RML Protocol

Robot Metabolism Network Communication Protocol

Authors: Riyaan Bakhda , Philippe Martin Wyder

Robot Link States

State	Description	Behavior
B	Busy	During boot & connecting process
N	Establish Server Connection	Establish Server Connection
H	Handshake	Hello State / Send Hello Message
R	Ready	Link is idle
C	Calibrating	Link is executing a calibration sequence
P	Position / Velocity command mode	Link is executing a position / velocity command
W	Walk (crawl mode)	Link is executing the inch-worm gate
S	Sinusoidal Actuation	Link is executing a sinusoidal function
L	List (processing command list)	Link is following/repeating a list of commands

Package Types

To	Type	Description	Effect	Bytes
SRV	H	Hello	Establishes connection with SRV via handshake.	2
SRV	T	Time	Server Start Time (epoch)	8
SRV	U	Update	Status update sent by link if something changed: pos, battery status, etc.	8
SRV	C	Calibrate Confirmation	Sends Calibration values	24
LNK	C	Calibrate Command	Initiates calibration sequence	0
LNK	L	List Command	Sends list of commands [pos, speed, start_time] + repeat_flag	<256
LNK	P	Position/Vel Command	Sets link into P-state and updates position targets and velocity	6
LNK	S	Sinusoidal Function	Sets link in S-state and updates sin-function parameters	14
LNK	W	Walk Command	Sets link in W-state and set # crawl steps	4

Package Format

1 byte	1 byte	~5 < 256 bytes	2 byte
Length	Type	Data	Checksum
Header		**Body**	**Footer**

Packages

Header Data - Size 2 bytes

```
union MSG_hdr{ //(use ntohl to fix endian byte ordering)
    struct HDR_DATA{
        unsigned char length; // length of body only
        char type;
    }get;
    uint8_t bytes[2];
}msg_hdr;
```

Footer - Size 2 bytes

```
union MSG_ftr{
    unsigned short checksum_bytes;
    uint8_t bytes[2];
} msg_ftr;
```

Hello Package - Size 10 bytes

```
union MSG_hello{
    struct MSG_HELLO_DATA{
        unsigned char device_id;
        unsigned char MAX_VEL;       // MAX_VEL * 10
    } get;
    uint8_t bytes[2];
} msg_hello;
```

MSG_Time Package - Size 8 bytes

```
union MSG_epoch{
    struct MSG_TIME_DATA{
        time_t server_start; // server start time
    } get;
    uint8_t bytes[8];
} msg_epoch;
```

MSG_position Package - Size 6 bytes

```
union MSG_position{
    struct MSG_POSITION_DATA{
        unsigned char srv0_pos;
        unsigned char srv1_pos;
        unsigned char srv0_vel; // 0 - MAX_VEL*10[mm/s]
        unsigned char srv1_vel; // 0 - MAX_VEL*10[mm/s]
```

```
        unsigned short start_time; // unix start time (8-byte: unsigned long)
    }get;
    uint8_t bytes[6];
}msg_position;
```

MSG_update Package - Size 8 bytes

```
union MSG_update{
    struct MSG_UPDATE_DATA{
        char device_status;
        unsigned char srv0_pos; // position in mm [0 - STROKE_LENGTH]
        unsigned char srv1_pos;
        unsigned char srv0_raw;
        unsigned char srv1_raw;
        unsigned char bat_status;
        unsigned char srv0_vel; // 0 - MAX_VEL*10[mm/s]
        unsigned char srv1_vel; // 0 - MAX_VEL*10[mm/s]
        unsigned short current_command_checksum;
    } get;
    uint8_t bytes[12];
} msg_update;
```

MSG_Calibrate Package - Size 0 bytes

```
// package without a body
```

MSG_Calibration_Conf - Size 24 bytes

```
union MSG_calibration_conf {
      struct SRV_CALIBRATE_DATA {
          uint8_t version;
          unsigned short srv0_min_ms, srv0_max_ms; // PWM value min/max for servo
                                                   // position change
          unsigned short srv1_min_ms, srv1_max_ms;  // same as above for servo 1
          unsigned short srv0_raw_min, srv0_raw_max; // raw position feedback at
                                                     // fully contracted/expanded location
          unsigned short srv1_raw_min, srv1_raw_max; // same as above for servo 1
      } get;
      uint8_t bytes[24];
}msg_calibration;
```

MSG_sin Package - Size 4 bytes

```
union MSG_sin{
    struct MSG_SIN_DATA{
```

```
        unsigned short start_time;  // 2 bytes - seconds relative to SERVER_EPOCH

        unsigned char a0;           // Amplitude (scales the function)
        unsigned char x0;           // base-position
        short phase_shift_0;        // milliseconds
        unsigned short period_0;    // milliseconds for complete

        unsigned char a1;
        unsigned char x1;
        short phase_shift_1;
        unsigned short period_1;

    } get;
    uint8_t bytes[14];
} msg_sin;
```

MSG_Walk Package - Size 4 byte

```
union MSG_walk{
    struct MSG_WALK_DATA{
        char nr_steps; // -128 (backwards) to 127 (forwards)
        unsigned short start_time; // unix start time (8-byte int: unsigned long)
    }get;
    uint8_t bytes[4];
}msg_walk;
```

List Packages - Actual and Subheader

```
union MSG_list_actual{
    struct MSG_LIST_ACTUAL{
        short nr_repeat;
        unsigned short start_time;
    } get;
    uint8_t bytes[256];
} msg_list_actual;

union MSG_list_subheader{
    struct MSG_LIST_SUBHEADER{
        unsigned char duration;    // seconds
        char type;
    } get;
    uint8_t bytes[2];
} msg_list_subheader;
```