Autobase Open Bus(AOB) Specification

Version 1.1

2014.10.30

목차

1. Autobase Open Bus

1.1. AOB 정의

Autobase Open Bus (AOB) 는 Master 장비와 Slave 장비의 통신을 표준화하여 서로 다른 기종의 디바이스간 통신을 쉽게 할 수 있도록 고안한 규약입니다.

1.2. AOB History

Version	Date	변화된 내용
1.1.0.1	2014-11-13	AOB_Slave Sample 프로그램에서 통신 시 다운되는 문제가 있어서 수정함
1.1	2014-10-30	Command RL 추가 – Read LONG Command RU 추가 – Read Double Command RT 추가 – Read Text Command WL 추가 – Write LONG Command WU 추가 – Write Double Command WT 추가 – Write Text AOB Master/Slave Sample 프로젝트에서 TCP통신 샘플 추가
1.0	2014-09-22	AOB 규약 최초 제정 Command RW – Read WORD Command RD – Read DWORD Command RF – Read Float Command WW – Write WORD Command WB – Write Bit Command WD – Write DWORD Command WF – Write Float Command SV – Get AOB Version Command er – Error Code

2. AOB 프로토콜 구조

2.1. AOB 프로토콜 기본 구조 (ASCII)

AOB 프로토콜은 아스키 통신을 기본으로 사용한다. (차후 버전에는 바이너리 통신도 지원 예정) START/END 는 STX/ETX를 사용하고 Command를 제외한 모든 블록은 "0123456789ABCDEF" 16개의 문자만을 사용하며 Command는 알파벳 문자만 사용하여 통신 시 오류 검출에 뛰어나고 프레임의 Start와 End를 찾기 쉬워서 코딩이 간단하다.

AOB 프로토콜 전체 통신 구조

Member	**STX**	**Length**	**Transaction id**	**Station**	**Command**	**Block**	**CRC**	**ETX**
Bytes	1	4	4	2	2	n	4	1
Sample	0x02	"0010"	"AB12"	"00"	"RW"	n	"78CD"	0x03

Member	**설명**	**값 범위**
STX	프레임 시작 코드 (항상 0x02)	0x02
Length	STX ~ ETX를 포함한 전체 프레임의 크기 (최소 크기는 18바이트)	0x0012~0xFFFF
Transaction id	각 통신 프레임의 고유 처리번호. 일반적으로 통신할 때마다 숫자를 1씩 증가시켜서 송신하고 수신한 프레임의 Transaction id 와 일치하면 정상 통신으로 받아 들인다.	0x0000~0xFFFF
Station	0x00~0xFF	0x00~0xFF
Command	영문자 두문자를 사용한다. 명령어가 대문자이면 Master ->Slave 통신이고 소문자이면 Slave->Master 통신이다.	AA~ZZ 또는 aa~zz
Block	각 Command에 따라서 크기와 내용이 다르다.	
CRC	STX와 ETX를 제외한 Length부터 Block까지의 SUM 값이다.	0x0000~0xFFFF
ETX	프레임 종료 코드 항상 (0x03)	0x03

2.2. CRC 계산법

CRC 는 Length부터 Block까지의 SUM값이다. 아래는 CRC 계산하는 C# 예제이다.

```
ushort GetCRC(byte[] data, int start, int size)
{
            ushort crc = 0;

            for (int i = 0, pos=start; i < size; i++, pos++)
            {
                crc += data[pos];
            }

            return crc;
}
```

2.3. Sample Code (Read WORD Block)

Master->Slave		
Member	**Hex**	**ASCII**
STX	0x02	" "
Length	0x30 0x30 0x31 0x43	"001C"
Transaction id	0x30 0x30 0x30 0x31	"0001"
Station	0x30 0x30	"00"
Command	0x52 0x57	"RW"
Table	0x30 0x30	"00"
Address	0x30 0x30 0x30 0x30	"0000"
Count	0x30 0x30 0x30 0x31	"0001"
CRC	0x30 0x34 0x37 0x46	"047F"
ETX	0x03	" "

Slave->Master		
Member	**Hex**	**ASCII**
STX	0x02	" "
Length	0x30 0x30 0x32 0x30	"0020"
Transaction id	0x30 0x30 0x30 0x31	"0001"
Station	0x30 0x30	"00"
Command	0x72 0x77	"rw"
Table	0x30 0x30	"00"
Address	0x30 0x30 0x30 0x30	"0000"
Count	0x30 0x30 0x30 0x31	"0001"
Data	0x30 0x30 0x30 0x30	"0000"
CRC	0x30 0x35 0x36 0x44	"056D"
ETX	0x03	" "

3. COMMAND

COMMAND는 두 글자의 영문자로 구성된다. Master에서 Slave로 전송할 때는 대문자 명령어를 사용하고 Slave에서 Master로 전송할 때는 소문자 명령어가 사용된다.

AOB 명령어는 계속 버전업 되면서 명령어가 많이 추가될 예정이다. 많은 명령어를 모두 지원할 필요는 없다. 명령어 중 디바이스에 해당되는 부분만 지원하면 되고 지원하지 않는 명령어는 지원하지 않는다고 오류 코드("er" Command)를 보내주면 된다.

3.1. AOB Slave 디바이스가 일반적으로 사용하는 COMMAND

Command	지원	비고
SV	필수 지원	AOB Version을 주고 받는 Command - **모든 디바이스는 반드시 이 명령어를 지원해야 한다.**
er	필수 지원	오류 발생 시 Slave에서 Master로 보내주는 오류 Command - **모든 디바이스는 반드시 이 명령어를 지원해야 한다. 특히 Master가 지원하지 않는 Command로 요청하면 Slave는 반드시 Master에게 오류코드 0x0001을 보내야 한다.**
RW	선택 지원	Slave의 16Bit 데이터를 읽어오는 Command – 반드시 지원할 필요는 없다.
WW	선택 지원	Slave의 16Bit 데이터에 출력하는 Command – 반드시 지원할 필요는 없다.

위와 같이 네 가지 명령만 지원하면 Master/Slave 간 WORD 배열 읽기/쓰기가 가능하다.

3.2. COMMAND – RW (Read WORD Block)

WORD (16bit) 배열 데이터를 읽어온다.

(Version 1.0 부터 지원)

Master->Slave

Member	Command (RW)	Table Number	Address	Count
Bytes	2	2	4	4

Slave->Master

Member	Command (rw)	Table Number	Address	Count	Data
Bytes	2	2	4	4	Count*4

3.3. COMMAND – RD (Read DWORD Block)

DWORD (32bit) 배열 데이터를 읽어온다.

(Version 1.0 부터 지원)

Master->Slave

Member	Command (RD)	Table Number	Address	Count
Bytes	2	2	4	4

Slave->Master

Member	Command (rd)	Table Number	Address	Count	Data
Bytes	2	2	4	4	Count*8

3.4. COMMAND – RF (Read FLOAT Block)

FLOAT (32bit) 배열 데이터를 읽어온다.

(Version 1.0 부터 지원)

Master->Slave

Member	Command (RF)	Table Number	Address	Count
Bytes	2	2	4	4

Slave->Master

Member	Command (rf)	Table Number	Address	Count	Data
Bytes	2	2	4	4	Count*8

3.5. COMMAND – RL (Read LONG Block)

LONG (64bit) 배열 데이터를 읽어온다.

(Version 1.1 부터 지원)

Master->Slave

Member	Command (RL)	Table Number	Address	Count
Bytes	2	2	4	4

Slave->Master

Member	Command (rl)	Table Number	Address	Count	Data
Bytes	2	2	4	4	Count*16

3.6. COMMAND – RU (Read Double Block)

Double (64bit) 배열 데이터를 읽어온다.

(Version 1.1 부터 지원)

Master->Slave

Member	Command (RU)	Table Number	Address	Count
Bytes	2	2	4	4

Slave->Master

Member	Command (ru)	Table Number	Address	Count	Data
Bytes	2	2	4	4	Count*16

3.7. COMMAND – RT (Read Text Block)

Text (Unicode) 배열 데이터를 읽어온다.

(Version 1.1 부터 지원)

Master->Slave

Member	Command (RT)	Table Number	Address	Count
Bytes	2	2	4	4

Slave->Master

Member	Command (rt)	Table Number	Address	Count	Data
Bytes	2	2	4	4	

Data 는 Size(4)+Data(size*2)+Size(4)+Data(size*2)+... 식으로 Count 까지 반복된다.

데이터를 보낼 때는 Unicode 문자열을 UTF-8 byte 배열로 변환해서 전송한다.

3.8. COMMAND – WW (Write WORD Block)

WORD (16bit) 배열 데이터를 Slave로 출력한다.

(Version 1.0 부터 지원)

Master->Slave

Member	Command (WW)	Table Number	Address	Count	Data
Bytes	2	2	4	4	Count*4

Slave->Master

Member	Command (ww)
Bytes	2

3.9. COMMAND – WB (Write Bit)

Slave의 특정 비트에 비트 출력한다.

비트 출력을 지원하지 않아도 WORD로 읽은 후 AND/OR 연산한 후 WORD 출력할 수 있다. 그러나 마스터에서는 한번 출력하기 위해서 두 번의 통신을 해야 하므로 통신의 부담이 있다. 비트 출력이 필요하다면 Slave에서는 지원하는 것이 좋다.

(Version 1.0 부터 지원)

Master->Slave

Member	Command (WB)	Table Number	Address	Bit Position	Data
Bytes	2	2	4	2	2

Data는 "00" 또는 "01" 을 출력한다.

Slave->Master

Member	Command (wb)
Bytes	2

3.10. COMMAND – WD (Write DWORD Block)

DWORD (32bit) 배열 데이터를 Slave로 출력한다.

(Version 1.0 부터 지원)

Master->Slave

Member	Command (WD)	Table Number	Address	Count	Data
Bytes	2	2	4	4	Count*8

Slave->Master

Member	Command (wd)
Bytes	2

3.11. COMMAND – WF (Write FLOAT Block)

FLOAT (32bit) 배열 데이터를 Slave로 출력한다.

(Version 1.0 부터 지원)

Master->Slave

Member	Command (WF)	Table Number	Address	Count	Data
Bytes	2	2	4	4	Count*8

Slave->Master

Member	Command (wf)
Bytes	2

3.12. COMMAND – WL (Write LONG Block)

LONG (64bit) 배열 데이터를 Slave로 출력한다.

(Version 1.1 부터 지원)

Master->Slave

Member	Command (WD)	Table Number	Address	Count	Data
Bytes	2	2	4	4	Count*16

Slave->Master

Member	Command (wl)
Bytes	2

3.13. COMMAND – WU (Write Double Block)

Double (64bit) 배열 데이터를 Slave로 출력한다.

(Version 1.1 부터 지원)

Master->Slave

Member	Command (WF)	Table Number	Address	Count	Data
Bytes	2	2	4	4	Count*16

Slave->Master

Member	Command (wf)
Bytes	2

3.14. COMMAND – WT (Write Text Block)

문자열 배열 데이터를 Slave로 출력한다.

(Version 1.1 부터 지원)

Master->Slave

Member	Command (WT)	Table Number	Address	Count	Data
Bytes	**2**	**2**	**4**	**4**	

Data 는 Size(4)+Data(size*2)+Size(4)+Data(size*2)+... 식으로 Count 까지 반복된다.

데이터를 보낼 때는 Unicode 문자열을 UTF-8 byte 배열로 변환해서 전송한다.

Slave->Master

Member	Command (wt)
Bytes	2

3.15. COMMAND – SV (Get AOB Version)

AOB Version – 디바이스가 현재 지원하고 있는 Autobase Open Bus 버전을 주고 받는다. Master/Slave 는 각각 서로의 Major.Minor 버전 정보를 참고하여 통신 가능 여부를 알 수 있다. Master 장치가 최초 통신을 시작할 때 이 명령어를 보내주도록 한다.

(Version 1.0 부터 지원)

Master->Slave

Member	Command (SV)	Version Major	Version Minor	Version Build	Version Revision
Bytes	2	4	4	4	4

Master가 현재 지원하고 있는 AOB 프로토콜 버전을 Slave로 보내준다.

Slave->Master

Member	Command (sv)	Version Major	Version Minor	Version Build	Version Revision
Bytes	2	4	4	4	4

Slave가 현재 지원하고 있는 AOB 프로토콜 버전을 Master로 보내준다.

버전 설명

Major : 프로토콜의 구조에 큰 변화가 있을 경우 변경됨. 버전이 틀릴 경우 통신이 불가능할 수도 있다. (디바이스가 현재 지원 중인 AOB의 Major 버전)

Minor : Command가 추가되거나 수정될 경우 변경. 버전이 틀릴 경우 통신은 문제가 없지만 특정 Command를 읽지 못할 수 있다. (디바이스가 현재 지원 중인 AOB의 Minor 버전)

Build : 보통 디바이스의 빌드 시 2000-1-1 이후의 일 수를 나타내지만 디바이스 특성에 맞도록 버전 정보를 사용하여도 좋다.

Revision : 같은날 빌드할 경우 시간 구분으로 사용할 수도 있고 디바이스 특성에 맞도록 버전 정보를 사용하여도 좋다.

3.16. COMMAND – ZA ~ ZZ (User Defined Command)

ZA ~ ZZ 는 사용자 정의 영역이다.

Master->Slave는 ZA~ZZ를 송신하고 Slave->Master는 za~zz로 응답한다.

장비의 특성에 맞게 특수한 기능의 프로토콜을 작성하고 할 때 이 명령어를 사용하도록 한다.

ZA ~ ZZ 까지 26개의 명령어를 사용할 수 있다.

3.17. COMMAND – er (Error Code Command)

오류 코드 COMMAND - Master에서 Slave로 통신을 요청할 때 처리할 수 없는 경우 Slave가 가만히 있으면 보통 Master는 일정 시간 동안(Timeout) 데이터가 들어오기를 기다리게 된다.

이런 경우 통신이 원활하게 진행될 수 있도록 프레임을 받으면 빨리 오류 코드를 발송하여 Mater가 다음 상황으로 빨리 진행할 수 있도록 해야 한다.

Slave->Master

Member	**Command (er)**	**Error Number**	**Message Count**	**Message**
Bytes	2	4	4	Message Count * 2

Error Number는 0x0000~0xFFFF사이의 값으로 고유 오류 번호이고 Message는 오류에 해당하는 적절한 메시지를 보내준다. 메시지 형식은 문자열으로 UTF-8형식이다.

아래의 오류 번호 중에서 해당 항목을 찾을 수 없을 경우는 오류번호 0x0000 을 사용하여 오류 메시지의 자세한 내용을 Text로 보내주면 되고, 이미 정의된 오류는 Message Count는 0으로 하고 오류 번호만 보내주면 된다.

Error Number

Error Number	**내용**	**추가된 시점**
0x0000	Text Error Message.	1.0
0x0001	**Command not supported.**	**1.0**
0x0002	Sub Command not supported.	1.0
0x0003	Station not exists.	1.0
0x0004	Table Number not exists.	1.1
0x0005	Address not exists.	1.1
0x0006	CRC mismatched.	1.1
0x0007 ~ 0xEFFF	Reserved	1.0
0xF000 ~ 0xFFFF	User defined error message	1.0

4. 기타

4.1. Unicode – UTF-8 변환

AOB 프로토콜 내의 모든 문자열은 UTF-8을 사용한다. UTF-8은 Unicode 문자열을 크기를 줄이기 위해 인코딩하는 형식이다.

C++에서는 아래 예제와 같이 WideCharToMultiByte, MultiByteToWideChar 함수를 사용하여 서로 변환할 수 있다.

```
void Sample_Convert()
{
	static char text_utf8[50];
	static WCHAR text_unicode[50];

	wcscpy(text_unicode, L"ABC가나다");

	int retn;

	// Convert Unocode -> UTF-8
	retn = WideCharToMultiByte(CP_UTF8, 0, text_unicode, wcslen(text_unicode),
text_utf8, 50, NULL, NULL);

	// Convert UTF-8 -> Unocode
	retn = MultiByteToWideChar(CP_UTF8, 0, text_utf8, strlen(text_utf8),
text_unicode, 50);
}
```