and more, individual tinkerers and innovators have been modifying existing automobiles to electric drive and often building electrically powered vehicles from scratch.

The early "controllers" that evolved to replace simple switching and resistive controls to control the speed of the driver motors were mostly pulse width modulated (PWM) devices to provide an averaged DC signal to a series DC motor. These simple chopper voltage control devices were generally referred to as "controllers" and translated driver input from "controls" such as the accelerator pedal, ignition switch, and brake to this motor driving voltage to control its speed and direction – and consequently the vehicle.

There are of course many other types of motors such as separately excited DC motors, brushless DC motors, permanent magnet motors and AC induction motors. Most of these polyphase motors required 3 phase "inverters" to convert the DC power of the battery pack into three phase drive signals varying the voltage and frequency of the power to the motor to accomplish the same thing.

Most of the DC "controllers" received the inputs directly by wiring from the sensors or controls in the cars.

In recent years, among automobile manufacturers, the use of the Controller Area Network bus protocol has been adopted for many of the items in the automobile, including the internal combustion engine – often called an Engine Control Unit or a Vehicle Control Unit which forms the central "brains" of the car. It is connected to various sensors and controls by wire but communicates to other subsystems of the vehicle such as the instrumentation, ABS system, transmission, environmental system using this CANbus as the common link.

Automakers have eschewed the DC motor in favor of either permanent magnet AC motors or AC induction motors. And the “inverters” developed to drive these were interfaced to this same CANbus model for control input.

Because of the varying nature of the cars, the ECU or VCU would be specifically designed for that vehicle. But the Inverters driving AC motors can be somewhat generalized to work in any car by using CANbus to control them.

The VCU would be specifically designed and software specifically written for that vehicle. This was hardcoded into flash memory and defined the operation of the vehicle. No end user input or options were provided. Firmware updates were accomplished by revising the hard code, recompiling, and having the controller reflashed with the new binaries at the dealership during normal maintenance.

And so thousands of these VCUs could be flashed when built, with the software specific to that particular vehicle.

The VCU would of course be completely inappropriate and non functional in another vehicle make or model.

As many automakers are experimenting with product introductions of plug-in electric vehicles and hybrid gasoline/electric vehicles, many of the components used in these vehicles are becoming available when the cars are salvaged and are recycled through salvage yards and such online services as eBay.

As such, they will become a resource to individual tinkerers and those converting existing cars to electric drive.

But almost all of these components were designed to be controlled by the Vehicle Control Unit that came with the original car.

It would be a serious advantage, to have a more generalized vehicle control unit that allowed some basic configuration allowing these conversions to modify operation of the VCU to accommodate THEIR car without the need to entirely rewrite the software and flash the VCU.

As almost all VCU accomplish the same basic functions, a simple configuration input that does not involve assembly language or C programming to change it would be of substantial utility.

This document describes the design of a Generalized Vehicle Control Unit. The multicontroller hardware will allow connection of the basic sensor set necessary to drive the car such as throttle signals, brake signals, ignition signal, and control the basic common outputs such as brake lights, fuel level, rpm, power usage, while serving in the central role of converting these inputs to CAN messages for the inverter to actually drive the car.

The operation of this device can be modified, within fairly narrow constraints, by a simple configuration "menu" style input that non-programmers can access and make changes to, in order to interface the VCU to the particular car they want to convert. The initial immediate need for this Generalized Vehicle Control Unit was the acquisition of a significant number of Siemens 1PV5135-4WS14-Z motors and Digital Motor Control (DMOC) 645 units from the liquidation of Azure Dynamics Corporation by EVtv.

This DMOC no longer has control inputs, but is rather designed to respond to CANbus commands solely.

The intent in developing this GEVCU, is to not only allow access to a number of OEM hardware components, but also to transfer the product support function faced by the manufacturers and developers of the Inverter Power Electronics to the GEVCU.

In theory, CANbus message digests for ANY motor inverter could be easily patched into the GEVCU to allow operation of that power inverter without ANY support from the developer of the power electronics.

This should quickly allow the developers of power electronics inverters to provide this equipment to individuals at the same or comparable prices as they do to the OEM automakers, without having to incur additional expense to train and support those users in the use of the product. Training and use becomes a function of using the GEVCU itself.

HARDWARE DESIGN PHILOSOPHY

GEVCU developers can use any hardware they can build or locate that accommodates the basic sensor inputs and instrumentation outputs described in this document. OPEN SOURCE hardware prototype equipment is preferred for GEVCU development and the software should be designed to operate on an open source prototype system. But it is perfectly acceptable to then design custom hardware from that reference design to reduce costs and eliminate extraneous unused components available in the open source prototype system design.

The hardware must provide:

1. At least one CANbus port capable of CAN 1.0 and 2.0b operation, including both 11 bit and 29 bit identifiers.

2. A sufficient number of analog to digital converter inputs of sufficient resolution to accommodate the sensor inputs described in this document. Vehicle sensors are overwhelmingly based on 0-5V architectures. And so these A/D inputs must accept signals in the 0-5v range, and the device must be able to provide 5v

supplies to those sensors in sufficient power levels to not alter the 5v reference level when all such sensors are connected.

3. Digital inputs capable of recognizing on/off signals from the vehicle given in 12v and ground states. Active can be either of active high 12v or active low gnd but active high is the normally preferred method as 12v is normally the enabling signal for other systems such as brake lights.

4. Relay driver outputs. Digital outputs suitable for engaging relays, contactors, and other relay like devices in the vehicle.

5. PWM outputs. Sufficient PWM outputs to drive legacy instrumentation such as tachometers and fuel gages as described in this document.

6. ETHERNET port. It is very important for the end user to be able to modify the operation of this device (configure it) without the need for unique external software programs running on specific operating systems. The user should be able to easily configure this device using any device with an Ethernet connection and an http browser. It is actually completely acceptable to also provide means of configuration using a browser via 802.b/g/n wireless, Bluetooth, or Universal Serial Bus (USB) interfaces.

   The overall requirement here is that no special type of computer, operation system, or unique configuration software external to the GEVCU can be required to easily modify GEVCU configuration and operation.

   Additionally we seek to AVOID the additional expense in the device of a dedicated display and input devices in the GEVCU hardware itself. Once configured, this hardware would be entirely redundant. Configuration is conceivably a one time or perhaps very infrequent event.

7. Processor power sufficient to perform the GEVCU tasks without delay in vehicle operation or control. This is quite important as many systems shut down without regular updated CAN messages to keep them alive and the results of any inability to make schedule could be catastrophic in a moving vehicle.

8. Popularity and user base. When selecting hardware, recognize the advantage of popular and widely adopted devices with large user base. That user base provides a level of additional support and utility that it forms a serious element of the ultimate value of the GEVCU.

9. Tool chain. Strong preference should be given to devices with simple and free compilers such as the Arduino IDE. An open hardware open software device makes little sense if it requires a $2800 compiler environment to make code changes.

SOFTWARE DESIGN PHILOSOPHY

The basic software design philosophy is to be an open source, object oriented software design. This is to accommodate a couple of needs for the GEVCU

1. Ability to add contributions from any external source or user that prove effective in practice. Anyone should be able to read, modify, recompile and run the original source code to the GEVCU. As such, it should be open source and use inexpensive or free, widely available tool chains to compile. If they are caught in the act of doing something useful, that improvement can be added to the main trunk of the software development.

2. Ability to add support for additional inverters. Almost all of the available inverters accomplish essentially the same functions and provide essentially the same information. But the specific CAN bus IDs, command structure and data formats vary entirely from one inverter to another.

   The GEVCU will be initially developed to support the Azure Dynamics DMOC645 inverter. But the ability to add the specific CAN bus message commands for other inverters, without having to make changes to the main program structure in a modular fashion is very important.

   Make a strong effort to isolate the inverter object from the rest of the code, with an eye towards allowing the use of other inverter objects with no changes to the non-inverter object code.

3. Object Oriented C++ code is an obvious choice. Assembly language subroutines where desirable or advantageous are perfectly acceptable.

4. Recognize an inevitable desire to add additional functions to this device quite beyond the initial GEVCU spec. The GEVCU, by virtue of it's central role in vehicle operation, would be the natural place to go for controlling electrically powered air conditioner compressors, CANbus controlled battery chargers, CANbus controlled Electronic Power Assisted Steering units (EPAS) etc.

COMPLIANCE

For teams developing GEVCU hardware and/or software, it should be noted that compliance with this specification is ENTIRELY voluntary.

That said, there is a slight motivator you might want to take into consideration. I happen to know that software developers are notoriously averse to actually documenting anything and hardware designers are even worse – they not only don't document but they can't remember either. They'll have to get with Charlie and get back to you later.

On the other hand, if you design TO the GEVCU specification, and when making necessary changes NOT in compliance get those changes incorporated BACK into the spec, you will find an odd resemblance between your newly finished product and my upcoming best selling novel :

INSTALLATION AND OPERATION OF THE GENERALIZED ELECTRIC VEHICLE CONTROL UNIT.

By Jack Rickard.

See the movie at a theatre near you. Coming this summer.

PART 1 - INPUTS

THROTTLE

The throttle or accelerator is undoubtedly the central control input for a controller. This is how we vary the power output of the inverter to the motor, and thus motor speed and torque, and consequently vehicle speed.

There are two broad types of throttles that must be accommodated: legacy throttles and modern dual signal drive by wire throttles found in currently manufactured vehicles.

The legacy throttle most widely used is a simple 5kohm potentiometer linked to a mechanical pedal usually by wire. The controller provides 5v and a reference ground to the two ends of the resistor, and the pickup arm of the potentiometer returns some fraction of that source voltage to the controller as the signal input.

There are also single channel hall-effect throttles available on the market. But their operation is largely identical. 5v and ref gnd is provided to them, and they return some value between 0 and 5v based on position.

Modern drive by wire throttles used in vehicles today use the same 5v and ref gnd input, but they will provide TWO signal inputs to the controller. These are often two signals of different scale: ie. 0.5 to 4.5 and 1.2 to 3.2. Or they might be inverse function where one signal is 1-4v while the other is 4-1v. In some cases they can be both differing in scale and direction.

These throttles are generally designed for safety in that the controller is suppose to compare the two signals, which have a known relationship, and if the two signals do not correspond, the assumption is that one or both are faulty. As a result, the controller reduces all motor torque and speed commands to zero. This prevents a runaway situation resulting from loss of ground, or loss of 5v, or other wiring incident or component failure.

PINS:

1. 5vdc supply voltage 50 ma out to sensor
2. Signal 0-5vdc ADC input ADC
3. Optional sanity check voltage ADC input
3. Reference return – ground out to sensor

CONFIGURATION VARIABLES:

THROTMIN: This parameter sets the minimum voltage where the throttle is considered active. Voltages below this value are read as no throttle input.

Example: 0.80 would indicate 0.80v

THROTMAX: This parameter sets the maximum voltage where the throttle is considered active. Voltages above this value are read as no throttle input.
Example: 4.75 would indicate 4.75v

THROTNOTMIN: This parameter sets the minimum voltage where the optional sanity check throttle input is considered active. Voltages below this value are read as no throttle input.

Example: 1.20 would indicate 1.20v

THROTNOTMAX: This parameter sets the maximum voltage where the optional sanity check throttle input is considered active. Voltages above this value are read as no throttle input.

Example: 3.60 would indicate 3.60v

The sanity check is NOT performed if either **THROTNOTMIN** or **THROTNOTMAX** is zero. If they are non zero, the percentage of maximum throttle signal between **THROTNOTMIN** and **THROTNOTMAX** is calculated. Also the percentage of maximium throttle signal between **THROTMIN** and **THROTMAX** is calculated. The two percentages are compared, and if not within 5% of each other, a fault is thrown and CANbus signals generated to put inverter at zero torque and zerospeed.

Note that this calculation MUST handle the case where **THROTNOTMIN** is a larger positive voltage value than **THROTNOTMAX** – the inverse case.

THROTREGEN: Voltages between this value and **THROTMIN** result in a linear increase in regenerative braking, with maximum regenerative braking established by **THROTMAXREGEN** at the **THROTMIN** voltage and essentially 0 percent of **MAXREGEN** at the **THROTREGEN** point.

Example: 1.50 indicates 1.50v

Zero regenerative braking at 1.50 v increasing to **THROTMAXREGEN** at **THROTMIN**.

In this way, as you begin easing off the throttle through the deadband, at **THROTREGEN** regenerative braking is still zero. As the voltage falls, regenerative braking INCREASES to the point set by **THROTMAXREGEN** when the voltage falls to **THROTMIN**. And so the further we remove the throttle, the greater the amount of regenerative torque is applied.

THROTFWD: Typically a small voltage value above **THROTREGEN** establishing a "dead band" between the **THROTREGEN** point and the **THROTFWD** point.
Example: 1.8v

In this example, from **THROTREGEN** (1.50v_ to **THROTFWD** (1.8v) is the deadband or coast zone. NO torque or speed commands are provided to the inverter if throttle input is both valid and in this band, allowing the motor and the vehicle to free roll.

The area between **THROTFWD** and **THROTMAX** defines the region of forward acceleration with the minimum torque/speed values occurring at **THROTFWD** and the maximum torque and speed values at **THROTMAX**.

THROTMAP: This variable represents a voltage between and **THROTMAX** where a 50% torque and speed level is defined.

A linear increase of torque and speed is mapped between **THROTFWD** and **THROTMAP** with 50% of available torque and speed occurring at **THROTMAP**.

A linear increase in torque and speed is also mapped between **THROTMAP** and **THROT MAX** representing the remaining 50% of available torque.

This entered value is then used to warp the throttle curve. For example:

THROTFWD: 1.80v
THROTMAP: 3.80v
THROTMAX: 4.65v

It can be seen that the first 50% of available torque is mapped between 1.80 and 3.80v while the second 50% of available torque is mapped linearly between 3.80v and 4.65v. In this way, low speed handling and maneuvering has greater resolution spread across 2.00v of the throttle range, while less resolution is provided across the upper 0.85v. Higher resolution is desired at take off, or while parking, or reversing, while it is generally not needed in going from 50% to full power when we are basically just wanting a big increase in power. **THROTMAP** can be used to slide this point between those two extremes.

THROTMAXREGEN: This variable defines the percentage of available maximum torque that is available for regenerative braking at the **THROTMIN** point.

Note that a value of 0 indicates no regenerative braking at all using the throttle.

MAXREGEN: This variable defines the percentage of available maximum torque that is ever available for regenerative braking.

MAXTORQ: This defines the maximum torque available for any purpose.

BRAKE

You might note that the throttle input also has provisions for regenerative braking. The brake signal does as well. We would assume that the vehicle has a fully functional normal brake system, and that it also has a brake switch used to turn on the brake lights. Regenerative braking is normally a function of the brake input.

PINS:

1. 5vdc supply voltage 50 ma out to sensor
2. Brake pressure sensor signal 0-5vdc input ADC
3. Reference return – ground out to sensor
4. Brake light input – 0 or 12v digital input

CONFIGURATION VARIABLES:

BRAKEMIN: This parameter sets the minimum voltage where the brake sensor is considered active. Voltages below this value are read as no brake input.

BRAKEMAX: The maximum voltage logic will read as valid brake pressure sensor input. Voltages above **BRAKEMAX** are read as no brake input.

BRAKEMAXREGEN. This is a value from 0 to 100 representing the percentage of **MAXTORQ** that can be applied during regenerative braking.

The actual value of regenerative braking during a braking event is a linear value between **MAXTHROTREGEN** at **BRAKEMIN** and **BRAKEMAXREGEN a**t **BRAKE MAX**.

In this way, when you remove your foot from the throttle, the regenerative level is set at the value of **MAXTHROTREGEN**. When your foot first applies to the brake, that SAME level of REGEN is maintained. As more brake pressure is applied, the regenerative braking torque increases until it reaches **BRAKEMAXREGEN** at the **BRAKEMAX** point.

Note that by setting **BRAKEMAX** to some voltage LESS than the actual maximum voltage provided by the sensor, you can slide the regenerative braking zone toward the very lightest application of brake, and as you continue to press the pedal, regenerative braking ceases and the normal brake system takes over. This can be effective to get maxium regenerative braking effect before the normal disk or drum brakes begin to become effective.

Normally, NO regenerative braking will be provided without a secondary signal from the brake light input. This is a secondary input from a separate source used to enable regenerative braking.

BRAKREGENRAMPTIME. This is a value between 0 and 255 representing the number of hundredths of a second to use to ramp from **THROTMAXREGEN** to **BRAKEMAXREGEN** if the brake sensor input value is below **BRAKEMIN** but the brake light input is active.

In this event, we assume we are NOT USING a brake pressure tranducer input at all. But we do have an active brake light switch signal. The regenerative braking will gradually increase from **THROTMAXREGEN** to **BRAKEMAXREGEN** in linear fashion across the entered time value with a max time of 2.5seconds.

This is counterintuitive and we've actually seen inverters with this function coded in backwards. But it doesn't work well when braking.

In the event we are not using regen on the throttle, and want to use it on the brakes, but have no transducer, the regenerative braking will gradually be applied when we step on the brake pedal and steadily increase. If we get too much braking, we simply remove our foot from the brake. Regen goes to zero. We can REAPPLY the brake pedal and again regenerative braking starts to increase. In practice, this feels like you are pumping the brakes.

REGENERATIVE BRAKING ADJUST

Regenerative braking is a highly subjective "feel" issue. It would be very nice to provide a knob on an electric vehicle where the driver could manually adjust the regen level just by turning the knob.

PINS:

1. 5vdc supply voltage 50 ma out to sensor
2. Regenerative brake adjust signal 0-5vdc input ADC
3. Reference return – ground out to sensor

CONFIGURATION VARIABLES:

REGENSCALE. This variable is set between 0 and 100% based on the voltage read on the regenerative braking adjust signal input. Voltages below 1v set this value to 0. Voltages between 1 and 4v are converted to a decimal value less than 1 indicating the percentage of regen desired. Voltages above 4 are read as 4.

THROTMAXREGEN and **BRAKEMAXREGEN** are scaled by multiplying the values entered on the configuration screen by **REGENSCALE**.

As a result, potentiometer settings resulting in an input voltage less than 1 effectively disables regenerative braking. Voltages between 1 and 5 scale regenerative braking.

IGNITION

The switched ignition 12v from the vehicle is used to turn on the GEVCU. It is also the source of power for the GEVCU.

PINS: 1. 12vdc supply voltage in from vehicle ignition switch.

CONFIGURATION VARIABLES: None.

FORWARD

The internal **DIRECTION** variable should be used to indicate forward (1), reverse (2), or neutral (0) operation. IF **REVERSE** and **NEUTRAL** input pins are inactive, set **DIRECTION** to 1.

PINS: 1. 12vdc digital input.

CONFIGURATION VARIABLES:

ACTIVEHIGH. This single configuration variable determines if input pins are active high (greater than 4 volts) or active low (less than 3 volts). All digital input pins should be able to endure voltage values up to 15v which may occur in vehicles. This variable applies to ALL digital input pins on the VCU.

REVERSE

The internal **DIRECTION** variable should be used to indicate forward (1), reverse (2), or neutral (0) operation. IF **FORWARD** and **NEUTRAL** input pins are inactive, set **DIRECTION** to 2.

PINS: 1. 12vdc digital input.

CONFIGURATION VARIABLES:

ACTIVEHIGH. This single configuration variable determines if input pins are active high (greater than 4 volts) or active low (less than 3 volts). All digital input pins should be able

to endure voltage values up to 15v which may occur in vehicles. This variable applies to ALL digital input pins on the VCU.

NEUTRAL

The internal **DIRECTION** variable should be used to indicate forward (1), reverse (2), or neutral (0) operation. IF **FORWARD** and **REVERSE** input pins are inactive, set **DIRECTION** to 0.

PINS: 1. 12vdc digital input.

CONFIGURATION VARIABLES:

ACTIVEHIGH. This single configuration variable determines if input pins are active high (greater than 4 volts) or active low (less than 3 volts). All digital input pins should be able to endure voltage values up to 15v which may occur in vehicles. This variable applies to ALL digital input pins on the VCU.

GENERAL ENABLE

This input can be used to enable the VCU operation. If active, AND the **GENERAL DISABLE** input is INACTIVE, sets internal variable **ENABLE** to 1. This internal variable is used to determine whether to allow drive speed and torque commands to the inverter. If this input goes inactive, the **ENABLE** variable is set to 0.

PINS: 1. 12vdc digital input.

CONFIGURATION VARIABLES:

ACTIVEHIGH. This single configuration variable determines if input pins are active high (greater than 4 volts) or active low (less than 3 volts). All digital input pins should be able to endure voltage values up to 15v which may occur in vehicles. This variable applies to ALL digital input pins on the VCU.

GENERAL DISABLE

This input can be used to enable the VCU operation. If any of these three input pins go active, sets internal variable **ENABLE** to 0 regardless of **GENERAL ENABLE** input. This internal variable is used to determine whether to allow drive speed and torque commands to the inverter. If this input goes inactive, NO change is made to the **ENABLE** variable and we rely on the general enable input to reset **ENABLE** to 1.

PINS:
1. 12vdc digital input – DISABLE 1.
2. 12vdc digital input - DISABLE 2
3. 12vdc digital input - DISABLE 3

CONFIGURATION VARIABLES:

ACTIVEHIGH. This single configuration variable determines if input pins are active high (greater than 4 volts) or active low (less than 3 volts). All digital input pins should be able to endure voltage values up to 15v which may occur in vehicles. This variable applies to ALL digital input pins on the VCU.

LIMP MODE

This input is used to force an immediate reduction in power available via speed and torque command. All speed and torque commands are scaled by the **LIMPSCALE** variable.

PINS:
1. 12vdc digital input.

CONFIGURATION VARIABLES:

ACTIVEHIGH. This single configuration variable determines if input pins are active high (greater than 4 volts) or active low (less than 3 volts). All digital input pins should be able to endure voltage values up to 15v which may occur in vehicles. This variable applies to ALL digital input pins on the VCU.

LIMPSCALE. 0-100. This configuration variable is used as a multiplier for ALL torque and speed commands to the vehicle. The default value is 100 indicating full power

Scotty. Lower values of course reduce output power to the percentage indicated in this variable. The variable is applied ONLY when the **LIMPMODE** input is active.

OUTPUTS

CONTROLLER STATUS

This output is used to indicate status of the GEVCU. If it is turned ON and left on in a steady state, this indicates the GEVCU has powered up, established communications with the inverter, and passed all internal operational checks.

PINS: 1. Digital output LED driver.

CONFIGURATION VARIABLES: None

If this LED is flashed 1 to 9 times, followed by off for two seconds, followed by another series of flashes 1 to 9, this indicates a fault code. Fault codes can be, 11 through 19, 21 through 29, 31 through 39 etc. up to 99.

Fault Code 99 is defined as FAULT - NO FAULT CODE PROVIDED.
Fault Code 11 is defined as NO INVERTER COMMUNICATIONS.

Any developer can register a fault code and we'll add it. First come, first served.

PRECHARGE RELAY OUTPUT

This output is used to close a precharge relay, applying high voltage to the Inverter through a precharge resistor. icates the GEVCU has powered up, established communications with the inverter, and passed all internal operational checks.

PINS: 1. Digital output 12v RELAY driver. 500ma

CONFIGURATION VARIABLES:

PRECHARGERELAY: This variable is set by the user using a checkbox or radio button. The variable is set to 1 if selected and 0 if not indicating the need for a precharge relay output on power up.

If 1, a digital output is provided as part of the startup sequence. If the digital output does not provide sufficient current, an external MOSFET may be required to drive the relay.

Many inverters such as Rinehart do their own precharge management. The DMOC 645 does not. So this will always be an optional output configurable by the user.

CONTACTOR RELAY OUTPUT

This output is used to close a contactor high current relay, applying high voltage to the Inverter directly. Most contactors can experience an initial closing current requirement of up to 1.5 amps and subsequently settle to 300 ma or so.

PINS: 1. Digital output 12v RELAY driver. 1.5 amps.

CONFIGURATION VARIABLES:

CONTACTOR: This variable is set by the user using a checkbox or radio button. The variable is set to 1 if selected and 0 if not indicating the need for a contactor relay output on power up.

If 1, a digital output is provided as part of the startup 3 seconds after the **PRECHARGERELAY** output. If the digital output does not provide sufficient current, an external MOSFET may be required to drive the contactor.

REVERSE LIGHT RELAY OUTPUT

This output is used to close a relay, applying 12v to the reverse lights of the vehicle. This will normally be done by the transmission, but on certain vehicles without transmission, could be reversed using the **REVERSE** input to reverse the motor speed and torque commands. This output is always available and set active whenever the internal **DIRECTION** variable is set to 0

PINS: 1. Digital output 12v RELAY driver. 500 ma.

CONFIGURATION VARIABLES: None

If the digital output does not provide sufficient current, an external MOSFET may be required to drive the 12v relay

COOLING RELAY OUTPUT

This output is used to close a relay, applying 12v to drive cooling pumps or fans or both. This is set to active whenever CANbus status signals are received from the inverter indicating a motor or inverter temperature reaching a certain level.

PINS: 1. Digital output 12v RELAY driver. 500 ma.

CONFIGURATION VARIABLES:

COOLINGTEMP: 0-300

This configuration variable allows the user to enter the temperature of either motor or inverter where the coolant output goes active. Zero to 300C.

MOTOR TEMPERATURE OUTPUT

This is a Pulse Width Modulated output representing motor temperature. It can be used to drive a temperature gage. It will be active only if motor temperature is available from inverter via CAN status message.

PINS: 1. 5v PWM output.

CONFIGURATION VARIABLES:

MOTORMINTEMP: 0-300

This variable allows user to enter a motor temperature corresponding to the lowest reading on their gage and nominally a PWM value of 0%.

MOTORMAXTEMP: 0-300

This variable allows the user to enter a motor temperature in degrees centigrade to correspond with a PWM output of 100%.

INVERTER TEMPERATURE OUTPUT

This is a Pulse Width Modulated output representing inverter temperature. It can be used to drive a temperature gage. It will be active only if inverter temperature is available from inverter via CAN status message.

PINS: 1. 5v PWM output.

CONFIGURATION VARIABLES:

INVMINTEMP: 0-300
This variable allows user to enter an inverter temperature corresponding to a PWM value of 0%.

INVMAXTEMP: 0-300
This variable allows the user to enter an inverter temperature in degrees centigrade to correspond with a PWM output of 100%.

Output should of course be scaled from **INVMINTEMP** to **INVMAXTEMP** 0 to 100%. Temperatures above max will be shown at 100% and temperatures below min will be shown at 0% but of course.

RPM OUTPUT

This is a pulsed digital output that can be used with a 12v opto isolator to drive a common tachometer. RPM from Inverter CAN status message is converted to a series of digital pulses..

PINS: 1. pulsed digital output.

CONFIGURATION VARIABLES:

TACHPPT:
This variable allows user to enter the number of pulses to output per turn PPT corresponding to their tachometer. The output is scaled to a frequency representing this number of pulses per rpm.

Example: 6000 rpm = 100 rps x 4ppt = 400Hz

BATTERY CURRENT OUTPUT

This is a pulsed digital output that can be used with a 12v opto isolator to drive a common tachometer. Battery current from Inverter CAN status message is converted to a series of digital pulses..

PINS: 1. pulsed digital output.

CONFIGURATION VARIABLES:

AMPPPT:
This variable allows user to enter the number of pulses to output per turn PPT corresponding to their tachometer. The output is scaled to a frequency representing this number of pulses per rpm.

Example: 600 amp = 100 rps x 4ppt = 400Hz

In this case 600 amps of battery current will appear on a tachometer as 6000 rpm if it is a 4 pulse per turn tachometer.

OTHER CONFIGURATION VARIABLES

MAX RPM – maximum RPM controller should command to inverter.

MAX TORQUE – maximum torque controller should command to inverter.

MAX REGEN – maximum regenerative braking torque controller should command.

GEVCU STATUS DISPLAY HTML

Battery voltage _____v

Battery current _____A

Battery power _____kW

Motor temp: _____C

Inverter temp: _____ C

Motor Torque: _____ %

Motor Speed: _____rpm

Power used since start: _____kWh

BUTTON – GEVCU SETUP

GEVCU CONFIGURATION

GENERAL LIMITS

MAXRPM (1000-10000) _____rpm

MAXTORQUE (0-100): _____%

ACTIVEHIGH [ ] yes [ ] no

LIMPSCALE(0-100) ____*%*

THROTTLE/ACCELERATOR

THROTMIN (0.00-5.00v*) :* _____*v*

THROTREGEN (0.00-5.00)*:* _____*v*

THROTFWD(0.00-5.00): _____*v*

THROTMAP(0.00-5.00): _____*v*

THROTMAX(0.00-5.00): _____*v*

THROTNOTMIN(0.00-5.00): _____*v*

THROTNOTMAX(0.00-5.00): _____*v*

0.00==THROTMIN==THROTREGEN==THROTFWD==THROTMAP=THROTMAX ==5.00

0.00==THROTNOTMIN==========THROTNOTMAX-====5.00
\

BRAKE INPUT

BRAKEMIN (0.00-5.00v*) : _____v*

BRAKEMAX (0.00-5.00)*: _____v*

REGENERATIVE BRAKING

MAXREGEN (0-100): _____%

REGENSCALE (0-100): _____%

THROTMAXREGEN (0-100) ____*%*

BRAKEMAXREGEN (0-100) ____*%*

BRAKEREGENRAMPTIME(0-255): _____*seconds*

OUTPUTS

PRECHARGE RELAY OUTPUT [ ] yes [ ] no

CONTACTOR OUTPUT [ ] yes [ ] no

COOLING OUTPUT(0-300): _____*centigrade*

MOTORMINTEMP(0-300): _____*centigrade*

MOTORMAXTEMP(0-300): _____*centigrade*

INVERTERMINTEMP(0-300): _____*centigrade*

INVERTERMAXTEMP(0-300): _____*centigrade*

TACHPPT (0-8) ____ *ppt*

AMPPPT (0-8) ____ *ppt*

BUTTON – ACCEPT CONFIG CHANGES

BUTTON – GEVCU STATUS

GEVCU CONFIGURATION TOOLTIPS

Tooltips that may be incorporated into GEVCU configuration html form.

GENERAL LIMITS

MAXRPM (1000-10000) _____rpm

The maximum motor speed the GEVCU will call for or allow. If this limit is reported by the Inverter, motor torque, speed or both will be reduced by CANbus command.

MAXTORQUE (0-100): _____%

The maximum motor torque that can be commanded by CANbus

ACTIVEHIGH [ ] yes [ ] no

Yes indicates activation of digital inputs will be read with 12v applied to pin. No indicates ACTIVE LOW with ground applied to pin read as ACTIVE.

LIMPSCALE(0-100) ____%

Percent of maximum torque and speed allowed when LIMPMODE input is active.

THROTTLE/ACCELERATOR

THROTMIN (0.00-5.00v) : _____v

Values below this voltage are read as a fault on the accelerator input.

THROTREGEN (0.00-5.00): _____v

Regenerative braking is linearly increased from this voltage/pedal position DOWN to THROTMIN.

THROTFWD(0.00-5.00): _____v

Region between THROTREGEN and THROTFORWD defines dead band or COAST area of throttle. Torque and speed are linearly increased from this point to THROTMAX.

THROTMAP(0.00-5.00): _____v

Defines point between THROTFWD and THROTMAX where 50% of available torque/speed is available. By setting this higher, better low speed resolution/control.

THROTMAX(0.00-5.00): _____v

Voltages above this are read as throttle faults.

THROTNOTMIN(0.00-5.00): _____*v*

Minimum voltage of second throttle input. 0 indicates no input. Any other value establishes ratio to THROTMIN

THROTNOTMAX(0.00-5.00): _____*v*

Maximum voltage of second throttle input. 0 disables second throttle input. Any other value corresponds to THROTMAX. It is acceptable for THROTNOTMIN to be greater than THROTNOTMAX, indicating a reversed linear input for comparison to the main throttle input.

0.00==THROTMIN==THROTREGEN==THROTFWD==THROTMAP=THROTMAX ==5.00

0.00==THROTNOTMIN==========THROTNOTMAX-====5.00
\

BRAKE INPUT

BRAKEMIN (0.00-5.00v*) :* _____*v*

Voltages below BRAKEMIN are read as brake sensor faults. Regenerative braking is linearly ramped between BRAKEMIN and BRAKEMAX in most instances.

BRAKEMAX (0.00-5.00)*:* _____*v*

Voltages above BRAKEMAX are read as brake sensor faults. Regenerative braking is linearly ramped between BRAKEMIN and BRAKEMAX in most instances.

REGENERATIVE BRAKING

MAXREGEN (0-100): _____%

Percentage of MAXTORQUE that can be applied as regenerative braking.

REGENSCALE (0-100): _____%

MAXREGEN, THROTMAXREGEN, and BRAKEMAXREGEN are all reduced identically if this value is less than 100.

BRAKEMAXREGEN (0-100) ____*%a*

Percent of MAXREGEN allowed for braking

BRAKEREGENRAMPTIME(0-255): _____*seconds*

Number of seconds to ramp from THROTTREGEN to BRAKEMAXREGEN in the event there is no brake pressure sensor input and we are doing regenerative braking from the brake light input only.

OUTPUTS

PRECHARGE RELAY OUTPUT [ ] yes [ ] no

Yes if you want a relay drive signal to engage an external relay. No if not.

CONTACTOR OUTPUT [ ] yes [ ] no

Yes if you want a 1.5amp relay output to engage high voltage contactor relay

COOLING OUTPUT(0-300): _____*centigrade*

Temperature of motor or inverter where a COOLING relay output is provided.

MOTORMINTEMP(0-300): _____*centigrade*

Minimum motor temperature corresponding to 1% PWM output on MOTOR TEMP output.

MOTORMAXTEMP(0-300): _____*centigrade*

Motor temperature corresponding to 99% PWM output on MOTOR TEMP .

INVERTERMINTEMP(0-300): _____*centigrade*

Minimum reported inverter temperature corresponding to 1% PWM output on INVERTER TEMP output.

INVERTERMAXTEMP(0-300): _____*centigrade*

Inverter temperature corresponding to 99% PWM output on INVERTER TEMP .

TACHPPT (0-8) ____ *ppt*

Pulses per turn for dashboard tachometer to read RPM output of GEVCU

AMPPPT (0-8) ____ *ppt*

Pulses per turn for dashboard tachometer to read amperes in hundreds on tachometer thousands of RPM gage.

BUTTON – ACCEPT CONFIG CHANGES

Sends these configuration changes to GEVCU

BUTTON – STATUS

Causes GEVCU to display current status screen.

GEVCU EEPROM MAPPING

This section describes the mapping and location of certain configuration variables necessary for the Generalized Vehicle Control Unit. When an end user sends the HTML form containing configuration items, those items will be extracted from the HTML and stored in the Macchina EEPROM so they are persistent between operations. The program can then retrieve these configuration variabales from EEPROM as part of the basic initialization procedure.

1000	**unsigned int**	**MAXRPM (1000-10000) _____rpm**
1002	**byte**	**MAXTORQUE (0-100): _____%**
1003	**bool**	**ACTIVEHIGH [ ] yes [ ] no**
1004	**byte**	**LIMPSCALE(0-100) ____%**
1005	**unsigned int**	**THROTMIN (0-500 /0.00-5.00v) : _____v**
1007	**unsigned int**	**THROTREGEN 0-500(0.00-5.00): _____v**
1009	**unsigned int**	**THROTFWD 0-500(0.00-5.00): _____v**
1011	**unsigned int**	**THROTMAP 0-500(0.00-5.00): _____v**
1013	**unsigned int**	**THROTMAX 0-500 (0.00-5.00): _____v**
1015	**unsigned int**	**THROTNOTMIN 0-500(0.00-5.00): _____v**
1017	**unsigned int**	**THROTNOTMAX0-500(0.00-5.00): _____v**
1019	**unsigned int**	**BRAKEMIN 0-500(0.00-5.00v) : _____v**
1021	**unsigned int**	**BRAKEMAX 0-500(0.00-5.00): _____v**
1023	**byte**	**MAXREGEN (0-100): _____%**
1024	**byte**	**REGENSCALE (0-100): _____%**
1025	**byte**	**THROTMAXREGEN (0-100) ____%**
1026	**byte**	**BRAKEMAXREGEN (0-100) ____%**
1027	**byte**	**BRAKEREGENRAMPTIME(0-255): _____seconds**
1028	**bool**	**PRECHARGE RELAY OUTPUT [ ] yes [ ] no**
1029	**bool**	**CONTACTOR OUTPUT [ ] yes [ ] no**
1030	**unsigned int**	**COOLING OUTPUT(0-300): _____centigrade**

1032	**unsigned int**	**MOTORMINTEMP(0-300): _____centigrade**
1034	**unsigned int**	**MOTORMAXTEMP(0-300): _____centigrade**
1036	**unsigned int**	**INVERTERMINTEMP(0-300): _____centigrade**
1038	**unsigned int**	**INVERTERMAXTEMP(0-300): _____centigrade**
1040	**byte**	**TACHPPT (0-8) ____ ppt**
1041	**byte**	**AMPPPT (0-8) ____ ppt**