Electronic Doorlock System

- Implementation -

Team 4, < JongJungJungKang >

Minku Knag,

Sun Jung Ahn,

Jong Chan Lee,

Jung Han Choi

Contents

- Implementation Goal
- Implementation Environment
- Implementation Point
- Implementation
- Supplementation

Implementation Goal

- Electronic Doorlock System의 실제 구현
- SRS, SRA, SA 등 작성한 문서를 바탕으로 구현한다.

Implementation Environment

Language : C Language

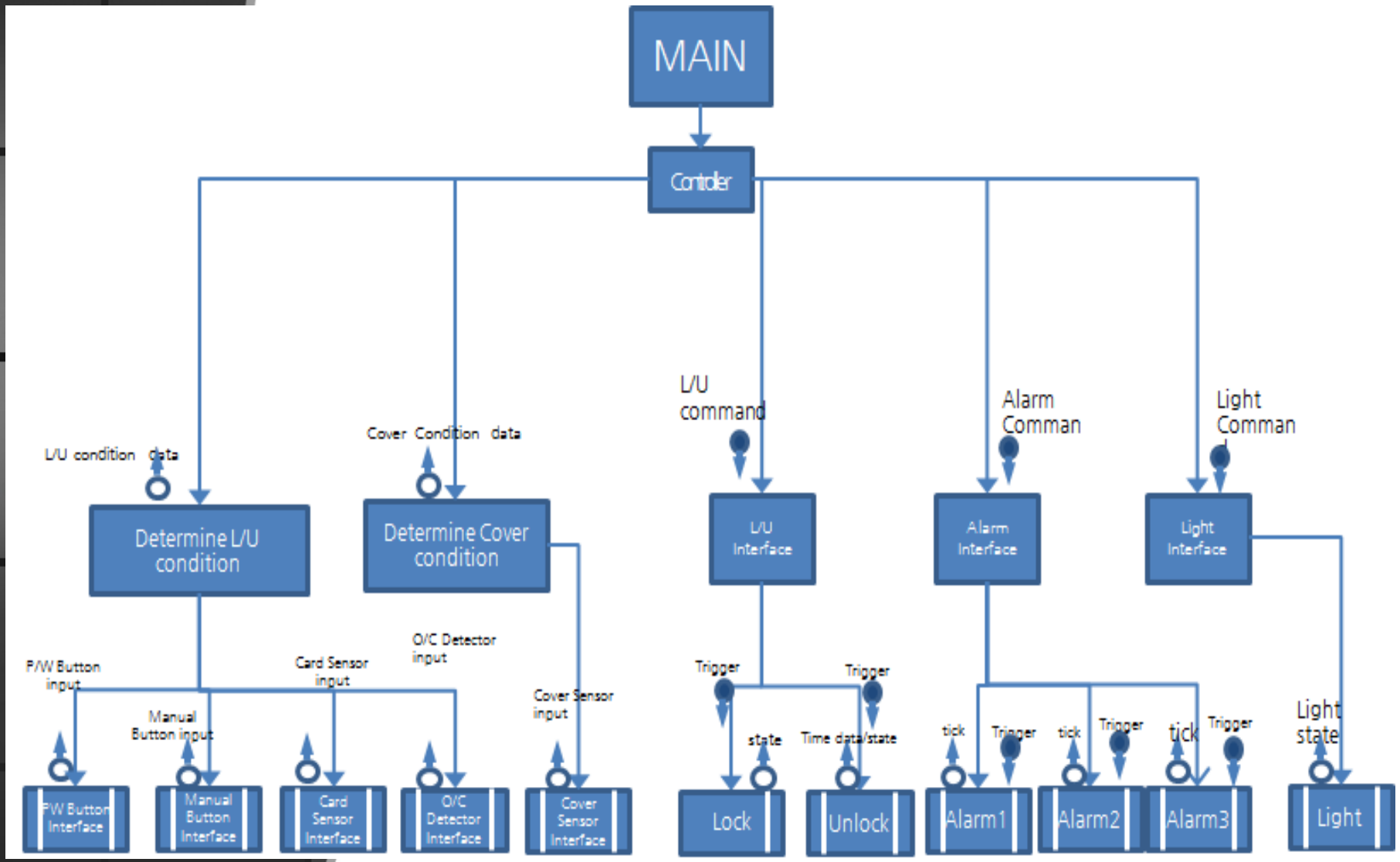
IDE : Eclipse for C/C++

Complier : GCC (in MinGW)

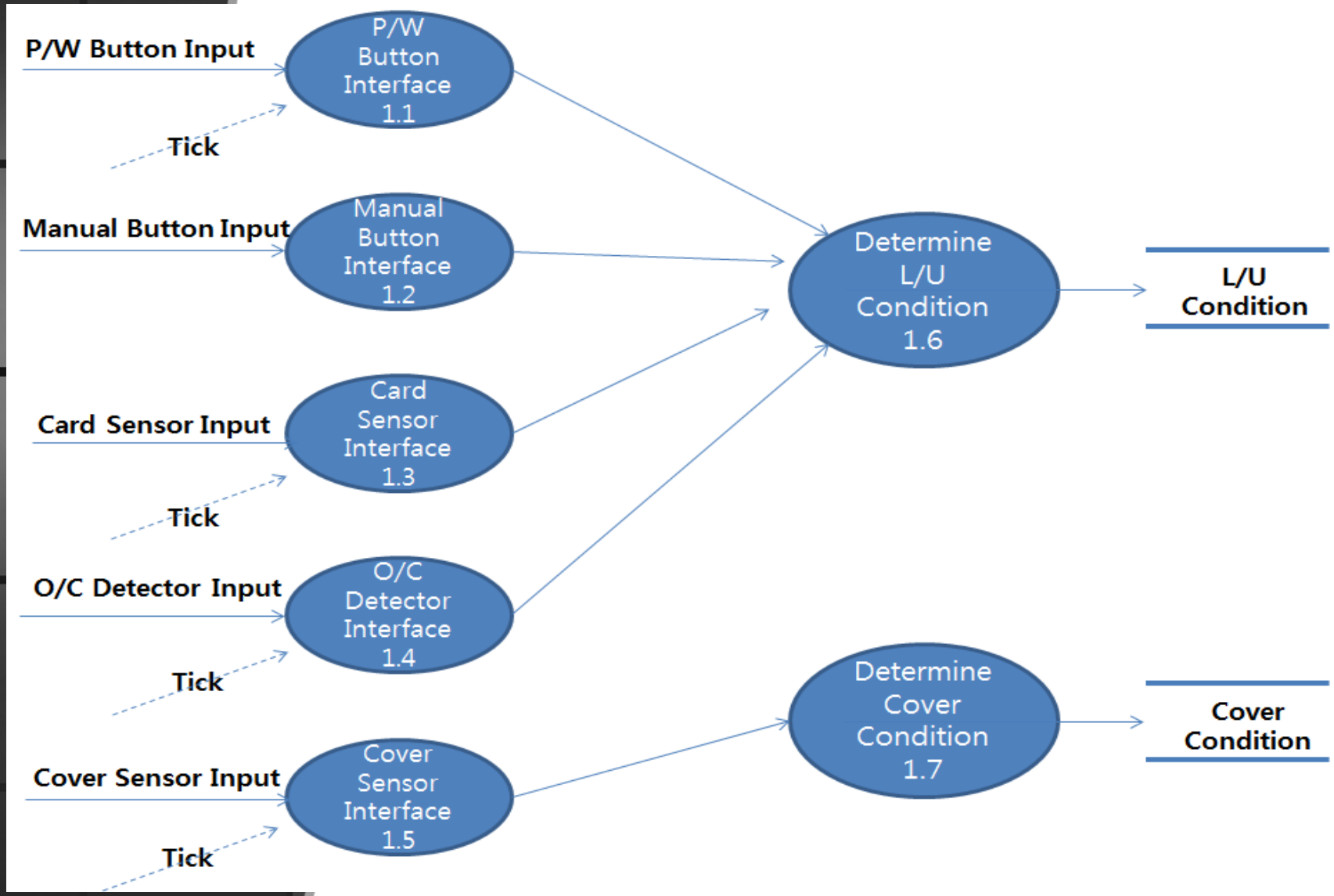
Implementation Point

- 하드웨어 고려 X.
- UI는 최대한 고려하지 않음.
  (UT를 예정중. UT시는 Text가 편할 것이라고 판단.최종 제작 때 Virtualization 적용예정)
- 제작한 문서를 최대한 반영.
- 전역변수 등 혼란을 야기하는 코딩 스킬 사용 X

Implementation



Implementation



Implementation

```
/* 1. 변수 설정 및 초기화 */

unsigned short pw = 0; // 패스워드는 입력받아 임시로 저장하는 변수
char card = ' '; // 카드키를 입력받아 임시로 저장하는 변수

password pw_saved = -1; // 패스워드가 저장되어 있는 변수. 0000 0000 0000 0000 16비트 자료형인 short를 4비트 단위로 끊어서 한자리씩 저장
char card_saved = ' '; // 카드키가 저장되어 있는 변수. a-z까지 알파벳 1자리를 저장하고 있다.

bool Mbutton = 0; // 수동잠금장치의 상태를 저장하는 변수. 1 <-> 0으로 상태가 전환된다.
bool tmp_Mb = 0; // 수동잠금장치의 전 상태를 저장하는 변수.
bool OC = 0; // 문의 개폐상태를 저장하는 변수. 1 : 열림 | 0: 닫힘
bool cover = 0; // 키커버의 상태를 저장하는 변수. 1 : 열림 | 0 : 닫힘

bool LU = 0; // 잠금장치의 상태를 저장하는 변수. 1: 열림 | 0: 닫힘

clock_t C_time = 0; // 문이 닫힌 직후부터 다시 열리기 전까지의 시간을 측정한것을 저장하는 변수
clock_t input_time = 0; // 입력을 받는데 걸린 시간을 측정한것을 저장하는 변수
clock_t cover_time = 0; // 키커버가 열린 직후부터 다시 닫히기 전까지의 시간을 측정한것을 저장하는 변수

/* 여기까지 */
```

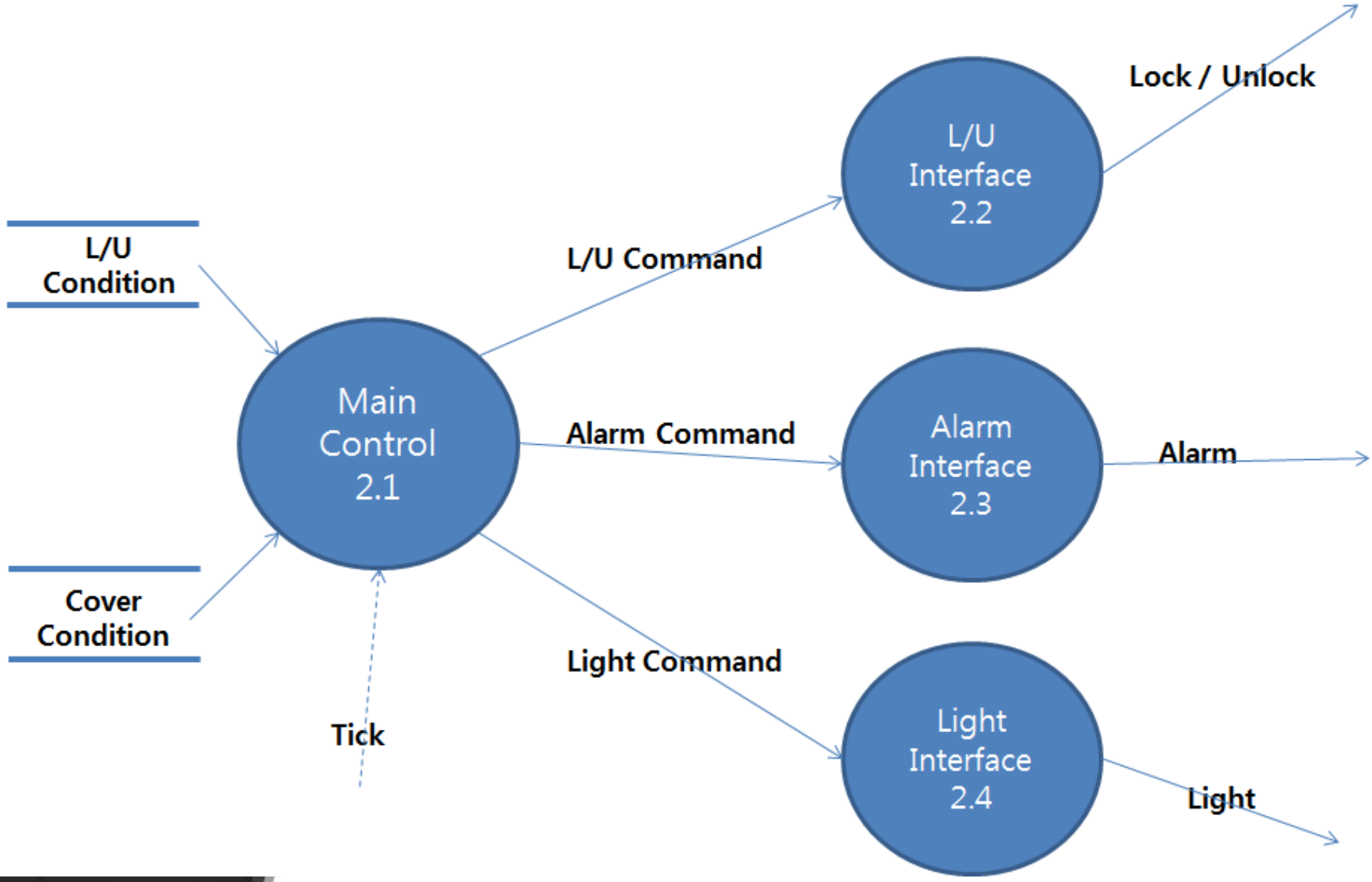
Implementation

```
/* 2. 암호 설정 부분 : PW Saved State*/

// 암호가 설정되있지 않으면 pw button 이외의 input 장치들은 작동하지 않는다. 단, output 장치들은 작동한다.

pw_saved = pw_input(); // 처음으로 입력받는 암호는 저장하여 추후에 비교의 대상으로 삼게됨.
card_saved = 'a';

/* 여기까지 */


/* 3. 각 Input Interface 들로 부터 입력 받는 부분 : Input Interfaces*/

cover_time = cover_condi(&cover);
C_time = oc_condi(&OC);
input_time = input_func(&pw, &card, &Mbutton);

/* 여기까지 */
```

Implementation

```
/* 4. 잠금장치의 Lock / Unlock 상태를 설정해주는 부분 : Determine L/U Condition */

if(Mbutton != tmp_Mb) // 수동개폐장치에 입력이 들어왔을 경우
{
    if(LU == 0) LU = 1;
    else if(LU == 1) LU = 0;
}

else if(pw_saved == pw) // 비밀번호 확인
{
    LU = 1;
}

else if(card_saved == card) // 카드키 확인
{
    LU = 1;
}

/* 여기까지 */
```

Implementation



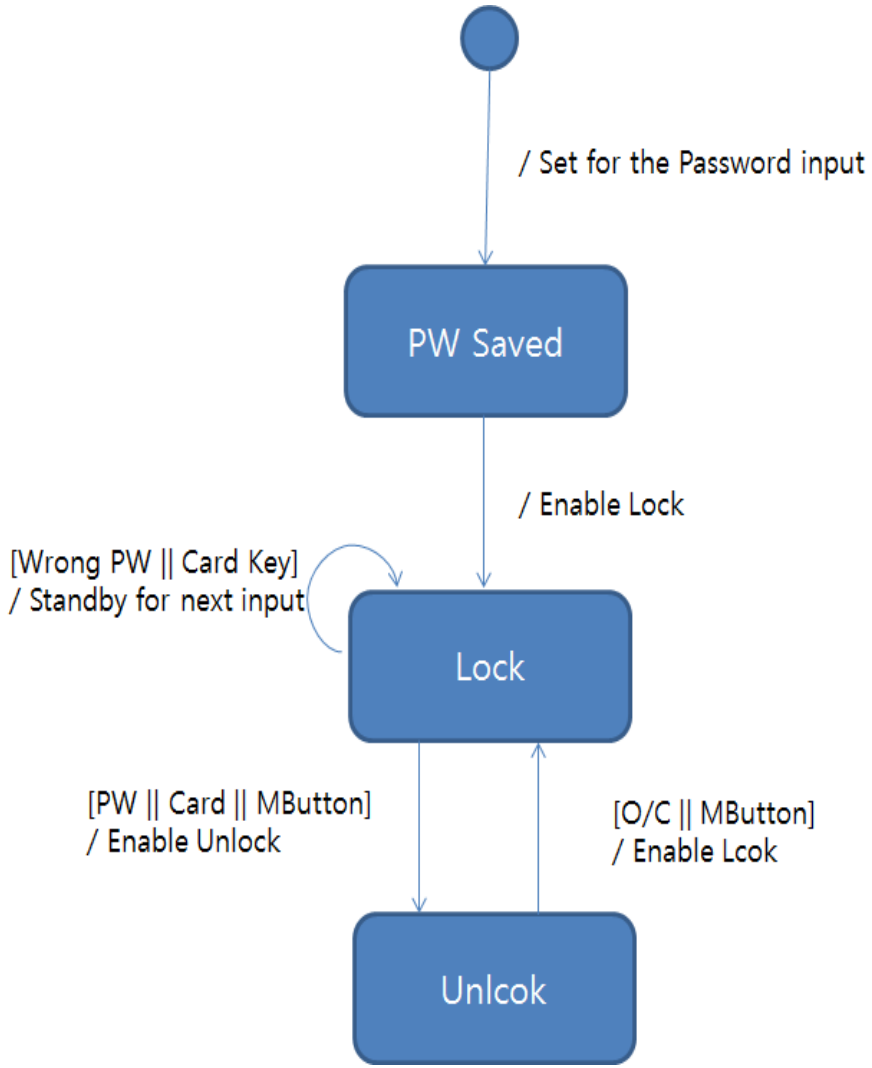
Implementation

```
/* 6. 종합하여 Lock / Unlock 및 Output Interface 들로 출력을 전달해 주는 부분 : Back light, Alarm, Lock / Unlock */

if(LU == 1 && (pw_saved == pw || card_saved == card)) // 비밀번호 혹은 카드키가 일치할경우
{
    unlock();
    alarm1();
}

else if(pw_saved != pw || card_saved != card) // 비밀번호 혹은 카드키가 일치하지 않을 경우
{
    alarm2();
}

else if(input_time > 10) // 입력하는 시간이 10초가 지난경우
{
    alarm3();
}

else if(LU == 0 && OC == 0 && C_time > 3) // 문이 닫힌채로 3초가 지난경우
{
    LU = 1;
    lock();
}

else if(LU == 0) unlock();
```

Implementation

```
if(cover_time < 10 && cover == 1) // 커버가 열려있고 그 상태가 10초 미만인 경우
{
    backlight(1);
}
else // 커버가 닫혀있거나 열린 상태가 10초가 지난 경우
{
    backlight(0);
}

/* 여기까지 */
```

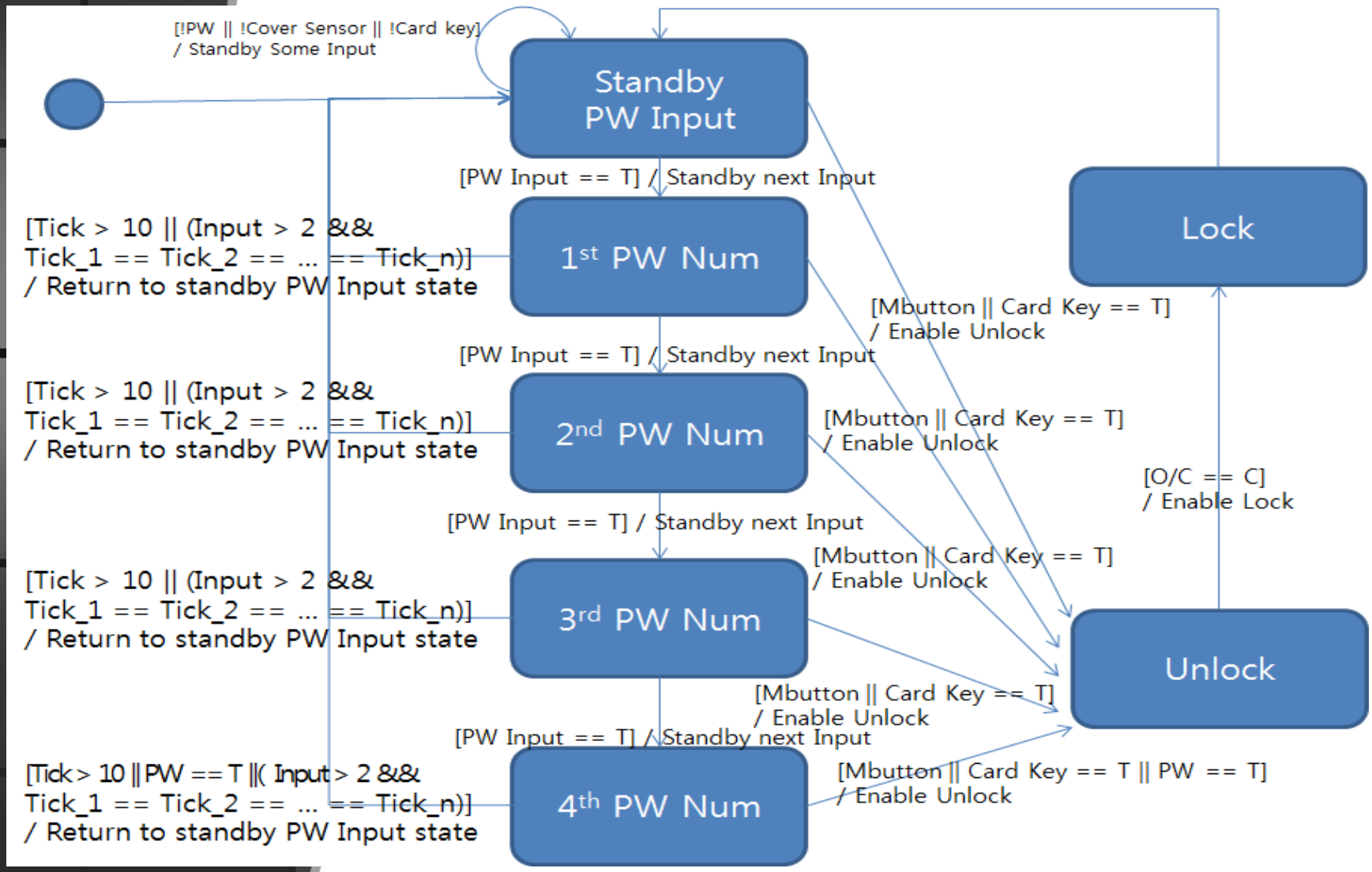
Implementation



Implementation

```
password pw_input()
{
    password pw;
    unsigned short input = 0; // 입력받은 비밀번호 각 자리를 임시로 저장하는 변수
    char i; // 루프문을 돌리기 위한 index 변수

    for(i = 0; i < 4; i++)
    {
        printf("%d번째 비밀번호를 입력하세요 : ", i+1); flush();
        scanf("%hd", &input); flush();

        pw = pw | ( input << (12 - (i*4)) );
    }

    return pw;
}
```

Implementation



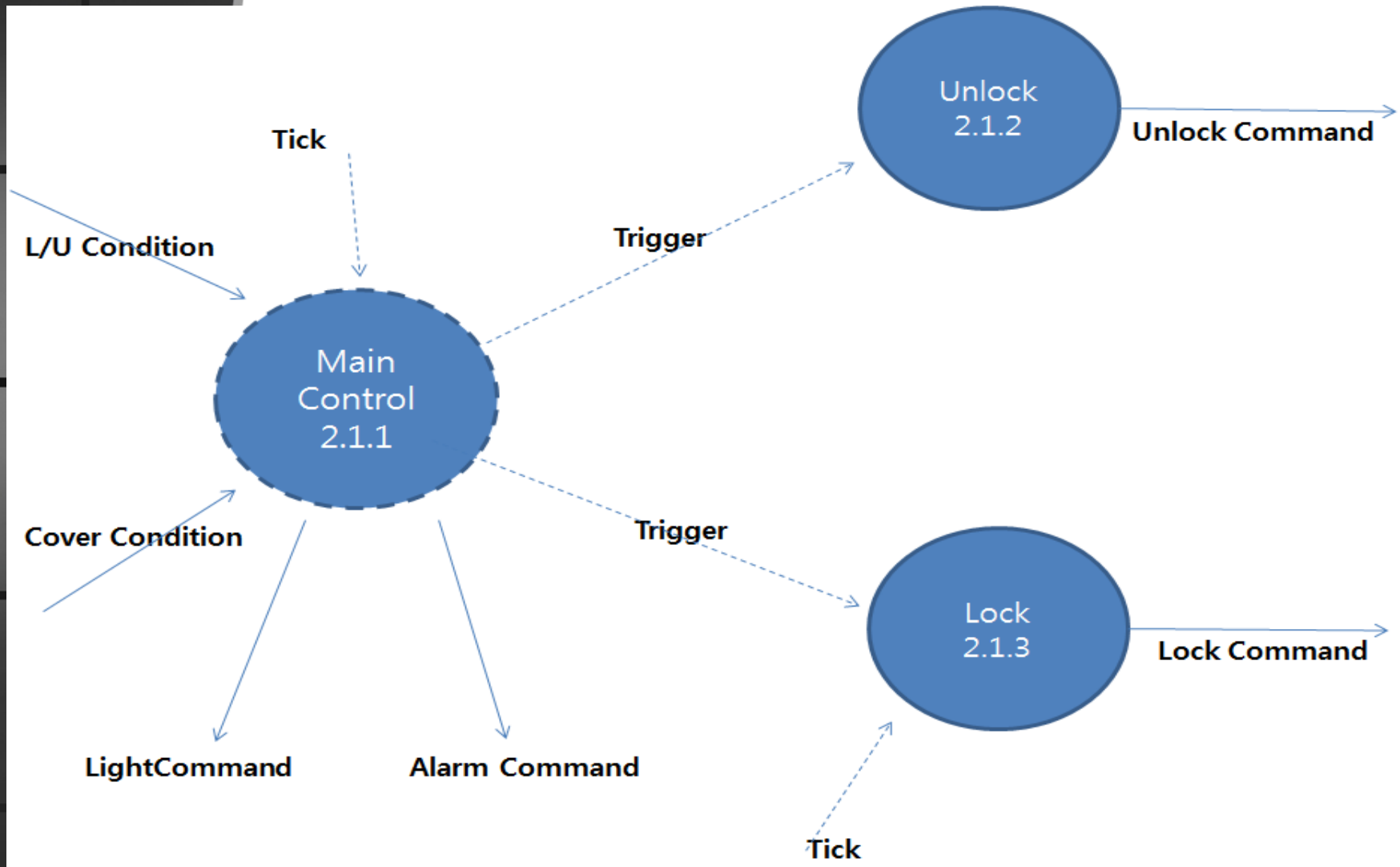
Implementation

```
clock_t input_func(unsigned short *pw, char *card, bool *Mbutton) // 비밀번호를 입력받는 함수
{
    unsigned short input = 0; // 입력받은 비밀번호 각 자리를 임시로 저장하는 변수
    char i; // 루프문을 돌리기 위한 index 변수

    clock_t t1, t2; // 시간을 재기 위한 변수

    t1 = clock(); // 시간 재기 시작
    for(i = 0; i < 4; i++)
    {
        printf("%d번째 입력(숫자 - 0~9 / 수동개폐장치 - ! / 카드키 - a~z /) : ", i+1); flush();
        scanf("%hd", &input); flush(); // short 자료형을 입력받는 format은 int의 자료형을 감소시키는 h옵션을 더한 %hd format으로 입력받는다.

        // 앞쪽에 있는 if문 일수록 우선순위가 더 높음

        if(input == '!') // 수동 개폐장치 입력이 들어왔을 경우 처리
        {
            *Mbutton = 1;
            break;
        }
        else if('a' <= input && input <= 'z') // 숫자 입력 중 카드키 입력이 들어왔을 경우 처리
        {
            *card = input;
            break;
        }
```

Implementation

```
        else if(input > 9) // 숫자가 동시에 입력되었을 경우 처리
        {
            i = 0;
            continue;
        }
        //else if() // 수동개폐장치 버튼을 동시에 많이 입력 하였을 경우 처리
        else *pw = *pw | ( input << (12 - (i*4)) ); // 최종적으로 숫자 4자리가 입력되었을 경우 처리

        t2 = clock(); // 시간 재기
        if ((t2 - t1) > (INPUT_TIME * CLOCKS_PER_SEC)) break; // 10초가 되면 끝남
    }

    return t2 - t1;
}
```

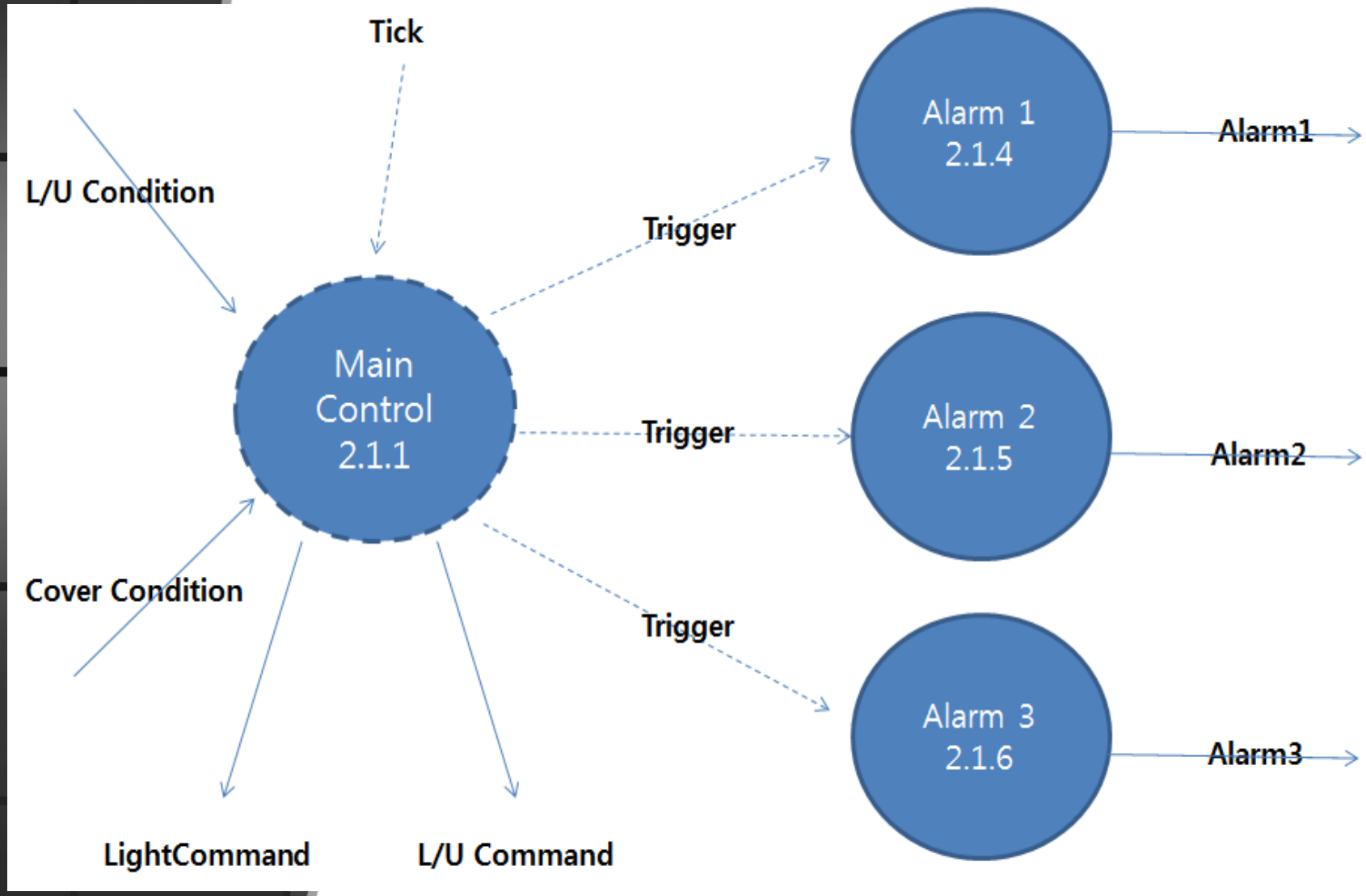
Implementation



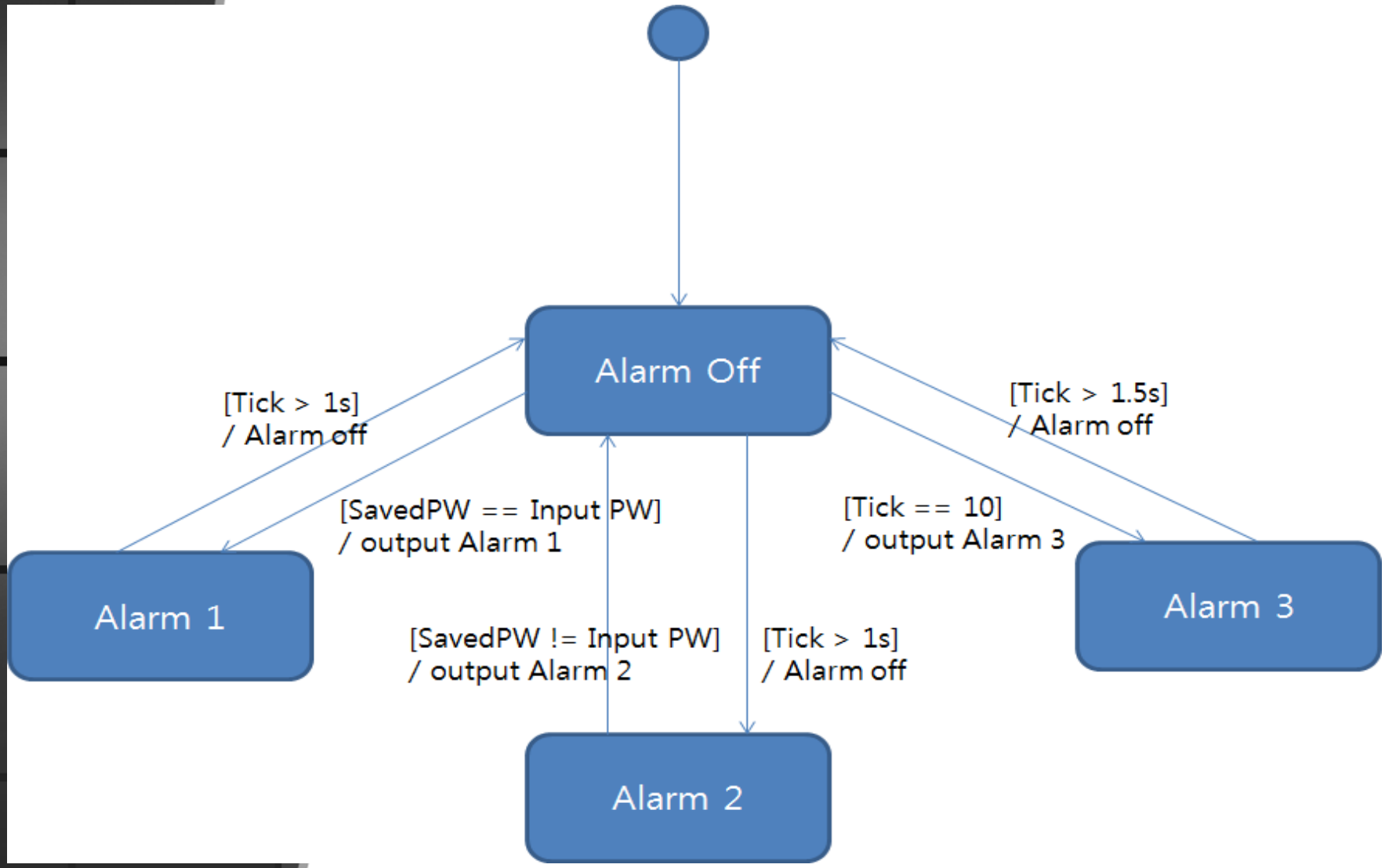
Implementation

```
void lock()
{
    printf("잠금 장치 잠김\n");
}

void unlock()
{
    printf("잠금 장치 열림\n");
}
```

Implementation



Implementation



Implementation

```
void alarm1(int flag) // unlock 시에 울리는 경보음 1
{
    clock_t t1, t2;

    if(flag == 1)
    {
        printf("알람1 출력\n");
        t1 = clock();
        for(;;)
        {
            t2 = clock();
            if ((t2 - t1) > (A1_TIME * CLOCKS_PER_SEC)) break;
        }
    }

    else printf("알람1 꺼짐\n");
}
```

Supplementation

- Virtualization 적용
- 버그 수정
- UT를 대비한 함수 제작

Thanks