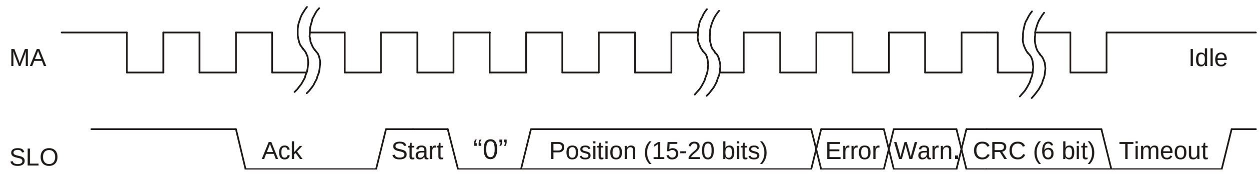
E201-9S Application note
Decoding the BiSS information

The E206-9S interrogates a BiSS-C encoder and allows the data to be read by a PC using simple ASCII commands over a USB connection and a virtual COM port.
Features of the BiSS data transmission and data packet details are described in this document.

BiSS-C timing diagram



MA line is idle high. Communication is initiated with first falling edge.
The encoder responds by setting SLO low on the second rising edge on MA.
When the encoder is ready for the next request cycle it indicates this to the master by setting SLO high. The absolute position and CRC data is in binary format and sent MSB first.

Encoder reading is started with sending the ASCII character "4" to the E201-9S interface. No CR character is required after the command.

E201-9S returns 16 character hexadecimal string + CR comprising 64 SLO bits synchronized to 64 MA clocks.
Note: Available in E201 interface version 1.16 (and later).

Data packet (containing position, status and CRC) starts right after the "010" sequence (ACK, Start, CDS).

If the ACK sequence is too long and the data packet falls out of the 64-bit word read by the E201-9S the MA clock frequency must be reduced.

E201-9S does not make the line-delay compensation, therefore it is possible that at some setting of the MA clock frequency the readout data is not stable and produces CRC errors. In such case choose different MA clock setting.

For MA clock setting see the "M" command in the E201 datasheet (E201D01).

Example 1

Encoder used in example 1:

Type: Linear absolute encoder, BiSS output
Resolution: 0.05 um
Position length: 32 bits
Status length: 2 bits (active low)
CRC length: 6 bits, polynomial $x^6 + x^1 + 1$ (Represented also as 0x43), inverted

Example of the response to the “4” command: **c0040030320ffac0** (hex)

Response																Format
C	0	0	4	0	0	3	0	3	2	0	F	F	A	C	0	Hex
1100	0000	0000	0100	0000	0000	0011	0000	0011	0010	0000	1111	1111	1010	1100	0000	Binary

Decoding the response into Position, Status and CRC:

1100 0000 0000 0100 0000 0000 0011 0000 0011 0010 0000 1111 1111 1010 1100 0000	
First two bits are always ‘1’	

1100 0000 0000 0100 0000 0000 0011 0000 0011 0010 0000 1111 1111 1010 1100 0000	ACK
Next eleven ‘0’ bits are the ACK bits. Number of ACK bits depends on encoder’s latency and BiSS frequency	

1100 0000 0000 0100 0000 0000 0011 0000 0011 0010 0000 1111 1111 1010 1100 0000	Start Bit
Start bit is always ‘1’	

1100 0000 0000 0100 0000 0000 0011 0000 0011 0010 0000 1111 1111 1010 1100 0000	CDS
CDS bit is always ‘0’	

1100 0000 0000 0100 0000 0000 0011 0000 0011 0010 0000 1111 1111 1010 1100 0000	POSITION
32 bits of POSITION = 0x00181907 = 1579271 counts	

1100 0000 0000 0100 0000 0000 0011 0000 0011 0010 0000 1111 1111 1010 1100 0000	STATUS
2 STATUS bits = 0x03	

1100 0000 0000 0100 0000 0000 0011 0000 0011 0010 0000 1111 1111 1010 1100 0000	CRC
6 CRC bits = 0x3D	

1100 0000 0000 0100 0000 0000 0011 0000 0011 0010 0000 1111 1111 1010 1100 0000	Ignored
Ignored bits	

Calculated encoder position = 1579271 counts * 0.05 um = 78963.55 um = 78.96355 mm
Status bits are 11. As they are active low, encoder operation is correct (no error, no warning).

Example 2

Encoder used in example 2:

Type: Linear absolute encoder, BiSS output
Resolution: 1 um
Position length: 26 bits
Status length: 2 bits (active low)
CRC length: 6 bits, polynomial $x^6 + x^1 + 1$ (Represented also as 0x43), inverted

Response to the “4” command: **c004c9ba71753000** (hex)
is decoded as:

Position = 0x19374E2 (26440930 decimal) = 26.44093 mm

Status = 0x03 (11 binary) = no error, no warning

CRC = 0x2A

6-bit CRC calculation with 0x43 polynome for BiSS

BiSS communication offers a CRC value to check the correctness of the data read from the encoder. This chapter gives an example of the CRC calculation on the receiver side. The CRC calculation must always be done over the complete set of data. The polynomial for the CRC calculation is $P(x) = x^6 + x^1 + 1$, also represented as 0x43.

Code example:

```
u8 tableCRC6[64] = {
                    0x00, 0x03, 0x06, 0x05, 0x0C, 0x0F, 0x0A, 0x09,
                    0x18, 0x1B, 0x1E, 0x1D, 0x14, 0x17, 0x12, 0x11,
                    0x30, 0x33, 0x36, 0x35, 0x3C, 0x3F, 0x3A, 0x39,
                    0x28, 0x2B, 0x2E, 0x2D, 0x24, 0x27, 0x22, 0x21,
                    0x23, 0x20, 0x25, 0x26, 0x2F, 0x2C, 0x29, 0x2A,
                    0x3B, 0x38, 0x3D, 0x3E, 0x37, 0x34, 0x31, 0x32,
                    0x13, 0x10, 0x15, 0x16, 0x1F, 0x1C, 0x19, 0x1A,
                    0x0B, 0x08, 0x0D, 0x0E, 0x07, 0x04, 0x01, 0x02};

u8 crcBiSS (u32 data)
{
        u8 crc;
        u32 tmp;
        tmp = (data >> 30) & 0x00000003;
        crc = ((data >> 24) & 0x0000003F);
        tmp = crc ^ tableCRC6[tmp];
        crc = ((data >> 18) & 0x0000003F);
        tmp = crc ^ tableCRC6[tmp];
        crc = ((data >> 12) & 0x0000003F);
        tmp = crc ^ tableCRC6[tmp];
        crc = ((data >> 6) & 0x0000003F);
        tmp = crc ^ tableCRC6[tmp];
        crc = (data & 0x0000003F);
        tmp = crc ^ tableCRC6[tmp];
        crc = tableCRC6[tmp];
        return crc;
}
```

Recommended literature:
- Painless guide to CRC error detection algorithm; Ross N. Williams.
- Cyclic Redundancy Code (CRC) Polynomial Selection For Embedded Networks; P. Koopman, T. Chakravarty

Document Revisions

Rev.	Date	Page	Changes
00	2015-08-03	-	First issue