Non-Conformance Addendum

This document describes the known non-conformances with the BarrettHand. A non-conformance is any aspect of the BarrettHand that may operate differently than the user expects. Barrett Technology is aware of the following non-conformances; and, where reasonably possible, will work to upgrade customers who are covered under Barrett's Warranty and Subscription Service. If you have any questions or concerns, please contact Barrett Technology at 617-252-9000 or <service@barrett.com>.

1. **CAUTION:** Never unplug the hand cable that connects the Power Supply to the BarrettHand while power is applied to the hand. Failure to turn power off before disconnecting the cable may damage the hand electronics.

2. **CAUTION:** You must strain relieve the hand cable that connects the Power Supply to the BarrettHand. The black, 10-pin connector on the exposed electronic board will not support large forces or torques. It is very important to strain relieve this cable close to the black connector. If the connector disconnects during operation, it may damage the hand electronics.

3. The spread motor (4) occasionally makes a buzzing noise when holding position. This buzzing noise is attributed to the control parameters for the motor. If a small servo-stiffness reduction is acceptable, then you may adjust the filter parameters. For the BH8-262, these parameters are FPG, FDZ, and FIP, to reduce the noise level. Leave FPG at its default value of 100 and set FDZ to 251 and FIP to 0 to reduce the buzzing noise. For the BH8-280, the parameters are KP, KD, and KI; try reducing KP until the noise is eliminated.

4. The spread may open only partially in response to an "all open" command when the hand begins from the full closed position (all fingers and spread closed). When a motor detects no motion for a pre-specified amount of time, stored in the firmware parameter TSTOP, it stops commanding the motor to move. Thus, slight interference between the tips of the fingers may prevent the spread from opening completely. Interference between fingers is sensitive to the relative finger and spread velocities. A simple workaround is to issue a "spread open" command (SO) after issuing an "all open" command to continue spread motion in the open direction.

5. The finger tensioner may loosen after extensive cycling. Follow the procedure in the BH8-Series User Manual to pretension the cable properly.

6. You may notice one of the fingers leading or lagging behind the other fingers. This is due to normal differences in joint friction. Lubricating the slower fingers may help reduce the friction and increase the velocity. Barrett does not make an attempt to control the fingers in Lock Step, nor do we intend to.

   If this type of motion is desired, it is possible to use the RealTime mode to produce these results. This requires control of individual finger velocities and the relative positions of each finger. For example, your control law can control the velocity of F1 ($\dot{\theta}_{F1}$) while minimizing the position differences ($\theta_{F1} - \theta_{F2}$) and ($\theta_{F1} - \theta_{F3}$).

7. During continuous cycles, you may notice an oily film forming along the Finger/Motor interface. This is due to the reduced viscosity of the grease at operating temperatures and will not interfere with the hand's operation.

8. It may be necessary to clamp the bench stand to a secure surface during high-speed hand operation to prevent the bench stand from moving.

9. **BH8-262:** Setting supervisory properties during RealTime mode results in a COM Port Read Timeout error.

10. **BH8-262:** The commanded velocity is more precise than the feedback velocity. The commanded velocity sent to the BarrettHand is 12-bits whole number and 4-bits fraction. The feedback velocity returned by the BarrettHand in RealTime mode is 8 bits whole and no fraction, resulting in a whole number. It is possible to differentiate the feedback position to determine the velocity more accurately. Note the feedback position non-conformance item Number 13.

11. **BH8-280:** The feedback velocity is set to the commanded velocity. It is possible to differentiate the feedback position to determine the actual velocity more accurately.

12. **BH8-262:** The Loop Feedback Delta Position (LFDP) value during RealTime mode may appear incorrect if the user does not monitor the last reported position. The last reported position is the position that was last used to determine the LFDP. The LFDP is calculated as the difference between the last reported position and the present Absolute Position.

Example: Enter RealTime mode, command Finger 1 to close at 65, and request Feedback Delta Position until the position is 17000. The last reported position is 17000. Exit RealTime mode and issue an Open command to Finger 1. Enter RealTime mode and command Finger 1 to close at 65 and request Feedback Delta Position. The last reported position was 17000 and the present Absolute Position is 0, therefore the first Feedback Delta Position would be calculated as –17000, but since the firmware can only send values between 127 and –127, it reports the largest negative value, -127. The Feedback Delta Position will continue to be -127, until all of the position difference has been reported. Refer to the example in the BH8-255 User Manual for more information.

13. **BH8-262:** The position returned to the user during RealTime mode is incorrect. The value is computed using three bytes of information, Most Significant Byte (MSB), Middle Significant Byte (MidSB) and Least Significant Byte (LSB). All bytes of information should be unsigned. However, the LSB is added to the absolute position as a signed byte instead of an unsigned byte. All values between 0 and 127 of the LSB will produce correct absolute positions.

    When the LSB is between 128 and 255, the number added is the 2's complement of that value. The typical way of generating 2's complement values is by inverting the bits and adding one to the result. For example, the four bit 2's complement form of -4 would be generated by starting with 4 written in base 2, i.e. 0100, inverting the bits to give 1011 and adding one to give 1100. This produces a pulsing position signal as shown in Figure 1. The absolute position can be corrected by dividing by 256 and adding 256 if the remainder is greater than 127. Example:

    (4200/256) = 16 with 104 remaining, therefore the position is 4200
    (3000/256) = 11 with 184 remaining, therefore the position is 3256
    See Figure 1 for a comparison of data.

    Use the following equation to correct the data in your code:

```
if(Position % 256 > 127) Position += 256;
```

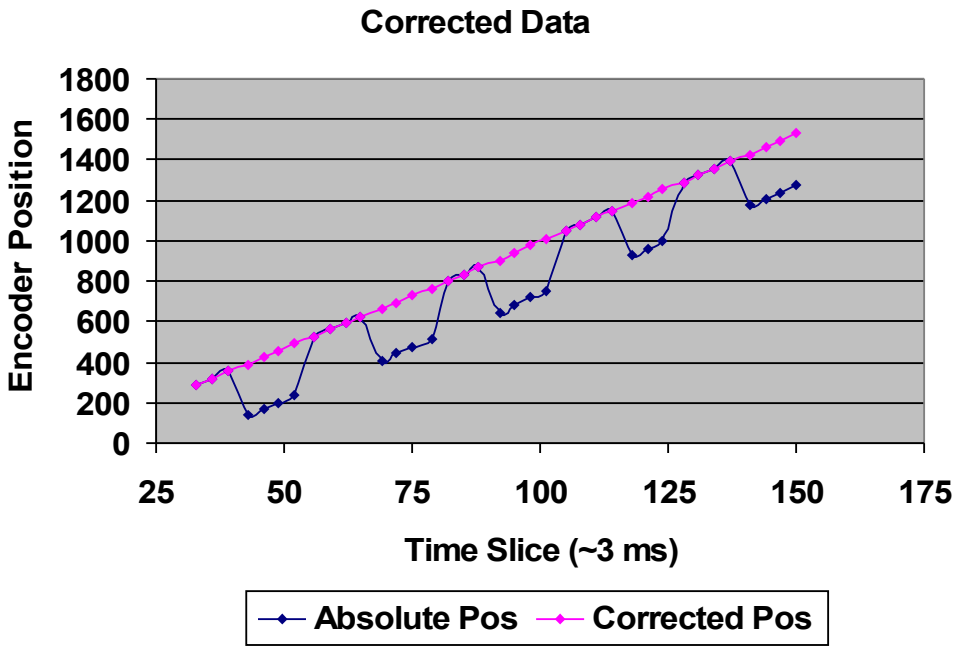

Figure 1: Plot of Corrected Position Data