DWS

6조

201011359 임종화

201614156 강현우

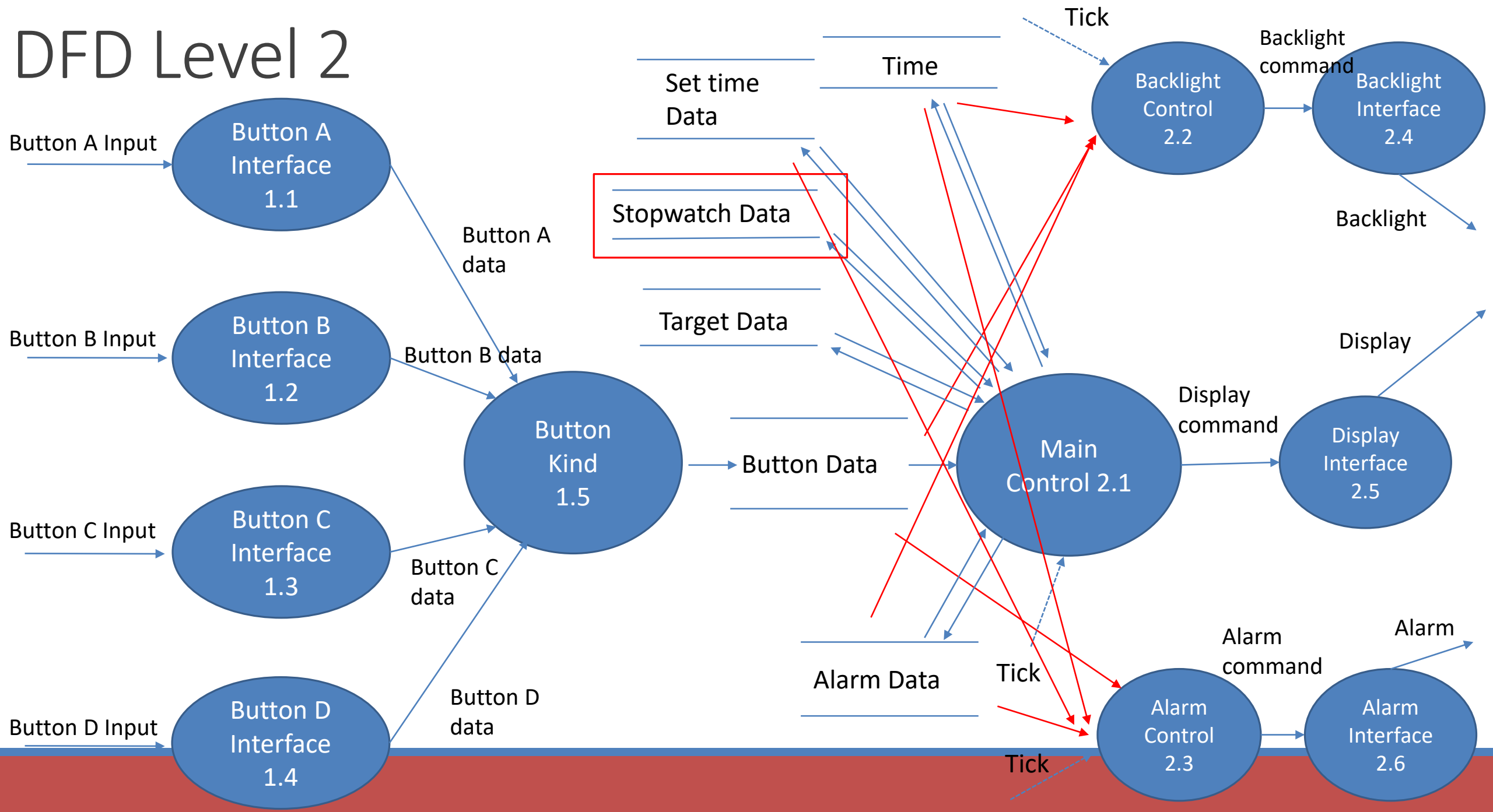
201714167 양현영

201715169 조영래

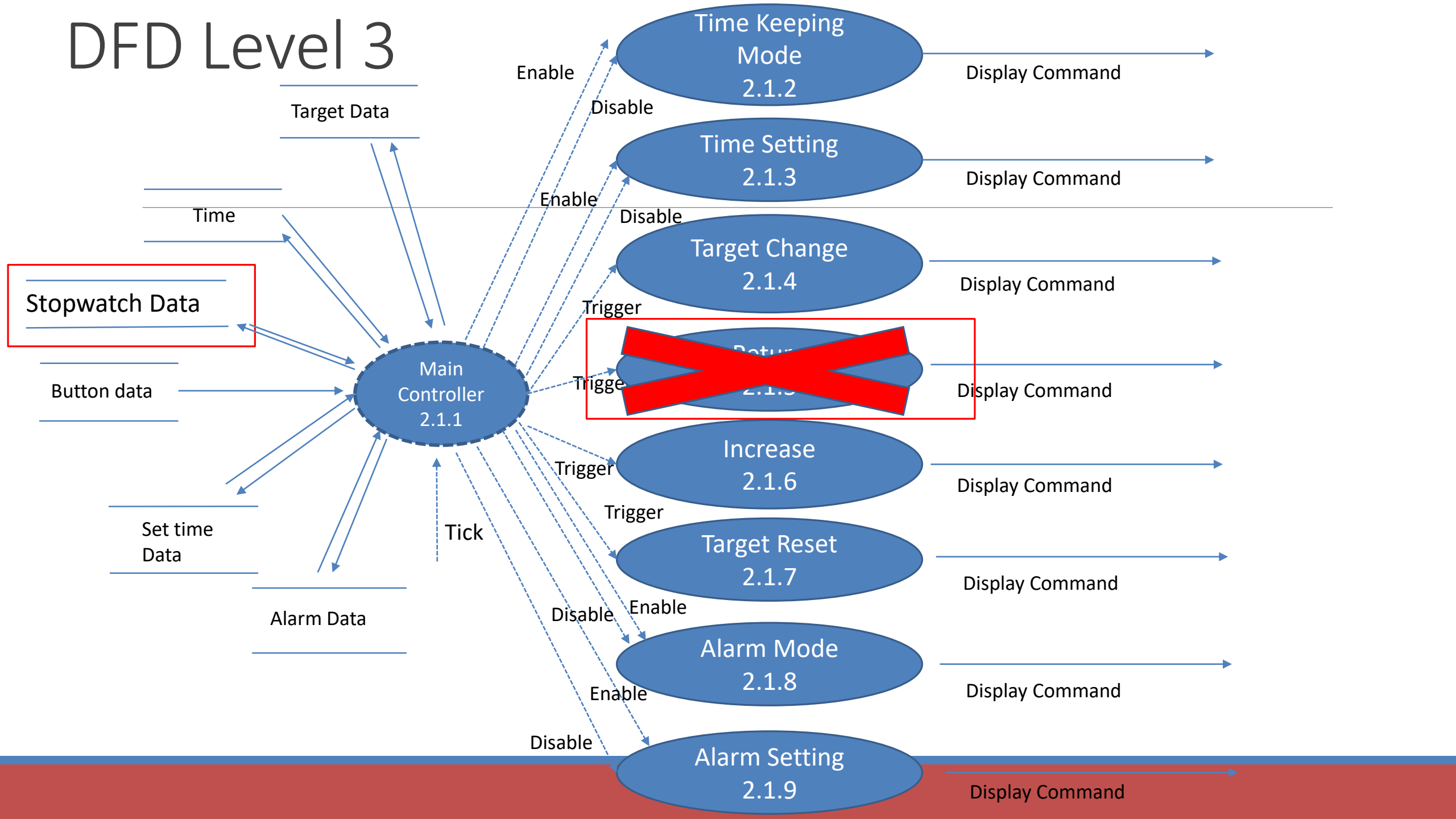
목차

SA수정사항

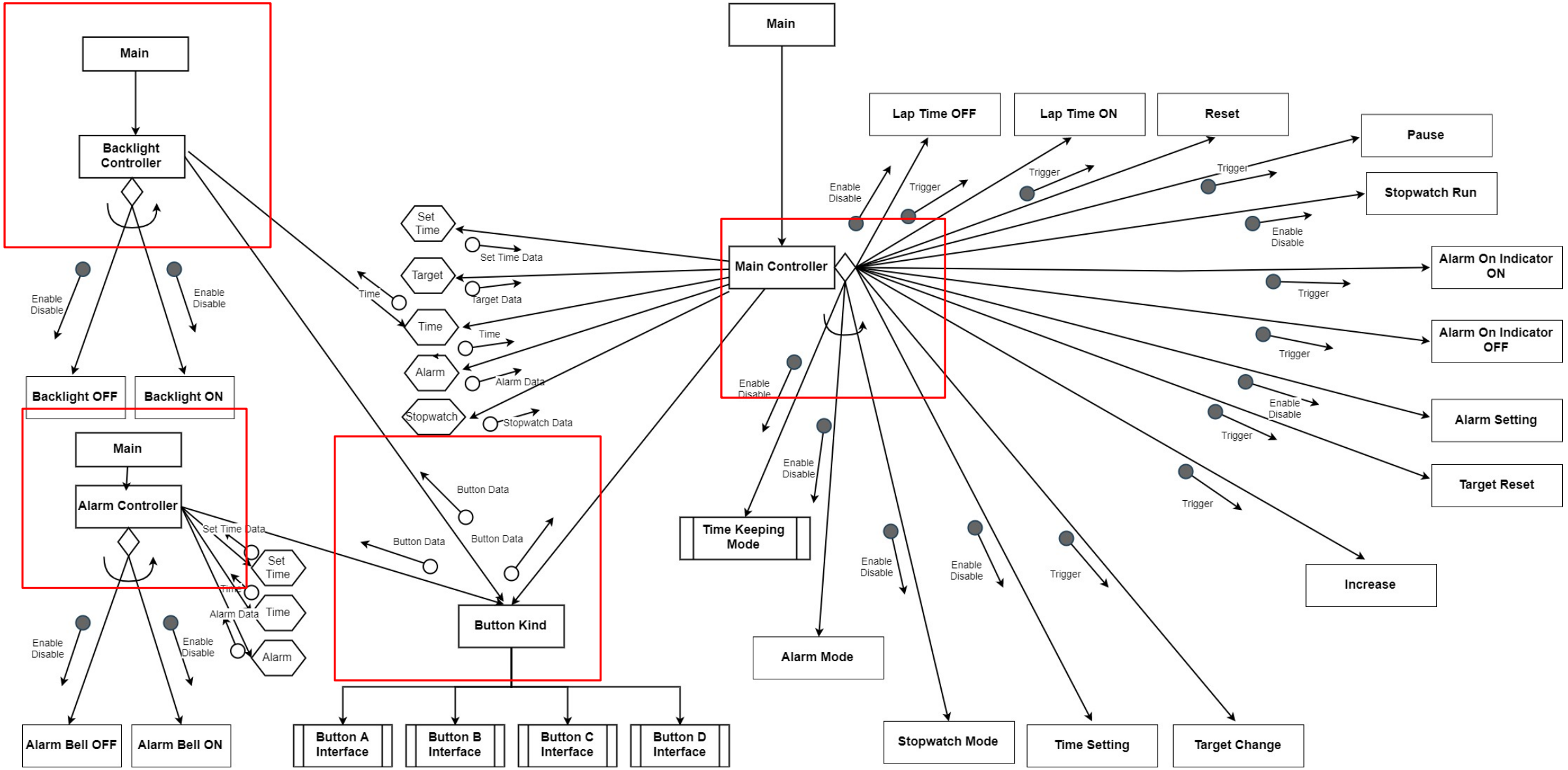
DFD Level 2


Button A Input
Button A Interface 1.1
Button B Input
Button B Interface 1.2
Button C Input
Button C Interface 1.3
Button D Input
Button D Interface 1.4
Button A data
Button B data
Button C data
Button D data
Button Kind 1.5
Button Data
Set time Data
Time
Stopwatch Data
Target Data
Alarm Data
Tick
Backlight Control 2.2
Backlight command
Backlight Interface 2.4
Backlight
Main Control 2.1
Display
Display command
Display Interface 2.5
Tick
Alarm command
Alarm
Alarm Control 2.3
Alarm Interface 2.6
Tick

DFD Level 3


Target Data
Time
Stopwatch Data
Button data
Set time Data
Alarm Data
Tick
Main Controller 2.1.1
Enable
Disable
Time Keeping Mode 2.1.2
Display Command
Time Setting 2.1.3
Display Command
Target Change 2.1.4
Display Command
Trigger
Trigger
Display Command
Trigger
Increase 2.1.6
Display Command
Trigger
Target Reset 2.1.7
Display Command
Disable
Enable
Alarm Mode 2.1.8
Display Command
Enable
Disable
Alarm Setting 2.1.9
Display Command

SD수정사항


Main
Backlight Controller
Enable Disable
Enable Disable
Backlight OFF
Backlight ON
Main
Alarm Controller
Set Time Data
Set Time
Time
Alarm Data
Alarm
Enable Disable
Enable Disable
Alarm Bell OFF
Alarm Bell ON
Set Time
Target
Time
Alarm
Stopwatch
Set Time Data
Target Data
Time
Alarm Data
Stopwatch Data
Time
Button Data
Button Data
Button Data
Button Kind
Button A Interface
Button B Interface
Button C Interface
Button D Interface
Main
Main Controller
Enable Disable
Enable Disable
Time Keeping Mode
Enable Disable
Alarm Mode
Enable Disable
Stopwatch Mode
Enable Disable
Time Setting
Trigger
Target Change
Lap Time OFF
Lap Time ON
Reset
Pause
Stopwatch Run
Alarm On Indicator ON
Alarm On Indicator OFF
Alarm Setting
Target Reset
Increase
Trigger
Trigger
Trigger
Trigger
Enable Disable
Trigger
Enable Disable
Trigger
Trigger
Trigger

개발 도구

-Cygwin

-gcc version 7.4.0 (GCC)

-사용 헤더파일

```
#include<stdio.h>
#include<unistd.h>
#include<time.h>
#include<stdlib.h>
#include <pthread.h>
#include <termio.h>
```

Main

```c
int main() {
        pthread_t trd_tick;
        pthread_t trd_tick_blink;
        pthread_t trd_alarm;
        pthread_t trd_blt;

        init_keyboard();

        // Creates Tick thread
        if(pthread_create(&trd_tick, NULL, tick_generator, NULL)){
                perror("Thread error.\n");
                exit(1);
        }
        // Creates Tick(for blink) thread
        if(pthread_create(&trd_tick_blink, NULL, blink_generator, NULL)){
                perror("Thread error.\n");
                exit(1);
        }
        // Creates Alarm thread
        if(pthread_create(&trd_alarm, NULL, alarm_check, NULL)){
                perror("Thread error.\n");
                exit(1);
        }
        // Create backlight thread
        if(pthread_create(&trd_blt, NULL, backlight_control, NULL)){
                perror("Thread error.\n");
                exit(1);
        }

        while(1){
                time_keeping_mode();
                alarm_mode();
                stopwatch_mode();
        }

        close_keyboard();

        // Wait for threads end and take resources back
        pthread_join(trd_tick, NULL);
        pthread_join(trd_tick_blink, NULL);
        pthread_join(trd_alarm, NULL);
        pthread_join(trd_blt, NULL);

        return 0;
}
```

Tick generator

```c
/* Thread1
 * Add current time +1 sec, per second
 */
void* tick_generator(void *data){
        while(1){
                add_time(6, TRUE);
                sleep(1);
        }
}

void* blink_generator(void *data){
        while(1){
                is_blink = is_blink ? FALSE : TRUE;
                usleep(500000);
        }
}
```

```c
/* <time.h>
 * tm_sec: second, 0~59
 * tm_min: minute, 0~59
 * tm_hour: hour, 0~23
 * tm_mday: day, 1~31
 * tm_mon: month, 0~11
 * tm_year: year, starts from 1900
 *
 * (tm_wday: 0~6, starts from sunday)
 *
 * year -= 1900, month -= 1
 */
//struct tm my_time = {0, 0, 0, 1, 1-1, 2019-1900, 0, 0, 0};

struct tm my_time = {40, 59, 23, 19, 11-1, 2019-1900, 0, 0, 0};
```

```c
/* Increses time value by +1
 * If reached to maximum, back to start value/stop at max value
 * 2019 <= year <= 2099
 *
 * Input:
 *      tm_select: Year/Month/Day/Hour/Minute/Second == 1/2/3/4/5/6
 *      is_carry: add by Tick/add by manually(time Setting) == TRUE/FALSE
 */
void add_time(const int tm_select, const char is_carry){
        // Carry is TRUE. +1 second
        if(is_carry){
                // Add second only before 2019-12-31 23:59:59
                if((int)mktime(&my_time) < (long)END_OF_2099){
                        my_time.tm_sec++;
                }
        }
        else{
                switch(tm_select){
                        // Year
                        case 1:
                                if(my_time.tm_year+1900 < 2099){
                                        my_time.tm_year++;
                                }
                                else{
                                        // Restart with 2091
                                        my_time.tm_year = 2019-1900;
                                }
                                break;
                        // Month
                        case 2:
                                if(my_time.tm_mon+1 < 12){
                                        my_time.tm_mon++;
                                }
                                else{
                                        my_time.tm_mon = 0;
                                }
                                break;
                        // Day
                        case 3:
```

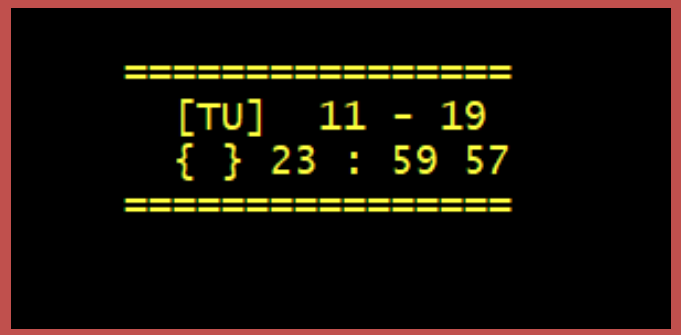
Alarm Check

```
struct _myvar_time {
        int var_hour;
        int var_min;
        int var_sec;
        int centi_sec;
} my_alarm_time = {0, 0, 0, 0};
```

```
/* Thread2
 * If current time == alarm time and,
 * if alarm indicator is on:
 *      make beep sound per alarm_count and,
 *      alarm_count--
 */
void* alarm_check(void *data){
        int alarm_count = ALARM_COUNT;

        while(1){
                if(is_alarm_on){
                        if(alarm_count > 0){
                                if(my_alarm_time.var_hour == my_time.tm_hour){
                                        if(my_alarm_time.var_min == my_time.tm_min){
                                                is_alarm_ring = TRUE;
                                                system("echo -e '\a'"); // Beep sound
                                                sleep(1);
                                                alarm_count--;
                                        }
                                }
                        }
                        else{
                                is_alarm_ring = FALSE;
                                is_alarm_on = FALSE;
                        }
                }
                else{
                        if(alarm_count != ALARM_COUNT){
                                alarm_count = ALARM_COUNT;
                        }
                }
        }
}
```

Backlight Mode

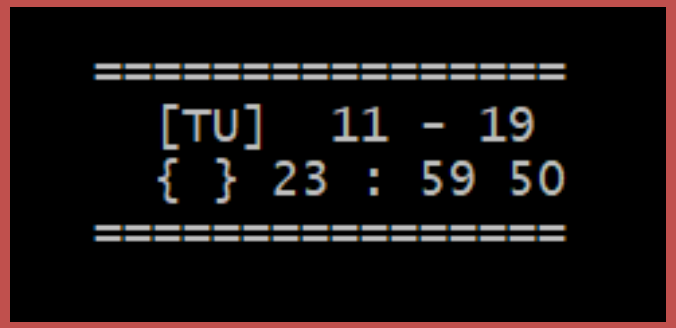
Button D 입력




```c
/* Thread3
 * Change the color of font by 2 second
 */
void* backlight_control(void *data){
        while(1){
                if(is_backlight_on){
                        printf(YELLOW_B);
                        usleep(2000000);
                        printf(RESET);

                        is_backlight_on = FALSE;
                }
        }
}
```

Timekeeping Mode

현재 시간 표시



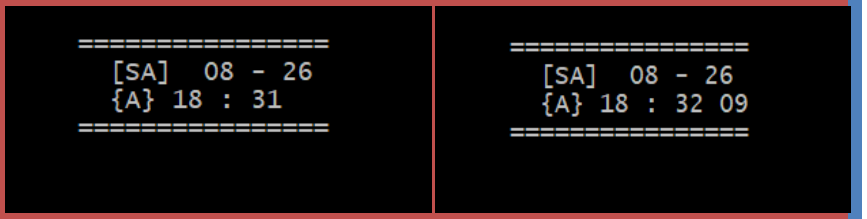

```c
/* Mode 1: Time Keeping Mode
 *
 * Input key:
 *      A:
 *      B:
 *      C: Mode change
 *      D: Backlight on
 */
void time_keeping_mode(){
        char ch = 0;
        int sec_detect = 0;

        while(1){
                // Refresh UI only if time changes
                if(sec_detect != my_time.tm_sec){
                        ui_time(1, FALSE);
                        sec_detect = my_time.tm_sec;
                }

                ch = getch_abcd();

                // If alarm ringing, ignore original key setting
                if(!is_alarm_ring){
                        switch (ch){
                                case 'd':
                                        is_backlight_on = TRUE;
                                        break;
                                case 'c':
                                        // Change mode, quit timekeeping mode
                                        return;
                                case 'a':
                                        // Timesetting mode
                                        time_setting_mode();
                                        break;
                                default:
                                        break;
                        }
                }
                else{
                        if(ch == 'a' || ch == 'b' || ch ==      || ch == 'd'){
                                is_alarm_on = FALSE;
                        }
                }

        }

        return;
}
```

Button A 입력

Time Setting mode 실행

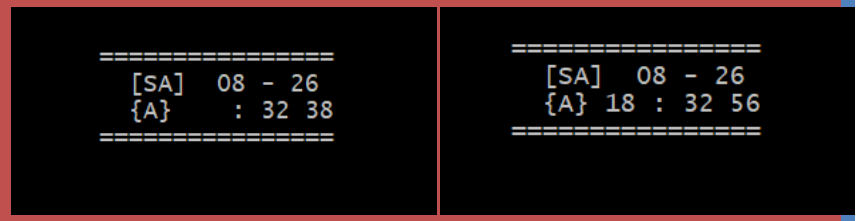
Time Setting

'초' 깜빡임



Button C 입력

'시' 깜빡임




```
void time_setting_mode(){
        char blink_detect = FALSE;
        char cursor_p = 6;
        char ch = 0;

        while(1){
                // Refresh UI only if blinks
                if(blink_detect != is_blink){
                        ui_time(cursor_p, is_blink);
                        blink_detect = is_blink;
                }

                ch = getch_abcd();

                // If alarm ringing, ignore original key setting
                if(!is_alarm_ring){
                        switch (ch){
                                case 'd':
                                        is_backlight_on = TRUE;
                                        break;
                                case 'c':
                                        switch(cursor_p){
                                                case 1:
                                                        cursor_p = 2;
                                                        break;
                                                case 2:
                                                        cursor_p = 3;
                                                        break;
                                                case 3:
                                                        cursor_p = 6;
                                                        break;
                                                case 4:
                                                        cursor_p = 5;
                                                        break;
                                                case 5:
                                                        cursor_p = 1;
                                                        break;
                                                case 6:
                                                        cursor_p = 4;
                                                        break;
                                        }
                                        break;
```

Time Setting

'월' 로 타겟 설정된 상태

```
================
  [TU]  09 - 26
  {A} 19 : 40 51
================
```

Button B 입력

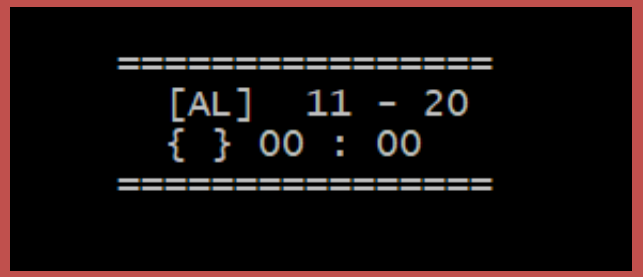
'월' 1증가(날짜에 맞춰 요일도 변함)

```
================
  [TH]  10 - 26
  {A} 19 : 41 23
================
```

```
                        case 'b':
                                switch(cursor_p){
                                        case 1:
                                                add_time(1, FALSE);
                                                break;
                                        case 2:
                                                add_time(2, FALSE);
                                                break;
                                        case 3:
                                                add_time(3, FALSE);
                                                break;
                                        case 4:
                                                add_time(4, FALSE);
                                                break;
                                        case 5:
                                                add_time(5, FALSE);
                                                break;
                                        case 6:
                                                add_time(6, FALSE);
                                                break;
                                }
                                ui_time(cursor_p, FALSE);
                                break;
                        case 'a':
                                // Change mode, quit time setting mode
                                return;
                        default:
                                break;
                }
        }
        else{
                if(ch == 'a' || ch == 'b' || ch == 'c' || ch == 'd'){
                        is_alarm_on = FALSE;
                }
        }

    }

    return;

}
```

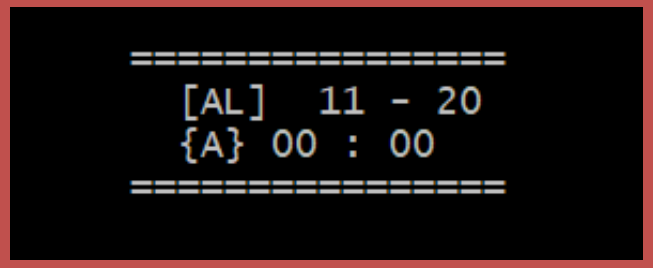
Alarm Mode

Alarm Indicator Off



Button B 입력

Alarm Indicator On




```c
/* Mode 2: Alarm Mode
 *
 * Input key:
 *      A: Alarm setting / back
 *      B: Alarm indicator on / off
 *      C: Mode change
 *      D: Backlight on
 */
void alarm_mode(){
        char blink_detect = FALSE;
        char ch = 0;

        ui_alarm(5, FALSE);

        while(1){
                // Refresh UI only if blinks
                if(blink_detect != is_blink){
                        ui_alarm(4, FALSE);
                        blink_detect = is_blink;
                }

                ch = getch_abcd();

                // If alarm ringing, ignore original key setting
                if(!is_alarm_ring){
                        switch (ch){
                                case 'd':
                                        is_backlight_on = TRUE;
                                        break;
                                case 'c':
                                        // Change mode, quit alarm mode
                                        return;
                                case 'b':
                                        // Toggle alarm indicator on / off
                                        is_alarm_on = is_alarm_on ? FALSE : TRUE;
                                        break;
                                case 'a':
                                        // Alarm setting mode
                                        alarm_setting_mode();
                                        break;
                                default:
                                        break;
                        }
                }
                else{
                        if(ch){
                                is_alarm_on = FALSE;
```

Button A 입력
Alarm Setting mode
실행

Alarm Setting

```
void add_alarm(const int var_select){
        switch(var_select){
                // Hour
                case 4:
                        if(my_alarm_time.var_hour < 23){
                                my_alarm_time.var_hour++;
                        }
                        else{
                                my_alarm_time.var_hour = 0;
                        }
                        break;
                // Minute
                case 5:
                        if(my_alarm_time.var_min < 59){
                                my_alarm_time.var_min++;
                        }
                        else{
                                my_alarm_time.var_min = 0;
                        }
                        break;
        }
}
```

```
void alarm_setting_mode(){
        char blink_detect = FALSE;
        char is_point_h = TRUE;
        char ch = 0;

        while(1){
                // Refresh UI only if blinks
                if(blink_detect != is_blink){
                        ui_alarm(is_point_h, is_blink);
                        blink_detect = is_blink;
                }

                ch = getch_abcd();

                // If alarm ringing, ignore original key setting
                if(!is_alarm_ring){
                        switch (ch){
                                case 'd':
                                        is_backlight_on = TRUE;
                                        break;
                                case 'c':
                                        // Toggle hour / min
                                        is_point_h = is_point_h ? FALSE : TRUE;
                                        break;
                                case 'b':
                                        if(is_point_h){
                                                add_alarm(4);
                                        }
                                        else{
                                                add_alarm(5);
                                        }
                                        ui_alarm(is_point_h, FALSE);
                                        break;
                                case 'a':
                                        // Change mode, quit alarm setting mode
                                        return;
                                default:
                                        break;
                        }
                }
                else{
                        if(ch == 'a' || ch == 'b' || ch == 'c' || ch == 'd'){
                                is_alarm_on = FALSE;
                        }
                }

        }
```

Alarm Setting Mode

```
================
  [AL]  08 - 26
  { }    : 00
================
```

Button C 입력

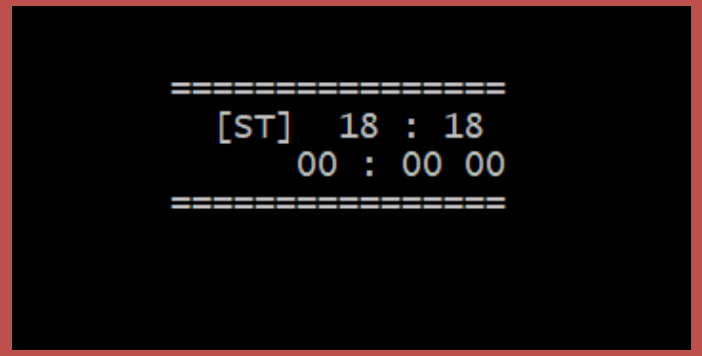
```
================
  [AL]  11 - 20
  {A} 00 :
================
```

Button B 입력

```
================
  [AL]  11 - 20
  {A} 01 : 00
================
```

Stopwatch Mode

Stopwatch Mode




```c
/* Mode 3: Stopwatch Mode
 *
 * Input key:
 *      A:
 *      B:
 *      C: Mode change
 *      D: Backlight on
 */
void stopwatch_mode(){
        clock_t start_c;
        struct _myvar_time tmp_stw = {0, 0, 0, 0};
        int sec_detect = 0;
        char ch = 0;

        ui_stopwatch(&tmp_stw);

        while(1){
                // Refresh UI only if time changes
                if(sec_detect != my_time.tm_sec){
                        ui_stopwatch(&tmp_stw);
                        sec_detect = my_time.tm_sec;

                        gotoxy(20, 25);
                        printf("st MODE seno");
                }

                ch = getch_abcd();

                // If alarm ringing, ignore original key setting
                if(!is_alarm_ring){
                        switch (ch){
                                case 'd':
                                        is_backlight_on = TRUE;
                                        break;
                                case 'c':
                                        // Change mode, quit stopwatch mode
                                        return;
                                case 'b':
                                        // Stopwatch on
                                        stopwatch_run(&tmp_stw);
                                        break;
                                case 'a':
                                        // Clear stopwatch
                                        tmp_stw.var_min = 0;
                                        tmp_stw.var_sec = 0;
                                        tmp_stw.centi_sec = 0;
                                        break;
                                default:
```

Stopwatch Run

```
void add_stopwatch(struct _myvar_time *my_stw_time){
        if(my_stw_time->centi_sec < 59){
                my_stw_time->centi_sec++;
        }
        else{
                my_stw_time->centi_sec = 0;
                if(my_stw_time->var_sec < 59){
                        my_stw_time->var_sec++;
                }
                else{
                        my_stw_time->var_sec = 0;
                        if(my_stw_time->var_min < 59){
                                my_stw_time->var_min++;
                        }
                        else{
                                my_stw_time->var_min = 59;
                                my_stw_time->var_sec = 59;
                                my_stw_time->centi_sec = 98;
                        }
                }
        }
}
```

```
void stopwatch_run(struct _myvar_time *my_stw_time){
        struct _myvar_time lap_time = {0, 0, 0, 0};
        int sec_detect = 0;
        char is_lap = FALSE;
        char ch = 0;

        while(1){
                //clock_delay(0.01);
                usleep(10);
                add_stopwatch(my_stw_time);

                // Refresh UI only if time changes
                if(sec_detect != my_stw_time->centi_sec){
                        if(is_lap){
                                ui_stopwatch(&lap_time);
                        }
                        else{
                                ui_stopwatch(my_stw_time);
                        }
                        sec_detect = my_stw_time->centi_sec;
                }

                gotoxy(20, 25);
                printf("st mode seye");

                ch = getch_abcd();

                // If alarm ringing, ignore original key setting
                if(!is_alarm_ring){
                        switch (ch){
                                case 'd':
                                        is_backlight_on = TRUE;
                                        break;
                                case 'b':
                                        // Stopwatch off or lap time off
                                        if(is_lap){
                                                is_lap = FALSE;
                                        }
                                        else{
                                                return;
                                        }
                                        break;
                                case 'a':
                                        // Lap time on, Renew lap time
                                        lap_time.var_min = my_stw_time->var_min;
                                        lap_time.var_sec = my_stw_time->var_sec;
                                        lap_time.centi_sec = my_stw_time->centi_sec;
```

Stopwatch Run

```
==============
  [ST]  18 : 20
       00 : 00 00
==============
```

Button B 입력

Laptime&Pause

```
==============
  [ST]  18 : 19
       01 : 39 51
==============
```

Laptime&Pause

Button B 입력

```
==============
  [ST]  18 : 20
       02 : 59 36
==============
```

Button A입력