과제번호	2020 -12

2020년도 전자공학과
종합설계 프로젝트 최종 보고서

스마트 수족관

Smart Aquarium

청주대학교 이공대학 전자공학과				
제출일자	2020년 06월 10일			
팀 구성원	학번	15420057	성명	남상호
	학번	15421034	성명	최영민
지도교수	한철수			

목 차

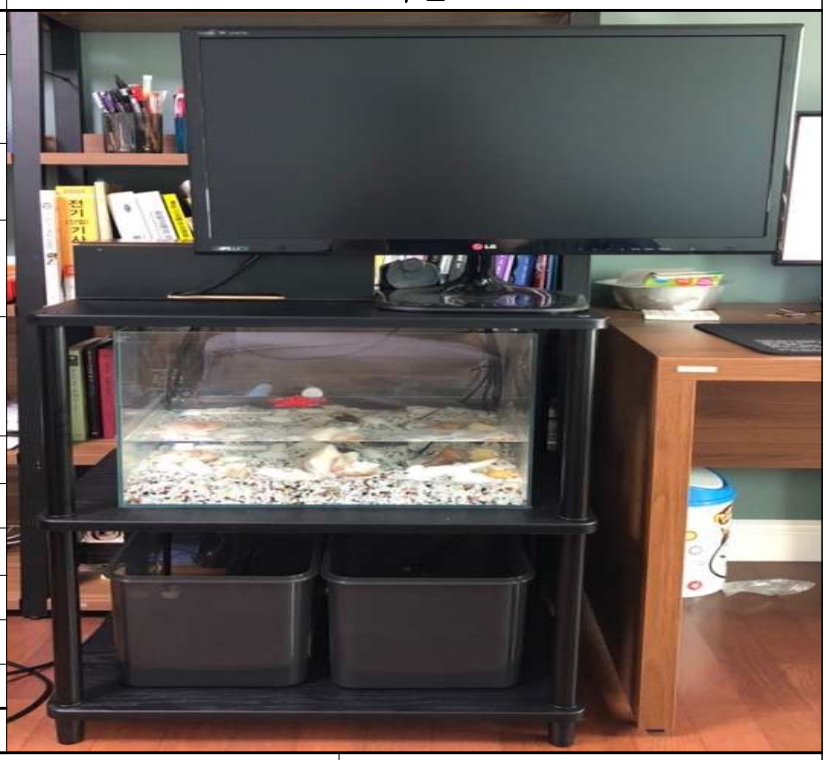
과제 요약서

과제번호	2020 - 12

작품 제목	스마트 수족관
	Smart Aquarium

제원		
하드웨어 (H/W)	크기	600x385x908
	무게	수량에 따라 유동적
	프로세서	ATmega128 Raspberry Pi
소프트웨어 (S/W)	OS	Raspbian AVR
	개발 TOOL	Python Codevision AVR

외관



개발 기간
2020.01.06 ~ 2020.05.31

개발 소요비	
재료비	263,920
제작비	0
기타 경비	0
총 소요비	263,920

과제 수행자	성명	남상호	최영민
	Phone	01034005061	01054364469
	Email	nsh9811@Naver.com	po0230@naver.com
	담당	S/W 설계	H/W 설계
	-	-	-

이와 같이 2020 년도 전자공학과 종합설계 프로젝트 최종 보고서를 제출합니다.

2020 년 06월 10일

과제 수행자 : 남상호 (인)

과제 수행자 : 최영민 (인)

지도교수 : 한철수 (인)

청주대학교 이공대학 전자공학과

I. 서론 스마트 시대

1. 목표 - 수족관을 손쉽게 관리 할 수 있고 반려동물의 삶의 질을 높힌다. 자동 급여기를 사용해 손쉽게 밥을 줄 수 있다. 수온에 민감한 열대어를 위해 항상 수온을 알맞게 유지시킨다. 까다로운 환수를 모터를 사용하여 손쉽게 할 수 있다. 수족관에 LED가 켜지게 하여 인테리어 기능을 추가한다.

2. 추진 배경 및 필요성

- 제목 : 스마트 수족관
- 평소에 관심이 있었던 반려동물을 기를 때 불편했던 점이나 생각했던 기능들을 전자공학을 접합시켜 구현해본다.
- 현재 시중에 판매되는 제품들은 사용자가 원하는 시간대에 조작이 어렵고 이미 설정되어 나오는 시간대에서만 조작이 가능하다. 이번에 제작한 수족관은 사용자가 원하는 시간대에 조작이 가능하고 터치스크린을 통하여 통합 조작이 가능하다.

3. 연구 / 개발 내용

- ATmega128 과 Raspberry Pi를 UART 통신을 데이터를 송/수신 하였다.
- Raspberry Pi에서 원하는 온도를 설정 한 뒤 온도 센서를 거쳐 ATmega128을 통하여 히터를 작동 시킨다.
- Raspberry Pi에서 LED ON/OFF 버튼을 눌러 ATmega128에 연결되어 있는 릴레이 모듈에 DC 12V 외부 전압과 연결되어 있는 LED를 ON/OFF 한다.
- Raspberry Pi에서 Water IN 버튼을 눌러 ATmega128에 연결되어 있는 릴레이 모듈에 ATmega128에서 출력되는 5V 전압(VCC)과 워터 펌프를 이용하여 상단부에 설치되어 있는 수위 센서의 값이 950초과가 될 때 까지 물을 채운다. 포트에서 바로 펌프에 연결하지 않는 이유는 포트 전압이 부족하기 때문이다.
- Raspberry Pi에서 Water OUT 버튼을 눌러 ATmega128에 연결되어 있는 아래쪽에 설치되어 있는 수위 센서의 값이 50미만이 될 때 까지 수조안에 물을 뺀다. 워터 펌프를 연결 할 시에 포트 한 개의 전압으로는 전압이 부족하여 포트 2개의 전압을 사용하여 동작 시킨다.
- Raspberry Pi에서 FEED 버튼을 눌러 ATmega128에 연결되어 있는 릴레이 모듈에 사료 통을 한 바퀴 회전 시킨다. 이 사료 통은 DC모터와 마이크로 리미트 스위치를 사용하여 제작되었다. 모터에는 스위치를 통하여 5V 전압이 상시 인가되어 있는데 모터가 한 바퀴 회전하여 스위치를 누르게 되면 회로가 개방되는

방식이다. 그러므로 DC모터 자체에 0.5초 동안만 릴레이를 통하여 5V전압을 준다.

4. 기대효과 및 활용방안

- 현재 시중에 판매되어 있는 사료 공급기와 온도 조절기는 온도와 사료 공급 빈도를 세팅되어 나오고 변경하기 어려워 사용자가 원하는 만큼 동작시키기에 어려움이 있다. 하지만 이번 프로젝트를 통하여 만든 제품은 단순히 코드에 있는 숫자를 바꿈에 따라 사용자가 원하는 만큼 온도와 사료 공급횟수를 변경할 수 있다.
- 터치스크린 하나로 기존 세팅되어 있는 값들을 동작 시킬 수 있어 누구나 사용이 가능하다.

Ⅱ. 설계

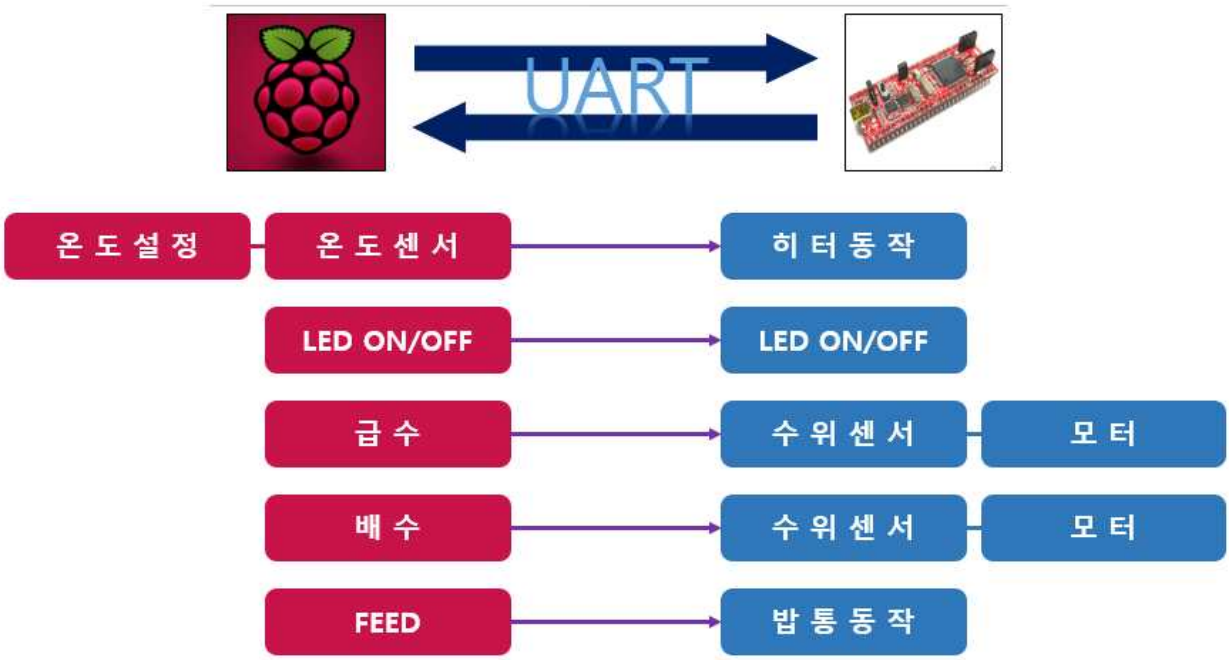
1. 관련이론 및 개발환경

ATmega128을 이용하여 CodeVision을 통한 모터 동작, ADC 변환, 릴레이 사용
ATmega128과 RaspberryPi를 UART통신을 이용하여 데이터 송/수신
RaspberryPi의 DS18B20 1_Wire 통신, Phython의 TKinter를 이용한 GUI 만들기

2. 개념설계

현재 시중에 판매되어 있는 사료 공급기와 온도 조절기는 온도와 사료 공급 빈도를 세팅되어 나오고 변경하기 어려워 사용자가 원하는 만큼 동작시키기에 어려움이 있어 사용자가 원하는 동작을 이번 프로젝트를 이용하여 가능하게 한다.

3. 상세설계



<그림 II-1> 프로젝트 흐름도

- 터치스크린에서 원하는 온도를 설정한 뒤에 온도센서를 거쳐 히터를 작동한다.
- 터치스크린에서 LED ON/OFF 버튼을 눌러 LED를 ON/OFF 한다.
- 터치스크린에서 Water IN 버튼을 눌러 워터 펌프를 이용하여 상단부에 설치되어 있는 수위 센서의 값이 950초과가 될 때 까지 물을 채운다.

- 터치스크린에서 Water OUT 버튼을 눌러 수위 센서의 값이 50미만이 될 때 까지 수조안에 물을 뺀다.
- 터치스크린에서 FEED 버튼을 눌러 사료 통을 한 바퀴 회전 시킨다.

Ⅲ. 개발 일정 및 자원 관리

1. 프로젝트 팀 구성 및 역할

이름	역할
남상호	- S/W 개발
최영민	- H/W 개발

2. 개발 일정

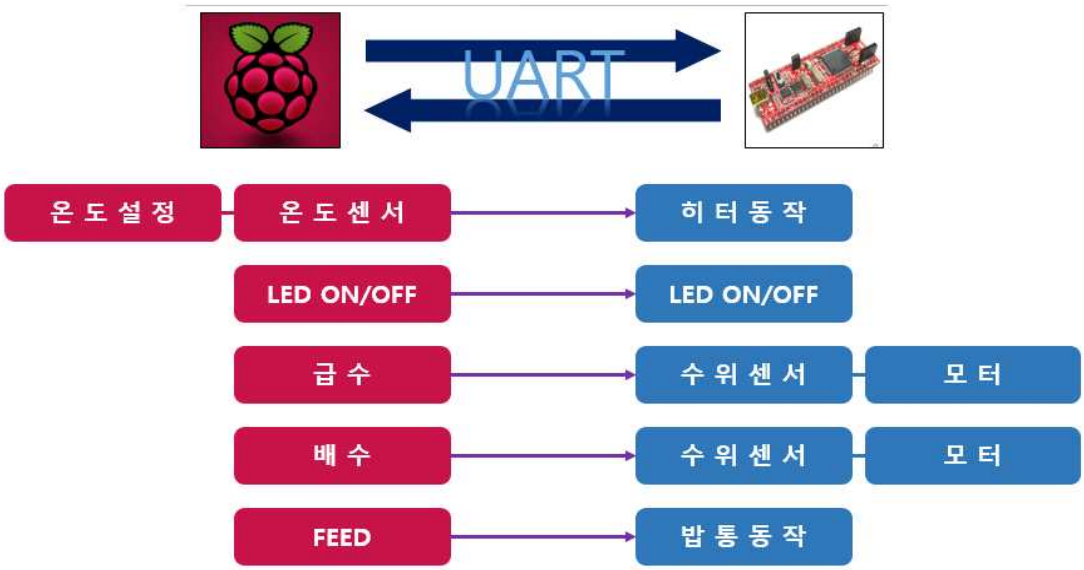
항목	세부내용	1 월	2 월	3 월	4 월	5 월	비고
요구사항분석	요구 분석	■					
	회로 작성	■					
	부품 주문	■					
관련분야연구	ATmega128 Raspberry Pi 연동		■	■			
	관련 시스템 분석		■	■			
설계	시스템 설계				■	■	
구현	코딩 및 모듈 테스트				■	■	
테스트	외관 구현 및 테스트				■	■	

3. 프로젝트 비용

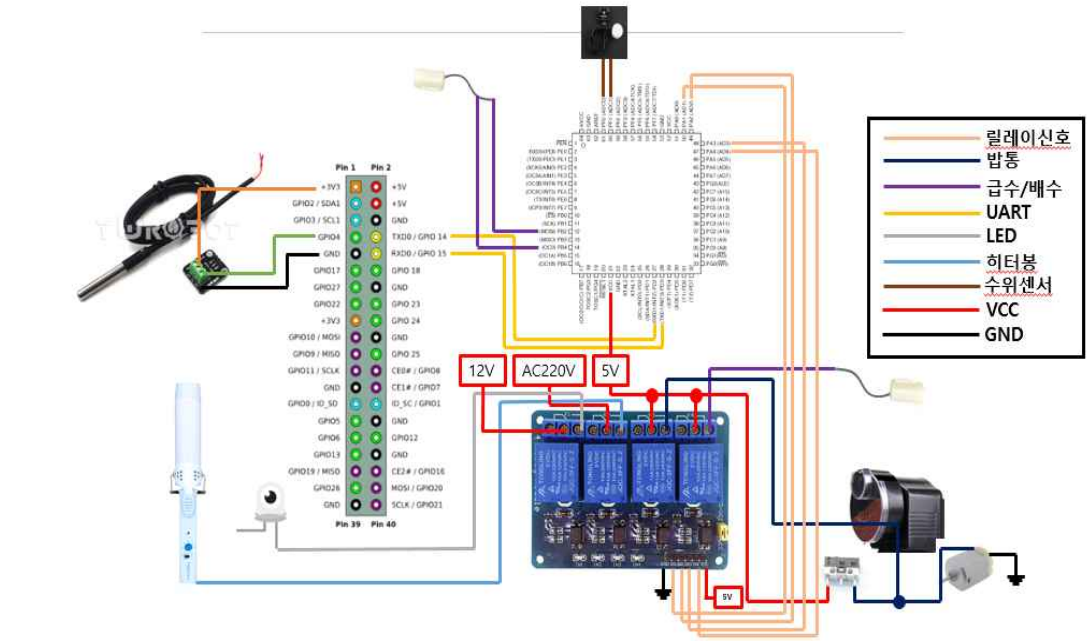
순번	물품 명	수량	단가	총액
1	DS18B20 방수형 온도센서 모듈 (연결용 터미널블럭모듈 포함) [SEN050007]	1	3,600	3,600
2	무접점 수위 센서 모듈 [SEN0204]	2	10,300	20,600
3	아두이노 4채널 5V 릴레이 모듈 [SZH-RLBG-012]	1	2,000	2,000
4	수중펌프모터 [SZH-GNP155]	2	3,800	7,600
5	물고기 자동 급여기	1	8,700	8,700
6	아쿠아플러스 아쿠아스톤수족관바닥재 4KG	2	4,140	8,280
7	고운물 종합 조개망 세트	1	6,230	6,230
8	룸앤홈 조립식 3단 진열장, 블랙	1	27,610	27,610
9	미니 LED 3528 LED바 단색	14	300	5,200
10	미니 LED 3528 LED바 부자재	7	300	2,100
11	실리쿡 투핸드 김치통	2	9,450	18,900
12	모비딕 45 슬림 유리 어항, 45x30x32	1	33,000	33,000
13	JTMOD-128-1	1	27,500	27,500
14	라즈베리파이3 기초 키트(본체+방열판+케이스+아답터+MicroSD+리더기+HDMI&LAN케이블)	1	64,000	64,000
15	[SunFounder] 라즈베리파이 3.5인치 TFT LCD 디스플레이 480x320 터치스크린 모니터 [CN0002]	1	28,600	28,600
총 합 계		38	229,530	263,920

Ⅳ. 제작

1. 구현



<그림 Ⅳ-1> 프로젝트 흐름도



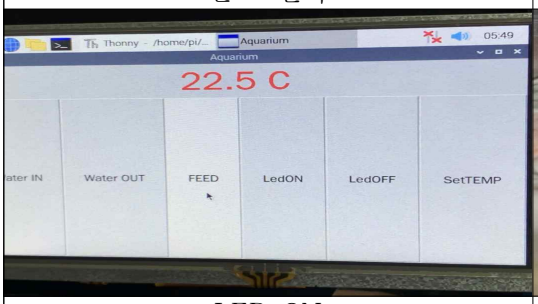
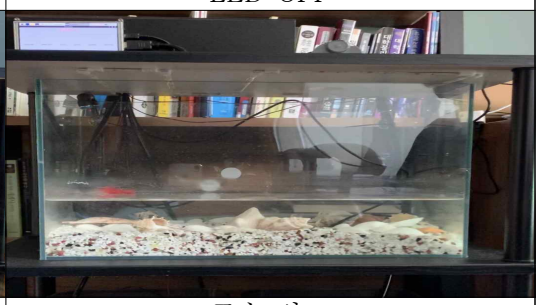
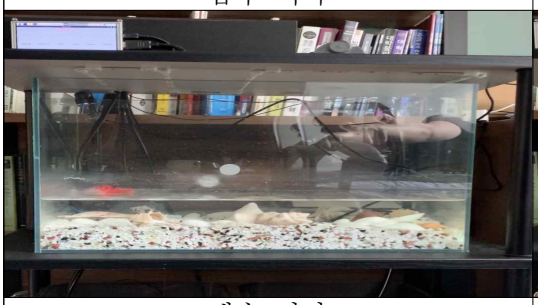
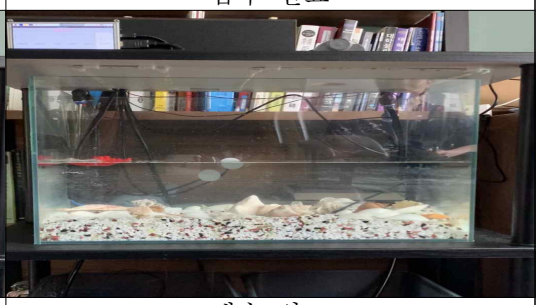
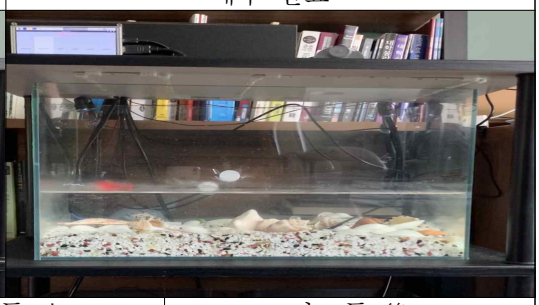
<그림 Ⅳ-2> 프로젝트 회로

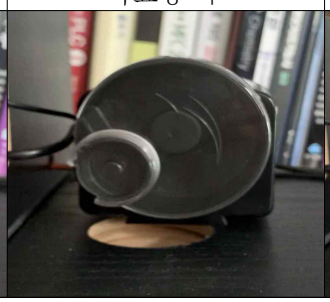
2. 성능평가

- 설정한 온도에 따라 히터가 동작/정지 50번 확인
- LED ON/OFF 버튼 클릭시 LED ON/OFF 50번 확인
- 급수 및 배수시 센서값에 따라 동작/정지 50번 확인
- FEED 버튼 클릭시 한 바퀴만 회전하는지 50번 확인

3. 결과 및 성능

<표 IV-1> 결과 표

온도 센서	히터
	
LED ON	LED OFF
	
급수 시작	급수 완료
	
배수 시작	배수 완료
	

사료통 前	사료통 中	사료통 後
		

Ⅴ. 결론

- Raspberry Pi 와 ATmega128을 연동하여 스마트 수족관을 제작함.
- Raspberry Pi 터치스크린의 버튼을 클릭시 버튼 클릭이 바로 되지않고 3~4초정도 딜레이가 생김
- 남상호 : 졸업 작품이라는 기회를 통하여 Raspberry Pi와 ATmega128을 연동해 보고 처음에는 막막하고 어려웠지만 연동을 완료하고 원하는 동작을 하나씩 완성 할 때 마다 성취감이 생겼고 완성하고 나서 졸업작품을 보니 자신감이 생겼습니다.
- 최영민 : 학교에서 서울 전자전을 견학 한 적이 있습니다. 전자전을 견학하면서 여러 가지 작품들과 기업에서 새롭게 선보이는 제품들이 있었습니다. 그것들을 보면서 느꼈던 것이 지금 현재 생활에 너무 적응하고 생활한 탓에 기술이 이정도로 발전 했었나 라는 놀라움이 있었습니다. 그래서 집으로 돌아와 현재 진행되고 있는 제 4차 산업혁명에 대해 검색을 해 보았습니다. 제 4차 산업혁명은 빅데이터, 인공지능, 사물인터넷 등을 이용해 여러 분야에 활용되어지고 있었습니다. 그 중 사물인터넷 기술을 사용해 평소에 관심이 있었던 반려동물 분야에 접합시켜 졸업 작품을 만들어 보자는 목표가 생겼고 필요하다고 생각했던 점이나 불편했던 점을 직접 개선하고 제작 해 보면서 발전하는 시대에 한걸음 다가가고 전자공학과를 다니는 학생 으로써 미래를 이끌어 나갈 준비를 시작했다는 생각이 들었습니다.

[참고문헌]

- 네이버 블로그

  http://blog.naver.com/PostView.nhn?blogId=gyurse&logNo=220995226526

[부록]

- Raspberry Pi 코드

```
import tkinter
import serial

ser = serial.Serial('/dev/ttyAMA0',9600)
tsensor='/sys/bus/w1/devices/28-03089794462b/w1_slave'

print(ser.portstr)

global a
global b
global c
global d
a=b=c=d='0'

def WaterINClick():
    ser.write(b'A')

def WaterOUTClick():
    ser.write(b'B')

def FeedClick():
    ser.write(b'C')

def LedONClick():
    ser.write(b'D')

def LedOFFClick():
    ser.write(b'E')

def T20():
    global a
```

```
    a = '1'
    if int(settemp())<20:
        ser.write(b'F')


def T23():
    global b
    b = '1'
    if int(settemp())<23:
        ser.write(b'F')

def T25():
    global c
    c = '1'
    if int(settemp())<25:
        ser.write(b'F')

def T27():
    global d
    d = '1'
    if int(settemp())<27:
        ser.write(b'F')

def close_TEMP():
    newwindow.destroy()

def TEMPBOX():
    newwindow = tkinter.Toplevel(window)
    newwindow.geometry("800x480+0+0")
    t20 = tkinter.Button(newwindow, text = "20", font=('Roman, Sans',25),
command = T20)
    t23 = tkinter.Button(newwindow, text = "23", font=('Roman, Sans',25),
command = T23)
    t25 = tkinter.Button(newwindow, text = "25", font=('Roman, Sans',25),
```

```
command = T25)
    t27 = tkinter.Button(newwindow, text = "27", font=('Roman, Sans',25),
command = T27)

    t20.pack(side = "left", fill="both",expand=True)
    t23.pack(side = "left", fill="both",expand=True)
    t25.pack(side = "left", fill="both",expand=True)
    t27.pack(side = "left", fill="both",expand=True)

def traw():
    f=open(tsensor,'r')
    lines=f.readlines()
    f.close()
    return lines

def readtemp():
    lines=traw()
    while lines[0].strip()[-3:]!='YES':
        time.sleep(0.2)
        lines=traw()
    tout=lines[1].find('t=')
    if tout!=-1:
        tstr=lines[1].strip()[tout+2:]
        tc = round(float(tstr)/1000,1)
        space='C'
        return tc,space

def settemp():
    lines=traw()
    while lines[0].strip()[-3:]!='YES':
        time.sleep(0.2)
        lines=traw()
    tout=lines[1].find('t=')
    if tout!=-1:
```

```
            tstr=lines[1].strip()[tout+2:]
            tc = round(float(tstr)/1000,1)

            return tc

def hitteroff():
    global a
    global b
    global c
    global d

    if int(settemp())<23:
        a = '0';

    if int(settemp())>25:
        b = '0';

    if int(settemp())>27:
        c = '0';

    if int(settemp())>30:
        d = '0';

    if a=='0' and b=='0' and c=='0' and d=='0':
        ser.write(b'G')

def on_alarm(top_level_window):
        global var
        var.set(readtemp())
        hitteroff()

        top_level_window.after(1000,on_alarm,top_level_window)

window=tkinter.Tk()
```

```
window.title("Aquarium")
window.resizable(True,True)

var=tkinter.StringVar()
label=tkinter.Label(window,font=('Roman, Sans',40),fg='red',textvariable = var)
label.pack(anchor="nw",fill="x")

WaterIN = tkinter.Button(window, text = "Water IN", font=('Roman, Sans',20),command = WaterINClick)
WaterOUT = tkinter.Button(window, text = "Water OUT", font=('Roman, Sans',20),command = WaterOUTClick)
Feed = tkinter.Button(window, text = "FEED", font=('Roman, Sans',20),command = FeedClick)
LedON = tkinter.Button(window, text= "LedON", font=('Roman, Sans',20),command = LedONClick)
LedOFF = tkinter.Button(window, text= "LedOFF", font=('Roman, Sans',20),command = LedOFFClick)
HITTER = tkinter.Button(window, text= "SetTEMP", font=('Roman, Sans',20),command = TEMPBOX)
WaterIN.pack(side="left",fill="both",expand=True)
WaterOUT.pack(side="left",fill="both",expand=True)
Feed.pack(side="left",fill="both",expand=True)
LedON.pack(side="left",fill="both",expand=True)
LedOFF.pack(side="left",fill="both",expand=True)
HITTER.pack(side="right",fill="both",expand=True)
window.after(1000, hitteroff)
window.after(1000, on_alarm, window)
window.mainloop()
```

- ATmega128 코드

```
#include <mega128.h>
#include <delay.h>
#include <stdio.h>
```

```
#ifndef RXB8
#define RXB8 1
#endif

#ifndef TXB8
#define TXB8 0
#endif

#ifndef UPE
#define UPE 2
#endif

#ifndef DOR
#define DOR 3
#endif

#ifndef FE
#define FE 4
#endif

#ifndef UDRE
#define UDRE 5
#endif

#ifndef RXC
#define RXC 7
#endif

#define FRAMING_ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<DOR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)

#pragma used+
char getchar1(void)
{
```

```
char status,data;
while (1)
        {
        while (((status=UCSR1A) & RX_COMPLETE)==0);
        data=UDR1;
        if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==0)
            return data;
        }
}
#pragma used-

#pragma used+
void putchar1(char c)
{
while ((UCSR1A & DATA_REGISTER_EMPTY)==0);
UDR1=c;
}
#pragma used-

#define FIRST_ADC_INPUT 0
#define LAST_ADC_INPUT 1
unsigned int adc_data[LAST_ADC_INPUT-FIRST_ADC_INPUT+1];
#define ADC_VREF_TYPE 0x00

interrupt [ADC_INT] void adc_isr(void)
{
static unsigned char input_index=0;
adc_data[input_index]=ADCW;
if (++input_index > (LAST_ADC_INPUT-FIRST_ADC_INPUT))
    input_index=0;
ADMUX=(FIRST_ADC_INPUT | (ADC_VREF_TYPE & 0xff))+input_index;
delay_us(10);
ADCSRA|=0x40;
}
void main(void)
{
int LOW;
```

```
int HIGH;
PORTA=0x00;
DDRA=0x1E;

PORTB=0x00;
DDRB=0x14;

UCSR1A=0x00;
UCSR1B=0x18;
UCSR1C=0x06;
UBRR1H=0x00;
UBRR1L=0x67;

UCSR0A=0x00;
UCSR0B=0x18;
UCSR0C=0x06;
UBRR0H=0x00;
UBRR0L=0x67;

ADMUX=FIRST_ADC_INPUT | (ADC_VREF_TYPE & 0xff);
ADCSRA=0xCC;

#asm("sei")

PORTA=0x1E;

while (1)
      {
      LOW = adc_data[0];
      HIGH = adc_data[1];


      printf(" LOW LEVEL : %d , HIGH LEVEL : %d\n\r",LOW,HIGH);

      LOW=adc_data[0];
      HIGH=adc_data[1];
```

```
if(LOW < 50)
{
PORTB=0x00;
}
if(HIGH>950)
{
PORTA.4=0x01;
}

switch(getchar1())
{
case 'A':
   printf("Press A\n\r");
   if(HIGH<900)
   PORTA.4=0x00;
   printf("START Water IN!!\n\r");
   break;

case'B':
   printf("Press B\n\r");
   if(LOW>900)
   {
   PORTB=0x14;

   printf("START Water OUT!!\n\r");
   }
   else
   printf("Little Water\n\r");

   break;

case 'C':
   printf("Press C\n\r");
   PORTA.3=0x00;
   delay_ms(500);
   PORTA.3=0x01;
   printf("FEED!!\n\r");
```

```
            break;

        case'D':
            printf("Press D\n\r");
            PORTA.1=0x00;
            printf("LED ON!!\n\r");

            break;


        case 'E':
            printf("Press E\n\r");
            PORTA.1=0x01;
            printf("LED OFF!!\n\r");

            break;

        case 'F':
            printf("Press E\n\r");
            PORTA.2=0x00;
            printf("HITTER ON !!\n\r");

            break;

        case 'G':
            printf("Press G\n\r");
            PORTA.2=0x01;
            printf("HITTER OFF!!\n\r");
            break;
        }
        delay_ms(2000);
        }
}
```