7Bot Communication Instruction

Communication Protocol

- Protocol Type: UART
- Baud Rate: 115200
- characteristic: 8 data bits, no parity, one stop bit

Communication Format

Byte 1 (Beginning Flag)	Byte 2 (Instruction Type)	Other Bytes (Data)
0xFE	0xF1~0xFC	0~127 (0x00~0x7F)

Send data to 7Bot

Set Motor Status: 7Bot supports 3 kinds of motor statuses. Normal Servo Status can change to Forceless Status & Protection Status, and vice versa. But Forceless Status & Protection Status can not change from each other.

Byte 1 (Beginning Flag)	Byte 2 (Instruction Type)	Byte 3 (Status)
0xFE	0xF5	0-forceless, 1-normal servo(default), 2-protection

Set Fluency & Speed: Fluency means the motor motions have acceleration and deceleration process, which makes motions much more fluent.

Byte 1 (Beginning Flag)	0xFE
Byte 2 (Instruction Type)	0xF7
Byte 3 (motor 0 data)	bit 6: 0-disable fluency, 1-enable fluency; bit 5~0: speed value(range:0~25, 10 means 100 degrees per second)
Byte 4 (motor 1 data)	bit 6: 0-disable fluency, 1-enable fluency; bit 5~0: speed value(range:0~25, 10 means 100 degrees per second)
Byte 5 (motor 2 data)	bit 6: 0-disable fluency, 1-enable fluency; bit 5~0: speed value(range:0~25, 10 means 100 degrees per second)
Byte 6 (motor 3 data)	bit 6: 0-disable fluency, 1-enable fluency; bit 5~0: speed value(range:0~25, 10 means 100 degrees per second)
Byte 7 (motor 4 data)	bit 6: 0-disable fluency, 1-enable fluency; bit 5~0: speed value(range:0~25, 10 means 100 degrees per second)
Byte 8 (motor 5 data)	bit 6: 0-disable fluency, 1-enable fluency; bit 5~0: speed value(range:0~25, 10 means 100 degrees per second)
Byte 9 (motor 6 data)	bit 6: 0-disable fluency, 1-enable fluency; bit 5~0: speed value(range:0~25, 10 means 100 degrees per second)

Set Motor Position: Each motor can rotate 180 degrees in maximum. Using 10 bits data (0~1000) to represent rotational angle(0~180°). So the controlling resolution is 0.18°

Byte 1 (Beginning Flag)	0xFE
Byte 2 (Instruction Type)	0xF9
Byte 3 (motor 0 data)	**[Hight Byte]** bit 2~0: 3 high bits of Motor Position
Byte 4 (motor 0 data)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position
Byte 5 (motor 1 data)	**[Hight Byte]** bit 2~0: 3 high bits of Motor Position
Byte 6 (motor 1 data)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position
Byte 7 (motor 2 data)	**[Hight Byte]** bit 2~0: 3 high bits of Motor Position
Byte 8 (motor 2 data)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position
Byte 9 (motor 3 data)	**[Hight Byte]** bit 2~0: 3 high bits of Motor Position
Byte 10 (motor 3 data)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position
Byte 11 (motor 4 data)	**[Hight Byte]** bit 2~0: 3 high bits of Motor Position
Byte 12 (motor 4 data)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position
Byte 13 (motor 5 data)	**[Hight Byte]** bit 2~0: 3 high bits of Motor Position
Byte 14 (motor 5 data)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position
Byte 15 (motor 6 data)	**[Hight Byte]** bit 2~0: 3 high bits of Motor Position
Byte 16 (motor 6 data)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position

IK6: Given joint6 position in 7Bot cartesian coordinate system and 2 direction vector vec56(from joint5 to joint6) and vec67(from joint6 to joint7). The IK algorithm will calculate and set the positions from motor0 to motor5. And motor6 position should be given here.

Byte 1 (Beginning Flag)	0xFE
Byte 2 (Instruction Type)	0xFA
Byte 3 (joint6.x)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 4 (joint6.x)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 5 (joint6.y)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 6 (joint6.y)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 7 (joint6.z)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 8 (joint6.z)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 9 (vec56.x)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 10 (vec56.x)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 11 (vec56.y)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 12 (vec56.y)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 13 (vec56.z)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 14 (vec56.z)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 15 (vec67.x)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 16 (vec67.x)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 17 (vec67.y)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 18 (vec67.y)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 19 (vec67.z)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 20 (vec67.z)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 21 (motor 6)	**[Hight Byte]** bit 2~0: 3 high bits of Motor Position
Byte 22 (motor 6)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position

IK5: Given joint6 position in 7Bot cartesian coordinate system and 1 direction vector vec56(from joint5 to joint6). The IK algorithm will calculate and set the positions from motor0 to motor4. And motor5 to motor6 positions should be given here.

Byte 1 (Beginning Flag)	0xFE
Byte 2 (Instruction Type)	0xFB
Byte 3 (joint6.x)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 4 (joint6.x)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 5 (joint6.y)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 6 (joint6.y)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 7 (joint6.z)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 8 (joint6.z)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 9 (vec56.x)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 10 (vec56.x)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 11 (vec56.y)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 12 (vec56.y)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 13 (vec56.z)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 14 (vec56.z)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 15 (motor 5)	**[Hight Byte]** bit 2~0: 3 high bits of Motor Position
Byte 16 (motor 5)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position
Byte 17 (motor 6)	**[Hight Byte]** bit 2~0: 3 high bits of Motor Position
Byte 18 (motor 6)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position

IK3: Given joint5 position in 7Bot cartesian coordinate system. The IK algorithm will calculate and set the positions from motor0 to motor2. And motor3 to motor6 positions should be given here.

Byte 1 (Beginning Flag)	0xFE
Byte 2 (Instruction Type)	0xFC
Byte 3 (joint5.x)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 4 (joint5.x)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 5 (joint5.y)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 6 (joint5.y)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 7 (joint5.z)	**[Hight Byte]** bit 3: 0-Positive Value, 1-Negative Value; bit 2~0: 3 high bits of this Value
Byte 8 (joint5.z)	**[Low Byte]** bit 6~0: 7 low bits of this Value
Byte 11 (motor 3)	**[Hight Byte]** bit 2~0: 3 high bits of Motor Position
Byte 12 (motor 3)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position
Byte 13 (motor 4)	**[Hight Byte]** bit 2~0: 3 high bits of Motor Position
Byte 14 (motor 4)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position
Byte 15 (motor 5)	**[Hight Byte]** bit 2~0: 3 high bits of Motor Position
Byte 16 (motor 5)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position
Byte 17 (motor 6)	**[Hight Byte]** bit 2~0: 3 high bits of Motor Position
Byte 18 (motor 6)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position

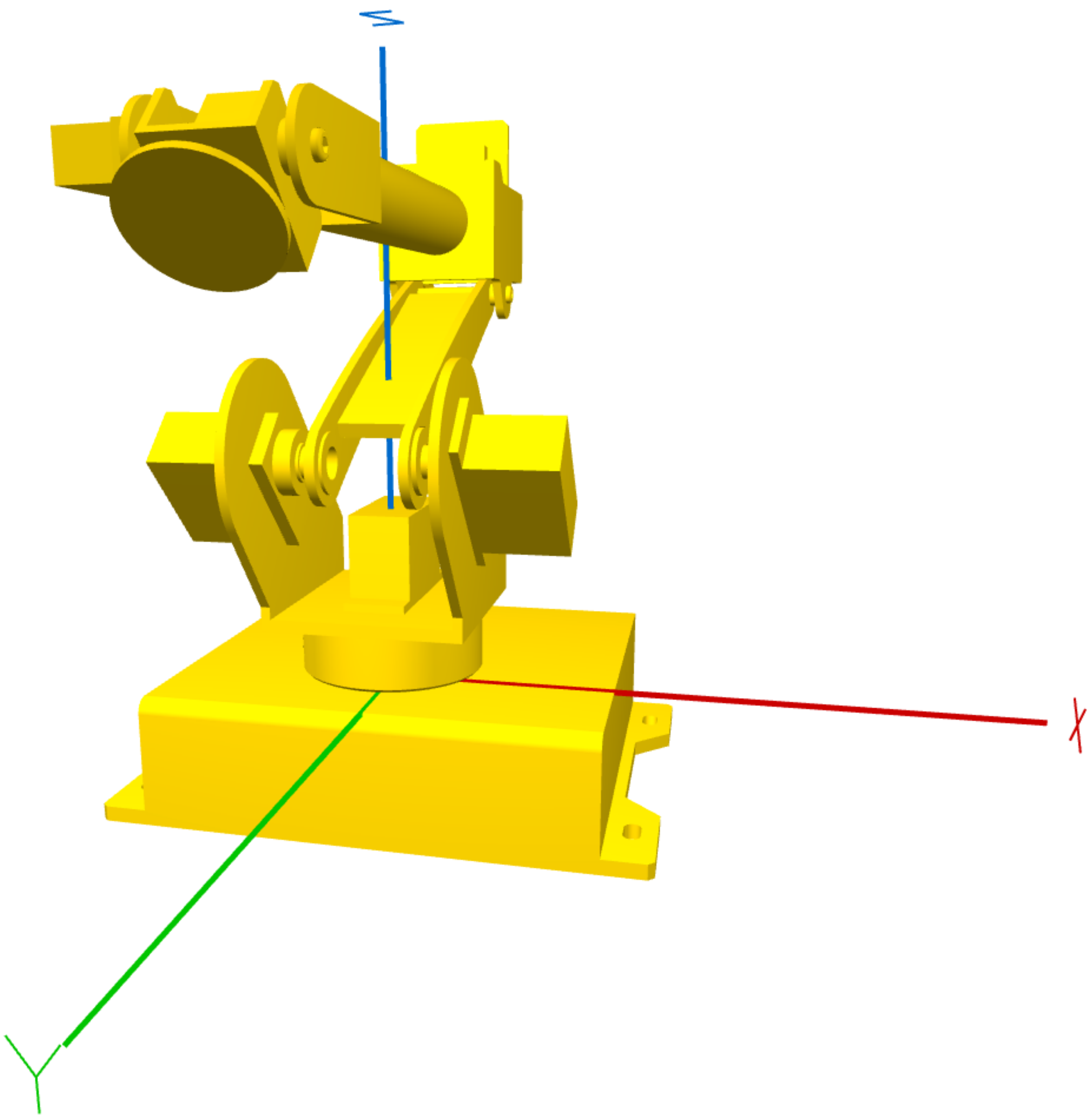
Get data from 7Bot

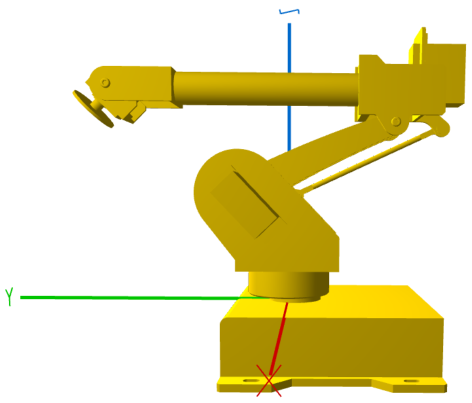
Feedback: Each motor can rotate 180 degrees in maximum. Using 10 bits data (0~1000) to represent rotational angle(0~180°). Force values range from -7 to 7.

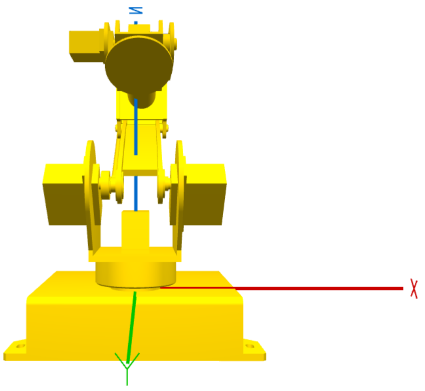
Byte 1 (Beginning Flag)	0xFE
Byte 2 (Instruction Type)	0xF9
Byte 3 (motor 0 data)	**[Hight Byte]** bit 6: force direction (0-positive, 1-negative) bit 5~3: force level (0~7) bit 2~0: 3 high bits of Motor Position
Byte 4 (motor 0 data)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position
Byte 5 (motor 1 data)	**[Hight Byte]** bit 6: force direction (0-positive, 1-negative) bit 5~3: force level (0~7) bit 2~0: 3 high bits of Motor Position
Byte 6 (motor 1 data)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position
Byte 7 (motor 2 data)	**[Hight Byte]** bit 6: force direction (0-positive, 1-negative) bit 5~3: force level (0~7) bit 2~0: 3 high bits of Motor Position
Byte 8 (motor 2 data)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position
Byte 9 (motor 3 data)	**[Hight Byte]** bit 6: force direction (0-positive, 1-negative) bit 5~3: force level (0~7) bit 2~0: 3 high bits of Motor Position
Byte 10 (motor 3 data)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position
Byte 11 (motor 4 data)	**[Hight Byte]** bit 6: force direction (0-positive, 1-negative) bit 5~3: force level (0~7) bit 2~0: 3 high bits of Motor Position
Byte 12 (motor 4 data)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position
Byte 13 (motor 5 data)	**[Hight Byte]** bit 6: force direction (0-positive, 1-negative) bit 5~3: force level (0~7) bit 2~0: 3 high bits of Motor Position
Byte 14 (motor 5 data)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position
Byte 15 (motor 6 data)	**[Hight Byte]** bit 6: force direction (0-positive, 1-negative) bit 5~3: force level (0~7) bit 2~0: 3 high bits of Motor Position
Byte 16 (motor 6 data)	**[Low Byte]** bit 6~0: 7 low bits of Motor Position
Byte 16 (converge flag)	0-not converge(motors still moving); 1-converge(motors finish moving)

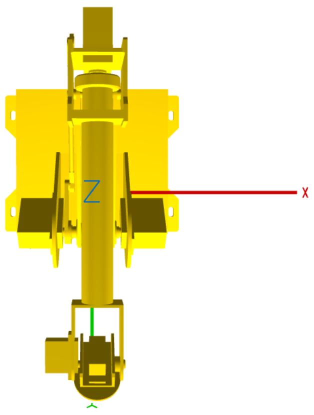
Appendix

Cartesian coordinate system of 7Bot

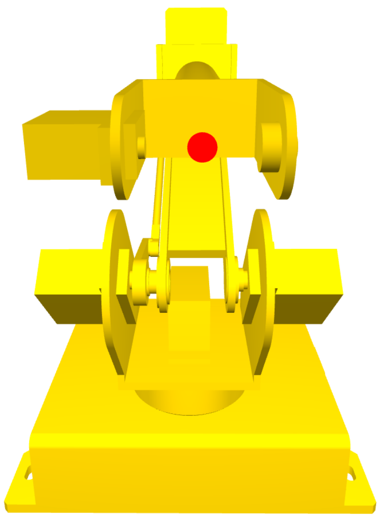


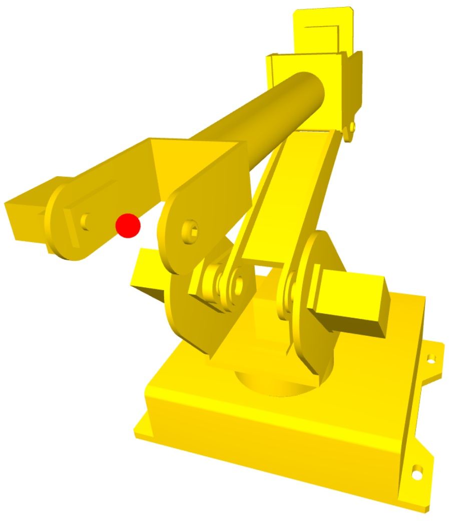






Position of **joint5**





Position of **joint6** & **joint7**