

2023 학년도 학사학위논문



오늘의 번호 도어락

지도교수: 성동수

금 1조

김영우 201810467

이형주 201812116

전장군 201812285

제출일: 2023 년 6 월 3 일

경기대학교 창의공과대학 전자공학과

요 약

본 논문에서는 현재 비밀번호를 1개만 사용하는 도어락의 보안문제를 해결하기 위해 유효기간이 존재하는 비밀번호 즉 오늘의 번호를 사용하는 것으로 2가지의 비밀번호를 사용하는 도어락을 제안한다. 오늘의 번호 도어락이란 현재 우리가 사용하는 메인 비밀번호도 사용할 수 있으면서 유효기간이 있는 비밀번호를 생성하여 일시적으로 방문을 할 인원이 있는 경우 메인 비밀번호가 아닌 유효기간이 있는 비밀번호를 알려주는 것으로 현재 대다수의 메인 비밀번호만 사용하는 도어락보다 우수한 보안성을 가진 도어락이다. 현재 대다수 가정에서 사용되는 도어락의 경우 비밀번호가 1개이기 때문에 유출될 경우 심각한 보안상의 문제가 발생하게 된다. 유효기한에서 착안한 유효기간이 있는 비밀번호를 이용하는 도어락은 2가지 종류의 비밀번호를 사용하여 향후 미래에 발생할 수 있는 가까운 사이의 무단침입 문제를 유효기간이 있는 오늘의 번호를 사용하는 것으로 해결할 수 있다. 오늘의 번호 도어락은 가정에서도 적용할 수 있는 방식이다. 따라서 본 논문에서는 유효기간이 있는 비밀번호를 현재 대다수의 메인 비밀번호 1개만 사용하는 도어락에 접목시켜 타인의 의한 무단침입으로 인한 위험 및 유효기간을 통한 보안성을 확보한 오늘의 번호 도어락을 제안한다.

라즈베리파이4(Raspberry pi)를 포함하여 TT모터, 모터 드라이버 등 다양한 부품을 이용해 오늘의 번호 도어락을 제작했다. 라즈베리파이에서는 L298N 모터드라이버와 TT모터를 연결하여 입력된 비밀번호에 따라 문의 개폐여부를 결정하여 신호전송을 통해 잠금 해제 및 잠금 기능을 수행한다. 올바른 메인 비밀번호와 유효기간이 유효한 오늘의 번호가 4*4 키패드를 통해 입력되었을 경우 모터구동 동작이 발생하여 문의 개폐가 이루어진다. 그 후 문의 잠금이 필요할 때 4*4 키패드에 B버튼을 누를 경우 모터구동 동작이 발생하여 문의 잠금이 이루어진다. 만약 올바른 메인번호가 아니거나 유효기간이 지난 오늘의 번호를 입력할 경우 모터구동이 이루어지지 않게 되어 다시 4*4 키패드에 입력이 필요한 초기상태로 회귀하게 된다. 번호의 입력여부를 확인은 인디케이터를 통해 이루어진다. 또 인디케이터에 입력이 일정이상 이루어지지 않을 경우 다시 초기화 되는 기능도 있어서 제한시간이 존재한다. 이 제한시간도 사용자가 설정할 수 있다. 라즈베리파이가 PC와 동일 네트워크에 연결되어 있는 경우 PC를 종합제어기로 활용하는 방식으로 진행된다.

본 과제는 메인 비밀번호, 유효기간이 유효한 오늘의 번호, 잘못된 메인번호, 유효기간이 지난 오늘의 번호 입력상황을 고려하여 제작되었다. 실제 도어락 환경을 구현하기 위해 박스를 이용하여 구조를 제작 후 시트지와 스프레이를 통한 외관작업 과정을 거쳐 제작되었다.

목 차

제 1 장 서론.....	8
1.1 과제 목적.....	8
1.2 제품의 용도.....	10
제 2 장 기존 제품 및 기술 현황.....	12
2.1 도어락.....	12
2.1.1 도어락 종류.....	12
2.2 과제 관련 상품 조사.....	14
2.3 기술 현황 분석.....	15
2.3.1 지문인식.....	16
2.3.2 카드키.....	16
2.3.3 무한 허수시스템.....	16
2.3.4 추가한 기술.....	16
2.4 주요 사용 기술.....	16
2.4.1 TT 모터.....	16
2.4.2 L298N 모터드라이버.....	17
2.4.3 4*4 키패드.....	18
제 3 장 설계 내용.....	19
3.1 전체 동작의 개념도.....	20
3.2 설계 내용.....	20
3.2.1 키패드 - 인디케이터.....	20
3.2.2 TT 모터 - L298N 모터드라이버.....	22
3.2.3 구조 - 가동범위제한 박스.....	23
3.2.4 소프트웨어 알고리즘.....	24
제 4 장 제작 및 결과.....	29
4.1 제작 방법 및 결과 성능 시험.....	29
4.1.1 구조.....	29
4.1.2 제작과정.....	32
제 5 장 결론.....	39

5.1 요약.....	39
소감.....	39
참고 문헌	42
부록	42
부록 1.1: 부품 목록	42
부록 1.2: 회로도.....	43
부록 1.3: 코드.....	44
부록 2: 현실적 제한 요소.....	60
부록 3: 종합설계 공학문제수준	62

그림 목차

- 그림 1-1 전 연인의 무단침입 관련 기사
- 그림 1-2 여성1인 노린 주거침입 기승 관련 기사
- 그림 1-3 1인가구 통계(통계청)
- 그림 1-4 리튬이온 배터리 부품 사진
- 그림 1-5 건전지 홀더 부품 사진
- 그림 1-6 4*4 키패드 부품 사진
- 그림 1-7 LED 부품 사진
- 그림 1-8 브레드보드 부품 사진
- 그림 1-9 100Ω 저항 부품 사진
- 그림 1-10 TT모터 + 휠 부품 사진
- 그림 1-11 L298N 모터 드라이버 부품 사진
- 그림 1-12 라즈베리파이4 부품 사진
- 그림 2-1 주키와 보조키 차이점
- 그림 2-2 코맥스 무타공 푸시폴 605 도어락
- 그림 2-3 삼성 SHS-G510 강화 유리 도어락의 종류
- 그림 2-4 MI - 5200S
- 그림 2-5 삼성 SHP-DP620 푸시폴 디지털도어락
- 그림 2-6 ZHANJIAN
- 그림 2-7 L298N 핀 배치
- 그림 2-8 4x4 3x4 매트릭스 키보드 키패드 모듈 사용 키 PIC AVR 스탬프
- 그림 2-9 아두이노 4x4 16key 멤브레인 매트릭스 키패드 스위치 HAM2913
- 그림 3-1 오늘의 번호 도어락 전체 동작 개념도
- 그림 3-2 Tinkercad와 ppt 이용하여 구성한 회로도
- 그림 3-3 인디케이터와 4*4키패드 설치 위치
- 그림 3-4 레이저 센서 보조 장치와 설치 사진
- 그림 3-5 점프케이블로 길이를 연장한 모습
- 그림 3-6 TT모터와 전선을 납땜하여 연결한 모습
- 그림 3-7 제작된 바닥, 벽, 문을 서로 연결하여 구현된 모습

그림 3-8 가동범위제한 박스를 설치한 모습

그림 4-1 1안) 설계된 잠금 장치 구조

그림 4-2 2안) 설계된 잠금 장치 구조

그림 4-3 2안으로 완성된 잠금 장치 모습

그림 4-4 중간1차 완성된 모습

그림 4-5 최종 완성된 모습

그림 4-6 사용자의 편의성을 고려한 키패드 및 인디케이터

그림 4-7 키패드 및 인디케이터 정면모습

그림 4-8 상자 절단 및 결합한 모습

그림 4-9 문을 완성한 모습

그림 4-10 잠금 막대 제작

그림 4-11 잠금 막대부를 모터에 연결

그림 4-12 부품 설치구역 설정

그림 4-13 키패드, 모터 연결 후 테스트

그림 4-14 라즈베리파이(오른쪽)와 무선통신을 이용한 원격연결(왼쪽)

그림 4-15,16 소프트웨어 설계

사진 4-17 구동범위제한 박스 제작

사진 4-18 와이어를 이용한 모터 설치

그림 4-19,20 북엔드 설치 후 완성된 도어락 구조 프로토타입

그림 4-21,22 LED 인디케이터 회로 설계 및 동작확인

그림 4-23,24 각 장치를 설치하는 모습

그림 4-25,26 외관 작업 전 완성된 모습

그림 4-27,28 외관작업을 진행하는 모습

그림 4-29,30 완성된 외관

그림 부록 1-2 회로도

그림 부록 1-2 전체 회로도

그림 부록 2-3 라즈베리파이 전체 코드

그림 부록 2-4 아두이노 전체 코드

표 목차

표 1-1 최근 2년간 데이트폭력 검거현황

표 1-2 주거침입 범죄 건수 (경찰청)

표 1-3 사용 제품의 용도

표 2-1 TT모터 제원

표 2-2 L298N 모터 드라이버 제원

표 2-3 키패드 제원 비교

표 3-1 제작한 바닥, 벽, 문 규격

표 4-1 1안과 2안의 특징 비교

표 부록 1 부품 목록

표 부록 2 제한요소 분석

제1장 서론

1.1 과제 목적

현재 주거침입에 의한 범죄는 증가하는 추세이다. 가족이나 연인간의 발생하는 범죄 또한 마찬가지이다. [표 1-1] 통계에 따르면 2017년 주거침입에 의한 범죄가 481건이며, 2018년에는 707건으로 증가한 추세를 보인다. 또 다른 자료에 의하면 1인가구가 증가함에 따라 1인 가구 비율이 늘면서 스토킹, 주거 침입 범죄가 늘어나고 있다.[1] 최근 [표 1-2]경찰청에서 확인한 자료에 따르면 2016년 11,631건이었던 전체 주거침입 범죄는 2020년 18,210건으로 5년간 56.6% 증가했다. 같은 기간 여성 피해 주거침입 범죄도 2016년 6,034건에서 2020년 9,751건으로 61.6% 경기 남부의 여성 피해 건수는 2016년부터 해마다 증가하여 2020년까지 총 6,979건이다. 한편, 최근 5년간 주거침입 발생 건수는 증가하는 것과는 반대로 주거침입자 검거율은 해마다 감소하고 있는 것으로 확인됐다. 2016년 75.7%였던 검거율은 2017년 75.3%, 2018년 75.1%, 2019년 72.3%, 2020년 72.6%로 나타났다. 이와 같이 보안이 취약한 1인 가구를 대상으로 한 주거침입죄 사건이 증가하고 혐의가 인정되면 초범이라도 실형을 피하기 어렵다.[2]

구분	주거침입	폭행상해	체포,감금,협박	살인(미수/기수)	성폭력	경범죄 등 기타	계
2017년	481	7552	1189	67(17/50)	138	876	1만 303
2018년	707	7461	1089	42(16/26)	99	847	1만 245

[표 1-1 최근 2년간 데이트폭력 검거현황]

	2016	2020	증가율
전체건수	11,631	18,210	56.6%
여성피해	6,034	9,751	61.6%

[표 1-2 경찰청 발표 주거침입 범죄 건수]

아래의 [그림 1-1]에서 전 연인에 의한 무단침입이 기사화된 것을 확인할 수 있다. 이처럼 연인관계에 있던 사람의 경우 쉽게 타인의 집에 쉽게 들어갈 수 있으며 비밀번호를 아는 경우가 발생하게 된다. 그래서 가까운 사이일수록 비밀번호에 관한 보안취약점이 발견되곤 한다.[3]

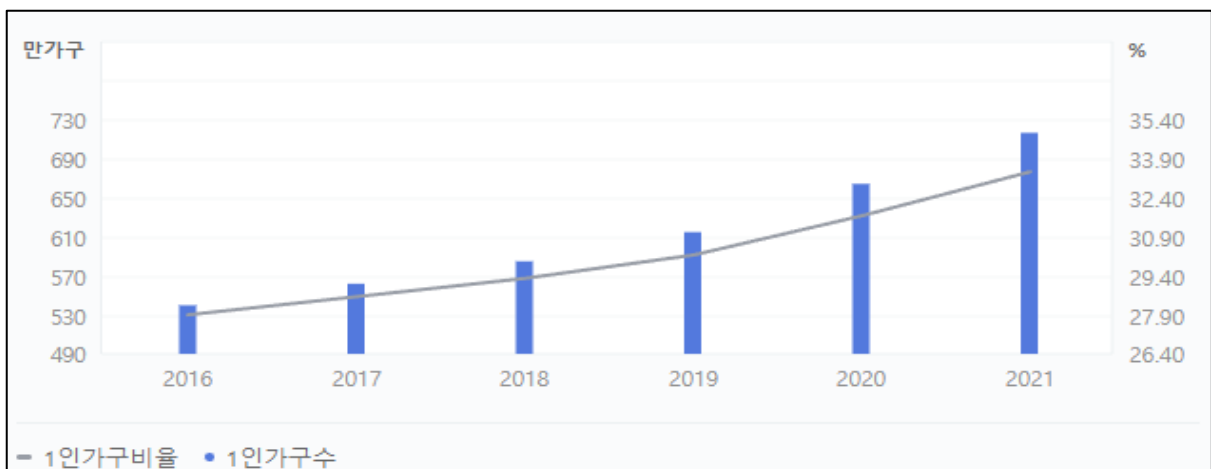


[그림 1-1 전 연인의 무단침입 관련 기사]

이러한 자료 및 통계들을 근거로 생면부지의 타인이나 가까운 사람에 의한 무단침입의 발생은 점점 증가한다고 유추할 수 있다. 누구에게나 발생할 수 있는 만큼 보안의 중요성은 점점 커진다. 특히 가족이나 연인에게 메인 비밀번호를 알려줄 경우에 미래 사이가 악화가 되어 이를 악용할 여지가 있다. 그렇기에 [표 1-2]에 나온 결과처럼 범죄건수가 증가하는 것을 확인할 수 있다. 매년 증가하는 건수를 보게 된다면 점점 무단침입에 의한 범죄는 더욱 증가할 것으로 예상된다. [그림 1-3] 1인가구 증가로 관련 1인 가구만을 노린 무단침입의 위험도 기승을 부리고 있다. [그림 1-2]는 비교적 최신 기사이며 1인 가구 중 여성을 노린 주거침입이 기승을 부리는 현 상황을 보여주고 있다.



[그림 1-2 여성 1인 노린 주거침입 기승 관련 기사]



[그림 1-3 1인가구 통계(통계청)]



이런 문제는 무단침입을 방지하는 역할인 도어락이 중요하게 된다. 하지만 무단침입 건수의 증가현상을 보면 제 역할을 완벽히 하지 못하는 것을 알 수 있다. 그러므로 현재 도어락이 가진 문제점이 있기에 무단침입이 꾸준히 발생한다고 볼 수 있다. 우리가 일상에서 주로 쓰는 도어락은 비밀번호가 1개이다. 즉 단 하나의 비밀번호가 유출이 된다면 보안이 매우 취약해진다. 이를 활용하여 전 연인이나 채무관계에 있는 친족이 무단침입하는 경우도 발생하게 된다. 그래서 비밀번호를 1개만 사용하는 방식이 제일 큰 문제라고 판단된다.

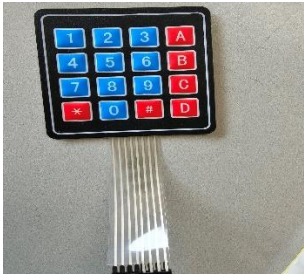




문제를 해결하기위한 방법으로 유효기간을 가진 비밀번호 즉 오늘의 번호와 메인 비밀번호를 같이 사용하는 방식을 제안하고자 한다. 기존의 방식에 유효기간을 가진 비밀번호를 더하여 보안성을 강화하는 것이다. 잠시 주거지에 들어올 필요가 있는 사람에게 오늘의 번호를 전달하는 것으로 향후 미래에 발생할 수 있는 무단침입의 가능성을 배제할 수 있다.

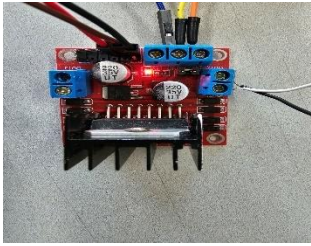
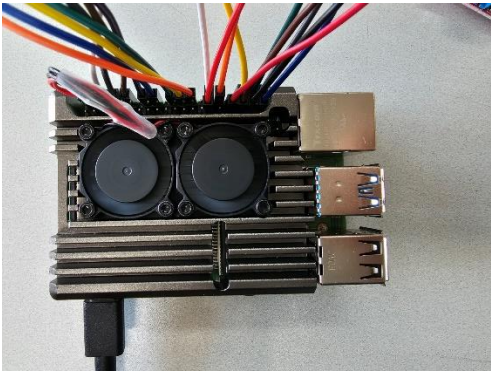
따라서 본 논문에서는 현재의 도어락에 있는 보안적인 취약점을 개선하여 사용자와 안전을 위해 유효기간이 있는 비밀번호 즉 오늘의 번호를 사용하는 도어락을 제안한다. 현재 제조되는 도어락에도 적용이 가능한 방식이다. 해당 제품은 현재 도어락 시장에 존재하지 않는 제품으로, 이를 제안하여 도어락을 통한 보안의 발전과 무단침입의 감소에 기여하고자 한다.

1.2 제품의 용도

본 과제에 이용된 제품의 용도를 [표 1-3]과 같이 기술하였다.

전원부	리튬이온 배터리  [그림 1-4]	모터의 이상적인 동작을 위해 정격 전압을 인가한다. L298N 모터 드라이버에 연결하여 안정적인 구동을 가능케 한다.
전원부	건전지 홀더  [그림 1-5]	L298N 모터 드라이버의 이상적인 동작을 위해 정격 전압을 인가한다. 건전지 홀더를 통해 전력을 통합하여 제공한다.

입력부	<p>4*4 키패드</p>  <p>[그림 1-6]</p>	<p>행과 열을 이용하여 입력을 감지하는 방식의 4*4키패드이다. 각각 4개의 행과 열로 구성되어 있어 연결핀 또한 8개로 구성되어 있다. 사용자 편의성을 위해 숫자와 문자가 표시된 키패드를 사용하였다.</p>
출력부	<p>LED</p>  <p>[그림 1-7]</p>	<p>인디케이터에 사용되는 부품으로 입력이 발생할 때 입력여부를 확인할 수 있게 해준다.</p>
출력부	<p>브레드보드</p>  <p>[그림 1-8]</p>	<p>인디케이터의 기관에 해당한다. LED와 저항, 점프케이블 등이 연결된다.</p>
출력부	<p>100Ω 저항</p>  <p>[그림 1-9]</p>	<p>안정적인 LED 동작을 위해 브레드보드 위에 연결하여 사용한다.</p>
동작부	<p>TT모터 + 휠</p>  <p>[그림 1-10]</p>	<p>전방, 후방 동작이 가능하여 문의 개폐장치로 적합하여 사용했다. 충분한 동력을 제공하기에 잠금 막대 설치에도 문제없이 충분한 동력을 가진다.</p>

<p>동작부</p>	<p>L298N 모터 드라이버</p>  <p>[그림 1-11]</p>	<p>모터의 안정적인 동작을 가능하게 해준다. 입력된 신호에 따라 알맞게 모터동작이 가능하도록 TT모터에 연결하여 사용한다.</p>
<p>제어부</p>	<p>라즈베리파이4</p>  <p>[그림 1-12]</p>	<p>라즈베리파이4를 이용하여 입력에 대한 신호처리를 하여 모터에 신호전송 여부를 결정한다. 문의 개폐여부도 판단하여 해당 신호도 제공한다.</p>

[표 1-3 사용 제품의 용도]

제2장 기존 제품 및 기술 현황

2.1 도어락(door lock)

비열쇠식 잠금장치로 손잡이 유무에 따라 주키와 보조키로 나뉜다. 다양한 보안 방식 및 기능에 따라 보안성과 편의성의 차이가 있으며, 고장났을 경우 제품 일부를 파손할 수 있다.[4]

2.1.1 도어락 종류

도어락은 우리가 사는 주거지에서 흔히 볼 수 있다. 최근에는 잠금 및 잠금해제 방식에 의해 그 종류가 구분된다. 또 설치하는 방식으로도 분류되기도 한다. 대략적으로 4가지 종류로 구분할 수 있다.

주키형 도어락은 손잡이와 디지털도어록 일체형으로 기능에 따라 터치식과 버튼식으로 나뉜다. 현관 손잡이를 떼어내고 설치하므로 방법이 다소 복잡하고 설치비용 여부의 고려가 필요하다.

보조키 도어락은 주키 도어록보다 설치방법이 간편하고, 타공비가 적게 들어 경제적이기 때문에 자주 이사하는 경우에 적합하다.

도어락 손잡이가 없으므로 기존 현관문의 현관정 위에 설치해야 하며, 현관정이 없을 경우 별도의 설치가 필요하다. 푸시풀 도어락은 밀거나 당기는 한 개의 동작만으로 개폐할 수 있다. 집 밖에서는 당기면서 열고, 집안에서는 밀면서 여는 방식이기에, 문열림이 편리해지며 비상시에 안전하고 신속하게 문의 개폐가 가능하다는 특징이 있다. 강화 유리문 도어락은 문 손잡이에 날개를 걸어서 설치하는 일반형 타입, 문 측면에 편리하게 끼우는 것을 통해 설치하는 클립형 도어락이 있다.

	주키 (손잡이 일체형)	보조키 (손잡이 분리형)
형태		
손잡이	있음	없음
설치 난이도	복잡함	간단함
가격	비쌌	저렴함

주키와 보조키의 차이점

[그림 2-1 주키와 보조키 차이점]



[그림 2-2 코맥스 무타공 푸시풀 605 도어락]



[그림 2-3 삼성 SHS-G510 강화 유리 도어락의 종류]

2.2 과제 관련 상품 조사

현재 시장에서 판매되는 도어락 중에서 기존에 것과 차별되는 것을 조사했다. [그림 2-3]은 밀레사의 MI-5200S이다. 이 제품의 특징으로는 무타공방식으로 편리한 설치 방식과 특히 무한허수 시스템을 탑재했다. 무한허수 시스템이란 임의의 숫자버튼을 제한없이 입력 후 마지막에 문을 열기 위한 비밀번호를 마지막에만 입력하여 보안성을 강화한 것이다. MI-5200S의 문제점으로는 결국 공유되는 비밀번호를 1개만 사용한다는 것인데 이는 기존의 도어락과 큰 차이가 없기에 보안적인 문제에 직면할 수 있다.

[그림 2-5]은 삼성 SHP-DP620 제품이다. 이 제품은 가족 귀가 문자 알림, 가족의 출입이력을 조회, 블루투스를 통한 문열림, 푸시폴 타입의 손잡이라는 특징을 가진다. 삼성 SHP-DP620의 문제점은 해당 application이 제대로 작동하지 않는 상황이 발생할 수 있다. 즉 application의 의존도가 높다는 문제가 있다.

[그림 2-6]은 중국에서 수입한 ZHANJIAN의 제품이다. 특징으로는 비밀번호, 지문사용, 카드키 사용가능이 있다. 문제점을 분석하자면 보안수단 중 지문과 카드키 자체는 공유가 힘들기에 안전하다고 볼 수 있으나 급할 때는 사용자 편의성이 불편하다는 점이 있고 특히 카드키는 분실의 위험이 있다.



[그림 2 - 4 MI - 5200S] [그림 2 - 5 삼성 SHP-DP620 푸시풀 디지털도락]



[그림 2 - 6 ZHANJIAN]

하지만 기존의 시장에서 판매되고 있는 제품을 분석한 결과 여러 기능이 있으나 비밀번호는 하나만 사용한다는 공통점이 있다. 이는 현재 도어락이 가진 보안상의 취약점이라고 판단된다. 따라서 우리는 사용자 편의성을 위해 기존의 도어락 방식을 유지하면서 유효기간이 있는 비밀번호를 도입하여 2가지 유형의 비밀번호를 사용하여 보안성을 더욱 강화한 도어락을 제안한다. 이는 기존 하나의 비밀번호만 사용하던 도어락과 비슷하면서도 기존 도어락의 문제를 해결할 수 있는 도어락이다.

2.3 기술 현황 분석

과제 관련 상품에서 조사한 것과 같이 다양한 기술이 현재 도어락에 적용되어 있다. 현존하는 도어락 관련 기술은 2.2절에서 조사한 바와 같다. 추가 부가적인 기능은 모든 도어락에 비슷하게 접친 것을 확인했다.

2.3.1 지문인식

먼저 지문인식 기능의 경우 주거지 실거주자만 사용가능하다는 점이 있기에 보안적으로는 우수하다고 볼 수 있으나 잠시 주거지를 들어갈 인원 에 따라 불편사항이 발생한다. 예를 들어 가족의 경우 지문을 미리 등록하면 되지만 가전제품 설치기사나 실거주가 부재 시 주거지에 들어가기 위해 비밀번호를 알려줘야 하는 상황이 발생한다. 이럴 경우 보안의 취약점이 드러나게 된다.

2.3.2 카드키

카드키의 경우 사용자만 들고 다닌다는 점에서 보안성이 있다 볼 수 있으나 카드키를 분실할 경우 재발급을 하는 번거로움이 있으며 카드키를 습득한 타인이 악용할 수 있다는 문제가 있다. 또 카드키가 RFID 방식이라면 복제기를 이용하여 복제가능하다는 보안취약점이 있다.

2.3.3 무한 허수시스템

무한 허수시스템은 번호를 훔쳐보거나 도어락에 밀가루나 형광펜같이 입력한 부분이 표시나도록 하는 방법으로 하여 비밀번호 습득하는 방식에 대응할 수 있기에 좋다고 볼 수 있다 그러나. 실질적인 비밀번호는 1개만 사용하기에 앞부분에서 수많은 허수를 입력하고 마지막에 비밀번호를 입력한다 하여도 비밀번호가 공유될 경우 무한 허수시스템의 허점이 드러나게 된다. 그리고 사용자에 따라 무한허수로 입력하는 수 개수가 차이 나기에, 무한 허수를 적게 입력하는 사용자일수록 보안이 더욱 취약해진다.

2.3.4 추가한 기술

기존의 도어락 개념에서 유효기간이라는 개념을 추가하여 보안성을 강화했다. 지금까지 사용자들이 사용한 방식을 유지하면서 유효기간이 있는 비밀번호를 추가하는 것으로 사용자 편의성과 보안성 2가지를 동시에 얻을 수 있었다. 앞서 설명한 기술에서 보았듯이 유효기간을 가진 비밀번호를 탑재한 도어락은 현재 시장에서 판매되지 않음을 알 수 있다. 1개 비밀번호를 사용하여 공유되거나 유출되는 문제에 대한 대책으로 유효기간이 존재하는 오늘의 번호를 사용하는 도어락을 설계하였다. 또 실질적인 문의 크기제작에 어려움이 있어 소형 세트장 사이즈로 크기를 줄여 제작하게 되었다.

잠금 및 잠금해제에 사용되는 TT모터와 관련 기술 등을 다음 2.4절에서 설명된다.

2.4 주요 사용 기술

2.4.1. TT모터

TT모터는 내부 중간에 기어박스가 내제되고 양쪽으로 샤프트 축이 나와있는 DC 기어 모터이다. TT모터라고 불리는 이유는 전체 외관을 보았을 때 T자형처럼 보이기 때문이다. 기어와 샤프

트를 제외하면 일반 모터와 동일한 외관을 가진다. 주로 수형 모터 자동차, RC카, 초소형 전기 로봇에 사용된다.

부품	단일축사이즈	쌍축사이즈	공회전속도	최대토크	부하전류	작동전압	권장전압
TT 모터	22*29*69mm	22*36*69mm	1:48 (3V 연결시)	800gf cm min (3V 연결시)	70mA (3V 연결시) 250mA MAX	3V~12V 직류	6V~8V 범위 내

[표 2-1 TT모터 제원]

문의 잠금 및 잠금해제가 이루어져야 하기에 전류, 후류 구동이 가능한 모터가 필수적이다. TT 모터는 전류, 후류 구동이 모두 가능하며 충분한 동력으로 힘을 제공하기에 채택되어 사용되었다.

2.4.2 L298N 모터드라이버 [5]

L298N 모터드라이버는 듀얼-H-브릿지 모터 드라이버 모듈이다. 기능으로는 2상스텝모터, DC모터 제어 및, PWM 제어를 통해 모터속도 제어도 가능하다. 라즈베리파이 핀에서 5V 전압이 출력되나 전류가 매우 작기에 모터제어 부분에서 많이 부족하다. 그래서 5V 전압을 신호로 사용하고 큰 동력이 필요하기에 모터는 그 신호를 사용하여 별도의 외부 전력을 끌어다가 구동할 수 있게 하는 역할을 한다.

부품	동작전압	동작전류	모터입력 전압	모터허용 전류	최대소비 전력	PCB사이즈
L298N	DC 5V	0mA ~ 36mA	DC 5V ~ 35V	2A (MAX)	25W	43*43*27mm

[표 2-2 L298N 모터 드라이버 제원]

[그림 2 - 7]핀 관련 설명은 다음에 기술한다.

VS, GND핀은 모터를 구동할 구동 전력 관련 핀이다. 모터 구동 전원에 따라 5V ~ 35V의 전원이 연결 가능하고 모터를 구동할 수 있게 하는 전원이므로 충분한 용량의 공급은 필수적이다.

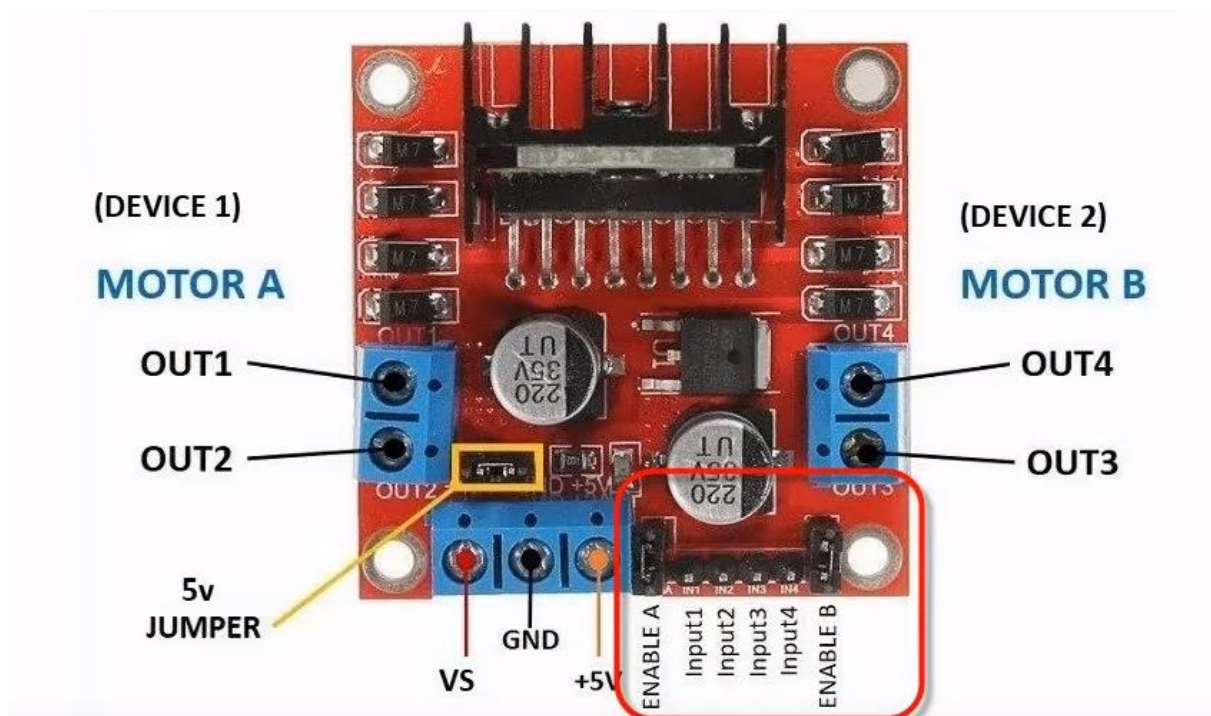
[그림 2 - 7]의 좌, 우측의 OUT단자에 실제 모터가 연결이 되고 L298N 모터 드라이버는 전류, 후류 양방향 구동 제어가 가능하다. 만약 2개의 모터 연결이 필요하다면 OUT1,2와 OUT3,4에 각각의 모터를 연결하면 된다.

[그림 2 - 7]의 빨간 박스 부분에 있는 2개의 점퍼핀=점프케이블(ENABLE A, B)을 포함하여 총 6개의 핀이 있는데 모두 아두이노 또는 라즈베리파이와 연결하여 모터의 구동을 제어하는데 사용되는 제어핀이다. 그 속에서 좌측 ENABLE A, INPUT1, INPUT2 핀은 좌측 OUT1, 2에 연결된 모터를 제어하게 되고, 우측 ENABLE B, INPUT3, INPUT4 핀은 좌측 OUT3, 4에 연결된 모터를 제어하게 된다.

ENABLE핀은 모터의 속도를 제어하며 점프케이블로 연결되어 있다. 그러나 점프케이블로 연결된 경우 모터의 속도는 제어 불가능해지고 최대 속도로만 동작하게 된다.

INPUT 핀은 모터의 구동방향을 결정해주고 INPUT 1에 High 신호, INPUT 2에 LOW 신호를 인가하면 모터는 시계방향(혹은 역시계방향)으로 회전하게 되며, INPUT 1에 LOW 신호, INPUT 2에 HIGH 신호를 인가하면 모터는 반대로 구동되어 역시계방향(혹은 시계방향)으로 회전한다.

라즈베리파이에서 모터의 속도를 제어하려면 라즈베리파이와 모터 드라이버에 있는 ENABLE핀을 라즈베리파이 PWM출력이 가능한 핀에 연결할 필요가 있다.



[그림 2-7 L298N 핀 배치도]

2.4.3 4*4 키패드

비밀번호의 입력을 위한 키패드를 선택하기 위해 2가지를 비교했다.

제품명	사진	특성
3x4 매트릭스 키보드 키패드 모듈 사용 키 PIC AVR 스탬프	 [그림 2-8]	가격 : 1976 납땜 필요 판매처: 알리익스프레스 배송기간: 6월 배송예정 전자세금계산서: 불가능
아두이노 4x4 16key 멤 브레인 매트릭스 키패드 스위치 HAM2913	 [그림 2-9]	가격 : 1100 납땜 불필요 판매처: JENO MALL(국 내) 배송기간: 2~3일 전자세금계산서: 가능

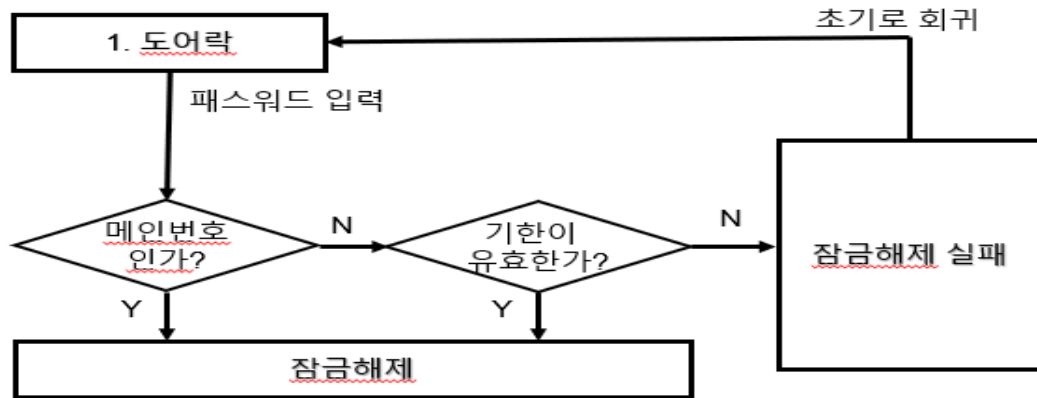
[표 2-3 키패드 제원 비교]

컴프케이블의 여유가 있으므로 라즈베리파이에 있는 핀들을 최대한 활용하고자 했다. 또 [그림 2 - 9]의 경우 행과 열로 입력을 구분하는 방식이라 코드작성의 용이성이 있다. 또 케이블 부분이 유연성이 있다는 부분에서 설계 시 배선에 강점이 있음을 확인하였다.

따라서 조사한 두 종류의 키패드를 상황 및 일정을 고려하여 [그림 2 - 9]를 구매하기로 결정했다.

제3장 설계 내용

3.1 전체 동작의 개념도

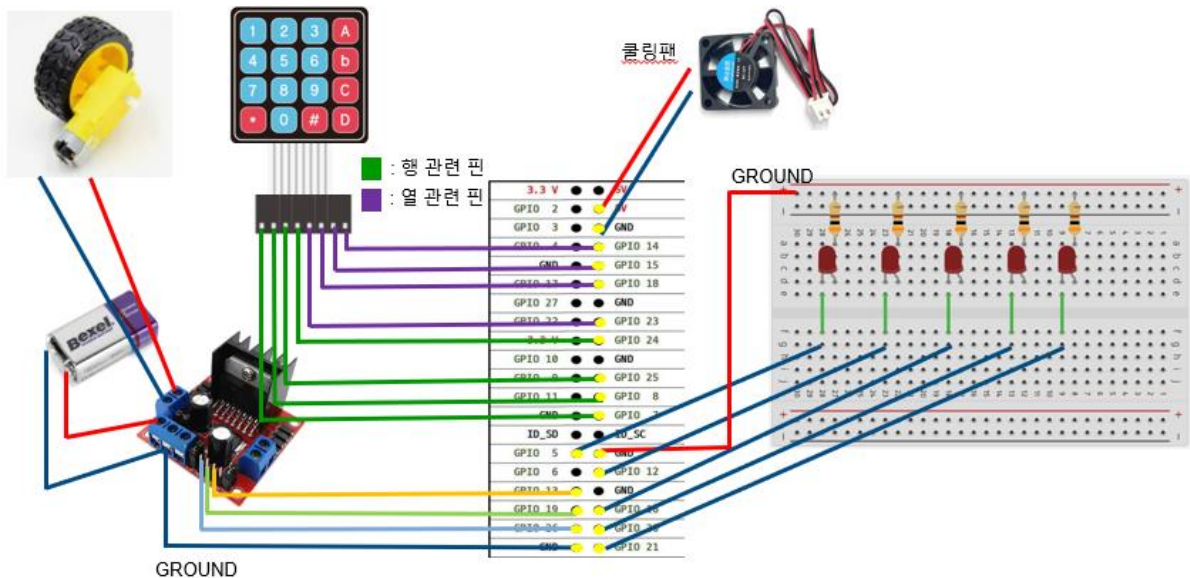


[그림 3-1 오늘의 번호 도어락 전체 동작 개념도]

[그림 3-1]은 오늘의 번호 도어락의 순서도이며 상세 내용은 아래 절에서 설명한다.

3.2 설계 내용

3.2.1 키패드 - 인디케이터 - 점프케이블



[그림 3-2 Tinkercad와 ppt 이용하여 구성한 회로도]

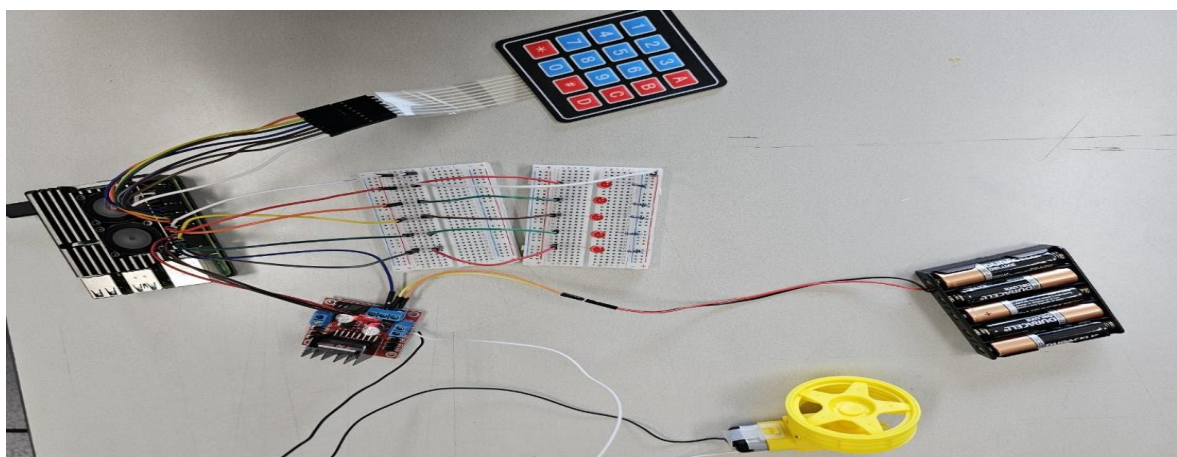
키패드를 통해 입력을 받게 된다. 입력이 인가되면 입력 여부를 인디케이터를 통해 확인 가능하다. 만약 중복 입력이 이루어지면 번호 4개를 다 입력시켜 일부러 틀리게 하여 초기 상태로 되돌아가거나 일정 이상 시간동안 아무 입력이 도중에 없게 되면 clear 동작이 진행되어 인디케이터의 LED의 점등이 꺼지게 되어 초기 입력대기 상태로 돌아가게 된다.

LED가 5개인 이유는 문자1개 + 비밀번호 4자리 조합을 사용하기 때문이다. 잠금을 해제를 위해서는 잠금해제 문자 A를 먼저 누른 뒤에 비밀번호 4자리를 입력한다. 잠금이 해제된 후 다시 문을 잠그거나 집 밖을 나설 때 B버튼만 누르면 잠금 장치가 구동되어 다시 잠금이 이루어진다. 이 방식을 채택한 이유는 차후 업그레이드나 추가기능을 구현할 때 남아있는 문자 C, D를 사용할 가능성이 있기 때문에 C, D문자를 활용한 비밀번호 조합 가능성도 열어 두었다.



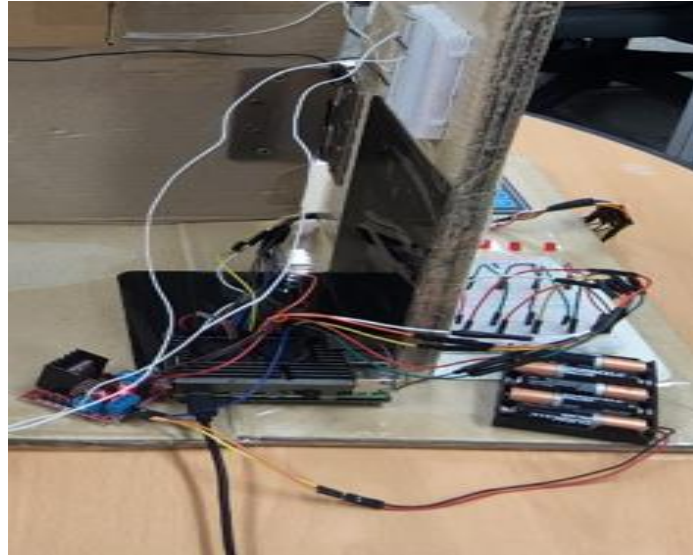
[그림 3-3 인디케이터와 4*4키패드 설치 위치]

또 초기에 인디케이터가 없을 때 키패드에 강한 힘으로 누르거나 한 번 입력한 것이 2번 입력된 것으로 인식되는 문제가 발생했다. 이를 사용자 편의성에 맞춰 알림 역할을 할 것을 설계하게 되었다. 따라서 브레드보드에 입력 여부를 확인할 수 있게 해주는 LED, 안정적인 회로 구동을 위해 설치한 100Ω 저항, 기판역할을 수행하기 위해 설치한 브레드보드를 구성하여 인디케이터를 설계하였다.



[그림 3-4 실제 완성 초기 회로도의 모습]

초기 설계한 회로도에 맞춰서 알맞게 구성 후 동작을 확인하였다. 그러나 4*4키패드와 인디케이터가 라즈베리파이로부터 가깝다는 문제가 있었다. 원활한 설치 가능 구역을 확보하기 위해 [그림 3-5]와 같이 점프케이블을 2~3중으로 연장한 뒤 테이프로 안정적인 연결성을 확보한 뒤 구현한 도어락 구조물에 설치하였다.



[그림 3-5 점프케이블로 길이를 연장한 모습]

3.2.2 TT모터 - L298N모터드라이버 - 리튬 배터리



[그림 3-6 TT모터와 전선을 납땜하여 연결한 모습]

단순히 TT모터에 전선을 연결해도 동작이 가능하나 전선이 흔들리는 등의 문제가 발생하여 안정적인 동작이 지속되지 않는 문제를 해결할 필요가 있었다. 따라서 [그림 3-6]과 같이 TT모터의 금속부분과 전선을 납땜을 통해 고정하여 안정적으로 전력을 공급받을 수 있도록 하였다.

TT모터에 라즈베리파이를 연결하면 5V의 신호가 인가되나 모터 구동에 필요한 전력은 부족하다. 따라서 모터 구동 전력을 공급할 드라이버가 필요하기에 L298N 모터드라이브를 채택하여 사용했다. TT모터에 L298N모터를 연결하여 모터 구동에 필요한 전력을 공급받도록 연결한다. 복

수개의 리튬 배터리를 건전지 홀더를 이용하여 L298N 모터 드라이버에 연결하여 모터 구동 시스템을 구축했다.

TT모터, L298N 모터 드라이버, 리튬 배터리의 원활한 연결이 이루어졌기에 4*4 키패드에 입력된 비밀번호에 유효 여부에 따라 발생하는 신호가 라즈베리파이에서 발생하여 L298N 모터 드라이버로 전송되어 모터의 구동여부를 결정하게 된다.

모터관련 코드를 수정하는 것을 통해 모터의 가동 범위 및 유지시간 등을 설정하여 커스텀마킹도 가능하다. 모터의 강한 힘이 있기에 잠금 장치에 속한 잠금 막대가 안정적인 잠금 역할을 할 수 있도록 받침대 및 가동범위를 제한하는 박스를 설치하였다.

3.2.3 구조 - 가동범위제한 박스

실제 도어락이 설치된 환경과 유사한 구조를 박스를 이용하여 제작한다. 두꺼운 두께가 필요한 경우 복수개의 박스를 겹쳐서 테이프를 사용하여 하나의 구조로 만들면 내구성도 확보된다. 이 방식으로 바닥, 벽, 문을 각각 제작한다.

	바닥	벽	문
가로	73 cm	20 cm	33 cm
세로	52 cm	52 cm	52 cm

[표 3-1 제작한 바닥, 벽, 문 규격]

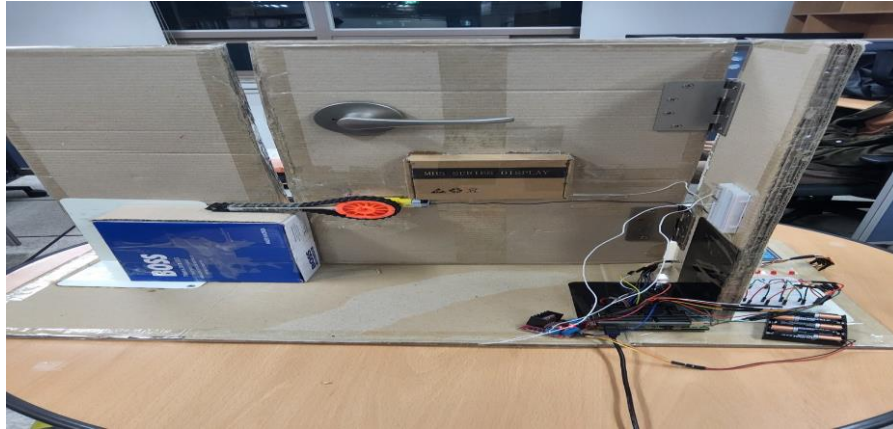
1점의 규격은 [표 3-1]와 같으며 단단한 내구성을 갖기 위해 총 4점으로 제작한다. 바닥위에 문과 벽을 세워야 하는데 고정시킬 것으로 북엔드를 사용한다. 벽의 경우 벽 양쪽에 북엔드를 위치시켜 단단히 벽을 세울 수 있도록 벽이 고정되게 북엔드르 바닥에 나사로 고정한다. 문의 경우 이미 고정된 벽에 경첩을 설치하여 실제 문을 구현하는 것과 동시에 바닥위에 세워지도록 한다.

이 과정을 마치면 아래 [그림 3-7]과 같은 모습을 하게 된다.



[그림 3-7 제작된 바닥, 벽, 문을 서로 연결하여 구현된 모습]

모터를 문에 설치한 후 시험가동을 하게 되면 중력에 의한 불규칙적인 동작범위를 보인다. 따라서 이를 해결하기 위해 잠금 장치의 잠금 막대의 가동범위를 제한하는 박스의 설치 필요하다. [그림 3-9]처럼 가동범위제한 박스를 설치하게 되면 잠금 막대의 가동 범위가 제한되어 안정적으로 일정한 범위의 동작이 가능해진다.



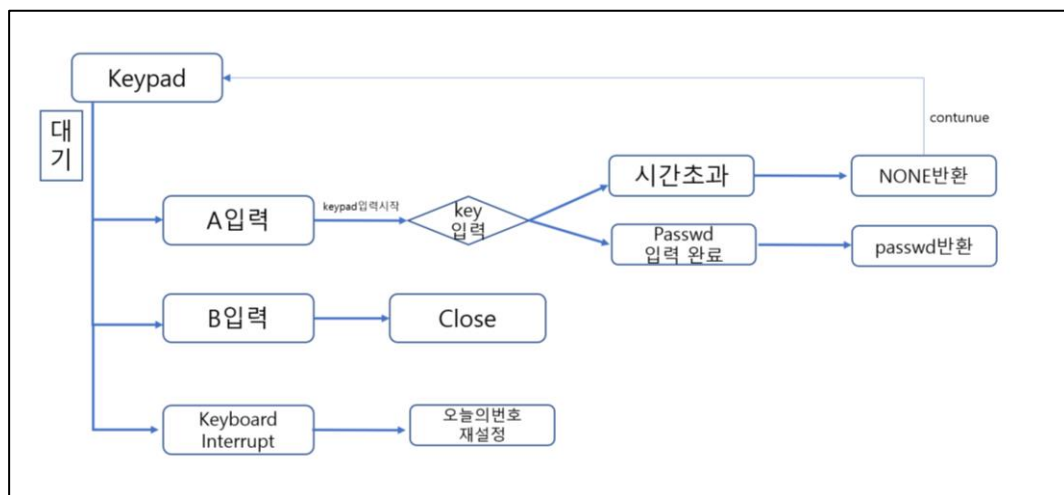
[그림 3-9 가동범위제한 박스를 설치한 모습]

그리고 잠금 막대의 모터의 회전을 견디고 가동범위제한 박스에 충격을 견디도록 내구성을 확보하기 위해 체인을 연결하여 잠금 막대의 안정성을 향상시킨다. 반복적인 실험에도 견디는 구조의 결과는 [그림 3-9]와 같다.

3.2.4 소프트웨어 알고리즘

주요 소프트웨어 알고리즘에는 숫자패드를 사용하여 4x4의 입력을 받을수 있는 keypad.py와 전 반적인 문의 개폐 및 doorlock의 역할을 해주는 doorlock.py, 문에 걸린 자물쇠를 열고 닫을 때 필요한 motor.py 마지막으로 오늘의 번호를 설정할 때 필요한 number_of_today.py가 있다.

—keypad



[그림 3-10 keypad flow chart]

Keypad 에 연결된 핀에는 키패드 인식시 행을 인식하기 위한 핀번호와 열을 인식하기위한 핀 번호 그리고 LED 와 연결된 핀번호가 있습니다. Timeout 에 keypad 가 활성화되고 오랫동안 passwd 를 입력하지않을시 초기화하기 위한 시간을 설정했다. 물리적 GPIO 핀에 설정변수들을 mapping 하여 연동시켰다. Readline 함수는 입력받은 행번호에 해당하는 키패드의 가로줄에 전류를 흘려보내고 키패드의 세로줄을 읽어 눌린 버튼의 문자를 입력받은 문자 리스트와 비교하여 입력 문자를 알아내고 그것을 전역변수인 키패드 버퍼에 저장한후 현재 패스워드의 길이를 반환합니다.

여기서 가로줄을 읽는다는 표현은 함수가 입력받은 행번호에 해당하는 핀에서 가로줄에 전류가 흐르도록 GPIO 라이브러리의 output 함수로 특정한 가로줄의 4 개의 버튼을 활성화합니다. 그 4 개중 하나의 버튼을 누르면 keypad 내부의 가로선과 세로선의 전류가 합선되면서 전류가 세로줄로 흘러 특정 열번호에 해당하는 입력핀에 전류가 흘러들어갑니다. 이 방식을 사용하여 keypad 에서 입력문자를 알아낼수있습니다. 여기서 global passwd 를 사용한 이유는 python 에서 전역변수가 readonly 로 사용되는 상수가 아니라 가변적이 변수일 때 그 변수를 함수 내부에서 사용하기 위한 선언이다. 또한 현재 passwd 의 길이,count 값으로 LED 를 순서대로 불을 키는데 이것은 keypad 입력시 keypad 가 제대로 눌러졌나 확인하는데 사용된다.

Open_button 함수는 1 행 4 열에있는 A 버튼을 눌렀는지 알려주는 함수이다. A 를 눌러야만 키패드가 활성화 되어 입력을 받을수 있게 설정하였다. close_button 은 B 버튼을 누르면 doorlock 함수에의해 모터가 작동하여 문의 잠금장치가 잠기는 기능을 설정하기위한 함수이다.

Alarm_handler 함수는 정해진 시간이 지연된후에 사전에 정의해둔 process 가 실행되도록 하는 게 signal 객체의 알람함수인데 도어락을 활성화하고 시간이 오래 지나면 꺼지는것처럼 timeout 만큼 시간이 지나면 keypad 에 입력받았던 passwd 를 지우기위해 정의했다.

keypad 함수에서 global passwd 를 사용한 이유는 위에서와 같이 전역변수가 가변적인 변수이기 때문에 그 변수를 함수 내부에서 사용하기 위해 선언을 해주었다. Try,while 을 사용하여 keypad 가 return 을 하지 않으면 계속 반복되게 설정을 해주었다. A 버튼을 누르면 timeout 시간만큼 alarm 이 설정되고 readline 을 사용하여 keypad 숫자를 입력받을수 있게 하였고 입력받은 4 자리 passwd 를 key 로 값을 설정하고 passwd 를 초기화한후 key 를 반환한다.

이때 key 를 반환한 것은 doorlock.py 에서 자물쇠 개폐에 사용된다. 만약 B 가 눌리는 경우에는 key="closeDoor"를 반환하며 역시 doorlock.py 에서 자물쇠를 닫기위해 사용된다. KeyboardInterrupt 는 오늘의 번호 갱신을 위해 사용되며 "TimeoutException"을 반환하고 TimeoutException 일 때는 Keyboard timeout 이 발생하였을 때 None 을 반환하여 doorlock.py 에서 continue 되어 다시 keypad 를 새로 시작한다.

—motor

```
# 회전각 만큼의 dutyCycle 정도를 계산
def degreeProcessing(degree):
    # 최대 회전각 90 도, 최소 회전각 0 도
```

```

if degree > 90:
    degree = 180
elif degree < 0:
    degree = 0

return min_duty + (degree * (max_duty - min_duty) / 180.0)

```

[motor 의 세기 조절함수]

motor 설정에 대해서는 motor 에 연결된 드라이버의 연결 위치, 모터 전류의 주파수, 모터의 회전방향, 회전세기 순서로 참조번호를 값으로 저장했다. 모터 핀역시 Enable_pin 과 모터 동작제어핀을 설정을 해주었다. degreeProcessing 함수는 아까 정의한 회전세기를 원하는 각도만큼 조절하도록 계산해서 반환하는 함수이다. 여기서 도어락 잠금에 필요한 각도는 0 도에서 최대 90 도까지라서 상한선과 하한선을 제한을 하였다.

setMotorControl 함수는 입력값에 따라 모터의 회전을 제어하는 함수이다. 입력되는 각도와 행동변수에 따라 적절한 각도와 회전방향으로 회전하는 내용들이 있다. FORWARD 의 경우 doorlock 문이 열려있을 때 잠구는데 이용이 되고 BACKWARD 는 문이 닫혀있을 때 열기위해 이용이 된다.

setMotor 함수에서는 motor 제어함수의 입력값이 너무 많은데 그중에서 각도변수와 행동변수를 제외하면 사용자가 바꿀 필요가 없기 때문에 encapsulation 시켰다.

setPinConfig 함수는 motor.py 의 설정변수들을 이용하여 코드와 모터를 연동시키는 함수이다. GPIO 핀들을 참조번호가 담겨있는 설정변수들을 이용해 mapping 하였다. 반환값으로는 motor 제어에 필요한 thread 의 참조를 반환한다. lock 함수와 unlock 함수는 외부 코드에서 실제로 motor 를 제어할 때 호출하는 함수들로 정방향 회전과 역방향 회전을 수행시키는 함수이다. 이때 여러 번의 실험을 통해 setMotor 함수의 입력값들의 크기를 바꾸어 가면서 실제 도어락이 열리고 닫히는데 가장 적절하다고 생각되는 입력값들의 조합을 찾고 캡슐화했다. 이때 time library 를 사용하여 시간을 조절하였는데 그 이유는 회전시간후에 일정시간 정지하지 않으면 관성에 의해 회전정도가 계속 다르게 나오기 때문에 time.sleep 을 사용하여 관성을 없애주었다.

-exception.py

TimeoutException 이 python 에서 제공하는 내장클래스인데 그 클래스는 정해진 루틴이있고 프로그램을 중지시킬수있기 때문에 아무것도 하지 않도록 재정의 하였다.

-number_of_main.py

```

with open('main.txt') as file:
    passwd = file.readline()
    return passwd.strip()

```

[number_of_main.py 핵심코드]

메인번호가 적혀있는 main.txt 파일을 열고 readline 을 사용하여 main.txt 에 적혀있는 비밀번호를 읽어들이어 passwd 에 저장한다. 이때 passwd.strip() 을 사용하는데 strip()은 시작또는 끝에 공백이 없으면 문자열을 있는 그대로 반환하고 공백이 있을시에는 없애주는 기능을 하는데 main 번호를 저장시 공백이 생기는 경우를 대비하여 넣어주었다.

-number_of_today.py

```
symbols = "0123456789*#"
def todayNumber():
    data = ""
    for itr in range(kp.passwdLength):
        idx = random.randrange(len(symbols))
        data = data + symbols[idx]
    return data
```

[number_of_main.py 핵심코드]

Symbols 는 keypad 의 마지막열 즉 ABCD 를 뺀 나머지 부분이다. todayNumber 함수에서 data 를 아무것도없는 문자열로 설정하고 passwd 의 길이만큼 반복하여 data 를 무작위로 설정해준다. 이때 data 를 심볼의 길이 12 중에 한숫자를 random 하게 고르고 심볼에 index 를 넣어주어 아까 symbols 에 있는 값들중 하나를 선택한다.

-doorlock.py

```
day = 1 * 2 * 60 # 시 * 분 * 초
target_date=datetime.datetime(2023,5,26,9,59,0)

def check_date():
    global target_date
    current_date=datetime.datetime.now()

    if current_date >= target_date:
        while current_date >= target_date:
            target_date=target_date+datetime.timedelta(seconds=day)
```

```

    return (target_date-current_date).total_seconds()
else:
    return (target_date - current_date).total_seconds()

```

[오늘의번호 갱신시간 설정]

Target_date 는 current_date 인 현재 시간과 비교하여 오늘의번호 갱신을 하기위해 사용하였다. 이때 target_date 에 day 즉 오늘의번호 갱신주기를 더함으로 target_date 를 업데이트 하며 기존에 설정해둔 target_date 에 갱신주기를 더한것과 현재 시간을 비교한후 각각 동작을 수행했다. 앞에서 전역변수 target_date 를 함수 내에서 쓰기위해 global 로 정의해주었고 datetime library 를 사용하여 current_date 에 현재 시간을 저장해주었다. Target_date 는 current_date 보다 커야하는데 처음 동작시나 나중에 보겠지만 keypad 에서 너무 길게있을 때 target_date 를 적절하게 갱신하기위해 datetime.timedelta 를 사용하여 주기를 더해주었다. 그후 target_date 가 current_date 보다 커지면 두개의 차이를 반환하였고 만약 target_date 가 current_date 보다 크면 두개의 차이를 반환해주었으며 doorlock 함수에서 오늘의번호 갱신을 위해 사용된다.

doorlock 함수를 while True 문을 사용하여 doorlock 함수를 실행시키면 계속 반복되게 설정했다. 먼저 위에서 설명한 check_date 함수를 사용하여 갱신되어야하는 시간에서 현재시간을 빼준시간의 정수형으로 반환받고 +1 을 하여 flag 에 저장하였다. 이때 +1 을 하는 이유는 만약 current_date 와 target_date 가 0.3 초정도 차이난데 정수형으로 반환을 받으면 0 이 되며 이때 signal.alarm 이 제대로 동작하지 않기 때문에 1 을 더해주었다. 그후 key=kp.keypad 를 사용하여 key 를 반환받는데 아까 keypad 부분에서 key 를 passwd 길이만큼 잘 누르면 key=" 입력받은 passwd" 를 반환하고 만약 keypad 를 너무 많이 쳐놓으면 None 을 반환하고 만약 오늘의 번호를 갱신할때는 key=" TimeOutException " 을 반환하고 문을 닫기위해 B 를 누르면 key=" closeDoor" 를 반환한다고 설명했다. 이 반환받은 key 에의해 doorlock 의 여러 기능들을 설정해놓았다. 만약 반환받은 key 가 메인번호 혹은 오늘의번호인경우 motor.py 의 lock,unlock 함수를 사용하여 문이 열리고 닫힌다. 이때 개폐사이에 time.sleep(7)을 넣어준 이유는 밖에서 문을 열 때 문이 열린후 바로 닫히면 안되기 때문에 적절하게 7 초라는 시간을 넣어주었다. 만약 key 가 None 인 경우에는 continue 를 사용하여 반복문을 다시 수행함으로써 keypad 를 초기화했다.

Key=" TimeOutException" 를 반환받는경우를 살펴보겠다. 위의 값을 반환받기 위해서는 앞의 keypad 에서 KeyboardInterrupt 가 수행되어야하는데 이것은 doorlock.py 에서 alarm_handler 에 의해 발생이 된다. 만약 alarm_handler 가 발생하면 passwd[1]을 nt.todayNumber 함수를 사용하여 갱신하고 오늘의 번호를 출력해준다. 그리고 keypad.py 의 KeyboardInterrupt 예외사항에 가서 key=" TimeoutException" 을 반환하게 되며 continue 를 사용하여 반복문을 계속 수행하게 하였다. 만약 key가 None, " TimeOutException" ," closeDoor" 가 아니고 오늘의 번호,main 번호가 아닌경우에는 다시 반복문을 수행한다.

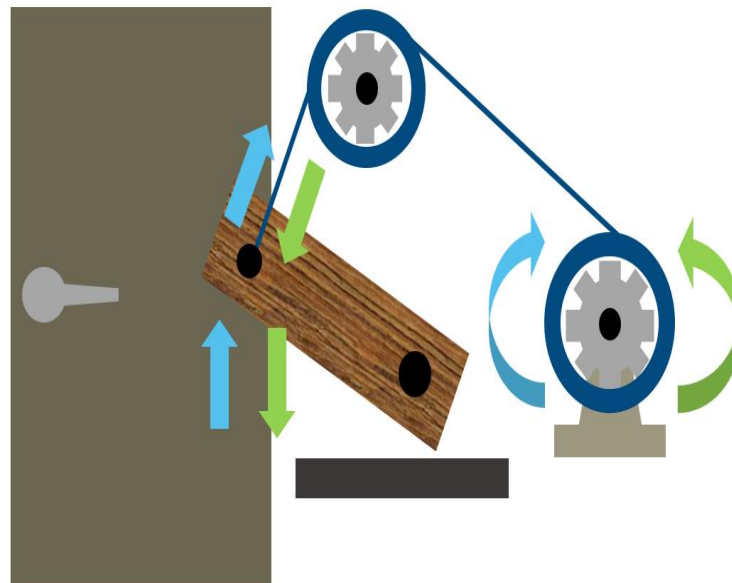
제 4장 제작 및 결과

4.1 제작 방법 및 결과 성능 시험

4.1.1 구조

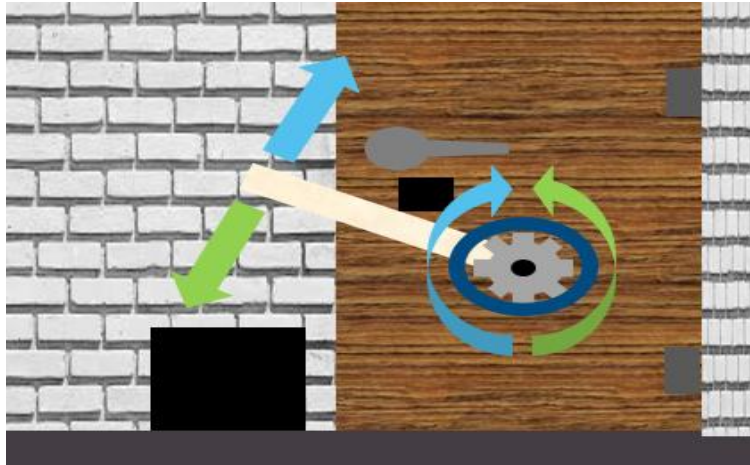
도어락 구조의 핵심은 잠금 장치이다. 그래서 잠금 장치의 설계 및 제작에 있어서 여러 구조를 설계했다. 여러 번 같은 동작을 수행하더라도 안정적으로 같은 동작을 원활하게 진행해야 하기에 각각을 분석했다.

1) 잠금장치



[그림 4-1 1안) 설계된 잠금 장치 구조]

[그림 4-1]의 경우 초반에 와이어를 이용하여 타워크레인과 비슷한 방식으로 잠금 막대를 들어올리는 방식으로 설계되었다. 잠금 막대 밑에 중력에 의한 불안정한 구동범위를 제한하는 가동범위제한 받침대가 설치 되어있다.



[그림 4-2 2안) 설계된 잠금 장치 구조]

[그림4-2]의 경우 TT모터자체에 잠금 막대를 설치하여 직접적인 모터의 동력을 받도록 설계되었다. 하지만 중력과 모터의 동력을 그대로 잠금 막대가 받는 문제가 있어 가동범위제한박스를 2개 설치하여 반복적으로 안정적인 동작을 수행할 수 있도록 했다. 아래 [표 4-1]에선 1안과 2안을 비교한다.

1안	2안
<p>와이어를 이용한 동력 전달</p> <p>TT모터 1개로는 부족하여 강한 동력이 필요</p> <p>영향을 주는 외력: 중력</p> <p>와이어의 꼬임의 문제 발생 가능</p>	<p>직접적인 동력 전달</p> <p>TT모터1개로 충분한 동력 전달가능</p> <p>영향을 주는 외력: 중력</p> <p>가동범위제한박스의 설치 필요</p>

[표 4-1 1안과 2안의 특징 비교]

1안의 경우 잠금 및 잠금 해제 동작을 반복적으로 수행할 때 와이어의 꼬임이나 중력의 문제로 간헐적인 오류가 발생할 가능성이 제기된다.

2안의 경우 TT모터를 1개만 사용하면서 가동범위제한박스의 설치만으로 반복적으로 안정적인 동작이 가능하기에 2안이 채택되어 최종적으로 제작되었다.



[그림 4-3 2안으로 완성된 잠금 장치 모습]

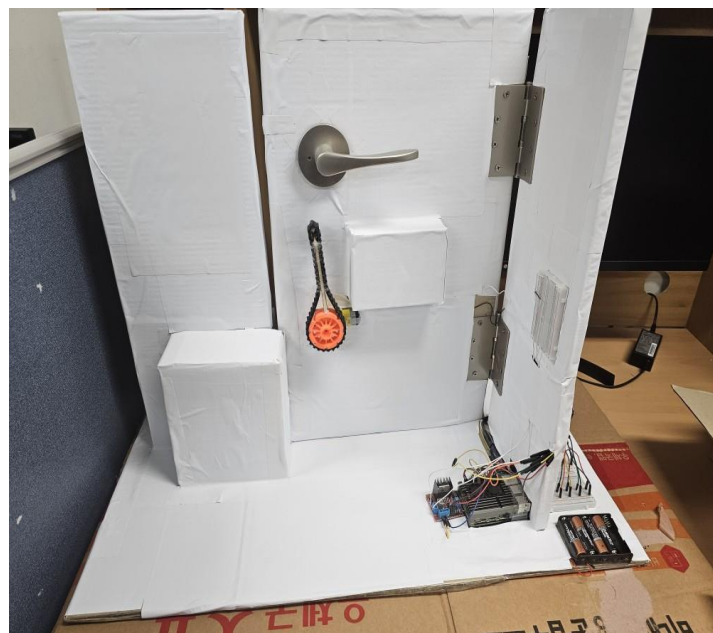
2) 외관 작업

제3장에서 상세한 설계 내용 및 제작 설명을 기술했다. 그 과정대로 제작하여 아래 [그림 4-4]가 되어 중간1차 때 모습이 된다.



[그림 4-4 중간1차 완성된 모습]

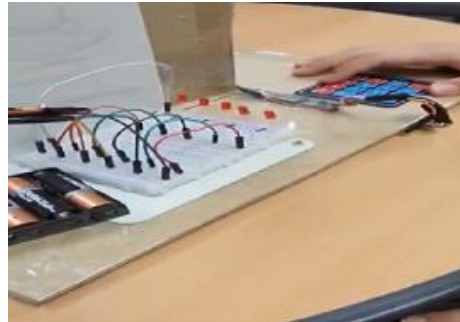
[그림 4-4]에 투박한 외관을 매끈히 하기 위해 하얀 시트지와 스프레이를 사용하여 외관작업을 진행한다. 부품 및 경첩이나 문고리에 마스킹 작업을 하여 외관작업 도중 발생할 수 있는 문제를 예방한다. 외관작업이 완료된 후 최종 시연 때의 모습이 나오게 된다.



[그림 4-5 최종 완성된 모습]

3) 사용자 편의성

도어락은 대부분의 사람이 사용하는 만큼 유저 친화적이어야 한다. 따라서 4*4키패드를 사용하여 한눈에 알 수 있게 구성했다. 그리고 사용자가 입력이 되었는지 확인하기 위한 방식으로 LED 점등을 통해 확인할 수 있도록 인디케이터를 구성했다. 이를 통해 중복입력이 발생하거나 잘못된 입력한 경우 바로 초기상태로 회귀할 수 있다.



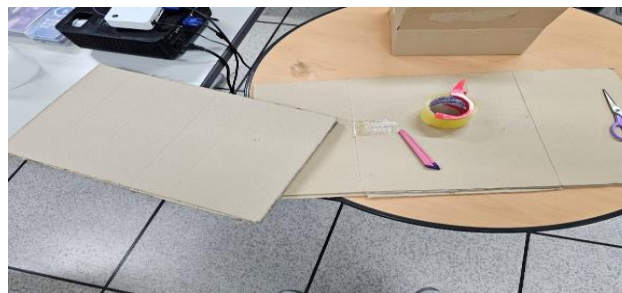
[그림 4-6 사용자의 편의성을 고려한 키패드 및 인디케이터의 모습]



[그림 4-7 키패드 및 인디케이터 정면모습]

다음내용에선 제작과정에 대한 것을 사진과 함께 확인 할 수 있다.

4.1.2 제작과정



[그림 4-8 상자 절단 및 결합한 모습]



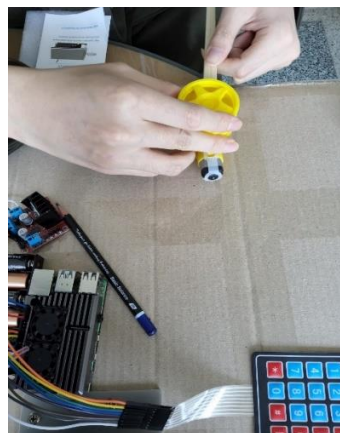
[그림 4-9 문을 완성한 모습]

먼저 필요한 벽과 바닥, 문을 제작하기 위해 [그림 4-8]과 같이 사용할 박스를 절단 후 결합한다. 그 다음 [그림 4-9]에서 나오듯이 문에 문고리를 설치하여 문의 형태를 갖추었다.



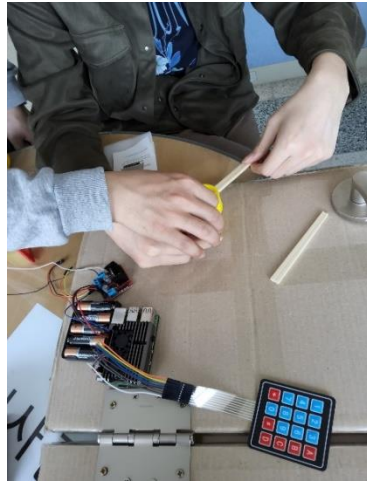
[그림 4-10 잠금 막대 제작]

[그림 4-10]에서는 잠금 장치에 속하는 잠금 막대를 제작하는 과정이다.

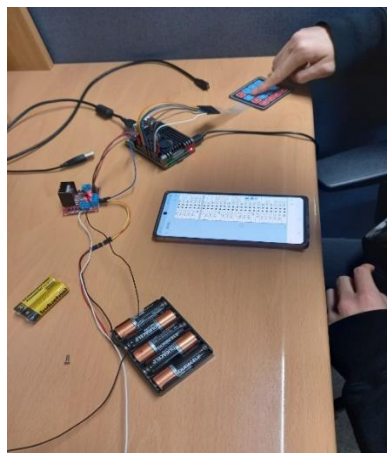


[그림 4-11 잠금 막대부를 모터에 연결]

[그림 4-11]에서 제작된 잠금 막대를 모터에 연결한 뒤 부품을 설치할 구역을 [그림 4-12] 럼 하였다.



[그림 4-12 부품 설치구역 설정]



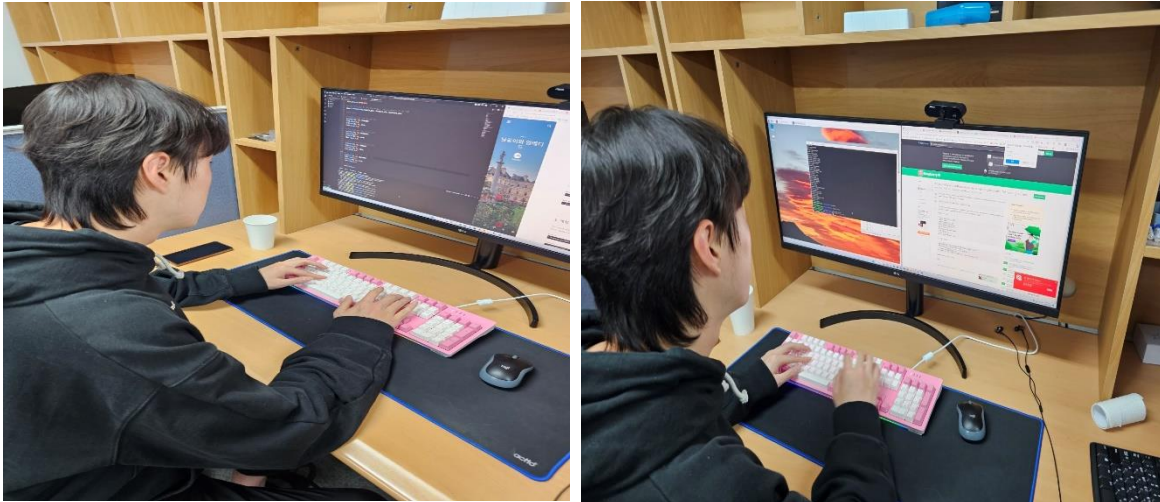
[그림 4-13 키패드, 모터 연결 후 테스트]

모터와 키패드를 연결하여 키패드의 입력이 알맞게 동작하는지 [그림 4-13]처럼 확인한다.



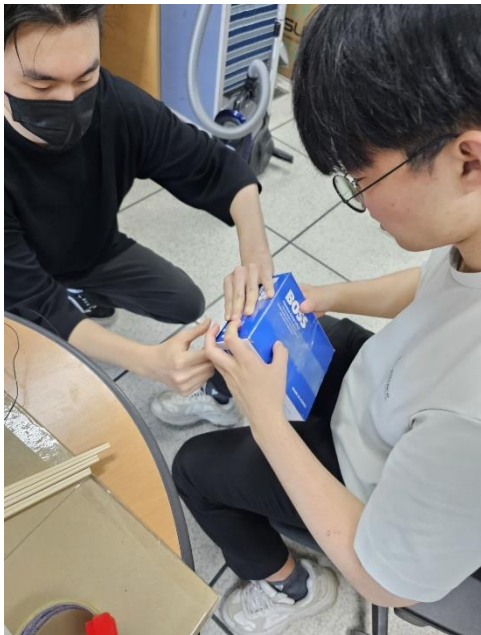
[그림 4-14 라즈베리파이(오른쪽)와 무선통신을 이용한 원격연결(왼쪽)]

[그림 4-14]와 같이 핵심인 라즈베리파이를 무선통신을 이용하여 원격으로 연결을 진행한다.

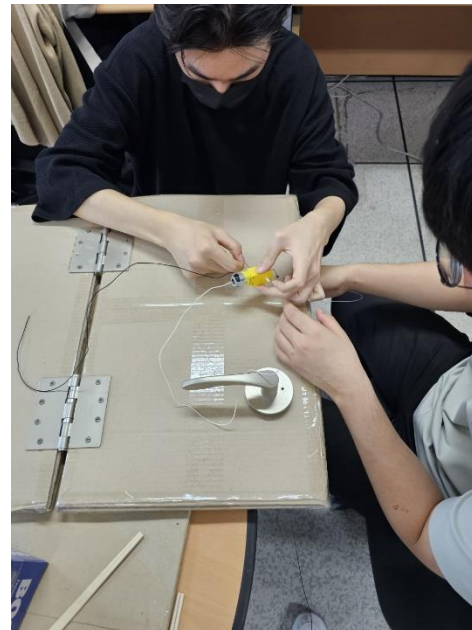


[그림 4-15,16 소프트웨어 설계]

무선 통신 연결이 된 후 [그림 4-15,16]처럼 도어락 동작관련 소프트웨어 설계를 진행한다.



[사진 4-17 구동범위제한 박스 제작]



[사진 4-18 와이어를 이용한 모터 설치]

[사진 4-17]과 같이 안정적으로 반복적인 문의 개폐를 위한 구동범위제한 박스를 제작한다.

와이어를 이용하여 문에 [사진 4-15]처럼 모터를 고정시킨다.



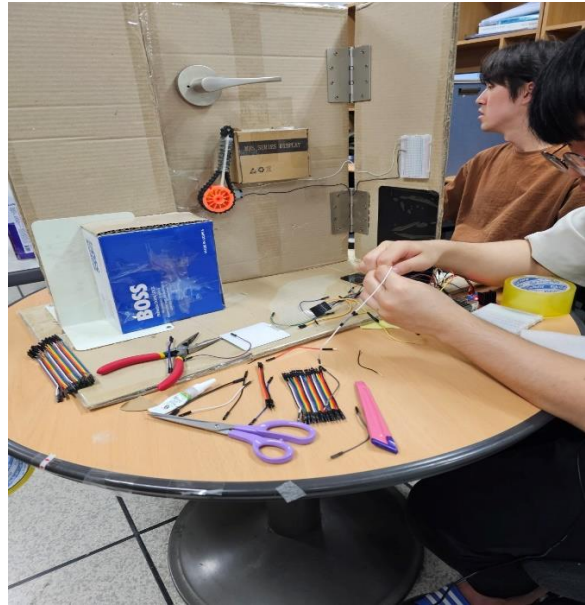
[그림 4-19,20 북엔드 설치 후 완성된 도어락 구조 프로토타입]

[그림 4-19,20]에서 볼 수 있듯이 북엔드를 통해 벽을 세우고 문에 경첩을 달아 벽에 고정하는 것을 통해 문 또한 바닥에 세울 수 있게 되어 기본적인 도어락 구조를 구현할 수 있게 되었다.



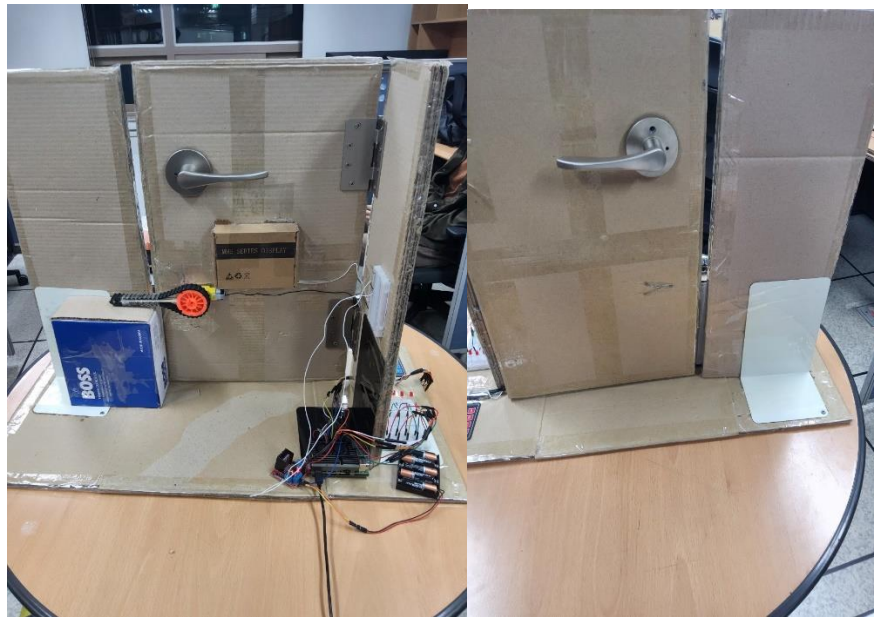
[그림 4-21,22 LED 인디케이터 회로 설계 및 동작확인]

사용자 편의성을 높이기 위해 인디케이터 회로를 설계 후 구현한 뒤 동작을 확인한다.



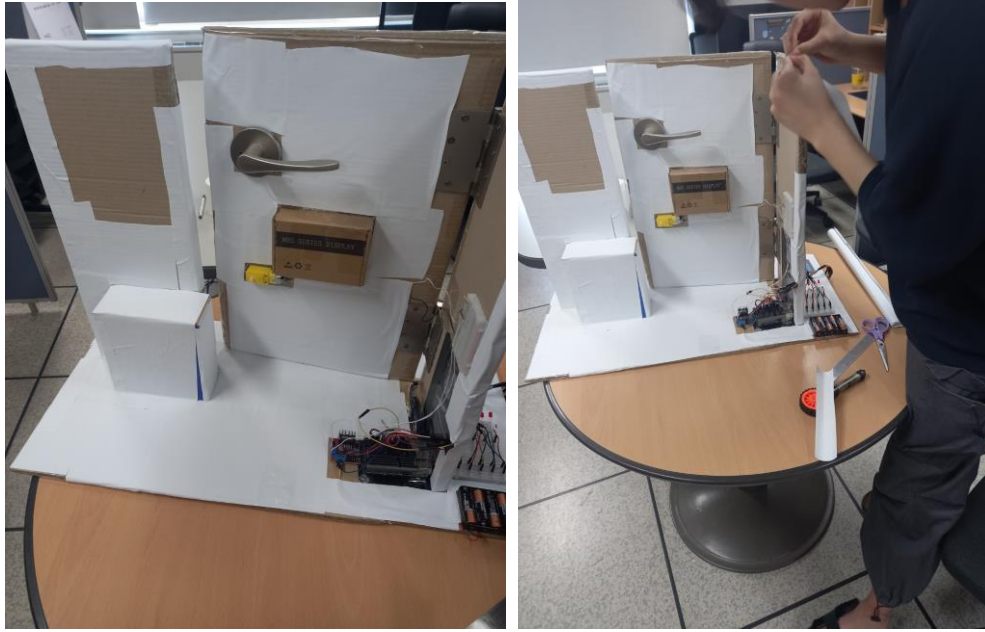
[그림 4-23,24 각 장치를 설치하는 모습]

이제 각 파츠가 완성되어 도어락 구조에 각각을 설치한다.



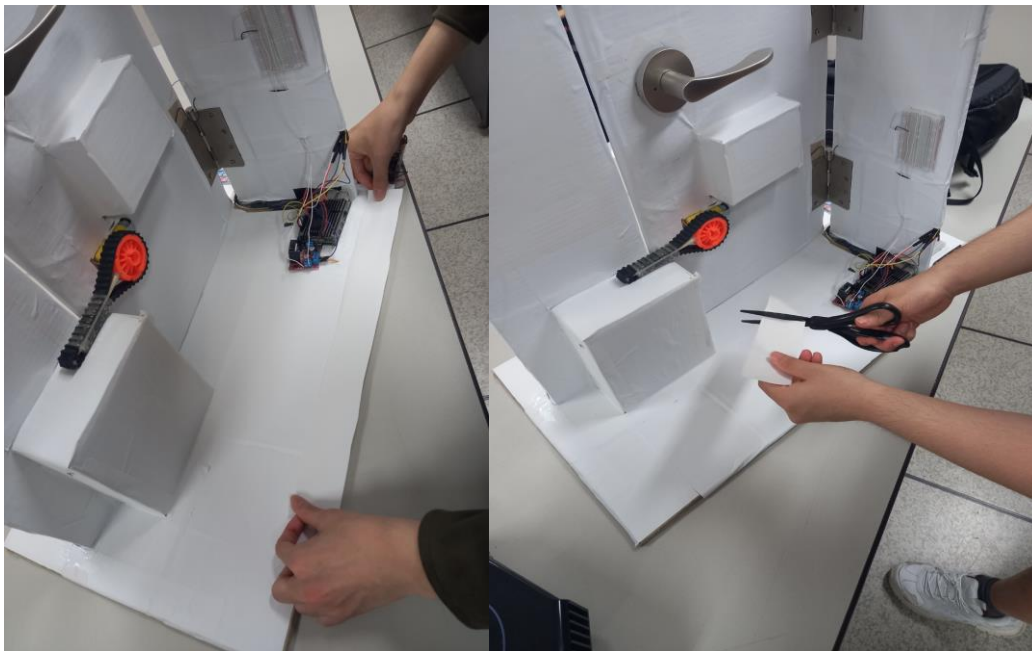
[그림 4-25,26 외관 작업 전 완성된 모습]

[그림 4-25,26]에 나오듯이 외관전에 박스의 투박함이 나타남을 알 수 있다. 추가적인 외관작업은 다음 과정에서 이루어진다.



[그림 4-27,28 외관작업을 진행하는 모습]

하얀 시트지와 스프레이로 박스의 투박함을 가리고 외관적인 완성도를 [그림 4-27,28]과 같이 더한다.



[그림 4-29,30 완성된 외관]

초기 박스의 투박함은 사라지고 깔끔한 외관이기에 외적인 완성도를 [그림 4-29,30]에서 확인할 수 있다.

제 5장 결론

5.1 요약

본 논문에서는 오늘의 번호 전자 도어락 설계 및 구현에 대한 프로젝트 내용을 충실히 설명하였다. 기존의 유일한 번호를 수동으로 설정하는 일반 전자 도어락에 비해, 유효기간이 존재하는 패스워드가 추가된 오늘의 번호 전자 도어락은, 위에서 설명한 기존의 도어락의 문제점을 해결하기 위해 제안됐을 뿐만 아니라, 추가된 기능의 적용이 소프트웨어 기술을 통해 자동화되어서 제공되었다.

외관상의 작품에 구성요소들은 크게 모터, 모터드라이버, 라즈베리파이, 키패드, LED, 연결선들을 제외하면, 대체가능한 부분들이다. 그렇기에 이 구성요소들의 사양과 호환되는 모듈들과 연결선의 길이를 확장할 수 있다면, 오늘의 번호 전자 도어락의 설치가능한 거주지의 구조 및 모양은 제한이 없다고 볼 수 있다.

오늘의 번호 전자 도어락의 소프트웨어적 메인 프로세스는 키패드 입력부와 모터 작동부를 철저히 모듈화시켜 상황과 순서에 맞게 반복적으로 제어하는 구조를 가지고 있다. 입력, 제어, 작동 매커니즘을 분리시켰기 때문에 예기치 않은 문제 발생시 원인파악 및 대응 그리 어렵지도 않다. 만약 프로세스가 갑작스럽게 종료된다 해도, 메인번호를 마이크로컨트롤러(라즈베리파이)의 프로세스상 메인메모리뿐만 아니라 SSD에 저장했고, 오늘의 번호 갱신이 상대적으로 시간이 아닌 날짜에 기반해서 이루어지기 때문에 물리적으로 파괴되지 않는한 메인 프로세스는 신뢰성을 보장한다.

따라서, 사용자는 추가된 기능을 사용할 때 별도의 작업이 필요없도록 구현되었기 때문에, 이용자의 편의성을 낮추지 않았다는 점에서 설계목적을 잘 따랐다고 할 수 있다.

소 감

이형주

4학년 1학기 캡스톤 디자인 과목을 진행하면서, 팀원과 브레인스토밍을 통해 아이디어 공유를 진행하면서 작품설계, 문제해결을 경험하면서 작품을 만들었습니다.

사회적 문제를 해결할 수 있는 졸업 작품을 만들고 나니, 계획적으로 활동한 것이 큰 도움이 되었습니다. 저번 3학년 2학기 때 같이 설계과제를 해본 팀원 그대로 다시 팀으로 결성하여 겨울방학 때부터 아이디어 회의에 집중했습니다. 덕분에 ‘오늘의 번호 도어락’이라는 훌륭한 아이디어가 도출되어 의미있는 작품을 급하게 설계하지 않으면서 최선을 다할 수 있었습니다. 첫 회의에서부터 꾸준히 일주일마다 아이디어 하나씩 내는 것을 규칙으로 하여 양질의 아이디어를 뽑아내고 계획을 세워 지체없이 과제를 진행해 나갔습니다. 팀에 SW개발, 통신전공의 팀원이 2명

이나 있어서 역할 분배를 통해 반도체를 전공한 저는 팀원들이 부족하거나 자신의 일에 집중할 수 있도록 문서와 관련된 전체 틀 및 회로도 작성을 도맡았으며 잠금 장치의 물리적인 매커니즘을 설계 및 제작했습니다. 카카오톡 오픈채팅방을 활용하여 비대면으로 공간의 제약없이 시간이 있을 때마다 회의를 진행했습니다. 그 속에서 좋은 아이디어나 재료가 나올 경우 즉각적으로 채팅을 이용해 메모하였습니다. 체계적으로 팀을 운영하면서 삼성노트를 활용하여 데이터를 정리했습니다.

여러 아이디어 중 오늘의 번호가 채택이 되었고 저번 설계과목과 다르게 하드웨어와 소프트웨어를 결합하는 과제를 목표로 삼았기에 알맞은 주제입니다. 저희 조는 유효기간이 있는 비밀번호를 사용한다는 아이디어를 구체화했으나 제여기와와의 통신관련 문제로 어떤 방식이 효율적인지 결정하기 어려웠습니다. 그래서 교수님께 조언을 구하여 괜찮은 아이디어임을 확인한 뒤 학부생 관점으로는 확인하기 어려운 부분도 알려주셔서 올바른 방향으로 과제를 이끌어 나갈 수 있었습니다.

제가 프로젝트에서 맡은 역할은 잠금 장치의 물리적인 매커니즘을 설계하고 인디케이터를 포함한 라즈베리파이의 회로도 및 문서 전체 틀 작성입니다. 평소 다루지 않았던 라즈베리파이의 핀 배치도를 보면서 회로도를 작성하는 것을 통해 간단하면서도 배울 것이 많다는 것을 느낄 수 있었습니다. 특히 핀 관련 연결케이블이 특이하여 배치에 대한 공부와 각 핀의 특성을 이해하고 연결하는 유익한 경험 또한 할 수 있었습니다.

그리고 발표 자료 등 문서의 틀 작성을 맡은 저는 소프트웨어, 통신 전공 팀원이 각자의 분배된 역할에 맞게 책임감을 가지고 집중할 수 있도록 나머지 부분을 도맡았기에 최고의 성과가 나왔다고 생각합니다.

도어락 구조를 제작할 때는 모두가 참여했으며 부품의 위치 등 의견 차이가 발생하였으나 모두가 동의하는 방향으로 조율하여 최적의 설계를 이루도록 노력하였습니다. 그 과정에서 갈등이나 이견이 있을 때 해결하는 경험을 미리 기르고 발전시킬 수 있었습니다.

끝으로 생각해보면 마지막 캡스톤디자인 과목을 진행하면서 사회적인 이슈에 대응할 수 있는 과제물을 만들었습니다. 이런 경험을 통해 실질적인 도움이 되는 것을 만들고 문제를 해결하고 극복하는 경험을 발판삼아 도약할 수 있다는 자신감이 생겼습니다. 겨울방학 때부터 지금까지 마지막 설계라고 생각하여 최선을 다해 체계적으로 달려왔기에 놓칠 수 있는 설계프로세스까지 챙겨 마무리할 수 있어서 뜻 깊습니다. 미래에 공정설계 등 직무를 수행할 때 전자공학을 전공한 사람으로서 이 경험이 미래에 설계 프로젝트에 적용할 만한 교훈을 남겨준다고 생각합니다. 능동적으로 스스로 참여하는 과목인 만큼 시작에서부터 끝을 직접 제 손으로 내는 만큼 공학 설계를 진행할 때의 진행과정을 이해하고 경험할 수 있었기에 유익했습니다.

전장군

3학년때 공학설계 때, 현재의 팀원들과 하드웨어적으로 애매한 주제로 프로젝트를 진행하고 나서 실세계의 문제를 해결할 수 있는 주제로 졸업작품을 구현해야겠다는 생각이 들었습니다.

그래서 부지런한 동료들과 함께 방학이 시작되나 마자 주기적인 화상회의를 통해 이번 프로젝트 주제에 대해 많은 고민을 했습니다.

처음에는 이번에도 딥러닝을 이용한 주제를 가지고 프로젝트를 진행할 뻔했었습니다. 저는 기계학습에 대해 관심이 많고, 재미있어했기 때문에 이전 프로젝트의 주제도 제가 제안했습니다. 그런데, 왜 기술에 자꾸 주제를 맞추려 할까라는 생각이 들었습니다. 저번에 지적받은 사항들의 원인은 사실 기술에 주제를 자꾸 맞추려하다가 발생한 사항들이 었기에, 이번에는 실세계에 필요한 일이 무엇일까 고민하다가 ‘오늘의 번호’ 라는 지극히 추상적인 개념을 팀원들에게 얘기를 했습니다. 그리고 나서 그것을 어떻게 구현할까 토론을 통해, 좀 더 실현가능하도록 설계계획이 세워졌습니다.

결과적으로 저희는 생각한 추상적인 아이디어를 물리적으로 구현했고, 중간중간에 많은 현실적인 문제들을 이겨내서, 많은 지식과 경험들을 얻었던 것 같습니다.

또한, 동료들도 정말 한명한명이 대체될 수 없는 아주 우수하고 열정적이고 예의있는 팀원들이라 생각합니다.

김영우

캡스톤 디자인은 제품을 구상하고 설계하는것부터 구현 그리고 테스트 및 동작시연까지 제대로 동작하는 제품을 만들어볼수있는 소중한 시간이었습니다.

캡스톤 디자인때 만들 제품을 구상할 때 가장 중요하게 생각했던 것은 주제정하기였습니다. 저희조는 3학년 2학기때 공학설계부터 같이했으며 방학중에도 아이디어를 공유하며 현재 사회에 어떤 문제점이 있는지에 대한 화두를 던지고 그것을 해결하기위한 방안을 같이 고민하였으며 그에따라 저희의 주제 “오늘의 번호를 사용한 도어락” 이라는 주제가 나오게 되었습니다.

주제를 정한후 저희는 현재 어떤 도어락이 사용되고있는지 확인하였으며 오늘의 번호를 사용하여 상용화된 제품이나 관련 논문이 없는 것을 확인하였습니다. 그후 저희가 생각한 제품을 어떤식으로 작동하게할지 구체화하는 시간을 가졌습니다. 이 과정에서 나와 다른사람의 아이디어가 조금 다를수 있고 한명한명씩 어떤식으로 구체화할지 그림을 그려가며 설명하고 조율해가는 과정이 필요하다는 것을 깨달았습니다.

하드웨어를 만들때는 모든 팀원이 같이 모여서 만들었습니다. 하드웨어를 만들기 전에는 쉽고 간단하게 만들수 있을것이라고 생각했는데 막상 만들기시작하니까 생각지도 못한 이유 때문에 진행이 막히거나 무엇을 해야할지 정확히 정하지않아 우왕좌왕하였습니다. 그래서 그 이후로는 만나기 전날에 이번에 만나서 어떤 것을 할지와 이것을 할 때 에러사항이 어떤 것이 있을까 고민하는 시간을 가졌습니다.

알고리즘 설계를 할 때는 사실 라즈베리파이를 처음만져보다보니까 GPIO핀이 무엇이며 어떤식으로 사용해야하는지 잘 모르고 라즈베리파이가 리눅스 기반이라 사용하기 어려웠는데 이 프로젝트를 진행하으로써 python실력도 많이 늘고 리눅스 기반으로 작업도 할수있을만큼 많이 성장했습니다.

학부생에서 하는 마지막 설계과목인 캡스톤 디자인을 할 때 전자과에서 지금까지 배워온 것들을 잘 활용하여 만들었으며 만약 문제가 발생하였을 때 그 문제를 해결할 수 있는 능력을 기를수있었습니다.

참고문헌

- [1] “주거침입 데이트폭력’ 형사입건 지난해 46% 급증 - [문화일보],” 2019,
- [2] “주거침입죄 및 야간주거침입죄, 강력범죄의 전조로 혐의 연루될 수 있어- 보건복지부 「장애인정책과, ” 2023, <https://www.ksilbo.co.kr/news/articleView.html?idxno=965171>
- [3] “옛 연인 집 무단침입한 30 대 前 남친 체포 - [세계일보],” 2021, <https://n.news.naver.com/mnews/article/022/0003650481?sid=102>
- [4] “도어록,
” <https://terms.naver.com/entry.naver?docId=2418506&cid=51399&categoryId=51399>
- [5] “L298N 모터 드라이버,” 네이버 블로그, 2023 년 05 월 29 일 접속, https://blog.naver.com/mapes_khkim/222502077852

<부록>

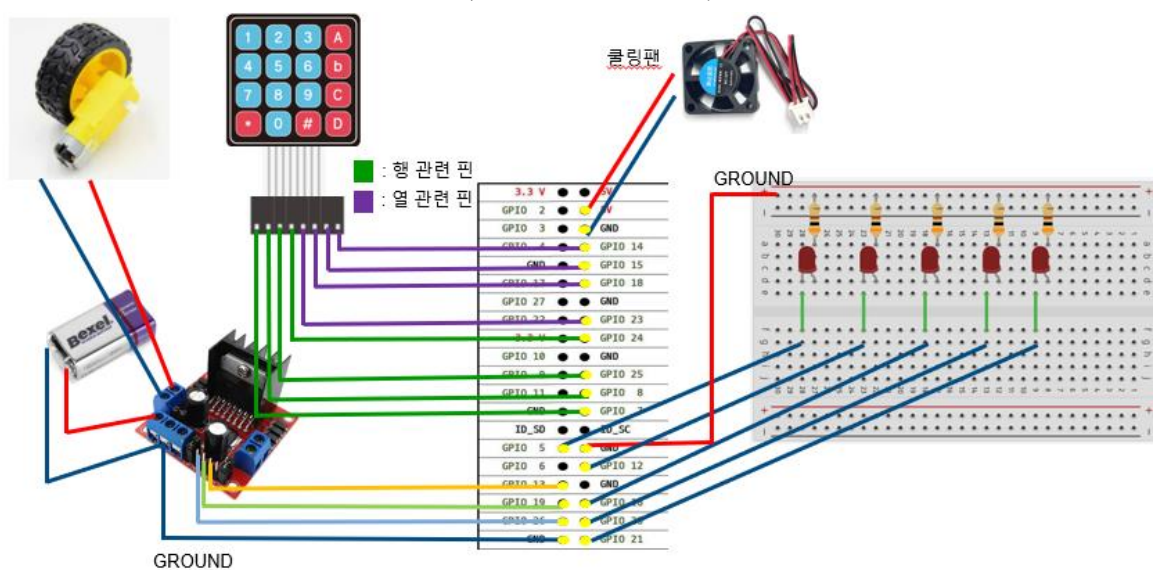
부록 1-1: 부품 목록

전원부	리튬이온 배터리	전압	용량	규격	중량
	AA	1.5V	1800mAh	14.5*50.5 mm	23 g
	건전지 홀더	직렬연결 6v		63*58*15mm	
입력부	4*4키패드 HAM2913	정격 전압	리바운드시간	규격	절연 저항
		35VDC	5ms 미만	70*158 mm	100 ohm

출력부	LED	+극 길이	-극 길이	Color	
		17.5mm	16mm	RED	
	100 ohm 저항	정격 전압	저항값	최대 사용 전압	최대 과부하 전압
		1/4W(0.25W)	100 ohm	150V	300V
	브레드보드	홀 수	사용 가능 전선	최대 전압	최대 전류
		400 홀	20 ~ 29AWG	300V	3-5A
동작부	TT 모터	작동전압	최대 토크	규격	공회전 속도
		DC 3~12V	800gf cm min(3V 연결)	22*36*69mm	1:48 (3V 연결시)
	L298N 모터 드라이버	동작전압	동작전류	규격	모터허용전류
		DC 5V	0~36mA	43*43*27mm	2A (MAX)
제어부	라즈베리파이 4	전원공급	메모리	프로세서	동작 온도
		5V DC via USB-C	8GB	Broadcom BCM2711	0~50 °C

[표 부록 1 부품 목록]

부록 1-2: 회로도



부록 1-3: 코드

```
#Doorlock.py

import RPi.GPIO as GPIO
import signal
from exception import TimeOutException
import moter as mt
import keypad as kp
import number_of_main as nm
import number_of_today as nt
import warnings

arnings.filterwarnings('ignore')

day = 60 # 60 * 60 * 24

passwd = ['', '']
passwd[0] = nm.mainNumber()
passwd[1] = nt.todayNumber()
print('오늘의 번호 갱신!\n', passwd, '\n')

def alarm_handler(signum, frame):
    passwd[1] = nt.todayNumber()
    print('오늘의 번호 갱신!\n', passwd)
    raise KeyboardInterrupt()
```

```

def doorlock():
    while True:
        try:
            signal.signal(signal.SIGALRM, alarm_handler)
            signal.alarm(day)
            key = kp.keypad()
            print("key 는 ", key, '\n')
            if key in passwd:
                mt.unlock()
                mt.lock()
                print("Open")
            elif key == None:
                continue
            elif key == "TimeoutException":
                raise TimeoutException()
            elif key == "closeDoor":
                mt.lock()
                print("Close")
        except TimeoutException as e:
            continue
        except KeyboardInterrupt:
            return;

doorlock()

```

```
#motor.py
```

```
import RPi.GPIO as GPIO # 라즈베리파이의 GPIO 관련 라이브러리
```

```
import time # 시간 관련 라이브러리
```

```
import warnings
```

```
warnings.filterwarnings('ignore')
```

```
#### 모터 설정 ####
```

```
# 모터 번호(추가 가능)
```

```
CH = 0
```

```
# 모터 PWM 주파수
```

```
freq = 50
```

```
# 모터 동작 종류
```

```
STOP = 0
```

```
FORWARD = 1
```

```
BACKWORD = 2
```

```
# 모터 dutyCycle
```

```
min_duty = 1
```

```
max_duty = 70
```

```

##### GPIO pin 설정 #####

# 모터 PWM enable 핀
Enable_pin = 26 #37 pin

# 모터 동작 제어 핀
forward_pin = 19 #37 pin
backward_pin = 13 #35 pin

##### 사용자 정의 함수들 #####

# 회전각 만큼의 dutyCycle 정도를 계산
def degreeProcessing(degree):
    # 최대 회전각 90 도, 최소 회전각 0 도
    if degree > 90:
        degree = 180
    elif degree < 0:
        degree = 0

    return min_duty + (degree * (max_duty - min_duty) / 180.0)

# 모터 제어 함수

```

```

def setMotorControl(pwm, forward_pin, backward_pin, degree,
action):
    #모터 회전 제어
    duty = degreeProcessing(degree)
    pwm.ChangeDutyCycle(duty)

    # 정방향 회전
    if action == FORWARD:
        GPIO.output(forward_pin, GPIO.HIGH)
        GPIO.output(backward_pin, GPIO.LOW)

    # 역방향 회전
    elif action == BACKWORD:
        GPIO.output(forward_pin, GPIO.LOW)
        GPIO.output(backward_pin, GPIO.HIGH)

    # 회전을 멈춤
    elif action == STOP:
        GPIO.output(forward_pin, GPIO.LOW)
        GPIO.output(backward_pin, GPIO.LOW)

# 모터 제어 설정 함수
def setMotor(ch, degree, action):
    # 모터 번호에 따라 적절한 핀들을 선택해서 제어함수 호출

```



```

    if ch == CH:
        setMotorControl(pwm, forward_pin, backward_pin,
            degree, action)

# 핀 설정 초기화
def setPinConfig(EN, INA, INB):
    GPIO.setup(EN, GPIO.OUT)
    GPIO.setup(INA, GPIO.OUT)
    GPIO.setup(INB, GPIO.OUT)

    pwm = GPIO.PWM(EN, freq)
    pwm.start(0)

    return pwm

def lock():
    setMotor(CH, 90, FORWARD)
    time.sleep(0.3)
    setMotor(CH, 90, STOP)
    time.sleep(1.5)

def unlock():
    # 역방향으로 90도 회전 후 잠깐 대기
    setMotor(CH, 90, BACKWORD)
    time.sleep(3)

```

```

setMotor(CH, 90, STOP)
time.sleep(1.5)

#### 프로그램 실행 ####

# GPIO 핀들의 번호를 지정하는 모드 설정
GPIO.setmode(GPIO.BCM)

#모터와 핀 바인딩 후, 모터 PWM enable 스레드(thread)의
참조를 반환
pwm = setPinConfig(Enable_pin, forward_pin, backward_pin)

if __name__ == "__main__":
    #제어 시작

    # 정방향으로 90 도 회전 후 잠깐 대기
    setMotor(CH, 90, FORWARD)
    time.sleep(0.3)
    setMotor(CH, 90, STOP)
    time.sleep(1.5)

    # 역방향으로 90 도 회전 후 잠깐 대기

```

```
setMotor(CH, 90, BACKWORD)
```

```
time.sleep(3)
```

```
setMotor(CH, 90, STOP)
```

```
time.sleep(1.5)
```

```
# 정방향으로 90 도 회전 후 잠깐 대기
```

```
setMotor(CH, 90, FORWARD)
```

```
time.sleep(0.3)
```

```
setMotor(CH, 90, STOP)
```

```
time.sleep(1.5)
```

```
# 역방향으로 90 도 회전 후 잠깐 대기
```

```
setMotor(CH, 90, BACKWORD)
```

```
time.sleep(3)
```

```
setMotor(CH, 90, STOP)
```

```
time.sleep(1.5)
```

```
# 정방향으로 90 도 회전 후 잠깐 대기
```

```
setMotor(CH, 90, FORWARD)
```

```
time.sleep(0.3)
```

```
setMotor(CH, 90, STOP)
```

```
time.sleep(1.5)
```

```
# 역방향으로 90 도 회전 후 잠깐 대기
```

```
setMotor(CH, 90, BACKWORD)
```

```
time.sleep(3)
setMotor(CH, 90, STOP)
time.sleep(1.5)

# 정방향으로 90 도 회전 후 잠깐 대기
setMotor(CH, 90, FORWARD)
time.sleep(0.3)
setMotor(CH, 90, STOP)
time.sleep(1.5)

# 역방향으로 90 도 회전 후 잠깐 대기
setMotor(CH, 90, BACKWORD)
time.sleep(3)
setMotor(CH, 90, STOP)
time.sleep(1.5)

# 종료
GPIO.cleanup()

print("성공적으로 프로그램이 수행됨")
```

```
#Keypad.py
```

```
import RPi.GPIO as GPIO
```

```
import time
```

```
import signal
```

```
from exception import TimeOutException
```

```
import warnings
```

```
warnings.filterwarnings('ignore')
```

```
R1 = 7
```

```
R2 = 8
```

```
R3 = 25
```

```
R4 = 24
```

```
C1 = 23
```

```
C2 = 18
```

```
C3 = 15
```

```
C4 = 14
```

```
L = [5, 12, 16, 20, 21]
```

```
passwd = []
```

```
passwdLength = 4
```

```
passwdCount = 1
```

```
timeout = 20
```

```

# Initialize the GPIO pins

#GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

GPIO.setup(R1, GPIO.OUT)
GPIO.setup(R2, GPIO.OUT)
GPIO.setup(R3, GPIO.OUT)
GPIO.setup(R4, GPIO.OUT)

GPIO.setup(C1, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(C2, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(C3, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(C4, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)

GPIO.setup(L[0], GPIO.OUT)
GPIO.setup(L[1], GPIO.OUT)
GPIO.setup(L[2], GPIO.OUT)
GPIO.setup(L[3], GPIO.OUT)
GPIO.setup(L[4], GPIO.OUT)

def readLine(line, characters, count):
    global passwd
    GPIO.output(line, GPIO.HIGH)
    if GPIO.input(C1) == 1:

```

```

    print(characters[0], '\n')
    passwd.append(characters[0])
    GPIO.output(L[count], GPIO.HIGH)
    count = count + 1
    if GPIO.input(C2) == 1:
        print(characters[1], '\n')
        passwd.append(characters[1])
        GPIO.output(L[count], GPIO.HIGH)
        count = count + 1
    if GPIO.input(C3) == 1:
        print(characters[2], '\n')
        passwd.append(characters[2])
        GPIO.output(L[count], GPIO.HIGH)
        count = count + 1
    if GPIO.input(C4) == 1:
        print(characters[3], '\n')
        passwd.append(characters[3])
        GPIO.output(L[count], GPIO.HIGH)
        count = count + 1
    GPIO.output(line, GPIO.LOW)
    return count

def open_button():
    GPIO.output(R1, GPIO.HIGH)

```

```

    if GPIO.input(C4) == 1:
        GPIO.output(L[0], GPIO.HIGH)
        GPIO.output(R1, GPIO.LOW)
        return True
    GPIO.output(R1, GPIO.LOW)
    return False

def close_button():
    GPIO.output(R2, GPIO.HIGH)
    if GPIO.input(C4) == 1:
        GPIO.output(L, GPIO.HIGH)
        GPIO.output(R2, GPIO.LOW)
        return True
    GPIO.output(R2, GPIO.LOW)
    return False

def alarm_handler(signum, frame):
    print("\nTime Out!")
    raise TimeoutException()

def keypad(count=passwdCount):
    global passwd
    try:
        while True:

```



```

    if open_button():
        print('입장 버튼 누름')
        time.sleep(0.25)
        signal.signal(signal.SIGALRM, alarm_handler)
        signal.alarm(timeout)
        while count <= passwdLength:
            count = readLine(R1, ["1","2","3","A"], count)
            count = readLine(R2, ["4","5","6","B"], count)
            count = readLine(R3, ["7","8","9","C"], count)
            count = readLine(R4, ["*","0","#","D"], count)
            time.sleep(0.25)
        GPIO.output(L, GPIO.LOW)
        key = ".join(s for s in passwd)
        passwd = []
        signal.alarm(0)
        return key
    elif close_button():
        print('퇴장 버튼 누름')
        GPIO.output(L, GPIO.LOW)
        time.sleep(0.25)
        key = "closeDoor"
        return key
except KeyboardInterrupt:
    GPIO.output(L, GPIO.LOW)
    passwd = []

```

```

        print("System Interrupt!")
        return "TimeOutException"
    except TimeOutException as e:
        GPIO.output(L, GPIO.LOW)
        passwd = []
        return None

if __name__ == "__main__":
    key = keypad()
    if key == None:
        print('None')

```

```

#exception.py

class TimeOutException(Exception):
    pass

```

```

#number_of_main

def mainNumber():
    with open('main.txt') as file:
        passwd = file.readline()
        return passwd.strip()

```

```

#Number_of_today

```

```
import random
import keypad as kp

symbols = "0123456789*#"

def todayNumber():
    data = ""
    for itr in range(kp.passwdLength):
        idx = random.randrange(len(symbols))
        data = data + symbols[idx]
    return data
```

```
#main.txt
1234
```

부록 2: 현실적 제한 요소

현실적인 제한 요소	
제한 요소	내용
원가 및 경제성	<ul style="list-style-type: none"> · 기존 도어락에 기능을 추가하는 방향으로 하여 최대한 원가를 낮추고 어디서나 설치가 가능 <ul style="list-style-type: none"> - 설치위치 제약이 적으며, 단순 장치만 설치하거나 기능만 더하는 것으로 오늘의 번호의 사용이 가능해지므로 기존의 도어락과 가격적인 큰 차이 없이 뛰어난 보안성을 가지도록 하여 경쟁력을 갖추도록 함 · 사용자의 편의성 고려하며 가격 대비 성능도 충족한 부품선택 <ul style="list-style-type: none"> - 키패드의 경우 누르는 사용자를 고려하여 선택하였으며 가격도 합리적 - 인디케이터는 간단한 회로이기에 매우 낮은 원가로 제작이 가능하고 외관작업이 이루어진다면 시장에서 값 싼 가격으로 경쟁 가능 - 모터 또한 라즈베리파이, 아두이노 사용자들이 범용적으로 사용하는 것으로 선택하여 쉽게 구할 수 있기에 교체 용이성을 두어 저렴한 교체비용 확보 가능 · 실험을 통하여 부품이 적합한지 판단 <ul style="list-style-type: none"> - 모터 동작 실험 - 키패드 입력 실험 · 도어락 구조 실현을 위해 박스로 제작하였으나 실제 실제품이 된다면 모터부와 기능만 추가하는 방식으로 저렴한 설치비용
환경	<ul style="list-style-type: none"> · 도어락 환경이 실내에 대부분 구축되어 있는 것을 고려한 설계 <ul style="list-style-type: none"> - 가동범위제한 박스 내부에 회로 탑재가 가능하며 이를 통해 라즈베리파이 등 전선, 드라이버 내장화 가능 - 추가적인 박스를 설치하는 것으로 두꺼비집처럼 내부를 확인할 수 있고 관련 부품 보호목적으로 사용가능
내구 및 신뢰성	<ul style="list-style-type: none"> · 모터의 힘을 잠금 막대가 견딜 수 있도록 체인을 부착하여 내구성 확보 · 문에 사용한 경첩의 경우 나사 5 개를 활용하여 4 겹의 박스로 구성된 벽에 단단히 고정시켜 빠지지 않게 제작 · 가동범위제한 박스의 설치로 중력에 영향을 받는 잠금 막대가 반복적인 문 개폐동작에도 안정적으로 동작함

안전성	<ul style="list-style-type: none"> · 보안에 대한 우려 <ul style="list-style-type: none"> - 문을 물리적으로 막는 구조라 실제 상품화하게 되면 강한 힘에도 견딤 - 유효기간이 있는 비밀번호를 사용하는 방식이라 메인 번호의 유출만 없다면 뛰어난 보안성을 가짐
제조성	<ul style="list-style-type: none"> · 간단한 구조를 통해 쉽게 제작 가능 <ul style="list-style-type: none"> - 조립식으로 제작 가능한 설계 고려 - 상품화 시 각 부품 별 분업생산 고려
무게 및 디자인	<ul style="list-style-type: none"> · 익숙한 현관문 디자인으로 구조를 제작하여 사용자들의 불편을 줄이고 빠른 적응이 가능
사회성	<ul style="list-style-type: none"> · 증가하는 무단침입 범죄를 방지하는 오늘의 번호 도어락 <ul style="list-style-type: none"> - 무단침입 범죄 건수 매년 증가 - 도어락 등 주거지에 대한 보안성 관심 증가
편리성	<ul style="list-style-type: none"> · 쉬운 비밀번호 입력방식 <ul style="list-style-type: none"> - 직관적인 4*4 키패드를 통해 한눈에 들어오고 기존의 키패드와 비슷한 구조 - 인디케이터를 통해 입력여부를 확인할 수 있어서 빠르게 중복입력 발생 시 빠르게 초기상태로 회귀 가능 · 간편한 잠금 방식 <ul style="list-style-type: none"> - 외출 시 B 버튼을 누르는 동작으로 쉽게 문이 잠김

[표 부록 2 제한요소 분석]

부록 3: 종합설계 공학문제수준

문제의 속성	공학문제수준설명	만족여부 (○, ×)
	심화된 공학문제가 속성 1(지식의 깊이)을 만족하고, 속성 2 ~ 속성 8 중 일부 또는 전부를 만족해야 한다.	
속성 1 (지식의 깊이)	최신 정보와 관련 연구 결과를 활용하고 있다.	○
속성 2 (상충되는 요건의 범위)	상충될 수 있는 기술적 또는 공학적 이슈를 다루고 있다.	X
속성 3 (분석의 깊이)	해답이 명확하지 않은 문제를 해결하기 위해 깊이 있는 사고와 분석과정을 다루고 있다.	X
속성 4 (생소한 주제)	자주 접하지 않는 공학문제를 다루고 있다.	○
속성 5 (문제의 범위)	전공분야의 일반적인 실무 영역을 벗어난 범위를 다루고 있다.	○
속성 6 (이해당사자의 요구 수준 및 범위)	다양한 이해당사자들의 요구사항들을 고려하고 있다.	○
속성 7 (상호의존성)	상호 의존적인 여러 세부문제들이 결합된 종합적인 문제로 구성되어 있다.	X
속성 8 (다양한 영향 고려)	다양한 분야에 미치는 영향을 고려하고 있다.	○
만족(○) 으로 표시된 속성에 대하여 어떻게 해당 속성을 만족하는지 설명		
<p>속성 1 (지식의 깊이) : 기존 시중에 나온 도어락들을 찾아보는 과정을 통해 다른 도어락들의 종류 및 기술들을 고려해서 설계하였다.</p> <p>속성 4 (생소한 주제) : 보통의 공학문제는 수학적 성능을 높이는데 중점을 두고있는데 저희는 성능 개발에 중점을 두어 기존의 제품을 확장하였습니다.</p> <p>속성 5 (문제의 범위) : 학교에서 배우지않은 라즈베리파이라는 OS 에 관해 디렉토리 구조나 GPIO 등의 활용을 스스로 찾아서 구현하였다.\</p> <p>속성 6 (이해당사자의 요구 수준 및 범위) : 보안의 취약성을 위해 이 제품을 개발했는데, 판매자의 경우에는 기존의 메인번호만 사용하는 도어록을 판매하는 입장에서 기존의 기능을 침해하지않으면서 업데이트 할수있다.</p> <p>속성 8 (다양한 영향 고려) : 비밀번호를 알아내 다른사람의 집에 무단침입을 하는 것을 막을수있고 자기집번호 유출에대한 불안감이 낮아진다.</p>		