1. Communication Specifications

인터페이스(Interface)	RS-485 [Isolation]
결선 방법(Connection method)	2 선 Full Duplex
동기 방법(Synchronous method)	Start, Stop 비트 방식
통신 속도	9600, 19200 BPS
데이터 비트 구성	Start Bit : 1 Data Bit : 8 Parity Bit : None Stop Bit : 1
전송모드(Signal transmission mode)	리모트 터미널 유니트 모드
Function code:	03H (데이터 읽기 명령어) 04H (데이터 읽기 명령어) 06H (한개의 데이터 쓰기 명령어) 08H (루프백 테스트) * Multi Write는 지원하지 않습니다.
에러 체크 방법(Error check method)	CRC-16
에러 코드(Error code)	1: Function 코드 에러 [없는 명령어의 수신] 2: Parameter 에러 [없는 Parameter의 수신] 3: Parameter 에러 [사용되지 않는 Parameter 수신] 4: Data 에러 [Data 범위 Over]

2. 통신 프로토콜

2.1 메시지 형태

메시지의 형태는 Slave 어드레스,Function 코드,데이터,CRC 로 구성 된다.

Slave 아이디
Function 코드
데이터
CRC-16

- 슬래이브 아이디

  슬래이브 어드레스는 0 에서부터 99 까지 설정 가능

- Function 코드

  Master에 의하여 전송 되는 명령어 이다.

- 데이터

  데이터는 2 바이트 인티저 타입을 지원.
- 에러 체크

  CRC - 16 에러 체크 코드를 사용 한다.

2.2 Function code

Function code contents

Function Code	Function	Contents
03H	데이터 읽기 명령어	디스플레이 상태 읽기
06H	한개의 데이터 쓰기 명령어	디스플레이 표시
08H	루프벡 테스트	루프벡 테스트

Function 코드 의 최대 메시지 길이

Function Code	Function	Query Message		Response Message	
		Min	Max	Min	Max
03H	데이터 읽기 명령어	8	8	7	57
06H	한개의 데이터 쓰기 명령어	8	8	8	8
08H	루프백 테스트	8	8	8	8

2.3 Communication mode

전송 모드

Items	Contents
데이터 비트의 길이	8 비트 바이너리
메시지의 스타트 문자	사용하지 안음
메시지 종료 문자	사용하지 안음
메시지의 길이	펑션 코드 참조
데이터의 인터벌	사용 보레이트의 24 비트 이내
에러 채크 방식	CRC-16

2.4 Slave의 응답

(1) 에러 메시지의 응답

Slave address
Function code
Error code
Error check CRC-16

에러 메시지의 Function 코드는 80hex 이상를 사용 하여야 함

Error code	Contents
1	Function 코드 에러
2	Parameter 코드 에러 (No Parameter)
3	Parameter 코드 에러 (No use Parameter)
4	Data 범위 Over

(2) 슬레이브의 응답이 없는 경우

슬레이브는 다음의 경우에 응답 하지 않는다.

수신한 아이디 가 자신의 어드레스와 일치 하지 않을 경우

수신한 메시지에 Parity 에러, 오버런 에러 등을 포함 하고 있을 경우

수신한 메시지의 길이를 초과 하였을 경우

메시지의 인터벌이 사용 Baudrate 24 비트를 초과 하였을 경우

2.5 CRC-16

Example

The function takes two arguments:

```
/* A pointer to the message buffer containing binary data to be used
for generating the CRC */
  unsigned char *puchMsg;
/* The quantity of bytes in the message buffer */
unsigned short usDataLen;
```

The function returns the CRC as a type unsigned short.

CRC Generation function

```
 unsigned short CRC16( puchMsg, usDataLen)
 unsigned char puchMsg;    /* message to calculate CRC upon */
 unsigned short usDataLen; /* quantity of bytes in message */

{
unsigned char uchCRCHi = 0xFF; /* high byte of CRC initialized */
unsigned char uchCRCLo = 0xFF; /* Low byte of CRC initialized */
while(usDataLen--){            /* Pass through message buffer */
             ulndex = uchCRCHi ^ pchMsg++;  /* calculate the CRC */
             uchCRCHi = uchCRCLo ^ auchCRCHi[ulndex];
uchCRCLo = auchCRCLo[ulndex];
}
return (uchCRCHi << 8 | uchCRCLo);
}
```

3. 메시지 형태

3.1 데이터 읽기 명령어 [03H]

(1) 메시지 형태

Slave 아이디		02H
읽기 명령어		03H
시작 어드레스	High	20H
	Low	00H
읽는 워드수	High	00H
	Low	01H
CRC-16	High	85H
	Low	F3H

(2) Slave의 응답

Slave 아이디		02H
읽기 명령어		**03H**
데이터의 수 (바이트)		02
데이터	High	00H
	Low	25H
CRC-16	High	3DH
	Low	9FH

(3) 에러 응답

Slave 아이디		02H
80H+Function 코드		83H
에러 코드		01H
CRC-16	High	70H
	Low	F0H

3.2 데이터 쓰기 명령어 [06H]

(1) 메시지 형태

Slave 아이디		02H
쓰기 명령어		06H
데이터 쓰기 어드레스	High	00H
	Low	10H
데이터	High	12H
	Low	34H
CRC-16	High	85H
	Low	4BH

(2) Slave 의 응답

Slave 아이디		02H
쓰기 명령어		06H
데이터 쓰기 어드레스	High	00H
	Low	10H
데이터	High	12H
	Low	34H
CRC-16	High	85H
	Low	4BH

(3) 에러 응답

Slave 아이디		02H
80H+명령어		86H
에러 코드		01H
CRC-16	High	70H
	Low	F0H

3.3 루프 백 테스트 [08H]

(1) 메시지 형태

Slave 아이디		02H	
명령어		08H	
사용 어드레스	High	00H	이 사용 Address는 항상 0이어야 한다.
	Low	00H	
데이터	High	1FH	임의의 데이터
	Low	34H	
CRC-16	High	E9H	
	Low	DFH	

(2) Slave의 응답

Slave 아이디		02H
명령어		08H
사용 어드레스	High	00H
	Low	00H
데이터	High	1FH
	Low	34H
CRC-16	High	E9H
	Low	DFH

마스터로부터 받은 동일한 데이터

(3) 에러 응답

Slave 아이디		02H
80H+ 명령어		88H
에러 코드		01H
CRC-16	High	70H
	Low	F0H

4. Data configuration

데이터의 영역은 -32788 + 32276 이다.

5. 데이터 메핑

항목	Address	Read/Write	데이터 길이	데이터 범위	출하값	기타
PV	0	Read Only	2 Byte	0 ~ 9999	-	현재지시값
Point	1	Read / Write	2 Byte	0 ~ 3	0	순시 Point
TOTAL	2	Read Only	2 Byte	0 ~ 9999____	-	토탈적산
	3		2 Byte	0 ~ ____9999		
C-TOTAL	4	Read Only	2 Byte	0 ~ 999____	-	일일적산
	5		2 Byte	0 ~ ____9999		
TOTAL Point	6	Read / Write	2 Byte	0 ~ 7	0	적산 Point
Analog Output	7	Read Only	2 Byte		-	현재출력값
Alarm State	8	Read Only	2 Byte		0	Alarm 상태
Alarm 1 Value	9	Read / Write	2 Byte	0 ~ 9999	1	Alarm1 Data
Alarm 2 Value	10	Read / Write	2 Byte		0	Alarm2 Data
High Scale	11	Read / Write	2 Byte	0 ~ 9999	1000	Display 범위
Low Scale	12	Read / Write	2 Byte		0	
Sensor Adjust	13	Read / Write	2 Byte	-999 ~ 999	0	Sensor 보정
Function Mode	14	Read / Write	2 Byte	0 ~ 1	0	Linear
					1	Root
Alarm 1 Mode	15	Read / Write	2 Byte	0 ~ 1	0	Low Alarm
					1	High Alarm
Alarm 2 Mode	16	Read / Write	2 Byte	0 ~ 1	0	Low Alarm
					1	High Alarm
Alarm Deadband	17	Read / Write	2 Byte	0 ~ 99	3	Deadband Data
High Output	18	Read / Write	2 Byte	0 ~ 9999	1000	출력범위
Low Output	19	Read / Write	2 Byte		0	
Factor Value	20	Read / Write	2 Byte	0 ~ 9999	3600	Factor
C-TOTAL Reset	21	Write	2 Byte	1234	1234	Data 1234입력시 Total Reset
TOTAL Reset	22	Write	2 Byte	6644	6644	Data 6644입력시 Total Reset

● TOTAL, C-TOTAL 데이터는 2Byte당 4-DIGIT입니다. Address 3(5)의 데이터는 적산 지시창의 “____9999”를 의미하며, TOTAL Address 2의 데이터는 적산 지시창의 “9999____”를 의미합니다. 그리고 C-TOTAL Address 4의 데이터는 적산 지시창의 “C999____”를 의미합니다.(C는 고정)

● C-Total Reset은 Address 21이며 Data를 1234로 Write하면 C-Total Value가 Reset되며, Total Reset은 Address 22이며 Data를 6644로 Write하면 Total Value 전체를 Reset하는 기능입니다.

● PV, Alarm1 Value, Alarm2 Value, High Scale, Low Scale, Sensor Adjust, Alarm Deadband, High Output, Low Output의 Point는 Address 1의 Point값과 동일합니다.

또한 Address 7의 Analog Output의 Point는 2로 고정입니다.

```
static unsigned char auchCRCHi [256]={ /* Table of CRC value for high
order byte*/
        0x00, 0xc1, 0x81, 0x40, 0x01, 0xc0, 0x80, 0x41, 0x01, 0xc0,
        0x80, 0x41, 0x00, 0xc1, 0x81, 0x40, 0x01, 0xc0, 0x80, 0x41,
        0x00, 0xc1, 0x81, 0x40, 0x00, 0xc1, 0x81, 0x40, 0x01, 0xc0,
        0x80, 0x41, 0x01, 0xc0, 0x80, 0x41, 0x00, 0xc1, 0x81, 0x40,
        0x00, 0xc1, 0x81, 0x40, 0x01, 0xc0, 0x80, 0x41, 0x00, 0xc1,
        0x81, 0x40, 0x01, 0xc0, 0x80, 0x41, 0x01, 0xc0, 0x80, 0x41,
        0x00, 0xc1, 0x81, 0x40, 0x01, 0xc0, 0x80, 0x41, 0x00, 0xc1,
        0x81, 0x40, 0x00, 0xc1, 0x81, 0x40, 0x01, 0xc0, 0x80, 0x41,
        0x00, 0xc1, 0x81, 0x40, 0x01, 0xc0, 0x80, 0x41, 0x01, 0xc0,
        0x80, 0x41, 0x00, 0xc1, 0x81, 0x40, 0x00, 0xc1, 0x81, 0x40,
        0x01, 0xc0, 0x80, 0x41, 0x01, 0xc0, 0x80, 0x41, 0x00, 0xc1,
        0x81, 0x40, 0x01, 0xc0, 0x80, 0x41, 0x00, 0xc1, 0x81, 0x40,
        0x00, 0xc1, 0x81, 0x40, 0x01, 0xc0, 0x80, 0x41, 0x01, 0xc0,
        0x80, 0x41, 0x00, 0xc1, 0x81, 0x40, 0x00, 0xc1, 0x81, 0x40,
        0x01, 0xc0, 0x80, 0x41, 0x00, 0xc1, 0x81, 0x40, 0x01, 0xc0,
        0x80, 0x41, 0x01, 0xc0, 0x80, 0x41, 0x00, 0xc1, 0x81, 0x40,
        0x00, 0xc1, 0x81, 0x40, 0x01, 0xc0, 0x80, 0x41, 0x01, 0xc0,
        0x80, 0x41, 0x00, 0xc1, 0x81, 0x40, 0x01, 0xc0, 0x80, 0x41,
        0x00, 0xc1, 0x81, 0x40, 0x00, 0xc1, 0x81, 0x40, 0x01, 0xc0,
        0x80, 0x41, 0x00, 0xc1, 0x81, 0x40, 0x01, 0xc0, 0x80, 0x41,
        0x01, 0xc0, 0x80, 0x41, 0x00, 0xc1, 0x81, 0x40, 0x01, 0xc0,
        0x80, 0x41, 0x00, 0xc1, 0x81, 0x40, 0x00, 0xc1, 0x81, 0x40,
        0x01, 0xc0, 0x80, 0x41, 0x01, 0xc0, 0x80, 0x41, 0x00, 0xc1,
        0x81, 0x40, 0x00, 0xc1, 0x81, 0x40, 0x01, 0xc0, 0x80, 0x41,
        0x00, 0xc1, 0x81, 0x40, 0x01, 0xc0, 0x80, 0x41, 0x01, 0xc0,
        0x80, 0x41, 0x00, 0xc1, 0x81, 0x40 };
```

```
unsigned char auchCRCLo [256]={ /* Table of CRC value for Low order
byte*/
        0x00, 0xc0, 0xc1, 0x01, 0xc3, 0x03, 0x02, 0xc2, 0xc6, 0x06,
        0x07, 0xc7, 0x05, 0xc5, 0xc4, 0x04, 0xcc, 0x0c, 0x0d, 0xcd,
        0x0f, 0xcf, 0xce, 0x0e, 0x0a, 0xca, 0xcb, 0x0b, 0xc9, 0x09,
        0x08, 0xc8, 0xd8, 0x18, 0x19, 0xd9, 0x1b, 0xdb, 0xda, 0x1a,
        0x1e, 0xde, 0xdf, 0x1f, 0xdd, 0x1d, 0x1c, 0xdc, 0x14, 0xd4,
        0xd5, 0x15, 0xd7, 0x17, 0x16, 0xd6, 0xd2, 0x12, 0x13, 0xd3,
        0x11, 0xd1, 0xd0, 0x10, 0xf0, 0x30, 0x31, 0xf1, 0x33, 0xf3,
        0xf2, 0x32, 0x36, 0xf6, 0xf7, 0x37, 0xf5, 0x35, 0x34, 0xf4,
        0x3c, 0xfc, 0xfd, 0x3d, 0xff, 0x3f, 0x3e, 0xfe, 0xfa, 0x3a,
        0x3b, 0xfb, 0x39, 0xf9, 0xf8, 0x38, 0x28, 0xe8, 0xe9, 0x29,
        0xeb, 0x2b, 0x2a, 0xea, 0xee, 0x2e, 0x2f, 0xef, 0x2d, 0xed,
        0xec, 0x2c, 0xe4, 0x24, 0x25, 0xe5, 0x27, 0xe7, 0xe6, 0x26,
        0x22, 0xe2, 0xe3, 0x23, 0xe1, 0x21, 0x20, 0xe0, 0xa0, 0x60,
        0x61, 0xa1, 0x63, 0xa3, 0xa2, 0x62, 0x66, 0xa6, 0xa7, 0x67,
        0xa5, 0x65, 0x64, 0xa4, 0x6c, 0xac, 0xad, 0x6d, 0xaf, 0x6f,
        0x6e, 0xae, 0xaa, 0x6a, 0x6b, 0xab, 0x69, 0xa9, 0xa8, 0x68,
        0x78, 0xb8, 0xb9, 0x79, 0xbb, 0x7b, 0x7a, 0xba, 0xbe, 0x7e,
        0x7f, 0xbf, 0x7d, 0xbd, 0xbc, 0x7c, 0xb4, 0x74, 0x75, 0xb5,
        0x77, 0xb7, 0xb6, 0x76, 0x72, 0xb2, 0xb3, 0x73, 0xb1, 0x71,
        0x70, 0xb0, 0x50, 0x90, 0x91, 0x51, 0x93, 0x53, 0x52, 0x92,
        0x96, 0x56, 0x57, 0x97, 0x55, 0x95, 0x94, 0x54, 0x9c, 0x5c,
        0x5d, 0x9d, 0x5f, 0x9f, 0x9e, 0x5e, 0x5a, 0x9a, 0x9b, 0x5b,
        0x99, 0x59, 0x58, 0x98, 0x88, 0x48, 0x49, 0x89, 0x4b, 0x8b,
        0x8a, 0x4a, 0x4e, 0x8e, 0x8f, 0x4f, 0x8d, 0x4d, 0x4c, 0x8c,
        0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42,
        0x43, 0x83, 0x41, 0x81, 0x80, 0x40};
```

[아이디 2]

1. 버전 요구

Host]

0x02 0x03 0x00 0x00 0x00 0x02 SumH SumL

아이디 명령어 어드레스하이 어드레스로우 요구갯수하이 요구갯수 로우 채크썸하이 채크썸하이

CT] 버전 01 일경우 응답

0x02 0x03 0x04 0x56 0x00 0x30 0x31 SumH SumL

아이디 명령어 응답바이트수 데이터'V' 데이터 데이터'0' 데이터'1'채크썸하이 채크썸하이

CT] 버전 10 일경우 응답

0x02 0x03 0x04 0x56 0x00 0x31 0x30 SumH SumL

아이디 명령어 응답바이트수 데이터'V' 데이터 데이터'0' 데이터'1'채크썸하이 채크썸하이

CT] 에러를 응답할 경우

0x02 0x83 0x02 SumH SumL

아이디 0x80+명령어 에러코드 채크썸하이 채크썸하이

2. 리모트/로칼

Host]

0x02 0x06 0x00 0x02 0x00 0x01 SumH SumL

아이디 명령어 어드레스하이 어드레스로우 요구갯수하이 요구갯수 로우 채크썸하이 채크썸하이

CT] 리모트로 설정할 경우

0x02 0x06 0x00 0x02 0x00 0x01 SumH SumL

아이디 명령어 어드레스하이 어드레스로우 요구갯수하이 요구갯수 로우 채크썸하이 채크썸하이

기타 사항 :

반드시 들어가야 하는 기능

데이터 메핑의2번 5번 6번 8번 12번 13번 14번 15번 19번 20번

나머지는 프로그래머의 판단에 따름.