Tritech International Limited

Software Notes

for controlling and operating

RS-232 Sonar* Heads

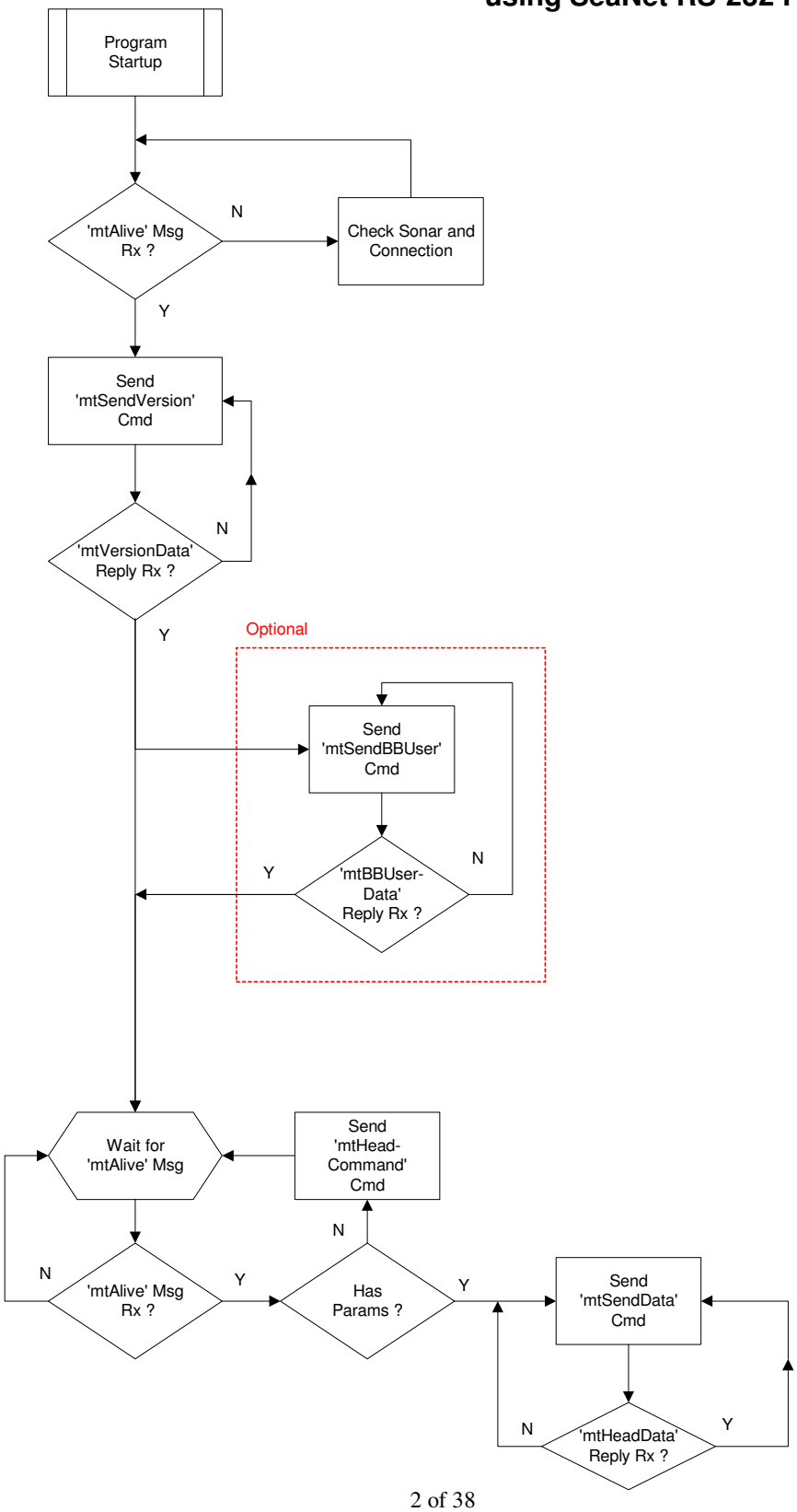
*** SeaKing DST / SeaPrince DST / Micron DST**

IMPORTANT:

Applies to **i)** SeaKing Sonar Head 'SEANET' software **ii)** 'MICRON' Sonar software.

Document Rev.**1**, 13th August 2003 km
Document Rev.**2**, 18th October 2004 km (added notes on Full/Half Duplex operation)
Document Rev.**3**, 2nd May 2006 km (Updated for Digital 'DST' Sonar range)
Document Rev.**4**, 22nd December 2006 km (added 'User Code' to 'FPGAVersionData' reply)

Basic Sonar Operation
using SeaNet RS-232 Protocol



RS-232 Command and Reply Protocol (SeaNet Program)

The RS-232 protocol used in SeaNet is a command and reply message format that closely follows the "packeted" format of the ARCNET LAN protocol.

Programmers who already have written control software based around the 'SONV3' RS-232 protocol will find that although the methodology of control is very similar, SeaNet uses a new library of command and message types.
There were 2 versions of protocol in the 'SONV3' program, V1.4 and V1.5. Many users will have written their programs under V1.4 which included commands such as 'StopCmd', 'ParamsCmd' & 'GetDataCmd'.

SONV3 V1.5 programmers will find that SeaNet uses a protocol that is much closer matched to this than V1.4. Programmers familiar with the 'SONV3' V1.4 RS-232 protocol will however find a number of differences that will now be explained;

First of all, SeaNet commands and replies all have a "@" character header (as opposed to "%" header used in V1.4). Also, in the Message Header there are ID fields identifying the senders Node number ('SID') and the intended recipients / destination Node number ('DID'). This is very much a reflection of the LAN operation used by the ARCNET protocol. This ID tagging has been introduced in order to operate the RS-232 to ARCNET conversion options, such as junction boxes, that were not possible under the older SONV3 V1.4 protocol.

Hint: All commands and replies are now terminated with a Line Feed character (0Ah) although it is NOT advisable that this is solely used for reading in message strings as 0Ah values may appear in the Sonar binary sample data of 'mtHeadData' replies. Instead, the message strings should be (additionally) read in using combination of Start Character and Hex/Bin length header values. The Line Feed termination character is NOT included in the Hex/Bin Length count.

Finally, on application of power, each device programmed with SeaNet software will now immediately broadcast "mtAlive" commands at a 1Hz rate to inform other connected devices that it has just been connected. This is a different approach to SONV3 V1.4 operation where the "Surface Controller" would have to establish the connection and then assume control. Now the Surface Controller can just listen into, for instance, the Serial port and then assume control when the subsea device (i.e. Sonar) has came online and started sending it's "mtAlive" commands.

Note: This broadcasting feature of SeaNet (& SONV3 V1.5) programmed devices can also be used to determine if the device that has been connected is programmed up to a SeaNet ready version of software. If the device does not broadcast 'mtAlive' commands then it may be programmed back on SONV3 "V1.4" in which case Tritech Software support should be contacted for advice.

Multi-packet mode *(not used as standard with RS-232 transmission protocol)*:

As aforementioned, the SeaNet RS-232 protocol is based on the "packeted" ArcNet LAN protocol that is mainly used by these devices. The maximum packet size in the LAN protocol is 254 bytes although most commands and replies will have message lengths below 254 bytes in size.
There is one exception to the max. packet size rule and this is with a 'mtBBUserData' reply message - the length of this is a fixed 263bytes and is the only message larger than 254bytes that will be transmitted in a single data packet.
Only "mtHeadData" replies will have message lengths that exceed 254 bytes. In devices that use the Multi-packet mode, these longer messages would be broken down into smaller individual packets for ArcNet LAN transmission ("mtHeadData" messages are broken up into data packets all sized within a 128-byte packet size limit).

However, for the direct RS-232 protocol the messages will normally be sent in single packets that can exceed the 254 byte maximum rule of the ArcNet LAN protocol. **Important:** Users should be aware that the RS-232 protocol has the facility of using the (<254byte) packeted approach to facilitate interfacing to certain device servers and multiplexers. It is therefore advisable that the user program be able to accept and process multi-packets in case it is ever necessary to implement such a mode in their systems.

It is possible to communicate with ArcNet configured Sonar heads via the SeaHub or SKIM-100 surface interface modules. The serial protocol applied in the Surface Controller will directly follow what is stated in this document apart from the fact that when the Sonar is configured as ArcNet then the "mtHeadData" replies will **always** be split using the packeted mode.
The SeaHub and SKIM-100 surface interface modules will convert the ArcNet Sonar data packets and reply messages into RS-232 for transmission to the serial RS-232 interface port on the Surface Controller. Likewise, the surface interface module will convert RS-232 packets into ArcNet for transmission of commands sent down to the Sonar head.
There is a separate Addendum document that gives all the additional information for interfacing via the SeaHub or SKIM-100 surface interface modules.

As aforementioned, 'mtHeadData' replies may exceed 254 bytes in length and in the RS-232 standard protocol this reply message would be sent in one packet. If the RS-232 protocol was updated to follow the multi-packet approach of the parent LAN protocol, then the message would be broken up and transmitted as separate <128 byte packets. Each of these packets is then transmitted in sequence by the sender (i.e. Sonar) and must be "stitched" together by the recipient (i.e. Serial Port / AIF) to re-create the original message. Each data packet contains a sequence number in its Message Header and an 'End Of Sequence' identifier is used to mark the last packet in the sequence.

The summary of SeaNet operation will be as follows *(refer to earlier flowchart)*;

The "Surface Controller" waits for an 'mtAlive' message to inform it whether the subsea device is in a power up state, whether Version or Configuration parameters have been sent, whether the device has a valid set of parameters and what its motor may be doing (it might be busy and unready for doing pings and sending data). For a simulated head, it can always be ready.
The Surface Controller then checks the 'mtAlive' message and may ask for Version and Configuration (UserBB) data if these haven't already been received. Then, if the subsea device has no parameters it sends them down before starting the 'mtSendData' sequence. The 'mtSendData' (same as SONV3 'GetDataCmd') is used to trigger the subsea device to perform one sample, i.e. 1 ping (or Scanline) for a Sonar, and retrieve this data back for surface processing.

Rev.2 Note

Important: For SeaKing/Micron/MiniKing RS-232 Sonar heads set to operate Full Duplex (mtBBUserData 'Half Duplex' bit = 0), for every 1 x 'mtSendData' command sent the Sonar head will reply with 2 x 'mtHeadData' reply messages. For RS-485 Micron/MiniKing heads, the 'Half Duplex' bit will always be set to =1 and only 1 x 'mtHeadData' reply will be sent in this case.

SeaNet General Packet Format is;

'@'	:	Message Header.
HHHH	:	Hex Length of whole binary packet (excluding LF Terminator).
BB	:	Binary Word of above Hex Length.
SID	:	Packet Source Identification (Tx Node number 0 - 255).
DID	:	Packet Destination Identification (Rx Node number 0 -255).
COUNT **	:	Byte Count of attached message that follows this byte.
MSG	:	Command / Reply Message (i.e. 'Alive' command, 'Data Reply' message).
TERM	:	Message Terminator = Line Feed (0Ah).

*** Set to 0 (zero) in 'mtHeadData' reply to indicate Multi-packet mode NOT used by device.*

List of Command / Reply messages ('MSG')

There are quite a number of Command and Reply Messages that can be exchanged between nodes on the SeaKing/SeaNet network, although only several are required for a simple Sonar operation for example. The SeaKing/SeaNet device will be programmed with a Node number, for instance an Imaging Sonar will be programmed with Node 2. **For users creating their own programs to operate**

the RS-232 Sonar through a Serial port, their programs must take on the Node number of 255. These programs should "listen into" the Serial Port that the Sonar is connected to and wait for the Sonar to come online and transmit its "Alive" ('mtAlive') commands. Once these have been received through the serial port, the User program can then start issuing Command messages and assume full control of Sonar operation. The complete list of Command and Reply messages are listed immediately below…

(The first Byte of the Message contains the Message I.D. number which is listed)

**	mtNull	=	0	*	mtAuxData	=	8	**	**mtHeadCommand**	=	19
**	**mtVersionData**	=	1	*	mtAdcData	=	9	*	mtEraseSector	=	20
**	**mtHeadData**	=	2	*	mtAdcReq	=	10	*	mtProgBlock	=	21
*	mtSpectData	=	3	*	mtLanStatus	=	13	*	mtCopyBootBlk	=	22
**	mtAlive	=	4	*	mtSetTime	=	14	**	**mtSendVersion**	=	23
*	mtPrgAck	=	5	*	mtTimeout	=	15	**	**mtSendBBuser**	=	24
**	**mtBBUserData**	=	6	*	**mtReBoot**	=	16	**	**mtSendData**	=	25
*	mtTestData	=	7	*	mtPerformanceData	=	17	*	mtSendPerf'nceData	=	26

*** = Needed for simple operation, * = More advanced operation (may not be described here).*

More recent Commands and Replies that have been added since Rev.2 are…

#	mtDopplerData	=	27	#	mtSendBathyProfile	=	46	#	mtSWParams	=	65
#	mtDopplerParams	=	28	*	mtSendFpgaFlashSt	=	47	*	**mtStopAlives**	=	66
#	mtAttitudeData	=	29	*	mtFpgaTestData	=	48	#	mtResponderPing	=	67
#	mtAttitudeParams	=	30	*	mtFpgaFlashStData	=	49	#	mtVideoCntrlCmd	=	68
#	mtBathyParams	=	31	#	mtSendTrnspdrStat	=	50	#	mtVideoCntrlData	=	69
#	mtBathyData	=	32	#	mtSendTrnspdrData	=	51	*	mtResetToDefaults	=	70
#	mtWspData	=	33	#	mtAMNavData	=	52	*	mtChangeVerData	=	71
#	mtWspParams	=	34	#	mtAMNavParams	=	53	*	mtFpgaProgUsrCde	=	72
#	mtStreamData	=	35	#	mtTrnspdrTxCntrl	=	54				
#	mtGenericData	=	36	#	mtTrnspdrConfig	=	55				
#	mtGUID_Data	=	37	*	mtSendFpgaVersion	=	56				
#	mtIPAQCntrlParam	=	38	*	**mtFpgaVersionData**	=	57				
#	mtIPAQCntrlData	=	39	#	mtScanHeader	=	58				
*	mtFpgaTest	=	40	#	mtScanData	=	59				
*	mtFpgaErase	=	41	#	mtGlobal	=	60				
*	mtFpgaProgram	=	42	*	mtFpgaDoCalibrate	=	61				
#	mtBathyInfo	=	43	*	mtSendFpgaCalData	=	62				
#	mtBathyProfile	=	44	*	**mtFpgaCalData**	=	63				
#	mtSendBathyPrfReq	=	45	*	mZeroFpgaCal	=	64				

*** = Needed for simple operation, * = More advanced operation (may not be described here)*
 # = Not Applicable.

1) 'mtSendVersion' Command Message (ID = '23')

By sending an 'mtSendVersion' command to the connected SeaKing/SeaNet device, the Sender can check what software version is programmed into the device. This can be beneficial for whenever any software updates are provided by Tritech when a software version check is necessary at any time. Software updates can be programmed into the subsea device through the 'SeaNet Setup' utility program. The 'Setup' program will detect the subsea device through the serial port or AIF connection and then, on command, download the software update into the subsea device's non-volatile memory.

The 'mtSendVersion' command would usually be sent before attempting scanning operation of the subsea device. On reception of an 'mtSendVersion' command, the subsea device will send its 'mtVersionData' reply.

N.B. The SeaNet 'mtSendVersion' command performs the same function as the SONV3 'VersionCmd'.

The sequence of Commands and Replies used to perform a Version check on a Sonar is as follows…

A. Serial Port sends **mtSendVersion** Command (14 byte)

40	**30**	**30**	**30**	**38**	**08**	**00**	**FF**	**02**	**03**	**17**	**80**	**02**	**0A**
Hdr '@'	Hex Length = 8 bytes				Bin Length = 8 bytes		Tx Nde 255	Rx Nde 02	No. Byte = 3	**mtSend Ver'**	Seq = End	Nde 02	LF

B. Sonar replies with **mtVersionData** Reply Message (25 byte) *- Full description below*

40	**30**	**30**	**31**	**33**	**13**	**00**	**02**	**FF**	**0E**	**01**	**80**	**02**	**31**
Hdr '@'	Hex Length = 19 bytes				Bin Length = 19 bytes		Tx Nde 02	Rx Nde 255	No. Byte 14	**mtVer'n D'ta**	Seq = End	Nde 02	CPU ID

11	**0D**	**8C**	**83**	**A8**	**00**	**00**	**3C**	**88**	**02**	**0A**
CPU ID = 8C0D1131H			Program Length = 43139				Checksum = 34876		Nde 02	LF

During initial writing of their program, the User should communicate with the subsea device and first receive an 'mtVersionData' reply back from the subsea device. Within this reply, there can be found 'Program Length' and 'Checksum' information and these should be retained in the User program and used for version comparison during run-time operation. The subsea device will be supplied with information of what software version is pre-installed inside it. It is the responsibility of the User to keep this information stored within their program along with the 'Program Length' and 'Checksum' values. If and when a software update is later supplied by Tritech, the User program should be updated with the new version information. A quick run-time check of the subsea device operation should then be performed to test software compatibility. Note: It is not common to change the structure of any of the command and reply protocols although new protocols may be added at any time to increase device functionality.

'mtVersionData' Reply (ID = '1') Packet format is;

Byte 1	:	Message Header = "@".
Bytes 2-5	:	Hex Length of binary packet from byte 6 onwards *(excludes LF)*.
Bytes 6,7	:	Binary Word of above Hex Length.
Byte 8	:	Packet Source Identification (Tx Node number 0 - 240).
Byte 9	:	Packet Destination Identification (Rx Node number = 255).
Byte 10	:	Byte Count of attached message that follows this byte.
Byte 11	:	Message Type = "1" for mtVersionData.
Byte 12	:	Message Sequence Bitset *(see below)*.
Byte 13	:	Tx Node number *(copy of Byte 8)*.
Bytes 14-17	:	CPU ID.
Bytes 18-21	:	Program Length.
Bytes 22,23	:	Checksum Length.
Byte 24	:	Node number ID programmed in Flash *(same as Byte 8)*.
Byte 25	:	Message Terminator = LF *(0Ah)*.

Message Sequence (Byte 12)

This is a Bitset that should be processed in order to determine if the message exceeds maximum packet length boundaries and will therefore be split into sequential packets. An 'mtVersionData' Reply message sent by the Sonar will always be only 1 packet.
For Sonar, Always = 80h

The Message Sequence Bitset is constructed as follows;

Bits 0 - 6 : Packet Sequence bits. 1st packet has Packet Sequence of 0, 2nd packet has Packet Sequence of 1 with bit 0 = 1 *(0000001)*, 3rd packet has Packet Sequence of 2 *(0000010), …etc.*

Bit 7 : End Sequence bit. Set to 1 when it is the last packet in sequence.
For instance, the last packet in a 1 packet sequence will have a BitSet of *10000000 (=80H)*.

CPU ID (Bytes 14-17)

This contains information on Transmitter Frequencies and BoardTypes and the CPU ID number should never really change without there first being a hardware change.

The 4-byte CPU ID number consists of;

Byte 14 : 'Software Version'. This is only applicable to the new range of digital (DST and Chirp) heads that Tritech produce. All other heads will always state a program version of "00". In the Setup program, the Software version will be preceded by the letter "S".
Note: This Byte previously was used to state Channel Frequencies. These frequencies should now be extracted from the 'mtBBUserData' reply message (Bytes 102 -> 109).

Byte 15 : 'InfoBits'. BitSet of;
['BT_0', 'BT_1', 'BT_2', 'BT_3', 'ID_IsV4', 'ID_SP1', 'ID_SP2', 'ID_SP3']
Bits 0 -> 3 ('BT_x') can be ignored. Bits 4 -> 7 ('ID_x') give Board ID.
Note: "V4" boards are used in Micron DST heads.

Bytes 16,17 : 'ID'. Serial number programmed into "CPU" PCB. It is not critical that there be a (non-zero) number programmed in but this ID can be used to record which Sonar is being operated if there are more than one being used in the system. Sonars will be factory programmed with a unique 'ID' number.
Note: This is no longer used in the digital DST heads and can therefore be ignored.

Program Length (Bytes 18-21) / Checksum (Bytes 22,23)

These values should be used to record / verify the Program revision of Sonar software. For instance, a Sonar may have been introduced from another field system and be programmed with earlier (or later) software. This may affect functionality so it is always wise to ensure that User Programs that have been written around a specific SeaNet software revision operate with Sonars programmed with that same revision.
N.B. The 'SeaNet Setup' Utility Program can be used to check and re-program the desired revision of software into the Sonar.

Node ID (24)

This is the Node ID that has been programmed into the subsea device. Because SeaKing devices are more commonly configured to operate using ARCNET LAN telemetry they require nodal network ID numbers.
Default Node ID numbers are;

Imaging Sonar	:	Node 2
Profiling Sonar	:	Node 20,21
Bathymeter	:	Node 40
Sidescan	:	Node 10
Sub-bottom	:	Node 15
SKIM-100 (i/f module)	:	Node 254
SeaHub (i/f module)	:	Node 252

2) 'mtSendBBUser' Command Message (ID = '24')

The main Message Reply to this Command is a device specific 'mtBBUserData' record that defines various settings and options programmed into the subsea device. The settings are not required for Sonar simulator programs and otherwise it is okay to do a quick verification of the settings particularly if special features have been programmed into the head (i.e. Sonar 'Aux' port serial interface, Pressure/Temp sensor interface).
N.B. The SeaNet 'mtSendBBUser' performs the same function as the SONV3 'GetBBUserCmd'.

For the new digital 'DST' range of Sonar heads, these devices have their firmware program stored in an onboard FPGA device. The DST Sonar heads will send 2 additional replies in response to the mtSendBBUser request and these 2 additional replies will immediately follow the mtBBuserData reply. The 2 additional digital Sonar replies are 'mtFpgaCalibrationData' and 'mtFpgaVersionData'.

The sequence of Commands and Replies used to perform an 'mtBBUserData' check on a DST Sonar is as follows…

A. Serial Port sends **mtSendBBUser** Command (14 byte)

40	**30**	**30**	**30**	**38**	**08**	**00**	**FF**	**02**	**03**	**18**	**80**	**02**	**0A**
Hdr '@'	Hex Length = 8 bytes				Bin Length = 8 bytes		Tx Nde 255	Rx Nde 02	No. Byte = 3	**mtSend BB'**	Seq = End	Nde 02	LF

B. DST Sonar replies with **mtBBUserData** Reply Message (264 byte) *- Full description further on*

40	**30**	**31**	**30**	**32**	**02**	**01**	**02**	**FF**	**FD**	**06**	**80**	**02**	**FF**
Hdr '@'	Hex Length = 258 bytes				Bin Length = 258 bytes		Tx Nde 02	Rx Nde 255	No. Byte 253	**mtBBUser..**	Seq = End	Nde 02	Aif Nde 255
00	**34**	**12**	**A1**	**00**	**0F**	**03**	**00**	**00**	**00**	**00**	**05**	**00**	**04**
D1	CK = 4660		UserBBLen = 161		B'rd Typ 15	Soft Ver = 3	Spare3		Spare4		LanBaud Lo = '5' = 78 kBaud		Lan Baud ->
00	**02**	**00**	**02**	**00**	**03**	**00**	**03**	**00**	**07**	**00**	**00**	**00**	**00**
Hi = '4' = 156	LanSens Lo = '2' = Neutral		LanSens Hi = '2' = Neutral		LanETmo = 3		Slu0 Baud Lo = '3' = 9.6 kBaud		Slu0 Baud Hi = '7' = 115.2 kB		Slu0 Parity Lo = '0' = None		Slu0 Pr'ty ->
00	**00**	**00**	**00**	**00**	**03**	**00**	**07**	**00**	**00**	**00**	**00**	**00**	**00**
Hi = '0' N'ne	Slu0 D'taBit Lo = '0' = 8 bits		Slu0 D'taBit Hi = '0' = 8 bits		Slu1 Baud Lo = '3' = 9.6 kBaud		Slu1 Baud Hi = '3' = 115kBaud		Slu1 Parity Lo = '0' = None		Slu1 Parity Hi = '0' = None		Slu1 DBit ->
00	**00**	**00**	**03**	**10**	**EB**	**09**	**00**	**B0**	**71**	**0B**	**00**	**2C**	**01**
Lo = '0' = 8	Slu1 D'taBit Hi = '0' = 8 bits		Mic' DST 6		CH1 Rx Start Freq = 650,000 Hz			CH1 Rx End Freq = 750,000 Hz				Ch1 Filter ->	
00	**00**	**33**	**85**	**1B**	**00**	**18**	**15**	**64**	**00**	**0A**	**00**	**00**	**73**
Length = 300			Ch1 Sample Rate = 1,803,571Hz			Ch1 Alpha Shading = 54%		Ch1 TVG Correction = 100		Ch1 Mac = 10	Ch1 Lim = 0	Ch1 Scal = 0	Ch1 Rssi Offs
32	**64**	**00**	**00**	**00**	**10**	**EB**	**09**	**00**	**B0**	**71**	**0B**	**00**	**2C**
Ch1 Det Thr'	Ch1 Rx Detect Min Pulse = 100		Ch1 Rx Detect Offs = 0		Ch1 Tx Start Freq = 650,000Hz				Ch1 Tx End Freq = 750,000Hz				Ch1 Tx ->
01	**00**	**00**	**09**	**FF**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**
Pulse Length = 300 usec			Ch1 D'ty = 9	Ch1 Tx Volt	CH2 Rx Start Freq = N/A				CH2 Rx End Freq = N/lA				Ch2 Filt ->
00	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**

Length = N/A			Ch2 Sample Rate = N/A				Ch2 Alpha Shading = N/A		Ch2 TVG Correction = N/A		Ch2 Mac N/A	Ch2 Lim N/A	Ch1 Scal = 0
00	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**
Ch2 Rssi Offs	Ch2 Det Thr'	Ch2 Rx Detect Min Pulse = N/A		Ch2 Rx Detect Offs = N/A		Ch2 Tx Start Freq = N/A				Ch2 Tx End Freq = N/A			
00	**00**	**00**	**00**	**00**	**00**	**01**	**01**	**C8**	**00**	**C8**	**00**	**20**	**03**
Ch2 Tx Pulse Length = N/A				Ch2 D'ty Cyc	Ch2 Tx Volt	Half Dpx = Y	Has Mot = Y	Mot Pan Gain	Mot Tilt Gain	Motor Pan Constant = 200		Max Speed Pan = 800	
00	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**
MechLL Pan = 0		MechRL Pan = 0		Motor Tilt Constant = 0		Max Speed Tilt = 0		MechLL Tilt = 0		MechRL Tilt = 0		Main = 232	Aux = 232
00	**E8**	**03**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**
Drv = Nrm	Aux Comms Rate = 1sec		No Dual Filt	Gain Adc N/A	Gain Adc N/A	1st Em-pty	…	…	…	…	…	…	…
00	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**
…	…	…	…	…	…	…	…	…	…	…	…	…	…
00	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**
…	…	…	…	…	…	…	…	…	…	…	…	…	…
00	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**
…	…	…	…	…	…	…	…	…	…	…	…	…	…
00	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**
…	…	…	…	…	…	…	…	…	…	…	…	…	…
00	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**
…	…	…	…	…	…	…	…	…	…	…	…	…	…
00	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**0A**		
…	…	…	…	…	…	…	…	…	…	89th Em-pty	LF		

C. DST Sonar replies with **mtFpgaCalibrationData** Reply Message (80 bytes) *- Details further on*

40	**30**	**30**	**34**	**41**	**4A**	**00**	**02**	**FF**	**45**	**3F**	**80**	**02**	**01**
Hdr '@'	Hex Length = 74 bytes				Bin Length = 74 bytes		Tx Nde 02	Rx Nde 255	No. Byte 69	**mtFpgaCal**	Seq = End	Nde 02	Cal'd = Yes
01	**FB**	**FF**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**
Adc Ch = 1	Adc Chan0 Offset = 65531		Adc Chan1 Offset = 0		Adc Chan2 Offset = 0		Adc Chan3 Offset = 0		Adc Chan4 Offset = 0		Adc Chan5 Offset = 0		Adc Ch ->
00	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**
6 Offs = 0	Adc Chan7 Offset = 0		Adc Chan8 Offset = 0		Adc Chan9 Offset = 0		Adc Ch10 Offset = 0		Adc Ch11 Offset = 0		Adc Ch12 Offset = 0		Adc Ch ->
00	**00**	**00**	**00**	**00**	**04**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**

13 Offs = 0	Adc Ch14 Offset = 0		Adc Ch15 Offset = 0		Adc Chan0 Quality = 4		Adc Chan1 Quality = 0		Adc Chan2 Quality = 0		Adc Chan3 Quality = 0		Adc Ch ->
00	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**
4 Qual = 0	Adc Chan5 Quality = 0		Adc Chan6 Quality = 0		Adc Chan7 Quality = 0		Adc Chan8 Quality = 0		Adc Chan9 Quality = 0		Adc Ch10 Quality = 0		Adc Ch ->
00	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**0A**				
11 Qual = 0	Adc Ch12 Quality = 0		Adc Ch13 Quality = 0		Adc Ch14 Quality = 0		Adc Ch15 Quality = 0		LF				

D. DST Sonar replies with **mtFpgaVersionData** Reply Message (28 bytes) *- Details further on*

40	**30**	**30**	**31**	**36**	**16**	**00**	**02**	**FF**	**11**	**39**	**80**	**02**	**02**
Hdr '@'	Hex Length = 24 bytes				Bin Length = 24 bytes		Tx Nde 02	Rx Nde 255	No. Byte 17	**mtFpga Ver**	Seq = End	Nde 02	ID = 2
93	**50**	**04**	**05**	**00**	**04**	**02**	**3B**	**02**	**06**	**20**	**11**	**23**	**0A**
Flash ID = 5045093h = 501050223				Blocks = 1024		CheckSum = 15106		Rev = 2	User Code = 588324870				LF

3) 'mtReBoot' Command Message (ID = '16')

At any time, the User can send this command to the subsea device to reset it. If the device is not responding or is transmitting unintelligible or unexpected data then it should be rebooted (using this command) and then re-sent an mtHeadCommand with "clean" parameters. The software reboot is particularly useful when a power reset on the Sonar is not always possible.
N.B. The SeaNet 'mtReBoot' has same functionality as the SONV3 'StopCmd' + 'GoCmd' operation.

Note: Whenever a ReBoot is performed the Sonar Parameter block will be cleared and the Motor will power ON as default. The Sonar will proceed to send 'mtAlive' broadcasts at a 1Hz rate. If the Sonar 'mtSendData' triggering does not start within 1 minute after the reboot then after this timeout period the Motor will disarm and power down. The motor will always power down after 1 minute of inactivity as a power saving measure.

An example of a Command used to perform an 'mtReBoot' operation on a Sonar is as follows…

A. Serial Port sends **mtReBoot** Command (14 byte)

40	**30**	**30**	**30**	**38**	**08**	**00**	**FF**	**02**	**03**	**10**	**80**	**02**	**0A**
Hdr '@'	Hex Length = 8 bytes				Bin Length = 8 bytes		Tx Nde 255	Rx Nde 02	No. Byte = 3	**mtReBoot**	Seq = End	Nde 02	LF

4) 'mtHeadCommand' Command Message (ID = '19')

The 'Device Parameter' block contained within this command will give the subsea device (i.e. Sonar) the vital instructions and control settings to inform it how to operate. Without these settings the device cannot go on to perform its routine functions and won't respond to 'mtSendData' routines.

Using a Sonar device for example, the 'Device Parameter' block will include the user controls that are available to the operator such as Gain, Scan Width and Range Scale. Any update to the user controls will enforce the sending of a new 'mtHeadCommand' command to the Sonar.
N.B. The SeaNet 'mtHeadCommand' performs the same function as the SONV3 'ParamsCmd'.

Note: If the device is Dual Channel (i.e. SeaKing Sonar) then a 16-byte "V3B" Gain Parameter block is appended at the end and Byte 14 must be set to 1Dh to indicate this. Else, for Single channel devices such as SeaPrince and MiniKing, do not append the "V3B" block and set Byte 14 to 01h to indicate this. Single channel DST Sonar heads will accept the Parameter Block with and without the extra 16-byte appendage.

The series of Command and Replies issued to perform a Device Parameter update to a Sonar is as follows…

A. Serial Port sends **mtHeadCommand** Command (82 bytes) = "V3B" Parameter block

16-byte 'V3B' Gain Parameter message is appended for Dual Channel operation (Byte 14 = 1Dh)

40	**30**	**30**	**34**	**43**	**4C**	**00**	**FF**	**02**	**47**	**13**	**80**	**02**	**1D**
Hdr '@'	Hex Length = 76 bytes				Bin Length = 76 bytes		Tx Nde 255	Rx Nde 02	No. Byte = 71	**mtH 'dC md**	Seq = End	Nde 02	V3B Para -ms
83	**23**	**02**	**99**	**99**	**99**	**02**	**66**	**66**	**66**	**05**	**A3**	**70**	**3D**
HdCtrl * = 9091		HdT ype = 02	TXN, Ch1 = 43620761 # *Ignored by DST Chirp*				TXN, Ch2 = 90596966 # *Ignored by DST Chirp*				RXN, Ch1 = 104689827 # *Ignored by DST*		
06	**70**	**3D**	**0A**	**09**	**28**	**00**	**3C**	**00**	**01**	**00**	**FF**	**18**	**51**
	RXN, Ch2 = 151666032 # *Ignored by DST Chirp*				TxPulseLen = 40 usec # *Ignored*		RangeScale = 6 metres		LeftLim = 1 (1/16 Grad)		RightLim = 6399 (1/16 Grad)		AD Sp'n = 81
08	**54**	**54**	**5A**	**00**	**7D**	**00**	**19**	**10**	**8D**	**00**	**5A**	**00**	**E8**
AD Low = 8	Iga-in, Ch1	Iga-in, Ch2	Slope, Ch1 = 90 # *Ignored*		Slope, Ch2 = 125 # *Ignored*		Mo' Tme = 25	Step Size = 16	ADInterval = 141		Nbins = 90		Max ADb uf
03	**97**	**03**	**40**	**06**	**01**	**00**	**00**	**00**	**50**	**51**	**09**	**08**	**54**
= 100 0	Lockout = 919 usec		MinorAxis Dir = 1600 (1/16 Grad)		Maj' Axis Pan	Ctl2 = 0	ScanZ = 0		AD Sp'n Ch1	AD Sp'n Ch2	AD Low Ch1	AD Low Ch2	Iga-in Ch1
54	**00**	**00**	**5A**	**00**	**7D**	**00**	**00**	**00**	**00**	**00**	**0A**		
Iga-in, Ch2	Adc SetP ,Ch1	Adc SetP ,Ch2	Slope, Ch1 = 90 # *Ignored*		Slope, Ch2 = 125 # *Ignored*		Slope Delay, Ch1 # *Ignored*		Slope Delay, Ch2 # *Ignored*		LF		

* HdCtrl = 9091 = 0010001110000011 *- Full description further on*
{8-bit ADC, Continuous Scan, ScanLeft, Upright Orientation, Motor On, Transmitter On, Spare Bit, Use Chan 2, Raw ADC, Has Motor, No Heading Offset, No PingPong, No Stare, ReplyASL, No hThrRec, Don't Ignore Centre Sensor}

 # The Digital DST heads with Chirp transmission pulses will not apply any of these Transmitter/Receiver parameters. These values have been moved into non-volatile memory area, within the mtBBUserData area of the program. You can access these values in the 'Micron Setup'/'Seanet Setup' utility program. Click on 'Action' | 'Setup' for the Node 2 DST Sonar, you will then be able to view these parameters. It is not recommended to change any of these parameters as they are factory tuned, however if so required then contact Tritech software support.
The mtHeadCommand TXN and RXN parameters can be filled with zeros or with the centre frequency of the Chirp Sonar transmit pulse as shown in the (dual frequency) example above.

N.B. For Single Channel devices, set Byte 14 to 01h and do not include V3B Parameter block. Although, single channel DST Sonar heads will accept the extra V3B block.
Remember to calculate 'Hex/Bin Length' and 'No. Byte' fields accordingly.

B. Sonar sends **mtAlive** Message notifying of Sent 'Device Parameters' (22 byte)

40	**30**	**30**	**31**	**30**	**10**	**00**	**02**	**FF**	**0B**	**04**	**80**	**02**	**80**
Hdr '@'	Hex Length = 16 bytes				Bin Length = 16 bytes		Tx Nde 02	Rx Nde 255	No. Byte = 11	**mtA live**	Seq = End	Nde 02	Will Send #

C4	37	00	00	80	0C	CA	0A
Head Time = 14276 msec				Motor Pos = 3200 (1/16 Grad)		He-Ad Inf*	LF

** Transducer Centred, Motor On, Has been Sent Device Param's*

C. Sonar sends **mtAlive** Message validating the received 'Device Parameters' (22 byte)

40	30	30	31	30	10	00	02	FF	0B	04	80	02	80
Hdr '@'	Hex Length = 16 bytes				Bin Length = 16 bytes		Tx Nde 02	Rx Nde 255	No. Byte = 11	**mtA live**	Seq = End	Nde 02	Will Send #

AD	3B	00	00	80	0C	8A	0A
Head Time = 15277 msec				Motor Pos = 3200 (1/16 Grad)		He-Ad Inf*	LF

** Transducer Centred, Motor On, Has Parameters*

The 'WillSend' Byte (Byte 14) is no longer used to signify whether the DST head has received a valid parameter set (i.e. 'mtHeadCommand') from the surface controller. This Byte was previously toggled from 80h to 00h whenever parameters had been received and validated by the Sonar. This is no longer the case and Byte 21 ('HeadInf'), Bits 6 and 7 are the only flags that should now be read to indicate the status of Parameter setting in the DST head.

'mtHeadCommand' Command (ID = '19') Packet format is;

Byte 1	:	Message Header = "@".
Bytes 2-5	:	Hex Length of binary packet from byte 6 onwards *(excludes LF)*.
Bytes 6,7	:	Binary Word of above Hex Length.
Byte 8	:	Packet Source Identification (Tx Node number 255).
Byte 9	:	Packet Destination Identification (Rx Node number 0 - 240).
Byte 10	:	Byte Count of attached message that follows this byte.
Byte 11	:	Message Type = "19" for mtHeadCommand.
Byte 12	:	Message Sequence Bitset *(see below)*.
Byte 13	:	Tx Node number *(copy of Byte 8)*.
Byte 14	:	mtHeadCommand Type; "1" = Normal , "29" = with appended 'V3B Gain Parameters' for Dual Channel. Extra Types (see Appendix): "30" = Reduced Command with Gain Parameters only, "15" = Scan Reverse Command (no Head Parameters attached).
Bytes 15-65	:	Head Parameter Info.
Bytes 66-81	:	Appended 'V3B Gain Parameters' (when Byte 14 = "29").

Description of 'Head Parameter Info' (Bytes 15 -> 65)

N.B. Read mtHeadCommand Byte 14 ('mtHeadCommand Type') to evaluate if the Head Parameter Info has an appended 'V3B Gain Parameter' block at the end for Dual Channel operation.

The 'Head Parameter Info' block includes the instruction set to inform the device how it should operate. It includes information such as transducer position and ping data, gain parameters and operating ranges.

Sonar 'Head Parameter Info' block

For an Imaging Sonar, the 'Head Parameter Info' block is 51 bytes in length…

Bytes 15,16	:	'HdCtrl' bytes.

Byte 17	:	'HdType'. Device Type *('02'/ '11' = Imaging Sonar; '11' = DST).*
Bytes 18-25	:	'TxN' Ch1/Ch2. Transmitter constants. *(N/A for DST)*
Bytes 26-33	:	'RxN' Ch1/Ch2. Receiver constants. *(N/A for DST)*
Bytes 34,35	:	'TxPulseLen'. Transmitter Pulse Length in usecs. *(N/A for DST)*
Bytes 36,37	:	'Range Scale' setting *(decimetre units).*
Bytes 38,39	:	'LLim'. Left Angle Limit *(1/16 Gradian units).*
Bytes 40,41	:	'RLim'. Right Angle Limit *(1/16 Gradian units).*
Byte 42	:	'ADSpan' *(1/255 units).*
Byte 43	:	'ADLow' *(1/255 units).*
Bytes 44,45	:	'IGain'. Initial Gain Setting for both channels *(1/210 units).*
Bytes 46-49	:	'Slope' setting for both channels *(1/255 units). (N/A for DST)*
Byte 50	:	'MoTime'. Motor Step Delay Time *(microsecond units).*
Byte 51	:	'Step'. Motor Step Angle Size *(1/16 Gradian units).*
Bytes 52,53	:	'AD Interval'.
Bytes 54,55	:	'Nbins'. Number of sample bins over scan-line.
Bytes 56,57	:	'MaxADbuf'.
Bytes 58,59	:	'Lockout' period *(microsecond units).*
Bytes 60,61	:	'Minor Axis' of dual-axis device *(1/16 Gradian units).* Ignore.
Byte 62	:	'Major Axis' *(1/16 Gradian units).* Always 1 for Sonar.
Byte 63	:	'Ctl2'.
Bytes 64,65	:	'ScanZ'.

V3B Gain Parameter Block (Dual Channel operation)

This should be appended to include Gain Parameters for both Channels. Byte 14 (mtHeadCommand Type) should then be set to "29" (Hex = 1D) to inform Dual Channel Sonar that the "V3B" Gain Parameters have been appended.

Bytes 66,67	:	'ADSpan' for both Channels.
Bytes 68,69	:	'ADLow' for both Channels.
Bytes 70,71	:	'IGain'. Initial Gain for both Channels.
Bytes 72,73	:	'Adc Setpoint'. Threshold for both Channels.
Bytes 74-77	:	'Slope' settings for both Channels. *(N/A for DST)*
Bytes 78-81	:	'Slope Delay' for both Channels. *(N/A for DST)*

HdCtrl (Bytes 19,20)

2-Byte Bitset controlling head operation;

Bit 0	:	'adc8on'	(* 0=4bit DataBins, 1=8Bit *)
Bit 1	:	'cont'	(* 0=SectorScan, 1=Continuous *)
Bit 2	:	'scanright'	(* ScanDirection 0=Left, 1=Right *)
Bit 3	:	'invert'	(* 0=Upright, 1=Inverted Orientation *)
Bit 4	:	'motoff'	(* 0=MotorOn, 1=MotorOff *)
Bit 5	:	'txoff'	(* 0=Tx on, 1=Tx off. For Test *)
Bit 6	:	'spare'	(* 0=Normal by default *)
Bit 7	:	'chan2'	(* hSON 0=Use Chan1, 1=Use Chan2 *)
			(* hSSS overridden BY pingpong *)
Bit 8	:	'raw'	(* 0=CookedADCmode, 1=RawADC *)
Bit 9	:	'hasmot'	(* 0=NoMotor, 1=HasMotor *)
Bit 10	:	'applyoffset'	(* 1=Applied Hdgoffset, 0=Ignore *)
Bit 11	:	'pingpong'	(* 1=pingpong Chan1/Chan2 e.g. hSSS *)
Bit 12	:	'stareLLim'	(* 1=Don't Scan, Point at LeftLim *)
Bit 13	:	'ReplyASL'	(* 1=ASLin ReplyRec, 0=NotIn *)
Bit 14	:	'ReplyThr'	(* 1=hThrRec Requested *)
Bit 15	:	'IgnoreSensor'	(* 1=Ignore the Centre Sensor *)

Description of 'HdCtrl' Bits

'**adc8on**' (Bit 0)

This bit governs whether the head returns 4-bit or 8-bit resolution from the sonar. When Bit 0 = 0 the head will return 4-bit packed echo data (0,,15) representing the amplitude of received echoes in a databin (Bin).
Default = 0 = 4-bit packed data.

'**cont**' (Bit 1)

This bit governs whether sonar scanning should be continuous rotation, or whether scanning should be restricted to a sector defined by the directions 'LeftLim' and 'RightLim'.
Default = 0 = Sector Scanning.

'**scanright**' (Bit 2)

This bit determines the scanning direction when cont=1. Scanright=0 = sonar rotates anticlockwise when viewed from top (scanning left). Scanright=1 causes clockwise rotation (scanning right). It is ignored when cont=0.
Default = 0 = ScanLeft.

'**invert**' (Bit 3)

This bit allows the rotation directions to be reversed if the sonar head is mounted inverted, i.e. when the sonar transducer boot is pointing downward rather than up.
Default = 0 = Sonar mounted upright, transducer boot pointing up.

'**motoff**' (Bit 4)

This bit allows the power to the scanning motor to be switched off for test purposes, or for power saving. The sonar will normally shut down motor power if it has not received any data or commands over a period of 30 seconds, but will automatically switch power on if new motor commands are received. Note: This mode is not available in certain Sonar models which includes the Micron and SeaSprite ranges.
Default = 0 = motor is enabled.

'**txoff**' (Bit 5)

This bit allows the sonar transmitter to be disabled for test purposes. The sonar will still act normally in all respects, except that no sonar transmissions will occur.
Default = 0 = sonar transmitter is enabled.

'**spare**' (Bit 6)

Reserved Bit. This bit should always set to 0
Default = 0 = Always.

'**chan2**' (Bit 7)

This bit selects which sonar channel is to be used. SeaKing DFS Sonars have 2 separate operating frequencies or channels (1=LF and 2=HF); chan2=0 = LF, chan2=1 = HF.
Usually, If the sonar head is a Sidescan, this bit will be over-ridden by the 'pingpong' bit.
Default = 0 = LF channel. Always = 0 for SeaPrince/MiniKing Sonars.

'**raw**' (Bit 8) (Byte 2, Bit 0)

This is a system level control bit, and should always be set to 1.
Default = 1 = Always.

'**hasmot**' (Bit 9) (Byte 2, Bit 1)

This bit is used to define whether the sonar head has a scanning motor. For a scanning sonar, hasmot=1. For a sidescan or sonar without any motor, hasmot = 0.
Default for Sonar = 1 = sonar has motor.

'**applyoffset**' (Bit 10)
This bit allows the direction of a scanning sonar to be dynamically modified by Heading Offset commands, typically generated by a compass, and is used for simple direction stabilisation operation of the sonar.
Default = 0 = Do not apply Heading Offsets.

'**pingpong**' (Bit 11)
This bit is typically used for Sidescan sonar operation, and causes dual frequency channels to alternate between channel 1 and channel 2, irrespective of the 'chan2' bit.
Default for scanning sonar = 0, Default for Sidescan sonar = 1.

'**stareLLim**' (Bit 12)
This bit allows a scanning sonar to 'Stare' in a fixed direction, defined by the 'LeftLim' direction parameter, and may be used for sidescanning or echosounding with a scanning sonar head when stareLLim = 1. Note: This mode is not available in certain Sonar models which includes the Micron and SeaSprite ranges.
Default = 0 = Don't "Stare" in fixed direction.

'**ReplyASL**' (Bit 13) (Byte 2, Bit 5)
This bit is set to cause analogue scan line (ASL) data to be returned from 'mtSendData' commands.
Default = 1 = Always for Sonar.

'**ReplyThr**' (Bit 14)
This bit is reserved for special functions and should be set to 0.
Default = 0 = Always.

'**IgnoreSensor**' (Bit 15) (Byte 2, Bit 7)
This is a diagnostic bit which disables scan motor position error checking for scanning sonar heads. If the sonar motor position sensor fails, then it may be set = 1 to allow the sonar to continue operating, but the direction of the sonar scan-lines will have unknown errors.
Default = 0 = Always, 1 in emergencies.

HdType (Byte 17)
The Device Type *(or Head Type)* informs the SeaKing Head as to what operating mode it should be in. Depending on the hardware, this operating mode will usually be fixed although it is possible to run a Profiler in an Imaging Sonar mode and vice-versa.

Imaging Sonar *(hSON)*	=	'02' / '11', *('11' = DST)*
Sidescan Sonar *(hSSS)*	=	'03',
Sub-Bottom Profiler	=	'03',
Profiling Sonar *(hPRF)*	=	'05'.

'TxN' / 'RxN' Transmitter Constants (Bytes 18-33)
The SeaKing DFS Sonar Heads have 2 independent Transmit and Receive channels (chan1, chan2) which can be programmed to operate at different frequencies.
SeaPrince/MiniKing/Micron Sonar Heads have 1 Transmit and Receive channel (chan1). It is recommended that for operation of these types of head, chan1=chan2.

Note: The Digital DST heads with Chirp transmission pulses will not apply these Transmitter/Receiver parameters. These parameters are now stored in non-volatile memory, within the mtBBUserData area of the program. For DST type Sonar heads, the TXN and RXN parameters can be filled with zeros or with the centre frequency of the Chirp Sonar transmit pulse.

The values for TXN and RXN are 32-bit LONGCARDINAL value, and is calculated from the following formula:

F = Transmitter Frequency in Hertz

TXN (chan) = F * 2^32 / 32e6

RXN (chan) = (F + 455000) * 2^32 / 32e6
E.g. chan1 = 325 kHz, chan2 = 675 kHz

TXN (chan1) BYTE 4,5,6,7 = 43620761
TXN (chan2) BYTE 8,9,10,11 = 90596966
RXN (chan1) BYTE 12,13,14,15 = 104689827
RXN (chan2) BYTE 16,17,18,19 = 151666032

TxPulseLen (Bytes 34,35)

This configures the length of the sonar transmit pulse, and is in units of microseconds. It is variable with the 'RangeScale' and for normal operation it should be constrained to between 50 .. 350 microseconds. A typical value is 100 microseconds.

TxPulseLen = [RangeScale(m) + Ofs] * Mul / 10 *(Use defaults; Ofs = 10, Mul =25)*

Note: The Digital DST heads with Chirp transmission pulses will not apply this Transmit Pulse Length parameter. This parameter is now stored in non-volatile memory, within the mtBBUserData area of the program. For DST type Sonar heads, the TxPulseLen parameter can be filled with zero or with a default value such as 100 microseconds.

Range Scale (Bytes 36,37)

The Range Scale field is not used to control the sonar head in any way, but is provided to allow Users to note the current Range Scale settings.
The low order 14 bits are set to a value of Rangescale * 10 units. Bit 6, Bit 7 of Byte 15 are used as a code (0..3) for the Range Scale units.
The user may define any range units, but the SeaKing system has default Range Scale units; 0 = Metres, 1 = Feet, 2 = Fathoms, 3 = Yards.

E.g.	20 metre range:	Range Scale = 200
	20 Yard range:	Range Scale = 200 + 2^14 + 2^15

Left Angle Limit (Bytes 38,39) / Right Angle Limit (Bytes 40,41)

These 2 words are used to determine the scanning sector of the Sonar Head, where 'LeftLim' *(Bytes 38,39)* refers to the Anti-Clockwise scan limit and 'RightLim' *(Bytes 40,41)* refers to the Clockwise scan limit.

Once set, the Sonar head will automatically steer to the next direction within these limits whenever a new 'mtSendData' command is received, and will automatically reverse scan direction at these scan limits if the 'cont' bit in 'HdCtrl' *(Bytes 15,16)* is clear.

The current units for these values are in 1/16th Gradian units, and are limited to the range 0..6399.

The SeaKing direction convention is as follows:

Left 90°	= 1600
Ahead	= 3200
Right 90°	= 4800
Astern	= 0 (or 6399)

The 'cont' bit in 'HdCtrl' will override these limits, and allow continuous rotation.
The 'stareLLim' bit in 'HdCtrl' will cause the head to be driven to the 'LeftLim' position and "stare" in that direction. If 'LeftLim' = 'RightLim', then the head will act as if the 'stareLLim' bit is set.

To Scan a 90° sector ahead, set:

LeftLim = 2400	(= Left 45°)
RightLim = 4000	(= Right 45°)

ADSpan (Byte 42) / ADLow (Byte 43)

'ADSpan' and 'ADLow' control the mapping of the received sonar echo amplitudes. The sonar receiver has an 80dB dynamic range, and signal levels are processed internally by the sonar head such that 0 .. 80dB = 0 .. 255.

For display purposes, a display dynamic range is required by the user, this is typically in the range 12dB to 24dB.

If adc8on in HdCtrl = 0, then the 80dB receiver signal is mapped to 4-bit, 16 level reply data values, to a display dynamic range defined by ADSpan such that:

For 12dB, ADSpan is set to 255 * 12 / 80 = ADSpan = 38
For 24dB, ADSpan is set to 255 * 24 / 80 = ADSpan = 77

The ADLow field determines the base position of the display dynamic range data in the 80dB sonar data window, and effectively provides some control over the sensitivity of the sonar. ADLow = 40 is a typical operating value.

To make the sonar appear more sensitive, set ADLow = 20, to make it appear less sensitive, set ADLow = 60.

E.g. ADSpan = 38 (12dB), ADLow = 40 (13dB)

4-bit Data Values 0..15 = Signal Amplitudes 0 = 13dB, 15 = 25dB

…if adc8on = 1 then the full 8-bit, 80dB dynamic range data bin amplitudes are returned to the user:

8-bit data Values 0..255 = Signal Amplitudes 0 = 0dB, 255 = 80dB

'ADSpan' and 'ADLow' are always returned in the mtHeadData Reply to allow the user to see what controls were effective during each 'ping', and may be used by the user to map full dynamic range 8-bit data to the users preferred display dynamic range.

IGain Setting (Bytes 44,45)

This is the 'Initial Gain' of the receiver. This control combined with the 'Slope' control applies the effective Gain to the sonar receiver. The 'Gain Setting' byte controls the initial gain of each sonar channel respectively (chan1 = Byte44, chan2 = Byte45), and this is in units 0..210 = 0..+80dB = 0..100%.
Typical default value of Sonar is 40% = 84.

Slope Setting (Bytes 46-49)

Slope values affect the Time Variable Gain (TVG) operation within the sonar head. The TVG compensates for spreading and attenuation losses of sound through water. A user can set any TVG value at any time. The table below indicates the normal Default SeaKing slope values. These would normally be variable by +/- 10 depending upon operating conditions. For other frequencies, select a default slope by interpolation. (Chan1 Slope = Bytes 46,47, Chan2 Slope = Bytes 48,49)

No TVG	Default Slope 0
200kHz	Default Slope 70
325 kHz	Default Slope 90
580 kHz	Default Slope 110
675 kHz	Default Slope 125
795 kHz	Default Slope 130
935 kHz	Default Slope 140
1210 kHz	Default Slope 150
2000 kHz	Default Slope 180

Note: The Digital DST heads with Chirp transmission pulses will not apply this TVG Slope parameter. This parameter is now stored in non-volatile memory, within the mtBBUserData area of the program. For DST type Sonar heads, the Slope Setting parameter can be filled with zero or with a default value from above table.

MoTime (Byte 50)

This byte sets the high speed limit of the scanning motor in units of 10 microseconds, and has a typical default value of…

MoTime = 25.

Step Angle Size (Byte 51)

This byte sets the scanning motor step angle between 'pings', and is in units of 1/16 Gradians. The SeaKing default settings are:

Low Resolution	= 32	(= 2 Gradians	= 1.8 °)
Medium Resolution	= 16	(= 1 Gradian	= 0.9°)
High Resolution	= 8	(= 0.5 Gradian	= 0.45°)
Ultimate Resolution	= 4	(= 0.25 Gradian	= 0.225°)

ADInterval (Bytes 52,53)

This data word is used to define the sampling interval of each Range Bin. It is in units of 640 nanoseconds. Its use is tightly coupled with the Number of Bins ('Nbins'). The maximum Nbins in SeaKing/SeaPrince/MiniKing/Micron Sonar heads is 800. In the newer digital DST range of Sonar heads, the maximum Nbins is increased to 1500. A practical minimum for ADInterval is about 5 (approx. 3 microseconds).
The ADInterval may also be rounded slightly to fit in with Receiver calculations (i.e. a request of 14830 may be adjusted to 14829 in the Sonar to fit into whole sampling periods for the required Range Scale.

Description of NBins (Bytes 54,55)

This data word defines the number of Range bins that the sonar will generate for the reply data message. The operating range of the sonar is defined by the time taken to do NBin samples at ADIntervals such that the scan line data time = Nbins of ADInterval * 640nanoseconds.

The maximum value of NBins is limited to 800 in SeaKing/SeaPrince/MiniKing/Micron Sonar heads. However, digital 'DST' heads are capable of up to 1500 Bins.

It is up to the user to decide whether fewer Bins at longer ADintervals, or more Bins at shorter ADIntervals are required.

E.G.
Require 10metre range, with 200 data samples (5cm range resolution) at VOS = 1500 m/sec.
Total Time for 'ping' sampling = 2 * 10 metres at 1500 m/sec = 13.3msecs

13.3msecs / 200 = 6.65 microsecs
66.5 microsecs / 640 nanosecs = 103.9

Choose ADInterval = 104.
Nbins = 200
Actual ScanTime = 200 * 104 * 640 nanoseconds
At 1500 m/sec = 13.312 milliseconds = 9.984 metres
= 49.92mm per bin.

MaxADbuf (Bytes 56,57)

This is a system setting, and should be set to default…

MaxADbuf = 500 *(Limit = 1000)*

Lockout (Bytes 58,59)
This value is the time in microseconds at which the Receiver will start sampling after it has transmitted. The factory default setting for this is…

Lockout = 100

Minor Axis Direction (Bytes 60, 61) / Major Axis Pan (Bytes 60, 62)
These are variable for Dual Axis Sonar devices. For the standard (Single Axis) devices they should be fixed at default…

Minor Axis Dir = 1600, Major Axis Pan = 1

Ctl2 (Byte 63)
The Ctl2 Byte was added to allow extra Sonar Control Functions to be implemented for operating and test purposes. This is an extension to HdCtrl byte.
For a Sub-Bottom Profiler, enable Bit 1 ('SendDetects') to enable the Detects within the Sonar which will result in an 'ExtraDetects' message appended at the end of the 'mtHeadData' reply.
Bits 2-7 are for test purposes. (Default = 0)

Bit 0	:	'SendTime'	(* Extra hSON Time Record *)
Bit 1	:	'SendDetects'	(* Extra Send Detects *)
Bit 2	:	'ReplyV4'	(* Test *)
Bit 3	:	'OnParamsGLLim'	(* FOR Sonar DAX Use *)
Bit 4	:	'V4Df_PositionOn'	(* 0=MotorOn, 1=MotorOff *)
Bit 5	:	'V4Df_AmplOn'	(* FOR V4 Test Ampl *)
Bit 6	:	'FlyBack'	(* FOR DAX SonFlyback *)
Bit 7	:	'SP7'	(* Spare bit *)

ScanZ (Byte 64)
This is for Special devices and should both be left at default values of 0.

5) 'mtSendData' Command Message (ID = '25')

This is the command that is sent to the subsea device (i.e. Sonar or Profiler) in order to instruct the device to perform one sample set.

Rev.2 Update

Issuing an 'mtSendData' to an RS-232 Sonar will instruct this type of device to perform **two** transmit and receive cycles (commonly called a 'Ping'). Then, at the end of each Ping, advances the transducer position ready for the next Ping before it will then transmit an 'mtHeadData' Reply Message back to the Sender of the 'mtSendData' (For RS-485 Micron/MiniKing heads, only 1 x Ping and mtHeadData Reply will be sent for each issued 'mtSendData'… see 'mtBBUserData' Half-Duplex bit).
Issuing an 'mtSendData' to a Profiler will instruct this type of device to perform **two** full scans, each containing a number of Pings. Each Ping will give a Slant Range measurement that can be used to construct a Scan Profile back at the surface, i.e. 1 Ping = 1 Point in the Scan Profile. After each scan, an 'mtHeadData' Reply will be transmitted with the Ping/Point data for that scan.

Rev.2 Note

Important: Read mtBBUserData 'Half-Duplex' bit to confirm if 1 or 2 'mtHeadData' Replies will be returned for every 1 x 'mtSendData' command sent to the head. If 'Half Duplex' = 1 then head is set to Half Duplex and 1 x 'mtHeadData' Reply will be returned for every 1 x 'mtSendData'. If 'Half Duplex' = 0 then head is set to Full Duplex and 2 x 'mtHeadData' Replies will be returned for every 1 x 'mtSendData'. In Full Duplex mode (RS-232 head only), 2 Pings are performed for every 'mtSendData' in order to speed up operation.

Once 'Device Parameters' have been sent to the subsea device, sending repeated 'mtSendData' commands to it are all that is then necessary to operate the device. Sonars and Profilers have the capability to buffer one 'mtSendData' whilst in the middle of executing another. Therefore, on

commencement of the 'mtSendData' sequence, the User program can send 2 x 'mtSendData' one after the other to the subsea device. The device will execute the 1st mtSendData and then buffer the 2nd. This way there is no delay in the device waiting for the User program to send the 2nd command down, and so on thereafter. Maintaining a write-ahead of 1 x mtSendData will speed up the inter-scan operation.
N.B. The SeaNet 'mtSendData' performs the same function as the SONV3 'GetDataCmd'.

The 'Current Time' is also sent to the subsea device using this command in order to update the device's onboard clock to be in synchronisation with the surface clock. If time synchronisation is not required then the 'Current Time' field can be set to zero.
For an Imaging Sonar or Sidescan Sonar the mtHeadData replies are not time-stamped and so 'Current Time' can be set to zero for these devices.

The sequence of Commands and Replies used to perform one Scanline for a Sonar is as follows…

A. Serial Port sends **mtSendData** Command (18 bytes)

40	**30**	**30**	**30**	**43**	**0C**	**00**	**FF**	**02**	**07**	**19**	**80**	**02**
Hdr '@'	Hex Length = 12 bytes				Bin Length = 12 bytes		Tx Nde 255	Rx Nde 02	No. Byte = 7	**mtSend D'ta**	Seq = End	Nde 02

CA	**64**	**B0**	**03**	**0A**
Current Time = 61891786 msec = 17:11:31.79				LF

B. Sonar replies with '**mtHeadData**' **Scanline Data Reply** (45 x 8-bit Data Bins) = Data

Scanline Packet Structure *- Full description further on*

40	**30**	**30**	**35**	**34**	**54**	**00**	**02**	**FF**	**00**	**02**	**80**	**02**	**4C**
Hdr '@'	Hex Length = 84 bytes				Bin Length = 84 bytes		Tx Nde 02	Rx Nde 255	Sin-gle pckt	**mtHead D'ta**	Seq = End	Nde 02	->

00	**02**	**10**	**05**	**85**	**A3**	**3C**	**00**	**66**	**66**	**66**	**05**	**6B**	**7D**
Count = 76	= Sonhead	Stat-us	Sw-eep	HdCtrl = 41861		Range = 6m		TxN = 1717986821				Gain = 51%	->

00	**32**	**2C**	**00**	**00**	**6B**	**00**	**40**	**06**	**C0**	**12**	**10**	**80**	**0A**
Slo-pe= 125	AD Sp'n	AD Low	Heading Offset = 0		AD Interval = 107		L.Limit = 1600		R.Limit = 4800 (1/16 Grad)		Ste-ps = 16	Bearing = 2688 (1/16 Grad)	

2D	**00**	**31**	**4B**	**78**	**76**	**75**	**65**	**4D**	**31**	**16**	**10**	**00**	**00**
Dbytes = 45		Bin 1	Bin 2	Bin 3	Bin 4	Bin 5	Bin 6	Bin 7	Bin 8	Bin 9	Bin 10	Bin 11	Bin 12

00	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**
Bin 13	Bin 14	Bin 15	Bin 16	Bin 17	Bin 18	Bin 19	Bin 20	Bin 21	Bin 22	Bin 23	Bin 24	Bin 25	Bin 26

00	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**	**00**
Bin 27	Bin 28	Bin 29	Bin 30	Bin 31	Bin 32	Bin 33	Bin 34	Bin 35	Bin 36	Bin 37	Bin 38	Bin 39	Bin 40

00	**00**	**00**	**00**	**00**	**0A**
Bin 41	Bin 42	Bin 43	Bin 44	Bin 45	LF

IMPORTANT: Byte 10 ("Byte Count") = 0 in above example to indicate that Multi-packet mode is not used by this device and that all 'mtHeadData' replies will be single packet messages. **Note:** Byte 10 will always display a byte count in other message types but will only display a byte count in 'mtHeadData' replies when that device uses Multi-packet mode.

> ** ***NOTE*** **
>
> ***A Further Example is provided in the Appendix that describes a multi-packet 'mtHeadData' Reply***

6) 'mtHeadData' Reply Message (ID = '2')

Rev.2 Update

These messages are the replies sent by the subsea device in response to 'mtSendData' commands. They contain all the device sample data and in addition a copy of the device parameters. For RS-232 heads with mtBBUSerData 'Half-Duplex' bit = 0, issuing one mtSendData will instruct the subsea device to perform **two** sample sets or Ping sequences (RS-485 heads will perform only one sample set or Ping sequence);

- In the case of an RS-232 Sonar, the mtSendData will trigger two Pings (two Transmit / Receive cycles) which will each return one Scanline of 'Reply Data' in an 'mtHeadData' reply.

- For a Profiler, the mtSendData will instruct the device to perform two Scans of 'Reply Data' which will return typically <800 profile points *(1 Ping = 1 point)* in each 'mtHeadData' reply.

At the end of the Ping sequence, the subsea device will transmit an 'mtHeadData' Reply Message back to the Sender of the 'mtSendData' *(i.e. Rx Node 255 = AIF card or Serial Port)*. This Reply Message is constructed of a 13-byte 'Message Header' followed by a 'Device Parameter' block and then 'Reply Data'.

Notes on Multi-packet mode:

For devices that use the Multi-packet mode, the Device Parameter block will always be sent in the 1st Data Packet. The Reply Data will be sent at the end of each data packet that may be part of a multi-packet sequence.

The Message Sequence byte (Byte12) and Byte Count (Byte10) should first be read to determine if this is a multi-packet sequence. The Message Sequence Byte also indicates what the current data packet sequence number is. All packets within the sequence will begin with the 13-byte Message Header. The Reply Data will immediately follow this unless it is the first (or only) packet in the sequence in which case there is a Device Parameter block that follows the Message Header and precedes the Reply Data. With multi-packet messages, the Message Header would be stripped from each packet and then decoded to identify the type and nature of the message. With the "mtHeadData" reply, the Device Parameters would then be stripped from the 1st packet before reading in and buffering the Reply Data that follows at the end of each packet. When the 'End Sequence' bit (Byte 12, bit 7) is eventually encountered, this informs the User that this is the last packet in the sequence. The full message would then be stitched back together at the surface, including the Device Parameter block and all the Reply Data, before further processing. Examples will be given later in this document to best describe the required methodology for this.

'mtHeadData' Packet format is;

Byte 1	:	Message Header = "@".
Bytes 2-5	:	Hex Length of binary packet from byte 6 onwards *(excludes LF)*.
Bytes 6,7	:	Binary Word of above Hex Length.
Byte 8	:	Packet Source Identification (Tx Node number 0 - 240).
Byte 9	:	Packet Destination Identification (Rx Node number 255).
Byte 10 **	:	Byte Count of attached message that follows this byte.
Byte 11	:	Message Type = "2" for mtHeadData.
Byte 12	:	Message Sequence Bitset *(see below)*.
Byte 13	:	Tx Node number *(copy of Byte 8)*.
Byte 14 ->	:	Device Parameters / Reply Data. N.B. First read the Message Sequence *(Byte 12)*.
End Byte	:	Message Terminator = LF *(0Ah)*.

** *Set to 0 (zero) in 'mtHeadData' reply to indicate Multi-packet mode NOT used by device.*

Message Sequence (Byte 12)

This is a Bitset that should be processed for Multi-packet modes in order to determine if the "mtHeadData" reply message exceeds 128 bytes and will therefore be split into sequential packets of less than 128 bytes in length.

Note: Should always = 80h for devices that do not use the Multi-packet mode.

The Message Sequence Bitset is constructed as follows;

Bits 0 - 6	:	Packet Sequence bits. 1st packet has Packet Sequence of 0, 2nd packet has Packet Sequence of 1 with bit 0 = 1 *(0000001)*, 3rd packet has Packet Sequence of 2 *(0000010)*, 4th packet has Packet Sequence of 3 *(0000011)…etc.*
Bit 7	:	End Sequence bit. Set to 1 when it is the last packet in sequence. For instance, the last packet in a 3-packet sequence will have a Bitset of *10000010 (=82H)*.

Description of 'Device Parameters' / 'Reply Data' (Bytes 14 ->)

Device Parameter Block

The 'Device Parameter' block will always be sent in the 1st data packet and immediately follow the 13-byte Message Header. The 'Device Parameter' block will always be followed by the 1st block of Reply Data. For Multi-packet mode devices, the 2nd data packet will have the 13-byte Message Header followed immediately by the 2nd block of Reply Data, and so on. The Device Parameter block is only sent once and this is in the 1st data packet of a sequence only.

Sonar 'Device Parameter' block

For a Sonar (Node 2), the 'Device Parameter' block is 30 bytes in length…

Bytes 14,15	:	Total Byte Count of Device Parameters + Reply Data (all packets).
Byte 16	:	Device Type *(02 = Sonar, 11 = DST Sonar)*.
Byte 17	:	'Head Status' byte.
Byte 18	:	Sweep Code.
Bytes 19,20	:	'HdCtrl' bytes
Bytes 21,22	:	Range Scale setting.
Bytes 23-26	:	TxN. Transmitter number *(can be ignored)*.
Byte 27	:	Gain Setting *(1/210 units)*.

Bytes 28,29	:	Slope Setting *(1/255 units)*.
Byte 30	:	ADSpan *(1/255 units)*.
Byte 31	:	ADLow *(1/255 units)*.
Bytes 32,33	:	Heading Offset.
Bytes 34,35	:	AD Interval.
Bytes 36,37	:	Left Angle Limit *(1/16 Gradian units)*.
Bytes 38,39	:	Right Angle Limit *(1/16 Gradian units)*.
Byte 40	:	Step Angle Size *(1/16 Gradian units)*.
Bytes 41,42	:	Transducer Bearing.
Bytes 43,44	:	'Dbytes'. This number indicates the quantity of Reply Data / Bins in the current message.

Total Byte Count (Bytes 14,15)

This Byte count includes the total number of bytes in the Device Parameter block + Reply Data that are contained in all packets. This should indicate the length of message that is re-created from all data packets after each of the 13-byte Message Headers has been stripped and Device Parameters and Reply Data are "stitched" back together from the data packet sequence.

Device Type (Byte 16)

The Device Type *(or Head Type)* informs the SeaKing Head as to what operating mode it should be in. Depending on the hardware, this operating mode will usually be fixed although it is possible to run a Profiler in an Imaging Sonar mode and vice-versa.

Imaging Sonar *(hSON)*	=	'02',
Sidescan Sonar *(hSSS)*	=	'03',
Sub-bottom Profiler	=	'03',
Profiling Sonar *(hPRF)*	=	'05',
Digital Img Sonar *(hDIG)*	=	'11'.

Head Status (Byte 17)

1-Byte Bitset where;

Bit 0	:	'HdPwrLoss'. Head is in Reset Condition.
Bit 1	:	'MotErr'. Motor has lost sync, re-send Parameters.
Bits 2,3	:	Always 0 ('PrfSyncErr', 'PrfPingErr').
Bit 4	:	When in 8-Bit ADC DataMode. Data is 0..255 = 0..80dB.
Bits 5,6	:	*RESERVED (Ignore)*.
Bit 7	:	Message Appended after last packet Data Reply.

Sweep Code (Byte 18)

'0'	=	Scanning_Normal,
'1'	=	Scan_AtLeftLimit,
'2'	=	Scan_AtRightLimit,
'3','4'	=	*RESERVED (Ignore)*,
'5'	=	Scan_AtCentre (Ahead) Position.

HdCtrl (Bytes 19,20)

- Refer to mtHeadCommand for description

Range Scale (Bytes 21,22)

- Refer to mtHeadCommand for description

'TxN' Transmitter Constant (Bytes 23,24,25,26)

- Refer to mtHeadCommand for description

Gain Setting (Byte 27)

- Refer to mtHeadCommand for description

Slope Setting (Bytes 28,29)

- Refer to mtHeadCommand for description

ADSpan (Byte 30) / ADLow (Byte 31)

- Refer to mtHeadCommand for description

Heading Offset (Bytes 32,33)

This offset is used with Sonars which have the compass option as a special configuration.
Default = 0 = Always for standard Sonars.

ADInterval (Bytes 34,35)

- Refer to mtHeadCommand for description

Left Angle Limit (Bytes 36,37) / Right Angle Limit (Bytes 38,39)

- Refer to mtHeadCommand for description

Step Angle Size (Byte 40)

- Refer to mtHeadCommand for description

Transducer Bearing (Bytes 41,42)

This is the position of the Transducer for the current Scanline (0 .. 6399 in 1/16 Gradian units).

'Dbytes' (Bytes 43,44)

Having read and processed the Device Parameter block data that was preceded, the 'DBytes' value is used to control reading the Sonar Scanline data Bins that follow.

'Dbytes' defines the number of Range Bins that the sonar will generate for the Head Data reply message. The operating range of the Sonar is defined by the time taken to do 'Nbin' samples at an ADInterval such that the Scanline data time = 'Nbins' of ADInterval * 640nanoseconds.

The maximum value of 'Nbins' is limited to 800 in SeaKing/SeaPrince/MiniKing/Micron heads and is limited to 1500 in newer digital 'DST' heads. The Sonar will always return an EVEN number of Bins, regardless of whether an odd number has been requested. For example, if 7 bins were requested then 8 bins would be sampled and returned. An odd number is always rounded up to the nearest Even number.

It is up to the user to decide whether fewer Bins at longer ADIntervals, or more Bins at shorter ADIntervals are required.

E.g. Require 10 metre range, with 200 data samples (5cm range resolution) at VOS = 1500 m/sec.

Total Time for 'Ping' sampling = 2 * 10 metres at 1500 m/sec = 13.3msecs

13.3msecs / 200 = 6.65 microsecs
66.5 microsecs / 640 nanosecs = 103.9

Choose ADInterval = 104.
Nbins = 200

Actual ScanTime = 200 * 104 * 640 nanosecs
At 1500 m/sec = 13.312 milliseconds = 9.984 metres (= 49.92mm per bin)

With the value for the Number of Bins ('Nbins') it can then be deduced how many Data Bytes ('Dbytes') there will be in the Head Data reply message;

If 4-bit ADC, 'Dbytes' = 'Nbins' / 2
If 8-bit ADC, 'Dbytes' = 'Nbins'

Description of Reply Data Bytes

Reply Data
The Reply Data will differ in format and this will depend on the type of sending device *(i.e. Sonar or Profiler)*. Bytes 43,44 (Sonar) / Bytes 39,40 (Profiler) of the 'Device Parameter' block should be read to determine the quantity of Reply Data there will be…

Profiling Sonar	:	'Dbytes' (Bytes 39,40) of Profiler Scan data. In this case 'Dbytes' signifies the **'Reply Data'** number of points in the Scan or in other words the number of pings taken and the sample data for each ping. Sample data for each ping will be transmitted using 2-bytes of data, each containing the echo reply time in microseconds *(time taken for ping to travel to reflective target and back)*.
Imaging Sonar 'Reply Data'	:	'Dbytes' (Bytes 43,44) of Sonar Scan-line data.

Byte 12 in the header will indicate what the sequence number of the current packet is and whether the 'End Sequence' bit (bit7) is set thus signifying that this is the last block of Reply Data sent by the transmitting device.

Sonar Scanline Data Points (1st or Single packet = Bytes 45->)
(2nd to 'End Sequence' packet = Bytes 14->)

This contains the data for each of the binary samples ('Bins') that collectively make up the Scanline data for 1 Ping. Each 'Bin' is either a 4-bit or 8-bit value…

If the data is in packed 4-bit mode then there will be twice as many Data Bins as Data Bytes.
For 4-bit and 8-bit bins, the Sonar will always return an EVEN number of range BINS, even if an Odd number has been requested. Odd numbers will be rounded up to the nearest Even value.

Examine HdCtrl.adc8on (BYTE19, BIT0)
adc8on, Bit0 (* 0=4bit Data Bins, 1=8bit *)

IF adc8on the Data Bins = Dbytes,
ELSE Data Bins = Dbytes * 2

If adc8on = 0 then the 4-bit data bins are packed as follows:

	BYTE45	BYTE46	BYTE47	……..
BIT	7654 3210	7654 3210	7654 3210	……..
BIN	0000 1111	2222 3333	4444 5555	……..

If adc8on = 1 then the 8-bit data bins are packed as follows:

	BYTE45	BYTE46	BYTE47	……..
BIT	7654 3210	7654 3210	7654 3210	……..
BIN	0000 0000	1111 1111	2222 2222	……..

7) 'mtAlive' Broadcast Message (ID = '4')

When a subsea device is first powered up it will initially go through a power-up routine. For a Sonar or Profiler this routine will involve 'arming' the stepper motor and centring the transducer so it will be ready to start it's scanning at any time. Once this routine is complete, the subsea device will start to transmit a short broadcast message at a 1-second time interval. This broadcast will be aimed at the Surface Controller which will either be an AIF card or Serial Port (both with Node 255 ID). On reception of an 'mtAlive' message, the Surface Controller will be aware of this new device connection and also be informed the current state of that device. In particular, the device will broadcast whether it has a valid set of Parameters or not and also the state of the Motor and Transducer.

'mtAlive' Packet format is;

Byte 1	:	Message Header = "@".
Bytes 2-5	:	Hex Length of binary packet from byte 6 onwards *(excludes LF)*.
Bytes 6,7	:	Binary Word of above Hex Length.
Byte 8	:	Packet Source Identification (Tx Node number 0 - 240).
Byte 9	:	Packet Destination Identification (Rx Node number 255).
Byte 10	:	Byte Count of attached message that follows this byte.
Byte 11	:	Message Type = "4" for mtAlive.
Byte 12	:	Message Sequence BitSet *(always = 80 for this command)*.
Byte 13	:	Tx Node number *(copy of Byte 8)*.
Byte 14	:	'WillSend' byte.
Bytes 15-18	:	Head Time of subsea device.
Bytes 19,20	:	Stepper Motor Position.
Byte 21	:	'HeadInf' byte (Motor and Transducer state).
Byte 22	:	Message Terminator = LF *(0Ah)*.

WillSend (Byte 14)

For the digital 'DST' range of Sonar heads, the 'WillSend' Byte is no longer used to signify that the Sonar has received and validated a set of parameters sent by the surface controller via the 'mtHeadCommand'. This Byte was previously toggled from 80h to 00h whenever parameters had been received and validated by the Sonar. This is no longer the case and Byte 21 ('HeadInf'), Bits 6 and 7 are the only flags that should now be read to indicate the status of Parameter setting in the DST head.

Head Time (Bytes 15,16,17,18)

SeaKing subsea devices including Sonars, Profilers and Bathys have their own on-board clock modules built in. These clocks are used to time-stamp the sample data before sending that data to the Surface Controller. For purposes of synchronising the device's clock with the Surface clock, the Head Time field can be used to verify this. Note that each mtSendData command sent to the device should contain the Current Time. This current time will be used to reset the onboard clock within the subsea device.

The subsea device 'Head Time' is the elapsed time into the day in milliseconds. On power up of the subsea device, the clock will be set at 0 (zero) milliseconds. A Time of Day in milliseconds will be sent to the device in the first and subsequent mtSendData commands. At midnight, the 'Head Time' clock will reset back to 0 (zero). During periods when no mtSendData commands are being received, the device's onboard clock will continue to update the Head Time.

Stepper Motor Position (Bytes 19,20)

When the subsea device has No Parameters, and has just been powered, it will re-centre. The 'HeadInf' byte will indicate that the Transducer has re-centred (or not!) and the 'Stepper Motor Position' will indicate the physical position of this. At power up, centre will be 180 degrees for a Profiler (i.e.

pointing down) or 0 degrees for Sonar (i.e. looking ahead). Note that both these positions are physically at the same point but the Profiler axis is 180 degree shifted to reflect it's downwards operating position. The Stepper Motor Position is given in units of 1/16 Gradians and should be checked along with the 'HeadInf' byte on device power up, or after an 'mtReBoot' hardware reset.

HeadInf BitSet (Byte 21)

1-Byte Bitset where;

Bit 0	:	'InCentre'. Transducer is in middle of a re-centre operation.
Bit 1	:	'Centred'. Transducer is at centre position ready to start scan.
Bit 2	:	'Motoring'. Motor is motoring (can be re-centring or in scan).
Bit 3	:	'Motor On'. Device is powered and Motor is primed.
Bit 4	:	'Dir'. Transducer is off-centre.
Bit 5	:	'InScan'. Device is currently performing a Scan/Ping routine.
Bit 6	:	'NoParams'. Device needs Parameters before it can Scan/Ping.
Bit 7	:	'SentCfg'. Acknowledgement of 'Params' received.

The 'HeadInf' byte is a Bitset that reflects the current Transducer and Motor states which can be used for verification when the device has just been powered up and to check if it is ready to start scanning (i.e. Has Parameters, Transducer Centred, Motor On).

Bits 6,7 Operation

If the Surface Controller is just taking control of the subsea device it must first ensure that the device has a Parameter set before it can start issuing mtSendData requests. Bits 6 and 7 should be read to determine if the Sonar has received a validated set of parameters from the surface controller (i.e. it has received a valid 'mtHeadCommand').

Bit 6 will be set to **1** if the device has No Parameters. If this is the case then an 'mtHeadCommand' message must be sent to the subsea device to upload the parameters to it.

If **Bit 6** is set to **0** then this will indicate that the device already Has Parameters. If the User program is just taking control of the subsea device then it would be advised at this stage to perform a hardware Stop and Reset. This is performed by sending an 'mtReBoot' command to the device which will clear the parameter block. The 'mtHeadCommand' should then be sent which will include a full set of known parameters.

Bit 7 will indicate whether the Sonar has ever received a set of Parameters (i.e. 'mtHeadCommand') from the surface controller since the Sonar was powered up and/or rebooted.
Bit 7 should be used in conjunction with Bit 6. If **Bit 7** is set to **1** then this would indicate that an mtHeadCommand has already been received. **Bit 6** should then be checked until it is set to **0** to indicate a validation of the received Parameters. The surface controller can then issue mtSendData requests once Bit 7 is 1 and Bit 6 is 0.

An example of an 'mtAlive' broadcast sent by a Sonar on power up is as follows…

A. Sonar sends **mtAlive** message on power up (22 byte)

40	**30**	**30**	**31**	**30**	**10**	**00**	**02**	**FF**	**0B**	**04**	**80**	**02**	**80**
Hdr '@'	Hex Length = 16 bytes				Bin Length = 16 bytes		Tx Nde 02	Rx Nde 255	No. Byte = 11	**mtA live**	Seq = End	Nde 02	Will Send

AA	**10**	**00**	**00**	**80**	**0C**	**5D**	**0A**
Head Time = 4266 msec				Motor Pos = 3200 (1/16 Grad)		He-Ad Inf*	LF

** Transducer In Centre, Motor On, No Parameters.*

On power-up of the Sonar, the 'HeadInf' byte will change in value to reflect the different states that the Sonar is going through during initialisation…

i) Power applied to Sonar;
1st Alive: HeadInf = '5D' (Head is in Re-Centre operation and has No Params). Byte 14 = 80h.
2nd Alive: HeadInf = '4D' (No 'Dir' so Transducer is back at centre)
3rd Alive: HeadInf = '4A' (Transducer is now centred and Not Motoring)

ii) 'mtHeadCommand' sent to Sonar;
4th Alive: HeadInf = 'CA' (Has been 'Sent Cfg'. Acknowledgement of 'ParamsCmd')
5th Alive: HeadInf = '8A' (Has Params. Parameters have been validated. Ready for 'GetData').

8) 'mtBBUserData' Reply Message – Digital DST Sonar Head (ID = '6')

This Reply is a device specific record that defines various settings and options programmed into the subsea device. The settings are not required for Sonar simulator programs and otherwise it is okay to do a quick verification of the settings particularly if special features have been programmed into the head (i.e. Sonar 'Aux' port serial interface, Pressure/Temp sensor interface). All these settings can also be accessed through the 'SeaNet Setup' Windows® Utility Program by clicking on 'Action' – 'Setup' for the Node.

For the DST Sonar heads with Chirp Technology, the Start and End Frequencies of the Chirp Pulse will be found here. It is also possible to make changes to these values in the 'Seanet Setup' utility program.

'mtBBUserData' Packet format is;

Byte 1	:	Message Header = "@".
Bytes 2-5	:	Hex Length of binary packet from byte 6 onwards *(excludes LF)*.
Bytes 6,7	:	Binary Word of above Hex Length.
Byte 8	:	Packet Source Identification (Tx Node number 0 - 240).
Byte 9	:	Packet Destination Identification (Rx Node number 255).
Byte 10	:	Byte Count of attached message that follows this byte.
Byte 11	:	Message Type = "6" for mtBBUserData.
Byte 12	:	Message Sequence Bitset *(see below)*.
Byte 13	:	Tx Node number *(copy of Byte 8)*.
Bytes 14->25	:	SystemBB Parameter Block.
Bytes 26->59	:	BaudBB Parameter Block.
Bytes 60->174	:	DigSonCfg Parameter Block.
Byte 175	:	Message Terminator = LF *(0Ah)*.

Description of 'UserBB' Parameter Block (Bytes 14->174)

The UserBB Parameter block is sectioned into 3 separate blocks; **1.** 'SystemBB', **2.** 'BaudBB', **3.** 'DigSonCfg'.

Each of these sections contains User configurations and settings that are programmed into the subsea device to give it an instruction set of how to operate. Again, these settings are accessed in the 'SeaNet Setup' Windows® Utility Program by entering the Node's setup section ('Action' – 'Setup'). It is in this section that settings can be changed and new settings downloaded/programmed into the subsea device. Because of the sensitivity of a number of the 'SonBB' settings, these can only be viewed and changed in a Supervisor access level that is a restricted "factory only" access level.

1. 'SystemBB' Parameter block (Bytes 14->25)

This parameter block is essentially a header ID that includes system information. This block is 12 bytes in length.

Byte 14	:	ID Number of Surface Controller. This will always be Node 255 for an AIF card or Serial Port.
Byte 15	:	'D1' Byte.
Bytes 16,17	:	'CK' Word.
Bytes 18,19	:	UserBBLen. This is the total number of bytes contained in the 'SystemBB', 'BaudBB' and ' DigSonCfg' Parameter blocks.
Byte 20	:	Board *Type ("15" = Generic Digital).*
Byte 21	:	Software Version *(Software programmed in using Seanet Setup).*
Bytes 22,23	:	Spare3 *(Reserved).*
Bytes 24,25	:	Spare4 *(Reserved).*

2. 'BaudBB' Parameter block (Bytes 26->59)

This parameter block defines what the telemetry settings are for each port in the subsea device. There are 3 x Telemetry ports; **1.** 'Lan' = ARCNET Port, **2.** 'Async 0' = RS-232 Main Port, **3.** 'Async 1' = RS-232 AUX Port.

Each of the 3 x Telemetry ports has 2 settings associated with it; a 'Low' Setting and a 'High' Setting. In the 'Action' – 'Setup' section of 'SeaNet Setup', the 'Low' setting is the Left Hand column and the 'High' setting is the Right Hand column. N.B. There is also a hardware switch located on the 'CPU' PCB inside the subsea device that is used to select between 'Low' and 'High'.

It is the 'Async 0' Main Port that is used in the subsea device to connect to the Serial Port in the Surface Controller. The 'Async 1' Aux Port can be used to communicate with, or receive and on-pass LineFeed terminated data from, another device such as an Altimeter.

This Parameter block is 34 bytes in length.

Bytes 26,27	:	'Lan' Baud, Low Setting ('0' = 2.5M*, '1' = 1.25M*, '2' = 625k*, '3' = 312k*, '4' = 156k, '5' = 78k). * = Not Used.
Bytes 28,29	:	'Lan' Baud, High Setting ('0' = 2.5M*, '1' = 1.25M*, '2' = 625k*, '3' = 312k*, '4' = 156k, '5' = 78k). * = Not Used.
Bytes 30,31	:	'Lan' Sensitivity, Low Setting ('0' = Increase, '1' = Decrease, '2' = Neutral*). * = Normal setting.
Bytes 32,33	:	'Lan' Sensitivity, Low Setting ('0' = Increase, '1' = Decrease, '2' = Neutral*). * = Normal setting.
Bytes 34,35	:	'LanETmo'. Lan Timeout value ('0' -> '3'). Should be '3'.
Bytes 36,37	:	'Async 0' Baud, Low Setting ('0' = 1.2k*, '1' = 2.4k*, '2' = 4.8k*, '3' = 9.6k#, '4' = 19.2k, '5' = 38.4k, '6' = 57.6k, '7' = 115.2k, '8' = 153.6k*). * = Not Used, # = Default.
Bytes 38,39	:	'Async 0' Baud, High Setting ('0' = 1.2k*, '1' = 2.4k*, '2' = 4.8k*, '3' = 9.6k#, '4' = 19.2k, '5' = 38.4k, '6' = 57.6k, '7' = 115.2k, '8' = 153.6k*). * = Not Used, # = Default.
Bytes 40,41	:	'Async 0' Parity, Low Setting ('0' = None#, '1' = Odd*, '2' = Even*, '3' = Mark*, '4' = Space). * = Not Used, # = Default.
Bytes 42,43	:	'Async 0' Parity, High Setting ('0' = None#, '1' = Odd*, '2' = Even*, '3' = Mark*, '4' = Space). * = Not Used, # = Default.
Bytes 44,45	:	'Async 0' Data Bits, Low Setting ('0' = 8 bits#, '1' = 7 bits*). * = Not Used, # = Default.
Bytes 46,47	:	'Async 0' Data Bits, High Setting ('0' = 8 bits#, '1' = 7 bits*). * = Not Used, # = Default.
Bytes 48,49	:	'Async 1' Baud, Low Setting *(same options as 'Async 0' - Bytes 36,37).*
Bytes 50,51	:	'Async 1' Baud, High Setting *(same options as 'Async 0' - Bytes*

38,39).

Bytes 52,53 : 'Async 1' Parity, Low Setting *(same options as 'Async 0' - Bytes 40,41).*

Bytes 54,55 : 'Async 1' Parity, High Setting *(same options as 'Async 0' - Bytes 42,43).*

Bytes 56,57 : 'Async 1' Data Bits, Low Setting *(same options as 'Async 0' - Bytes 44,45).*

Bytes 58,59 : 'Async 1' Data Bits, High Setting *(same options as 'Async 0' - Bytes 46,47)*

3. 'DigSonCfg' Parameter block (Bytes 60->174)

Byte 60 : 'Generic Board Type' ('1' = SeaKing V6, '2' = SeaPrince V6, '3' = Micron DST V6).

Byte 61-64 : 'CH1 Rx Start Frequency' (in Hertz, Min = 10000, Max = 200000).

Byte 65-68 : 'CH1 Rx End Frequency' (in Hertz, Min = 10000, Max = 200000).

Byte 69-72 : 'CH1 Filter Length' (in microseconds, should be <= expected Rx Pulse Length).

Byte 73-76 : 'CH1 Sample Rate' (in Hertz, the frequency of the Rx Filter).

Byte 77,78 : 'CH1 Filter Alpha Shading' (1/100 %).

Bytes 79,80 : 'CH1 TVG Slope Correction' (%).

Byte 81 : 'CH1 MacBlocks' (Number of blocks in Match Filter).

Byte 82 : 'CH1 Phase Limit'.

Byte 83 : 'CH1 Phase Scalar'.

Byte 84 : 'CH1 Rssi Offset'.

Byte 85 : 'CH1 Rx Detect Threshold (dB, 0 = Off).

Bytes 86,87 : 'CH1 Rx Detect Min Pulse Width (microseconds).

Bytes 88,89 : 'CH1 Rx Detect Correction Offset'.

Bytes 90-93 : 'CH1 Tx Start Frequency' (in Hertz, Min = 10000, Max = 200000).

Byte 94-97 : 'CH1 Tx End Frequency' (in Hertz, Min = 10000, Max = 200000).

Byte 98-101 : 'CH1 Tx Pulse Length' (in microseconds, Min = 10, Max = 250000).

Byte 102 : 'CH1 Tx Duty Cycle' (1..15).

Byte 103 : 'CH1 Tx Voltage' (Off, 2V, 3V, 6V, 12V, 48V or appropriate).

Byte 104-107 : 'CH2 Rx Start Frequency' (in Hertz, Min = 10000, Max = 200000).

Byte 108-111 : 'CH2 Rx End Frequency' (in Hertz, Min = 10000, Max = 200000).

Byte 112-115 : 'CH2 Filter Length' (in microseconds, should be <= expected Rx Pulse Length).

Byte 116-119 : 'CH2 Sample Rate' (in Hertz, the frequency of the Rx Filter).

Byte 120,121 : 'CH2 Filter Alpha Shading' (1/100 %).

Bytes 122,123 : 'CH2 TVG Slope Correction' (%).

Byte 124 : 'CH2 MacBlocks' (Number of blocks in Match Filter).

Byte 125 : 'CH2 Phase Limit'.

Byte 126 : 'CH2 Phase Scalar'.

Byte 127 : 'CH2 Rssi Offset'.

Byte 128 : 'CH2 Rx Detect Thresholf (dB, 0 = Off).

Bytes 129,130 : 'CH2 Rx Detect Min Pulse Width (microseconds).

Bytes 131,132 : 'CH2 Rx Detect Correction Offset'.

Bytes 133-136 : 'CH2 Tx Start Frequency' (in Hertz, Min = 10000, Max = 200000).

Byte 137-140 : 'CH2 Tx End Frequency' (in Hertz, Min = 10000, Max = 200000).

Byte 141-144 : 'CH2 Tx Pulse Length' (in microseconds, Min = 10, Max = 250000).

Byte 145 : 'CH2 Tx Duty Cycle' (1..15).

Byte 146 : 'CH2 Tx Voltage' (Off, 2V, 3V, 6V, 12V, 48V or appropriate).

Byte 147 : 'Half Duplex' ('0' = No*, '1' = Yes). * = *Normal RS-232 Setting.*

Byte 148 : 'Has Motor' ('0' = No Motor, '1' = Has Motor*). * = *Normal setting.*

Byte 149 : 'Motor Amp Gain, Pan Motor' (0-100%).

Byte 150 : 'Motor Amp Gain, Tilt Motor' (0-100%).

Bytes 151,152 : 'Motor Pan Constant'

Bytes 153,154	:	'Max Speed Pan'
Bytes 155,156	:	'MechLL Pan'
Bytes 157,158	:	'MechRL Pan'
Bytes 159,160	:	'Motor Tilt Constant'
Bytes 161,162	:	'Max Speed Tilt'
Bytes 163,164	:	'MechLL Tilt'
Bytes 165,166	:	'MechRL Tilt'
Byte 167	:	'Main Port Comms Mode' ('0' = RS-232*, '1' = RS-485, '2' = ArcNet). * = *Normal.*
Byte 168	:	'Aux Port Comms Mode' ('0' = RS-232*, '1' = RS-485, '2' = ArcNet). * = *Normal.*
Byte 169	:	'Tx Drive' ('0' = Normal*, '1' = Both, '2' = Sub Bottom device). * = *Normal.*
Bytes 170,171	:	'Aux Comms Update Rate' (milliseconds)..
Byte 172	:	'Dual Filter' ('0' = No*, '1' = Yes). * = *Normal RS-232 Setting.*
Byte 173	:	'CH1 ADC8100 Gain' (0..6, SeaKing V6 Only).
Byte 174	:	'CH2 ADC8100 Gain' (0..6, SeaKing V6 Only).

9) 'mtFpgaCalibrationData' Reply Message – Digital DST Sonar Head (ID = '63')

This Reply is a digital device specific record that informs of the calibration data stored within the subsea device's FPGA module. This reply is sent as part of a 3 x message response to the 'mtSendBBUser' command issued to a digital DST Sonar head.
The 'mtFpgaCalibrationData' message does not need to be processed by the user in any way. Its contents will be used by the factory during calibration checking and setup.
The calibration data can also be viewed through the 'Seanet Setup' Windows® Utility Program by clicking on 'Action' – 'Setup' for the Node and then selecting the 'Calibration' page.

'mFpgaCalibrationData' Packet format is;

Byte 1	:	Message Header = "@".
Bytes 2-5	:	Hex Length of binary packet from byte 6 onwards *(excludes LF).*
Bytes 6,7	:	Binary Word of above Hex Length.
Byte 8	:	Packet Source Identification (Tx Node number 0 - 240).
Byte 9	:	Packet Destination Identification (Rx Node number 255).
Byte 10	:	Byte Count of attached message that follows this byte.
Byte 11	:	Message Type = "63" (=3Fh) for mtFpgaCalibrationData.
Byte 12	:	Message Sequence Bitset.
Byte 13	:	Tx Node number *(copy of Byte 8).*
Byte 14	:	Calibrated (Boolean flag T/F?).
Byte 15	:	Number of ADC Channels.
Bytes 16->47	:	ADC Offset bytes [0..15 Channels] (16-bit values for 16 channels, used for factory calibration).
Byte 48-79	:	ADC Quality bytes [0..15 Channels] (16-bit values for 16 possible channels, used for factory calibration).
Byte 80	:	Message Terminator = LF *(0Ah).*

Calibrated (Byte 14)
This Boolean should be set to 'True' at all times.

Number of ADC Channels (Byte 15)
For a single channel Micron DST this will be 1. Other DST devices such as SeaKing and SeaPrince will have 2 or more channels.

ADC Offset Bytes (Bytes 16-47)
Factory calibration values.

ADC Quality Bytes (Bytes 48-79)
Factory calibration values.

10) 'mtFpgaVersionData' Reply Message – Digital DST Sonar Head (ID = '57')

This Reply is a digital device specific record that informs of the firmware version programmed into the subsea device's FPGA module. This reply is sent as part of a 3 x message response to the 'mtSendBBUser' command issued to a digital DST Sonar head.
The 'mtFpgaVersionData' message does not need to be processed by the user in any way. Instead, like the 'mtVersionData' reply, 'Checksum' and 'Device Revision' can be noted to determine if the FPGA firmware matches the surface controller software program. Refer to the 'mtVersionData' reply for further details.

'mFpgaVersionData' Packet format is;

Byte 1	:	Message Header = "@".
Bytes 2-5	:	Hex Length of binary packet from byte 6 onwards *(excludes LF)*.
Bytes 6,7	:	Binary Word of above Hex Length.
Byte 8	:	Packet Source Identification (Tx Node number 0 - 240).
Byte 9	:	Packet Destination Identification (Rx Node number 255).
Byte 10	:	Byte Count of attached message that follows this byte.
Byte 11	:	Message Type = "57" (=39h) for mtFpgaVersionData.
Byte 12	:	Message Sequence Bitset.
Byte 13	:	Tx Node number *(copy of Byte 8)*.
Byte 14	:	Device ID.
Bytes 15-18	:	Flash ID.
Bytes 19,20	:	Blocks.
Bytes 21,22	:	CheckSum.
Byte 23	:	Device Revision.
Byte 24	:	Message Terminator = LF *(0Ah)*.

Device ID (Byte 14)
Factory Use only!

Flash ID (Bytes 15-18)
Factory Use only!

Blocks (Bytes 19,20)
Factory Use only!

Checksum (Bytes 21,22)
This value should be used to record / verify the Fpga firmware revision programmed into the Sonar. For instance, a Sonar may have been introduced from another field system and be programmed with earlier (or later) firmware. This may affect functionality so it is always wise to ensure that User Programs that have been written around a specific Seanet software revision operate with Sonar heads programmed with that same revision.
N.B. The 'Seanet Setup' Utility Program can be used to check and re-program the desired revision of Fpga firmware into the Sonar.

Device Revision (Byte 23)
Factory Use only!

APPENDIX

Section 1. EXTRA COMMANDS

3a) 'mtHeadCommand' – Shortened 'GainB' Command Message (ID = '19')

Once the Sonar head is powered and has received a valid set of Parameters (i.e. full 'mtHeadCommand') from the surface controller, it can start to be triggered with 'mtSendData' requests. During this time it may be necessary to make instant changes to the Receiver Gain parameters during the triggering period. For this matter, it is possible to send a shortened 'mtHeadCommand' to the Sonar head which includes only the Gain Parameters. Byte 14 of the 'mtHeadCommand' must be set to Type "30" to indicate to the Sonar that this is a shortened 'mtHeadCommand' with only the 'GainB' parameter record attached.

'mtHeadCommand' Command (ID = '19') Packet format is;

Byte 1	:	Message Header = "@".
Bytes 2-5	:	Hex Length of binary packet from byte 6 onwards *(excludes LF)*.
Bytes 6,7	:	Binary Word of above Hex Length.
Byte 8	:	Packet Source Identification (Tx Node number 0 - 240).
Byte 9	:	Packet Destination Identification (Rx Node number 255).
Byte 10	:	Byte Count of attached message that follows this byte.
Byte 11	:	Message Type = "19" for mtHeadCommand.
Byte 12	:	Message Sequence Bitset *(see below)*.
Byte 13	:	Tx Node number *(copy of Byte 8)*.
Byte 14	:	mtHeadCommand Type "30" = Reduced Command with Gain Parameters only.
Bytes 15-30	:	'GainB' Parameter Info.

Sonar 'GainB Parameter Info' block

For an Imaging Sonar, the 'GainB Parameter Info' block is 16 bytes in length…

Bytes 15,16	:	'ADSpanB' Ch1/Ch2' *(1/255 units)*.
Byte 17,18	:	'ADLowB' Ch1/Ch2 *(1/255 units)*.
Bytes 19,20	:	'IGainB' Ch1/Ch2. Initial Gain for both channels *(1/210 units)*.
Bytes 21,22	:	Spare Bytes *(was 'ADC Setpoint' Ch1/Ch2)*.
Bytes 23-26	:	'SlopeB' Ch1/Ch2 *(1/255 units). (N/A for DST)*
Bytes 27-30	:	'SlopeB Delay' Ch1/Ch2 Slope Holdoff *(msecs). (N/A for DST)*

An example of a shortened mtHeadCommand issued to the Sonar to perform a Gain Parameter update is as follows…

A. Serial Port sends shortened **mtHeadCommand** Command with **GainB** record (31 bytes)

16-byte 'GainB' Gain Parameters attached (Byte 14 = 1Eh)

40	**30**	**30**	**32**	**35**	**19**	**00**	**FF**	**02**	**47**	**13**	**80**	**02**	**1E**
Hdr '@'	Hex Length = 25 bytes				Bin Length = 25 bytes		Tx Nde 255	Rx Nde 02	No. Byte = 71	**mtH 'dC md**	Seq = End	Nde 02	Gain B
03	**97**	**03**	**40**	**06**	**01**	**00**	**00**	**00**	**50**	**51**	**09**	**08**	**54**
AD Sp'n Ch1	AD Sp'n Ch2	AD Low Ch1	AD Low Ch2	Iga-in Ch1	Iga-in, Ch2	Spa-re	Spa-re	Slope, Ch1 = 90 # *Ignored*		Slope, Ch2 = 125 # *Ignored*		Slope Delay, Ch1 # *Ignored*	
54	**00**	**0A**											
Slope Delay, Ch2 # *Ignored*		LF											

3b) 'mtHeadCommand' – Shortened 'Scan Reversal' Cmd Message (ID = '19')

Once the Sonar head is powered and has received a valid set of Parameters (i.e. full 'mtHeadCommand') from the surface controller, it can start to be triggered with 'mtSendData' requests. During the scanning it is possible to instantly change the direction of scanning by issuing a shortened mtHeadCommand without any Parameters attached. Byte 14 of the 'mtHeadCommand' must be set to Type "15" to indicate to the Sonar that this is a shortened 'mtHeadCommand' with no Parameters attached – this will result in an instant reversal of the scan direction.

'mtHeadCommand' Command (ID = '19') Packet format is;

Byte 1	:	Message Header = "@".
Bytes 2-5	:	Hex Length of binary packet from byte 6 onwards *(excludes LF)*.
Bytes 6,7	:	Binary Word of above Hex Length.
Byte 8	:	Packet Source Identification (Tx Node number 0 - 240).
Byte 9	:	Packet Destination Identification (Rx Node number 255).
Byte 10	:	Byte Count of attached message that follows this byte.
Byte 11	:	Message Type = "19" for mtHeadCommand.
Byte 12	:	Message Sequence Bitset *(see below)*.
Byte 13	:	Tx Node number *(copy of Byte 8)*.
Byte 14	:	mtHeadCommand Type "15" = Reduced 'Scan Reversal' Command, with no Parameters attached.

An example of a shortened mtHeadCommand issued to the Sonar to perform a Scan Direction Reversal is as follows…

B. Serial Port sends shortened **mtHeadCommand** Command (15 bytes)

15-byte 'Scan Reversal' command with no Parameters attached (Byte 14 = 0Fh)

40	**30**	**30**	**30**	**39**	**09**	**00**	**FF**	**02**	**47**	**13**	**80**	**02**	**0F**
Hdr '@'	Hex Length = 9 bytes				Bin Length = 9 bytes		Tx Nde 255	Rx Nde 02	No. Byte = 71	**mtH 'dC md**	Seq = End	Nde 02	Scan Rev erse

0A
LF

Section 2. MULTI-PACKET EXAMPLES

1. Example of Multi-Packet 'mtHeadData' Reply, 4-bit ADC

A. Serial Port sends **mtSendData** command to Sonar (18 bytes)

40	**30**	**30**	**30**	**43**	**0C**	**00**	**FF**	**02**	**07**	**19**	**80**	**02**
Hdr '@'	Hex Length = 12 bytes				Bin Length = 12 bytes		Tx Nde 255	Rx Nde 02	No. Byte = 7	**mtSend D'ta**	Seq = End	Nde 02
CA	**64**	**B0**	**03**	**0A**								
Current Time = 61891786 msec = 17:11:31.79				LF								

B. Sonar replies with **2 x 'mtHeadData' Scanline Data Reply** (296 x 4-bit Data Bins) = Data

ScanlineA Packet

40	**30**	**30**	**36**	**32**	**62**	**00**	**02**	**FF**	**5D**	**02**	**00**	**02**	**B3**
Hdr '@'	Hex Length = 98 bytes				Bin Length = 98 bytes		Tx Nde 02	Rx Nde 255	No. Byte = 93	**mtHead D'ta**	Start of Seq.	Nde 02	->
00	**02**	**00**	**00**	**02**	**23**	**C8**	**00**	**9A**	**99**	**99**	**02**	**28**	**96**
Count = 179	= Sonhead	Stat-us	Sw-eep	HdCtrl = 8962		Range = 20m		TxN = 43620762				Gain = 16%	->
00	**2D**	**28**	**00**	**00**	**00**	**00**	**00**	**00**	**F0**	**18**	**10**	**D0**	**0E**
Slo-pe= 150	AD Sp'n	AD Low	Heading Offset = 0		AD Interval = 0		L.Limit = 0		R.Limit = 6384 (1/16 Grad)		Ste-ps = 16	Bearing = 3792 (1/16 Grad)	
94	**00**	**FD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**
Dbytes = 148		Bins 1,2	Bins 3,4	Bins 5,6	Bins 7,8	Bins 9,10	Bins 11, 12	Bins 13, 14	Bins 15, 16	Bins 17, 18	Bins 19, 20	Bins 21, 22	Bins 23, 24
DD	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**
Bins 25, 26	Bins 27, 28	Bins 29, 30	Bins 31, 32	Bins 33, 34	Bins 35, 36	Bins 37, 38	Bins 39, 40	Bins 41, 42	Bins 43, 44	Bins 45, 46	Bins 47, 48	Bins 49, 50	Bins 51, 52
DD	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**
Bins 53, 54	Bins 55, 56	Bins 57, 58	Bins 59, 60	Bins 61, 62	Bins 63, 64	Bins 65, 66	Bins 67, 68	Bins 69, 70	Bins 71, 72	Bins 73, 74	Bins 75, 76	Bins 77, 78	Bins 79, 80
DD	**DD**	**DD**	**DD**	**DE**	**DD**	**DD**	**ED**	**DD**	**DD**	**DE**	**DD**	**ED**	**DE**
Bins 81, 82	Bins 83, 84	Bins 85, 86	Bins 87, 88	Bins 89, 90	Bins 91, 92	Bins 93, 94	Bins 95, 96	Bins 97, 98	Bins 99, 100	Bins 101, 102	Bins 103, 104	Bins 105, 106	Bins 107, 108
ED	**DD**	**DE**	**DD**	**ED**	**0A**								
Bins 109, 110	Bins 111, 112	Bins 113, 114	Bins 115, 116	Bins 117, 118	LF								

ScanlineB Packet

40	**30**	**30**	**36**	**31**	**61**	**00**	**02**	**FF**	**5C**	**02**	**81**	**02**	**DE**
Hdr '@'	Hex Length = 97 bytes				Bin Length = 97 bytes		Tx Nde 02	Rx Nde 255	No. Byte = 92	**mtHead D'ta**	End of Seq.	Nde 02	Bins 119, 120
DE	**DD**	**DE**	**DD**	**DD**	**ED**	**DD**	**ED**	**DE**	**DD**	**ED**	**DD**	**DD**	**ED**
Bins 121, 122	Bins 123, 124	Bins 125, 126	Bins 127, 128	Bins 129, 130	Bins 131, 132	Bins 133, 134	Bins 135, 136	Bins 137, 138	Bins 139, 140	Bins 141, 142	Bins 143, 144	Bins 145, 146	Bins 147, 148
DD	**DD**	**DD**	**DE**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**
Bins 149, 150	Bins 151, 152	Bins 153, 154	Bins 155, 156	Bins 157, 158	Bins 159, 160	Bins 161, 162	Bins 163, 164	Bins 165, 166	Bins 167, 168	Bins 169, 170	Bins 171, 172	Bins 173, 174	Bins 175, 176
DD	**DD**	**FD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**
Bins 177, 178	Bins 179, 180	Bins 181, 182	Bins 183, 184	Bins 185, 186	Bins 187, 188	Bins 189, 190	Bins 191, 192	Bins 193, 194	Bins 195, 196	Bins 197, 198	Bins 199, 200	Bins 201, 202	Bins 203, 204
DD	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**
Bins 205, 206	Bins 207, 208	Bins 209, 210	Bins 211, 212	Bins 213, 214	Bins 215, 216	Bins 217, 218	Bins 219, 220	Bins 221, 222	Bins 223, 224	Bins 225, 226	Bins 227, 228	Bins 229, 230	Bins 231, 232
DD	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**	**DD**
Bins 233, 234	Bins 235, 236	Bins 237, 238	Bins 239, 240	Bins 241, 242	Bins 243, 244	Bins 245, 246	Bins 247, 248	Bins 249, 250	Bins 251, 252	Bins 253, 254	Bins 255, 256	Bins 257, 258	Bins 259, 260
DD	**DD**	**DD**	**DD**	**DE**	**DD**	**DD**	**ED**	**DD**	**DD**	**DE**	**DD**	**ED**	**DE**
Bins 261, 262	Bins 263, 264	Bins 265, 266	Bins 267, 268	Bins 269, 270	Bins 271, 272	Bins 273, 274	Bins 275, 276	Bins 277, 278	Bins 279, 280	Bins 281, 282	Bins 283, 284	Bins 285, 286	Bins 287, 288
ED	**DD**	**DE**	**DD**	**0A**									
Bins 289, 290	Bins 291, 292	Bins 293, 294	Bins 295, 296	LF									

2. Example of Multi-Packet 'mtHeadData' Reply, 8-bit ADC (<u>Note</u>: 'Status' byte = 10h)

A. Serial Port sends **mtSendData** command to Sonar (18 bytes)

40	**30**	**30**	**30**	**43**	**0C**	**00**	**FF**	**02**	**07**	**19**	**80**	**02**
Hdr '@'	Hex Length = 12 bytes				Bin Length = 12 bytes		Tx Nde 255	Rx Nde 02	No. Byte = 7	**mtSend D'ta**	Seq = End	Nde 02
CA	**64**	**B0**	**03**	**0A**								
Current Time = 61891786 msec = 17:11:31.79				LF								

B. Sonar replies with **2 x** '**mtHeadData**' **Scanline Data Reply** (210 x 8-bit Data Bins) = Data

<overpage>

ScanlineA Packet

40	**30**	**30**	**38**	**31**	**81**	**00**	**02**	**FF**	**7C**	**02**	**00**	**02**	**F1**
Hdr '@'	Hex Length = 129 bytes				Bin Length = 129 bytes		Tx Nde 02	Rx Nde 255	No. Byte 124	**mtHead D'ta**	Start of Seq.	Nde 02	->
00	**02**	**10**	**00**	**03**	**23**	**C8**	**00**	**9A**	**99**	**99**	**02**	**28**	**96**
Count = 241	= Sonhead	Status = 10	Sweep	HdCtrl = 8962		Range = 20m		TxN = 43620762				Gain = 16%	->
00	**2D**	**28**	**00**	**00**	**00**	**00**	**00**	**00**	**F0**	**18**	**10**	**F0**	**00**
Slope= 150	AD Sp'n	AD Low	Heading Offset = 0		AD Interval = 0		L.Limit = 0		R.Limit = 6384 (1/16 Grad)		Steps = 16	Bearing = 240 (1/16 Grad)	
D2	**00**	**5B**	**51**	**50**	**50**	**4F**	**50**	**50**	**4F**	**50**	**50**	**AD**	**5F**
Dbytes = 210		Bin 1	Bin 2	Bin 3	Bin 4	Bin 5	Bin 6	Bin 7	Bin 8	Bin 9	Bin 10	Bin 11	Bin 12
5B	**50**	**50**	**51**	**4F**	**4F**	**50**	**4F**	**84**	**80**	**54**	**51**	**51**	**4F**
Bin 13	Bin 14	Bin 15	Bin 16	Bin 17	Bin 18	Bin 19	Bin 20	Bin 21	Bin 22	Bin 23	Bin 24	Bin 25	Bin 26
50	**51**	**4F**	**50**	**50**	**56**	**51**	**4F**	**51**	**51**	**50**	**51**	**50**	**50**
Bin 27	Bin 28	Bin 29	Bin 30	Bin 31	Bin 32	Bin 33	Bin 34	Bin 35	Bin 36	Bin 37	Bin 38	Bin 39	Bin 40
50	**50**	**50**	**51**	**50**	**50**	**50**	**50**	**51**	**50**	**4F**	**50**	**51**	**50**
Bin 41	Bin 42	Bin 43	Bin 44	Bin 45	Bin 46	Bin 47	Bin 48	Bin 49	Bin 50	Bin 51	Bin 52	Bin 53	Bin 54
4F	**4F**	**51**	**4F**	**4F**	**4F**	**50**	**50**	**50**	**50**	**50**	**50**	**50**	**4F**
Bin 55	Bin 56	Bin 57	Bin 58	Bin 59	Bin 60	Bin 61	Bin 62	Bin 63	Bin 64	Bin 65	Bin 66	Bin 67	Bin 68
51	**4F**	**50**	**50**	**4F**	**50**	**4F**	**50**	**4F**	**50**	**50**	**51**	**4F**	**50**
Bin 69	Bin 70	Bin 71	Bin 72	Bin 73	Bin 74	Bin 75	Bin 76	Bin 77	Bin 78	Bin 79	Bin 80	Bin 81	Bin 82
4F	**50**	**4F**	**4F**	**50**	**51**	**51**	**50**	**0A**					
Bin 83	Bin 84	Bin 85	Bin 86	Bin 87	Bin 88	Bin 89	Bin 90	LF					

ScanlineB Packet

40	**30**	**30**	**38**	**30**	**80**	**00**	**02**	**FF**	**78**	**02**	**81**	**02**	**50**
Hdr '@'	Hex Length = 128 bytes				Bin Length = 128 bytes		Tx Nde 02	Rx Nde 255	No. Byte 120	**mtHead D'ta**	End of Seq.	Nde 02	Bin 91
51	**51**	**4F**	**4F**	**51**	**4F**	**51**	**51**	**50**	**50**	**50**	**4F**	**50**	**50**
Bin 92	Bin 93	Bin 94	Bin 95	Bin 96	Bin 97	Bin 98	Bin 99	Bin 100	Bin 101	Bin 102	Bin 103	Bin 104	Bin 105
50	**50**	**50**	**4F**	**50**	**51**	**4F**	**4F**	**51**	**50**	**51**	**51**	**50**	**4F**
Bin 106	Bin 107	Bin 108	Bin 109	Bin 110	Bin 111	Bin 112	Bin 113	Bin 114	Bin 115	Bin 116	Bin 117	Bin 118	Bin 119
50	**4F**	**4F**	**50**	**50**	**52**	**50**	**50**	**50**	**50**	**52**	**50**	**50**	**50**

Bin 120	Bin 121	Bin 122	Bin 123	Bin 124	Bin 125	Bin 126	Bin 127	Bin 128	Bin 129	Bin 130	Bin 131	Bin 132	Bin 133
4F	**51**	**A8**	**6B**	**66**	**66**	**51**	**58**	**51**	**5C**	**54**	**84**	**C4**	**75**
Bin 134	Bin 135	Bin 136	Bin 137	Bin 138	Bin 139	Bin 140	Bin 141	Bin 142	Bin 143	Bin 144	Bin 145	Bin 146	Bin 147
65	**5A**	**57**	**58**	**6B**	**54**	**54**	**52**	**6A**	**8B**	**53**	**51**	**51**	**51**
Bins 148	Bin 149	Bin 150	Bin 151	Bin 152	Bin 153	Bin 154	Bin 155	Bin 156	Bin 157	Bin 158	Bin 159	Bin 160	Bin 161
52	**52**	**5B**	**59**	**51**	**56**	**50**	**4F**	**51**	**50**	**51**	**52**	**50**	**51**
Bin 162	Bin 163	Bin 164	Bin 165	Bin 166	Bin 167	Bin 168	Bin 169	Bin 170	Bin 171	Bin 172	Bin 173	Bin 174	Bin 175
50	**51**	**51**	**50**	**50**	**51**	**50**	**50**	**50**	**50**	**50**	**52**	**50**	**52**
Bin 176	Bin 177	Bin 178	Bin 179	Bin 180	Bin 181	Bin 182	Bin 183	Bin 184	Bin 185	Bin 186	Bin 187	Bin 188	Bin 189
51	**51**	**50**	**51**	**51**	**51**	**50**	**50**	**51**	**51**	**50**	**52**	**50**	**50**
Bin 190	Bin 191	Bin 192	Bin 193	Bin 194	Bin 195	Bin 196	Bin 197	Bin 198	Bin 199	Bin 200	Bin 201	Bin 202	Bin 203
53	**51**	**50**	**50**	**51**	**50**	**50**	**0A**						
Bin 204	Bin 205	Bin 206	Bin 207	Bin 208	Bin 209	Bin 210	LF						