;; Auto-generated. Do not edit!

(when (boundp 'ur\_msgs::MasterboardDataMsg)

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

(make-package "UR\_MSGS"))

(shadow 'MasterboardDataMsg (find-package "UR\_MSGS")))

(unless (find-package "UR\_MSGS::MASTERBOARDDATAMSG")

(make-package "UR\_MSGS::MASTERBOARDDATAMSG"))

(in-package "ROS")

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

(defclass ur\_msgs::MasterboardDataMsg

:super ros::object

:slots (\_digital\_input\_bits \_digital\_output\_bits \_analog\_input\_range0 \_analog\_input\_range1 \_analog\_input0 \_analog\_input1 \_analog\_output\_domain0 \_analog\_output\_domain1 \_analog\_output0 \_analog\_output1 \_masterboard\_temperature \_robot\_voltage\_48V \_robot\_current \_master\_io\_current \_master\_safety\_state \_master\_onoff\_state ))

(defmethod ur\_msgs::MasterboardDataMsg

(:init

(&key

((:digital\_input\_bits \_\_digital\_input\_bits) 0)

((:digital\_output\_bits \_\_digital\_output\_bits) 0)

((:analog\_input\_range0 \_\_analog\_input\_range0) 0)

((:analog\_input\_range1 \_\_analog\_input\_range1) 0)

((:analog\_input0 \_\_analog\_input0) 0.0)

((:analog\_input1 \_\_analog\_input1) 0.0)

((:analog\_output\_domain0 \_\_analog\_output\_domain0) 0)

((:analog\_output\_domain1 \_\_analog\_output\_domain1) 0)

((:analog\_output0 \_\_analog\_output0) 0.0)

((:analog\_output1 \_\_analog\_output1) 0.0)

((:masterboard\_temperature \_\_masterboard\_temperature) 0.0)

((:robot\_voltage\_48V \_\_robot\_voltage\_48V) 0.0)

((:robot\_current \_\_robot\_current) 0.0)

((:master\_io\_current \_\_master\_io\_current) 0.0)

((:master\_safety\_state \_\_master\_safety\_state) 0)

((:master\_onoff\_state \_\_master\_onoff\_state) 0)

)

(send-super :init)

(setq \_digital\_input\_bits (round \_\_digital\_input\_bits))

(setq \_digital\_output\_bits (round \_\_digital\_output\_bits))

(setq \_analog\_input\_range0 (round \_\_analog\_input\_range0))

(setq \_analog\_input\_range1 (round \_\_analog\_input\_range1))

(setq \_analog\_input0 (float \_\_analog\_input0))

(setq \_analog\_input1 (float \_\_analog\_input1))

(setq \_analog\_output\_domain0 (round \_\_analog\_output\_domain0))

(setq \_analog\_output\_domain1 (round \_\_analog\_output\_domain1))

(setq \_analog\_output0 (float \_\_analog\_output0))

(setq \_analog\_output1 (float \_\_analog\_output1))

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

(setq \_robot\_voltage\_48V (float \_\_robot\_voltage\_48V))

(setq \_robot\_current (float \_\_robot\_current))

(setq \_master\_io\_current (float \_\_master\_io\_current))

(setq \_master\_safety\_state (round \_\_master\_safety\_state))

(setq \_master\_onoff\_state (round \_\_master\_onoff\_state))

self)

(:digital\_input\_bits

(&optional \_\_digital\_input\_bits)

(if \_\_digital\_input\_bits (setq \_digital\_input\_bits \_\_digital\_input\_bits)) \_digital\_input\_bits)

(:digital\_output\_bits

(&optional \_\_digital\_output\_bits)

(if \_\_digital\_output\_bits (setq \_digital\_output\_bits \_\_digital\_output\_bits)) \_digital\_output\_bits)

(:analog\_input\_range0

(&optional \_\_analog\_input\_range0)

(if \_\_analog\_input\_range0 (setq \_analog\_input\_range0 \_\_analog\_input\_range0)) \_analog\_input\_range0)

(:analog\_input\_range1

(&optional \_\_analog\_input\_range1)

(if \_\_analog\_input\_range1 (setq \_analog\_input\_range1 \_\_analog\_input\_range1)) \_analog\_input\_range1)

(:analog\_input0

(&optional \_\_analog\_input0)

(if \_\_analog\_input0 (setq \_analog\_input0 \_\_analog\_input0)) \_analog\_input0)

(:analog\_input1

(&optional \_\_analog\_input1)

(if \_\_analog\_input1 (setq \_analog\_input1 \_\_analog\_input1)) \_analog\_input1)

(:analog\_output\_domain0

(&optional \_\_analog\_output\_domain0)

(if \_\_analog\_output\_domain0 (setq \_analog\_output\_domain0 \_\_analog\_output\_domain0)) \_analog\_output\_domain0)

(:analog\_output\_domain1

(&optional \_\_analog\_output\_domain1)

(if \_\_analog\_output\_domain1 (setq \_analog\_output\_domain1 \_\_analog\_output\_domain1)) \_analog\_output\_domain1)

(:analog\_output0

(&optional \_\_analog\_output0)

(if \_\_analog\_output0 (setq \_analog\_output0 \_\_analog\_output0)) \_analog\_output0)

(:analog\_output1

(&optional \_\_analog\_output1)

(if \_\_analog\_output1 (setq \_analog\_output1 \_\_analog\_output1)) \_analog\_output1)

(:masterboard\_temperature

(&optional \_\_masterboard\_temperature)

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

(:robot\_voltage\_48V

(&optional \_\_robot\_voltage\_48V)

(if \_\_robot\_voltage\_48V (setq \_robot\_voltage\_48V \_\_robot\_voltage\_48V)) \_robot\_voltage\_48V)

(:robot\_current

(&optional \_\_robot\_current)

(if \_\_robot\_current (setq \_robot\_current \_\_robot\_current)) \_robot\_current)

(:master\_io\_current

(&optional \_\_master\_io\_current)

(if \_\_master\_io\_current (setq \_master\_io\_current \_\_master\_io\_current)) \_master\_io\_current)

(:master\_safety\_state

(&optional \_\_master\_safety\_state)

(if \_\_master\_safety\_state (setq \_master\_safety\_state \_\_master\_safety\_state)) \_master\_safety\_state)

(:master\_onoff\_state

(&optional \_\_master\_onoff\_state)

(if \_\_master\_onoff\_state (setq \_master\_onoff\_state \_\_master\_onoff\_state)) \_master\_onoff\_state)

(:serialization-length

()

(+

;; int16 \_digital\_input\_bits

2

;; int16 \_digital\_output\_bits

2

;; int8 \_analog\_input\_range0

1

;; int8 \_analog\_input\_range1

1

;; float64 \_analog\_input0

8

;; float64 \_analog\_input1

8

;; int8 \_analog\_output\_domain0

1

;; int8 \_analog\_output\_domain1

1

;; float64 \_analog\_output0

8

;; float64 \_analog\_output1

8

;; float32 \_masterboard\_temperature

4

;; float32 \_robot\_voltage\_48V

4

;; float32 \_robot\_current

4

;; float32 \_master\_io\_current

4

;; uint8 \_master\_safety\_state

1

;; uint8 \_master\_onoff\_state

1

))

(:serialize

(&optional strm)

(let ((s (if strm strm

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

;; int16 \_digital\_input\_bits

(write-word \_digital\_input\_bits s)

;; int16 \_digital\_output\_bits

(write-word \_digital\_output\_bits s)

;; int8 \_analog\_input\_range0

(write-byte \_analog\_input\_range0 s)

;; int8 \_analog\_input\_range1

(write-byte \_analog\_input\_range1 s)

;; float64 \_analog\_input0

(sys::poke \_analog\_input0 (send s :buffer) (send s :count) :double) (incf (stream-count s) 8)

;; float64 \_analog\_input1

(sys::poke \_analog\_input1 (send s :buffer) (send s :count) :double) (incf (stream-count s) 8)

;; int8 \_analog\_output\_domain0

(write-byte \_analog\_output\_domain0 s)

;; int8 \_analog\_output\_domain1

(write-byte \_analog\_output\_domain1 s)

;; float64 \_analog\_output0

(sys::poke \_analog\_output0 (send s :buffer) (send s :count) :double) (incf (stream-count s) 8)

;; float64 \_analog\_output1

(sys::poke \_analog\_output1 (send s :buffer) (send s :count) :double) (incf (stream-count s) 8)

;; float32 \_masterboard\_temperature

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

;; float32 \_robot\_voltage\_48V

(sys::poke \_robot\_voltage\_48V (send s :buffer) (send s :count) :float) (incf (stream-count s) 4)

;; float32 \_robot\_current

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

;; float32 \_master\_io\_current

(sys::poke \_master\_io\_current (send s :buffer) (send s :count) :float) (incf (stream-count s) 4)

;; uint8 \_master\_safety\_state

(write-byte \_master\_safety\_state s)

;; uint8 \_master\_onoff\_state

(write-byte \_master\_onoff\_state s)

;;

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

(:deserialize

(buf &optional (ptr- 0))

;; int16 \_digital\_input\_bits

(setq \_digital\_input\_bits (sys::peek buf ptr- :short)) (incf ptr- 2)

;; int16 \_digital\_output\_bits

(setq \_digital\_output\_bits (sys::peek buf ptr- :short)) (incf ptr- 2)

;; int8 \_analog\_input\_range0

(setq \_analog\_input\_range0 (sys::peek buf ptr- :char)) (incf ptr- 1)

(if (> \_analog\_input\_range0 127) (setq \_analog\_input\_range0 (- \_analog\_input\_range0 256)))

;; int8 \_analog\_input\_range1

(setq \_analog\_input\_range1 (sys::peek buf ptr- :char)) (incf ptr- 1)

(if (> \_analog\_input\_range1 127) (setq \_analog\_input\_range1 (- \_analog\_input\_range1 256)))

;; float64 \_analog\_input0

(setq \_analog\_input0 (sys::peek buf ptr- :double)) (incf ptr- 8)

;; float64 \_analog\_input1

(setq \_analog\_input1 (sys::peek buf ptr- :double)) (incf ptr- 8)

;; int8 \_analog\_output\_domain0

(setq \_analog\_output\_domain0 (sys::peek buf ptr- :char)) (incf ptr- 1)

(if (> \_analog\_output\_domain0 127) (setq \_analog\_output\_domain0 (- \_analog\_output\_domain0 256)))

;; int8 \_analog\_output\_domain1

(setq \_analog\_output\_domain1 (sys::peek buf ptr- :char)) (incf ptr- 1)

(if (> \_analog\_output\_domain1 127) (setq \_analog\_output\_domain1 (- \_analog\_output\_domain1 256)))

;; float64 \_analog\_output0

(setq \_analog\_output0 (sys::peek buf ptr- :double)) (incf ptr- 8)

;; float64 \_analog\_output1

(setq \_analog\_output1 (sys::peek buf ptr- :double)) (incf ptr- 8)

;; float32 \_masterboard\_temperature

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

;; float32 \_robot\_voltage\_48V

(setq \_robot\_voltage\_48V (sys::peek buf ptr- :float)) (incf ptr- 4)

;; float32 \_robot\_current

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

;; float32 \_master\_io\_current

(setq \_master\_io\_current (sys::peek buf ptr- :float)) (incf ptr- 4)

;; uint8 \_master\_safety\_state

(setq \_master\_safety\_state (sys::peek buf ptr- :char)) (incf ptr- 1)

;; uint8 \_master\_onoff\_state

(setq \_master\_onoff\_state (sys::peek buf ptr- :char)) (incf ptr- 1)

;;

self)

)

(setf (get ur\_msgs::MasterboardDataMsg :md5sum-) "a4aa4d8ccbd10a18ef4008b679f6ccbe")

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

(setf (get ur\_msgs::MasterboardDataMsg :definition-)

"# This data structure contains the MasterboardData structure

# used by the Universal Robots controller

#

# MasterboardData is part of the data structure being send on the

# secondary client communications interface

#

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

#

# Dokumentation can be found on the Universal Robots Support Wiki

# (http://wiki03.lynero.net/Technical/DataStreamFromURController?rev=8)

int16 digital\_input\_bits

int16 digital\_output\_bits

int8 analog\_input\_range0

int8 analog\_input\_range1

float64 analog\_input0

float64 analog\_input1

int8 analog\_output\_domain0

int8 analog\_output\_domain1

float64 analog\_output0

float64 analog\_output1

float32 masterboard\_temperature

float32 robot\_voltage\_48V

float32 robot\_current

float32 master\_io\_current

uint8 master\_safety\_state

uint8 master\_onoff\_state

")

(provide :ur\_msgs/MasterboardDataMsg "a4aa4d8ccbd10a18ef4008b679f6ccbe")