Data Structure [reqdata]

(Date : 2022/03/01)

V 4.3.1

[ENG]

Rainbow-Robotics

www.rainbow-robotics.com

Data Structure [reqdata]

- **Caution**
  This document is a structure description of status data that can be acquired through port 5001.
- The contents may be updated depending on the software version, and this document is based on version 4.3.1.

```
// ----------------------------------------------------------------------------
// Based on Software Version 4.3.1
// ----------------------------------------------------------------------------
#define MAX_SHARED_DATA    145
// ----------------------------------------------------------------------------

typedef union{
   struct{
      char                    header[4];
      // Byte count ~: 4
      float                   time;
      float                   jnt_ref[6];
      float                   jnt_ang[6];
      float                   jnt_cur[6];
      // Byte count ~: 80
      float                   tcp_ref[6];
      float                   tcp_pos[6];
      // Byte count ~: 128
      float                   analog_in[4];
      float                   analog_out[4];
      int                     digital_in[16];
      int                     digital_out[16];
      // Byte count ~: 288
      float                   jnt_temperature[6];
      // Byte count ~: 312
      int                     task_pc;
      int                     task_repeat;
      int                     task_run_id;
      int                     task_run_num;
      int                     task_run_time;
      int                     task_state;
      // Byte count ~: 336
      float                   default_speed;
      int                     robot_state;
      int                     information_chunk_1;//information bit combination
      // Byte count ~: 348
      float                   reserved_1[6];
      int                     jnt_info[6];
      // Byte count ~: 396
      int                     collision_detect_onoff;
      int                     is_freedrive_mode;
      int                     real_vs_simulation_mode;
      // Byte count ~: 408
      int                     init_state_info;
      int                     init_error;
      // Byte count ~: 416
      float                   tfb_analog_in[2];
      int                     tfb_digital_in[2];
```

```
    int                     tfb_digital_out[2];
    float                   tfb_voltage_out;
    // Byte count ~: 444

    int                     op_stat_collision_occur;
    int                     op_stat_sos_flag;
    int                     op_stat_self_collision;
    int                     op_stat_soft_estop_occur;
    int                     op_stat_ems_flag;
    // Byte count ~: 464
    int                     information_chunk_2;
    int                     information_chunk_3;
    // Byte count ~: 472
    int                     inbox_trap_flag[2];
    int                     inbox_check_mode[2];
    // Byte count ~: 488
    float                   eft_fx;
    float                   eft_fy;
    float                   eft_fz;
    float                   eft_mx;
    float                   eft_my;
    float                   eft_mz;
    // Byte count ~: 512
    int                     information_chunk_4;
    // Byte count ~: 516
    float                   extend_io1_analog_in[4];
    float                   extend_io1_analog_out[4];
    unsigned int            extend_io1_digital_info;
    // Byte count ~: 552
    float                   aa_joint_ref[6];
    // Byte count ~: 576
    unsigned int            safety_board_stat_info;
    // Byte count ~: 580

  }sdata;
  float fdata[MAX_SHARED_DATA];
  int idata[MAX_SHARED_DATA];
}systemSTAT;
```

char header[4]

Header of this data structure
 	
header[0] = 0x24;
header[1] = size & 0xFF;
header[2] = (size >> 8) & 0xFF
header[3] = 0x03; // Type of this data

float time
Basic Timer (unit: second)

float jnt_ref[6]
Reference angle of each joint. (unit: degree) 0 = Base / 1 = Shoulder / 2 = Elbow / 3 = Wrist1 / 4 = Wrist2 / 5 = Wrist3

float jnt_ang[6]
Real-encoder (measured) angle of each joint. (unit: degree) 0 = Base / 1 = Shoulder / 2 = Elbow / 3 = Wrist1 / 4 = Wrist2 / 5 = Wrist3

float jnt_cur[6]
Measured current of each joint. (unit: Ampere) 0 = Base / 1 = Shoulder / 2 = Elbow / 3 = Wrist1 / 4 = Wrist2 / 5 = Wrist3

float tcp_ref[6]
TCP posture info based on reference-joint-angles (unit: mm & degree) 0 = X / 1 = Y / 2 = Z / 3 = Rx / 4 = Ry / 5 = Rz

float tcp_pos[6]
TCP posture info based on encoder-joint-angles (unit: mm & degree) 0 = X / 1 = Y / 2 = Z / 3 = Rx / 4 = Ry / 5 = Rz ※ It is being transmitted overwritten based on the current reference.

float analog_in[4]
Control box analog input measurement information of each channel (unit: Voltage) Channel number: 0~3

float analog_out[4]
Control box analog output information of each channel (unit: Voltage) Channel number: 0~3

int **digital_in[16]**
Control box digital input measurement information of each channel (value: 0 or 1) Channel number: 0~15

int **digital_out[16]**
Control box digital output information of each channel (value: 0 or 1) Channel number: 0~15

float **jnt_temperature[6]**
Measured temperature of each joint. (unit: celsius) 0 = Base / 1 = Shoulder / 2 = Elbow / 3 = Wrist1 / 4 = Wrist2 / 5 = Wrist3

int **task_pc (Not for user)**
Target program counter position during [STEP] function.

int **task_repeat (Not for user)**
Target program execution number in [PLAY] page.

int **task_run_id (Not for user)**
Running program counter position.

int **task_run_num (Not for user)**
Current program execution number in [PLAY] page.

int **task_run_time (Not for user)**
Time since the program started (unit: second)

int **task_state**
Basic state of ‘Program Execution’ 1 = Program not run / Idle 3 = Program is running 2 = Program is running + but ‘Paused’ state

float **default_speed**
Default speed multiplier value of robot motion (=speed bar in UI) (value: 0 ~ 1)

int robot_state

Move (motion) state
if(robot_state == 1){
  // no motion command, idle
}else{
  // robot motion command is executing
}
1 = No motion command / Idle
3 = Executing motion command(s)
5 = No motion (Move) command + but executing Conveyor or Force control mode
60+index = Under MovePB/ITPL/Pro command / index is passing waypoint number

int information_chunk_1
Information chunk to deliver various state information (power and others) It consists of a combination of bits. (information_chunk_1 >> 0) & 0b01 = Control Box's 48V input state (information_chunk_1 >> 1) & 0b01 = Control Box's 48V output state (information_chunk_1 >> 2) & 0b01 = Control Box's 24V input state (information_chunk_1 >> 3) & 0b01 = Control Box's E-Stop state 1 (information_chunk_1 >> 4) & 0b01 = Control Box's User Switch state (information_chunk_1 >> 5) & 0b01 = Control Box's E-Stop state 2 (information_chunk_1 >> 6) & 0b01 = Whether power is applied to the robot arm (information_chunk_1 >> 7) & 0b01 = TFB's Direct teaching button is pressed (information_chunk_1 >> 30) & 0b01 = Program Load state (Whenever the Program load process is successful, 1 and 0 are continuously converted.) (information_chunk_1 >> 31) & 0b01 = Program Transmit state (via TCP/IP Tablet UI, not for user)

float reserved_1[6]
Reserved / Not used

int jnt_info[6]
Basic state of each joint. 0 = Base / 1 = Shoulder / 2 = Elbow / 3 = Wrist1 / 4 = Wrist2 / 5 = Wrist3 Each int (4byte) consists of a combination of bits. (jnt_info[#] >> 0) & 0b01 = Joint #'s FET state (jnt_info[#] >> 1) & 0b01 = Joint #'s RUN state (jnt_info[#] >> 2) & 0b01 = Joint #'s INIT state (jnt_info[#] >> 3) & 0b01 = Joint #'s MODE state (jnt_info[#] >> 4) & 0b01 = Joint #'s encoder state (Nonius err) (jnt_info[#] >> 5) & 0b01 = Joint #'s encoder state (LowBatt err) (jnt_info[#] >> 6) & 0b01 = Joint #'s encoder state (Calibration mode) (jnt_info[#] >> 7) & 0b01 = Joint #'s encoder state (Multi-turn err) (jnt_info[#] >> 8) & 0b01 = Joint #'s Error state (JAM err) (jnt_info[#] >> 9) & 0b01 = Joint #'s Error state (CUR err) (jnt_info[#] >> 10) & 0b01 = Joint #'s Error state (BIG err) (jnt_info[#] >> 11) & 0b01 = Joint #'s Error state (INP err) (jnt_info[#] >> 12) & 0b01 = Joint #'s Error state (FLT err) (jnt_info[#] >> 13) & 0b01 = Joint #'s Error state (TMP err) (jnt_info[#] >> 14) & 0b01 = Joint #'s Error state (PS1 err) (jnt_info[#] >> 15) & 0b01 = Joint #'s Error state (PS2 err) **bits 16 ~ 31 are reserved** **Ex)** In position control mode: RUN = 1 / MODE = 0 In direct teaching (current control mode): RUN = 0 / MODE = 1

int collision_detect_onoff
Out collision detection On/Off State (1=On / 0 = Off)

int is_free_drive_mode
Free-drive (Gravity-compensation) On/Off State (1=On / 0 = Off)

int real_vs_simulation_mode
Mode of operation: Simulation mode=1 / Real Robot mode=0

int init_state_info
Robot arm activation (Initialization) stage info (0 -> 6) 0: default 1: Power check 2: Device check 3: Servo Initialization check 4: Parameter check 5: Payload check 6: Activation done

int init_error (Not for user)
Error code during the arm activation (return value for UI)

float tfb_analog_in[2]
Robot-Tool-Flange analog input measurement information of each channel (unit: Voltage) Channel number: 0~1

int tfb_digital_in[2]
Robot-Tool-Flange digital input measurement information of each channel (value: 0 or 1) Channel number: 0~1

int tfb_digital_out[2]
Robot-Tool-Flange digital output information of each channel (value: 0 or 1) Channel number: 0~1

float tfb_voltage_out
Robot-Tool-Flage output voltage level (unit: Voltage)

int op_stat_collisioin_occur
Whether out-collision is detected (0 or 1)

int op_stat_sos_flag
Robot Arm device error code during operation. 0 = None / 1=Encoder err (PVL) / 2=CPU err / 3=Big err / 4=Input err / 5=JAM err / 6 = Over current err / 7 = Position bound err / 8 = Mode err / 9 = Match err / 10 = Over current/Low voltage err / 11 = Temperature err / 12 = Speed over err

int op_stat_self_collision
Whether self-collision is detected (0 or 1)

int op_stat_soft_estop_occur
Pause state flag (0 or 1)

int op_stat_ems_flag
Software (kinematics) emergency stop situation 0 = None / 1 = Arm Stretch / 2= Cartesian Limit / 3=Joint Limit / 4=Un-solvable

int information_chunk_2
Information chunk to deliver various state information. It consists of a combination of bits. (information_chunk_2 >> 0) & 0b11 = Config digital input 16 (0 or 1) (Not for user) (information_chunk_2 >> 2) & 0b1111111111111111 = Target welding voltage * 100

int information_chunk_3
Information chunk to deliver various state information. It consists of a combination of bits. (information_chunk_3 >> 0) & 0b11 = Config digital input 17 (0 or 1) (Not for user)

int inbox_trap_flag[2]
Whether or not detected by the Inbox # check-function. # = In Box number: 0 or 1

int inbox_check_mode[2]
Check-function mode of Inbox #. # = In Box number: 0 or 1 0 = None / 1 = Check Tool Flange center / 2 = Check TCP / 3 = Check Tool Box / 4 = Check all

float eft_fx, eft_fy, eft_fz, eft_mx, eft_my, eft_mz
External F/T (force/torque) sensor value Fx, Fy, Fz (unit: N) Mx, My, Mz (unit: Nm)

int information_chunk_4
Information chunk to deliver various state information. It consists of a combination of bits. (information_chunk_4 >> 0) & 0b11 = No-Arc Function On/Off (0 or 1) (information_chunk_4 >> 2) & 0b111111 = Selected Tool List number (information_chunk_4 >> 8) & 0b11 = External Joint (External axis) Jog On/Off (0 or 1) (information_chunk_4 >> 10) & 0b01 = Tool Flange Digital Input 2 (information_chunk_4 >> 11) & 0b01 = Tool Flange Digital Input 3 (information_chunk_4 >> 12) & 0b01 = Tool Flange Digital Input 4 (information_chunk_4 >> 13) & 0b01 = Tool Flange Digital Input 5 (information_chunk_4 >> 14) & 0b01 = Arc Light On state (Not for user) (information_chunk_4 >> 15) & 0b111111111111 = Target welding current * 10 (information_chunk_4 >> 28) & 0b11 = Target welding voltage option (0 or 1)

float extend_io1_analog_in[4]
Extended I/O board analog input measurement information of each channel (unit: Voltage) Channel number: 0~3

float extend_io1_analog_out[4]
Extended I/O board analog output information of each channel (unit: Voltage) Channel number: 0~3

unsigned int extend_io1_digital_info
Extended I/O board digital input/output information It consists of a combination of bits. (extend_io1_digital_info >> 0) & 0b01 = Extend I/O digital input # 0 (extend_io1_digital_info >> 1) & 0b01 = Extend I/O digital input # 1 .. (extend_io1_digital_info >> 15) & 0b01 = Extend I/O digital input # 15 (extend_io1_digital_info >> 16) & 0b01 = Extend I/O digital output # 0 (extend_io1_digital_info >> 17) & 0b01 = Extend I/O digital output # 1 .. (extend_io1_digital_info >> 31) & 0b01 = Extend I/O digital input # 15

float aa_joint_ref[6]
Reference angle of each external-joint (auxiliary joint). (unit: degree) External joint number: 0~5

unsigned int safety_board_stat_info (Not for user)
Data chunk about the control box safety board

WE
TOUCH
THE CORE .

- Rainbow Robotics Research Center -