;; Auto-generated. Do not edit!

(when (boundp 'robotnik\_msgs\_new::InverterStatus)

(if (not (find-package "ROBOTNIK\_MSGS\_NEW"))

(make-package "ROBOTNIK\_MSGS\_NEW"))

(shadow 'InverterStatus (find-package "ROBOTNIK\_MSGS\_NEW")))

(unless (find-package "ROBOTNIK\_MSGS\_NEW::INVERTERSTATUS")

(make-package "ROBOTNIK\_MSGS\_NEW::INVERTERSTATUS"))

(in-package "ROS")

;;//! \htmlinclude InverterStatus.msg.html

(if (not (find-package "STD\_MSGS"))

(ros::roseus-add-msgs "std\_msgs"))

(defclass robotnik\_msgs\_new::InverterStatus

:super ros::object

:slots (\_header \_ac\_voltage \_dc\_voltage \_load \_percentage \_temperature \_on \_serial\_number ))

(defmethod robotnik\_msgs\_new::InverterStatus

(:init

(&key

((:header \_\_header) (instance std\_msgs::Header :init))

((:ac\_voltage \_\_ac\_voltage) 0.0)

((:dc\_voltage \_\_dc\_voltage) 0.0)

((:load \_\_load) 0.0)

((:percentage \_\_percentage) 0.0)

((:temperature \_\_temperature) 0.0)

((:on \_\_on) nil)

((:serial\_number \_\_serial\_number) "")

)

(send-super :init)

(setq \_header \_\_header)

(setq \_ac\_voltage (float \_\_ac\_voltage))

(setq \_dc\_voltage (float \_\_dc\_voltage))

(setq \_load (float \_\_load))

(setq \_percentage (float \_\_percentage))

(setq \_temperature (float \_\_temperature))

(setq \_on \_\_on)

(setq \_serial\_number (string \_\_serial\_number))

self)

(:header

(&rest \_\_header)

(if (keywordp (car \_\_header))

(send\* \_header \_\_header)

(progn

(if \_\_header (setq \_header (car \_\_header)))

\_header)))

(:ac\_voltage

(&optional \_\_ac\_voltage)

(if \_\_ac\_voltage (setq \_ac\_voltage \_\_ac\_voltage)) \_ac\_voltage)

(:dc\_voltage

(&optional \_\_dc\_voltage)

(if \_\_dc\_voltage (setq \_dc\_voltage \_\_dc\_voltage)) \_dc\_voltage)

(:load

(&optional \_\_load)

(if \_\_load (setq \_load \_\_load)) \_load)

(:percentage

(&optional \_\_percentage)

(if \_\_percentage (setq \_percentage \_\_percentage)) \_percentage)

(:temperature

(&optional \_\_temperature)

(if \_\_temperature (setq \_temperature \_\_temperature)) \_temperature)

(:on

(&optional \_\_on)

(if \_\_on (setq \_on \_\_on)) \_on)

(:serial\_number

(&optional \_\_serial\_number)

(if \_\_serial\_number (setq \_serial\_number \_\_serial\_number)) \_serial\_number)

(:serialization-length

()

(+

;; std\_msgs/Header \_header

(send \_header :serialization-length)

;; float32 \_ac\_voltage

4

;; float32 \_dc\_voltage

4

;; float32 \_load

4

;; float32 \_percentage

4

;; float32 \_temperature

4

;; bool \_on

1

;; string \_serial\_number

4 (length \_serial\_number)

))

(:serialize

(&optional strm)

(let ((s (if strm strm

(make-string-output-stream (send self :serialization-length)))))

;; std\_msgs/Header \_header

(send \_header :serialize s)

;; float32 \_ac\_voltage

(sys::poke \_ac\_voltage (send s :buffer) (send s :count) :float) (incf (stream-count s) 4)

;; float32 \_dc\_voltage

(sys::poke \_dc\_voltage (send s :buffer) (send s :count) :float) (incf (stream-count s) 4)

;; float32 \_load

(sys::poke \_load (send s :buffer) (send s :count) :float) (incf (stream-count s) 4)

;; float32 \_percentage

(sys::poke \_percentage (send s :buffer) (send s :count) :float) (incf (stream-count s) 4)

;; float32 \_temperature

(sys::poke \_temperature (send s :buffer) (send s :count) :float) (incf (stream-count s) 4)

;; bool \_on

(if \_on (write-byte -1 s) (write-byte 0 s))

;; string \_serial\_number

(write-long (length \_serial\_number) s) (princ \_serial\_number s)

;;

(if (null strm) (get-output-stream-string s))))

(:deserialize

(buf &optional (ptr- 0))

;; std\_msgs/Header \_header

(send \_header :deserialize buf ptr-) (incf ptr- (send \_header :serialization-length))

;; float32 \_ac\_voltage

(setq \_ac\_voltage (sys::peek buf ptr- :float)) (incf ptr- 4)

;; float32 \_dc\_voltage

(setq \_dc\_voltage (sys::peek buf ptr- :float)) (incf ptr- 4)

;; float32 \_load

(setq \_load (sys::peek buf ptr- :float)) (incf ptr- 4)

;; float32 \_percentage

(setq \_percentage (sys::peek buf ptr- :float)) (incf ptr- 4)

;; float32 \_temperature

(setq \_temperature (sys::peek buf ptr- :float)) (incf ptr- 4)

;; bool \_on

(setq \_on (not (= 0 (sys::peek buf ptr- :char)))) (incf ptr- 1)

;; string \_serial\_number

(let (n) (setq n (sys::peek buf ptr- :integer)) (incf ptr- 4) (setq \_serial\_number (subseq buf ptr- (+ ptr- n))) (incf ptr- n))

;;

self)

)

(setf (get robotnik\_msgs\_new::InverterStatus :md5sum-) "d693ae8f6a65a0d35c771d1a15a9e199")

(setf (get robotnik\_msgs\_new::InverterStatus :datatype-) "robotnik\_msgs\_new/InverterStatus")

(setf (get robotnik\_msgs\_new::InverterStatus :definition-)

"Header header

float32 ac\_voltage # Output Voltage in Volts (Mandatory)

float32 dc\_voltage # Input Voltage in Volts (If unmeasured NaN)

float32 load # Current percentage load on 0 to 1 range (If unmeasured NaN)

float32 percentage # Charge percentage on 0 to 1 range (If unmeasured NaN)

float32 temperature # Current temperature in centigrads (If unmeasured NaN)

bool on # True if the inverter is on

string serial\_number # The best approximation of the battery serial number

================================================================================

MSG: std\_msgs/Header

# Standard metadata for higher-level stamped data types.

# This is generally used to communicate timestamped data

# in a particular coordinate frame.

#

# sequence ID: consecutively increasing ID

uint32 seq

#Two-integer timestamp that is expressed as:

# \* stamp.sec: seconds (stamp\_secs) since epoch (in Python the variable is called 'secs')

# \* stamp.nsec: nanoseconds since stamp\_secs (in Python the variable is called 'nsecs')

# time-handling sugar is provided by the client library

time stamp

#Frame this data is associated with

string frame\_id

")

(provide :robotnik\_msgs\_new/InverterStatus "d693ae8f6a65a0d35c771d1a15a9e199")