한이음

SW개발/HW제작 설계서

프로젝트 명 : Quantum-Safe 기술을 이용한 자율 주행 자동차와 차세대 교통 정보 체계(V2X)의 안전한 연동에 필요한 보안 모바일 망 서비스 구현

2021. 08. 29

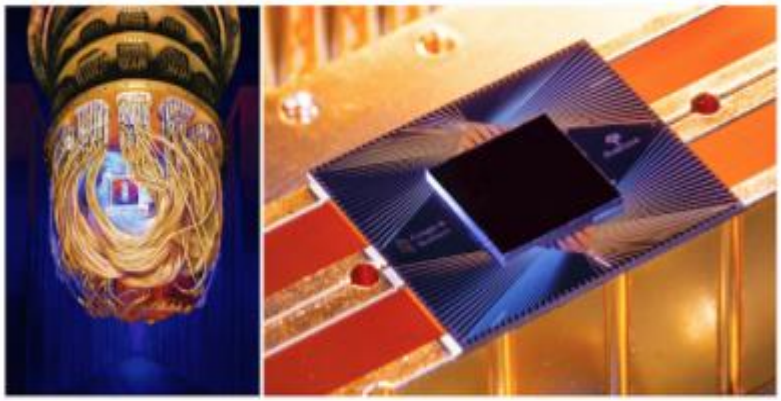


그림 1. 구글 AI Blog: Quantum Supremacy Using a Programmable Superconducting Processor

그림 2. 중국과학기술대의 광자기반 양자컴퓨터

최근 구글, 중국에서 기존 컴퓨터의 계산능력을 뛰어넘는 양자 우위 또는 이점(Quantum supremacy 또는 Quantum advantage)을 달성했다는 기사가 나오는 등 양자컴퓨터 기술이 급속하게 발전하고 있다. 양자컴퓨터의 발전이 긍정적인 부분이 많지만, 현대 암호체계에는 큰 위협이 되고 있다.

현재 사용하고 있는 공개키 암호를 대체하기 위하여, 미국 NIST에서는 양자 내성 암호(Post quantum Cryptography) 연구는 공개키 암호가 처음 개발된 70년대 후반부터 지속해서 연구가 진행되어왔고, 양자컴퓨터 이슈가 되었던 2000년대 초반에 연구가 확대되었고, 일부는 실용화도 되었다. 미국 NIST는 기존 개발결과와 새로운 우수한 공개키 암호를 확보하기 위한 공모사업을 2016년부터 진행하고 있으며, 2024년에 표준화 완료를 목표로 하고 있다.

Table 1. NIST PQC round 3 finalists

	Signature	PKE/KEM
Lattice	CRYSTALS -DILIGHIUM, FALCON	CRYSTALS -KYBER, NTRU, SABER
Code	-	Classic McEliece
Multivariate	Rainbow	-

Table 2. The size of the key and ciphertext for NIST PQC round 3 finalists (unit: byte)

	NTRU hps4096821	KYBER 1024	Fire SABER
sk	1,590	3,168	3,040
pk	1,230	1,568	1,312
ct	1,230	1,568	1,472

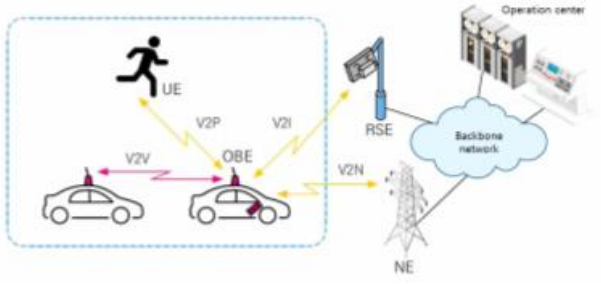


그림 1. V2X 통신 개념도

2016년 공모사업 발표 이후 다양한 문제를 기반으로 하는 64개의 알고리즘이 1라운드 후보 알고리즘으로 공개되었으며 안전성, 성능에 대한 평가를 거쳐 2020년 7개 알고리즘이 3라운드 최종 후보(finalist)로 선정되었다. Table 1.과 같이 PKE/KEM에 속하는 4개의 최종 후보 알고리즘 중 NTRU,CRYSTALS-KYBER, SABER 3개의 알고리즘이 격자기반 암호로 격자기반 기술이 상당한 비율을 차지한다.

NTRU는 동일한 안전성을 제공하는 다른 최종 후보들과 비교하여 KEM 과정에서 전달해야 하는 키와 암호문의 크기가 작다는 점에서 효율성을 가진다. 또한, NTRU는 다른 최종후보와 비교하여 최고 수준의 성능을 내지는 못하지만 가장 긴 역사를 가지고 오랫동안 분석을 통해 안전성이 검증되어 온 것이 장점이다.

자율주행차에 관한 관심이 높아지면서, 신뢰성이 높고 지연시간이 낮은 무선접속 기술개발이 매우 중요해지고 있다. 자율주행의 필수 기술인 V2X는 미국의 DSRC와 유럽의 C-ITS가 주도해 왔다.

최근 이동통신사업자들은 고신뢰성, 저지연성, 높은 스루풋 성능을 갖는 C-V2X(cellular-V2X) 기술을 도입하면서 국내외적으로 향후 V2X 방식 결정이 큰 이슈가 되고 있다.

1. 교통정보체계

구분	기능	세부 기능	설명
S/W	인프라 데이터 처리	인프라 데이터 수집	각 인프라로부터 POST 요청으로 받은 데이터를 수집한다.
		인프라 데이터 가공	수집한 데이터를 바탕으로 자동차의 움직임을 제어하기 위한 메시지를 생성한다.
	키 생성 및 교환	NTRU Key 생성	양자내성암호에 사용될 공개키와 개인키를 생성한다.
		공개키 교환	교통정보체계와 자율주행자동차간 양자내성암호통신을 위한 공개키를 교환한다.
	메시지 암호화 및 전송	대칭키 생성	메시지를 암호화할 대칭키를 생성한다.
		메시지 암호화	대칭키로 메시지를 암호화한다.
		대칭키 암호화	상대방의 공개키로 대칭키를 암호화한다.
		암호문 병합 및 전송	암호화된 메시지와 대칭키를 병합하여 암호문을 만든다.
	유저 인터페이스	웹 소켓 통신 활성화	start 버튼을 눌러 웹 소켓 서버를 활성화 시킨다.
		공개키 교환 대기	자동차 웹 소켓 클라이언트와 연결되면 공개키가 교환되길 대기한다.
		암호화 과정 출력	대칭키, 공개키를 통해 암호화되는 과정을 보여준다.

2. 자동차(통신모듈)

구분	기능	세부 기능	설명
H/W	시리얼통신	아두이노와 통신	시리얼 통신을 통해 자동차 아두이노에게 데이터를 전달한다.
S/W	키 생성 및 교환	NTRU key 생성	양자내성암호화에 사용될 공개키와 대칭키를 생성한다.
		공개키 교환	교통 정보체계와 자율주행 자동차 간의 양자 내성 암호 통신을 위한 공개키를 교환한다.
	암호문 복호화	암호문 분해	암호화된 대칭키와 암호화된 메시지로 암호문을 분해한다.
		대칭키 복호화	개인키를 통해 대칭키를 복호화한다.
		메시지 복호화	대칭키를 통해 메시지를 복호화한다.
	유저 인터페이스	웹 소켓 통신 활성화	start버튼을 눌러 웹 소켓을 활성화한다.
		공개키 교환 대기	교통정보관리체계와 연결되면 공개키 교환을 대기한다.
		복호화 과정 출력	대칭키, 개인키를 통해 복호화 되는 과정을 보여준다.
	시리얼통신	시리얼통신 활성화	라인트레이서 아두이노와의 시리얼 통신을 활성화 한다.
		메시지 전송	복호화된 메시지를 라인트레이서 자동차에게 전송한다.

3. 자동차(자율주행)

구분	기능	세부 기능	설명
H/W	라인트레이서 모듈	트랙 위치 파악	적외선 송수신을 통해 흰색/검은색의 선을 구분하여 라인의 위치를 인지한다.
	모터쉴드	DC 모터 제어	DC모터에 걸리는 전압을 조절하여 자동차의 속도를 제어한다.
	교통정보체계와의 연동	자동차 라즈베리파이와 아두이노간의 시리얼 통신	교통정보체계 라즈베리파이와 자동차 라즈베리파이와의 통신에서 받은 입력을 시리얼 통신으로 직접 아두이노와 연결하여 원하는 동작을 할 수 있도록 한다.
S/W	자동차 동작 기능	입력된 시리얼 값에 따른 직진	자동차 라즈베리파이와의 시리얼 통신으로 입력된 문자가 'T'이면 라인을 따라 직진한다.
		입력된 시리얼 값에 따른 정지	자동차 라즈베리파이와의 시리얼 통신으로 입력된 문자가 'F'이면 정지한다.

4. 인프라(정류장)

구분	기능	세부 기능	설명
H/W	정류장의 사람 유무 측정	사람 인식	적외선 근접센서를 통해 정류장에 사람이 있는지 측정한다.
S/W	위치 정보 데이터 전송	웹 서버연결	교통정보체계와 HTTP 통신을 위해 연결 관계를 수립한다.
		사람 인식 데이터 전송	측정한 정보를 교통정보체계에 HTTP 방식으로 전송한다.

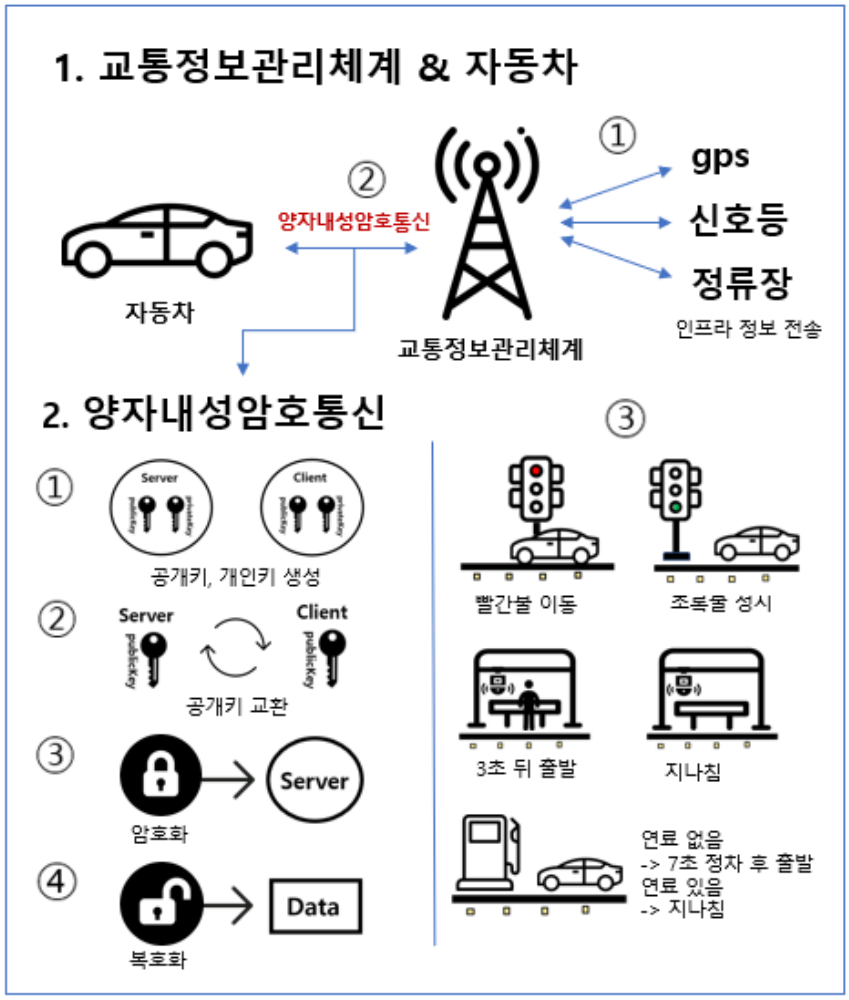
5. 인프라(신호등)

구분	기능	세부 기능	설명
H/W	신호등 LED 점등	LED 시간 차 점등 조절	신호등 모듈의 LED 점등으로 신호 표시를 알려준다.
S/W	인프라 데이터 처리	인프라 데이터 전송	신호등 모듈 인프라로부터 HTTP 요청을 통해 교통정보체계로 데이터를 넘겨준다.
		ESP8266 웹 서버 연결	ESP8266를 웹 서버와 연동하여 통신이 가능하고, 신호등 모듈로부터 측정값을 받아온다.

6. 인프라(위치정보)

구분	기능	세부 기능	설명
H/W	조도센서 값 수집	14개의 조도센서 각 아날로그 핀에서 값 수집	조도센서로 자동차의 현재위치를 파악한다. 지나갈 위치의 조도센서 값을 읽는다.
S/W	위치 정보 데이터 전송	ESP-01 웹서버 연결	ESP-01모듈을 사용하여 Wifi를 통해 웹서버에 접속한다.
		위치정보 데이터 전송	기준 미만의 조도센서 값을 파악하여 위치 정보 데이터를 교통정보체계로 넘겨준다.

| 서비스 구성도



1. 교통정보관리체계 & 자동차

① 교통정보관리 체계는 인프라 데이터를 JSON 형식으로 병합한다.

② 병합한 데이터를 하이브리드 암호화 체계를 통해 양자내성암호화하여 암호문을 웹 소켓을 통해 자동차로 전송한다.

③ 자동차는 전달받은 데이터를 토대로 연산하여 동작을 요구하는 신호에 따라 가거나 멈춘다.

2. 양자내성암호통신

① 클라이언트와 서버는 NTRU 알고리즘을 통해 공개키와 개인키를 생성한다.

② 클라이언트가 서버에 연결되면 공개키를 교환한다.

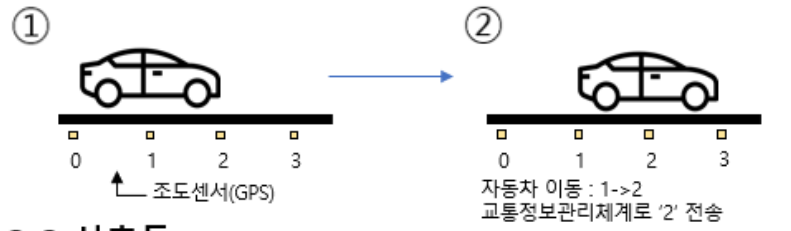
③ 일회용 대칭키를 생성하고 대칭키로 암호화한 데이터와 공개키로 암호화한 대칭키를 합친 암호문을 서버로 전달한다.

④ 암호문을 분리하여 개인키로 대칭키를 복호화하고 대칭키로 데이터를 복호화하여 데이터를 얻는다.

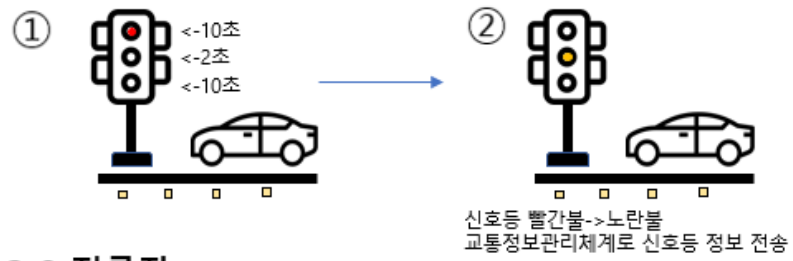
| 서비스 구성도

2. 인프라

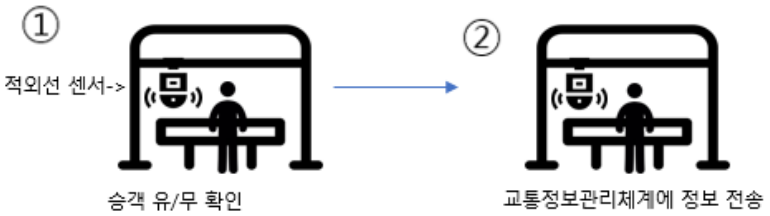
2-1 gps



2-2 신호등



2-3 정류장



2-1 gps

① 도로 위에 14개의 조도센서를 일정 간격으로 두어 0~13의 위치로 지정한다.

② 자동차가 지나가는 위치의 정보를 읽고 다음 위치로 이동할 때마다 교통정보관리체계에 위치정보를 전송한다.

2-2 신호등

① 빨간불 다음 노란불 다음 초록불을 각 10초, 2초, 10초 동안 켜지도록 설정한다.

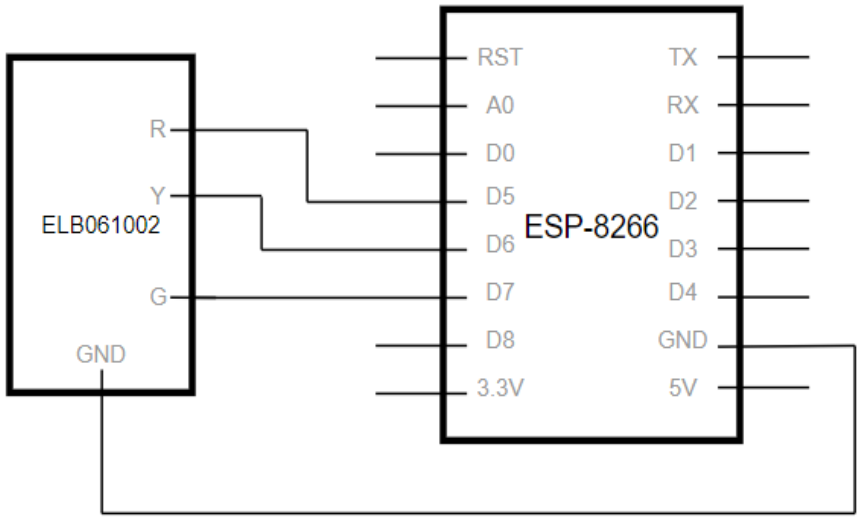
② 신호등의 색이 변화할 때마다 신호등 색의 정보를 교통정보관리체계에 전송한다.

2-3 정류장

① 적외선 센서로 정류장에 승객 유/무를 확인한다.

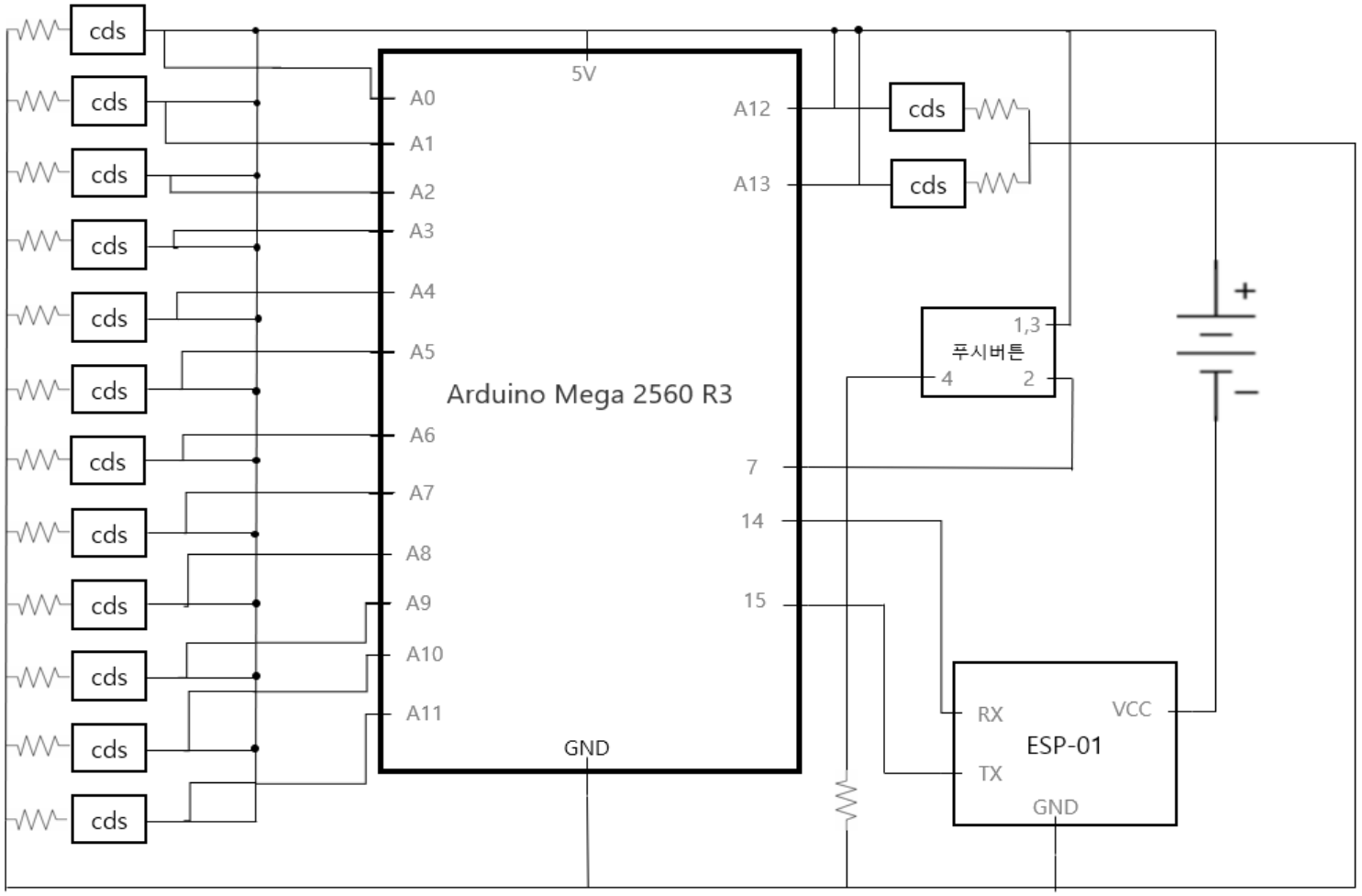
② 교통정보관리체계에 승객 정보를 전송한다.

1. 인프라(신호등)



센서 종류	연결 핀	설명
ESP-8266	GND	신호등 모듈의 GND에 연결
	D5	신호등 모듈의 R 핀에 연결
	D6	신호등 모듈의 Y 핀에 연결
	D7	신호등 모듈의 G 핀에 연결
ELB061002	GND	ESP-8266의 GND에 연결
	R	ESP-8266의 D5 핀에 연결
	Y	ESP-8266의 D6 핀에 연결
	G	ESP-8266의 D7 핀에 연결

2. 인프라(교통정보)

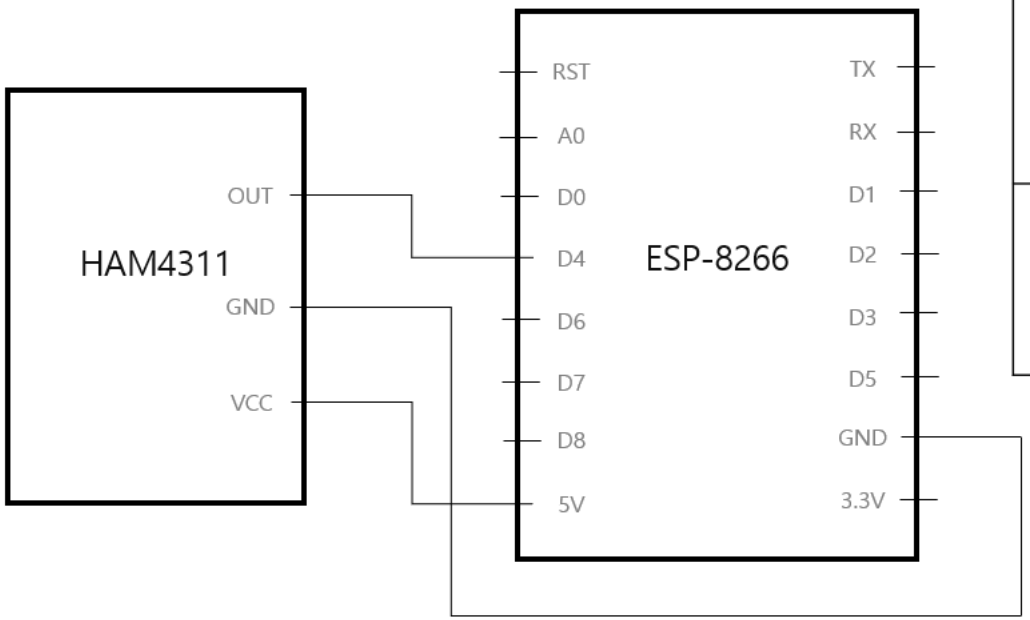

cds
cds
cds
cds
cds
cds
cds
cds
cds
cds
cds
cds
5V
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
GND
Arduino Mega 2560 R3
A12
A13
7
14
15
cds
cds
1,3
푸시버튼
4
2
RX
VCC
ESP-01
TX
GND

2. 인프라(교통정보)

센서	연결 핀	설명
조도 센서	VCC	아두이노의 5V, A0에 연결
	GND	아두이노의 GND에 연결
	VCC	아두이노의 5V, A1에 연결
	GND	아두이노의 GND에 연결
	VCC	아두이노의 5V, A2에 연결
	GND	아두이노의 GND에 연결
	VCC	아두이노의 5V, A3에 연결
	GND	아두이노의 GND에 연결
	VCC	아두이노의 5V, A4에 연결
	GND	아두이노의 GND에 연결
	VCC	아두이노의 5V, A5에 연결
	GND	아두이노의 GND에 연결
	VCC	아두이노의 5V, A6에 연결
	GND	아두이노의 GND에 연결
	VCC	아두이노의 5V, A7에 연결
	GND	아두이노의 GND에 연결
	VCC	아두이노의 5V, A8에 연결
	GND	아두이노의 GND에 연결
	VCC	아두이노의 5V, A9에 연결
	GND	아두이노의 GND에 연결
	VCC	아두이노의 5V, A10에 연결
	GND	아두이노의 GND에 연결
	VCC	아두이노의 5V, A11에 연결
	GND	아두이노의 GND에 연결
	VCC	아두이노의 5V, A12에 연결
	GND	아두이노의 GND에 연결
	VCC	아두이노의 5V, A13에 연결
	GND	아두이노의 GND에 연결

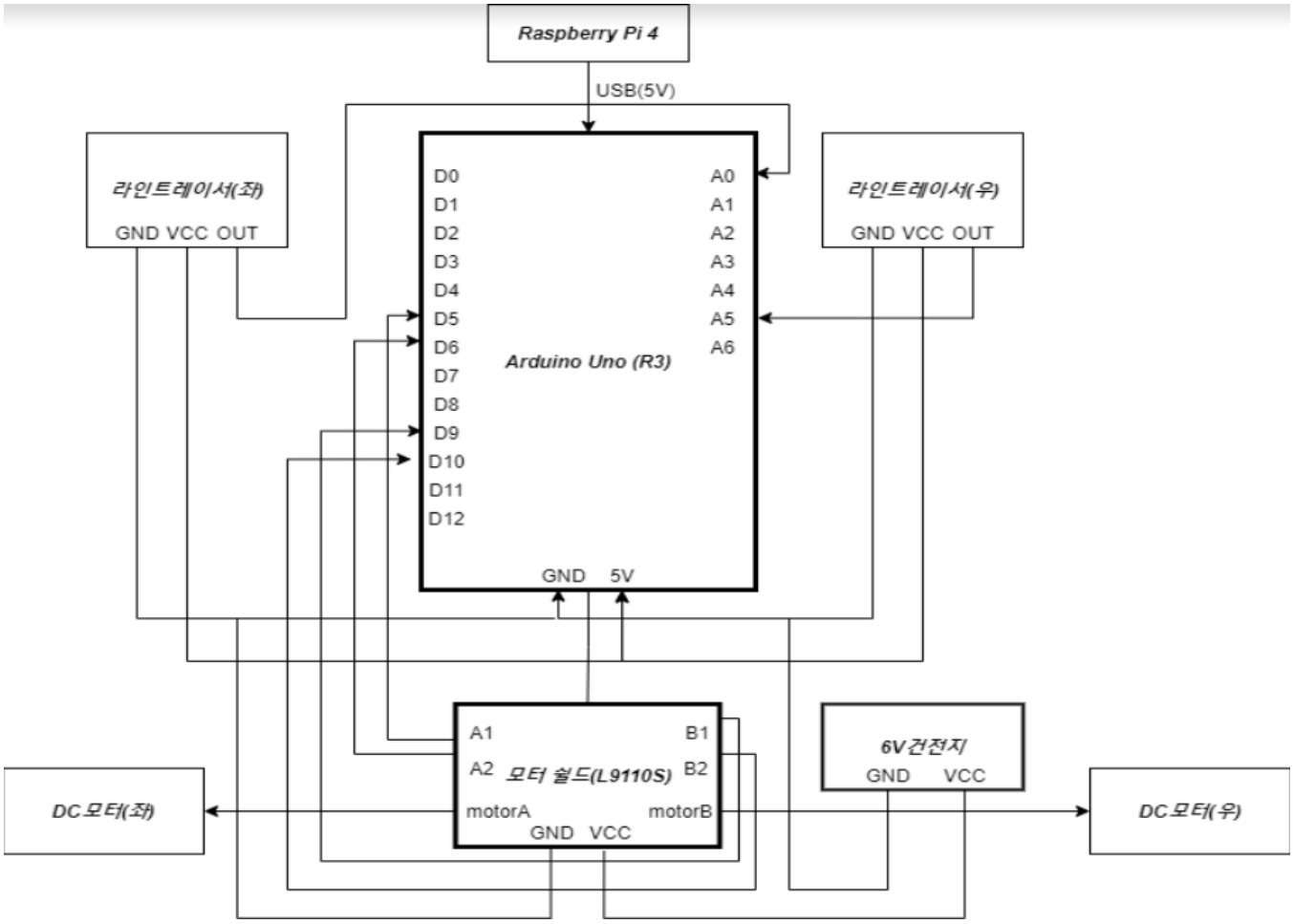
센서	연결 핀	설명
ESP-01 어댑터	VCC	아두이노의 5V에 연결
	GND	아두이노의 GND에 연결
	TX	아두이노 15번 핀에 연결
	RX	아두이노 14번 핀에 연결
ESP-01	VCC	ESP-01 어댑터에 연결
	GPIO0	
	GPIO2	
	Reset	
	Enable	
	GND	
	RX	
	TX	
푸시버튼	1	아두이노의 5V에 연결
	2	아두이노 7번 핀에 연결
	3	아두이노의 5V에 연결
	4	아두이노의 GND에 연결

3. 정류장



센서	연결 핀	설명
HAM4311	VCC	ESP-8266의 5V에 연결
	GND	ESP-8266의 GND에 연결
	OUT	ESP-8266의 D5 핀에 연결
ESP-8266	GND	IR적외선센서모듈의 GND에 연결
	D5	IR적외선센서모듈의 OUT에 연결
	5V	IR적외선센서모듈의 VCC에 연결

4. 자동차


Raspberry Pi 4
USB(5V)
라인트레이서(좌)
GND VCC OUT
라인트레이서(우)
GND VCC OUT
D0
D1
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
A0
A1
A2
A3
A4
A5
A6
Arduino Uno (R3)
GND 5V
A1
A2
B1
B2
모터 쉴드(L9110S)
motorA
motorB
GND VCC
6V건전지
GND VCC
DC모터(좌)
DC모터(우)

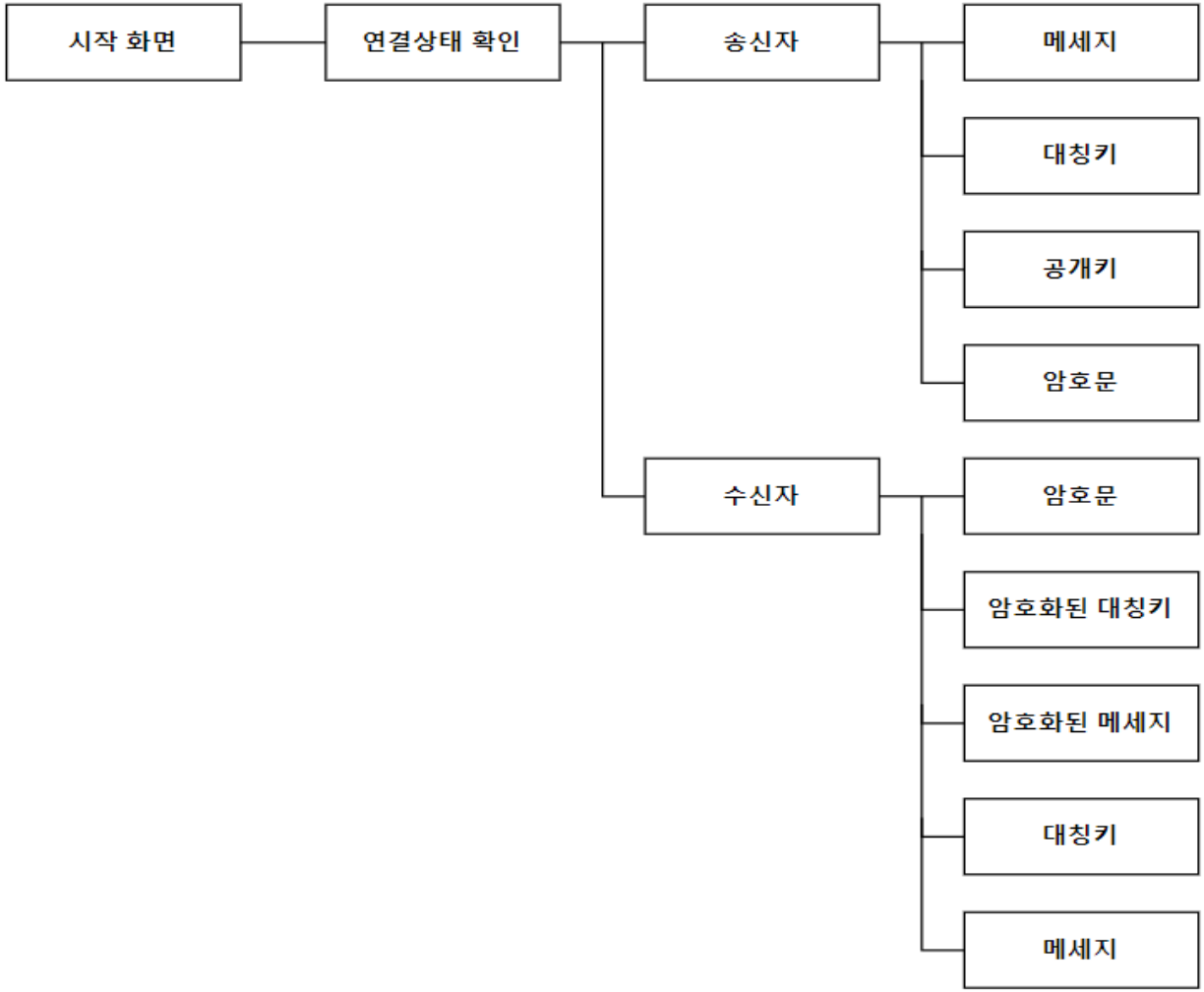
4. 자동차

센서	연결 핀	설명
Arduino Uno R3	D5	모터쉴드의 A1에 연결
	D6	모터쉴드의 A2에 연결
	D9	모터쉴드의 B1에 연결
	D10	모터쉴드의 B2에 연결
	A0	라인트레이서 모듈(좌)의 OUT에 연결
	A5	라인트레이서 모듈(우)의 OUT에 연결
	GND	라인트레이서 모듈, 모터쉴드, 6V건전지의 GND에 연결
	VCC	라인트레이서 모듈의 VCC에 연결
라인트레이서 모듈(좌)	GND	아두이노의 GND에 연결
	VCC	아두이노의 5v에 연결
	OUT	아두이노의 A0에 연결
라인트레이서 모듈(우)	GND	아두이노의 GND에 연결
	VCC	아두이노의 5v에 연결
	OUT	아두이노의 A5에 연결

센서	연결 핀	설명
모터쉴드 L9110S	motorA	DC모터(좌)에 연결
	motorB	DC모터(우)에 연결
	A1	아두이노의 D5에 연결
	A2	아두이노의 D6에 연결
	B1	아두이노의 D9에 연결
	B2	아두이노의 D10에 연결
	GND	아두이노의 GND에 연결
	VCC	6v 건전지의 VCC에 연결
DC모터(좌)	VCC GND	모터쉴드의 motorA에 연결
DC모터(우)	VCC GND	모터쉴드의 motorB에 연결
6V 건전지	GND	아두이노의 GND에 연결
	VCC	모터쉴드의 VCC에 연결

| 메뉴 구성도

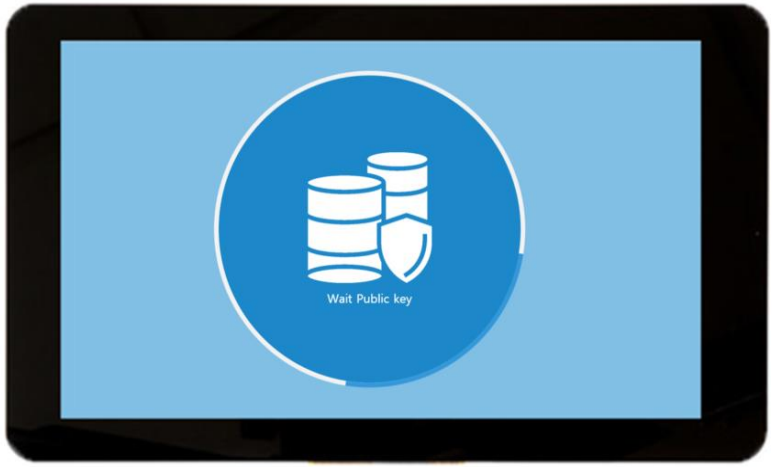
○ 메뉴 구성도


시작 화면
연결상태 확인
송신자
메세지
대칭키
공개키
암호문
수신자
암호문
암호화된 대칭키
암호화된 메세지
대칭키
메세지

1. 시작화면

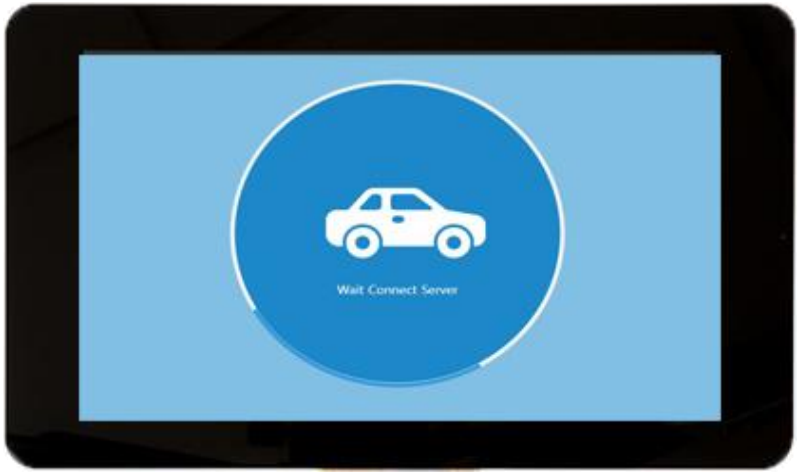
기능 번호	START_01
기능명	시작
기능설명	첫 화면, Start 버튼을 누르면 TCP 통신이 활성화된다.
처리내용	■ TCP 통신이 활성화되어 있지 않은 경우 Start 버튼을 누르면 TCP 통신이 활성화되며 다음 화면으로 이동한다. ■ TCP 통신이 활성화되어 있는 경우 해당 화면을 거치지 않고, 다음 기능 번호(EXCHANGE_01)로 넘어간다.
요구사항 명	유저 인터페이스(웹 소켓 통신 활성화)

2. 공개키 교환(교통정보관리체계)



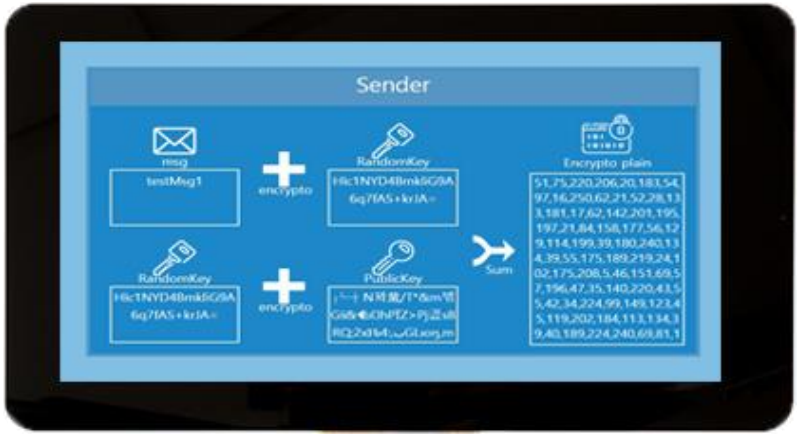
기능 번호	EXCHANGE_01
기능명	공개키 교환
기능설명	양자내성암호 통신에 사용될 공개키를 교환한다.
처리내용	■ 자동차의 공개키를 갖고 있지 않은 경우 공개키가 교환되길 대기한다. ■ 자동차의 공개키를 갖고 있는 경우 해당 화면을 거치지 않고, 다음 기능 번호(ENCRYPTO_01)로 넘어간다.
요구사항 명	유저 인터페이스(공개키 교환 대기), 키 생성 및 교환

3. 공개키 교환(자동차)



기능 번호	EXCHANGE_02
기능명	공개키 교환
기능설명	양자내성암호 통신에 사용될 공개키를 교환한다.
처리내용	■ 교통정보관리체계의 공개키를 갖고 있지 않은 경우 공개키가 교환되길 대기한다. ■ 교통정보관리체계의 공개키를 갖고 있는 경우 해당 화면을 거치지 않고, 다음 기능 번호(DECRYPTO_01)로 넘어간다.
요구사항 명	유저 인터페이스(웹 소켓 통신 활성화), 키 생성 및 교환

4. 암호화 통신(Sender)

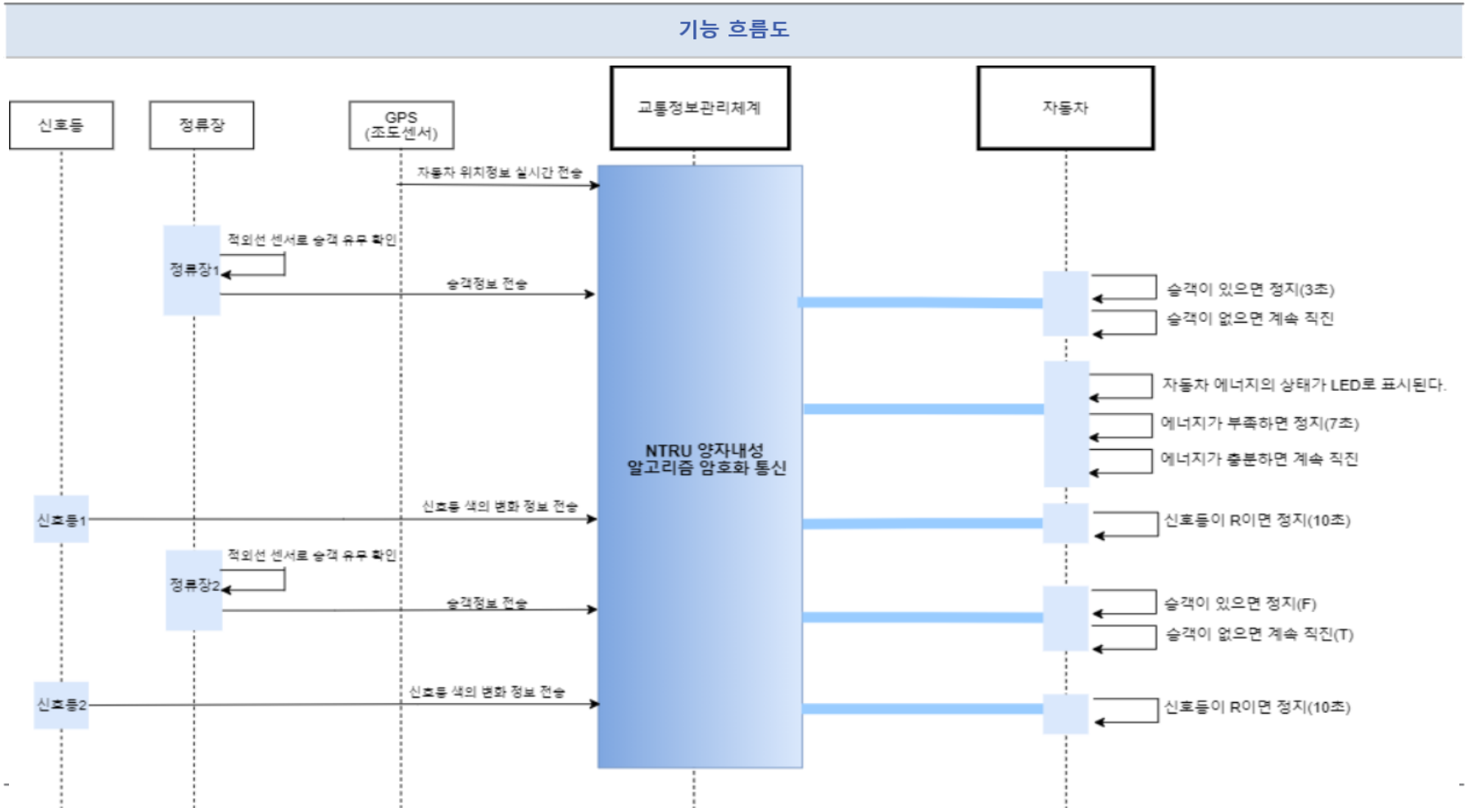


기능 번호	ENCRYPTO_01
기능명	암호화 전송
기능설명	양자내성암호화를 통해 메시지를 암호화하는 과정을 보여준다.
처리내용	■ 메시지를 전송하지 않으면 대기한다. ■ 메시지를 전송하면 메시지가 대칭키로, 대칭키가 공개키로 암호화되고 둘을 병합하여 암호문이 만들어지는 과정을 보여준다.
요구사항 명	유저 인터페이스(암호화 과정 출력), 인프라 데이터 처리, 메시지 암호화 및 전송

5. 암호화 통신(Receiver)

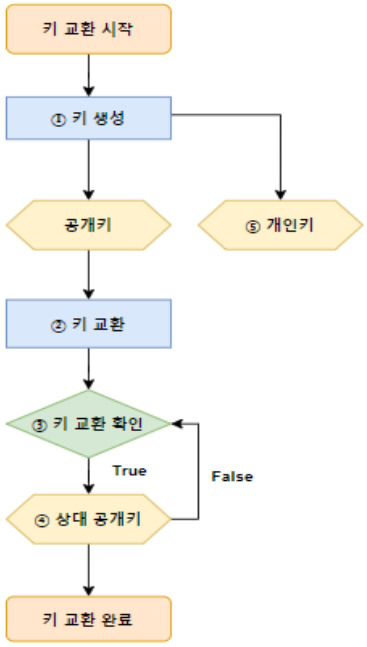
기능 번호	DECRYPTO_01
기능명	암호화 수신
기능설명	양자내성암호 알고리즘을 통해 암호문을 복화하는 과정을 보여준다.
처리내용	■ 메시지가 수신되지 않으면 대기한다. ■ 메시지를 수신되면 암호문을 분해하고 개인키와 복호화된 랜덤키로 메시지가 복호화 되는 과정을 보여준다.
요구사항 명	유저 인터페이스(복호화 과정 출력), 암호문 복호화, 시리얼통신(메시지전송)

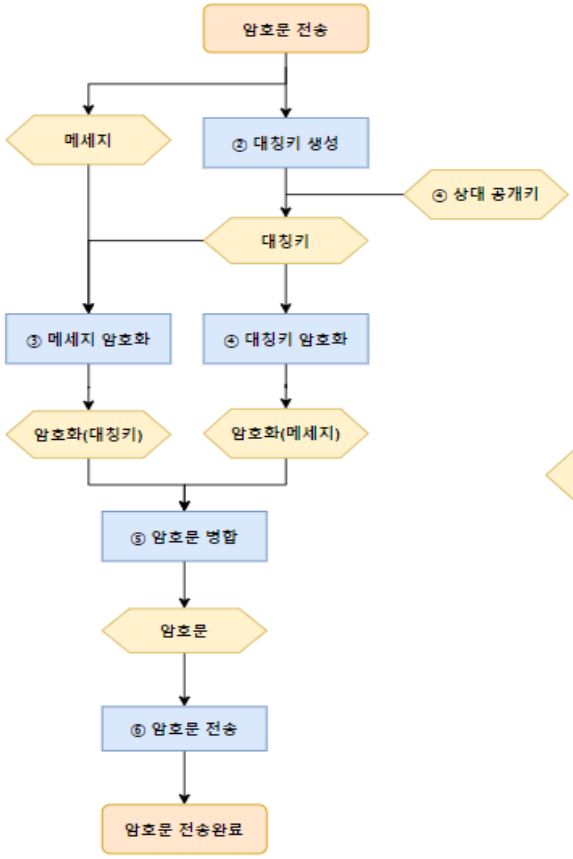
| 기능 흐름도

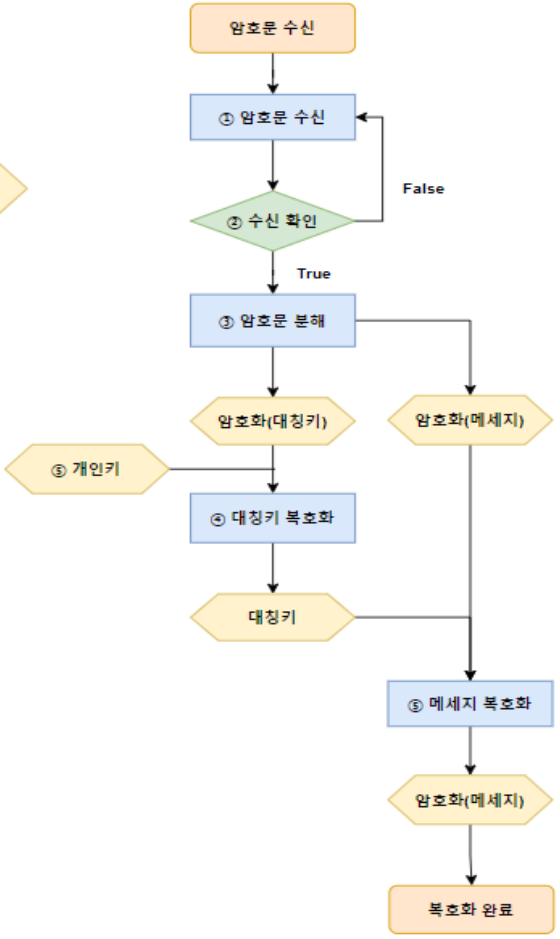

기능 흐름도
신호등
정류장
GPS
(조도센서)
교통정보관리체계
자동차
자동차 위치정보 실시간 전송
적외선 센서로 승객 유무 확인
정류장1
승객정보 전송
승객이 있으면 정지(3초)
승객이 없으면 계속 직진
자동차 에너지의 상태가 LED로 표시된다.
에너지가 부족하면 정지(7초)
에너지가 충분하면 계속 직진
NTRU 양자내성
알고리즘 암호화 통신
신호등1
신호등 색의 변화 정보 전송
신호등이 R이면 정지(10초)
적외선 센서로 승객 유무 확인
정류장2
승객정보 전송
승객이 있으면 정지(F)
승객이 없으면 계속 직진(T)
신호등2
신호등 색의 변화 정보 전송
신호등이 R이면 정지(10초)

I 알고리즘 명세서

1. 양자내성암호화 알고리즘 순서도







2. 내용

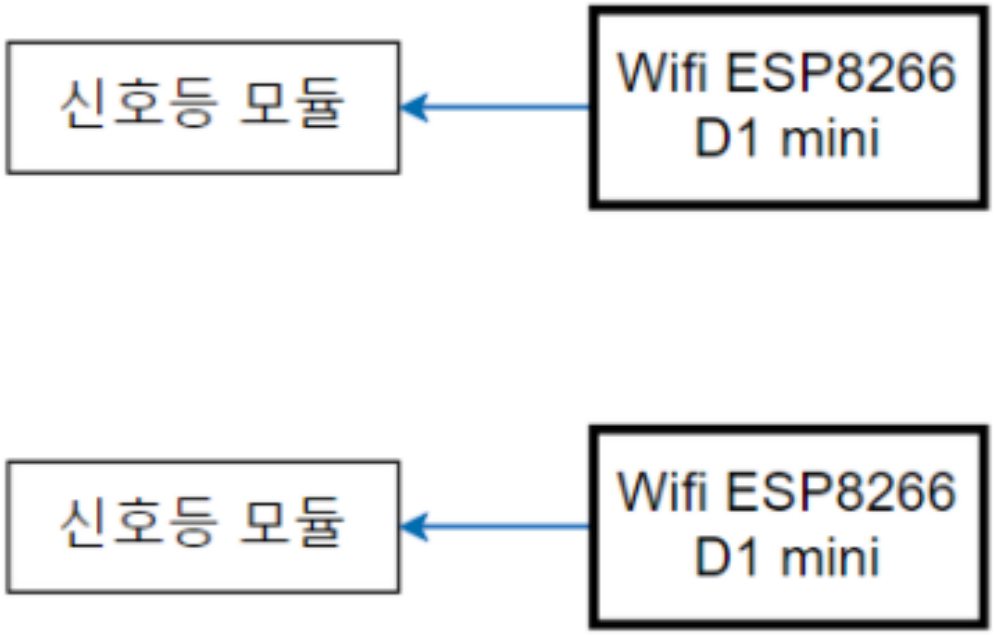
알고리즘 명세서 (키 교환)	
① 키 생성	자신의 공개키와 개인키를 생성한다. 공개키(④)는 교환하여 상대방이 암호화하는데 사용하며 개인키(⑤)는 공개키로 암호화된 암호문을 해독하는 용도로 사용된다.
② 키 교환	상대방의 공개키를 얻는다.
③ 키 교환 확인	상대방의 공개키를 얻지 했다면 ②로 얻었다면 ④로 이동한다.

알고리즘 명세서 (암호문 전송)	
① 메시지 입력	전송할 메시지를 받는다.
② 대칭키 생성	메시지를 암호화할 20byte 길이의 난수를 생성한다.
③ 메시지 암호화	②에서 생성한 대칭키를 가지고 aes-256-cbc 알고리즘으로 메시지를 암호화한다.
④ 대칭키 암호화	키 교환(④)에서 획득한 공개키를 통해 ②에서 생성한 대칭키를 NTRU 알고리즘으로 암호화한다.
⑤ 암호문 병합	③와 ④를 병합하여 보낼 암호문을 만든다. 암호문의 끝을 표시하기 위해 '.' 문자를 마지막에 추가한다.
⑥ 암호문 전송	⑤에서 만든 암호문을 TCP Socket을 통해 전송한다.

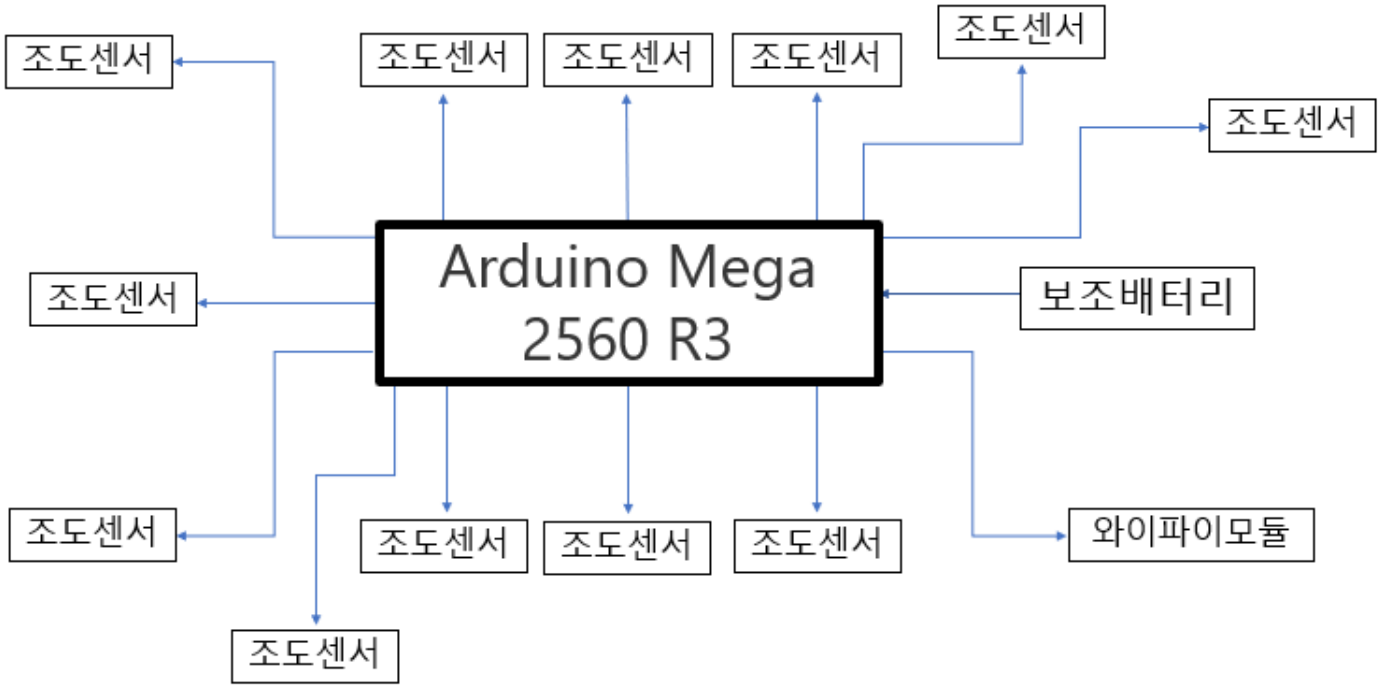
2. 내용

알고리즘 명세서 (암호문 수신)	
① 암호문 수신	암호문을 수신한다.
② 수신 확인	수신을 확인한다. '.'이 도착하면 모든 패킷이 도착한 것으로 판단한다.
③ 암호문 분해	암호문을 암호화된 대칭키와 암호화된 메시지로 분해한다.
④ 대칭키 복호화	키 교환(②)에서 생성한 개인키(⑤)를 가지고 NTRU 알고리즘으로 대칭키를 복호화한다.
⑤ 메시지 복호화	④에서 복호화한 대칭키를 가지고 aes-256-cbc 알고리즘으로 메시지를 복호화한다.

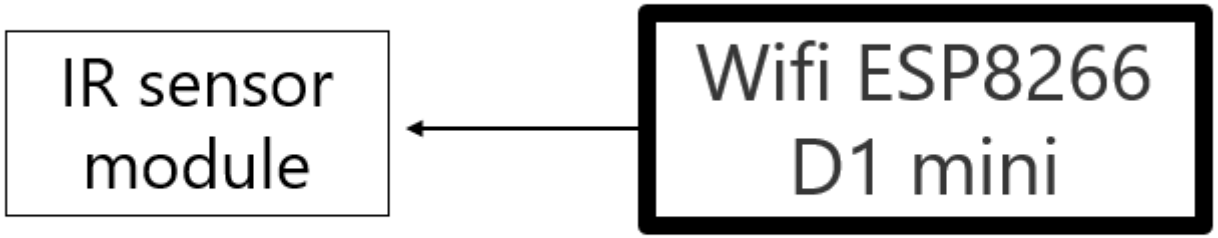
1. 인프라(신호등)



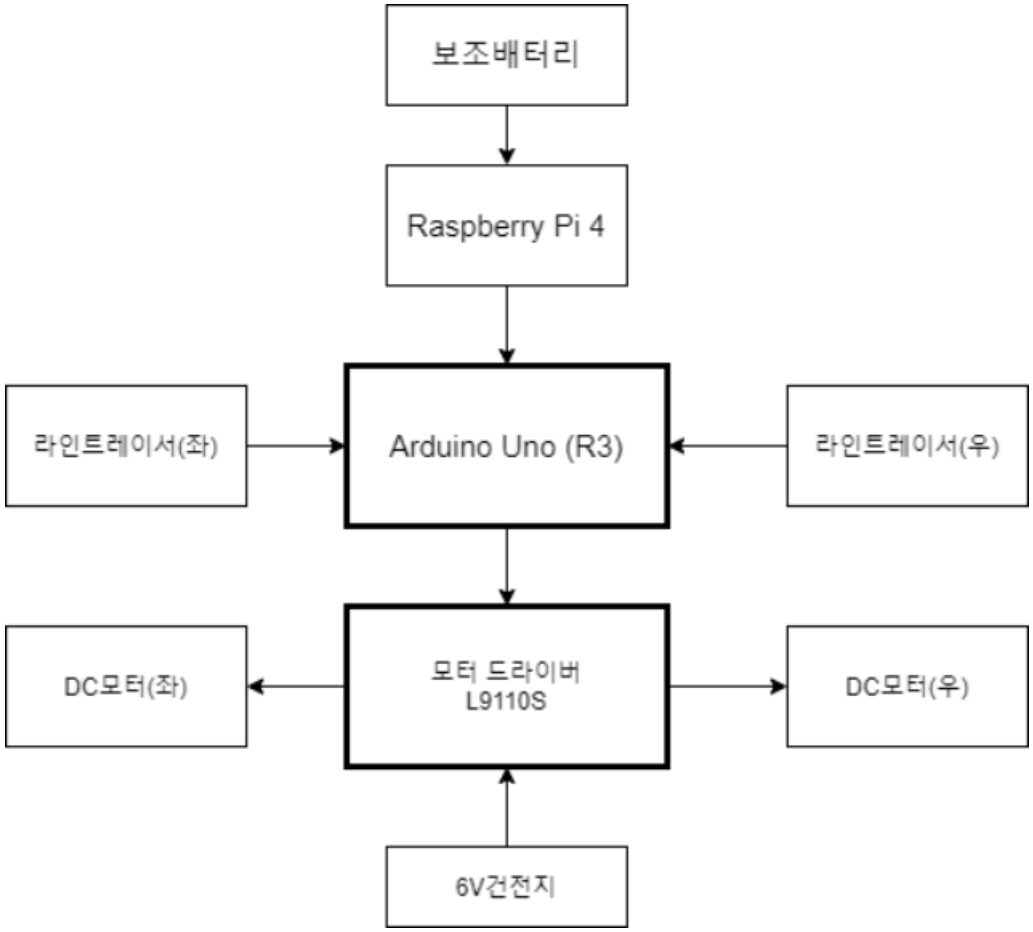
2. 인프라(위치정보)


조도센서
조도센서
조도센서
조도센서
조도센서
조도센서
조도센서
Arduino Mega
2560 R3
보조배터리
조도센서
조도센서
조도센서
조도센서
조도센서
와이파이모듈
조도센서

3. 인프라(정류장)



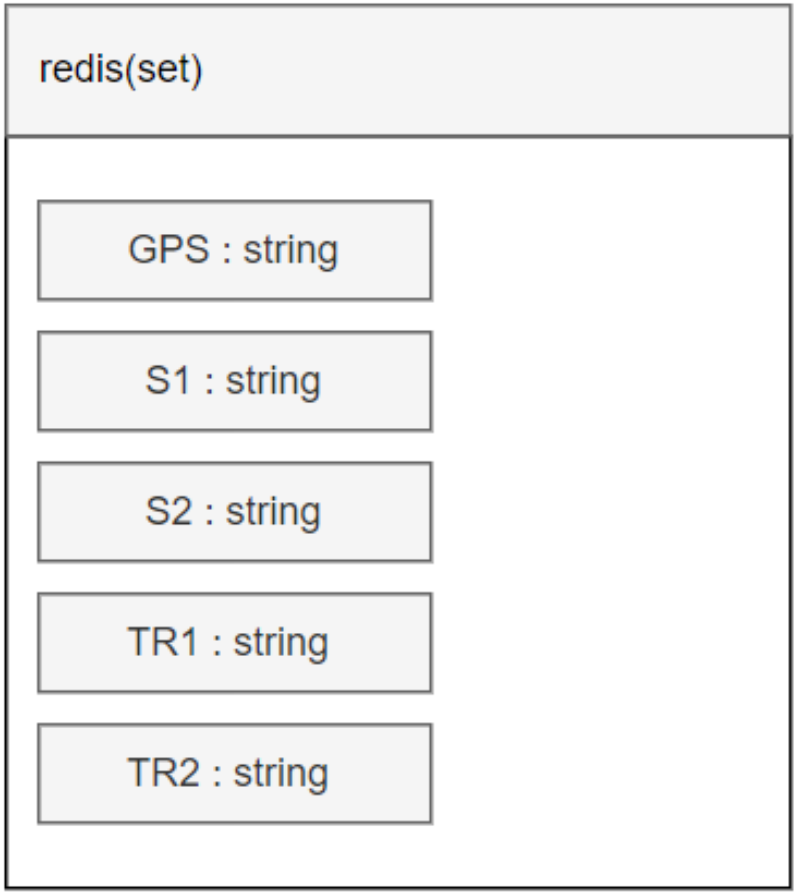
4. 자율주행자동차


보조배터리
Raspberry Pi 4
라인트레이서(좌)
Arduino Uno (R3)
라인트레이서(우)
DC모터(좌)
모터 드라이버
L9110S
DC모터(우)
6V건전지

I 프로그램 목록

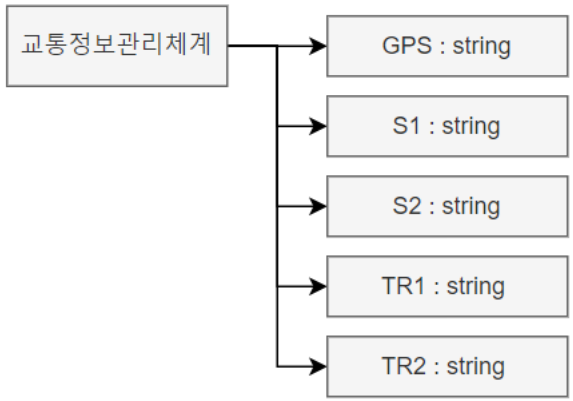
기능 분류	기능번호	기능 명
Sender	START_01	시작
	EXCHANGE_01	공개키 교환
	ENCRYPTO_01	암호화 전송
Receiver	START_01	시작
	EXCHANGE_02	공개키 교환
	DECRYPTO_01	암호문 수신

| 엔티티관계도 - ERD


redis(set)
GPS : string
S1 : string
S2 : string
TR1 : string
TR2 : string

| 테이블 정의서 - ERD

1. 테이블 구성도



2. 테이블 내용

항목명	Type	필수/선택	값 목록	설명
GPS	string	필수	0 ~ 13	차량 위치 정보
S1	string	필수	T, F	정류장 사람정보
S2	string	필수	T, F	정류장 사람정보
TR1	string	필수	R, Y, G	신호등 정보
TR2	string	필수	R, Y, G	신호등 정보

I 핵심 소스코드

1. 공개키 대칭키 생성

```
async function keyGeneration() {
var keyPair = await ntru.keyPair();
keyList.server_private = keyPair.privateKey;
keyList.server_public = keyPair.publicKey;
console.log("Key generation complete")
}
```

2. 데이터 송수신

```
webSocket.on('data', function (chunk) {
    buffers += chunk.toString();
    if (buffers[buffers.length - 1] == '.') {
        if (keyList.server_public != 0) {
            var decryptedMsg = decrypted(buffers.slice(0, buffers.length - 1), keyList.client_private);
            decryptedMsg.then(function (result){
                think(JSON.parse(result));
            });
        } else {
            keyList.server_public = new Uint8Array(buffers.slice(0, buffers.length - 1).split(','));
            console.log("Finish to exchange publicKey.");
            connection_state = "running";
            io.sockets.emit('state', connection_state);
                        catState = "T";
            carPort.write(carState);
        }
        buffers = ''; // 버퍼 초기화
    }
});
```

I 핵심 소스코드

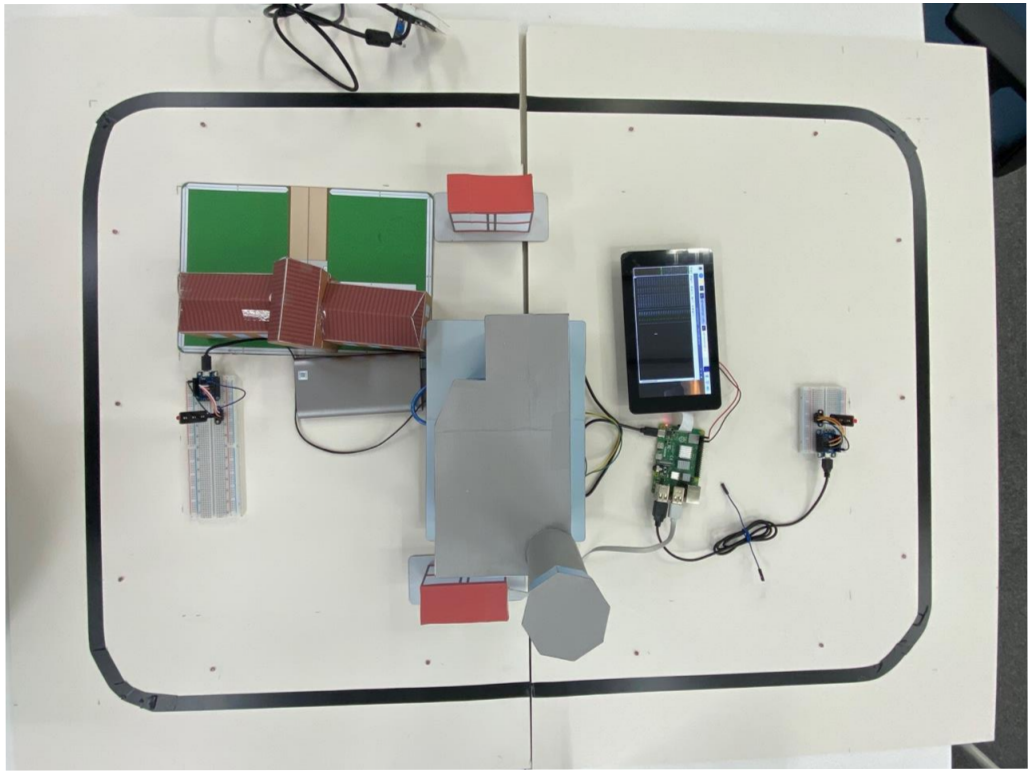
3. 암호화 전송

```
async function encrypted_send(msg, publicKey) { // 공동키 암호화
var symmetricKey = crypto.randomBytes(20).toString('base64');
var utf8_array = new Uint8Array(str.stringToUTF8Array(symmetricKey));
var encryptedKey = await ntru.encrypt(utf8_array, publicKey);
// 내용 암호화
const cipher = crypto.createCipher('aes-256-cbc', symmetricKey);
let encryptedMsg = cipher.update(msg, 'utf8', 'base64');
encryptedMsg += cipher.final('base64');
// 결합
var result = encryptedKey + ',' + str.stringToUTF8Array(encryptedMsg) + '.';
// socketIO
io.sockets.emit('send', [msg, symmetricKey, str.UTF8ArraytoString(publicKey), result]);
// 전송
socketp.write(result);
}
```

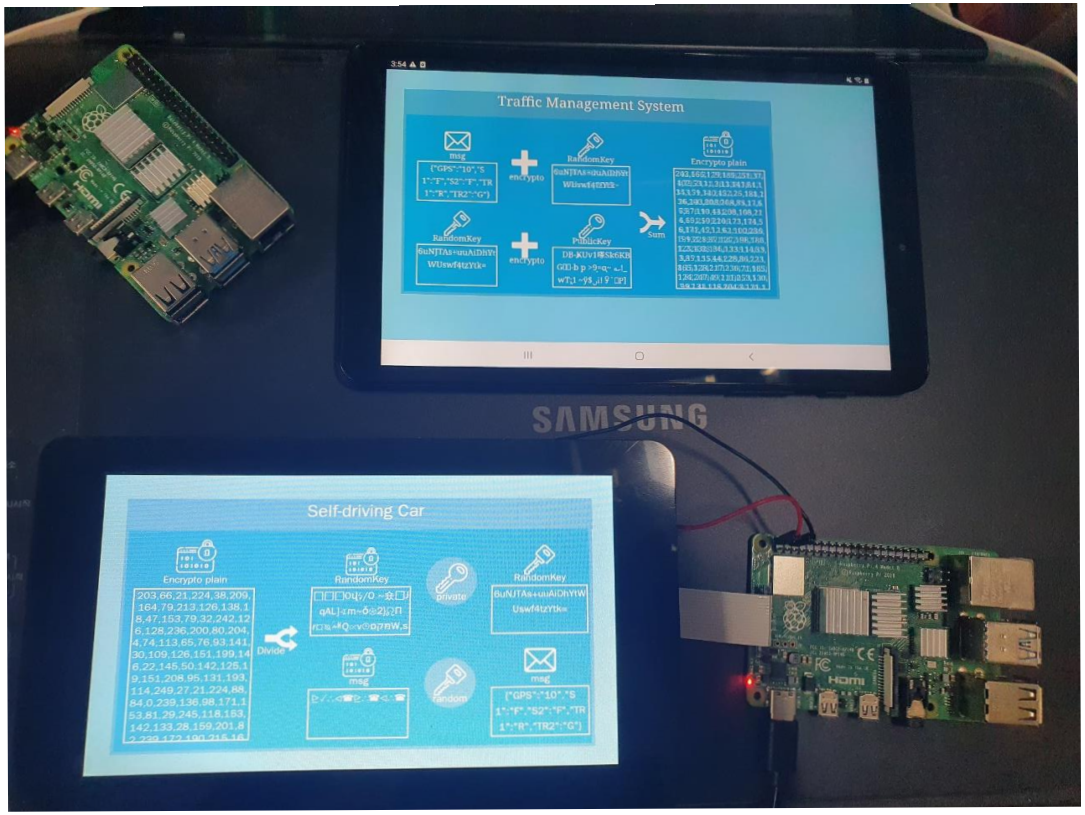
4. 암호문 복호화

```
async function decrypted(msg, privateKey) { // 대칭키와 메시지 분리
var utf8_array = new Uint8Array(msg.split(','));
var encryptedKey = utf8_array.slice(0, 1022);
var encryptedMsg = str.UTF8ArraytoString(utf8_array.slice(1022,));
// 대칭키 복호화
var symmetricKey = str.UTF8ArraytoString(await ntru.decrypt(encryptedKey, privateKey));
// 메시지 복호화
const decipher = crypto.createDecipher('aes-256-cbc', symmetricKey);
let decryptedMsg = decipher.update(encryptedMsg, 'base64', 'utf8');
decryptedMsg += decipher.final('utf8');
return decryptedMsg;
}
```

| 참조- H/W 기능 실사사진



| 참조-S/W 기능 실사 사진


Traffic Management System
msg
RandomKey
encrypto
Encrypto plain
SAMSUNG
Self-driving Car
Encrypto plain
RandomKey
Divide
private
random
msg

Thank you