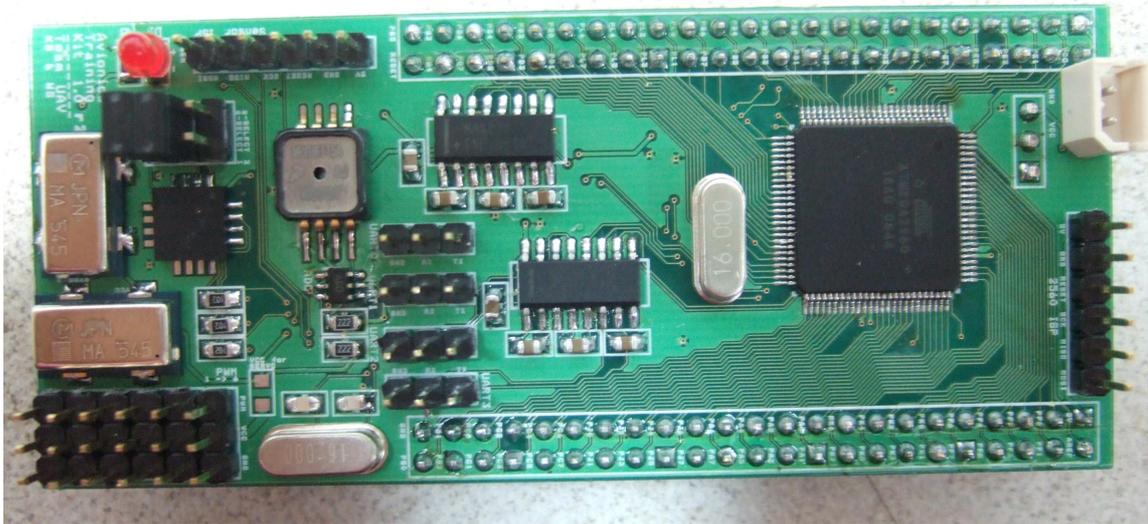


DATA SHEET



ATK - Avionics Training Kit

Document Revision History

Revision Date	Document Version	Pages	Description
2009-03-31	1.0.1	All	Initial release by Woosung, Lee. and Keunbum, Park.

☞ 1. Document Revision History

목차

1. 개요	6
1.1. 본 문서에 대해서	6
1.2. 대상	6
1.3. 문서번호	6
1.4. 기술지원	6
2. ATK 사양	6
2.1. 외형	7
2.2. ATK 내부 Diagram	9
2.3. 적용 부품	10
2.3.1. 전원부	10
2.3.2. 사용자 영역	10
2.3.3. 시리얼 포트	10
3. ATK 기본 사용방법	10
3.1. 전원 연결	10
3.1.1. 전원 단자 위치	10
3.1.2. 전원 연결	11
3.2. ISP 연결	11
3.2.1. ISP 단자 위치	11
3.2.2. ISP 단자 순서	12
3.3. 사용자 영역 접근	12
3.3.1. 권장 컴파일러	13
3.3.2. ATmega2560 연결 확인	13
4. ATK 기능 사용방법	13
4.1. 시리얼 연결	13
4.1.1. 시리얼 포트 위치	13
4.1.2. 시리얼 포트 사용 방법	14
4.1.2.1. 헤더파일	14
4.1.2.2. 외부 장치와의 연결	16
4.2. PWM 출력	16
4.2.1. PWM 커넥터 위치	16
4.2.2. PWM 커넥터 전원 연결 방법	17
4.2.3. PWM 제어 방법	17
4.3. 외부 핀 사용	20
4.3.1. 외부 핀 연결 회로	20
4.4. 센서 데이터 사용	22
4.4.1. 데이터 프로토콜	22
4.4.2. 센서데이터 수신	22
5. UAV(Unmanned Aero Vehicle;무인비행기) 적용 사례	22
5.1. Platform	22
5.1.1. 부품 선정	22
5.1.1.1. 기체	22
5.1.1.2. 송/수신기	23

5.1.1.3. 서보모터	23
5.1.1.4. 모터/변속기/프로펠러	24
5.1.1.5. 배터리	24
5.1.2. 개조	25
5.1.2.1. 에일러론 조종면 추가	25
5.1.2.2. 브러시리스 모터 장착	26
5.1.2.3. 방진 마운트	27
5.1.3. 플랫폼 개별 부품 소모 비용	28
5.2. 스위치	28
5.3. 시스템 전체 구성	29
5.4. 장착 모습	30
5.5. 사용자 영역 코드	30
5.6. 비행 영상	32
6. 개발팀 Comment	32
6.1. 이우성	32
6.2. 박근범	33

<표 차례>

표 1. Document Revision History	2
표 2 . 크기	9
표 3 . ISP 단자 순서	12
표 4 . 센서데이터 시리얼 포트 설정	22
표 5 . 모터/변속기/프로펠러 사용 제품	24
표 6 . 개별 부품 소모 비용	28

<그림 차례>

그림 1 . ATK 1.0	1
그림 2 . ATK 앞면	7
그림 3 . ATK 뒷면	8
그림 4 . 크기	9
그림 5 . ATK Diagram	10
그림 6 . 전원 단자 위치	11
그림 7 . ISP 단자 위치	12
그림 8 . 시리얼포트 위치	14
그림 9 . PWM 커넥터 위치	16
그림 10 . PWM 전원 패드	17
그림 11 . 외부 핀 연결도	21
그림 12 . Easy Star	23
그림 13 . 송수신기, Futaba 6EX 2,4 T/R Set	23
그림 14 . HS-55	23
그림 15 . 브러시리스 모터	24
그림 16 . 프로펠러	24
그림 17 . Li-Po 배터리	25
그림 18 . 에일러론 서보모터	26

그림 19 . 에일러론 힌지26
그림 20 . 개조된 에일러론26
그림 21 . 모터 마운트27
그림 22 . 마운트 윗면27
그림 23 . 마운트 밑면28
그림 24 . 스위치29
그림 25 . 스위치 크기 비교29
그림 26 . 비행 전 전체 시스템 블럭도29
그림 27 . 비행 전 장착 모습30
그림 28 . 비행 전 장착 모습30

1. 개요

본 문서에 대한 설명이다.

1.1. 본 문서에 대해서

이 문서는 1)ATK-1.0 에 대한 전반적인 사양, 내부 센서 데이터에 대한 설명, 사용방법 등에 대한 설명을 포함하고 있다.

1.2. 대상

이 문서는 ATK를 이용하려는 사람을 대상으로 한다.

1.3. 문서번호

이 문서의 버전관리는 다음과 같다.

< 첫 번째 자리 >

Main 업데이트. ATK 모듈의 큰 기능 변화가 있을 때 업데이트 된다.

< 두 번째 자리 >

Minor 업데이트. ATK 모듈의 작은 기능 변화가 있을 때 업데이트 된다. 홀수의 경우에는 문서의 내용에 있어 불확실한 경우가 있을 수 있다. 짝수의 경우에는 문서의 내용이 모두 확인되고 테스트가 끝난 상태이다.

< 세 번째 자리 >

수시 업데이트. 문서내용의 모든 변화가 포함된다.

1.4. 기술지원

이 문서의 모든 내용에 대한 기술 문의는 다음의 인원에 의하여 이루어진다.
문의 방법은 이메일, 블로그를 통해서 이루어진다.

< 박근범 >

센서데이터에 대한 문의

UAV 적용사례 .c 코드에 대한 문의

asd441@naver.com

<http://blog.naver.com/asd441>

< 이우성 >

개발 보드 회로에 대한 문의

UAV 적용사례 .h 코드에 대한 문의

email.woosung@gmail.com

<http://leewoosung.net>

2. ATK 사양

이 장에서는 ATK의 사양을 설명한다.

1) 이후 ATK라 함은 ATK-1.0을 의미한다.

2.1. 외형

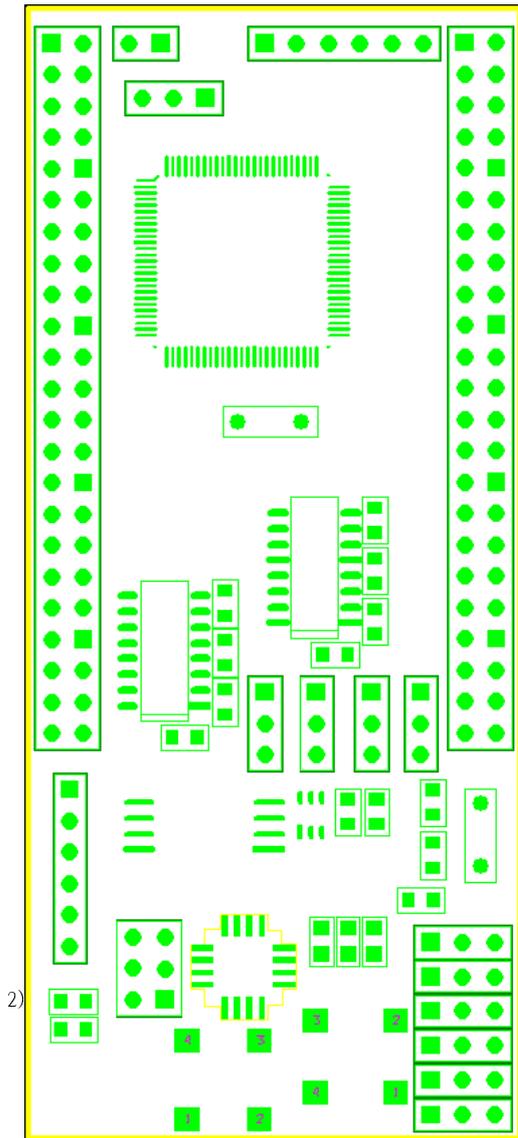


그림 2 . ATK 앞면

2) 그림 1. 앞면 은 이후의 설명에서 특정 장치의 위치를 설명하는데 쓰인다.

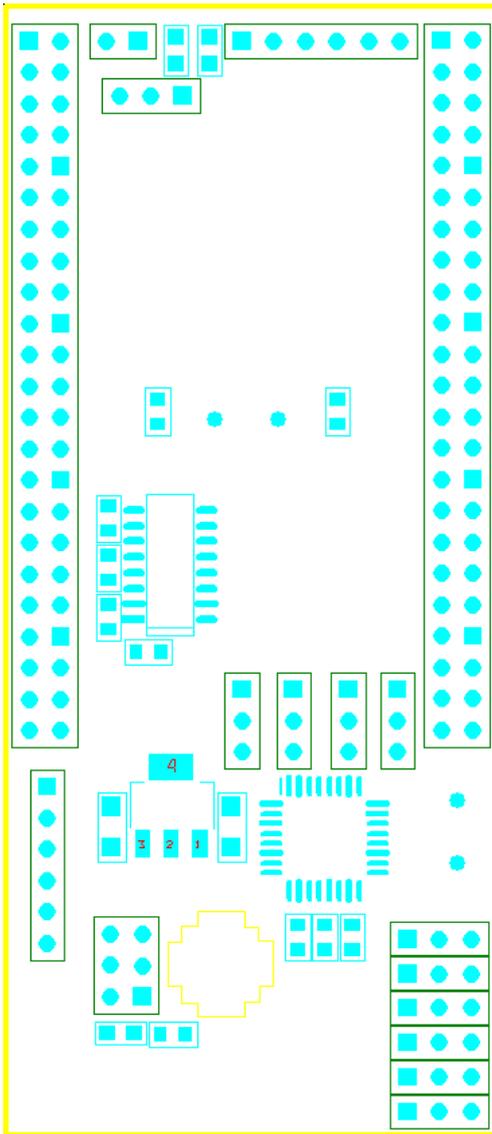


그림 3 . ATK 뒷면

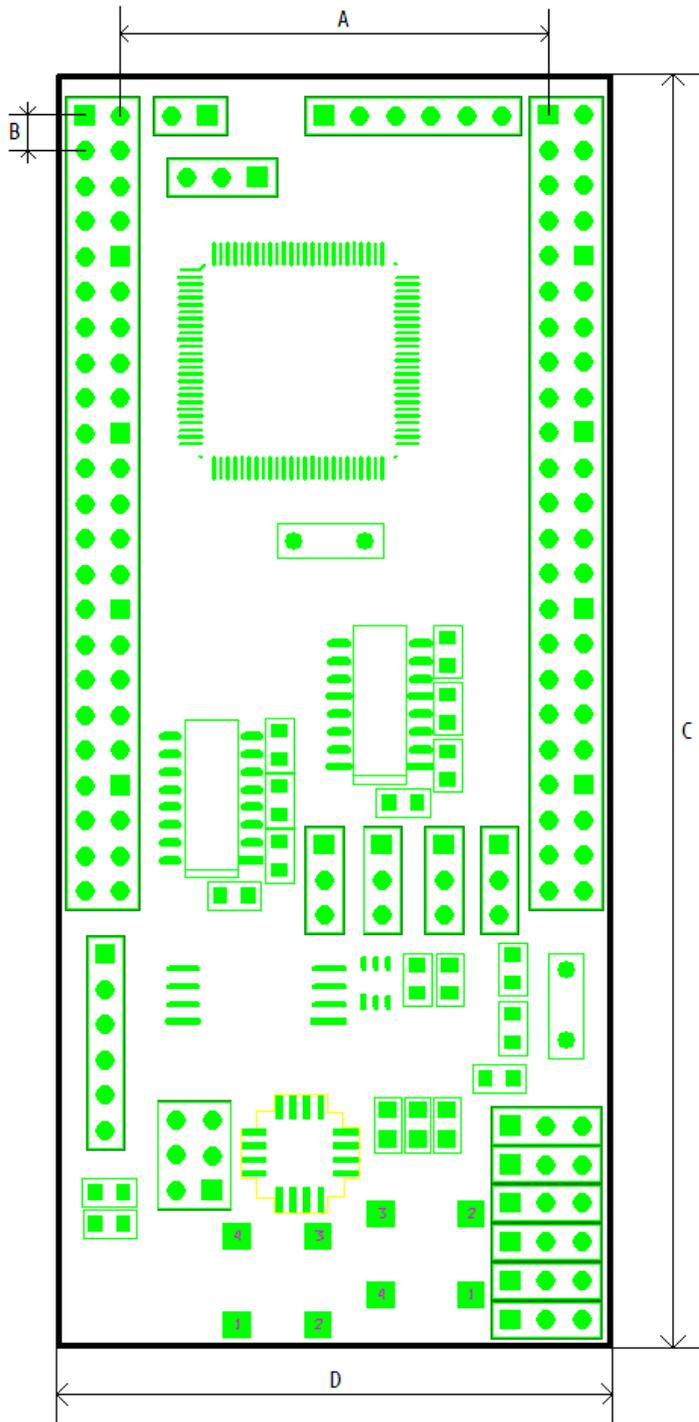


그림 4 . 크기

마크	크기 (mm)	설명
A	30.48	양쪽 외부 핀 간격
B	2.54	일반 핀 간격
C	91.24	세로 길이
D	39.15	가로 길이

표 2 . 크기

2.2. ATK 내부 Diagram

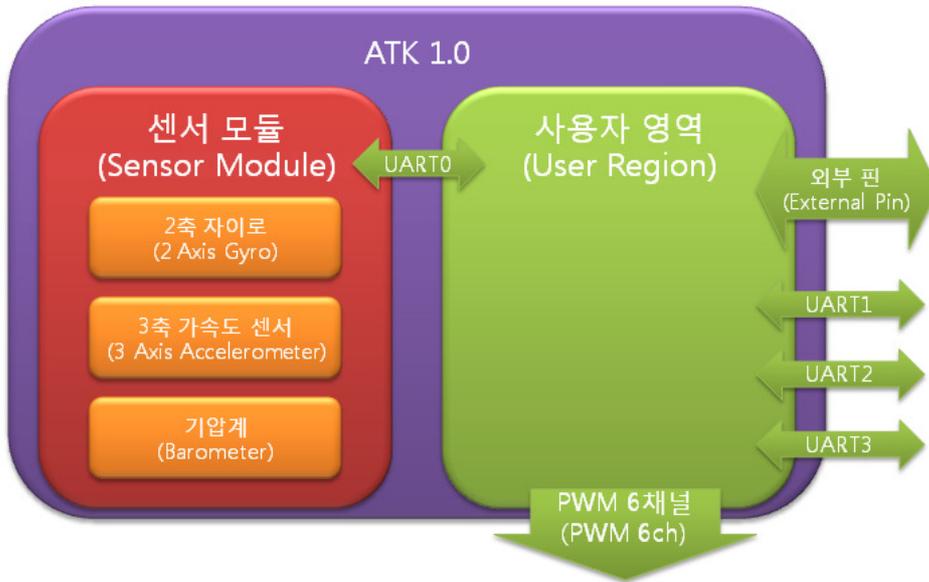


그림 5 . ATK Diagram

센서모듈부분은 사용자가 임의로 접근할 수 없다. 센서모듈로부터 나오는 정보는 사용자 영역에서의 UART0 포트를 이용하여 접근할 수 있다. 사용자 영역에서의 ATmega2560의 모든 핀은 외부의 핀으로 나와있기 때문에 사용자가 임의로 추가 모듈을 제작하여 사용할 수 있다.

2.3. 적용 부품

2.3.1. 전원부

5V 리니어 레귤레이터, 78T05 시리즈

사용 가능한 부품은 다음의 링크에서 참조 가능하다.

<http://www.alldatasheet.co.kr/view.jsp?Searchword=78T05>

2.3.2. 사용자 영역

ATmega2560

16MHz 오실레이터로부터 클럭을 공급받고 있다.

2.3.3. 시리얼 포트

MAX232를 이용하여 RS-232 레벨 출력의 4개 포트를 포함하고 있다. 이 중 UART0는 센서모듈로부터 사용자 영역으로의 데이터 전송을 위해 사용되고 있다.

3. ATK 기본 사용방법

이 장에서는 ATK의 기본 사용방법을 설명한다.

3.1. 전원 연결

3.1.1. 전원 단자 위치

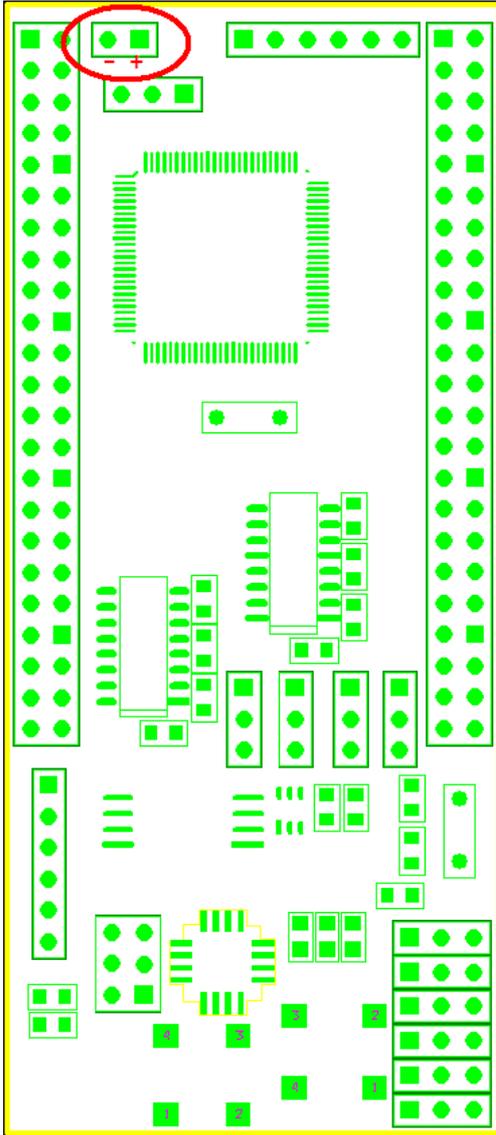


그림 6 . 전원 단자 위치

ATK의 전원 단자는 MOLEX 5264-02 커넥터를 사용한다. 극성은 윗면에서 보았을 때 왼쪽이 GND, 오른쪽이 VCC이다. ATK 윗면에 GND와 VCC의 위치가 쓰여 있다.

3.1.2. 전원 연결

ATK는 5V 레귤레이터를 내장하고 있으므로 사용된 레귤레이터의 허용범위 내의 전원을 인가하면 된다. On Semiconductor 사의 MC7805CT 모델이 장착된 경우 5V~18V가 가능하다.

3.2. ISP 연결

3.2.1. ISP 단자 위치

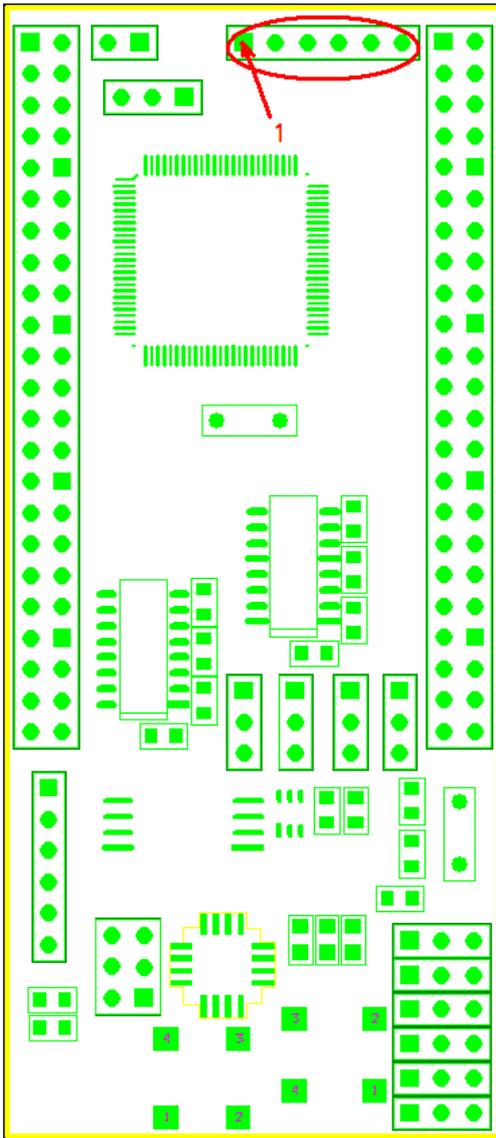


그림 7 . ISP 단자 위치

3.2.2. ISP 단자 순서

ISP 단자는 왼쪽 1번 핀부터 다음의 순서로 되어 있다.

번호	명칭
1	5V
2	GND
3	RESET
4	SCK
5	MISO
6	MOSI

표 3 . ISP 단자 순서

NewTech 사의 ³⁾AD-USBISP V02-L와 같은 제품을 사용하면 된다.

3.3. 사용자 영역 접근

3) 이후 ISP라 함은 AD-USBISP V02 를 의미한다.

이 장에서는 사용자 영역에 접근하는 것에 대한 설명한다.

3.3.1. 권장 컴파일러

권장되는 컴파일러는 4)CodeVision 2.03.4 버전이다. AVR Studio를 사용해도 무방하다.

3.3.2. ATmega2560 연결 확인

CodeVision의 메뉴에서 [Settings] - [Programmer] 를 실행한다.

Programmer Type은 [Atmel STK500/AVRISP] 로 지정하고, [내 컴퓨터]-[속성]-[장치관리자]에서 ISP가 인식된 Port 번호를 찾아 Communication Port 번호를 지정한다.

CodeVision의 메뉴에서 [Tools] - [Chip Programmer] 를 실행한다.

Chip을 ATmega2560으로 선택하고, [Read]-[Chip Signature] 를 실행한다. ATmega2560으로 인식하면 정상적으로 연결된 것이다. 만일 Chip Signature를 잘 읽지 못 하면 SCK Freq.를 좀더 낮은 수치로 바꾸어 시도해 본다.

4. ATK 기능 사용방법

이 장에서는 ATK의 기능을 사용하는 방법에 대해서 설명한다. 설명에 사용된 컴파일러는 CodeVision 이다. CodeVision의 CodeWizard 기능을 이용하여 생성된 코드를 기본으로 하며 설명에 필요한 추가 코드는 각 절에 첨부한다.

4.1. 시리얼 연결

ATK는 총 4개의 시리얼 포트를 제공한다. 이 중 1번은 센서보드와의 통신을 위해 사용되고 있고, 나머지 3개의 시리얼 포트가 사용자가 직접 사용할 수 있는 부분이다.

4.1.1. 시리얼 포트 위치

4) 이후 CodeVision 이라 함은 2.03.4 버전을 의미한다.

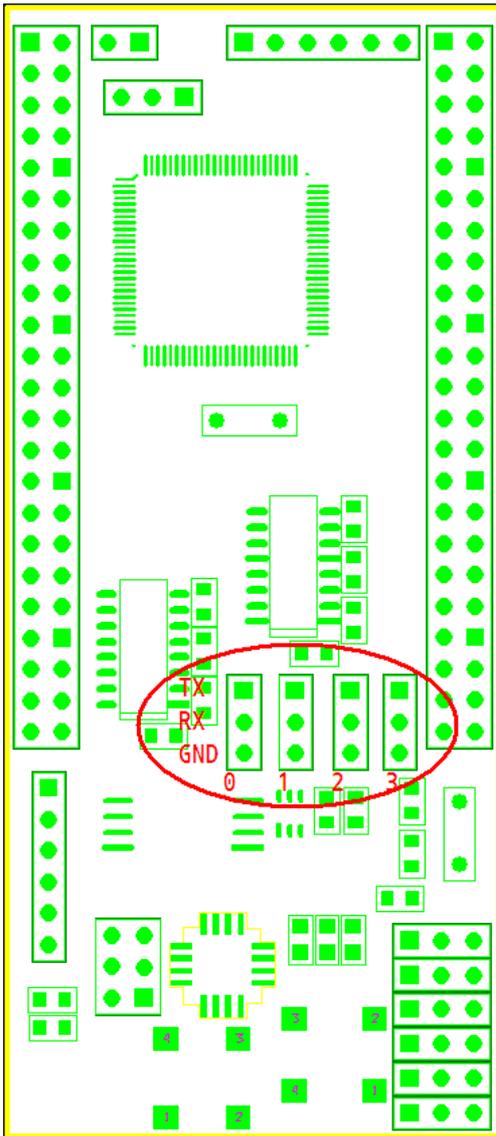


그림 8 . 시리얼포트 위치

4.1.2. 시리얼 포트 사용 방법

4.1.2.1. 헤더파일

다음의 헤더파일을 이용하여 시리얼 포트를 사용할 수 있다. 기재된 내용은 CodeWizard에서 UART 관련 설정을 가져온 것이다.

```
// uart.h
// Standard Input/Output functions
#include <stdio.h>

#define RXB8 1
#define TXB8 0
#define UPE 2
#define OVR 3
#define FE 4
#define UDRE 5
#define RXC 7

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

```

void Init_uart (void) {
    //baudrate 및 interrupt/polling 선택 등 USART 동작 방식에 대한
    //관련 레지스터 사용법은 데이터시트 p.223, 22.9 Register Description에 나와있다.

    // USART0 initialization
    UCSR0A=0x00;
    UCSR0B=0x18;
    UCSR0C=0x06;
    UBRR0H=0x00;
    UBRR0L=0x10;

    // USART1 initialization
    UCSR1A=0x00;
    UCSR1B=0x18;
    UCSR1C=0x06;
    UBRR1H=0x00;
    UBRR1L=0x08;

    // USART2 initialization
    UCSR2A=0x00;
    UCSR2B=0x18;
    UCSR2C=0x06;
    UBRR2H=0x00;
    UBRR2L=0x67;

    // USART3 initialization
    UCSR3A=0x00;
    UCSR3B=0x18;
    UCSR3C=0x06;
    UBRR3H=0x00;
    UBRR3L=0x67;
}

// Get a character from the USART1 Receiver
#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-

//// Write a character to the USART1 Transmitter
#pragma used+
void putchar1(char c)
{
    while ((UCSR1A & DATA_REGISTER_EMPTY)==0);
    UDR1=c;
}
#pragma used-

// Get a character from the USART2 Receiver
#pragma used+
char getchar2(void)
{
    char status,data;
    while (1)
    {
        while (((status=UCSR2A) & RX_COMPLETE)==0);
        data=UDR2;
        if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==0)
            return data;
    };
}
#pragma used-

// Write a character to the USART2 Transmitter
#pragma used+
void putchar2(char c)
{
    while ((UCSR2A & DATA_REGISTER_EMPTY)==0);
    UDR2=c;
}
#pragma used-

// Get a character from the USART3 Receiver
#pragma used+
char getchar3(void)
{

```

```

char status,data;
while (1)
{
    while (((status=UCSR3A) & RX_COMPLETE)==0);
        data=UDR3;
    if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==0)
        return data;
};
}
#pragma used-

// Write a character to the USART3 Transmitter
#pragma used+
void putchar3(char c)
{
    while ((UCSR3A & DATA_REGISTER_EMPTY)==0);
        UDR3=c;
}
#pragma used-
    
```

4.1.2.2. 외부 장치와의 연결

ATK에서 제공하는 시리얼 포트는 MAX232를 사용하여 RS-232 레벨(시그널 레벨이 ±6V)로 변환된 뒤에 출력된다. 따라서 PC에서 시리얼 포트를 사용하고자 할 때에는 추가 장비 없이 바로 연결하면 된다. 외부 모듈을 부착하려는데 TTL/CMOS 레벨 (시그널이 0~5V레벨)일 경우에는 외부 핀 중 RX/TX 핀을 사용하면 된다.

시리얼 포트는 왼쪽에서부터 0, 1, 2, 3번 순서로 배열되어 있으며, 위쪽에서부터 RX, TX, GND 순서로 배열되어 있다.

4.2. PWM 출력

ATK는 6개의 PWM 출력 핀을 가지고 있다. 사용자는 PWM 커넥터에 연결될 외부장치의 5V 전원을 ATK로부터 공급할 수 있다. 출하 상태의 ATK는 PWM 커넥터로 PWM 시그널만을 공급하고 5V 전원은 공급하지 않는다.

4.2.1. PWM 커넥터 위치

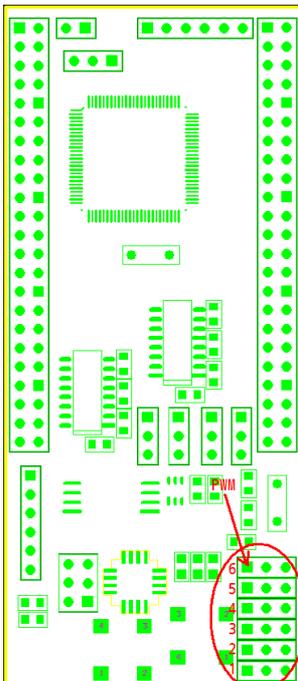


그림 9 . PWM 커넥터 위치

아래쪽에서부터 PWM 1번부터 6번까지 배열되어 있다. 가장 왼쪽 핀이 PWM 출력 핀이고 가운데 핀은 VCC, 가장 오른쪽 핀은 GND이다.

4.2.2. PWM 커넥터 전원 연결 방법

ATK의 PWM 커넥터는 가운데가 VCC 핀이다. 이 핀을 통해 외부 장치에 전원을 공급하려면 [VCC for Servo] 패드를 서로 연결해주어야 한다. 해당 패드의 위치는 다음과 같다.

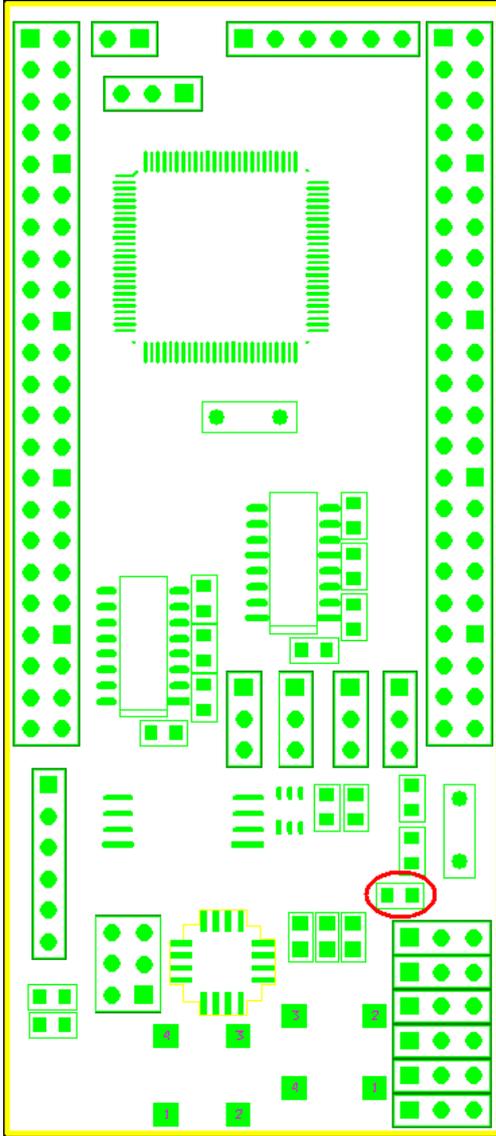


그림 10 . PWM 전원 패드

표시된 패드를 서로 연결시켜주면 ATK의 5V 전원을 이용하여 외부에 전원을 공급해주게 된다.⁵⁾

4.2.3. PWM 제어 방법

다음의 헤더파일을 이용하여 PWM 커넥터를 사용할 수 있다. 기재된 내용은 CodeWizard에서 Timer 관련 설정을 가져온 것이다.

5) 외부장치가 서보모터일 경우, 서보모터가 동작 시 많은 전류를 소모하여 ATK가 순간적으로 리셋되는 경우가 있으므로, 꼭 필요한 경우가 아니라면 사용하지 않기를 권한다.

```
// pwm.h
// -----
// PWM 관련 타이머 셋팅
// -----

void Init_pwm(void) {
    // Timer/Counter 0 initialization
    TCCR0A=0x00;
    TCCR0B=0x00;
    TCNT0=0x00;
    OCR0A=0x00;
    OCR0B=0x00;
    DDRB = DDRB | 0xE0;

    // Timer/Counter 1 initialization
    TCCR1A=0xA8;
    TCCR1B=0x12;
    TCNT1H=0x00;
    TCNT1L=0x00;
    ICR1H=0x4E;
    ICR1L=0x20;
    OCR1AH=0x03;        //high 8bit of OCR1A
    OCR1AL=0xE8;        //low 8bit of OCR1A
    OCR1BH=0x03;        //high 8bit of OCR1A
    OCR1BL=0xE8;        //low 8bit of OCR1A
    OCR1CH=0x03;        //high 8bit of OCR1A
    OCR1CL=0xE8;        //low 8bit of OCR1A

    // Timer/Counter 2 initialization
    ASSR=0x00;
    TCCR2A=0x00;
    TCCR2B=0x00;
    TCNT2=0x00;
    OCR2A=0x00;
    OCR2B=0x00;
    DDRE = DDRE | 0x38;

    // Timer/Counter 3 initialization
    TCCR3A=0xA8;
    TCCR3B=0x12;
    TCNT3H=0x00;
    TCNT3L=0x00;
    ICR3H=0x4E;
    ICR3L=0x20;
    OCR3AH=0x03;
    OCR3AL=0xE8;
    OCR3BH=0x03;
    OCR3BL=0xE8;
    OCR3CH=0x03;
    OCR3CL=0xE8;

    // Timer/Counter 4 initialization
    TCCR4A=0x00;
    TCCR4B=0x00;
    TCNT4H=0x00;
    TCNT4L=0x00;
    ICR4H=0x00;
    ICR4L=0x00;
    OCR4AH=0x00;
    OCR4AL=0x00;
    OCR4BH=0x00;
    OCR4BL=0x00;
    OCR4CH=0x00;
    OCR4CL=0x00;

    // Timer/Counter 5 initialization
    TCCR5A=0x00;
    TCCR5B=0x00;
    TCNT5H=0x00;
    TCNT5L=0x00;
    ICR5H=0x00;
    ICR5L=0x00;
    OCR5AH=0x00;
    OCR5AL=0x00;
    OCR5BH=0x00;
    OCR5BL=0x00;
    OCR5CH=0x00;
    OCR5CL=0x00;

    // Timer/Counter 0 Interrupt(s) initialization
    TIMSK0=0x00;
    // Timer/Counter 1 Interrupt(s) initialization
    TIMSK1=0x00;
    // Timer/Counter 2 Interrupt(s) initialization
    TIMSK2=0x00;
```

```
// Timer/Counter 3 Interrupt(s) initialization
TIMSK3=0x00;
// Timer/Counter 4 Interrupt(s) initialization
TIMSK4=0x00;
// Timer/Counter 5 Interrupt(s) initialization
TIMSK5=0x00;
}

// -----
// PWM 관련 함수
// -----

unsigned long int AngleToPWM (int angle) {
    //angle 허용 범위
    // 0 <= angle <=90
    if ( angle < 0 )
        angle = 0;
    else if ( angle > 90 )
        angle = 90;

    return (angle*111)/10 +1000;
}

void Servo_1(int angle) {
    unsigned long int ms;
    unsigned long int msHigh;
    unsigned long int msLow;

    ms = AngleToPWM(angle);
    msHigh = (ms & 0xFF00) >> 8;
    msLow = ms & 0x00FF;

    OCR1AH=msHigh;
    OCR1AL=msLow;
}

void Servo_2(int angle) {
    unsigned long int ms;
    unsigned long int msHigh;
    unsigned long int msLow;

    ms = AngleToPWM(angle);
    msHigh = (ms & 0xFF00) >> 8;
    msLow = ms & 0x00FF;

    OCR1BH=msHigh;
    OCR1BL=msLow;
}

void Servo_3(int angle) {
    unsigned long int ms;
    unsigned long int msHigh;
    unsigned long int msLow;

    ms = AngleToPWM(angle);
    msHigh = (ms & 0xFF00) >> 8;
    msLow = ms & 0x00FF;

    OCR1CH=msHigh;
    OCR1CL=msLow;
}

void Servo_4(int angle) {
    unsigned long int ms;
    unsigned long int msHigh;
    unsigned long int msLow;

    ms = AngleToPWM(angle);
    msHigh = (ms & 0xFF00) >> 8;
    msLow = ms & 0x00FF;

    OCR3AH=msHigh;
    OCR3AL=msLow;
}

void Servo_5(int angle) {
    unsigned long int ms;
    unsigned long int msHigh;
    unsigned long int msLow;

    ms = AngleToPWM(angle);
    msHigh = (ms & 0xFF00) >> 8;
    msLow = ms & 0x00FF;

    OCR3BH=msHigh;
    OCR3BL=msLow;
}
```

```
void Servo_6(int angle) {
    unsigned long int ms;
    unsigned long int msHigh;
    unsigned long int msLow;

    ms = AngleToPWM(angle);
    msHigh = (ms & 0xFF00) >> 8;
    msLow = ms & 0x00FF;

    OCR3CH=msHigh;
    OCR3CL=msLow;
}
```

4.3. 외부 핀 사용

ATK는 ATmega2560의 XTAL1과 XTAL2를 제외한 모든 핀을 외부 핀으로 제공한다. 사용자는 외부 핀을 이용하여 일반 ATmega2560 Kit를 사용하듯이 ATK를 사용할 수 있다.

4.3.1. 외부 핀 연결 회로

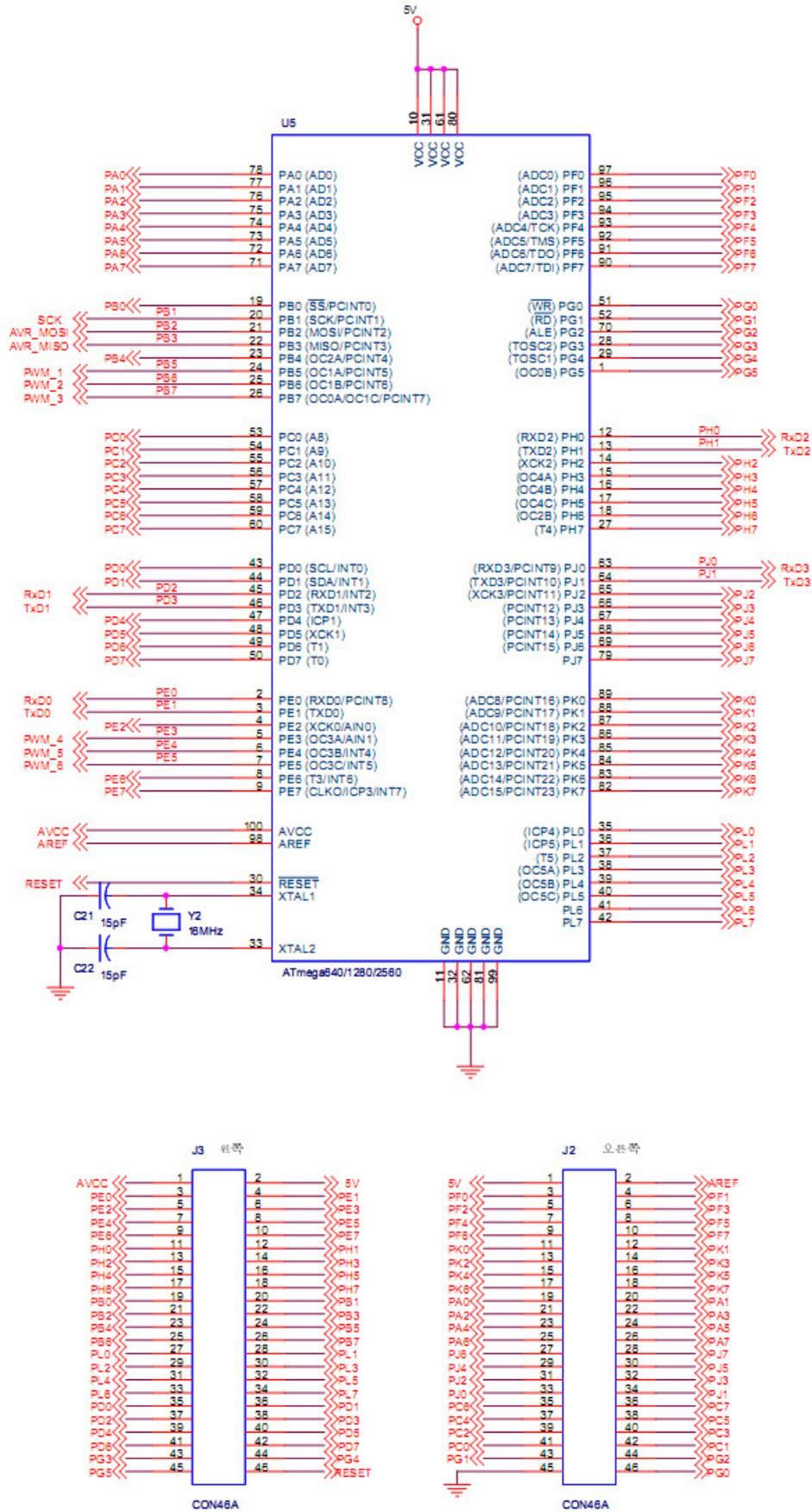


그림 11 . 외부 핀 연결도

4.4. 센서 데이터 사용

ATK는 내부의 센서 모듈로부터 사용자 영역으로 현재 자세와 기압에 대한 정보를 제공한다.

4.4.1. 데이터 프로토콜

센서데이터는 다음과 같은 프로토콜을 따른다.

```
//프로토콜 정의
RP[Roll] [Pitch] [Barometer]
```

실제 예는 다음과 같다.

```
//데이터 수신 예
RP23 11 32567
```

4.4.2. 센서데이터 수신

센서데이터는 UART0를 통해서 사용자 영역으로 들어온다. 사용자는 UDR0 (UART0 의 수신 데이터 레지스터)를 확인하는 등의 방법으로 센서데이터를 읽어들이 수 있다.

시리얼포트의 설정은 다음과 같다.

구분	값
Baudrate	57600
Data bits	8
Parity	None
Stop bits	1

표 4 . 센서데이터 시리얼 포트 설정

5. UAV(Unmanned Aero Vehicle;무인비행기) 적용 사례

ATK를 무인비행기에 적용한 사례이다. 여기서는 일정 고도에서 자동 모드로 변경 시에 ATK가 기체의 Roll 각을 제어하여 선회하는 Bank-Turn을 구현하는 예제를 선보인다.

5.1. Platform

5.1.1. 부품 선정

5.1.1.1. 기체

어떤 프로젝트이든지 실패 없이 성공하는 경우는 없다. 그런데 무인기 프로젝트의 경우, 단 한번의 실패는 기체 파손이라는, 실패에 대한 댓가가 너무 큰 것이 문제이다. 따라서 초심자에게는 기체 자체로서 안정성을 가지는 상반각이 있는 고익기를, 개/보수가 쉬운 EPP재질로 선택하는 것을 권장한다.

본 적용 사례에서는 Multiplex 사의 EasyStar를 사용하였다.



그림 12 . Easy Star

5.1.1.2. 송/수신기

무인기 프로젝트는 수동/자동 조종을 안정적으로 전환할 수 있는 시스템 구축이 필수적이다. 이를 위해서 송수신기의 채널수 개수가 넉넉하고 주파수 혼선을 최대한 피할 수 있는 제품을 사용하는 것을 권장한다.

본 적용 사례에서는 FUTABA 6EX 2.4 T/R Set를 사용하였다.



그림 13 . 송수신기. Futaba 6EX 2.4 T/R Set

5.1.1.3. 서보모터

RC항공기용 서보모터로는 크기가 작고, 무게는 가볍고, 추종속도가 빠르고, 중립점이 정확해야 한다. 부가적으로 충격에 강한 메탈기어일 경우 더욱 좋다.

본 적용 사례에서는 HS-55를 사용하였다.



그림 14 . HS-55

5.1.1.4. 모터/변속기/프로펠러

일반 RC용으로는 추력이 어느 정도까지 만이라도 좋지만 무인항공기 연구에 있어서는 모듈의 무게도 고려해야 하므로 가벼우면서도 추력이 센 제품을 선택하는 것을 권장한다.

본 적용 사례에서는 다음과 같은 부품들을 사용하였다.

구분	제 품
모 터	E-MAX CF2812(Kv1534)
변속기	E-Max 35A V2
프 롬	APC 6X5.5E 전동용 Prop
프롬아답터	프롬 아답터(모터축3.0mm, 프롬축4.6mm)

표 5 . 모터/변속기/프로펠러 사용 제품



그림 15 . 브리시리스 모터



그림 16 . 프로펠러

!!!주의!!!

RC용 부품이지만 프로펠러는 동작 중에는 예초기에 준하는 위험을 가지고 있다. 따라서 모터 마운트가 확실히 고정되어 있지 않다면 절대 구동을 삼가야 하고, 절대로 프로펠러의 회전방향에는 사람이 있어서는 안 된다.

5.1.1.5. 배터리

가벼우면서도 방전율이 좋은 조건을 만족시키려면 Li-Po (리튬 폴리머) 배터리를 선택하는 것이 좋다. 다만 관리가 까다로운데, 다음의 관리 조건을 준수해야 한다.

1) 과방전에 주의한다.

과방전시 배터리 사용이 불가능할 수 있다.

2) 전용충전기를 사용한다.

2~3개 셀을 합친 것을 많이 사용하는데, 이 때 각 셀의 전압을 안정적으로 밸런싱 해주면서 충전해야 한다.

3) 전극이 붙는 것을 주의한다.

방전율이 좋은 배터리라서 전극이 말랐을 때 많은 전류가 도선을 타고 흐르면서 도선이 과열로 끊어지거나 배터리가 손상될 수 있다.

4) 장기 보관이 예상되면 스토리지 모드로 충전상태를 조절한다.

자연 방전이나 완충 상태에서 배터리 효율이 나빠지는 것을 미연에 방지한다.

5) 사용횟수가 많아지면 배터리 성능을 확인한다.

약 50회 이상 충/방전을 거치면 실제로 성능이 떨어지게 된다.



그림 17 . Li-Po 배터리

5.1.2. 개조

5.1.2.1. 에일러론 조종면 추가

자세를 수정하는데 있어서 러더보다는 에일러론을 조작하는 것이 직관적인데, 이지스타는 에일러론이 없는 러더기이다. 따라서 러더를 고정시키고 에일러론을 추가하는 작업을 진행한다.

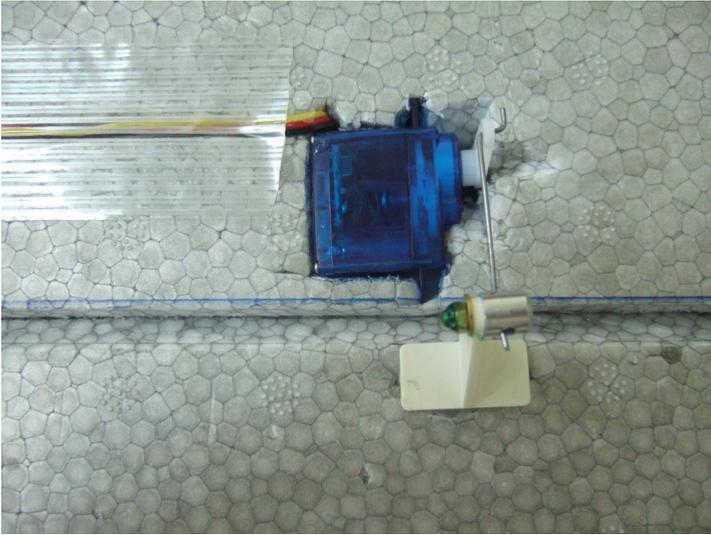


그림 18 . 에일러론 서보모터



그림 19 . 에일러론 힌지



그림 20 . 개조된 에일러론

5.1.2.2. 브러시리스 모터 장착

이지스타는 DC모터가 기본 패키지로 들어있으나, 추력이 약하기 때문에 브러시리스 모터를 사용할 수 있도록 개조해야 한다. 따라서 브러시리스 모터를 사용할 수 있는 모터 마운트를 만들고 기체에 단단히 고정시키는 작업을 진행한다.

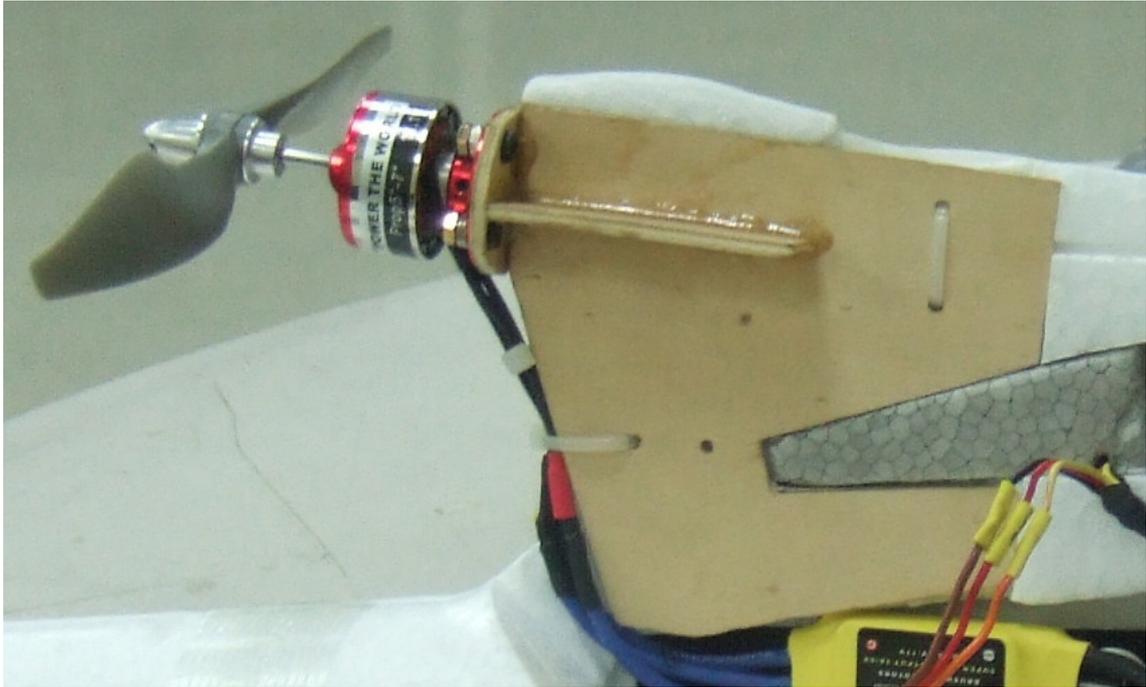


그림 21 . 모터 마운트

5.1.2.3. 방진 마운트

전자시스템을 기체에 부착할 때에는 EMI 현상과 진동을 고려해야 한다. 마운트 또한 최대한 가볍게 제작을 해야 하므로 폼백스를 이용하여 틀을 만들고 고무줄을 이용하여 방진처리를 하였다. 이런 방식은 FCC6)가 가벼울 때만 가능하다.



그림 22 . 마운트 윗면

6) Flight Control Computer, 여기서는 ATK가 그 역할을 수행한다.

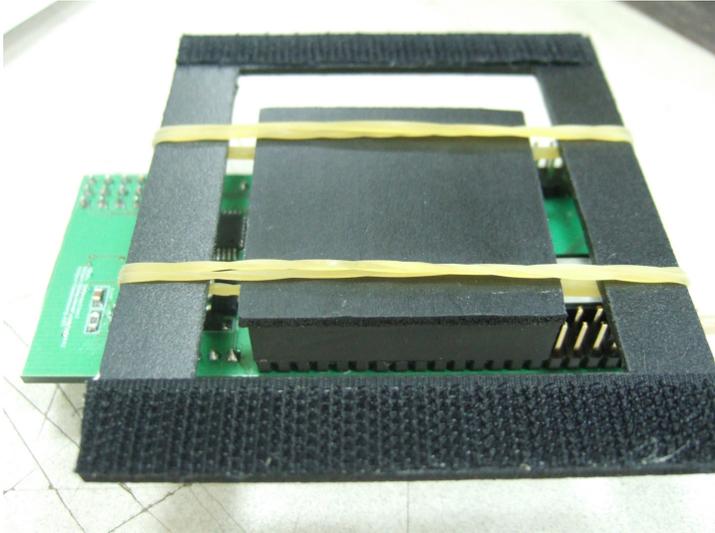


그림 23 . 마운트 밀면

5.1.3. 플랫폼 개별 부품 소모 비용

구분	제품	가격 (원, 부가세 포함)	구매처
기체	이지스타	79900	용산RC
송수신기	FUTABA 6EX 2.4 T/R Set	226700	용산RC
서보모터	HS-55	21800	용산RC
모터	E-MAX CF2812(Kv1534)	12500	용산RC
변속기	E-Max 35A V2	41000	용산RC
프로펠러	APC 6X5.5E 전동용 Prop	2900	용산RC
프롭아답터	프롭 아답터(모터축3.0mm,프롭축4.6mm)	2750	용산RC
배터리	PT-B2100N-PL	68500	헬리넷
기타	마운트용 목재 및 포맥스	10000	Alpha 문구점

표 6 . 개별 부품 소모 비용

5.2. 스위처

무인항공기를 연구함에 있어 자동/수동 전환에 대한 신뢰도는 가장 중요한 부분이 된다. 때문에 AVR을 이용하지 않고 CPLD를 이용하여 게이트 레벨의 시스템을 구성하여 신뢰도를 높여 사용하고 있다. 스위처에 대한 내용은 이 문서의 범위를 벗어나기 때문에 여기서는 다루지 않고 문서번호 KW-SW-001을 참조하기 바란다.

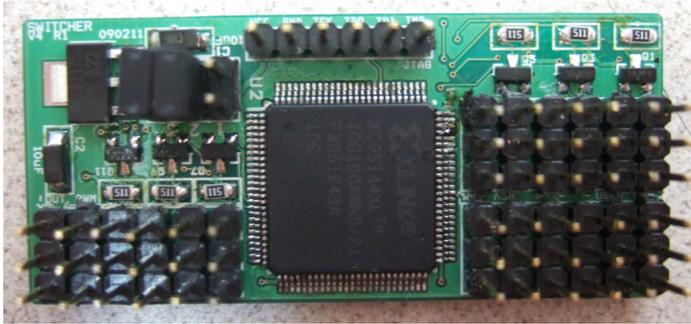


그림 24 . 스위처



그림 25 . 스위처 크기 비교

5.3. 시스템 전체 구성

비행 준비가 완료된 시스템의 전체 구성은 다음과 같다.

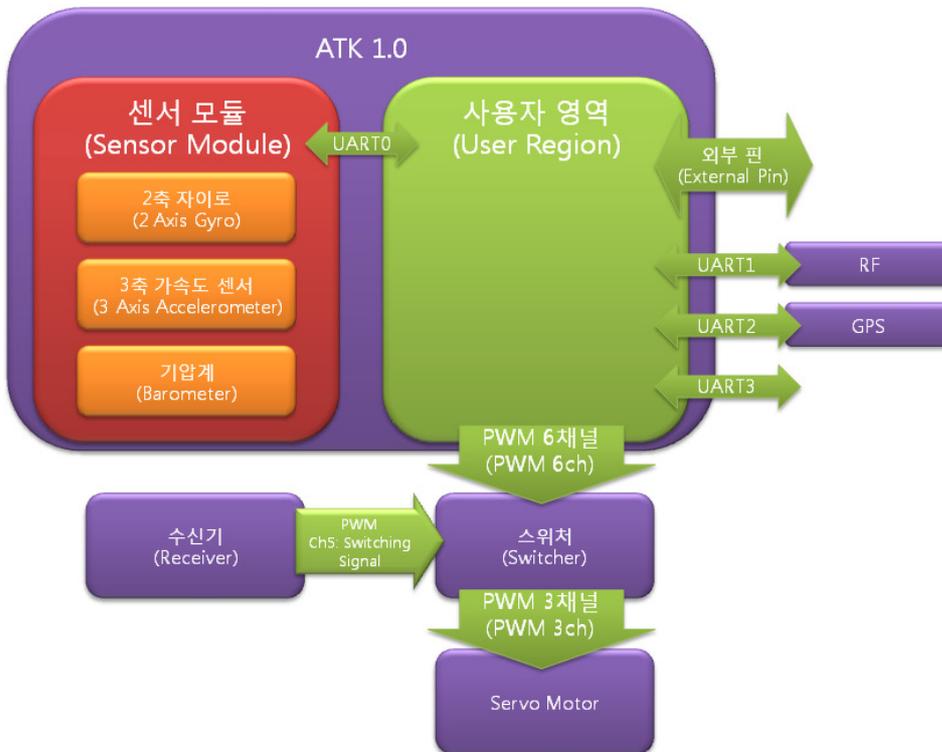


그림 26 . 비행 전 전체 시스템 블럭도

5.4. 장착 모습

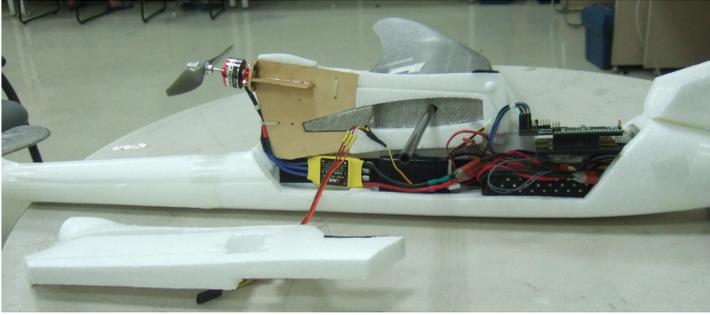


그림 27 . 비행 전 장착 모습



그림 28 . 비행 전 장착 모습

5.5. 사용자 영역 코드

이 장에서는 뱅크턴을 구현하기 위해 사용자 영역에 올라간 코드를 선보인다.

```
#include <mega2560.h>
#include <delay.h>
#include "hardware.h"

#define BUFFERLENGTH 1024

void ParsingData(char *Str, int *Output, int DataNum)
{
    int index;
    int Pindex = 0;
    int Sign = 1;

    for(index = 0; index < DataNum; index++)
        Output[index] = 0;

    for(index = 0;; index++)
        if(!Str[index])
            break;
        else
        {
            if(('0' <= Str[index]) && (Str[index] <= '9'))
            {
                Output[Pindex] *= 10;
                Output[Pindex] += (Str[index] - '0');
            }

            if(Str[index] == ' ')

```

```

        {
            Output[Pindex] *= Sign;
            Sign = 1;
            Pindex++;
        }

        if(Str[index] == '-')
            Sign = -1;
    }
}

void main(void)
{
    static char ReadBuffer[BUFFERLENGTH];
    static char WriteBuffer[BUFFERLENGTH];

    int ReadIndex = 0;
    int Temp;

    double GainRoll, GainPitch;
    int GoalRoll, GoalPitch, Thro;

    int index;                // For General Purpose

    int Pitch, Roll;
    int ParseData[10];
    Init();
    printf("Init OK!");

    // Gain과 원하는 자세를 놓는 부분이다.
    GainRoll = 1.0;
    GainPitch = -1.0;

    GoalRoll = 2;
    GoalPitch = -4;
    Thro = 30;

    while (1)
    {
        if( UCSR0A & RX_COMPLETE ) //UDR0 데이터 확인
        {
            ReadBuffer[ReadIndex] = UDR0;

            if(ReadBuffer[ReadIndex] == '\n') // 끝부분이면
            {
                for(index = 0; index <= ReadIndex; index++)
                    putchar1(ReadBuffer[index]);

                //scanf(&ReadBuffer[2], "%d %d", &Roll, &Pitch); // 데이터 파싱

                ParsingData(ReadBuffer, ParseData, 2);
                Roll = ParseData[0];
                Pitch = ParseData[1];

                sprintf(WriteBuffer, "%d %d\r\n", Roll, Pitch);
                for(index = 0;; index++)
                    if(!WriteBuffer[index])
                        break;
                    else
                        putchar1(WriteBuffer[index]);

                // 비례제어 루틴 롤 부분
                Temp = GoalRoll - Roll;
                Temp = (int)(Temp * GainRoll) ;
                Temp += 45;

                if(Temp > 90)
                    Temp = 90;

                if(Temp < 0)
                    Temp = 0;
                Servo_1(Temp);

                // 비례제어 루틴 피칭 부분
                Temp = GoalPitch - Pitch;
                Temp = (int)(Temp * GainPitch) ;
                Temp += 45;

                if(Temp > 90)
                    Temp = 90;

                if(Temp < 0)

```

```

        Temp = 0;
        Servo_2(Temp);
        Servo_3(Thro);
        Servo_4(45);
        ReadIndex = 0;
    }
    ReadIndex++;
}
};
}

```

5.6. 비행 영상

비행 영상은 아래의 링크에서 볼 수 있다.

< YouTube.com >

<http://www.youtube.com/user/seaofmagic>

< 개발자 블로그 >

<http://blog.nave.com/asd441>

<http://leewoosung.net>

6. 개발팀 Comment

6.1. 이우성

최근 10여년간 국내의 많은 대학교에서는 로봇 관련 학과 및 동아리가 생겨났다. 로봇 제작에 필요한 부품을 구하는 것도 많이 쉬워졌을 뿐 아니라 간단한 로봇을 제작하는데 필요한 비용도 과거에 비해서 많이 줄어들었다. 이러한 변화는 많은 학생들이 로봇이라는 주제에 쉽게 접근할 수 있도록 진입장벽을 낮춘 긍정적인 변화로 볼 수 있다.

그러나 무인항공기 관련 연구에 있어서는 아직 우리나라에서는 비용에 대한 진입장벽이 너무나 높다. 항공기 제어에 필요한 가장 기본적인 정보를 제공하는 자세 추정장치만을 구하려 해도 기실 수백만원을 호가한다. 이러한 제약은 무인항공기에 대한 연구가 대학 연구실 정도 이상이 아니면 힘들게 만들고 있다. 지난 10여년간의 각종 로봇 대회를 살펴보면 알 수 있듯이 대학교의 로봇 관련 학과 모임 및 동아리가 창의력의 산실이자 다양한 시도의 원천이며 후진 양성의 요람인데, 무인항공기 관련 연구에 있어서 자금이라는 문제는 일반 학생들이 시작하기에는 너무나 큰 장벽임이 분명하다.

우리 팀은 운 좋게도 한국항공대학교에서, 삼성소프트웨어멤버십에서 무인항공기 연구에 대한 기회를 남들에 비하면 조금은 쉽게 얻을 수 있었다. 다른 모든 것을 차치하고서라도 멀리 가지 않고도 활주로를 써 볼 수 있었고, 실제 대학원 연구팀을 만나볼 수 있었다는 것은 한국항공대이기 때문에 가능한 일이었고, 좋은 장비와 장소를 제공받았다는 것은 멤버십이기에 가능한 것이었다. 그러나 우리 팀 또한 자금이라는 문제에 부딪혔고, 개발을 위한 기간은 학업과 프로젝트뿐만이 아니라 자금을 마련하기 위해 여러 대회에 출전하거나 외부 아르바이트를 하는 등 가윗일들을 하는데 소모되었다. 훌륭한 팀원과 마음이 맞아 많은 순간 고생을 고생이라 여기지 않으며 포기하지 않을 수 있었고 각종 대회에서의 긍정적인 결과들을 얻어낼 수 있었기에 다행이었다.

그러한 과정을 거쳐 약 1년 반이 지난 지금, 기본적인 무인비행이 가능한 모듈을 내놓을 수 있게 된 것은 큰 기쁨을 안겨주고 있다. 부디 이 모듈을 시작으로 하여 학생 및 일반인들이 앞으로 더 좋고 더 저렴한 키트들과 함께 무인항공기 연구를 시작할 수 있으면 한다. 그 작지만 무수히 많은 시도들이

우리나라 무인항공기 산업에 수많은 창의력을 제공하고 수없는 도전의 시작이 되어 향후 21세기 핵심 산업이자 미래 국방력의 기준이 될 항공우주 산업 분야에 있어서 대한민국이 앞선 국가가 되기를 진심으로 기원한다.

2009년 3월 새벽. 신촌. 일출을 바라보며.

ps.

모든 것을 혼자 이뤄낼 수는 없다. 개발 과정에서 도움을 준 사람을 꼽으라면 밤하늘의 별과 같이 수없이 많지만 그 중에서도 특히 두 사람에게 고마움을 표한다.

프로젝트를 처음 제안한 사람이자 멤버십을 알려주고, 하루 24시간 중 잠자는 시간 빼고 거의 모든 시간을 함께 했던, 대단한 후배이자 믿음직한 PL이며 마음을 함께하는 동료인 **근범이**. 프로젝트를 병행하는 와중에도 학업에 소홀하지 않을 수 있게 흔들리는 나를 옆에서 다잡아 주고 모르는 것이 있으면 심혈을 기울여 알려준, 내 소중한 친구인 **중혁이**. 너희들이 있었기에 힘든 순간들을 버티며 학교도, 일도 할 수 있었어. 고마워. 정말 고마워.

6.2. 박근범

연구는 꿈이 있는 사람이 하는 것이지, 돈이 있는 사람이 하는 것이 아니다. 라는 생각으로 프로젝트를 한 것이 1년이란 세월을 훌쩍 넘겼다. 지나고 보면, 그때그때 어떻게 된 것과 도움 받은 것이 많은 시간이었고, 지금 되돌아보면 무모했던 시도들도 많이 있었다. 2007년, 11월. 내 수중에 70만원이 있었다. 그걸로 뭐가 어떻게 될 것이라고 생각했을까. 하지 말라는 주위의 만류들과 실패하였을 때의 리스크들이 머릿속에서 사라지지 않은 시간들이었다. 잃지 않으면 얻을 수 없다. 2008년이란 시간동안, 7개의 수상, 12개의 프로젝트 이면에는 수없이 잃어버린 인연들과 지금은 나약했음이라고 지칭하던 감정들이 있었다. 프로젝트를 하는 이에게는 이상을 가지면 안 된다고 생각하며 살고 있다. 그것이 마치 한계점처럼 작용하는 날이 올 것이기 때문이다. 기술도, 돈도, 될 거라는 명확한 확신도 없던 시기와 오늘 내가 운동을 하지 않아도 돼서 너무 행복하다던 은퇴한 축구 선수처럼 심각한 형태의 매너리즘에서 구해준 것은 언제나 내가 얻은 삶의 의지, 그 하나 였다. 무언가에게서 기대지 않고 살아가지 않음이라는 것을 얻어내는데, 그만한 시간과 정신적 방황과 고난과 고행이 필요했었다.

이제는 지나간 일들에 대해 감사해야 할 시간들이 아닐까. 나에게 갈아임을 옷 세벌이 있어서 감사하고, 바쁜 척 하는 놈에게 가끔씩 찾아와주는 친구들이 있어 감사하고, 그날 술 마실 만 원짜리 두 장이 있어 감사하다. 그리고 항상 자리에 있어주는 연구할 것들과 도와주시는 명훈 형과 우성이형에게 감사하다.

이 프로젝트는 그러한 취지에서 시작되었던, 무인기를 시작하는 이들에게 줄 보시와도 같은 것이며, 또한 숙제이기도 하다. 이제 어떻게 나느냐 보다는 왜, 무엇을 위해 나느냐는 숙제를 후배들에게 남겨주고 싶기 때문이다. 부디 조국의 하늘에 날 수 있는 곳이 한 평이라도 허락되길 기원하겠다.