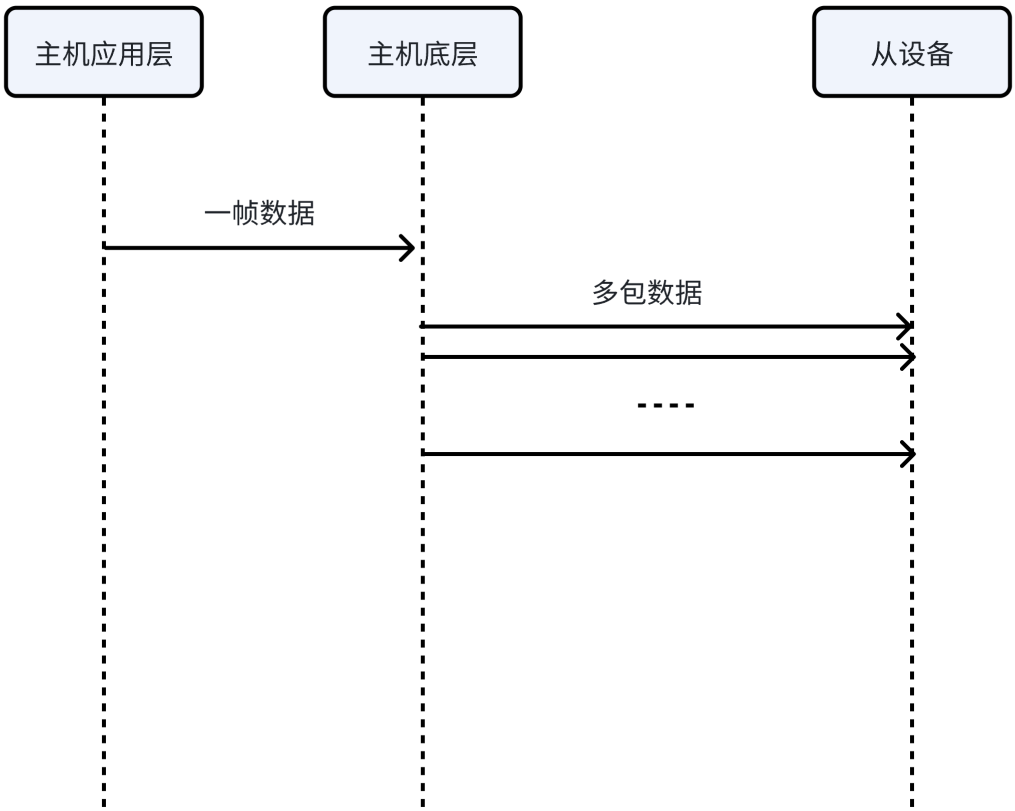
协议

1. 协议概述

由于CAN协议自身限制，CAN每包数据最长8个字节，即如果想要发送大于8个字节的数据帧，需要发送多包数据。

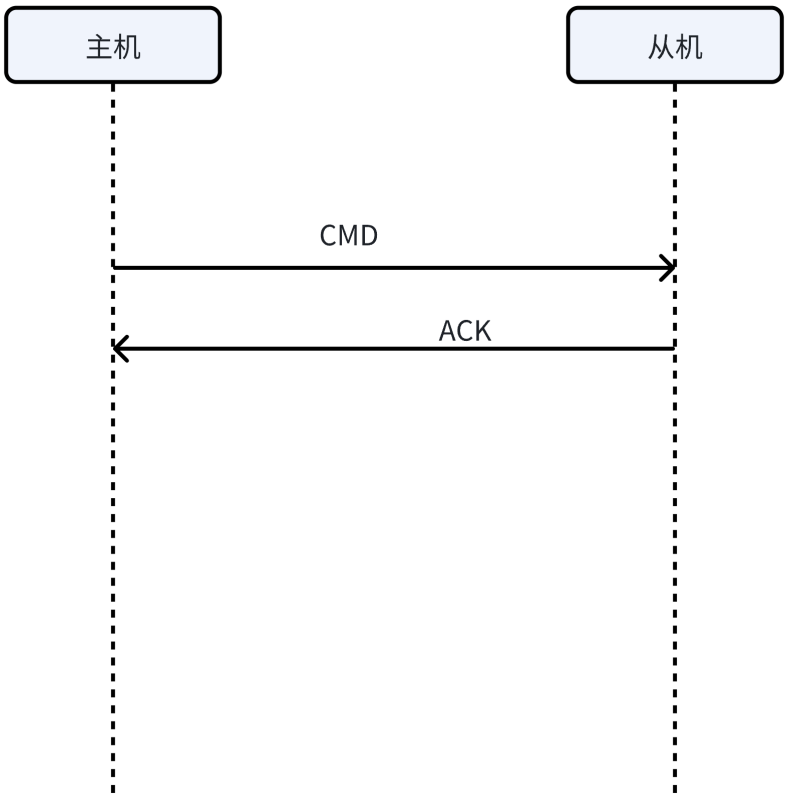
所以协议上，底层逻辑负责发送分包，和接收组包，对于上层来说，单包最大长度为255*8，目前人为限制(512个字节)



2. 交互模型

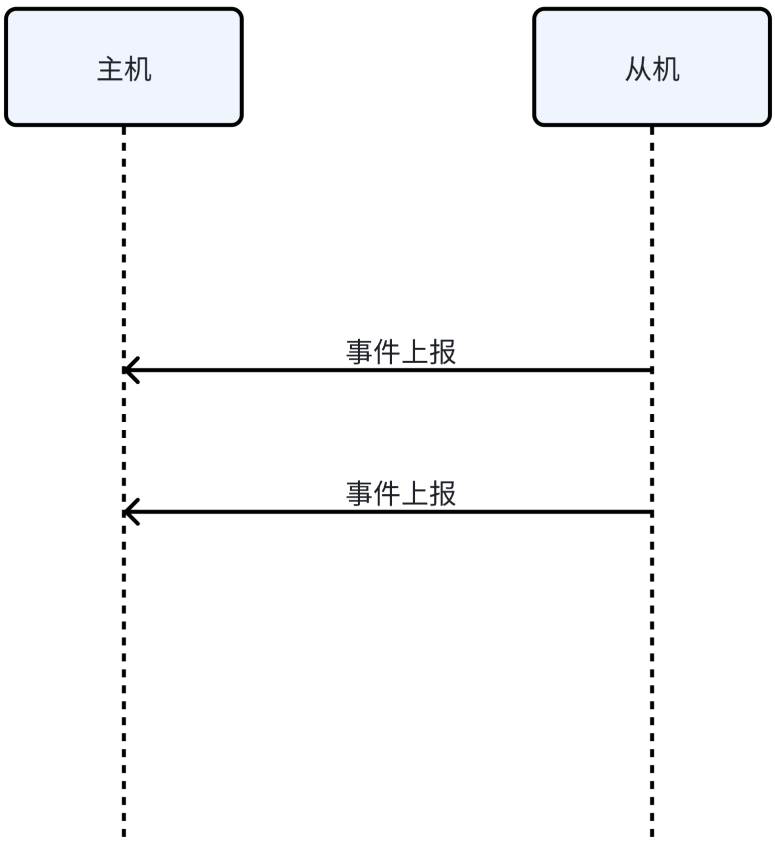
2.1 一问一答

主机下发指令，从机接收到指令，发送hui主机，CMD和ACK通过**消息ID**进行匹配。



2.2 事件模型

部分子模块支持事件上报，上报事件的消息ID为从机自增。



3. 一包数据格式

上层应用无需关心

```
1. can协议使用扩展帧
```

CAN仲裁段					CAN数据段
1bit	4bit	8bit	8bit	8bit	byte0->byte7
1/*写1*/	优先级	from	frameNum	frameId	存放包数据

4. 一帧数据格式

4.1 协议格式

字节偏移	0	2	4	5	6	...
字节长度	2	2	1	1	2	...
含义	packetindex	main_cmd_id	sub_cmd_id	指令类型	moduleid	数据

4.2 cmd_id-list

```
PS:
    CMDID(1, 0) //1对应着main_cmd_id，0对应着sub_cmd_id
```

```
  kmodule_ping                                  = CMDID(1, 0),
  kmodule_stop                                  = CMDID(1, 1),
  kmodule_break                                 = CMDID(1, 2),
  kmodule_get_last_exec_status                  = CMDID(1, 3),
  kmodule_get_status                            = CMDID(1, 4),
  kmodule_set_reg                               = CMDID(1, 5),
  kmodule_get_reg                               = CMDID(1, 6),
  kmodule_readio                                = CMDID(1, 7),
```

```
    kmodule_writeio                                          = CMDID(1, 8),
    kmodule_read_adc                                         = CMDID(1, 9),
    kmodule_get_error                                        = CMDID(1, 10),
    kmodule_clear_error                                      = CMDID(1, 11),
    kmodule_set_inited_flag                                  = CMDID(1, 12),
    kmodule_get_inited_flag                                  = CMDID(1, 13),
    kmodule_factory_reset                                    = CMDID(1, 14),
    kmodule_flush_cfg                                        = CMDID(1, 15),
    kmodule_active_cfg                                       = CMDID(1, 16),
    kmodule_read_raw                                         = CMDID(1, 19),
    kmodule_enable                                           = CMDID(1, 20),
    kmodule_start                                            = CMDID(1, 21),
    kmotor_enable                                            = CMDID(2, 1),
    kmotor_rotate                                            = CMDID(2, 2),
    kmotor_move_by                                           = CMDID(2, 3),
    kmotor_move_to                                           = CMDID(2, 4),
    kmotor_rotate_acctime                                    = CMDID(2, 5),
    kmotor_move_by_acctime                                   = CMDID(2, 6),
    kmotor_move_to_acctime                                   = CMDID(2, 7),
    kmotor_rotate_with_torque                                = CMDID(2, 8),
    kmotor_move_to_zero_forward                              = CMDID(2, 9),
    kmotor_move_to_zero_backward                             = CMDID(2, 10),
    kmotor_read_pos                                          = CMDID(2, 11),
    kmotor_set_current_pos_by_change_shift                   = CMDID(2, 12),
    kmotor_motor_move_to_zero_forward_and_calculated_shift   = CMDID(2, 13),
    kmotor_motor_move_to_zero_backward_and_calculated_shift  = CMDID(2, 14),
    kmotor_move_to_torque                                    = CMDID(2, 15),
    kmotor_calculated_pos_by_move_to_zero                    = CMDID(2, 16),
    kmotor_easy_rotate                                       = CMDID(2, 17),
    kmotor_easy_move_by                                      = CMDID(2, 18),
    kmotor_easy_move_to                                      = CMDID(2, 19),
    kmotor_easy_move_to_zero                                 = CMDID(2, 20),
    kmotor_easy_set_current_pos                              = CMDID(2, 21),
    kxymotor_enable                                          = CMDID(3, 1),
    kxymotor_move_by                                         = CMDID(3, 2),
    kxymotor_move_to                                         = CMDID(3, 3),
    kxymotor_move_to_zero                                    = CMDID(3, 4),
    kxymotor_move_to_zero_and_calculated_shift               = CMDID(3, 5),
    kxymotor_read_pos                                        = CMDID(3, 6),
    kxymotor_calculated_pos_by_move_to_zero                  = CMDID(3, 7),
    kcode_scaner_start_scan                                  = CMDID(4, 1),
    kcode_scaner_stop_scan                                   = CMDID(4, 2),
    kcode_scaner_read_scaner_result                          = CMDID(4, 3),
    kpipette_ctrl_init_device                                = CMDID(5, 1),
    kpipette_ctrl_put_tip                                    = CMDID(5, 2),
    kpipette_ctrl_move_to_ul                                 = CMDID(5, 3),
```

4.3 cmd_id-helpinfo

```
module_active_cfg (mid)
module_break (mid)
module_clear_error (mid)
module_enable (mid, enable)
module_factory_reset (mid)
module_flush_cfg (mid)
module_get_error (mid)
module_get_inited_flag (mid)
module_get_last_exec_status (mid)
module_get_reg (mid, param_id)
module_get_status (mid)
module_ping (mid)
module_read_adc (mid,adc_id, adcindex)
module_read_raw (mid, startindex)
module_readio (mid)
module_set_inited_flag (mid, flag)
module_set_reg (mid, param_id, param_value)
module_start (mid)
module_stop (mid)
module_writeio (mid,ioindex,io)
motor_calculated_pos_by_move_to_zero (mid)
motor_easy_move_by (mid, distance)
motor_easy_move_to (mid, position)
motor_easy_move_to_zero (mid, direction)
motor_easy_rotate (mid, direction)
motor_easy_set_current_pos (mid, pos)
motor_enable (mid, enable)
motor_move_by (mid, distance, motor_velocity, acc)
motor_move_by_acctime (mid, distance, motor_velocity, acctime)
motor_move_to (mid, position, motor_velocity, acc)
motor_move_to_acctime (mid, position, motor_velocity, acctime)
motor_move_to_torque (mid, pos, torque, overtime)
motor_move_to_zero_backward (mid, findzerospeed, findzeroedge_speed, acc, overti
motor_move_to_zero_backward_and_calculated_shift (mid, findzerospeed, findzeroed
motor_move_to_zero_forward (mid, findzerospeed, findzeroedge_speed, acc, overtim
motor_move_to_zero_forward_and_calculated_shift (mid, findzerospeed, findzeroedg
motor_read_pos (mid)
motor_rotate (mid, direction, motor_velocity, acc)
motor_rotate_acctime (mid, direction, motor_velocity, acctime)
motor_rotate_with_torque (mid, pos, torque)
motor_set_current_pos_by_change_shift (mid, pos)

pipette_ctrl_init_device (mid)
pipette_ctrl_move_to_ul (mid, ul)
```

```
pipette_ctrl_put_tip (mid)

xymotor_calculated_pos_by_move_to_zero (mid)
xymotor_enable (mid, enable)
xymotor_move_by (mid, dx, dy, motor_velocity)
xymotor_move_to (mid, x, y, motor_velocity)
xymotor_move_to_zero (mid)
xymotor_move_to_zero_and_calculated_shift (mid)
xymotor_read_pos (mid)
```

4.4 指令类型

指令类型	
cmd	0
ack	1
error_ack	2
event	3

4.5 CMD指令的数据格式

为了简化协议，牺牲数据部分传输消息，每条指令的每一个参数均为int32_t数据格式，数据传输采用小端数据传输。

例如机械臂移动指令**xymotor_move_to (mid, x, y, motor_velocity)**，指令组包为

字节偏移	0	2	4	5	6	8	12	16
字节长度	2	2	1	1	2	4	4	4
含义	packetindex	main_cmd_id	sub_cmd_id	指令类型	moduleid	x	y	v
	0	3	3	0(cmd)	1(模块ID)			

4.6 ACK的数据格式

ACK的协议格式和4.1是一致的，大多数的指令，是没有应答数据的，小部分读指令是存在应答数据的。一般来说无特殊说明，应答数据格式依然是int32_t。

例如**module_get_reg，**指令回执格式为

字节偏移	0	2	4	5	6	8
字节长度	2	2	1	1	2	4
含义	packetindex	main_cmd_id	sub_cmd_id	指令类型	moduleid	regval
	0	1	6	1(ack)	1(模块ID)	

4.7 ACK的数据格式列表

	ACK列表
module_get_error	int32_t
module_get_inited_flag	int32_t
module_get_last_exec_status	int32_t
module_get_reg	int32_t
module_get_status	int32_t
module_read_raw	uint8_t[]

4.8 ErrorACK的数据格式

字节偏移	0	2	4	5	6	8
字节长度	2	2	1	1	2	4
含义	packetindex	main_cmd_id	sub_cmd_id	指令类型	moduleid	errorval
	0	1	6	2(error_ack)	1(模块ID)	

4.9 event的数据格式

event的数据格式是不同，目前支持两种事件，一种是寄存器修改事件，一种是模块致命错误上报事件。

事件协议的数据格式根据**main_cmd_id**和**sub_cmd_id**进行区分

```
kevent_bus_reg_change_report  = CMDID(0, 100)
kevent_bus_module_fatal_error = CMDID(0, 101)
```

4.9.1 reg_change_report

字节偏移	0	2	4	5	6	8	12	16
字节长度	2	2	1	1	2	4	4	4
含义	packetindex	main_cmd_id	sub_cmd_id	指令类型	moduleid	regIndex	reg_old_val	reg_new_val
	0	0	100	3(event)	1(模块ID)			

4.9.2 module_fatal_error

字节偏移	0	2	4	5	6	8
字节长度	2	2	1	1	2	4
含义	packetindex	main_cmd_id	sub_cmd_id	指令类型	moduleid	errorval
	0	0	101	3(event)	1(模块ID)	

5. 附录

5.1 REG

```
#pragma once
#include <stdint.h>

namespace iflytop {
using namespace std;

#define REG_INDEX(type, offset, subconfigindex) (type * 100 + offset + subconfig
typedef enum {
  /*****************************************************************************
   *                                模块通用配置和状态
   *****************************************************************************/
  kreg_module_version             = REG_INDEX(0, 0, 0),    // 模块版本
  kreg_module_type                = REG_INDEX(0, 0, 1),    // 模块类型
```

```
14  kreg_module_status                = REG_INDEX(0, 0, 2),   // 0idle,1busy,2error
15  kreg_module_errorcode             = REG_INDEX(0, 0, 3),   // inited_flag
16  kreg_module_initflag              = REG_INDEX(0, 0, 4),   // inited_flag
17  kreg_module_enableflag            = REG_INDEX(0, 0, 5),   // 0idle,1busy,2error
18  kreg_module_errorbitflag0         = REG_INDEX(0, 0, 6),   // 模块异常标志，bit，
19  kreg_module_errorbitflag1         = REG_INDEX(0, 0, 7),   //
20  kreg_module_input_state           = REG_INDEX(0, 0, 8),   //
21  kreg_module_output_state          = REG_INDEX(0, 0, 9),   //
22  kreg_module_raw_sector_size       = REG_INDEX(0, 0, 10),  // sector_size
23  kreg_module_raw_sector_num        = REG_INDEX(0, 0, 11),  //
24  kreg_module_is_online             = REG_INDEX(0, 0, 12),  //
25  kreg_module_last_cmd_exec_status = REG_INDEX(0, 0, 20),  // 上一条指令执行的状态
26  kreg_module_last_cmd_exec_val0    = REG_INDEX(0, 0, 21),  // 上一条指令执行的结果
27  kreg_module_last_cmd_exec_val1    = REG_INDEX(0, 0, 22),  // 上一条指令执行的结果.
28  kreg_module_last_cmd_exec_val2    = REG_INDEX(0, 0, 23),  // 上一条指令执行的结果.
29  kreg_module_last_cmd_exec_val3    = REG_INDEX(0, 0, 24),  // 上一条指令执行的结果.
30  kreg_module_last_cmd_exec_val4    = REG_INDEX(0, 0, 25),  // 上一条指令执行的结果.
31  kreg_module_last_cmd_exec_val5    = REG_INDEX(0, 0, 26),  // 上一条指令执行的结果.
32  kreg_module_private0              = REG_INDEX(0, 0, 30),  // 模块私有状态0
33  kreg_module_private1              = REG_INDEX(0, 0, 31),  // 模块私有状态1
34  kreg_module_private2              = REG_INDEX(0, 0, 32),  // 模块私有状态2
35  kreg_module_private3              = REG_INDEX(0, 0, 33),  // 模块私有状态3
36  kreg_module_private4              = REG_INDEX(0, 0, 34),  // 模块私有状态4
37  kreg_module_private5              = REG_INDEX(0, 0, 35),  // 模块私有状态5
38  kreg_module_private6              = REG_INDEX(0, 0, 36),  // 模块私有状态6
39  kreg_module_private7              = REG_INDEX(0, 0, 37),  // 模块私有状态7
40  kreg_module_private8              = REG_INDEX(0, 0, 38),  // 模块私有状态8
41  kreg_module_private9              = REG_INDEX(0, 0, 39),  // 模块私有状态9
42  kreg_module_do_action0            = REG_INDEX(0, 0, 40),  // 方法0
43  kreg_module_action_param1         = REG_INDEX(0, 0, 41),  // 方法1
44  kreg_module_action_param2         = REG_INDEX(0, 0, 42),  // 方法2
45  kreg_module_action_param3         = REG_INDEX(0, 0, 43),  // 方法3
46  kreg_module_action_param4         = REG_INDEX(0, 0, 44),  // 方法4
47  kreg_module_action_param5         = REG_INDEX(0, 0, 45),  // 方法5
48  kreg_module_action_param6         = REG_INDEX(0, 0, 46),  // 方法6
49  kreg_module_action_param7         = REG_INDEX(0, 0, 47),  // 方法7
50  kreg_module_action_param8         = REG_INDEX(0, 0, 48),  // 方法8
51  kreg_module_action_param9         = REG_INDEX(0, 0, 49),  // 方法9
52  kreg_module_action_ack1           = REG_INDEX(0, 0, 51),  // ACK1
53  kreg_module_action_ack2           = REG_INDEX(0, 0, 52),  // ACK2
54  kreg_module_action_ack3           = REG_INDEX(0, 0, 53),  // ACK3
55  kreg_module_action_ack4           = REG_INDEX(0, 0, 54),  // ACK4
56  kreg_module_action_ack5           = REG_INDEX(0, 0, 55),  // ACK5
57  kreg_module_action_ack6           = REG_INDEX(0, 0, 56),  // ACK6
58  kreg_module_action_ack7           = REG_INDEX(0, 0, 57),  // ACK7
59  kreg_module_action_ack8           = REG_INDEX(0, 0, 58),  // ACK8
60  kreg_module_action_ack9           = REG_INDEX(0, 0, 59),  // ACK9
```

```
61
62  /*****************************************************************************
63   *                                    SENSOR
64   *****************************************************************************
65  kreg_sensor_current                                    = REG_INDEX(1, 0, 0),   /
66  kreg_sensor_voltage                                    = REG_INDEX(1, 0, 1),   /
67  kreg_sensor_temperature                                = REG_INDEX(1, 0, 2),   /
68  kreg_sensor_humidity                                   = REG_INDEX(1, 0, 3),   /
69  kreg_sensor_wind_speed                                 = REG_INDEX(1, 0, 4),   /
70  kreg_sensor_wind_dir                                   = REG_INDEX(1, 0, 5),   /
71  kreg_sensor_air_press                                  = REG_INDEX(1, 0, 6),   /
72  kreg_sensor_pm25                                       = REG_INDEX(1, 0, 7),   /
73  kreg_sensor_pm10                                       = REG_INDEX(1, 0, 8),   /
74  kreg_sensor_co                                         = REG_INDEX(1, 0, 9),   /
75  kreg_sensor_co2                                        = REG_INDEX(1, 0, 10),  /
76  kreg_sensor_no2                                        = REG_INDEX(1, 0, 11),  /
77  kreg_sensor_so2                                        = REG_INDEX(1, 0, 12),  /
78  kreg_sensor_o3                                         = REG_INDEX(1, 0, 13),  /
79  kreg_sensor_light_intensity                            = REG_INDEX(1, 0, 14),  /
80  kreg_sensor_radiation                                  = REG_INDEX(1, 0, 15),  /
81  kreg_sensor_hydrogen_peroxide_volume                   = REG_INDEX(1, 0, 16),  /
82  kreg_sensor_h2o_h2o2_rs                                = REG_INDEX(1, 0, 17),  /
83  kreg_sensor_relative_humidity                          = REG_INDEX(1, 0, 18),  /
84  kreg_sensor_absolute_hydrogen_peroxide                 = REG_INDEX(1, 0, 19),  /
85  kreg_sensor_h2o_h2o2dew_point_temperature              = REG_INDEX(1, 0, 20),  /
86  kreg_sensor_water_volume                               = REG_INDEX(1, 0, 21),  /
87  kreg_sensor_water_vapor_pressure                       = REG_INDEX(1, 0, 22),  /
88  kreg_sensor_absolute_humidity                          = REG_INDEX(1, 0, 23),  /
89  kreg_sensor_water_vapor_saturation_pressure_h2o        = REG_INDEX(1, 0, 24),  /
90  kreg_sensor_h2o2_vapor_pressure                        = REG_INDEX(1, 0, 25),  /
91  kreg_sensor_water_vapor_saturation_pressure_h2o_h2o2 = REG_INDEX(1, 0, 26),  /
92
93  kreg_sensor_temperature0 = REG_INDEX(2, 0, 0),    // 温度1
94  kreg_sensor_temperature1 = REG_INDEX(2, 0, 1),    // 温度2
95  kreg_sensor_temperature2 = REG_INDEX(2, 0, 2),    // 温度3
96  kreg_sensor_temperature3 = REG_INDEX(2, 0, 3),    // 温度4
97  kreg_sensor_temperature4 = REG_INDEX(2, 0, 4),    // 温度5
98  kreg_sensor_temperature5 = REG_INDEX(2, 0, 5),    // 温度6
99  kreg_sensor_temperature6 = REG_INDEX(2, 0, 6),    // 温度7
100 kreg_sensor_temperature7 = REG_INDEX(2, 0, 7),    // 温度8
101 kreg_sensor_temperature8 = REG_INDEX(2, 0, 8),    // 温度9
102 kreg_sensor_temperature9 = REG_INDEX(2, 0, 9),    // 温度10
103 kreg_sensor_pressure0    = REG_INDEX(2, 10, 0),   // 压力0
104 kreg_sensor_pressure1    = REG_INDEX(2, 10, 1),   // 压力1
105 kreg_sensor_pressure2    = REG_INDEX(2, 10, 2),   // 压力2
106 kreg_sensor_pressure3    = REG_INDEX(2, 10, 3),   // 压力3
107 kreg_sensor_pressure4    = REG_INDEX(2, 10, 4),   // 压力4
```

```
108   kreg_sensor_pressure5    = REG_INDEX(2, 10, 5),  // 压力5
109   kreg_sensor_pressure6    = REG_INDEX(2, 10, 6),  // 压力6
110   kreg_sensor_pressure7    = REG_INDEX(2, 10, 7),  // 压力7
111   kreg_sensor_pressure8    = REG_INDEX(2, 10, 8),  // 压力8
112   kreg_sensor_pressure9    = REG_INDEX(2, 10, 9),  // 压力9
113   kreg_sensor_humidity0    = REG_INDEX(2, 20, 0),  // 湿度0
114   kreg_sensor_humidity1    = REG_INDEX(2, 20, 1),  // 湿度1
115   kreg_sensor_humidity2    = REG_INDEX(2, 20, 2),  // 湿度2
116   kreg_sensor_humidity3    = REG_INDEX(2, 20, 3),  // 湿度3
117   kreg_sensor_humidity4    = REG_INDEX(2, 20, 4),  // 湿度4
118   kreg_sensor_humidity5    = REG_INDEX(2, 20, 5),  // 湿度5
119   kreg_sensor_humidity6    = REG_INDEX(2, 20, 6),  // 湿度6
120   kreg_sensor_humidity7    = REG_INDEX(2, 20, 7),  // 湿度7
121   kreg_sensor_humidity8    = REG_INDEX(2, 20, 8),  // 湿度8
122   kreg_sensor_humidity9    = REG_INDEX(2, 20, 9),  // 湿度9
123
124   /*****************************************************************************
125    *                                  机械人通用配置                            ›
126    *****************************************************************************
127
128   /*****************************************************************************
129    *                                  MOTOR_DEFAULT
130    *****************************************************************************
131   kreg_robot_move     = REG_INDEX(10, 0, 0),  // 机器人X是否在移动
132   kreg_robot_pos      = REG_INDEX(10, 0, 1),  // 机器人x坐标
133   kreg_robot_velocity = REG_INDEX(10, 0, 2),  // 机器人x速度
134   kreg_robot_torque   = REG_INDEX(10, 0, 3),  // 机器人x电流
135
136   kreg_motor_shift                          = REG_INDEX(10, 50, 0),   // x偏移
137   kreg_motor_shaft                          = REG_INDEX(10, 50, 1),   // x轴是否反转
138   kreg_motor_one_circle_pulse               = REG_INDEX(10, 50, 2),   // x轴一圈脉冲
139   kreg_motor_one_circle_pulse_denominator = REG_INDEX(10, 50, 3),   // 设置一圈脉
140   kreg_motor_default_velocity               = REG_INDEX(10, 50, 4),   // 默认速度
141   kreg_motor_default_acc                    = REG_INDEX(10, 50, 5),   // 默认加速度
142   kreg_motor_default_dec                    = REG_INDEX(10, 50, 6),   // 默认减速度
143   kreg_motor_default_break_dec              = REG_INDEX(10, 50, 7),   // 默认减速度
144   kreg_stepmotor_ihold                      = REG_INDEX(10, 50, 8),   // 步进电机电
145   kreg_stepmotor_irun                       = REG_INDEX(10, 50, 9),   // 步进电机电
146   kreg_stepmotor_iholddelay                 = REG_INDEX(10, 50, 20),  // 步进电机电
147   kreg_motor_run_to_zero_max_d              = REG_INDEX(10, 50, 21),  // x轴回零最大
148   kreg_motor_look_zero_edge_max_d           = REG_INDEX(10, 50, 22),  // x轴找零边缘
149   kreg_motor_run_to_zero_speed              = REG_INDEX(10, 50, 23),  // 回零速度
150   kreg_motor_run_to_zero_dec                = REG_INDEX(10, 50, 24),  // 回零减速度
151   kreg_motor_look_zero_edge_speed           = REG_INDEX(10, 50, 25),  // 找零边缘速
152   kreg_motor_look_zero_edge_dec             = REG_INDEX(10, 50, 26),  // 找零边缘减
153   kreg_motor_default_torque                 = REG_INDEX(10, 50, 27),  // 默认扭矩
154
```

```
155  /*****************************************************************************
156   *                                  MOTOR_X
157   *****************************************************************************
158  kreg_robot_x_move     = REG_INDEX(11, 0, 0),  // 机器人X是否在移动
159  kreg_robot_x_pos      = REG_INDEX(11, 0, 1),  // 机器人x坐标
160  kreg_robot_x_velocity = REG_INDEX(11, 0, 2),  // 机器人x速度
161  kreg_robot_x_torque   = REG_INDEX(11, 0, 3),  // 机器人x电流
162
163  kreg_motor_x_shift                = REG_INDEX(11, 50, 0),   // x偏移
164  kreg_motor_x_shaft                = REG_INDEX(11, 50, 1),   // x轴是否反转
165  kreg_motor_x_one_circle_pulse     = REG_INDEX(11, 50, 2),   // x轴一圈脉冲数
166  kreg_motor_x_default_velocity     = REG_INDEX(11, 50, 3),   // 默认速度
167  kreg_motor_x_default_acc          = REG_INDEX(11, 50, 4),   // 默认加速度
168  kreg_motor_x_default_dec          = REG_INDEX(11, 50, 5),   // 默认减速度
169  kreg_motor_x_default_break_dec    = REG_INDEX(11, 50, 6),   // 默认减速度
170  kreg_stepmotor_x_ihold            = REG_INDEX(11, 50, 7),   // 步进电机电流配置
171  kreg_stepmotor_x_irun             = REG_INDEX(11, 50, 8),   // 步进电机电流配置
172  kreg_stepmotor_x_iholddelay       = REG_INDEX(11, 50, 9),   // 步进电机电流配置
173  kreg_motor_run_to_zero_max_x_d    = REG_INDEX(11, 50, 20),  // x轴回零最大距离
174  kreg_motor_look_zero_edge_max_x_d = REG_INDEX(11, 50, 21),  // x轴找零边缘最大距
175  kreg_motor_x_run_to_zero_speed    = REG_INDEX(11, 50, 22),  // 回零速度
176  kreg_motor_x_run_to_zero_dec      = REG_INDEX(11, 50, 23),  // 回零减速度
177  kreg_motor_x_look_zero_edge_speed = REG_INDEX(11, 50, 24),  // 找零边缘速度
178  kreg_motor_x_look_zero_edge_dec   = REG_INDEX(11, 50, 25),  // 找零边缘减速度
179  kreg_motor_x_default_torque       = REG_INDEX(11, 50, 26),  // 默认扭矩
180
181  /*****************************************************************************
182   *                                  MOTOR_Y
183   *****************************************************************************
184  kreg_robot_y_move     = REG_INDEX(12, 0, 0),  // 机器人X是否在移动
185  kreg_robot_y_pos      = REG_INDEX(12, 0, 1),  // 机器人x坐标
186  kreg_robot_y_velocity = REG_INDEX(12, 0, 2),  // 机器人x速度
187  kreg_robot_y_torque   = REG_INDEX(12, 0, 3),  // 机器人x电流
188
189  kreg_motor_y_shift                = REG_INDEX(12, 50, 0),   // x偏移
190  kreg_motor_y_shaft                = REG_INDEX(12, 50, 1),   // x轴是否反转
191  kreg_motor_y_one_circle_pulse     = REG_INDEX(12, 50, 2),   // x轴一圈脉冲数
192  kreg_motor_y_default_velocity     = REG_INDEX(12, 50, 3),   // 默认速度
193  kreg_motor_y_default_acc          = REG_INDEX(12, 50, 4),   // 默认加速度
194  kreg_motor_y_default_dec          = REG_INDEX(12, 50, 5),   // 默认减速度
195  kreg_motor_y_default_break_dec    = REG_INDEX(12, 50, 6),   // 默认减速度
196  kreg_stepmotor_y_ihold            = REG_INDEX(12, 50, 7),   // 步进电机电流配置
197  kreg_stepmotor_y_irun             = REG_INDEX(12, 50, 8),   // 步进电机电流配置
198  kreg_stepmotor_y_iholddelay       = REG_INDEX(12, 50, 9),   // 步进电机电流配置
199  kreg_motor_run_to_zero_max_y_d    = REG_INDEX(12, 50, 20),  // x轴回零最大距离
200  kreg_motor_look_zero_edge_max_y_d = REG_INDEX(12, 50, 21),  // x轴找零边缘最大距
201  kreg_motor_y_run_to_zero_speed    = REG_INDEX(12, 50, 22),  // 回零速度
```

```
202   kreg_motor_y_run_to_zero_dec      = REG_INDEX(12, 50, 23),  // 回零减速度
203   kreg_motor_y_look_zero_edge_speed = REG_INDEX(12, 50, 24),  // 找零边缘速度
204   kreg_motor_y_look_zero_edge_dec   = REG_INDEX(12, 50, 25),  // 找零边缘减速度
205   kreg_motor_y_default_torque       = REG_INDEX(12, 50, 26),  // 默认扭矩
206
207   /*****************************************************************************
208    *                                   MOTOR_Z
209    *****************************************************************************
210   kreg_robot_z_move     = REG_INDEX(13, 0, 0),  // 机器人X是否在移动
211   kreg_robot_z_pos      = REG_INDEX(13, 0, 1),  // 机器人x坐标
212   kreg_robot_z_velocity = REG_INDEX(13, 0, 2),  // 机器人x速度
213   kreg_robot_z_torque   = REG_INDEX(13, 0, 3),  // 机器人x电流
214
215   kreg_motor_z_shift                = REG_INDEX(13, 50, 0),   // x偏移
216   kreg_motor_z_shaft                = REG_INDEX(13, 50, 1),   // x轴是否反转
217   kreg_motor_z_one_circle_pulse     = REG_INDEX(13, 50, 2),   // x轴一圈脉冲数
218   kreg_motor_z_default_velocity     = REG_INDEX(13, 50, 3),   // 默认速度
219   kreg_motor_z_default_acc          = REG_INDEX(13, 50, 4),   // 默认加速度
220   kreg_motor_z_default_dec          = REG_INDEX(13, 50, 5),   // 默认减速度
221   kreg_motor_z_default_break_dec    = REG_INDEX(13, 50, 6),   // 默认减速度
222   kreg_stepmotor_z_ihold            = REG_INDEX(13, 50, 7),   // 步进电机电流配置
223   kreg_stepmotor_z_irun             = REG_INDEX(13, 50, 8),   // 步进电机电流配置
224   kreg_stepmotor_z_iholddelay       = REG_INDEX(13, 50, 9),   // 步进电机电流配置
225   kreg_motor_run_to_zero_max_z_d    = REG_INDEX(13, 50, 20),  // x轴回零最大距离
226   kreg_motor_look_zero_edge_max_z_d = REG_INDEX(13, 50, 21),  // x轴找零边缘最大距
227   kreg_motor_z_run_to_zero_speed    = REG_INDEX(13, 50, 22),  // 回零速度
228   kreg_motor_z_run_to_zero_dec      = REG_INDEX(13, 50, 23),  // 回零减速度
229   kreg_motor_z_look_zero_edge_speed = REG_INDEX(13, 50, 24),  // 找零边缘速度
230   kreg_motor_z_look_zero_edge_dec   = REG_INDEX(13, 50, 25),  // 找零边缘减速度
231   kreg_motor_z_default_torque       = REG_INDEX(13, 50, 26),  // 默认扭矩
232
233   kreg_xyrobot_robot_type = REG_INDEX(20, 50, 0),  // 机器人类型 0:hbot 1:corexy
234
235   /*****************************************************************************
236    *                                PID控制器(3000->4000)
237    *****************************************************************************
238   kreg_pid_target       = REG_INDEX(30, 0, 0),   // 目标数值
239   kreg_pid_nowoutput    = REG_INDEX(30, 0, 1),   // 当前输出
240   kreg_pid_feedbackval  = REG_INDEX(30, 0, 2),   // 当前输出
241   kreg_pid_kp           = REG_INDEX(30, 50, 0),  // kp
242   kreg_pid_ki           = REG_INDEX(30, 50, 1),  // ki
243   kreg_pid_kd           = REG_INDEX(30, 50, 2),  // kd
244   kreg_pid_max_output   = REG_INDEX(30, 50, 3),  // 最大输出
245   kreg_pid_min_output   = REG_INDEX(30, 50, 4),  // 最小输出
246   kreg_pid_max_integral = REG_INDEX(30, 50, 5),  // 最大积分
247   kreg_pid_min_integral = REG_INDEX(30, 50, 6),  // 最小积分
248   kreg_error_limit      = REG_INDEX(30, 50, 7),  // 误差限制
```

```
    kreg_compute_interval = REG_INDEX(30, 50, 8),  // 计算间隔

    /*****************************************************************************
     *                                  风扇控制
     *****************************************************************************/
    kreg_fan0_ctrl_speed_level = REG_INDEX(31, 0, 0),  // 风扇转速0,1,2,3
    kreg_fan1_ctrl_speed_level = REG_INDEX(31, 0, 1),  // 风扇转速0,1,2,3
    kreg_fan2_ctrl_speed_level = REG_INDEX(31, 0, 2),  // 风扇转速0,1,2,3
    kreg_fan3_ctrl_speed_level = REG_INDEX(31, 0, 3),  // 风扇转速0,1,2,3
    kreg_fan4_ctrl_speed_level = REG_INDEX(31, 0, 4),  // 风扇转速0,1,2,3

    kreg_fan0_speed_level = REG_INDEX(31, 10, 0),  // 风扇实时转速0,1,2,3
    kreg_fan1_speed_level = REG_INDEX(31, 10, 1),  // 风扇实时转速0,1,2,3
    kreg_fan2_speed_level = REG_INDEX(31, 10, 2),  // 风扇实时转速0,1,2,3
    kreg_fan3_speed_level = REG_INDEX(31, 10, 3),  // 风扇实时转速0,1,2,3
    kreg_fan4_speed_level = REG_INDEX(31, 10, 4),  // 风扇实时转速0,1,2,3

    kreg_pwm_pump0_ctrl_speed_level = REG_INDEX(32, 0, 0),  // PWM水泵0,1,2,3
    kreg_pwm_pump1_ctrl_speed_level = REG_INDEX(32, 0, 1),  // PWM水泵0,1,2,3
    kreg_pwm_pump2_ctrl_speed_level = REG_INDEX(32, 0, 2),  // PWM水泵0,1,2,3
    kreg_pwm_pump3_ctrl_speed_level = REG_INDEX(32, 0, 3),  // PWM水泵0,1,2,3
    kreg_pwm_pump4_ctrl_speed_level = REG_INDEX(32, 0, 4),  // PWM水泵0,1,2,3

    kreg_pwm_pump0_speed_level = REG_INDEX(32, 10, 0),  // PWM水泵0,1,2,3
    kreg_pwm_pump1_speed_level = REG_INDEX(32, 10, 1),  // PWM水泵0,1,2,3
    kreg_pwm_pump2_speed_level = REG_INDEX(32, 10, 2),  // PWM水泵0,1,2,3
    kreg_pwm_pump3_speed_level = REG_INDEX(32, 10, 3),  // PWM水泵0,1,2,3
    kreg_pwm_pump4_speed_level = REG_INDEX(32, 10, 4),  // PWM水泵0,1,2,3

    /*****************************************************************************
     *                                  移液枪状态
     *****************************************************************************/
    kreg_pipette_pos_ul           = REG_INDEX(40, 0, 0),   // 移液枪位置
    kreg_pipette_capactitance_val = REG_INDEX(40, 0, 1),   // 移液枪电容值
    kreg_pipette_tip_state        = REG_INDEX(40, 0, 2),   // 移动液枪tip状态
    kreg_pipette_limit_ul         = REG_INDEX(40, 50, 0),  // 移液枪ul限制

    /*****************************************************************************
     *                                  smartADC
     *****************************************************************************/
    kreg_self_reflecting_laser_sensor_transmitting_power  = REG_INDEX(41, 0, 0),
    kreg_self_reflecting_laser_sensor_receiving_tube_gain = REG_INDEX(41, 0, 1),
    kreg_self_reflecting_laser_sensor_sample_interval_ms  = REG_INDEX(41, 0, 2),
    kreg_self_reflecting_laser_sensor_num_samples         = REG_INDEX(41, 0, 3),

    /*****************************************************************************
     *                                  smartADC
```

```
296     ********************************************************************************
297   // scan action
298   kreg_boditech_optical_scan_type                    = REG_INDEX(42, 0, 0),  // 0
299   kreg_boditech_optical_scan_start_pos               = REG_INDEX(42, 0, 1),
300   kreg_boditech_optical_scan_direction               = REG_INDEX(42, 0, 2),
301   kreg_boditech_optical_scan_step_interval           = REG_INDEX(42, 0, 3),
302   kreg_boditech_optical_scan_pointnum                = REG_INDEX(42, 0, 4),
303   kreg_boditech_optical_channel_select_num           = REG_INDEX(42, 0, 5),
304   kreg_boditech_optical_laster_gain                  = REG_INDEX(42, 0, 6),
305   kreg_boditech_optical_scan_gain                    = REG_INDEX(42, 0, 7),
306   kreg_boditech_optical_trf_uvled_on_duration_us     = REG_INDEX(42, 0, 8),
307   kreg_boditech_optical_trf_uvled_off_duration_us    = REG_INDEX(42, 0, 9),
308   kreg_boditech_optical_trf_scan_delay_us            = REG_INDEX(42, 0, 10),
309   kreg_boditech_optical_trf_scan_duration_us         = REG_INDEX(42, 0, 11),
310   kreg_boditech_optical_scan_gain_adjust_suggestion = REG_INDEX(42, 0, 12),
311   kreg_boditech_optical_adc_result_overflow          = REG_INDEX(42, 0, 13),
312   kreg_boditech_optical_laster_intensity             = REG_INDEX(42, 0, 14),
313
314 } reg_index_t;
315
316 }  // namespace iflytop
317
```