Dynamic Motor Motion
Technology Corporation

DMM Technology Corp.
120 - 21320 Gordon Way
Richmond, BC V6W 1J8
Canada

TEL : +1 604 370 4168
MAIL : info@dmm-tech.com | sales@dmm-tech.com
WEB : www.dmm-tech.com

ABS-16-GP1 ABSOLUTE ENCODER SPECIFICATION

Features

- Non-Contact Shaft-Sensor Coupling
- 16-bit resolution, Single-Turn
- Magnetic Sensor Type - Robust Against Environment
- Internal Supply Voltage Filter
- Active Sensor Optimization - Ultra High Reliability
- 4-Wire Interface
- 50mm diameter, 19mm height body (00 Model)
- 33mm diameter, 18.3mm height body (01 Model)
- 32-Byte EEPROM memory

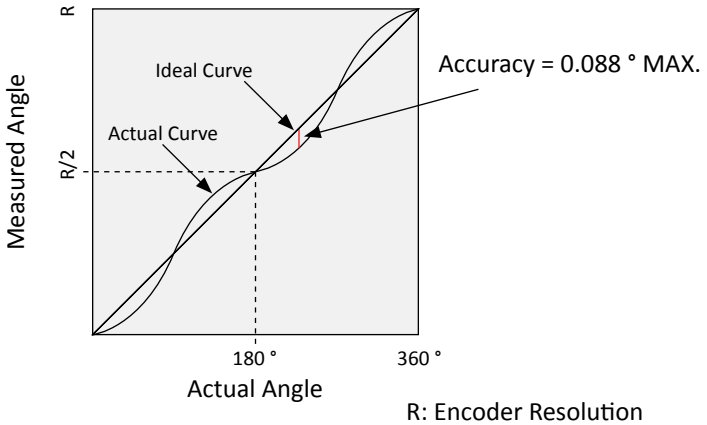
ABS-16-GP1C5-00

ABS-16-GP1C5-01

Specification

Encoder Type	Absolute
Resolution	16-bit (65,536ppr) Single Turn
Accuracy	12-bit Max.
Positive Direction	Clockwise (viewed from motor shaft)
Rotor Inertia	0.01 kg·cm^2
Operating Temperature	-30 °C ~ +100 °C
Storage Temperature	-40 °C ~ +100 °C
Relative Humidity	< 85%
Weight	0.2 kg
Protection	IP00
Mount Tolerance	0.1mm (Radial + Axial)
Shaft Axial Movement	± 0.5 ° Max.
Interface Circuit	RS485 Differential Proprietary Protocol
Supply Voltage	+5 VDC ± 10 %
Supply Current	150 mA Max.
Pulse Rise/Fall Time	50 ns MAX.
Insulation Voltage	800V
Sensor Type	Magnetic - InAs Hall
Magnetic Source	Neodynium [NdFeB]

Accuracy Measurement:



R: Encoder Resolution
*Accuracy mathematically calculated within 4096 pulses.

Part Number

ABS - 16 - GP1 C5-00

Size Type	
00	50mm Diameter
01	33mm Diameter

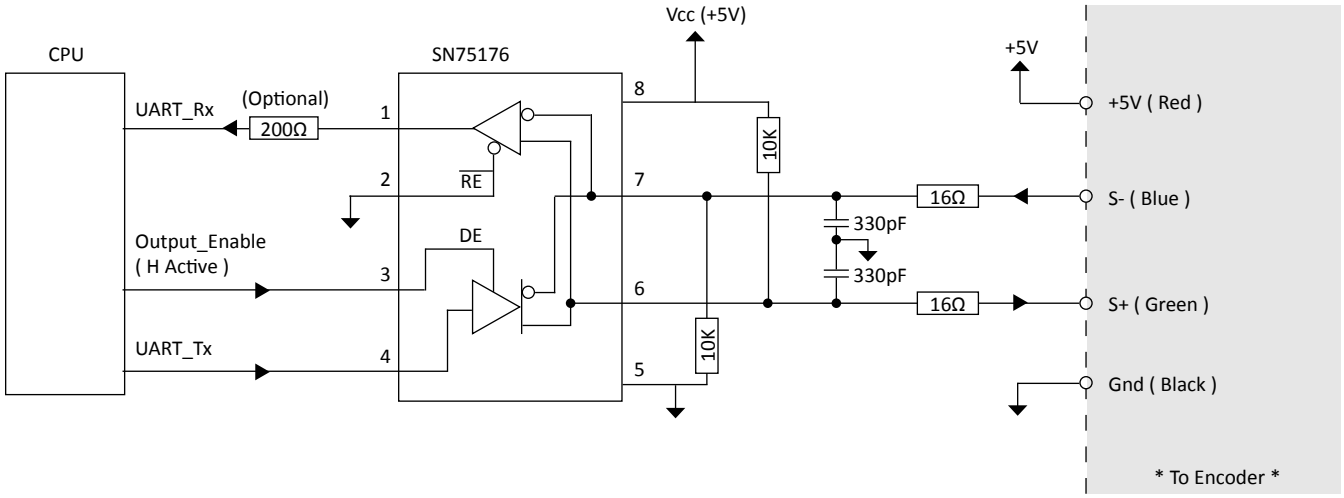
C5 - Firmware Version

GP1 - General Purpose

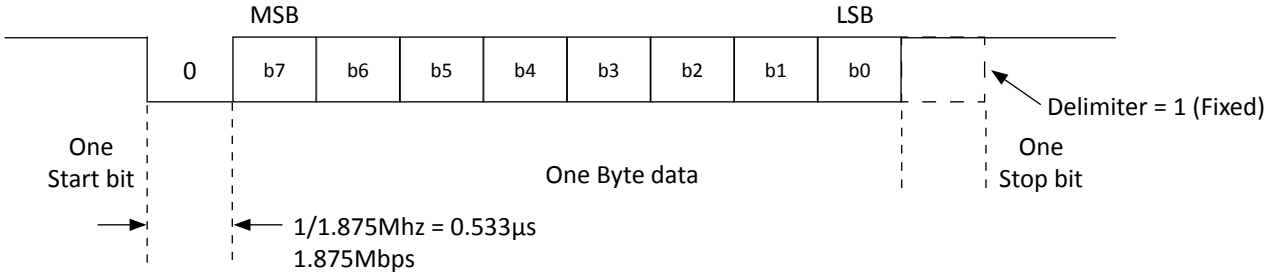
16-bit Resolution

DMM Absolute Magnetic Encoder

Interface



General Single-Byte Data Signal Framing

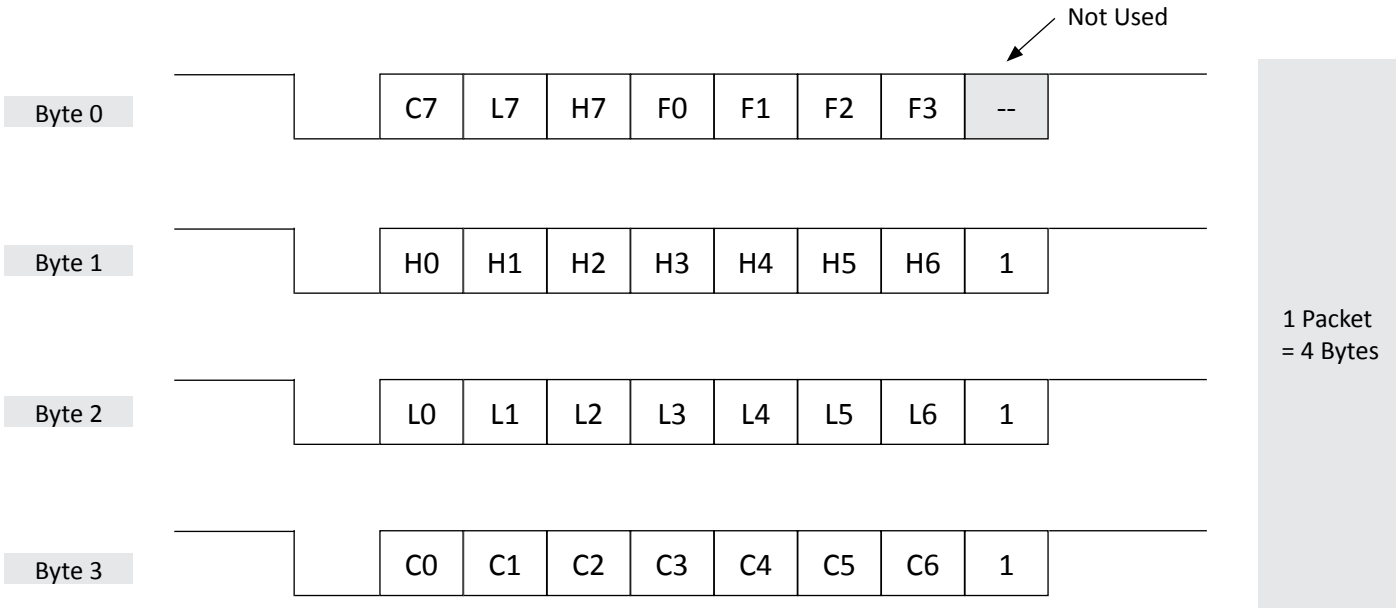


Data Packet Structure

- Each frame consists of 8 data bits and 2 start/stop bits (as shown above).
- Each data packet consists of 4 frames (4 bytes).

16bits position = H7 H6 H5 H4 H3 H2 H1 H0 (High Byte) , L7 L6 L5 L4 L3 L2 L1 L0 (Low Byte)

Function Code (4 bits) = F3 F2 F1 F0
CRC Code (8 bits) = C7 C6 C5 C4 C3 C2 C1 C0



Function Code Description:

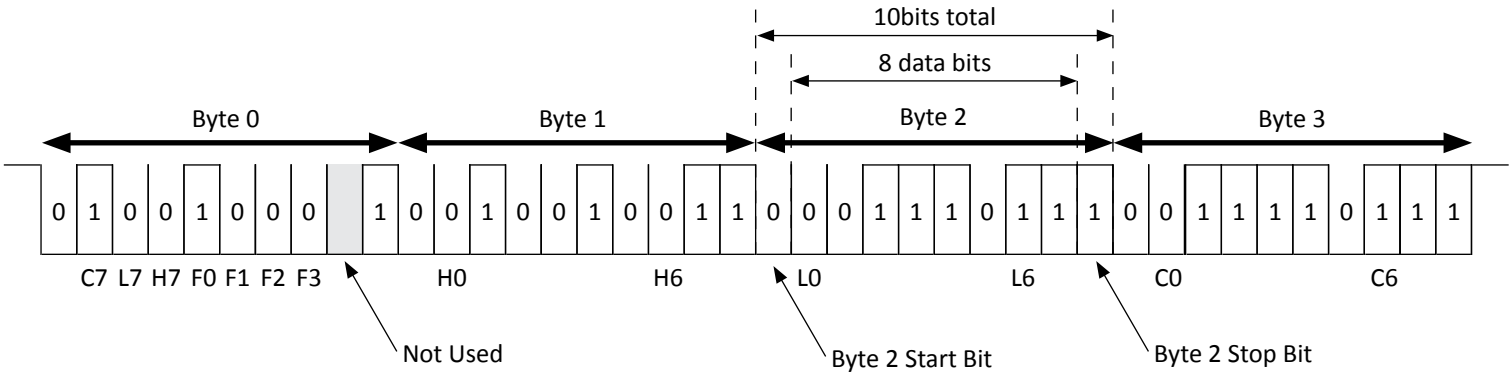
Data				Function
F3	F2	F1	F0	
0	0	0	0	Normal encoder function. Data send from encoder is normal 16-bit position data.
0	0	0	1	Write data to encoder EEPROM. 16-bits data: Address8bit (High Byte) , Data8bit (Low Byte) - Address8bit = 0~31 EEPROM address - Data8bit = Write Data byte - Drive should send write command packet 4 times to save data
0	0	1	0	Read data from encoder EEPROM. 16-bits data: Address8bit (High Byte) , Data8bit (Low Byte) - Address8bit = 0~31 EEPROM address - Data8bit = Write Data byte - When encoder returns data, returned data packet also has function code 0010 - Drive should send read command packet 4 times to save data - Encoder returns read data packet 1 time

Example Data Packet

16bits position = 4700 (0001 0010 0101 1100) = Write 92 (0101 1100) to EEPROM address 18 (0001 0010)

Function = Write Data to Encoder = (0001)

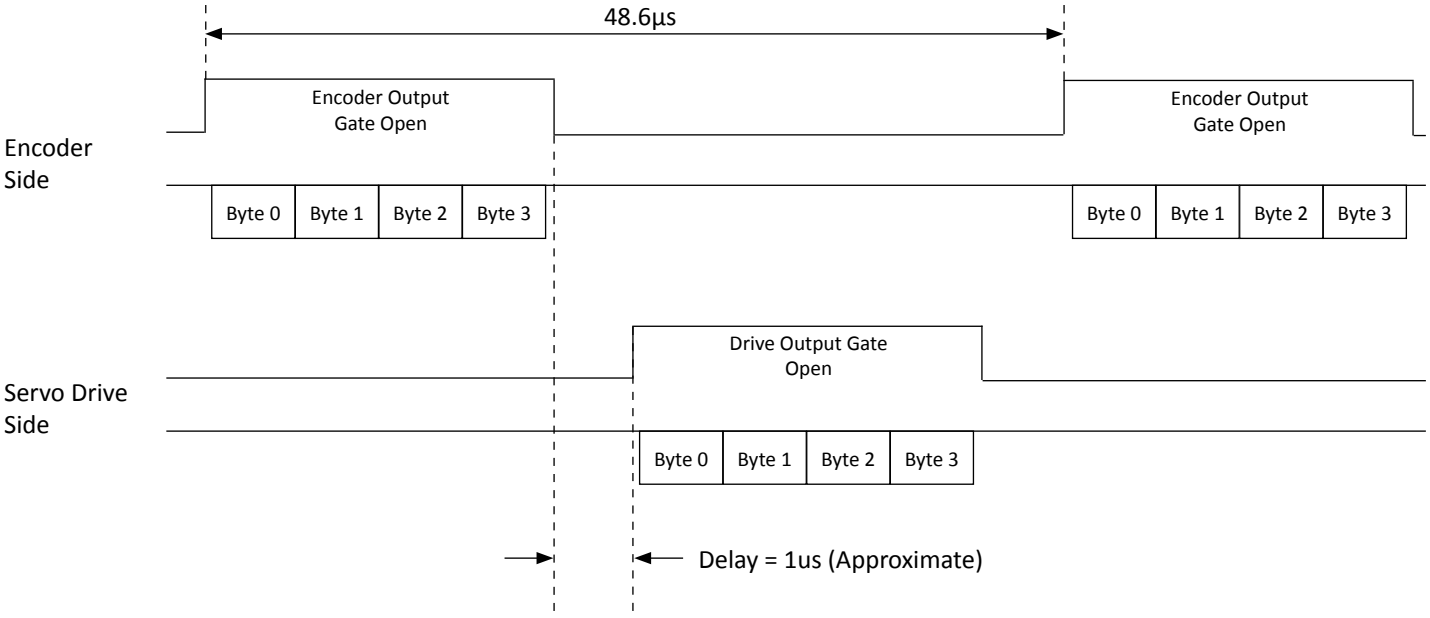
CRC = 222 (1101 1110)



Communication Timing

As soon as the power is turned ON, the ABS-16-GP1 encoder begins outputting data packet.

Example communication with AC Servo Drive:



Servo Motor Tuning Procedure

1. Mount the encode onto servo motor shaft. Servo motor shaft must be free (no load attached). Power ON encoder and servo drive.

2. Apply 3-phase current to motor coil. Motor must be rotating in clockwise direction (from shaft side view). When rotating clockwise, the absolute position increases. Max current (Imax) equal to 1/3 of motor peak current. For example, if servo motor peak current is 10A, Imax = 3.3A.

   The cycle must be started at Ia = Imax/3, Ib=Ic= - Ia/2

   Period of sinusoidal current equals 2/P seconds, where P = motor pole pair number. Motor shaft rotates at 0.5revolutions per second. Rotate for 16 cycles.

3. When the 16 cycle finishes, hold the current at Ia = Imax/3, Ib=Ic= - Ia/2. Hold this current for 3 seconds.

4. After 3 seconds holding, send 0xAABB data command to encoder 4 times. When sending 0xAABB command, the format is the same as standard data packet with 4 bytes with 0xAABB as the data. Function code is 0000. CRC is calculated as normally, from 0xAABB.

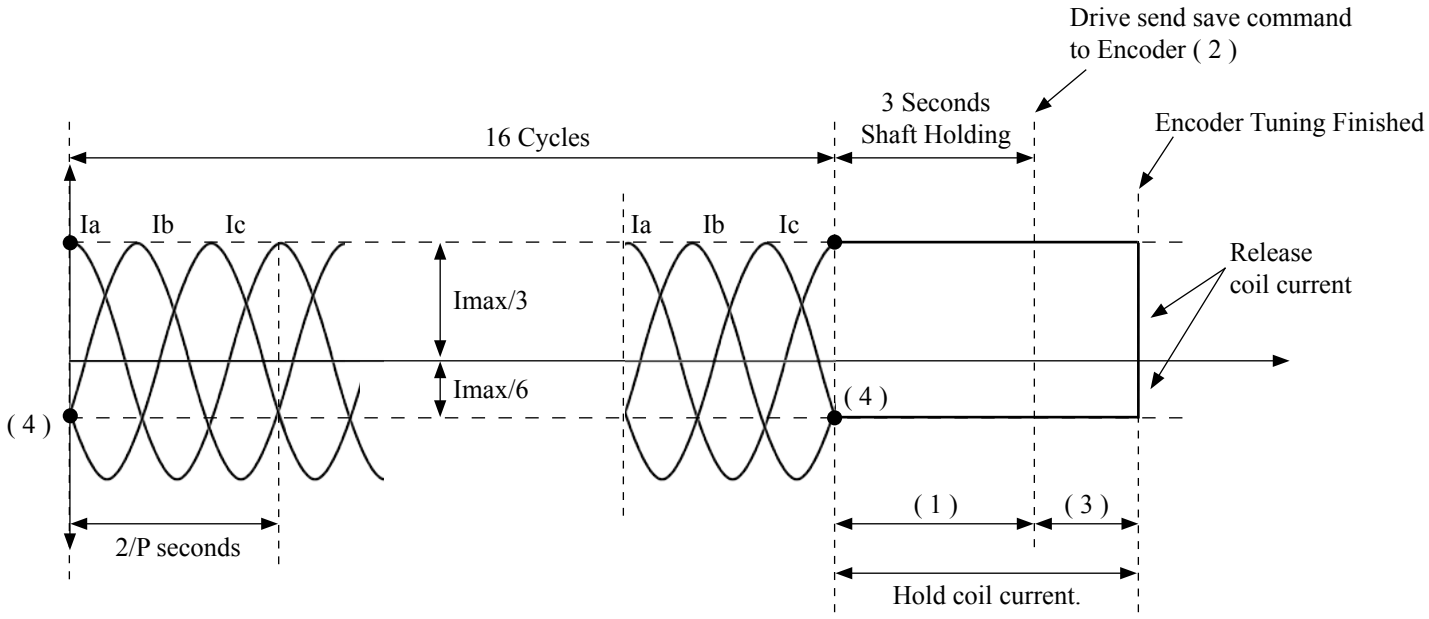
   Each command does not have to be continuous, but can be up to 50ms apart. It is recommended to send the command after each second encoder position is received. So after one 0xAABB command is sent, wait two encoder data reads before sending next 0xAABB command.

   After 4 0xAABB command is sent, wait 100ms for encoder to save absolute zero position. Then release current to motor coil. Motor shaft becomes free. Encoder tuning completed.

5. The absolute zero position should be used as the servo/torque control start reference point. The torque should be applied at absolute position ±90° (electrical angle).

CAUTION:
- Do not touch motor shaft when rotating or holding position.
- Motor must be rotating in clockwise direction when viewed from shaft.

Servo Motor Tuning Procedure



Imax = Max current servo motor.
P = Motor Pole Pair Number.

(1) During 3 seconds shaft holding, the current must be:
Ia = Imax/3
Ib= Ib = -Ia/2

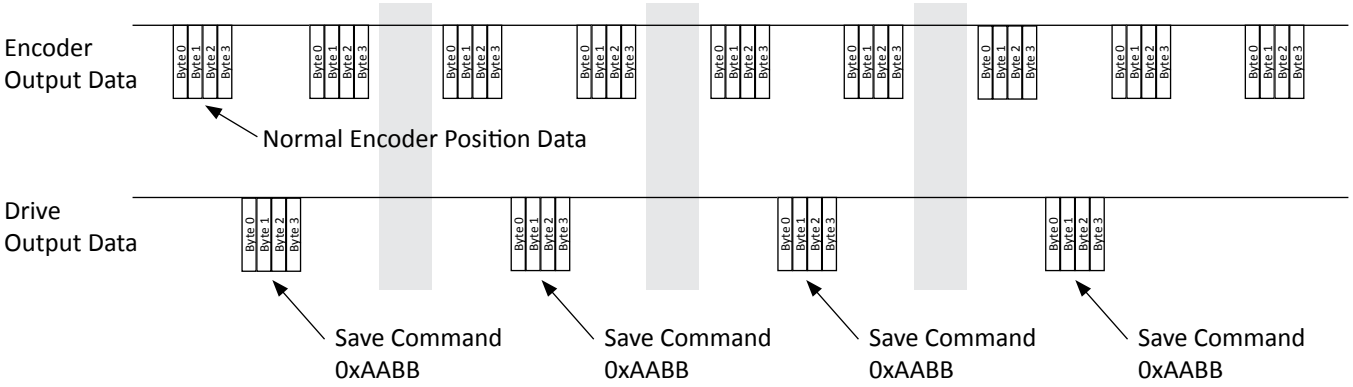
(2) The servo drive must send the save data command “0xAABB” to the encoder 4 times. The encoder needs to receive 0xAABB data command 4 times to save tuning parameters.

(3) 100ms encoder saving parameter time.

(4) The cycle must be started and finished at Ia = Imax/3, Ib=Ic= - Ia/2.

Save Command “0xAABB“ Timing

The 0xAABB command should be sent after every second encoder data is read. So after the one save command 0xAABB is sent, wait two encoder position data before sending next 0xAABB command.



Operation Example 1 - Read Encoder Position (Normal Operation)

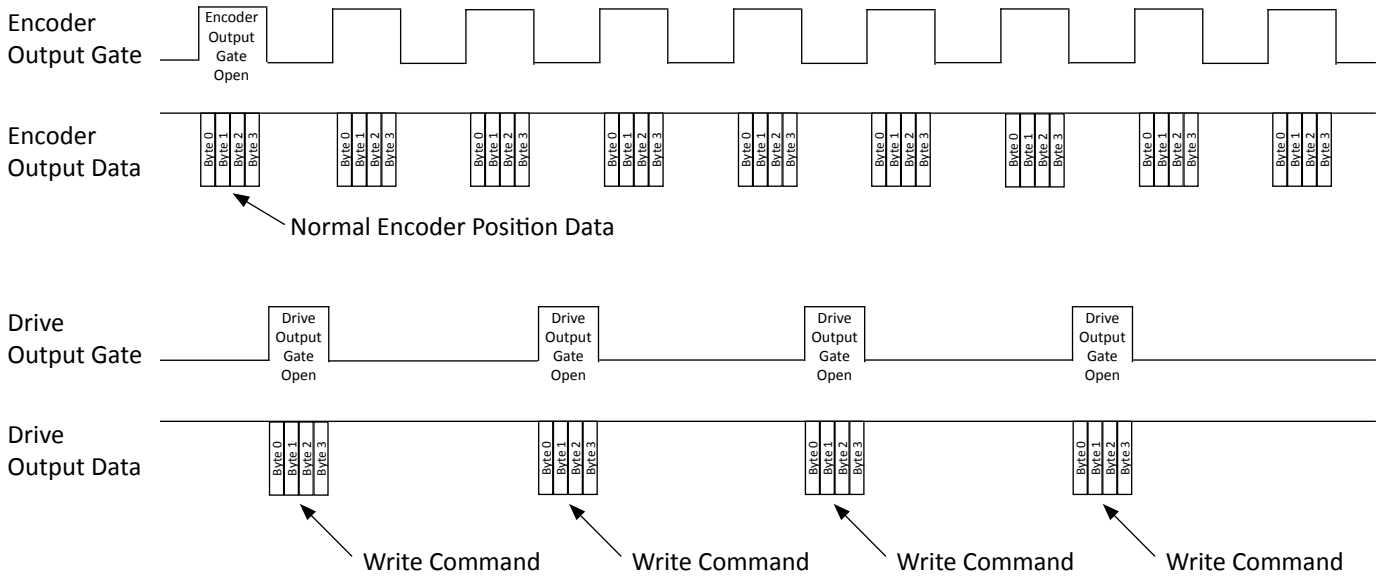
1. When encoder is sending normal position data, the function code bits is 0000
2. The encoder begins transmitting position data as soon as power is turned ON

Operation Example 2 - Write Data to Encoder

1. Controller must send write command 4 times with function code bits 0001

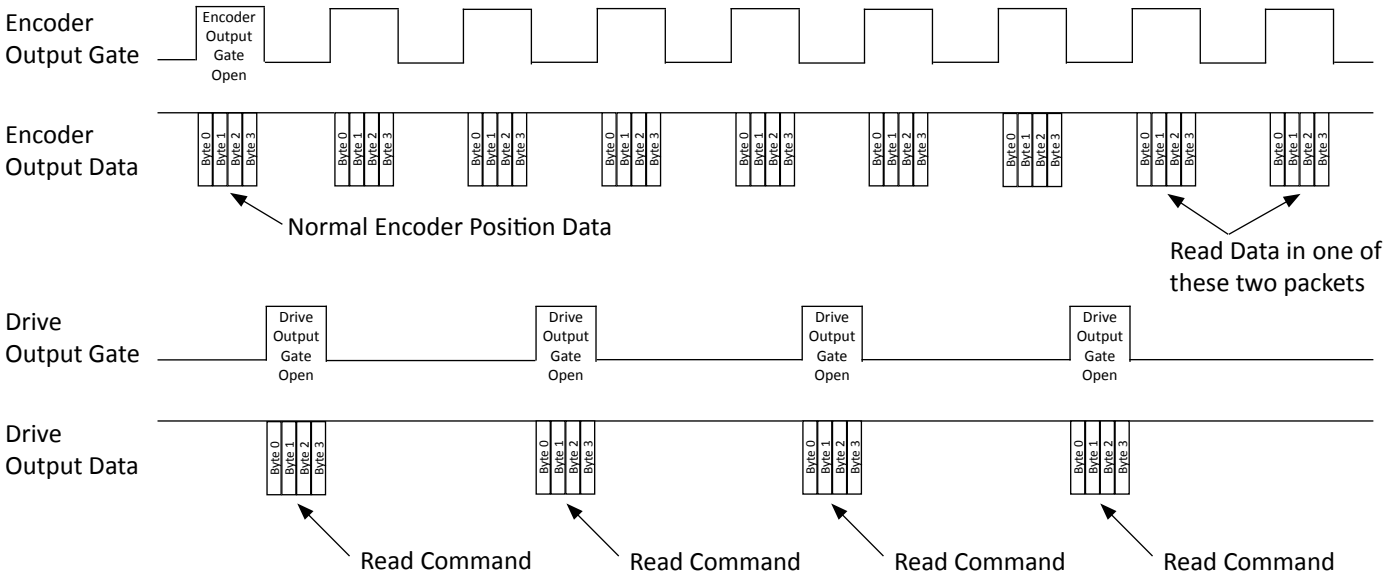
* Recommended: Write command should be sent between *every other* encoder data. Do not send consecutively after every encoder data. After one write command is sent, wait two encoder data before sending next write command.

2. Encoder saves data to eeprom address after write command received 4 times. Wait 100us for encoder to write data



Operation Example 3 - Read Data from Encoder

1. Controller must send read command 4 tiles with function code bits 0010
* Recommended: Write command should be sent between *every other* encoder data. Do not send consecutively after every encoder data
2. The encoder replies with function code 0010 to indicate that this data is memory read data
3. Depending on the timing, the encoder's memory read data is in the next sent data packet, or the following one.
4. Controller should monitor the function code. If the function code is 0000, then the data is absolute position. If the function code from the encoder is 0010, then the data is memory read data.



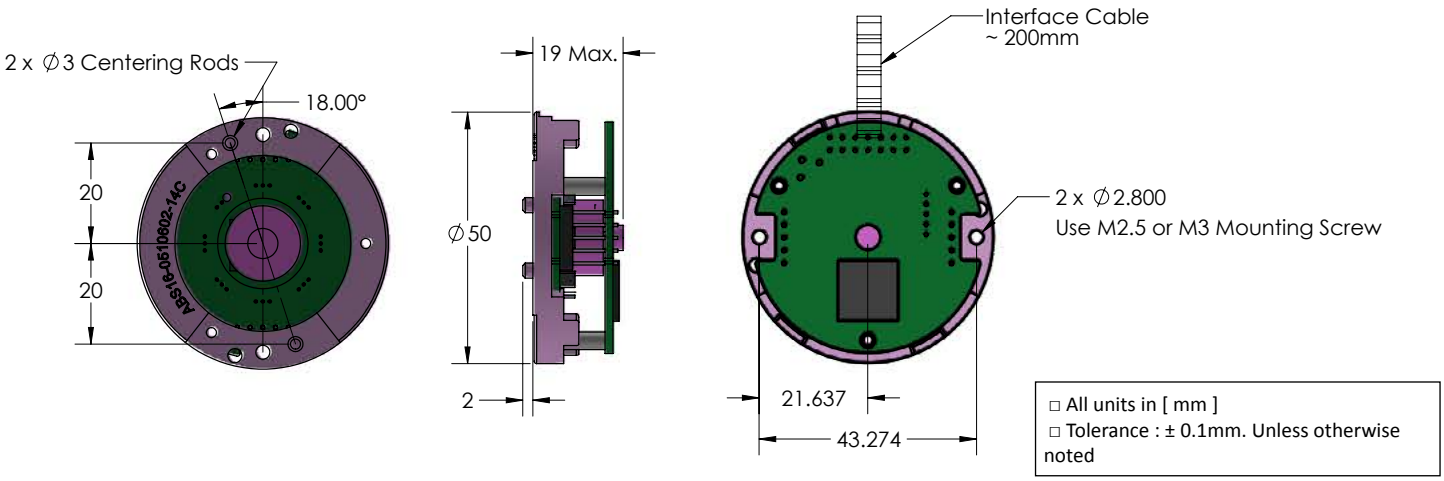
CRC Calculation Sample Code

The receiver should use the following sample code to generate the CRC code in Byte 3.

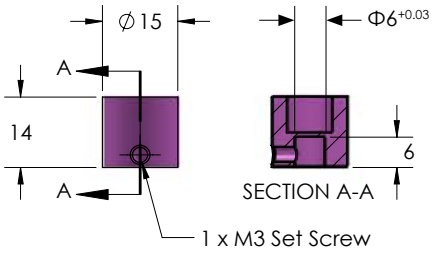
```
unsigned char SGenerate8BitsCRC(Int16 Data16)              /*Input 16bits Data, make 8 bits CRC for X^8 + X^3 + 1    */
{
   Int16 StateX,StateY,i;
   unsigned char CRC8bits;

   StateX = Data16&0xFF00;   //High 8bits for the States
   Data16 = Data16<<8;
   for(i=0;i<8;i++)
   {
      StateY = StateX<<1;

      if((StateX & 0x8000) != 0)
      {       //X7 = 1

         if((Data16 & 0x8000) != 0)
            StateY = StateY & 0xFE00;       //D15=1 so X0=0
         else
            StateY = StateY | 0x0100;       //D15=0 so X0=1

         if((StateX & 0x0400) != 0)
            StateY = StateY & 0xF700;       //X2 = 1 so X3 = 0
         else
            StateY = StateY | 0x0800;       //X2 = 0 so X3 = 1
      }
      else
      {       //X7 = 0

         if((Data16 & 0x8000) != 0)
            StateY = StateY | 0x0100;       //D15=1 so X0=1
         else
            StateY = StateY & 0xFE00;       //D15=0 so X0=0

         if((StateX & 0x0400) != 0)
            StateY = StateY | 0x0800;       //X2 = 1 so X3 = 1
         else
            StateY = StateY & 0xF700;       //X2 = 0 so X3 = 0
      }

      StateX = StateY;
      Data16 = Data16<<1;
   }

   StateX = StateX >> 8;
   CRC8bits = StateX & 0x00FF;
   return(CRC8bits);
}
```

Dimensions

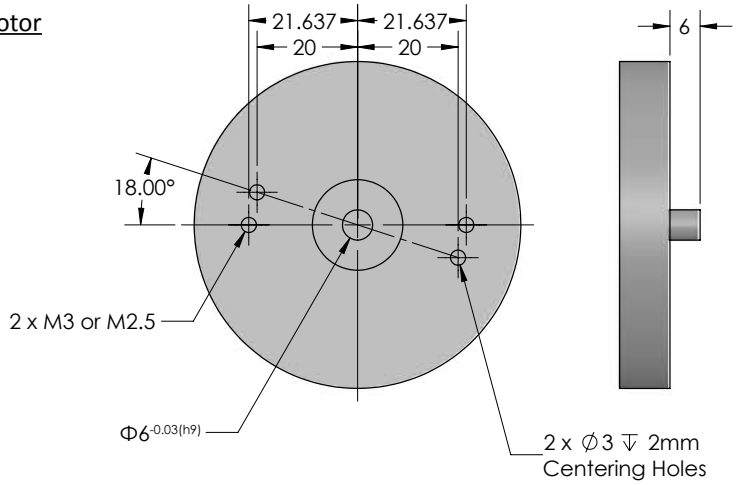
ABS-16-GP1C5-00 Encoder



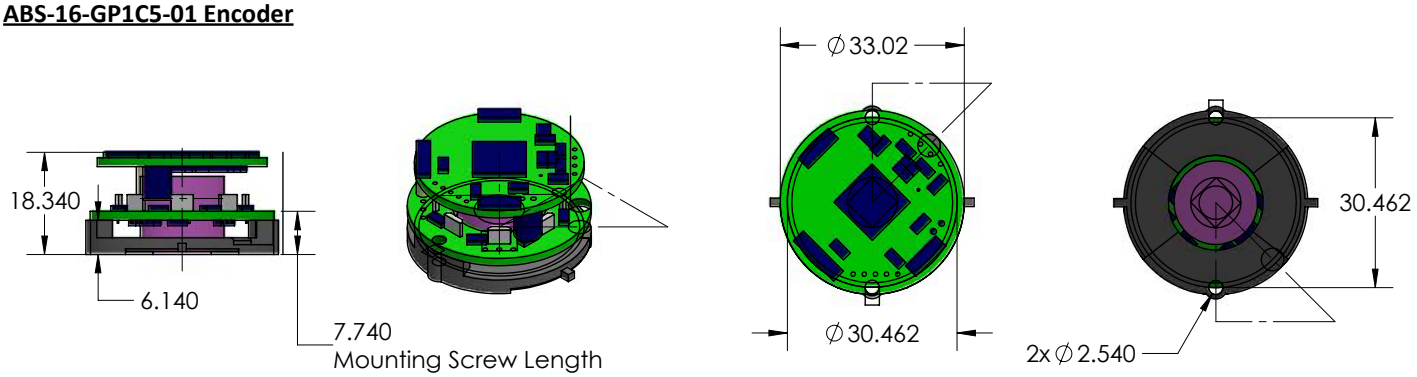
Magnet



Motor

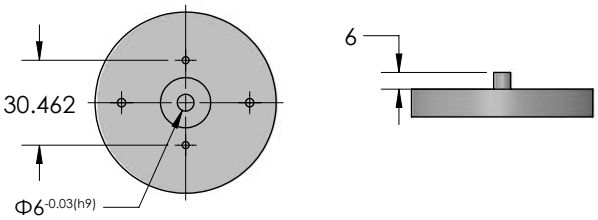


ABS-16-GP1C5-01 Encoder



* Use 2 x M2 or M2.5 to mount. Mounting screw secures both encoder PCB and base to motor.

Motor



Mounting Procedure

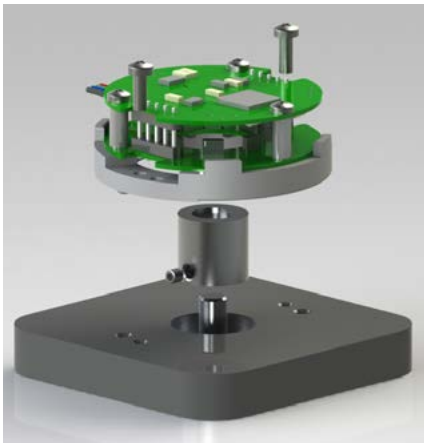
ABS-16-GP1C5-00

Step 4) Apply thread lock on M3 mounting screw. Secure encoder body on motor body. M3 mounting screw tightening torque: 1.2Nm

Step 3) Install encoder body over magnet onto motor body. Make sure centering rod on encoder is fully inserted into centering holes.

Step 2) Mount magnet onto motor shaft, then secure using M3 set screw (set screw supplied with the magnet). M3 set screw tightening torque: 0.7Nm

Step 1) Apply glue to motor shaft or magnet inside hub. Recommended glue is Loctite 648 or equivalent.



ABS-16-GP1C5-01

Step 4) Apply thread lock on M2.5 mounting screw. Secure encoder body on motor body. M2.5 mounting screw tightening torque: 0.7Nm

Step 3) Install M2.5 mounting screw through encoder pcb and base. The M2.5 mounting screw secures both the encoder pcb and base to the motor body. Recommended mounting screw: M2.5x10mm.

Step 2) Mount magnet onto motor shaft, then secure using M3 set screw (set screw provided with the magnet). M3 set screw tightening torque: 0.7Nm

Step 1) Apply glue to motor shaft or magnet inside hub. Recommended glue is Loctite 648 or equivalent.

Interface Cable

- □ Shielded, Twisted Pair Lead Wire
- □ Heat-resistant pvc sheath 105 °C 30V
- □ 14-position, 0.4mm O.D. copper conductor (AWG28)
- □ Connector: HILP-04V-1-S (JST)
  Pin: SHIF-01T-P0.5 (JST)

Twisted Pair	Color	Data
1	Red	+5 VDC
	Black	GND
2	Green	S +
	Blue	S -