

Format of Deep SOLO X messages: Argo Version Manual/Decoder V0.3
latest update: 24 April 2014
[For ROM]

An X message is used to transfer data from ISU to GS or from GS to ISU. The data is assumed to be binary and each byte can have any value from 0x00 to 0xff. The format of the message is the same regardless of direction of transmission:

Xnnmmddp<data>\$cc>

X = the character X

nn = number of data characters in the message following after nn. The count does not include X ,nn, or anything from \$ to the end >. The count is in 2 binary bytes with MSB first and LSB second.

mm = serial number of SOLOII. The SN is in 2 binary bytes with MSB first and LSB second.

dd = the dive number in 2 binary bytes with MSB first and LSB second.
: Dive number begins at -1 for the start-up, increments to 0 for the test dive, increments +1 for all normal' (0xE2) dives.

p = one-byte packet ID index, range 0 to 255. Used to identify multiple X messages within a dive cycle. The data for each dive cycle starts with p=0.

<data> = binary data characters. The length of <data> = nn -5. The contents of the <data> section is described below.

\$ = a dollar sign delimiter at start of the checksum

cc = the 8 bit byte-wise checksum from X to the byte preceding the \$. The 8 bit sum is coded as 2 4bit nibbles. The binary value of a nibble is converted to a visible character by adding 0x30. Thus a value of 0x0 -> 0x30 = character '0', 0x1 -> 0x31 = '1', 0xe -> 0x3e = '>', and 0xf -> 0x3f = '?'.

> = a > delimiter at end of checksum which also serves as a prompt to GS that the ISU is done transmitting and that the GS may now transmit to ISU.

The remainder of this document describes the format of the <data> portion of the message sent from SOLOII to the ground station (**GS**). The format of commands from **GS** sent to SOLOII will be described in another document.

Highlights in document

Fields that are moved relative to the previous float version are highlighted in cyan

New/Modified fields relative to the previous version are highlighted in yellow

Overview of changes

Additions/mods by JTS 10Jan13

v0.3 Additions/mods by JTS 21Apr14: search for Apr14.

Fall, Rise Diagnostic data: time resolution change: 1 count = 10 s (old was 1 s/count).

Pump Diagnostic: Added time-of-pump variable (resolution = 1 count = 60s).

Also, multiple pump packets are allowed (sort by time-of-pump variable). Each packet can have a maximum of 26 pump events.

Drift Time in the Argo information packet is now 5 minutes/count (old was 1 minute/count).

The <data> section contains information from multiple sensors. Data from successive sensors are separated by a semicolon (';' = 0x 3b); the final sensor is terminated by a ';' (immediately preceding the \$ delimiter).

```
IDjj<sensor_data>;
ID = one-byte sensor ID code.
jj = Number of bytes for this sensor. The count includes ID, jj, and the trailing ;.
The count is in 2 binary bytes with MSB first and LSB second.
<sensor_data> = binary data characters. The length of <sensor_data> = jj-4 bytes, and its
contents are described below for each sensor.
; = delimiter at the end of each sensor's data.
```

The **ID** byte is divided into two 4-bit nibbles. The MS nibble identifies the sensor and the second nibble specifies the message number for that sensor. For example, the ID for first Pressure message is 0x10, the second is 0x11, the third 0x12, etc. For a 1000 sample profile, there will be 6 messages for each of the pressure, salinity and temperature sensors.

Sensor	ID byte(hex)	
GPS	00	fix at end of first diagnostic dive at start of mission
GPS	01	fix at before leaving surface
GPS	02	fix at end of normal profiling ascent
GPS	03	fix following mission abort
GPS	05	fix during BITest
Pressure	1x	depths of CTD readings (scaled 1st difference) x=0-7: upper ocean bin averaged data x=8-F: deep spot sampled data
Temperature	2x	depth series of temperature (scaled 1st difference) x=0-7: upper ocean bin averaged data x=8-F: deep spot sampled data
Salinity	3x	depth series of salinity (scaled 1st difference) x=0-7: upper ocean bin averaged data x=8-F: deep spot sampled data
Fall Rate	40	series of time, depth during SOLO II downward profile
Rise Rate	50	series of time, depth from drift depth to surface
Pump Series	60	pressure, time, voltage, current, vacuum for each pump
Time	7x	depth series of time-referenced to 0x40 (scaled 1st difference): x=8-F: Only returned during deep spot sampling
High Resolution Pressure	9x	High Resolution Pressure (scaled 1 st difference)
High Resolution Temperature	ax	High Resolution Temperature (scaled 1 st difference)
High Resolution Salinity	bx	High Resolution Salinity (scaled 1 st difference)
Mission EEPROM	d0	ASCII dump of mission parameters in EEPROM
Engineering	e0	diagnostic data in first diagnostic dive
Engineering	e2	engineering data in normal profiling dive
Engineering	e3	engineering data following mission abort
Engineering	e5	engineering data BIT test
EEPROM dump	d0	Float configuration dump
Argo Data	f0	Mission parameter list
Test pattern	f1	<i>ID reserved, format not yet defined</i>

GPS data (ID=0x00, 0x01, 0x02, 0x03, 0x05)

The LS nibble of the ID indicates in what phase of the mission the fix was taken. The remainder of the data is the same for all mission phases. The length of GPS data is in bytes 1 and 2. GPS fix data starts in byte 3:

Byte	Contents
0	Mission phase: 0 = 1st diagnostic dive at the start of a mission 1 = beginning of normal dive cycle (just before leaving surface) 2 = end of a normal dive cycle 3 = following mission abort 5 = during BITest
1-2	Number of bytes in the message, 24 = 0x18 with the format as described here
3	0 if fix is invalid, +2 if longitude is East, -2 if longitude is West
4-7	Signed latitude degrees * 1e7
8-11	Signed longitude degrees * 1e7 range (+180 to -180 degrees)
12-13	GPS week (traditional GPS week =0 to 1023 in LS 10 bits; rollover fix in MS 6 bits)
14	GPS day of week, 0=Sunday, 6=Saturday
15	UTC hour
16	UTC minutes
17	Time to get fix = (seconds to get fix)/10 , range 0 to 255 = 0 to 2550 seconds
18	Number of satellites used in fix
19	Minimum signal level
20	Average signal level
21	Maximum signal level
22	10*Horiz. dilution of precision
23	; terminator (0x3B)

Pressure data (ID=0x1n)

Temperature data (ID=0x2n)

Salinity data (ID=0x3n)

Time data (ID=0x7n, n=8:f)

Profile data from the pressure, temperature, and salinity sensors are all processed in the same way and the message format differs only in the ID code. The SeaBird CTD takes a profile as the SOLO-D ascends/descends and stores the values internally. When SOLO-D reaches the surface/park pressure, it takes the continuously sampled data (0-2000dbar) data from the CTD and block averages it in depth into **PRO_BINS** (= 1000) bins. Data sampled deeper than 2000dbar is recorded in spot sampled mode and returned in different messages. Time of the spot sampled data is also returned.

The size of depth bins can vary with depth. The averaging scheme is determined by 5 parameters: **BLOK**, **PB1**, **PB2**, **AV1**, and **AV2**. The smallest bin size is **BLOK** decibars. Bins 0 thru **PB1-1** have a vertical extent of **BLOK** decibars. Bins **PB1** thru **PB2-1** are **AV1*BLOK** decibars tall while bins **PB2** thru **PRO_BINS-1** are **AV2*BLOK** decibars. In the special case that **PB1 >= PRO_BINS**, then all of the bins are **BLOK** decibars in extent, and the values of **PB2**, **AV1**, and **AV2** are ignored.

The data series from all channels are processed in the same way and are synchronous with each other. Each depth series is broken into sub-blocks of 25 samples, and a first-differencing method is applied to each sub-block to reduce the number of bytes required to transmit the data. Because the data series will generally be longer than the 189 bytes available in a 9601 SBD message, it is divided into multiple messages. Each message has an integral number of sub-blocks in it. The final sub-block of the time series may have fewer than 25 samples in it. The data message looks like:

IDjj<sub-block 0><sub-block 1> . . . <sub-block m>;

ID = one-byte sensor ID code and index. The low order hex digit is the message index for this sensor. For example, the pressure messages would have ID's:10,11,12...

jj = Number of bytes for this message. The count includes **ID**, jj, the data, and the trailing ;. The count is in 2 binary bytes with MSB first and LSB second.

<sub-block i> = first-differenced data from the ith sub-block where i=1,...,m =number of sub-blocks. If i<m, the sub-block will have 25 values in it and will have a total length of 22 bytes. The mth sub-block will have between 1 and 25 values and a length between 3 and 27 bytes.

Suppose a sub-block has the n values v[0], v[1],...v[n-1]. Then this sub-block will be transmitted as:

Sub-block Byte	Contents
0	one-byte scaling factor S, range = 1 to 255. S is chosen so that the scaled first-differences fit in one byte, i.e. diff <= 127.
1	MS byte of v[0]
2	LS byte of v[0]
3	LS byte of { v[1] - v[0] }/S
4	LS byte of { v[2] - v[1] }/S
...	
n+1	LS byte of { v[n-1] - v[n] }/S

The pressure series will have gaps in it if there is no valid CTD data in a block. In that case, all of the profile series will be missing the same gap. If a block average contains no valid data, that block is ignored and is not transmitted. For example, suppose the pressure bin size is 1 db and that bin 0 has P=0. Suppose there is no valid data in bin 5. Then the sub-block will contain:

1	0000	01	01	01	01	02	01	...
^	^	^	^	^	^	^	^	
								+ P=0007
								+ P=0006
								+ P=0004
								+ P=0003
								+ P=0002
								+ P=0001
								+ P=0000

Note that the 6th bin, for which P=5, will be omitted from the pressure, temperature, and salinity messages.

Each sub-block requires n+2 bytes so the longest sub-block uses 27 bytes. If each sensor has 1000 blocks then it will require 50 sub-blocks, each with 27 bytes. 8 sub-blocks will fit into each message (189/22) so 7 messages are needed per sensor. The total bytes then is 50*22 +7*16 which equals 1212. Thus a CTD profile with 1000 blocks can be sent in 3*1212 = 3636 bytes.

After the sub blocks have been reassembled into a sequence of observations, the counts are converted to scientific units by:

dBar = pressure counts * Pgain - Poff

degC = temperature counts * Tgain - Toff

psu = salinity counts * Sgain - Soff

High Resolution Pressure data (ID=0x9n n=0-7)

High Resolution Temperature data (ID=0xan n=0-7)

High Resolution Salinity data (ID=0xbn n=0.7)

SOLOII/SOLO-D has the ability to return a high resolution P,T,S profile spanning a subsection of the primary binned profile (upper 2000dbar). Data is packed and decoded similarly to the binned profile (ID=0x10, 0x20, 0x30). The High Resolution profile can return every scan of the CTD (1 Hz) or every other scan (1/2 Hz). The data is limited to 1024 values. [Note: When the High Resolution data is requested, the averaging of the primary binned profile must be done by the float (not within the CTD). Typical SOLOII/SOLO-D averaging uses every other CTD scan. However if the High Resolution profile includes every scan, the bin averages will also use every scan. Thus the averaging of the primary binned profile may differ between the subsection with High Resolution data and all other spans.]

Fall Rate data (ID=0x40)

As it falls from the surface to its drift depth, SOLO-D periodically interrogates the SeaBird for a depth reading. This time series is sent back in this data message.

The data message looks like:

```
IDjj<start_time><time(1),depth(1)> . . . <time(m),depth(m)>;  
  ID = one-byte sensor ID code = 0x40.  
  jj = Number of bytes in the message. The count includes ID, jj, the data, and the  
        trailing ;. The count is in 2 binary bytes with MSB first and LSB second.  
  start_time = SOLO time at start of fall (seconds since 1Jan2000) in 4 bytes (MSB first).  
  time(i) = time since start_time in 2 bytes, i=1, ..., m  
            v0.3Apr14: transmitted time resolution =10 s/count  
  depth(i) = depth (LSB=Pgain db) at time(i) in 2 bytes, i=1, ..., m.  
            dBar = Pgain * depth(i) -Poff  
            depth(i) = 0xffff if the pressure reading is invalid
```

Each depth observation takes 4 bytes. The SOLO-D logs values every <SkSLsc> s during the continuous-profile. Others are recorded at: the same time as the first spot sample; at the end of sink(); at the beginning of each seek(); and at the beginning of park().

There may be more than 1 0x40 message with different data. Sort via start_time+time(i).

Rise Rate data (ID=0x50)

The rise rate message is identical in structure to the fall rate message.

For the SOLO-D the rise rate time series begins at the end of park() (start of ascend()), sampling periodically (<AsSLsc>), until the surface is reached. Each temporal count equals 10s.

There may be more than 1 0x50 message with different data. Sort via start_time+time(i).

Pump data (ID=0x60)

v0.3 Apr14, added time variable since start-of-fall.

V0.3 Apr14: Multiple pump messages are allowed, they will all be labeled 0x60. Use t_mn time series to correctly sort.

The data message looks like:

```
IDjj< depth(1),time(1),voltage(1),current(1),vac0(1),vac1(1)> . . .
< depth(m),time(m),voltage(m),current(m),vac0(m),vac1(m);
  ID = one-byte sensor ID code = 0x60.
  jj = Number of bytes in the message. The count includes ID, jj, the data, and the
        trailing ;. The count is in 2 binary bytes with MSB first and LSB second.
  depth(i) = depth (LSB=Pgain db) at time(i) in 2 bytes, i=1, ..., m.
             dBAr = Pgain * depth(i) - Poff. This is the depth when the pump STARTED.
             depth(i) = 0xffff if the pressure reading is invalid
  time(i) = seconds the pump ran in 2 bytes (signed)
  voltage(i) = average pump battery counts while pumping in 2 bytes (0.01V)
  current(i) = average pump current counts while pumping in 2 bytes, LSB=1ma
  vac0(i) = vacuum counts after pump starts in 1 byte
  vac1(i) = vacuum counts before pump stops in 1 byte
  t_mn(i) = (2 bytes) time[minutes] since start-of-fall that this pump event STOPPED.
             Thus, the time of the above depth measurement can be computed as
             t_mn - [pmp_duration_time] (reported in minutes)/60.
```

Engineering data (ID=0xe0, 0xe2, 0xe3, 0xe5)

The engineering data is used to diagnose SOLOII/SOLO-D anomalies. A different format is used in each of the 3 distinct phases of a SOLOII/SOLO-D mission. The LS nibble of the ID indicates the phase of the mission.

Byte	Contents
0	ID/Mission phase: 0xe0 = 1st diagnostic dive at the start of a mission 0xe2 = end of a normal dive cycle 0xe3 = following mission abort 0xe5 = BITtest
1-2	Number of bytes in the message, depends on mission phase as described below
3 -> ??	Depends on mission phase as described below

ID=0xe0, Engineering message in 1st diagnostic dive at start of mission

Byte	Contents
0	ID/Mission phase = 0xe0
1-2	Number of bytes = 76= 0x4c
3	Engineering message version =3
4	#packets in current session
5-10	0 (dummy filler)
11-12	EP -> sattime
13-14	DP->Vcpu = CPU battery voltage counts 0.01V
15-16	DP->Vpmp = Pump battery counts at surface(0.01V)
17-18	DP->Vple = Pump battery counts at end of last pump(0.01V)
19-20	BTvac = pcase vacuum at beginning of BIT in 0.01 inHg
21-22	DP->Air[1] = vac before filling bladder at surface 0.01 inHg
23-24	DP->Air[2] = vac after filling bladder at surface 0.01 inHg

25-26 DP->ISRID = i.d. of last interrupt
 27-28 DP->HPavgI = average pump current at bottom, LSB=1ma
 29-30 DP->HPmaxI = maximum pump current at bottom, LSB=1ma
 31-32 Total seconds pumped to surface
 33-34 Seconds pumped at Surface
 35-36 DP->P[5] = surf press counts @ end of ASCEND (LSB=Pgain dBar)
 37-38 SPRX = Surf press before resetoffset (pertains to prev dive)
 39-40 SPRXL = press after resetoffset (pertains to prev dive)
 41-42 diagP[0] = Press when "in water" sensed
 43-44 diagT[0] = Temp when "in water" sensed
 45-46 diagS[0] = Salinity when "in water" sensed
 47-48 SBnscan = # scans recorded by SBE
 // -1 (0xffff) indicates unable to get scan count from SBE
 // -2 (0xfffe) indicates SBE never started so SBE didn't reset
 // scan count before returning an old value
 49-50 Compacted SBntry,SBstrt,SBstop status (see misspec.h):
 ((DP->SBntry&0xf)<<4) | ((DP->SBstrt&0x3)<<2) | (DP->SBstop&0x3))
 51-52 diagP[1] = Shallowest press in profile
 53-54 diagT[1] = Shallowest Temp in profile
 55-56 diagS[1] = Shallowest Salinity in profile
 57-58 BTvac = BIT vacuum in 0.01 inHg
 59-60 BTPcur = BIT motor current OUT, LSB=1mA
 61-62 BTPsec = BIT Pump seconds
 63 BTPvac[0] = BIT Pump vacuum at beginning of test, before pumping
 64 BTPvac[1] = BIT Pump vacuum after pumping
 65-66 BTVple = BIT pump batt 0.01V
 67-68 BTVcpu = BIT CPU batt 0.01V
 69-70 exception flags
 71 vent data; MSB=#0.1 seconds vent motor ran
 72 LSB LLD status before/after vent ran
 73-74 AbrtCd = code for what caused abort_miss
 75 ; terminator

ID=0xe2, Engineering message in normal dive cycle

Byte	Contents
0	ID/Mission phase = 0xe2
1-2	Number of bytes = 98 = 0x62
3	Engineering message version =3
4	#packets sent in current surface session
5-6	#tries to connect in previous surface session
7-8	parse_X_reply status in previous surface session
9-10	ATSBD return status in previous surface session
11-12	EP->sattime Seconds taken in previous surface session to send all SBD messages
13-14	DP->Vcpu = CPU battery voltage counts 0.01V
15-16	DP->Vpmp = Pump battery counts at surface(0.01V)
17-18	DP->Vple = Pump battery counts at end of last pump(0.01V)
19-20	DP->Air[0] = pcase vac during sinking @50db with oil all inside pcase ,0.01 inHg
21-22	DP->Air[1] = pcase vac before filling oil bladder at surface 0.01 inHg
23-24	DP->Air[2] = pcase vac after filling bladder at surface 0.01 inHg
25-26	DP->ISRID = i.d. of last interrupt
27-28	DP->HPavgI = average pump current at bottom, LSB=1ma
29-30	DP->HPmaxI = maximum pump current at bottom, LSB=1ma For SOLO-D, HPmaxI=0 as dummy-fill
31-32	Total seconds pumped to surface
33-34	Seconds pumped at Surface
35-36	SPRX = Surf press before resetoffset (pertains to prev dive)
37-38	SPRXL = press after resetoffset (pertains to prev dive)

39-40 diagP[0] = Pressure before pumping for ascent
 41-42 diagT[0] = Temp before pumping for ascent
 43-44 diagS[0] = Salinity before pumping for ascent
 45-46 diagP[1] = Last (shallowest) Pressure scan on ascent
 47-48 diagT[1] = Last (shallowest) Temperature scan on ascent
 49-50 diagS[1] = Last (shallowest) Salinity scan on ascent
 51-52 SBnbad = # bad bins from SBE
 53-54 SBnscan = # scans recorded by SBE
 // -1 (0xffff) indicates unable to get scan count from SBE
 // -2 (0xffffe) indicates SBE never started so SBE didn't reset
 // scan count before returning an old value
 55-56 Compacted SBntry,SBstrt,SBstop status (see misspec.h):
 ((DP->SBntry&0x)<<4) | ((DP->SBstrt&0x3)<<2) | (DP->SBstop&0x3))
 57-58 DP->P[0] = press counts before begin of FALL (LSB=Pgain dBar)
 59-60 DP->P[1] = press counts at end of FALL (LSB=Pgain dBar)
 61-62 DP->P[2] = press counts at beginning of DRIFT (LSB=Pgain dBar)
 63-64 DP->P[3] = press counts at end of DRIFT (LSB=Pgain dBar)
 65-66 DP->P[5] = surf press counts @ end of ASCEND (LSB=Pgain dBar)
 67-68 DP->PAVG[0]=average pressure over first half of DRIFT
 69-70 DP->TAVG[0]=average temperature over first half of DRIFT
 71-72 DP->SAVG[0]=average salinity over first half of DRIFT
 73-74 DP->PAVG[1]=average pressure over second half of DRIFT
 75-76 DP->TAVG[1]=average temperature over second half of DRIFT
 77-78 DP->SAVG[1]=average salinity over second half of DRIFT
 79-80 DP->fall_time = seconds from open air valve to end of settle
 81-82 DP->fall rate = avg mm/sec while sinking
 83-84 DP-> SeekT = seconds pumped in 1st settle to drift
 85-86 DP-> SeekP = change of depth (signed 0.1 dbar in 1st settle)
 87-88 exception flags (See table)
 89 vent data; # 0.1 seconds vent motor ran
 90 vent data; LLD status before and after vent ran
 91-92 SBE P offset(*800)
 93-94 PP->SeekSc; tenths of seconds pumped to target depth
 95-96 Number of Packets sent in previous cycle
 97 ; terminator

ID=0xe3, Engineering message following mission abort

Byte	Contents
0	ID/Mission phase = 0xe3
1-2	Number of bytes = 30 = 0x1e
3	Engineering message version =3
4	#packets sent in current surface session
5-6	#tries to connect in last surface session
7-8	parse_X_reply status in last surface session
9-10	ATSBD return status in last surface session
11-12	Seconds taken in sending last SBD message
13-14	current CPU battery voltage counts 0.01V
15-16	current pump battery counts 0.01V
17-18	DP->Air[1] = pcase vacuum at beginning of abort 0.01inHg
19-20	DP->Air[0] = pcase vacuum at end of last xmit (previous cycle) 0.01 inHg
23-24	DP->ISRID = i.d. of last interrupt
25-26	AbtCd = code for what caused abort_miss 0 = no error 1 = current time is later than RTCabort 2 = unable to WakeOST 3 = unable to send Dive number to SOLO II (LOdiveNo) 4 = Iridium ground station commanded to go to abort 5 = FnIDiv was completed. Mission is done 6 = Diagnostic dive failed to get GPS fix, pressure never>dBarGo, or unable to send message to Iridium 7 = pressure sensor failure
27-28	Empty
29	; terminator

ID=0xe5, Engineering message following BITest

Byte	Contents
0	ID/Mission phase = 0xe5
1-2	Number of bytes = 58 = 0x3a
3	Engineering message version =3
4	#packets sent in this surface session
5-6	SBE P Offset(*800)
7-8	CPU battery voltage 0.01 V
9-10	no load pump battery voltage 0. 01 V
11-12	pump battery voltage counts at end of last pump (0.01V)
13-14	DP->HPavgl = average pump current at bottom, LSB=1ma
15-16	seconds pumped out during test
17	Oil vacuum before filling bladder 0.01inHG
18	Oil vacuum after filling bladder 0.01 inHG
19-20	DP-> Air[0] = Pcase Vacuum at beginning of BIT. (Oil Bladder Empty) 0.01 inHg
21-22	DP -> Air[1] = Pcase Vacuum at end of BIT with air bladder inflated. 0.01 inHg
23	Number of tries needed to open valve
24	Number of tries to close valve
25-26	i.d. of last interrupt
27-56	string returned from SBE pt command
57	; terminator

Mission EEPROM dump (ID=0xdn)

Byte	Contents
0	ID/Mission phase = 0xd0,0xd1,0xd2,0xd3 [Possible values 0:d]
1-2	len=Number of bytes (variable, typically 888 for SOLO II)
3- (len-2)	ASCII listing of mission parameters Each EEPROM parameter has a 6 character name and 5 char value: NAMExx=vvvvv The = & signs are present in the listing of each parameter. (15 bytes/parameter) Successive parameters follow without gaps.
len-1	; terminator at the end of the dump

An example showing only the initial 3 and final 2 elements follows:

PchSec= -1|MaxMin= -1|dBaRGo= -1|... UBZmax= -1| UBn= -1|;

The EEPROM dump message is sent only in response to a command "P" from the ground station. It is sent over 3 SBD messages.

Argo Data ID=0xf0 Relayed in normal cycles

Byte	Contents
0	ID/Mission phase = 0xf0
1-2	Number of bytes = 37 = 0x25
3	Data Version (Minor version in high order nibble, major version in low order)=0.3
4-5	Target profile depth
6-7	Target parking depth
8-9	Maximum rise time in minutes
10-11	Target (maximum) fall to parking depth time in minutes
12-13	Maximum fall-from-parking-to-profile-depth time in second
14-15	Target drift time in minutes. V0.3, Apr14: 1 count =5 minutes.
16	Float Version: 1=Deep SOLO, 0=SOLO
17	Target ascent rate while profiling
18-19	Number of seeks
20-21	Surface Time
22-23	Seek Interval in minutes Following added for Deep SOLO (Float Version =1) [sci_units] = [cnts]/SBE_GAIN - SBE_OFF
24-25	SBE_Pgain
26-27	SBE_Poff
28-29	SBE_Tgain
30-31	SBE_Toff
32-33	SBE_Sgain
34-35	SBE_Soff
36	; terminator

Test Data (ID=0xf1)

Byte	Contents
0	ID/Mission phase = 0xf1
1-2	Number of bytes = variable
3	modulo
4-n	test data

Exception Flag (Engineering Message) Table [Value sent by float can be sum from multiple errors]

Hex	Value	Description	Mission
0x0001	1	Valve failed to open	
0x0002	2	Valve failed to close	
0x0004	4	Questionable pressure	
0x0008	8	Antenna was toggled	Surface
0x0010	16	Antenna switch failure (no satellites even after toggling)	Surface
0x0020	32	GPS communication error: <u>No GPS</u>	Surface
0x0040	64		
0x0080	128		
0x0100	256	Float took too long to leave the surface. (toggled valve)	Surface
0x0200	512		
0x0400	1024	Excessive Fall Speed: <u>Abort Mission (return to surface immediately)</u>	Fall,Drift
0x0800	2048		
0x1000	4096	Valve failure during Sink phase of mission	
0x2000	8192	Valve failure during Ascend phase of mission	
0x4000	16384		
0x8000	32768		