Format of SOLOII X messages: Argo Manual/Decoder Version 0.1
Last Updated 03 September 2014
[For ROM SBE602 01Feb10]

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 **$** to the end **>.** The count is in 2 binary bytes with MSB first and LSB second.
- mm = serial number of SOLOII. The SN is in 2 binary bytes with MSB first and LSB second.
- dd = the dive number in 2 binary bytes with MSB first and LSB second.
  : Dive number begins at 1 for the test cycle (Equated to Cycle 0 for Argo)
- p = one-byte packet ID index, range 0 to 255. Used to identify multiple X messages within a dive cycle.The data for each dive cycle starts with p=0.
- <data> = binary data characters. The length of <data> = nn -5. The contents of the <data> section is described below.
- **$** = a dollar sign delimitor 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 > delimitor 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.

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 **$** delimitor).

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.
;	=	delimitor 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 acsent
GPS	03	fix following mission abort
Pressure	1x	depths of CTD readings (scaled 1st difference)
Temperature	2x	depth series of temperature (scaled 1st difference)
Salinity	3x	depth series of salinity (scaled 1st difference)
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
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
Test pattern	f0	*ID reserved, format not yet defined*

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

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 (Beacon phase)
1-2	Number of bytes in the message, 23 = 0x17 with the format as described here
3	0 if fix is invalid, +1 if longitude is East, -1 if longitude is West
4	Signed latitude degrees, >0 = North, <0 = South , range +-90 degrees
5	Integer latitude minutes (unsigned), range 0 to 59
6	Fractional latitude minutes (unsigned) in .01 degrees, range 0 to 99
7	Unsigned longitude degrees, range 0 to 179 degrees
8	Integer longitude minutes (unsigned), range 0 to 59
9	fractional longitude minutes (unsigned) in .01 degrees, range 0 to 99
10	# of messages received from SOLO II (Special for SOLOII replaces Spray Wing Index)
11-12	GPS week (traditional GPS week =0 to 1023 in LS 10 bits: rollover fix in MS 6 bits)
13	GPS day of week, 0=Sunday, 6=Saturday
14	UTC hour
15	UTC minutes
16	Time to get fix = (seconds to get fix)/10 , range 0 to 255 = 0 to 2550 seconds
17	Number of satellites used in fix
18	Minimum signal level
19	Average signal level
20	Maximum signal level
21	10*Horiz. dilution of precision
22	**;** terminator

Pressure data (ID=0x10)
Temperature data (ID=0x20)
Salinity data (ID=0x30)

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 SOLOII ascends and stores the values internally. When SOLOII reaches the surface, it takes the data from the CTD and block averages it in depth into **PRO_BINS** (= 1000) bins.

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 *.04 -10.
degC = temperature counts *.001 -5.
psu = salinity counts *.001 -1.000

Fall Rate data (ID=0x40)

As it falls from the surface to its drift depth, SOLOII 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)	=	seconds since start_time in 2 bytes, i=1, ..., m
depth(i)	=	depth (LSB=0.04 db) at time(i) in 2 bytes, i=1, ..., m. dBar = .04 * depth(i) -10 depth(i) = 0xffff if the pressure reading is invalid

Each depth observation takes 4 bytes. The first time is taken when the valve is opened to leave the surface. The next two times are when the float passes 50m and 100m. After 100 m, pressures are logged every 30 minutes. Typically we allow for 500 (**FallIn**) minutes for the SOLOII to fall 1000 meters so there will be about 16 more measurements. The last record is at the 500 minute mark.

Rise Rate data (ID=0x50)

The rise rate message is identical in structure to the fall rate message. The rise rate time series begins when the SOLO II opens its valve to descent from the drift depth to the profile depth. It logs a pressure/time record 10 times during its descent to the profile depth (interval = **PwaitN**/10). At the bottom of dive, whether determined by timing out (exceeding **PwaitN**) or by reaching the target depth (**ZproN**), another pressure/time record is logged. At this point, the float pumps for **PmpBtm** seconds. A pressure/time record is logged every 30 minutes while the float is ascending.

Pump data (ID=0x60)

. 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=0.04 db) at time(i) in 2 bytes, i=1, ..., m. dBar = .04 * depth(i) -10 depth(i) = 0xffff if the pressure reading is invalid
time(i)	=	seconds the pump ran in 2 bytes
voltagei)	=	average pump battery counts while pumping in 2 bytes (0.01V)
current(i)	=	average pump current at bottom 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

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

The engineering data is used to diagnose SOLOII anomalies. A different format is used in each of the 3 distinct phases of a SOLOII 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
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
3-4	0 (dummy filler for #packets in last session)
5-6	0 (dummy filler for #tries to connect in last session)
7-8	0 (dummy filler for parse_X_reply status in last session)
9-10	0 (dummy filler for ATSBD return status in last session)
11-12	0 (dummy filler for 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 = vac during BIT 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 at end of Ascend
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	BTVple = BIT pump batt 0.01V
62-64	BTVcpu = BIT CPU batt 0.01V
65-66	exception flags (not set)
67	vent data; MSB =#0.1 secs vent motor ran
68	,LSB LLD status before/after vent ran
69-74	empty
75	; terminator

ID=0xe2, Engineering message in normal dive cycle

Byte	Contents
0	ID/Mission phase = 0xe2
1-2	Number of bytes = 84 = 0x54
3-4	#packets sent in previous 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 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
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	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
47-48	Compacted SBntry,SBstrt,SBstop status (see misspec.h): ((DP->SBntry&0xf)<<4) | ((DP->SBstrt&0x3)<<2) | (DP->SBstop&0x3))
49-50	DP->P[0] = press counts before begin of FALL (LSB=.04dBar)
51-52	DP->P[1] = press counts at end of FALL (LSB=.04dBar) NOTE: P[1] will not be valid if the float fall exceeds profile depth
53-54	DP->P[2] = press counts at beginning of DRIFT (LSB=.04dBar)
55-56	DP->P[3] = press counts at end of DRIFT (LSB=.04dBar)
57-58	DP->P[5] = surf press counts @ end of ASCEND (LSB=.04dBar)
59-60	DP->PAVG[0]=average pressure over first half of DRIFT
61-62	DP->TAVG[0]=average temperature over first half of DRIFT
63-64	DP->SAVG[0]=average salinity over first half of DRIFT
65-66	DP->PAVG[1]=average pressure over second half of DRIFT
67-68	DP->TAVG[1]=average temperature over second half of DRIFT
69-70	DP->SAVG[1]=average salinity over second half of DRIFT
71-72	DP->fall_time = seconds from open air valve to end of sink
73-74	DP->fall_rate = avg mm/sec while sinking
75-76	DP->SeekT = seconds of pumping in 1st seek of drift
77-78	DP->SeekP = change of depth (signed 0.1 dbar) in 1st seek
79-80	exception flags
81	vent data; # 0.1 seconds vent motor ran,
82	vent data; LLD status before and after vent ran
83	`;  terminator`

ID=0xe3, Engineering message following mission abort

Byte	Contents
0	ID/Mission phase = 0xe3
1-2	Number of bytes = 30 = 0x1e
3-4	#packets sent in last 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	AbrtCd = 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 = FnlDiv 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

Mission EEPROM dump (ID=0xd0)

Byte	Contents
0	ID/Mission phase = 0xd0,0xd1,0xd2,0xd3
1-2	len=Number of bytes (variable)
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|... **DATtry**= -1|**DATsec**= -1|;

The EEPROM dump message is sent only in response to a command "**P**" from the ground station. It is sent over 4 SBD messages.

Test pattern (ID=0xf0)

ID reserved, format not yet defined