;; Auto-generated. Do not edit!

(when (boundp 'ur\_msgs\_new::RobotModeDataMsg)

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

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

(shadow 'RobotModeDataMsg (find-package "UR\_MSGS\_NEW")))

(unless (find-package "UR\_MSGS\_NEW::ROBOTMODEDATAMSG")

(make-package "UR\_MSGS\_NEW::ROBOTMODEDATAMSG"))

(in-package "ROS")

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

(defclass ur\_msgs\_new::RobotModeDataMsg

:super ros::object

:slots (\_timestamp \_is\_robot\_connected \_is\_real\_robot\_enabled \_is\_power\_on\_robot \_is\_emergency\_stopped \_is\_protective\_stopped \_is\_program\_running \_is\_program\_paused ))

(defmethod ur\_msgs\_new::RobotModeDataMsg

(:init

(&key

((:timestamp \_\_timestamp) 0)

((:is\_robot\_connected \_\_is\_robot\_connected) nil)

((:is\_real\_robot\_enabled \_\_is\_real\_robot\_enabled) nil)

((:is\_power\_on\_robot \_\_is\_power\_on\_robot) nil)

((:is\_emergency\_stopped \_\_is\_emergency\_stopped) nil)

((:is\_protective\_stopped \_\_is\_protective\_stopped) nil)

((:is\_program\_running \_\_is\_program\_running) nil)

((:is\_program\_paused \_\_is\_program\_paused) nil)

)

(send-super :init)

(setq \_timestamp (round \_\_timestamp))

(setq \_is\_robot\_connected \_\_is\_robot\_connected)

(setq \_is\_real\_robot\_enabled \_\_is\_real\_robot\_enabled)

(setq \_is\_power\_on\_robot \_\_is\_power\_on\_robot)

(setq \_is\_emergency\_stopped \_\_is\_emergency\_stopped)

(setq \_is\_protective\_stopped \_\_is\_protective\_stopped)

(setq \_is\_program\_running \_\_is\_program\_running)

(setq \_is\_program\_paused \_\_is\_program\_paused)

self)

(:timestamp

(&optional \_\_timestamp)

(if \_\_timestamp (setq \_timestamp \_\_timestamp)) \_timestamp)

(:is\_robot\_connected

(&optional \_\_is\_robot\_connected)

(if \_\_is\_robot\_connected (setq \_is\_robot\_connected \_\_is\_robot\_connected)) \_is\_robot\_connected)

(:is\_real\_robot\_enabled

(&optional \_\_is\_real\_robot\_enabled)

(if \_\_is\_real\_robot\_enabled (setq \_is\_real\_robot\_enabled \_\_is\_real\_robot\_enabled)) \_is\_real\_robot\_enabled)

(:is\_power\_on\_robot

(&optional \_\_is\_power\_on\_robot)

(if \_\_is\_power\_on\_robot (setq \_is\_power\_on\_robot \_\_is\_power\_on\_robot)) \_is\_power\_on\_robot)

(:is\_emergency\_stopped

(&optional \_\_is\_emergency\_stopped)

(if \_\_is\_emergency\_stopped (setq \_is\_emergency\_stopped \_\_is\_emergency\_stopped)) \_is\_emergency\_stopped)

(:is\_protective\_stopped

(&optional \_\_is\_protective\_stopped)

(if \_\_is\_protective\_stopped (setq \_is\_protective\_stopped \_\_is\_protective\_stopped)) \_is\_protective\_stopped)

(:is\_program\_running

(&optional \_\_is\_program\_running)

(if \_\_is\_program\_running (setq \_is\_program\_running \_\_is\_program\_running)) \_is\_program\_running)

(:is\_program\_paused

(&optional \_\_is\_program\_paused)

(if \_\_is\_program\_paused (setq \_is\_program\_paused \_\_is\_program\_paused)) \_is\_program\_paused)

(:serialization-length

()

(+

;; uint64 \_timestamp

8

;; bool \_is\_robot\_connected

1

;; bool \_is\_real\_robot\_enabled

1

;; bool \_is\_power\_on\_robot

1

;; bool \_is\_emergency\_stopped

1

;; bool \_is\_protective\_stopped

1

;; bool \_is\_program\_running

1

;; bool \_is\_program\_paused

1

))

(:serialize

(&optional strm)

(let ((s (if strm strm

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

;; uint64 \_timestamp

#+(or :alpha :irix6 :x86\_64)

(progn (sys::poke \_timestamp (send s :buffer) (send s :count) :long) (incf (stream-count s) 8))

#-(or :alpha :irix6 :x86\_64)

(cond ((and (class \_timestamp) (= (length (\_timestamp . bv)) 2)) ;; bignum

(write-long (ash (elt (\_timestamp . bv) 0) 0) s)

(write-long (ash (elt (\_timestamp . bv) 1) -1) s))

((and (class \_timestamp) (= (length (\_timestamp . bv)) 1)) ;; big1

(write-long (elt (\_timestamp . bv) 0) s)

(write-long (if (>= \_timestamp 0) 0 #xffffffff) s))

(t ;; integer

(write-long \_timestamp s)(write-long (if (>= \_timestamp 0) 0 #xffffffff) s)))

;; bool \_is\_robot\_connected

(if \_is\_robot\_connected (write-byte -1 s) (write-byte 0 s))

;; bool \_is\_real\_robot\_enabled

(if \_is\_real\_robot\_enabled (write-byte -1 s) (write-byte 0 s))

;; bool \_is\_power\_on\_robot

(if \_is\_power\_on\_robot (write-byte -1 s) (write-byte 0 s))

;; bool \_is\_emergency\_stopped

(if \_is\_emergency\_stopped (write-byte -1 s) (write-byte 0 s))

;; bool \_is\_protective\_stopped

(if \_is\_protective\_stopped (write-byte -1 s) (write-byte 0 s))

;; bool \_is\_program\_running

(if \_is\_program\_running (write-byte -1 s) (write-byte 0 s))

;; bool \_is\_program\_paused

(if \_is\_program\_paused (write-byte -1 s) (write-byte 0 s))

;;

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

(:deserialize

(buf &optional (ptr- 0))

;; uint64 \_timestamp

#+(or :alpha :irix6 :x86\_64)

(setf \_timestamp (prog1 (sys::peek buf ptr- :long) (incf ptr- 8)))

#-(or :alpha :irix6 :x86\_64)

(setf \_timestamp (let ((b0 (prog1 (sys::peek buf ptr- :integer) (incf ptr- 4)))

(b1 (prog1 (sys::peek buf ptr- :integer) (incf ptr- 4))))

(cond ((= b1 -1) b0)

((and (= b1 0)

(<= lisp::most-negative-fixnum b0 lisp::most-positive-fixnum))

b0)

((= b1 0) (make-instance bignum :size 1 :bv (integer-vector b0)))

(t (make-instance bignum :size 2 :bv (integer-vector b0 (ash b1 1)))))))

;; bool \_is\_robot\_connected

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

;; bool \_is\_real\_robot\_enabled

(setq \_is\_real\_robot\_enabled (not (= 0 (sys::peek buf ptr- :char)))) (incf ptr- 1)

;; bool \_is\_power\_on\_robot

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

;; bool \_is\_emergency\_stopped

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

;; bool \_is\_protective\_stopped

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

;; bool \_is\_program\_running

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

;; bool \_is\_program\_paused

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

;;

self)

)

(setf (get ur\_msgs\_new::RobotModeDataMsg :md5sum-) "867308ca39e2cc0644b50db27deb661f")

(setf (get ur\_msgs\_new::RobotModeDataMsg :datatype-) "ur\_msgs\_new/RobotModeDataMsg")

(setf (get ur\_msgs\_new::RobotModeDataMsg :definition-)

"# This data structure contains the RobotModeData structure

# used by the Universal Robots controller

#

# This data structure is send at 10 Hz on TCP port 30002

#

# Note: this message does not carry all fields from the RobotModeData structure as broadcast by the robot controller, but a subset.

uint64 timestamp

bool is\_robot\_connected

bool is\_real\_robot\_enabled

bool is\_power\_on\_robot

bool is\_emergency\_stopped

bool is\_protective\_stopped

bool is\_program\_running

bool is\_program\_paused

")

(provide :ur\_msgs\_new/RobotModeDataMsg "867308ca39e2cc0644b50db27deb661f")