1. STONE\_CNT ( SendPacketStoneCnt) // [고대(비젼) > 고대(전략)] 스톤 갯수
2. STONE\_INFO ( SendPacketStoneInfo) // [고대(비젼) > 고대(전략)]스톤의 위치
3. ROBOT\_INFO( SendPacketRobotInfo)// [고대(비젼) > DGIST] 투구 중 로봇 각도 위치
4. STONE\_INFO\_ACK( SendPacket~~StoneInfoAck~~)// [전략 > 비젼] 스톤 정보 응답
5. RELEASE( SendPacketRelease)// [전략 > 영남대] 릴리즈 속력, 각도, 컬방향, (위치, 목표 지점)
6. FLAG

SendPacketThrowFlag// [영남대 > 고대(비전)] 투구 플래그

SendPacketRioThrowFlag// [영남대 > DGIST] 투구 플래그

1. SPEED\_PROF( SendPacketSpeedProf)// [영남대 > DGIST] 투구 중 로봇 각도, 속도 프로파일
2. POS\_PROF( SendPacketPositionProf) // [영남대 > DGIST] 위치 프로파일
3. RESET( SendPacketRioreset)// [DGIST > 영남대] RIO Reset
4. MODE( SendPacket~~Runningmode~~)// [DGIST > 영남대] 주행모드
5. RUN( SendPacket~~predrive~~) // [DGIST > 영남대] 주행준비
6. START(SendPacket~~Drivestart~~)// [DGIST > 영남대] 주행시작
7. RESET\_KU( SendPacket~~KUReset~~) // [영남대 > DGIST] KU\_RESET
8. ROBOT\_MODE(SendPacketThrowmode) // [고대(전략) > 영남대] 투구로봇 ID
9. PRE\_MODE(SendPacketPremode) // [고대(전략) > 영남대] 연습모드
10. MY\_TURN(SendPacketMyturn) // [고대(전략) > 영남대] 로봇 투구 차례
11. CALL\_STONE\_INFO(SendPacketCallstoneinfo) // [전략 > 비젼] 스톤 정보 호출
12. CALL\_ROBOT\_INFO(SendPacketCallrobotinfo) // [영남대 > 비젼] 로봇 정보 호출
13. EMERGENCY(SendPacketEMERGENCY) // [고대(전략) > 영남대] 긴급 정지
14. RESTART( SendPacketRestart) // [고대(전략) > 영남대] 재시작
15. CALIB\_DATA( SendPacketCalibdata) // [고대(전략) > 영남대] 캘리브레이션
16. TARGET\_ANGLE(SendPacket~~Targetangle~~) // [영남대 > DIGST] 목표 각도 전송
17. CALIB\_ANGLE(SendPacket~~CalibAngle~~) // [영남대 > DIGST] 각도 보정 전송
18. HOG\_DIST(SendPacketHogdistance)// [영남대 > 전략] 투구되는 호그라인 거리 전송
19. RIO\_ENCODER(SendPacketRioencoder)// [DGIST > 영남대] 투구되는 호그라인 거리 전송

* SendPacketCRLF//복귀와 개행

1. INFO\_TIME(SendPacketInfoTime) // [전략 > 영상]info time
2. INFO\_RESULT(SendPacketInfoResult) // [영상 > 전략]info result

패킷 종류: 27개, send 함수 29개

PacketCv와 PacketCvtd의 차이: 공용체 내부의 자료형 및 크기

1. int RecvClassify
2. RecvPacketStoneCnt// [고대(비젼) > 고대(전략)] 스톤 갯수
3. int RecvPacketStoneInfo// [고대(비젼) > 고대(전략)] 스톤의 위치
4. RecvPacketRobotInfo// [고대(비젼) > DGIST] 투구 중 로봇 각도 위치
5. RecvPacketRelease // [고대(전략) > 영남대] 릴리즈 속력, 각도, 컬방향
6. bool isThrowFlag // [영남대 > 고대(비젼)] 지금 받은 packet이 투구 플래그인지
7. RecvPacketSpeedProf// [영남대 > DGIST] 투구 중 로봇 각도, 속도 프로파일
8. RecvPacketPositionProf// [DGIST > 영남대] 각속도 프로파일
9. RecvPacketRioReset// [고대(전략) > 영남대] RESET\_KU
10. RecvPacketThrowmode// [고대(전략) > 영남대] 투구로봇 ID 전송
11. RecvPacketPremode// [고대(전략) > 영남대] 연습모드, 경기모드 선택
12. RecvPacketMyturn// [고대(전략) > 영남대] 로봇 투구 차례인지 선택
13. bool RecvPacketCallstoneinfo// [고대(전략) > 고대(비젼)] 스톤 정보 요청
14. bool RecvPacketCallrobotinfo// [영남대 > 고대(비젼)] 로봇 정보 요청
15. bool RecvPacketEmergency// [고대(전략) > 영남대] 비상 정지
16. bool RecvPacketRestart// [고대(전략) > 영남대] 재시작
17. bool RecvPacketCalibdata// [고대(전략) > 영남대] Calibration data로 사용
18. RecvPacketHogdistance// [영남대 > 고대(전략)] 투구되는 호그라인 거리 전송
19. RecvPacketRioencoder// [DGIST > 영남대] 투구되는 호그라인 거리 전송
20. RecvPacketInfoTime// info time
21. RecvPacketInfoResult// info result
22. RecvPacketAcK// 패킷응답
23. SendPacketStoneCnt

SendPacketStoneInfo에 해당하는 stone의 개수를 전송

* SOCKET s
* cnt
* posArr
* isRedArr

<data>

헤더설정

STONE\_CNT 패킷형

cnt를 넣는다.

Send

SendPacketStoneInfo 함수 실행

* SOCKET s :s
* cnt :pos
* posArr:isRed

1. SendPacketStoneInfo

Stone의 위치

* SOCKET s
* pos
* isRed

<data>

헤더설정

STONE\_INFO 패킷형

pos.x의 주소를 4칸에 나누어 넣는다.

pos.y의 주소를 4칸에 나누어 넣는다.

isRed를 넣는다.

Send

1. SendPacketRobotInfo

* SOCKET s
* angle
* pos
* hogDist
* hogOffs

<data>

헤더설정

ROBOT\_INFO 패킷형

angle의 주소를 넣는다.

pos.x의 주소를 4칸에 나누어 넣는다.

pos.y의 주소를 4칸에 나누어 넣는다.

hogDist의 주소를 4칸에 나누어 넣는다.

hogOffs의 주소를 4칸에 나누어 넣는다.

데이터의 길이를 확인할 checkSum을 넣는다.

Send

1. SendPacketStoneInfoAck

* SOCKET s
* type

<data>

헤더설정

STONE\_INFO\_ACK 패킷형

type을 넣는다.

(0 : premode, 1: robot info, 2 : call stone info, 3: stone info, 4: release)

send

1. SendPacketRelease

* SOCKET s
* speed
* angle
* isRight
* pos
* tar\_pos

<data>

헤더설정

RELEASE 패킷형

speed의 주소를 4칸에 나누어 넣는다.

angle의 주소를 4칸에 나누어 넣는다.

isRight를 넣는다.

pos.x의 주소를 4칸에 나누어 넣는다.

pos.y의 주소를 4칸에 나누어 넣는다.

tar\_pos.x의 주소를 4칸에 나누어 넣는다.

tar\_pos.y의 주소를 4칸에 나누어 넣는다.

send

1. SendPacketThrowFlag

* SOCKET s

<data>

헤더설정

FLAG 패킷형

send

1. SendPacketRioThrowFlag

Rio에게 보내는 패킷

* SOCKET s

<data>

헤더설정

FLAG 패킷형

emp의 주소를 4칸에 나누어 넣는다.

emp의 주소를 4칸에 나누어 넣는다.

send

1. SendPacketSpeedProf

* SOCKET s
* angle
* speed

<data>

헤더설정

SPEED\_PROF 패킷형

angle의 주소를 4칸에 나누어 넣는다.

speed의 주소를 4칸에 나누어 넣는다.

send

1. SendPacketPositionProf

* SOCKET s
* posx
* posy

<data>

헤더설정

POS\_PROF 패킷형

posx의 주소를 4칸에 나누어 넣는다.

posy의 주소를 4칸에 나누어 넣는다.

send

1. SendPacketRioreset// [DGIST > 영남대] RIO Reset

* SOCKET s
* reset
* emp

<data>

헤더설정

RESET 패킷형

reset의 주소를 4칸에 나누어 넣는다.

emp의 주소를 4칸에 나누어 넣는다.

send

1. SendPacketRunningmode// [DGIST > 영남대] 주행모드

* SOCKET s
* mode
* emp

<data>

헤더설정

MODE 패킷형

mode의 주소를 4칸에 나누어 넣는다.

emp의 주소를 4칸에 나누어 넣는다.

send

1. SendPacketpredrive// [DGIST > 영남대] 주행준비

* SOCKET s
* Preparation
* emp

<data>

헤더설정

RUN 패킷형

Preparation의 주소를 4칸에 나누어 넣는다.

emp의 주소를 4칸에 나누어 넣는다.

send

1. SendPacketDrivestart// [DGIST > 영남대] 주행시작

* SOCKET s
* start
* emp

<data>

헤더설정

START 패킷형

start 주소를 4칸에 나누어 넣는다.

emp의 주소를 4칸에 나누어 넣는다.

send

1. SendPacketKUReset

* SOCKET s
* reset
* emp

<data>

헤더설정

RESET\_KU 패킷형

reset 주소를 4칸에 나누어 넣는다.

emp의 주소를 4칸에 나누어 넣는다.

send

1. SendPacketThrowmode

* SOCKET s
* number

<data>

헤더설정

ROBOT\_MODE 패킷형

number 주소를 4칸에 나누어 넣는다.

send

1. SendPacketPremode

* SOCKET s
* pre
* number

<data>

헤더설정

PRE\_MODE 패킷형

pre 주소를 4칸에 나누어 넣는다.

number 주소를 4칸에 나누어 넣는다.

send

1. SendPacketMyturn
2. int RecvClassify
3. RecvPacketStoneCnt// [고대(비젼) > 고대(전략)] 스톤 갯수
4. int RecvPacketStoneInfo// [고대(비젼) > 고대(전략)] 스톤의 위치
5. RecvPacketRobotInfo// [고대(비젼) > DGIST] 투구 중 로봇 각도 위치
6. RecvPacketRelease // [고대(전략) > 영남대] 릴리즈 속력, 각도, 컬방향
7. bool isThrowFlag // [영남대 > 고대(비젼)] 지금 받은 packet이 투구 플래그인지
8. RecvPacketSpeedProf// [영남대 > DGIST] 투구 중 로봇 각도, 속도 프로파일
9. RecvPacketPositionProf// [DGIST > 영남대] 각속도 프로파일
10. RecvPacketRioReset// [고대(전략) > 영남대] RESET\_KU
11. RecvPacketThrowmode// [고대(전략) > 영남대] 투구로봇 ID 전송
12. RecvPacketPremode// [고대(전략) > 영남대] 연습모드, 경기모드 선택
13. RecvPacketMyturn// [고대(전략) > 영남대] 로봇 투구 차례인지 선택
14. bool RecvPacketCallstoneinfo// [고대(전략) > 고대(비젼)] 스톤 정보 요청
15. bool RecvPacketCallrobotinfo// [영남대 > 고대(비젼)] 로봇 정보 요청
16. bool RecvPacketEmergency// [고대(전략) > 영남대] 비상 정지
17. bool RecvPacketRestart// [고대(전략) > 영남대] 재시작
18. bool RecvPacketCalibdata// [고대(전략) > 영남대] Calibration data로 사용
19. RecvPacketHogdistance// [영남대 > 고대(전략)] 투구되는 호그라인 거리 전송
20. RecvPacketRioencoder// [DGIST > 영남대] 투구되는 호그라인 거리 전송
21. RecvPacketInfoTime// info time
22. RecvPacketInfoResult// info result
23. RecvPacketAcK// 패킷응답
24. int RecvClassify

헤더검사 및 패킷을 포함하는지 검사 후 패킷형 반환

1. RecvPacketStoneCnt// [고대(비젼) > 고대(전략)] 스톤 개수

cnt변수에 스톤개수(msg[3]) 대입

1. int RecvPacketStoneInfo// [고대(비젼) > 고대(전략)] 스톤의 위치

PacketCvt

매개변수로 받은 msg배열 패킷형 뒤(msg[3])부터 PacketCvt형 공용체인 tmp의 ch배열에 index 0부터 복사

받은 배열(msg)과 checkSum이 일치하는지 확인(오류 검사)

1. RecvPacketRobotInfo// [고대(비젼) > DGIST] 투구 중 로봇 각도 위치

PacketCvt

1. RecvPacketRelease // [고대(전략) > 영남대] 릴리즈 속력, 각도, 컬방향

PacketCvt

1. bool isThrowFlag // [영남대 > 고대(비젼)] 지금 받은 packet이 투구 플래그인지

패킷형이 FLAG형이면 1 반환, 아니면 0 반환

1. RecvPacketSpeedProf// [영남대 > DGIST] 투구 중 로봇 각도, 속도 프로파일

PacketCvt

1. RecvPacketPositionProf// [DGIST > 영남대] 각속도 프로파일

RecvPacketAccelProf

PacketCvt

1. RecvPacketRioReset// [고대(전략) > 영남대] RESET\_KU

PacketCvt

1. RecvPacketThrowmode// [고대(전략) > 영남대] 투구로봇 ID 전송

PacketCvt

1. RecvPacketPremode// [고대(전략) > 영남대] 연습모드, 경기모드 선택

PacketCvt

1. RecvPacketMyturn// [고대(전략) > 영남대] 로봇 투구 차례인지 선택

turn이라는 변수가 bool변수이다. myturn이라면1, 아니라면 0반환하는듯

1. bool RecvPacketCallstoneinfo// [고대(전략) > 고대(비젼)] 스톤 정보 요청

CALL\_STONE\_INFO형 패킷 검사

1. bool RecvPacketCallrobotinfo// [영남대 > 고대(비젼)] 로봇 정보 요청

CALL\_ROBOT\_INFO형 패킷 검사

1. bool RecvPacketEmergency// [고대(전략) > 영남대] 비상 정지

EMERGENCY형 패킷 검사

1. bool RecvPacketRestart// [고대(전략) > 영남대] 재시작

RESTART형 패킷 검사

1. bool RecvPacketCalibdata// [고대(전략) > 영남대] Calibration data로 사용

CALIB\_DATA형 패킷 검사

패킷형이 일치하면 1반환, 아니면 0반환

1. RecvPacketHogdistance// [영남대 > 고대(전략)] 투구되는 호그라인 거리 전송

PacketCvt

1. RecvPacketRioencoder// [DGIST > 영남대] 투구되는 호그라인 거리 전송

거꾸로 들어간다. tmp.ch[7 - i] = msg[idx++];

PacketCvtd

1. RecvPacketInfoTime// info time

// info time

void SendPacketInfoTime(SOCKET s, char \* msg) {

char data[256];

int idx = 0;

// Header : 0xAA, 0x00 설정

data[idx++] = 0xAA;

data[idx++] = 0x00;

// Packet Type : INFO\_TIME

data[idx++] = INFO\_TIME;

// Set Data : data

for (int i = 0; data[idx++] = msg[i]; i++);

data[idx++] = '\0';

send(s, data, idx, 0);

}

// info time

char \*RecvPacketInfoTime(char\* msg) {

return &msg[3];

}

1. RecvPacketInfoResult// info result

PacketCvt

1. RecvPacketAcK// 패킷응답

STONE\_INFO\_ACK 패킷형이면 0:premode, 1: robot info, 2 : call stone info, 3: stone info, 4: release인지를 반환