;; Auto-generated. Do not edit!

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

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

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

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

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

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

(in-package "ROS")

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

(defclass ur\_msgs\_new::IOStates

:super ros::object

:slots (\_digital\_in\_states \_digital\_out\_states \_flag\_states \_analog\_in\_states \_analog\_out\_states ))

(defmethod ur\_msgs\_new::IOStates

(:init

(&key

((:digital\_in\_states \_\_digital\_in\_states) (let (r) (dotimes (i 0) (push (instance ur\_msgs\_new::Digital :init) r)) r))

((:digital\_out\_states \_\_digital\_out\_states) (let (r) (dotimes (i 0) (push (instance ur\_msgs\_new::Digital :init) r)) r))

((:flag\_states \_\_flag\_states) (let (r) (dotimes (i 0) (push (instance ur\_msgs\_new::Digital :init) r)) r))

((:analog\_in\_states \_\_analog\_in\_states) (let (r) (dotimes (i 0) (push (instance ur\_msgs\_new::Analog :init) r)) r))

((:analog\_out\_states \_\_analog\_out\_states) (let (r) (dotimes (i 0) (push (instance ur\_msgs\_new::Analog :init) r)) r))

)

(send-super :init)

(setq \_digital\_in\_states \_\_digital\_in\_states)

(setq \_digital\_out\_states \_\_digital\_out\_states)

(setq \_flag\_states \_\_flag\_states)

(setq \_analog\_in\_states \_\_analog\_in\_states)

(setq \_analog\_out\_states \_\_analog\_out\_states)

self)

(:digital\_in\_states

(&rest \_\_digital\_in\_states)

(if (keywordp (car \_\_digital\_in\_states))

(send\* \_digital\_in\_states \_\_digital\_in\_states)

(progn

(if \_\_digital\_in\_states (setq \_digital\_in\_states (car \_\_digital\_in\_states)))

\_digital\_in\_states)))

(:digital\_out\_states

(&rest \_\_digital\_out\_states)

(if (keywordp (car \_\_digital\_out\_states))

(send\* \_digital\_out\_states \_\_digital\_out\_states)

(progn

(if \_\_digital\_out\_states (setq \_digital\_out\_states (car \_\_digital\_out\_states)))

\_digital\_out\_states)))

(:flag\_states

(&rest \_\_flag\_states)

(if (keywordp (car \_\_flag\_states))

(send\* \_flag\_states \_\_flag\_states)

(progn

(if \_\_flag\_states (setq \_flag\_states (car \_\_flag\_states)))

\_flag\_states)))

(:analog\_in\_states

(&rest \_\_analog\_in\_states)

(if (keywordp (car \_\_analog\_in\_states))

(send\* \_analog\_in\_states \_\_analog\_in\_states)

(progn

(if \_\_analog\_in\_states (setq \_analog\_in\_states (car \_\_analog\_in\_states)))

\_analog\_in\_states)))

(:analog\_out\_states

(&rest \_\_analog\_out\_states)

(if (keywordp (car \_\_analog\_out\_states))

(send\* \_analog\_out\_states \_\_analog\_out\_states)

(progn

(if \_\_analog\_out\_states (setq \_analog\_out\_states (car \_\_analog\_out\_states)))

\_analog\_out\_states)))

(:serialization-length

()

(+

;; ur\_msgs\_new/Digital[] \_digital\_in\_states

(apply #'+ (send-all \_digital\_in\_states :serialization-length)) 4

;; ur\_msgs\_new/Digital[] \_digital\_out\_states

(apply #'+ (send-all \_digital\_out\_states :serialization-length)) 4

;; ur\_msgs\_new/Digital[] \_flag\_states

(apply #'+ (send-all \_flag\_states :serialization-length)) 4

;; ur\_msgs\_new/Analog[] \_analog\_in\_states

(apply #'+ (send-all \_analog\_in\_states :serialization-length)) 4

;; ur\_msgs\_new/Analog[] \_analog\_out\_states

(apply #'+ (send-all \_analog\_out\_states :serialization-length)) 4

))

(:serialize

(&optional strm)

(let ((s (if strm strm

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

;; ur\_msgs\_new/Digital[] \_digital\_in\_states

(write-long (length \_digital\_in\_states) s)

(dolist (elem \_digital\_in\_states)

(send elem :serialize s)

)

;; ur\_msgs\_new/Digital[] \_digital\_out\_states

(write-long (length \_digital\_out\_states) s)

(dolist (elem \_digital\_out\_states)

(send elem :serialize s)

)

;; ur\_msgs\_new/Digital[] \_flag\_states

(write-long (length \_flag\_states) s)

(dolist (elem \_flag\_states)

(send elem :serialize s)

)

;; ur\_msgs\_new/Analog[] \_analog\_in\_states

(write-long (length \_analog\_in\_states) s)

(dolist (elem \_analog\_in\_states)

(send elem :serialize s)

)

;; ur\_msgs\_new/Analog[] \_analog\_out\_states

(write-long (length \_analog\_out\_states) s)

(dolist (elem \_analog\_out\_states)

(send elem :serialize s)

)

;;

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

(:deserialize

(buf &optional (ptr- 0))

;; ur\_msgs\_new/Digital[] \_digital\_in\_states

(let (n)

(setq n (sys::peek buf ptr- :integer)) (incf ptr- 4)

(setq \_digital\_in\_states (let (r) (dotimes (i n) (push (instance ur\_msgs\_new::Digital :init) r)) r))

(dolist (elem- \_digital\_in\_states)

(send elem- :deserialize buf ptr-) (incf ptr- (send elem- :serialization-length))

))

;; ur\_msgs\_new/Digital[] \_digital\_out\_states

(let (n)

(setq n (sys::peek buf ptr- :integer)) (incf ptr- 4)

(setq \_digital\_out\_states (let (r) (dotimes (i n) (push (instance ur\_msgs\_new::Digital :init) r)) r))

(dolist (elem- \_digital\_out\_states)

(send elem- :deserialize buf ptr-) (incf ptr- (send elem- :serialization-length))

))

;; ur\_msgs\_new/Digital[] \_flag\_states

(let (n)

(setq n (sys::peek buf ptr- :integer)) (incf ptr- 4)

(setq \_flag\_states (let (r) (dotimes (i n) (push (instance ur\_msgs\_new::Digital :init) r)) r))

(dolist (elem- \_flag\_states)

(send elem- :deserialize buf ptr-) (incf ptr- (send elem- :serialization-length))

))

;; ur\_msgs\_new/Analog[] \_analog\_in\_states

(let (n)

(setq n (sys::peek buf ptr- :integer)) (incf ptr- 4)

(setq \_analog\_in\_states (let (r) (dotimes (i n) (push (instance ur\_msgs\_new::Analog :init) r)) r))

(dolist (elem- \_analog\_in\_states)

(send elem- :deserialize buf ptr-) (incf ptr- (send elem- :serialization-length))

))

;; ur\_msgs\_new/Analog[] \_analog\_out\_states

(let (n)

(setq n (sys::peek buf ptr- :integer)) (incf ptr- 4)

(setq \_analog\_out\_states (let (r) (dotimes (i n) (push (instance ur\_msgs\_new::Analog :init) r)) r))

(dolist (elem- \_analog\_out\_states)

(send elem- :deserialize buf ptr-) (incf ptr- (send elem- :serialization-length))

))

;;

self)

)

(setf (get ur\_msgs\_new::IOStates :md5sum-) "3033784e7041da89491b97cc4c1105b5")

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

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

"Digital[] digital\_in\_states

Digital[] digital\_out\_states

Digital[] flag\_states

Analog[] analog\_in\_states

Analog[] analog\_out\_states

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

MSG: ur\_msgs\_new/Digital

uint8 pin

bool state

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

MSG: ur\_msgs\_new/Analog

uint8 VOLTAGE=0

uint8 CURRENT=1

uint8 pin

uint8 domain # can be VOLTAGE or CURRENT

float32 state

")

(provide :ur\_msgs\_new/IOStates "3033784e7041da89491b97cc4c1105b5")