□ 교통신호제어센터의 교차로 신호정보 제공 방안

1. 개요

1) 본 문서는 성남시 신호제어서버에서 ITS서버에 신호정보 제공을 위한 인터페이스를 정의한다.

2) 연계 구성

2. 통신 방식

1) 통신 방식 : UDP 소켓 통신

Server 측	Client 측	포트	규격
교통신호제어센터	외부시스템	7072 (TBD)	UDP

2) Byte Ordering : Big-Endian

- 최상위 바이트(MSB)를 먼저 보내고, 최하위 바이트(LSB)는 맨나중에 보냄

3) 데이터 형식 : Byte 형식

4) 문자 인코딩 : UTF-8

5) 포트번호 : 7072

3. 프레임 구조

1) 본 시스템의 프레임은 "HEADER + DATA"로 구성하며, "HEADER"의 구성과 "COMMAND" 목록은 다음과 같다.

■ HEADER 구성 (10 BYTES)

항목	설명	데이터크기	데이터 유형
STX1	통신프레임의 시작 부호 1 (0x7E)	1	BYTE
STX2	통신프레임의 시작 부호 2 (0x7E)	1	BYTE
SEQUENCE	순차번호 (0 ~ 255)	1	BYTE
TIME	현재시각 (32BIT)	4	BYTE
COMMAND	명령코드	1	BYTE
DATA LENGTH	데이터 프레임의 BYTE 길이	2	BYTE

■ COMMAND 목록

번호	COMMAND	설명	송수신 방향	비고
1	0xF0	교차로 신호운영현황 전송-1	신호서버 → 외부서버	
~~2~~	~~0xF1~~	~~교차로 신호운영현황 응답 (ACK)~~	~~신호서버 ← 외부서버~~	예비
3	0xF2	교차로 신호운영현황 전송-2	신호서버 → 외부서버	
4	~~0xF3~~	~~교차로 신호운영현황 응답 (ACK)~~	~~신호서버 ← 외부서버~~	예비
5	0xF4	교차로 주기정보 전송	신호서버 → 외부서버	
6	~~0xF5~~	~~교차로 주기정보 응답 (ACK)~~	~~신호서버 ← 외부서버~~	
7	0xF6	교차로 DB 전송	신호서버 → 외부서버	
8	~~0xF7~~	~~교차로 DB 응답 (ACK)~~	~~신호서버 ← 외부서버~~	
9	0x16	긴급차량서비스 시작	신호서버 → 외부서버	긴급차량
10	0x18	긴급차량서비스 제어	신호서버 → 외부서버	긴급차량
11	0x1A	긴급차량서비스 종료	신호서버 → 외부서버	긴급차량
12				

■ 전체 메시지 구성 (HEADER + DATA)

HEADER						DATA
STX1	STX2	SEQ.	TIME	COMMAND	LENGTH	DATA
1 byte	1 byte	1 byte	4 byte	1 byte	2 byte	[Length] Bytes

4. 데이터 상세 구조

1) 교차로 신호운영현황 정보 전송-1 **(COMMAND = 0xF0)**

- 1초 단위로 교차로 신호운영현황을 최소 정보만 요약하여 전송한다.
- 정보전송 **"교차로 시작번호"**부터 N개의 교차로를 연속적으로 전송한다

BYTE	항목	BIT	설명	
1	교차로 번호	16 BIT	정보전송 교차로 시작번호 (1 – 9999)	
2				
3	현시 코드	7 – 4	RING B의 PHASE (0 ~ 7)	
		3 – 0	RING A의 PHASE (0 ~ 7)	
4	제어기 운영상태	7	예비	0
		6	운용 중인 맵 번호	0 : 일반제, 1~5 : 시차제
		5		
		4		
		3	예비	0
		2	교통신호기 운영모드	0 : SCU 고정주기 모드 1 : 감응하지 않는 OFFLINE 제어모드 2 : 감응되는 OFFLINE 제어모드 4 : 감응되는 온라인 제어모드 5 : 감응하지 않는 온라인 제어모드 6 ~7 : 예비
		1		
		0		
5	제어기 상태	7	제어기 통신상태	1 : 통신 FAIL, 0 : 정상
		6	예비	0
		5	긴급제어 상태	1 : PHASE HOLD, 0 : 정상
		4	수동 상태	1 : 수동, 0 : 자동
		3	모순 상태	1 : 모순, 0 : 정상
		2	소등 상태	1 : 소등, 0 : 정상
		1	점멸 상태	1 : 점멸, 0 : 정상
		0	데이터베이스 상태	1 : 이상, 0 : 정상
6 - 8	2번 교차로 신호운영정보 (3 Bytes)			
9 - 11	2번 교차로 신호운영정보 (3 Bytes)			
-	...			
-	N번 교차로 신호운영정보 (3 Bytes)			

1	현시 코드	7 – 5	RING A의 PHASE (0 ~ 7)	
		4 – 0	~~RING A의 이동류번호 (1 ~ 21)~~ RING A의 STEP (0 ~ 31)	
2		7 – 5	RING B의 PHASE (0 ~ 7)	
		4 – 0	~~RING B의 이동류번호 (1 ~ 21)~~ RING B의 STEP (0 ~ 31)	
3	제어기 운영상태	7	제어기 통신상태	1 : 통신 FAIL, 0 : 정상
		6	모순 상태	1 : 모순, 0 : 정상
		5	소등 상태	1 : 소등, 0 : 정상
		4	점멸 상태	1 : 점멸, 0 : 정상
		3	수동 상태	1 : 수동, 0 : 정상
		2	교통신호기 운영모드	0 : SCU 고정주기 모드 1 : 감응하지 않는 OFFLINE 제어모드 2 : 감응되는 OFFLINE 제어모드 4 : 감응되는 온라인 제어모드 5 : 감응하지 않는 온라인 제어모드
		1		
		0		

2) 교차로 신호운영현황 정보 전송-2

- 1초 단위의 교차로 신호운영현황 정보
- 정보전송 **"교차로 시작번호"**부터 N개의 교차로를 연속적으로 전송한다

BYTE	항목	BIT	설명	
1 2	교차로 번호	16 BIT	정보전송 교차로 시작번호 (1 – 9999)	
3	현시 코드	7 – 5	RING A의 PHASE (0 ~ 7)	
		4 – 0	RING A의 STEP (0 ~ 31)	
4		7 – 5	RING B의 PHASE (0 ~ 7)	
		4 – 0	RING B의 STEP (0 ~ 31)	
5	제어기 운영상태	7	제어기 통신상태	1 : 통신 FAIL, 0 : 정상
		6	운용 맵번호	0 : 일반제, 1~5 : 시차제, 6 : 전용맵
		5		
		4		
		3	교차로 연등	0 : 일반교차로, 1 : 연등교차로
		2	교통신호기 운영모드	0 : SCU 고정주기 모드 1 : 감응하지 않는 OFFLINE 제어모드 2 : 감응되는 OFFLINE 제어모드 4 : 감응되는 온라인 제어모드 5 : 감응하지 않는 온라인 제어모드
		1		
		0		
6	제어기 상태	7	POLICE PANEL 수동진행 SW	1 : ON, 0 : OFF
		6	POLICE PANEL 수동 SW	1 : ON, 0 : OFF
		5	POLICE PANEL 점멸 SW	1 : ON, 0 : OFF
		4	POLICE PANEL 소등 SW	1 : ON, 0 : OFF
		3	모순 상태	1 : 모순, 0 : 정상
		2	소등 상태	1 : 소등, 0 : 정상
		1	점멸 상태	1 : 점멸, 0 : 정상
		0	데이터베이스 상태	1 : 이상, 0 : 정상
7		-	주기 COUNT	0 ~ 255 초
8		-	현 주기	0 ~ 255 초
9		-	연동	실제 계측 OFFSET
10	예비	-	~~A-ring 이동류번호~~	~~1 – 21~~
11	예비	-	~~B-ring 이동류번호~~	~~1 – 21~~
12 - 20	교차로 시작번호+1 신호운영정보 (9 Bytes)			
21 - 29	교차로 시작번호+2 신호운영정보 (9 Bytes)			
	...			
...	교차로 시작번호+N 신호운영정보 (9 Bytes)			

3) 교차로 주기정보 : 주기 종료 후 Split 정보 제공

- 주기 종료 후 Ring-A, Ring-B의 운영한 현시정보를 전송한다.
- 정보전송은 N개의 교차로 정보를 연속하여 전송이 가능하다

BYTE	항목	BIT	설명
1 2	교차로 번호	16 BIT	정보전송 교차로번호 (1 – 9999)
3 - 10	RING-A 운영시간	8 Bytes	PHASE 1 – 8
11 - 18	RING-B 운영시간	8 Bytes	PHASE 1 – 8
	2번째 교차로 번호/ Ring-A / RING-B 운영시간 (18 Bytes)		
	3번째 교차로 번호/ Ring-A / RING-B 운영시간 (18 Bytes)		
	4번째 교차로 번호/ Ring-A / RING-B 운영시간 (18 Bytes)		
	....		

4) 교차로 DB 정보 : 교차로 DB 변경시 제공 (JSON 표기)

① 교차로 WEEK PLAN

■ JSON 표기

WEEK PLAN JSON 표기	WEEK PLAN data 설명

{
"lcid" : number,
"type" : "weekplan",
"data" : number[],
}
	data : 7 plan_no
	plan_no = 1 ~ 10

■ week plan table 정의

일요일	월요일	화요일	수요일	목요일	금요일	토요일
0	0	0	0	0	0	0

② 교차로 DAY PLAN

■ JSON 표기

DAYPLAN JSON 표기	DAYPLAN data 설명

{
"Icid" : number,
"type" : "dayplan",
"plan" : {
"plan_no": number,
"time" **: number[],**
"data" : number[][],
}[ ]
}
	plan_no = 1 ~ 10
	time **: [A_flow_no, A_red_time, A_yellow_time, A_min_time,**
	B_flow_no, B_red_time, B_yellow_time, B_min_time]
	data : [hour, min, cycle, offset, split[16]]

■ FLOW/RED/YELLOW

현시	A-ring				B-ring			
현시 1	이동류번호	red_time	yellow_time	min_time	이동류번호	red_time	yellow_time	min_time
현시 2								
현시 3								
현시 4								
현시 5								
현시 6								
현시 7								
현시 8								

■ DAY PLAN TIME Table

	시	분	주기	연동	1A	1B	2A	2B	3A	3B	4A	4B	5A	5B	6A	6B	7A	7B	8A	8B
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

③ 교차로 HOLIDAY PLAN

■ JSON 표기

HOLIDAY PLAN JSON 표기	HOLIDAY PLAN data 설명

{
"lcid": number,
"type": "holidayplan",
"data": number[],
}
	[month, day, plan_no] * 30
	plan_no = 1 ~ 10

■ HOLIDAY Plan Table

	월	일	계획번호
1	0	0	0
2	0	0	0
3	0	0	0
4	0	0	0
5	0	0	0
6	0	0	0
7	0	0	0
8	0	0	0
9	0	0	0
10	0	0	0
11	0	0	0
12	0	0	0
13	0	0	0
14	0	0	0
15	0	0	0
16	0	0	0
17	0	0	0
18	0	0	0
19	0	0	0
20	0	0	0
21	0	0	0
22	0	0	0
23	0	0	0
24	0	0	0

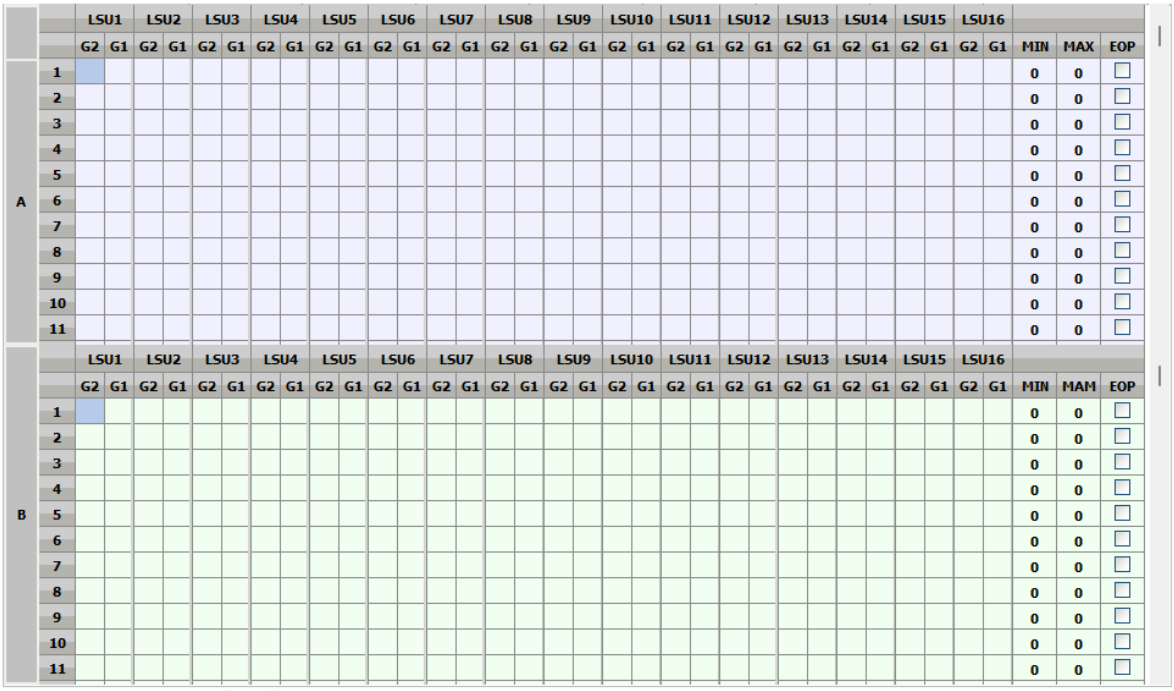
④ SIGNAL MAP (Ring-A , Ring-B)

■ JSON 표기

SIGNAL MAP JSON 표기	SIGNAL MAP data 설명

{
"lcid": number,
"type": "signal_map",
"data": {
"map_no" : number,
"a_ring" : number[],
"b_ring" : number[],
}[],
}
	map_no = 1 - 6
	[output[16], min, max, eop] * 32 step
	
	G1 : [d0 – d3] 하위 nibble
	G2 : [d4 – d7] 상위 nibble
	output code : R(0), G(1), Y(2), YF(3), RF(4), GF[5]
	eop : 1 = Phase 종료 표시

■ SIGNAL MAP Table



⑤ 교차로 기반정보 구성

■ JSON 표기

교차로 기반정보 구성 JSON 표기	교차로 기반정보 구성 data 설명

```
{
  "lcid": number,
  "type": "geo_map",
  "intName": string,          // 교차로명
  "intType": number,          // 교차로유형 (0: 주교차로, 1: 연등교차로)
  "lcType": number,           // 제어기 유형 (1: 2004년형, 2: 2010 년형, 3: 2023년형)
  "lampType": number,         // 등기 유형 (1: 4색등, 2: 3색등)
  "mainIntNo": number,        // 연등교차로인 경우 주교차로 번호
  "groupNo": number,          // 그룹번호
  "nodeNo": number,           // 교차로 노드 ID
  "intLat": number,           // 교차로 위도  (ex : 36.123500)
  "intLng": number,           // 교차로 경도  (ex : 127.123500)
  "ppcType": number,          // PPC 유형 (0 : 설치안됨, 1 : 현장방식, 2 : 중앙제어방식)
  "mainP": number,            // 주현시번호 (1 – 8)
  "aring_flow0": number[][], // 이동류#, 화살표 시작점 위도/경도, 중간점 위도/경도, 종료점 위도/경도,
  "bring_flow0": number[][], // 이동류#, 화살표 시작점 위도/경도, 중간점 위도/경도, 종료점 위도/경도,
  "aring_flow1": number[][], // 이동류#, 화살표 시작점 위도/경도, 중간점 위도/경도, 종료점 위도/경도,
  "bring_flow1": number[][], // 이동류#, 화살표 시작점 위도/경도, 중간점 위도/경도, 종료점 위도/경도,
  "aring_flow2": number[][], // 이동류#, 화살표 시작점 위도/경도, 중간점 위도/경도, 종료점 위도/경도,
  "bring_flow2": number[][], // 이동류#, 화살표 시작점 위도/경도, 중간점 위도/경도, 종료점 위도/경도,
}
```

■ 이동류 표시 예제

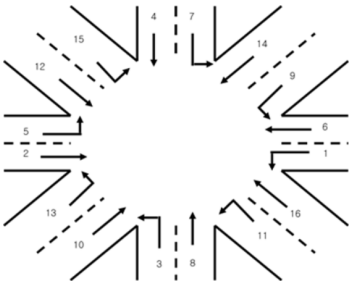
일반제 맵 : aring_flow0, bring_flow0
시차제 맵 : aring_flow1, bring_flow1, ... (시차제 있는 경우만 데이터 있음)

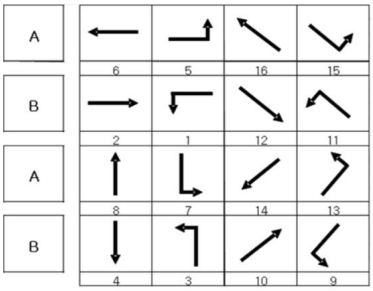
```
"aring_flow": [
        { [1,  x1, y1, x2, y2, x3, y3], [2, x1, y1, x2, y2, x3, y3], [3, x1, y1, x2, y2, x3, y3], }, // 1현시
        { [4,  x1, y1, x2, y2, x3, y3], [5, x1, y1, x2, y2, x3, y3], [6, x1, y1, x2, y2, x3, y3], }, // 2현시
        { [], [], [], },                                                                            // 3현시
        { [4,  x1, y1, x2, y2, x3, y3], [5, x1, y1, x2, y2, x3, y3], [6, x1, y1, x2, y2, x3, y3], }, // 4현시
        ],
"bring_flow": [
        { [1,  x1, y1, x2, y2, x3, y3], [2, x1, y1, x2, y2, x3, y3], [3, x1, y1, x2, y2, x3, y3], }, // 1현시
        { [4,  x1, y1, x2, y2, x3, y3], [5, x1, y1, x2, y2, x3, y3], [6, x1, y1, x2, y2, x3, y3], }, // 2현시
        ],
```

숫자 1~8 : 이동류 번호입니다.
  x1, y1 : 시작점 좌표
  x2, y2 : 중간점 좌표
  x3, y3 : 종료점 좌표

■ 이동류 번호

이동류 번호	이동류 방향	설명	이동류 번호	이동류 방향	설명	이동류 번호	이동류 방향	설명
1		좌회전 (동 → 남)	9		좌회전 (북동 → 남동)	17	·	보행신호
2		직진 (서 → 동)	10		직진 (남서 → 북동)	18	·	신호 없음
3		좌회전 (남 → 서)	11		좌회전 (남동 → 남서)	21	·	신호우회전
4		직진 (북 → 남)	12		직진 (북서 → 남동)			
5		좌회전 (서 → 북)	13		좌회전 (남서 → 북서)			
6		직진 (동 → 서)	14		직진 (북동 → 남서)			
7		좌회전 (북 → 동)	15		좌회전 (북서 → 북동)			
8		직진 (남 → 북)	16		직진 (남동 → 북서)			





5) 긴급차량 정보제공

① 긴급차량서비스 시작 요청 (Command =0x16)

- 긴급차량우선신호서버는 긴급차량서비스를 제공하는 교차로를 대상으로 남은거리/속도/유지현시번호를 신호제어서버에 제공하고, 신호제어서버는 해당 교차로제어를 시작한다.
- 통신 오류가 발생시 제어중인 교차로를 자동으로 종료 처리해야 한다.

항목	설명	데이터크기
서비스 ID	년+월+일+시+분+초+순번(0~99), CHAR	16
교차로 번호	교차로 번호 = (지자체 번호) * 10000 + (지자체 교차로 순번)	4
교차로남은거리	교차로 남은 거리에 따라 신호제어	4
차량속도	긴급차량의 이동속도	4
유지현시번호	현시번호(1-8)	1
긴급차량 위치정보	위도 (DD.DDDDDD x 1,000,000),	4
	경도 (DDD.DDDDDD x 1,000,000)	4
....		

② 긴급차량서비스 제어 요청 (Command = 0x18)

- 긴급차량우선신호서버는 긴급차량서비스를 제공하는 교차로를 대상으로 남은거리/속도/유지현시번호를 신호제어서버에 제공하고,
- 신호제어서버는 해당 긴급차량 서비스 교차로제어를 수행한다.

항목	설명	데이터크기
서비스 ID	년+월+일+시+분+초+순번(0~99), CHAR	16
교차로 번호	교차로 번호 = (지자체 번호) * 10000 + (지자체 교차로 순번)	4
교차로남은거리	교차로 남은 거리에 따라 신호제어	4
차량속도	긴급차량의 이동속도	4
유지현시번호	현시번호 (1-8)	1
....		

- 긴급차량의 위치정보, 차량ID, 차종(구급차, 소방차 ...등), 출발지, 목적지, 기타정보 등은 정보 추가를 협의 한다.

③ 긴급차량서비스 제어 요청 (Command = 0x1A)

- 통합단말기의 긴급차량서비스 자동/강제 종료요청 정보

항목	설명	데이터크기
서비스 ID	년+월+일+시+분+초+순번(0~99), CHAR	16