

Remote Helm Manual



Contents

1. Introduction to Backseat Driver Setup.....	2
1.1 AMD Motherboard Setup.....	2
1.1.1 Serial Port Setup.....	3
1.1.2 Ethernet Setup.....	5
1.2 Intel Atom Motherboard Setup	7
2. Remote Helm Commands.....	8
2.1 Introduction to Remote Helm Commands.....	8
2.2 Remote Helm Control Models.....	8
2.2.1 Manual and Servo Mode.....	8
2.2.2 Normal Mode.....	9
2.2.3 Park Mode.....	10
2.2.4 Mission Mode.....	10
2.2.5 Command operation during a safety violation.....	11
2.3 Simulating remote helm commands using SubTester.....	11
2.4 Definition of all remote helm commands.....	14

APPENDIX A – BACKSEAT DRIVER COM PORTS AND THE MAIN AMD CPU SERIAL PORTS

1 AND 4.....	35
--------------	----

APPENDIX B – SCHEMATIC COM PORTS 1 AND 2 INSERTED INTO THE BACKPLANE OF THE

AMD CPU.....	36
--------------	----

APPENDIX C – BACKSEAT AND MAIN INTEL ATOM MOTHERBOARD SPECIFICATIONS INTO

THE BACKPLANE OF THE AMD CPU.....	37
-----------------------------------	----

Note: This document assumes that the user has read the Operating Guide and the Mission Planning manuals.

1. Introduction to Backseat Driver Setup

To communicate with the Iver2 (main CPU) through the second or backseat driver CPU, a serial connection must first be created between the backseat and main CPU. The main CPU receives backseat driver commands through a serial port, which permits the user to control the vehicle. After the serial connection between the two CPU's is completed, the serial port used to communicate with the main CPU must be added to the serial input list within UVC (Underwater Vehicle Console). To add a serial port in UVC, refer to the Operating Guide *Appendix B - How to Use the UVC*. In UVC all input serial ports are parsed by a universal parser, all defined NMEA and remote helm commands can be parsed on any port.

1.1 AMD Motherboard Setup

Figure 1.1 illustrates the physical setup of the backseat driver CPU. The first generation backseat driver was an AMD motherboard, newer models have an Intel Atom based CPU covered in a later section of this manual. Dependent upon the Iver2's configuration, the backseat driver CPU has two available serial ports. If the Iver2 has side scan sonar or another additional sensor, then only one serial port can be used to interact with the main CPU. For further detail and schematics of the backseat and main CPU serial ports, refer to Appendix A and B.



Figure 1.1 - Rear of the Iver internal rack with the user and backseat driver CPU.

1.1.1 Serial port setup

Figure 1.2 shows the two serial ports from the backseat driver plugged in the Iver2's backplane. The two serial ports are wired down the backplane to COMA and COMB, shown in Figure 1.3. As mentioned earlier, dependent upon the Iver2's configuration, the backseat driver CPU might have only one available serial port.

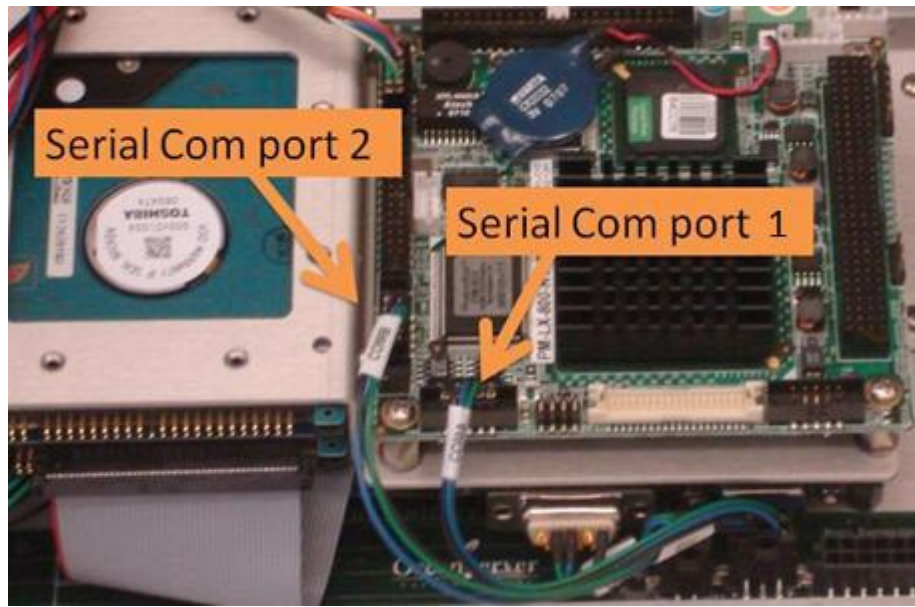


Figure 1.2 – Connection of the two available serial ports to the backplane

Figure 1.3 illustrates the serial ports coming from the backseat driver down the backplane. Below is a description of Figure 1.3.

- **W** : Com port 1 of the main CPU – alternate use is for side scan sonar
- **X** : Com port 2 or COMB of the backseat driver
- **Y** : Com port 1 or COMA of the backseat driver
- **Z** : Com port 4 main CPU – alternate use is for the YSI sensor

In Figure 1.3 Com1 of the main CPU is connected to Com1 of the backseat driver. The cable connected between the backseat serial port and the main CPU must have the transmit and receive terminals flipped, as shown in Figure 1.4. The serial ports must both be configured at the same baud rate. If serial port four is open on the backplane then a six pin serial connection must be made, refer to appendix A for the pin out.

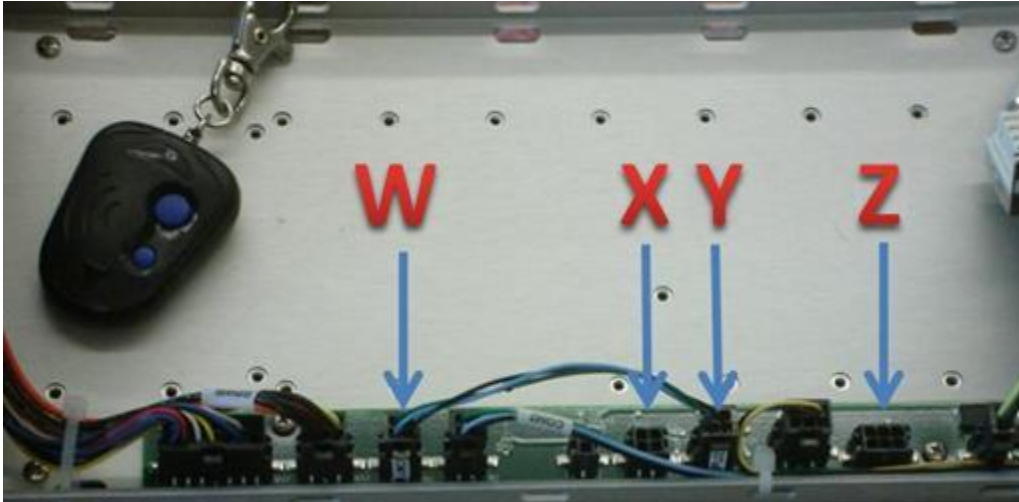


Figure 1.3 – Com port connection between COMA to serial port 1 on the backplane

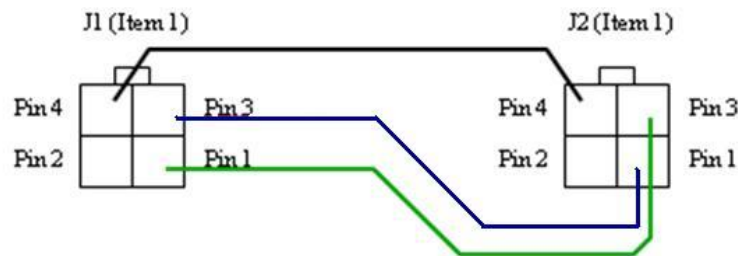


Figure 1.4 – Two serial ports with the transmit and receive terminals flipped

1.1.2 Ethernet setup

The Ethernet connection allows the user to transfer data and remotely connect to the backseat driver CPU. Figure 1.5 shows the Ethernet connection between the backseat and main CPU. A cross over cable for the Ethernet connection is needed from the main CPU to the second CPU. If a Microsoft Windows operating system has been installed on the backseat driver, follow the steps below in order to setup a connection between CPU's:

1. Front CPU has a static IP addresses of: 192.168.1.100
2. Back CPU has a static IP addresses of: 192.168.2.101
3. The backseat driver must belong to WORKGROUP.
4. Both must have the same subnet mask

Next, the user must set priority of the LAN and the wireless connection on the main CPU by following the steps below.

1. Open the Control Panel Network and Dial-up Connections applet. Right-click *Local Area Connection* and choose *Properties*
2. Select *Internet Protocol* and click *Properties*
3. Click *Advanced*
4. Make sure that the Automatic metric check box is selected
5. Open the Control Panel Network and Dial-up Connections applet. Right-click *Wireless Network Connection* and choose *Properties*
6. Repeat step 2 and 3
7. Clear the Automatic metric box and enter a 1 in the interface metric box

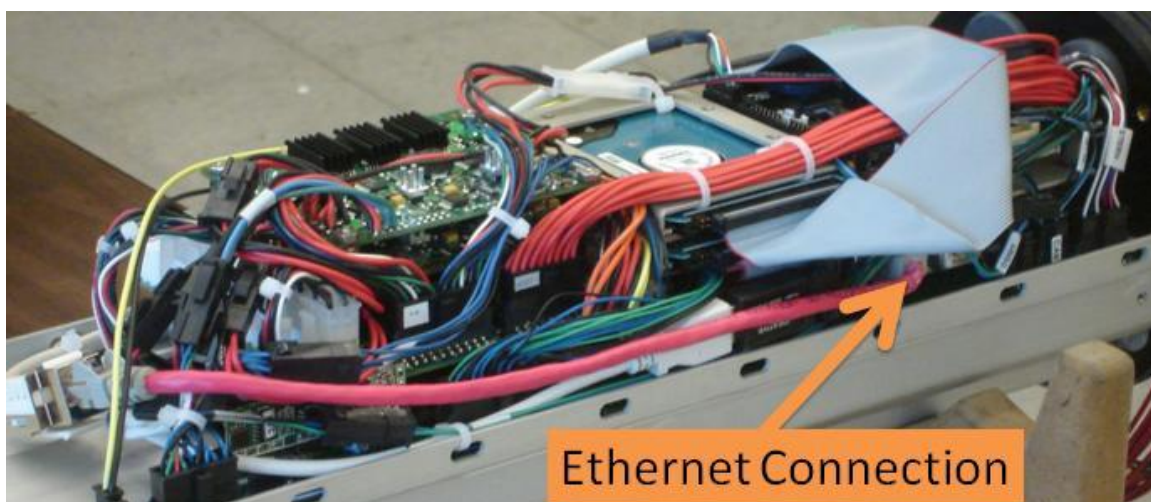


Figure 1.5 – Ethernet connection between the backseat and main CPU

1.2 Intel Atom Motherboard Setup

The specification of the Intel Atom motherboard is located in Appendix C. The very latest Iver2 AUV EP models include two Intel Atom CPUs (main and backseat/second). The main CPU serial port COM9 is connected to the second CPU serial port COM9. MS windows renames the serial ports, so COM9 on the main CPU is COM13, the same is true if windows is loaded on the backseat CPU. To remotely connect to the second CPU the static IP address of the main and second CPU are already setup. The main CPU has a static IP address of: 192.168.2.1, and the second CPU has a static IP address of: 192.168.2.2.

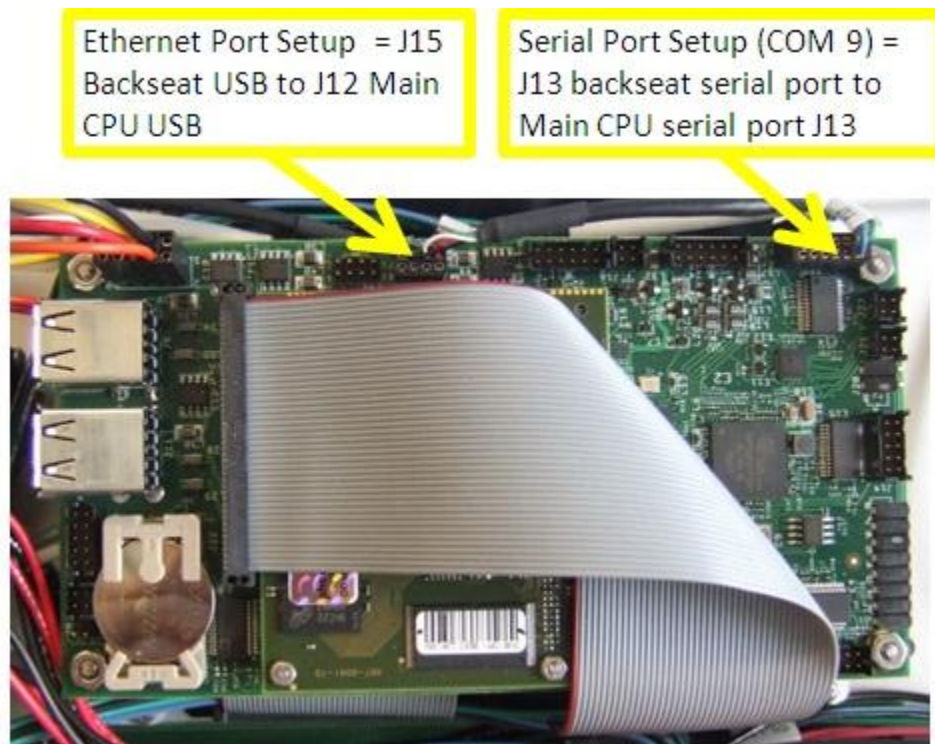


Figure 1.6 – Ethernet and serial connection between the backseat and main CPU

2. Remote Helm Commands

2.1 Introduction to remote helm commands

The Iver2 system through its UVC2 (Underwater Vehicle Control System) software allows the user to take control of the vehicle through a backseat driver CPU, by sending remote helms commands.

2.2 Remote helm control models

The remote helm model supports five types of commands, described below.

- 1) Servo – (\$OMS) Servo mode allows the backseat driver to provide heading, depth, max pitch angle and speed, which permit the vehicle to change its course and behavior based on the commands.
- 2) Primitive – (\$OMP) Through Primitive control the backseat driver runs the vehicle as a controller. Primitive control overrides the settings of the control fins and propulsion.
- 3) Normal – Normal operating mode when a mission is running and main CPU is in control of the vehicle.
- 4) Park – (\$OPK) Park mode occurs when the backseat driver instructs the vehicle of the park location time.
- 5) Mission – (\$OMW) Mission mode becomes active when the vehicle is issued a mission

2.2.1 Manual and Servo Mode

The operating model for controlling the vehicle revolves around commands (\$OMP

“Primitive” and \$OMS “Servo”) which are each given a life. The vehicle switches from AUTO mission (where UVC has control) to Primitive or Servo, but will revert back to AUTO after the life timer expires to zero. The commands have a maximum life of 120 seconds. The user is expected to send a \$OMS or \$OMP sentence at a regular rate under the timeout value to keep the vehicle under the backseat driver’s control. Before the backseat driver can take control, the UVC requires the user to start a mission. The \$OMP command can be executed when UVC is not running a mission. If the vehicle is parking and an OMP or OMS command is issued, the new command will be executed and when control is given back to the UVC, the vehicle will return to its park point. The park time will continue from where the vehicle was interrupted and not start over again.

2.2.2 Normal Mode

Normal mode is active when the main CPU is executing a preloaded mission. The following commands are used during normal operation.

- \$OJW – Jump to waypoint; allows the backseat driver to jump to any waypoint during a mission. The waypoint number must be contained in the range of waypoints in the present mission. OJW also overrides the \$OMP, \$OMS and \$OPK commands.
- \$OPOS – Set current GEO position, speed and/or conductivity and temperature. If the command is used for setting the vehicles current location and speed then it is only utilized while the vehicle is underwater. This command should only be used if the backseat driver has a better estimate of position and speed than the UVC. While the vehicle is on the surface it is receiving a GPS location, so the OPOS location will be overridden when there is a valid GPS fix. This command is equivalent to sending the GPS “\$GPRMC” sentence.
- \$OSD – Request sensor data permits the backseat driver to request data from the vehicle, such as the vehicle’s current state, Compass, GPS, State, Power and YSI data.
- \$OLOGL – Log label as column header. This command allows the backseat

driver to include user defined label column headers in the log file. This command must be sent before a mission is started.

- \$OLOGD – Log backseat user data. Each time data is written in the log file the last user data, corresponding to valid column headers, are written in the log. If there is no new data the last data received will be repeated.
- \$OMSTOP – Stop current mission
- \$OMLOAD – Load mission will be performed if the OMSTOP command is sent first
- \$OMSTART – Start mission will be executed if OMLOAD is sent
- \$ODVL – The DVL command, permits the backseat driver to maintain the correct position and speed by sending the X and Y velocity from a

2.2.3 Park Mode

Park mode is activated when a \$OPK command is sent from the backseat driver. The \$OPK command (*table 2.17*) permits the backseat driver to park the vehicle for a certain amount of time (minutes) given a GPS location and a travel speed in knots. If the \$OPK command is interrupted by another command then it will exit its park location and will not return to park after it has been interrupted.

2.2.4 Mission Mode

Mission mode allows the user to interrupt the mission, and provides the main CPU with a mission, by sending a series of waypoints. The \$OMW (*table 2.20*) command puts the Iver in missions mode. The user can clear the mission at anytime but send a clear signal or interrupting the mission with another backseat driver command that will force the vehicle into another state. At any point while the vehicle is running the mission, a new waypoint can be added to the mission. Once the vehicle has completed the mission it will continue where it left off during the original mission.

2.2.5 Command operation during a safety violation

In the event a safety rule occurs while the back seat driver is operating the vehicle, backseat driver control will be lost for thirty seconds (the \$OMP, \$OMS, \$OPK, \$OJW, \$OMW commands are blocked). After the thirty seconds, if the vehicle is stuck underwater the UVC will pulse the vehicles motor in reverse (assuming this safety rule is enabled) in order to bring the vehicle to the surface. After the vehicle has tried to pulse the motor in reverse, the \$OMP and \$OMS commands are unblocked (while underwater). The \$OPK, \$OMW and \$OJW commands are unblocked once the vehicle has reached the surface. The \$OMP and \$OMS commands will be unblocked earlier if the sub is stuck and the backseat driver has been programmed by the user to remove it from specific dangerous situation. After the backseat driver resumes control and the user wants to continue with the mission, it must first stop and restart the mission. The mission must be restarted, because once a safety rule has been triggered, the mission is stopped and the vehicle will park or execute its SRP(Safety Return Path) mission. During loading and starting of the SRP mission the \$OMP, \$OMS, \$OPK, \$OJW and \$OMW commands are blocked until the vehicle has started the mission.

2.3 Simulating remote helm commands using SubTester

SubTester is a windows program, written in Visual Basic 6, which generates and receives proper remote helm sentences (*Figure 2.1*). The SubTester program is provided in source code form to aid the user in developing a working backseat driver interface. The program is also used by OceanServer development to test and debug the backseat driver interface. To observe the backseat driver commands the UVC has a debug facility, in which the user can view all incoming serial data (*Figure 2.2*).

If the user thinks there is a bug in the backseat driver they need to reproduce it using the SUTESTER code and only then report it to support@ocean-server.com.

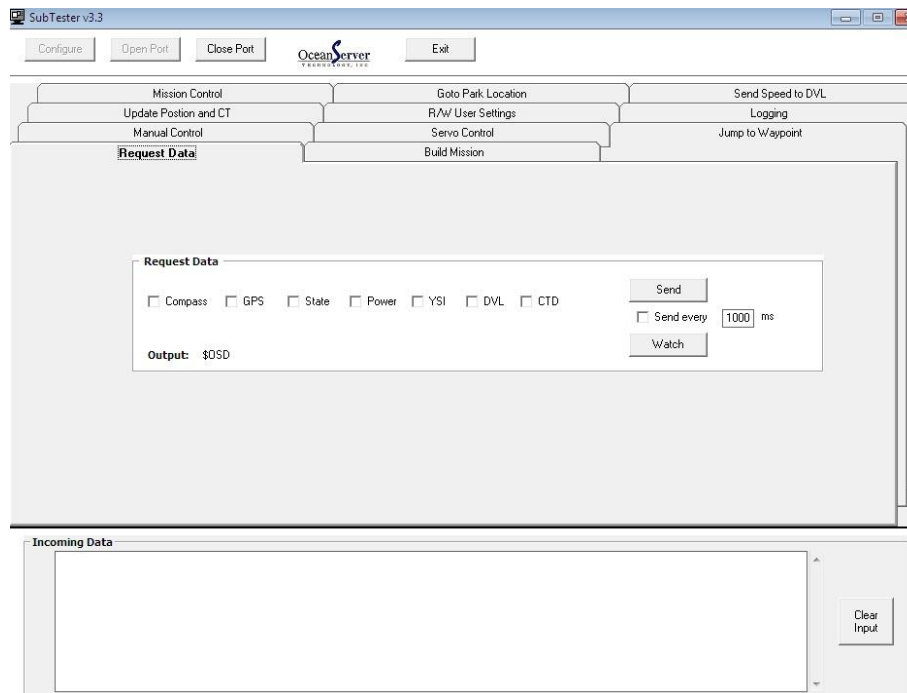


Figure 2.1 - SubTester screen showing how to test UVCs remote helm commands.



Figure 2.2 - UVC control program – Debug Button – Activate NMEA debugger

Figure 2.3 shows the debug window running on a normal system. The user can see the system parsing all of the NMEA sentences that it's receiving. The area to the right of the

screen shows the state of the control surfaces and motor as well as the present controller state, which can be auto, primitive, servo or park. If the user sent a command, the debug window will display the command and the vehicle state will change depending upon the command. Some commands are given a timeout life, and the user can see the life of the command count down next to *Timeout*.

Figure 2.4 illustrates the result of a \$OMP command being issued and the result of the command on the vehicle state. The FINS were commanded to go to 00, but the servo hardware limits them to a minimum value of 5.

All sentences are required to have the simple checksum byte used in NMEA style sentences. The user keeps an XOR sum of all characters from the \$ to the * exclusive and outputs this in Hexadecimal. If this isn't present UVC will reject sentences as defective.

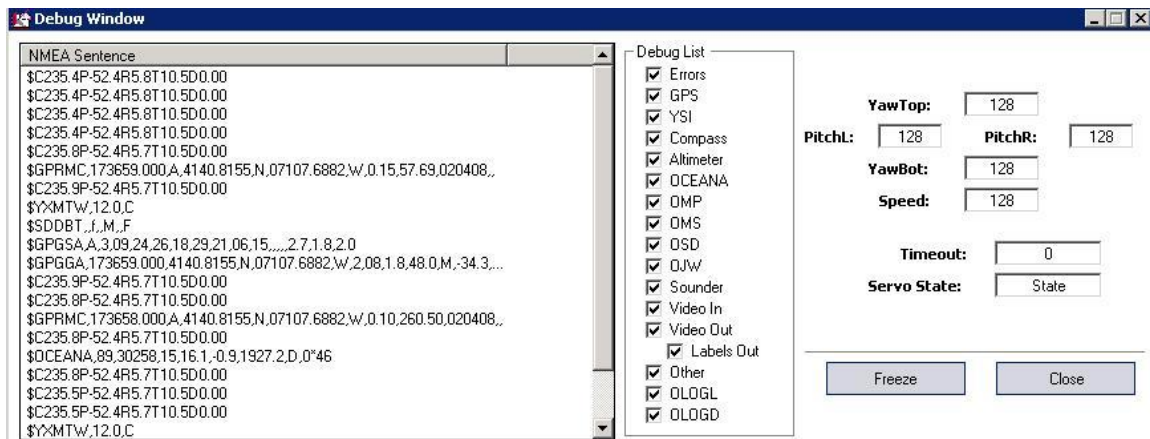


Figure 2.3 - Debug window AUTO mode (Normal operation)

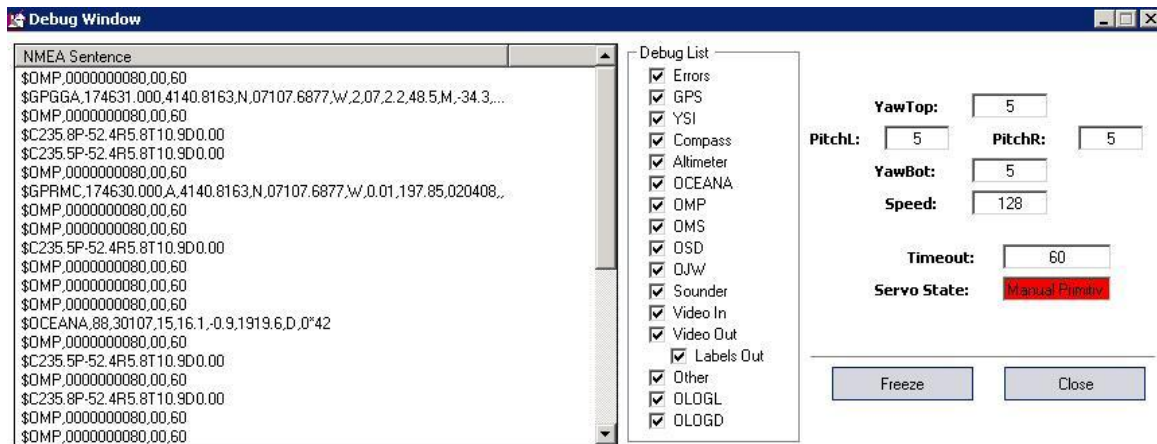


Figure 2.4 - Debug window PRIMITIVE mode (After \$OMP issued and active)

2.4 Definition of all Remote Helm commands

The control and data messages will use a packet structure derived for NMEA 0813. The generic message structure is shown in Table 2.1. The tables below define the supported commands. A brief description of the backseat driver commands are listed below:

- \$OMP – (table 2.2) Primitive mode.
- \$OMS – (table 2.3) Servo mode.
- \$OJW – (table 2.5) Jump to waypoint.
- \$OPOS – (table 2.10) Set current GEO position, speed and/or conductivity and temperature.
- \$OSD – (table 2.4) Request vehicle data.
- \$OLOGL – (table 2.11) Insert column header.
- \$OLOGD – (table 2.12) Log user data.
- \$OMSTOP – (table 2.13) Stop current mission.
- \$OMLOAD – (table 2.14) Load mission.
- \$OMSTART – (table 2.15) Start mission.
- \$ORWSET – (table 2.16) Read and Write current vehicle user settings
- \$OPK – (table 2.17) Park mode.
- \$ODVL – (table 2.19) Send Speed over ground.

- \$OMW– (table 2.20) Mission mode.

The generic format of the backseat driver commands are shown in Table 2.1

Format:	\$Oxx,p1,p2,p3..... *cc
Where:	
\$Oxx	Is the command name, “xx” ASCII alpha characters
“ , ”	Comma delimiters
“<null>”	No characters
p1, p2,p3,.....	Message parameters, there may be one or more values, the number of characters is variable without leading zero padding except just to the left of a decimal point unless indicated otherwise: The negative sign (-) only appears in the negative values, no corresponding (+) appears for positive values.
*cc	
\r\n	(CR) Carriage Return and (LF) Line feed

Table 2.1 - Generic Message Format

Format:	\$OMP,<FYT><FYB><FPL><FPR><MS>, <RP>,<TO><*cc>	
Direct Fin and main thruster control command		
Direction: Host (backseat) to UVC		
Where:	Description	Values
\$OMP	Message type: Primitive Control	“\$OMP”
<FYT>	Fin Control: Yaw Top Fin, 2 character, Hexadecimal, unsigned	Range: 0 to 255 (HEX:00 to FF) see figure 3.1.2-1
<FYB>	Fin Control: Yaw Bottom Fin, 2 character, Hexadecimal, unsigned	Range: 0 to 255 (HEX:00 to FF) see figure 3.1.2-1
<FPL>	Fin Control: Pitch Left Fin, 2 character, Hexadecimal, unsigned	Range: 0 to 255 (HEX:00 to FF) see figure 3.1.2-1
<FPR>	Fin Control: Pitch Right Fin, 2 character, Hexadecimal, unsigned	Range: 0 to 255 (HEX:00 to FF) see figure 3.1.2-1
<MS>	Motor Speed, 2 character, Hexadecimal, Signed	Range: 0 to 255, 255 full forward, 0= full reverse, 128 = no thrust
<RP>	Reserved Parameter	00
<TO>	Time Out: Command life if not overwritten by next \$OMP or \$OMS command	0 to 120 seconds
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.2 - Message: Primitive Control

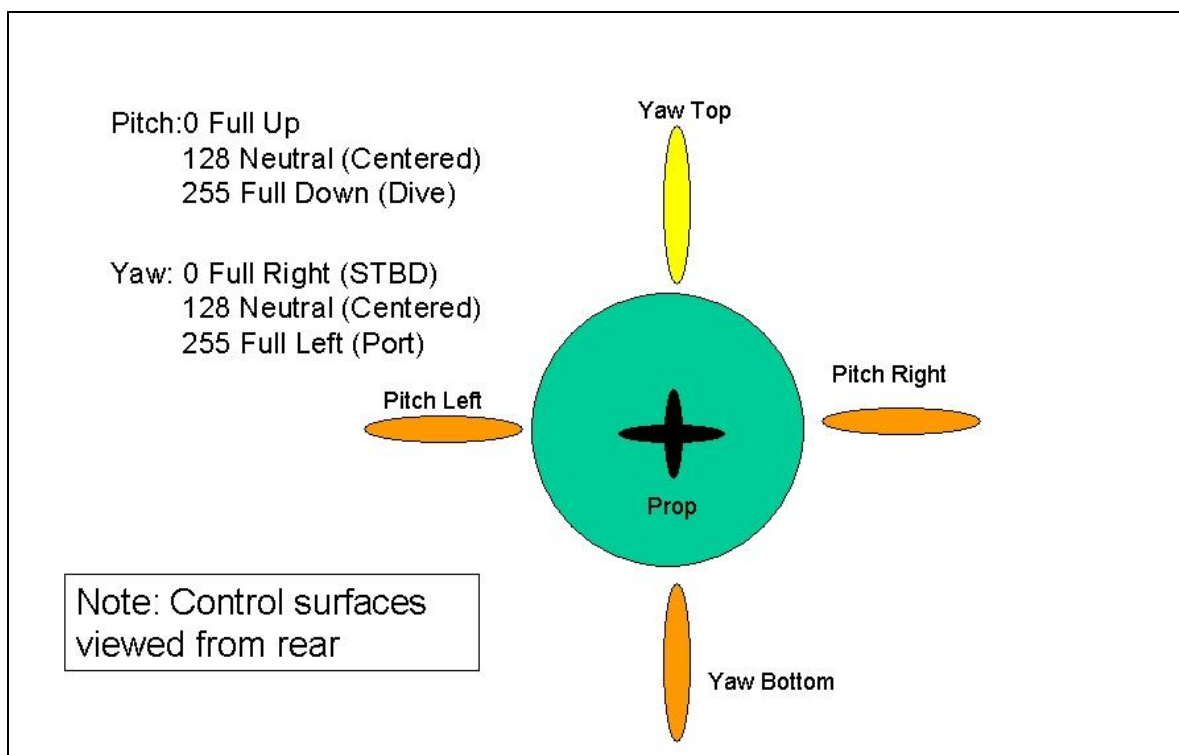


Figure 2.1 - Fin Control Reference Diagram

Format:	\$OMS,<Heading>, <Depth>,<Max Angle >,<Speed>,<TO> <*cc>	
Autopilot set points for servo control		
Direction: Host (backseat) to UVC		
Where:	Description	Values
\$OMS	Message type: Servo Control	“\$OMS”
<Heading>	Heading set point in degrees relative to magnetic north	Range: 0.0 to 360
<Depth>	Depth(ft) set point of the AUV referenced to depth gauge port at one standard atmosphere	Range: 0.0 (surface) to 999.9 in feet
<Max Angle>	Maximum pitch angle of the vehicle for changing depth (up or down) in degrees <i>Note: The max angle will override the max pitch angle of the vehicle, which is found in the servo data section</i>	Range: 0 to 45
<Speed>	Forward speed set point in water column in knots	Range: 0.0 to 9.9
<TO>	Time Out: Command life is	0 to 120 seconds

	overwritten by next \$OMS	
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.3 - Message: Servo Control

Format:	\$OSD,<data1>,<data2>, <data3>,<data4>, <data5>,<*cc>	
Request sensor data from vehicle		
Direction: Host (backseat) to UVC		
Where:	Description	Values
\$OSD	Message type: Send Sensor Data	“\$OSD
<GPS>	Send GPS Data (see Table 2.6)	“G” or “<null>”
<Compass>	Send Compass Data (see Table 2.7)	“C” or “<null>”
<State>	Send State Data (see Table 2.8)	“S” or “<null>”
<Power>	Send Power Data (see Table 2.9)	“P” or “<null>”
<YSI>	Send the last YSI string to the user	“Y” or “<null>”
<DVL>	Send DVL Data (see Table 2.21)	“D” or “<null>”
<CTD>	Send CTD Data (see Table 2.22)	“T” or “<null>”
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.4 Message: Send Sensor Data

Format:	\$OJW,<WPN><*cc>	
Force the value for the next waypoint to this value		
Direction: Host (backseat) to UVC		
Where:	Description	Values
\$OJW	Message type: Set next way point	“\$OJW”
<WPN>	The next waypoint value is this value	0 to 999
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.5 - Message: Set next waypoint

Format:	\$GPRMC,<GPS Parameters><*cc>	
GPS message data		
Direction: UVC to Host (backseat)		
Where:	Description	Values
\$GPRMC	Message type: GPS Data	“\$GPRMC”
<GPS Parameters>	Message “sentence” compliant to NMEA 0183	See NMEA 0183 Reference for \$GPRMC

		sentence
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.6- Message: GPS Data

Format:	\$C<Heading>P<Pitch>R<Roll>T<Temp>D<Depth> ><*cc>	
OS Compass message data. <i>Note: The negative sign (-) only appears in the negative values, no corresponding (+) appears for positive values.</i>		
Direction: UVC to Host (backseat)		
Where:	Description	Values
\$C	Message type: Compass Data	“\$C”
<Heading>	Magnetic heading in degrees	Range: 0.0 to 359.9
<Pitch>	Pitch angle in degrees, Proceeded by “P”	Range: (-)0.0 to 90.0
<Roll>	Roll angle in degrees, Proceeded by “R”	Range: (-)0.0 to 90.0
<Temp>	Temperature of Compass in degrees C, Proceeded by “T”	Range (-)0.0 to 99.9
<Depth>	Depth in feet of seawater, Proceeded by “D”	Range 0.0 to 999
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.7 - Message: Compass Data

The vehicle’s state data is comprised of the vehicle’s servo values and mode, the next waypoint and distance to the next waypoint, current position, error state, altimeter and remaining park time. In the error state section, if an error is encountered the vehicle will follow the normal behavior of stopping the motor and proceeding to the surface. If the error start “Snc” (Compass has stopped working) is activated the vehicle will stop the current mission.

Format:	\$OSI, <FINMOTOR>,<MODE>,<NEXTWP>,<LATITUDE>,<LONGITUDE>,<SPEED>,<DISTANCETONEXT>,<ERROR>,<ALTIMETER>,<PARK TIME><*cc>	
Vehicle State Data		
Direction: UVC to Host (backseat)		
Where:	Description	Values

\$OSI	Message type: Vehicle State Data	“\$OSI”
<FINMOTOR>	<i>Yaw Top Fin: 2 character Hexadecimal</i> <i>Yaw Bottom Fin: 2 character Hexadecimal</i> <i>Pitch Left Fin: 2 character Hexadecimal</i> <i>Pitch Right Fin: 2 character Hexadecimal</i> <i>Motor Value: 2 character Hexadecimal</i>	Hex 00 to FF
<MODE>	Returns the Servo mode if the vehicle is running a mission or the backseat driver has taken control	“N” = Normal operating UVC “S” = Stopped no mission running “M”=Manual override active “P”=Manual park active “A”=Servo mode “W”=Mission mode active
<NEXTWP>	The waypoint the vehicle is navigating toward	ASCII counting number 1 to max
<LATITUDE>	Current Vehicles Estimated Latitude as a result of GPS input and dead reckoning.	Nnn.nnnnnn decimal degrees
<LONGITUDE>	Current Vehicles Estimated Longitude as a result of GPS input and dead reckoning.	Nnn.nnnnnn decimal degrees
<SPEED>	Estimated speed	Knots
<DISTANCETO NEXT>	Distance to the target (next) waypoint	Nautical miles
<ERROR>	Error state	“N”=No errors, “E”=Battery capacity below 10%, *Safety rules engaged “Sop”=Over pitch “Stl”=Exceed time limit “Sle”=Leak “Sfp”=No forward progress “Sed”= Exceed max depth “Sup”= No upward progress “Stf”= Safety tow float engaged “Srp”= Safety return

		path engaged and SRP mission started “Snd”= DFS has not changed “Snc”= Compass has stopped working
<ALTIMETER>	Current altimeter (depth to bottom) reading	Feet
<PARK TIME>	If the vehicle is parking, the time remaining to park in sec will be transmitted. If it not parking then the time will be zero	P(secs)
<MAGNETIC DECLINATION>	The angle between the local magnetic field and true north	Degrees
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.8 - Message: Vehicle State Data

Format:	\$OPI,<PC>,< RC>,< PWR>,<V>,<C>,<TT>,<BS>,< LK><*cc>	
Vehicle Power System Status		
Direction: UVC to Host (backseat)		
Where:	Description	Values
\$OPI	Message type: Vehicle Power System Status	“\$OPI”
<PC>	Percent remaining battery capacity	Range: 0 to 100 (percent)
<RC>	Remaining Capacity in Watt-hours	Range: 0 to 999
<PWR>	Power in watts being charged or discharged	Range: 0 to 999
<V>	Voltage being used	Range: 0 to 999
<C>	Current in amps, Positive value indicates charging, negative indicates discharging	Range: (+ or -) 0 to 99
<TT>	“Time to” full or empty based on battery state in minutes	Range: 0 to 999
<BS>	Battery State	“C” = Charging, “D” = Discharging, “F”= Fault
<LK>	Leak indicator	0 = No Leak 1 = Leak
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.9 - Message: Vehicle Power System Status

The OPOS commands updates vehicle position and speed while it is underwater and not

receiving a valid GPS position. Updating the vehicle's position can also be done by sending a \$GPRMC GPS string.

Format:	\$OPOS,<LATITUDE>,<LONGITUDE>,<SPEED><*cc>	
Update the vehicle’s present position and speed		
Direction: Host (backseat) to UVC		
Where:	Description	Values
<LATITUDE>	User Provided update to vehicle current position. The backseat may have a more accurate position estimate and want to update the Dead reckoned position to reduce position errors.	(-)Nnn.nnnnn decimal degrees
<LONGITUDE>	User provided longitude	(-)Nnn.nnnnn
<SPEED>	Estimated vehicle speed	nn.nnn knots
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.10 - Message: Updating vehicle's current Location and speed

The \$OLOGL command allows the user to log their own data by setting up labels in the vehicles log. The user must send a \$OLOGL command before a mission is started. Once the labels have been sent, they will remain in the log until the user assigns new labels or clears the labels (*Table 2.11.1*). Once the \$OLOGL command has been sent the UVC will send an acknowledgment command to the backseat driver. The acknowledgment command is described in *Table 2.16*.

Format:	\$OLOGL,<ST1>,<ST2>,<ST3>,<ST4>,...<ST12><*cc>	
Enter labels for the data elements the USER will be logging into the mission LOG file.		
Direction: Host (backseat) to UVC		
Where:	Description	Values
<ST1>	String that names the variables that the user is logging.	“label string” quoted
.... 1-12 elements		
<Stn>	We support up to 12 colums of user data in our geo/time stamped log file.	“label string” quoted
<*cc>	Check sum	XOR byte in Hex

\r\n	(CR) Carriage Return and (LF) Line feed	
------	---	--

Table 2.11 - Message: User Data Logger, Labels

To clear the labels from the log file send the following command described in *Table 2.11.1*

Format:	\$OLOGL,<Clr><*cc>	
Clear user log labels		
Direction: Host (backseat) to UVC		
Where:	Description	Values
<Clr>	Clear user labels used in the log file	NA
<*cc>	Check sum	XOR byte in Hex
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.11.1 – Clear user labels

The \$OLOGD command allows the user to send data for logging in real time to the UVC (*Figure 2.5*). This data will be included in the user log; new data will be updated and written to the log as we parse it. The backseat driver can then send data to be logged by the mission log file. By default the update rate of the system logger is 1 Hz. If the user is sending data at a rate of once every 10 seconds the last data received will be repeated in the log file until new data is sent. If the user is sending data faster than the logging rate some data will be missed.

Format:	\$OLOGD,<ST1>,<ST2>,<ST3>,<ST4>,...<ST12><*cc>	
Log user data for inclusion in the system mission log file. Data is logged at the rate the user has defined in UVC up to 2 Hz.		
Direction: Host (backseat) to UVC		
Where:	Description	Values
<Data1>	Numeric data element to be logged	Number as integer or real (stored as a double).
.... 1-12 elements		
<Data12>	Numeric data element to be logged	Number as integer or real (stored as a double).
<*cc>	Check sum	XOR byte in Hex
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.12 - Message: User Data Logger, Data

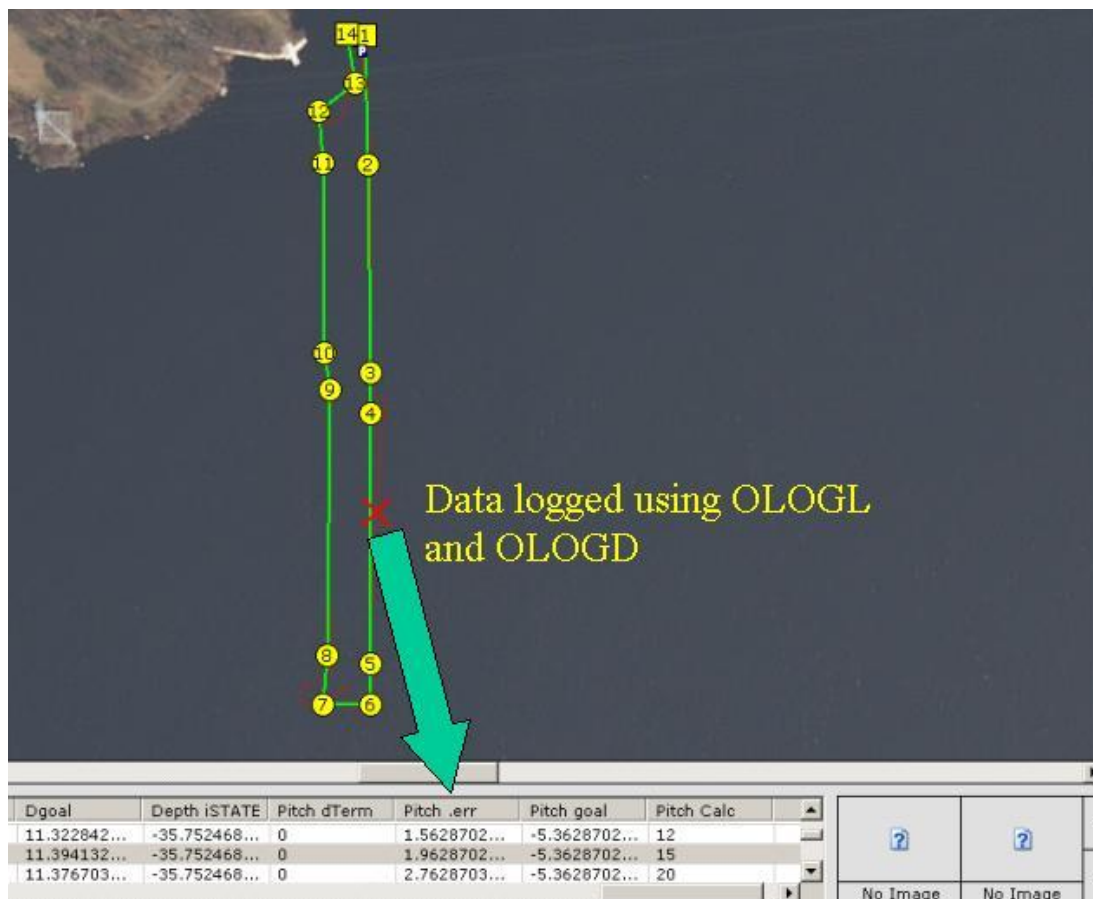


Figure 2.5 - A mission log overlaid with data logged using the \$OLOGD string

The next three commands OMSTOP, OMLOAD and OMSTART allow the backseat driver to control the Iver2's missions by stopping, loading and starting new missions. After each command the UVC will send back an acknowledgement string, which is explained in Table 2.18. The OMSTART command allows the user to start a mission with different options. The first option is to start the mission by ignoring the GPS, but the vehicle will be expecting an OPOS command so its starting position is known. This option is useful if the backseat driver wishes to execute a mission underwater. The second option is to ignore the depth sounder. The last option is to calibrate the pressure transducer, which should be done when the vehicle is on the surface.

Format:	\$OMSTOP,<FL><*cc>	
Stop Mission		
Direction: Host (backseat) to UVC		
Where:	Description	Values
\$OMSTOP	Message type: Stop Current Mission	“\$OMSTOP”
<FL>	Message Flag – the null flag is always included	0 : Null
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.13 - Message: Stop Current Mission

Format:	\$OMLOAD,<DR>,<FL><*cc>	
Load Mission		
Direction: Host (backseat) to UVC		
Where:	Description	Values
\$ OMLOAD	Message type: Load Mission	“\$OMLOAD”
<DR>	Directory or name of mission file	“C:\ mission name” OR “mission name”
<FL>	Message Flag	0 : Null
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.14 - Message: Load New Mission

Format:	\$OMSTART,<FL><*cc>	
Start Mission		
Direction: Host (backseat) to UVC		
Where:	Description	Values
\$OMSTART	Message type: Start Mission	“\$OMSTART”
<FL>	Message Flag – the null flag is included if no other flags are chosen	0 : Null 1 : Ignore GPS (wait for backseat driver to send OPOS) 2 : Ignore Sounder 3 : Calibrate Pressure Transducer
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.15 - Message: Start New Mission

Before starting a mission, the backseat driver can change current user settings, by writing new values to a particular setting. The backseat driver can issue only one read or write command at a time. If a read command is requested, the current setting will be sent to the backseat driver through the \$ACK command, refer to Table 2.18 for more detail. When issuing a read command only include the setting's name and not the state or values. For example if the backseat driver wants the current state of the dive override speed, then the following command will be issued \$ORWSET,R,uDiveSpd,cc. If the dive override is enabled with a speed of 3 knots and no errors have occurred then the following command is sent to the backseat driver \$ACK,12,0,uDiveSpd,2,1,3,cc

If the backseat driver wants to alter the current user settings then the state and the values must be included. For example if the backseat driver wants to disable the dive override speed, then the following will be issued \$ORWSET,R,uDiveSpd,0,cc. If the dive override speed needs to be change from 3 to 4 knots, then the following will be issued \$ORWSET,R,uDiveSpd,1,4,cc. After the mission is completed, the user settings will be reset to there original configuration.

Write Format:	\$ORWSET,< Requested Setting>,<W>,<STATE or VALUES>,<*cc>	
Read Format:	\$ORWSET,< Requested Setting>,< R>,<*cc>	
Write current vehicle user settings		
Direction: Host (backseat) to UVC		
Where:	Description	Values
\$ORWSET	Message type: Read and write current vehicle settings	“\$ORWSET”
< R/W>	Set R/W State	R = Read current setting W = Write setting
<uTrvlonSurf,St>	Travel on Surface at first and last waypoint,State	State = 0 = disable State = 1 = enable <i>State not included in Read</i>
<uDiveSpd,St,Val>	Dive Override Speed,State,Value Enable dive override of <i>Value</i> knots if	State = 0 = disable State = 1 = enable Value = Speed in Knots

	<i>State</i> is enabled	<i>State and Value not included in Read</i>
<uSurfSpd,Val>	Limit Surface Speed,Value Limits surface if it exceeds <i>Value</i> knots	Value = Speed in Knots <i>Value not included in Read</i>
<uWpRadius,Val1,ParkRadius,Val2>	Waypoint victory radius,Value1,Park Radius,Value2 Proceed to the next waypoint is the vehicle is within <i>Value1</i> meters of target WP. Park radius is <i>Value2</i> % of WP radius	Value1 = distance in meters Value2 = Percentage of value1 <i>Values not included in Read</i>
<sMxPitch,St,Val1,Val2>	Max Pitch Angle,State,Value1,Value2 Abort mission if <i>State</i> is enabled and vehicle exceeds a pitch angle of <i>Value1</i> degrees for more than <i>Value2</i> seconds	State = 0 = disable State = 1 = enable Value1= pitch angle degrees Value2= seconds <i>State and Values not included in Read</i>
<sMxTime,St,Val>	Max Time Limit,State,Value Abort mission if <i>State</i> is enabled and vehicle time exceeds <i>Value</i> hours plus total mission time	State = 0 = disable State = 1 = enable Value = hours <i>State and Value not included in Read</i>
<sLeak,St,Val> ,	Leak Detected,State,Value Abort mission if <i>State</i> is enabled and a leak is detected more more than <i>Value</i> seconds	State = 0 = disable State = 1 = enable Value = seconds <i>State and Value not included in Read</i>
<sFwdProg,St,Val>	No Forward progress,State,Value Abort mission if <i>State</i> is enabled and forward progress hasn't been made in <i>Value</i> seconds	State = 0 = disable State = 1 = enable Value = seconds <i>State and Value not included in Read</i>
<sMnDTB,St,Val>	Min depth to bottom, State,Value Adjust depth goal if <i>State</i> is enabled and the current DTB is less than <i>Value</i>	State = 0 = disable State = 1 = enable Value = meters <i>State and Value not included in Read</i>
<sStpDive,St,Val>	Stop Dive,State,Value	State = 0 = disable

	Stop diving if <i>State</i> is enabled and the vehicle has not dove in <i>Value</i> seconds	State = 1 = enable Value = seconds <i>State and Value not included in Read</i>
<sMxDepth,St,Val>	Max Depth,State,Value Abort mission if <i>State</i> is enabled and the vehicle has dove to <i>Value</i> or below	State = 0 = disable State = 1 = enable Value = meters <i>State and Value not included in Read</i>
<sUpProg,St,Val>	Upward Progress,State,Value Abort mission if <i>State</i> is enabled and upward progress hasn't been made in <i>Value</i> seconds	State = 0 = disable State = 1 = enable Value = seconds <i>State and Value not included in Read</i>
<sRevMot;St,Val1,Val2>	Reverse motor,State,Value1,Value2 Reverse the motor <i>Value1</i> times for <i>Value2</i> seconds if a safety rule has been enabled and <i>State</i> is enabled and the vehicle is not floating up	State = 0 = disable State = 1 = enable Value1 = Range: 1 to 10 Value1 = seconds <i>State and Values not included in Read</i>
<sTowFlt,St,Val1>	Safety Tow Float,State,Value1 Enable the tow float if the <i>State</i> is enabled and the vehicle aborts mission and is not making upward progress to the surface in <i>Value</i> seconds	State = 0 = disable State = 1 = enable Value = seconds <i>State and Value not included in Read</i>
<sSafRtPth,St>	Safety Return Path,State Enable the return path if the <i>State</i> is enabled and the vehicle aborts mission and the GPS and compass are working properly	State = 0 = disable State = 1 = enable <i>State and Value not included in Read</i>
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.16 - Message: Read/Write current vehicle user and safety settings

Format:	\$OPK,<LAT>,<LNG>,<TM>,<SP><*cc>
Park Here	

Direction: Host (backseat) to UVC		
Where:	Description	Values
\$OPK	Message type: Go to park location	“\$OPK”
< LAT>	Latitude	(-)Nnn.nnnnn decimal degrees
< LNG>	Longitude	(-)Nnn.nnnnn decimal degrees
<TM>	Time	Minutes
<SP>	Speed	Knots
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.17 - Message: Park location and time

The acknowledgment command send is sent to the backseat driver once the UVC has received a backseat driver command.

Format:	\$ACK,<MT>,<ST>,<ER>,<USRSET>,<USRNUM>,<USRVAL>,<*cc>	
Acknowledgment of mission control commands		
Direction: UVC to Host (backseat)		
Where:	Description	Values
\$ACK	Message type: Load Mission	“\$ACK”
< MT >	Message Type	1 : ACK of OMSTOP command 2 : ACK of OMLOAD command 3 : ACK of OMSTART command 4 : ACK of OLOGD command 5 : ACK of OLOGL command 6 : ACK of OPOS command 7 : ACK of OJW command 8 : ACK of OSD Command 9 : ACK of OMS command 10 : ACK of OMP command

		11 : ACK of OPK command 12 : ACK of ORWSET command 13 : ACK of successfully loading the SRP mission 14 : Sensor Malfunction 15 : ACK of ODVL command 16 : ACK of OMW command
<ST>	Status	0 : No error 1 : Error occurred
<ER>	Error Number	1 : Mission file not found (OMLOAD) 2 : Folder directory not found (OMLOAD) 3 : Errors occurred during loading a mission (OMLOAD) 4 : Sonar Com port not configured (UVC) 5 : Sonar Com port not configured (Sonar Control) 6 : UVC Com port not configured (Sonar Control) 7 : INI file not found, load defaults (Sonar Control) 8 : Error occurred when loading INI file (Sonar Control) 9 : Video Com port not configured (UVC) 10 : UVC Com port not configured (Video Control) 11 : INI file not found, load defaults (Video Control- OMLoad) 12 : Error occurred when loading INI file (Video Control-

		OMLOAD) 13 : Mandatory Com port not configured (UVC-OMLOAD) 14 : Mission not started due to Compass data age (OMSTART) 15 : Mission not started due to GPS data age (OMSTART) 16 : Mission not started due to Sounder data age (OMSTART) 17: Errors occurred during Starting a mission (OMSTART) 18: OPOS incorrect formation/values 19: OJW incorrect formation/values 20: OMS incorrect formation/values 21: OMP incorrect formation/values 22: OPK incorrect formation/values 23: ORWSET incorrect formation/values 24: OLOGL incorrect formation/values 25: OLOGD incorrect formation/values 26: Safety rule engaged, command not processed 27: OSD incorrect formation/values 28 : OMSTOP incorrect formation/values 29 : OMLOAD incorrect formation/values 30 : OMSTART incorrect formation/values 31 : OMLOAD SRP mission not found or
--	--	--

		<p>bad mission file</p> <p>32 : Pressure malfunction</p> <p>33: Camera malfunction (Video Control)</p> <p>34: Compass parameters not correct</p> <p>35: ODVL incorrect formation/values</p> <p>36: OMW incorrect formation/values</p> <p>37 : DeltaT Com port not configured (UVC)</p> <p>38 : DeltaT Com port not configured (DeltaT Control)</p> <p> </p> <p>**If a Mandatory serial port is not setup then the mission will not start, if Sonar or Video comp port is not setup then the device will be ignored</p>
<USRSET>	<p>User setting requested to read</p> <p><i>(The next three labels are only used when the backseat driver issues a read from the \$ORWSET command)</i></p>	User Setting Name
<USRNUM>	Number of values, which includes the state and the values in the returned user setting	Range: 0 to 3
<USRVAL>	<p>Current user setting values, will follow the format: State, Value1, Value2</p> <p><i>(The state and values will change dependent upon the requested user setting)</i></p>	<p>State = 0 = disable</p> <p>State = 1 = enable</p> <p>Value1 = ""</p> <p>Value2 = ""</p>
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.18 - Message: Acknowledge of backseat driver commands

Format:	\$ODVL,<XSPEED>,< YSPEED >,<TMOUT><*cc>
----------------	---

DVL X and Y speed		
Direction: Host (backseat) to UVC		
Where:	Description	Values
\$ODVL	Message type: Go to park location	“\$ODVL”
< XSPEED >	X velocity	Meters / Second
< YSPEED >	Y velocity	Meters / Second
< TMOUT >	Timeout value	0 to 120 seconds
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.19 - Message: Send X and Y Speed from DVL

Format:	\$OMW,<LAT>, <LNG>,< Depth1>,<Depth2>,< Max Angle > <SPD> ,<Park>, <Sensors> <*cc>	
Send waypoints parameters. <i>Any field not used leave blank.</i>		
Direction: Host (backseat) to UVC		
Where:	Description	Values
\$OMW	Message type: Servo Control	“\$OMS”
<LAT>	Latitude	(-)Nnn.nnnnn decimal degrees
< LNG >	Longitude	(-)Nnn.nnnnn decimal degrees
< Depth1>	Enter Depth (+) DFS (-) HFB	Feet
< Depth2>	Enter Depth for undulation. (+) DFS (-) HFB	Feet
< Max Angle>	Maximum pitch angle of the vehicle for changing depth (up or down) in degrees <i>Note: The max angle will override the max pitch angle of the vehicle.</i>	Range: 0 to 45
<SPD>	Speed	Knots
<Park>	Park Time	Minutes
< Sensors>	Send Sensor Data (see Table 2-9)	
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.20 - Message: Mission mode command

When the sensors are enabled the data will be stored in a folder with the following format: WP “original waypoint number” – “current waypoint number in mission mode”. For example if the vehicle was traveling to waypoint 2 and it was interrupted and given a waypoint, the sensor data would be recorded in the folder called WP 2-1.

Format:	OMW Sensor Command,<SS>,< MB >,< Cam1>,< Cam2>	
Request Sensor Data: <i>Any field not used leave blank.</i>		
Direction: Host (backseat) to UVC		
Where:	Description	Values
<SS>	Sidescan Parameters = <Type>-<Gain>-<Range>->-<Freq>-<Channel>	<u>Type</u> : Imagnex = IMG, OceanServer = OSS <u>Gain</u> (db): 0 to 30 <u>Range</u> (m): 15, 30, 60, 90, 120 <u>Freq</u> : H, L <u>Channel</u> : P=Port,S=Starboard, B=Both Ex: SS-21-030-P
<MB>	Multibeam DeltaT Parameters = DT-<Gain>-<AutoGain>-<RangeType>-<Range>	<u>Gain</u> (db): 0 to 20 <u>AutoGain</u> : E = Enable, D=Disable <u>Range Type</u> : Fx=Fixed, 2x = 2x Depth auto range, 3x=3x Depth auto range <u>Range</u> (m): 10,20,30,40,50,60, 80, 100 Ex: DT-17-E-2x-20
<Cam1>	Camera1 Parameters =VC1-<Type>-< Resolution >-<Rate>-<Text>	<u>Type</u> : Stills = S, Video = V <u>Resolution</u> : H=High, M=Medium, L=Low <u>Rate</u> (ms): 500 to 20000 <u>Overlay Text</u> : N=None,D=Depth, P=Poition, B=Both Ex: VC-S-H-1000-3
<Cam2>	Camera2 Parameters= VC2-<Type>-<Resolution >-<Rate>-<Text>	<u>Type</u> : Stills = S, Video = V <u>Resolution</u> : H=High, M=Medium, L=Low <u>Rate</u> (ms): 500 to 20000 <u>Overlay Text</u> : N=None,D=Depth, P=Poition, B=Both Ex: VC-S-H-1000-3

Table 2.21 - Message: Sensor command structure from Mission mode

Format:	\$DVL,<XSpd>,< YSpd>,< ZSpd>,<XDIST>,< YDIST>,<DFS>,< ALT><*cc>	
Vehicle DVL Status		
Direction: UVC to Host (backseat)		
Where:	Description	Values
\$DVL	Message type: Vehicle DVL Status	“\$DVL”
< XSpd>	X Speed (m/s)	If a value of 999.99 is received then the device is not receiving valid data
< YSpd>	Y Speed (m/s)	
< ZSpd>	Z Speed (m/s)	
< XDIST>	X Distance (m)	
< YDIST>	Y Distance (m)	
< DFS>	Depth(m) from upward sounder	
< ALT>	Altitude (m)	
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.22 - Message: Vehicle Power System Status

Format:	\$CTD,<CD>,< TMP>,< SAL>,<SNDSPD><*cc>	
Vehicle CTD Status		
Direction: UVC to Host (backseat)		
Where:	Description	Values
\$CTD	Message type: Vehicle CTD Status	“\$CTD”
< CD>	Conductivity (mmhos/cm)	
< TMP>	Temperature (c)	
< SAL>	Salinity (ppt)	
< SNDSPD >	Sound Speed (m/s)	
<*cc>	Check sum	
\r\n	(CR) Carriage Return and (LF) Line feed	

Table 2.23 - Message: Vehicle Power System Status

APPENDIX A – Backseat driver serial ports and the main AMD CPU serial ports 1 and 4

• COM ports for backseat driver to communicate with the main CPU:

YSI Sensor (COM4 YSI Sonde)
 Mating Connector, Molex Micro-fit 3mm
 43025-0600 Rec. housing
 43030-0009 Rec. crimp terminal
 J18 = COM4
 J9.1 = GND
 J9.2 = No Connect
 J9.3 = Rx
 J9.4 = 12V
 J9.5 = GND
 J9.6 = Tx

COM1 (Sonar)
 Mating Connector,
 Molex Micro-fit 3mm
 43025-0400 Rec.
 housing
 43030-0009 Rec. crimp
 terminal
 J9 = COM1
 J9.1 = Rx
 J9.2 = 5V
 J9.3 = Tx
 J9.4 = GND

• Backseat driver com port
 "input" connector - located near CPU

CPU2 COMA-B
 Molex Micro-fit 3.0
 43025-0600 Rec. housing
 43030-0009 Rec. crimp terminal
 J14 = COMA-B
 J14.1 = COM2_Rx
 J14.2 = GND
 J14.3 = COM1_Rx
 J14.4 = COM2_Tx
 J14.5 = GND
 J14.6 = COM1Tx

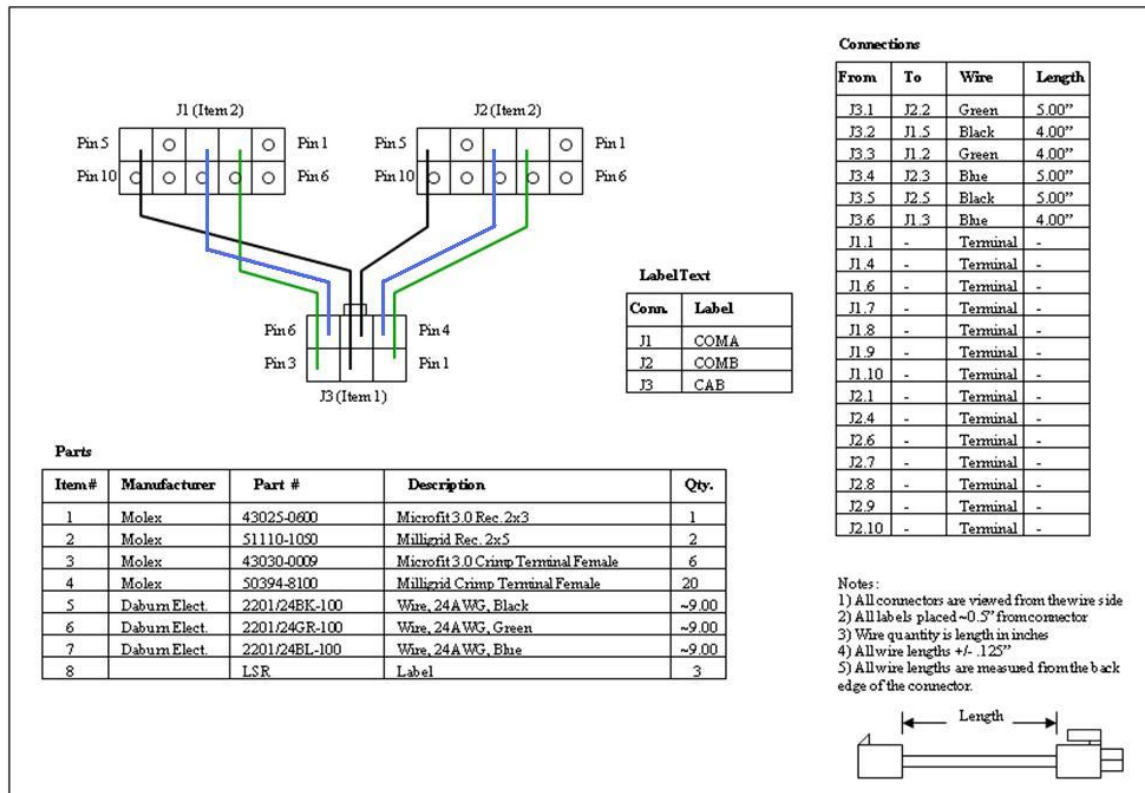
• Backseat driver com port connection on the backplane

COMA
 COM1 from backseat driver
 J8 = COMA
 J8.1 = Rx
 J8.2 = No Connect (default),
 5V (install 0 ohm at R8),
 or 12V (install 0 ohm at R7)
 J8.3 = Tx
 J8.4 = GND

COMB
 COM2 from backseat driver
 J11 = COMB
 J11.1 = Rx
 J11.2 = No Connect (default),
 5V (install 0 ohm at R6),
 or 12V (install 0 ohm at R5)
 J11.3 = Tx
 J11.4 = GND

CPU COMPort connectors:
 Molex Milli-grid
 51110-1050 housing
 50394-8100 crimp terminal

APPENDIX B – Schematic of the Backseat driver serial ports 1 and 2 inserted into the backplane of the AMD CPU



APPENDIX C – Backseat and Main Intel Atom Motherboard Specifications

- OTI CPU: Intel® Atom™ CPU Z530 @ 1.60GHz
- System Chipset: Intel Atom
- BIOS:
- I/O
 - 8 x USB 2.0
 - 15 x RS-232
- System Memory
 - DDR2 400/533MHz
 - 1GB
- Video
 - VGA Video Supports up to 1280x1024 Pixels
**Higher resolutions may be distorted/unusable*
- Environmental
 - Temperature Range: -20....+60 C
 - Storage Temperature: -40....+85 C
 - Temperature Change: max. 10 K / 30 minutes
 - Humidity (relative): 10...90%
 - Pressure 450...1100hPa
- Mechanical
 - Dimensions (L x W): 5.75" x 3.0"
 - Weight: ~ 3.2 oz
- Power
 - 7W (Typical), 10W Max

Hardware Note: USB Keyboards may have issues operating with Windows XP

I/O Connectors

COM PORTS (1-16):

J17: COM 1 (2mm 2x2 Connector)			
Pin	Description	Pin	Description
1	Com 1 Tx	2	Com 1 Rx
3	12V	4	Gnd

J2: COM 2-3 (2mm 2x5 Connector)			
Pin	Description	Pin	Description
1	5V	2	Com 2 Tx
3	Gnd	4	Com 2 Rx
5	12V	6	Com 3 Tx
7	Gnd	8	Com 3 Rx
9	Reserved	10	Reserved

**Com 4 Reserved*

J9: COM 5 (2mm 2x4 Connector)			
Pin	Description	Pin	Description
1	Com 5 Tx	2	Com 5 Rx
3	5V	4	Gnd
5	Reserved	6	Gnd
7	Reserved	8	Gnd

J18: COM 6 (2mm 2x6 Connector)			
Pin	Description	Pin	Description
1	Com 6 Tx	2	Com 6 Rx
3	<i>Reserved</i>	4	Gnd
5	<i>Reserved</i>	6	Reserved
7	<i>Reserved</i>	8	Reserved
9	<i>Reserved</i>	10	Gnd
11	<i>Reserved</i>	12	Gnd

J10: COM 7-8 (2mm 2x6 Connector)			
Pin	Description	Pin	Description
1	Com 8 Tx	2	Com 8 Rx
3	5V	4	Gnd
5	Com 7 Rx	6	Com 7 Tx
7	Reserved	8	Gnd
9	Reserved	10	Reserved
11	Reserved	12	Reserved

J16: COM 10 (2mm 2x2 Connector)			
Pin	Description	Pin	Description
1	Com 10 Tx	2	Com 10 Rx
3	12V	4	Gnd

J13: COM 9, 11-13 (2mm 2x6 Connector)			
Pin	Description	Pin	Description
1	Com 9 Tx *	2	Com 9 Rx *
3	Gnd	4	Gnd
5	Com 11 Tx	6	Com 11 Rx
7	Com 12 Tx	8	Com 12 Rx
9	Gnd	10	Gnd
11	Com 13 Tx	12	Com 13 Rx

*Com 9 is wired to the Front seat Driver

J14: COM 14-16 (2mm 2x5 Connector)			
Pin	Description	Pin	Description
1	Com 14 Tx	2	Com 14 Rx
3	Gnd	4	Gnd
5	Com 15 Tx	6	Com 15 Rx
7	Com 16 Tx	8	Com 16 Rx
9	Gnd	10	NC

USB PORTS (A-H):

J11: USB A-B	
Jack	Description
Top	USB Port A
Bottom	USB Port B

J5: USB C-D	
Jack	Description
Top	USB Port C
Bottom	USB Port D

J12: USB E-F (2mm 2x4 Connector)			
Pin	Description	Pin	Description
1	VCC_USB_E	2	GND_USB_F
3	USB_E_Data -	4	USB_F_Data +
5	USB_E_Data +	6	USB_F_Data -
7	GND_USB_E	8	VCC_USB_F

J15: USB G-H (2mm 2x4 Connector)			
*USB 2.0 Devices ONLY			
Pin	Description	Pin	Description
1	VCC_USB_G	2	GND_USB_H
3	USB_G_Data -	4	USB_H_Data +
5	USB_G_Data +	6	USB_H_Data -
7	GND_USB_G	8	VCC_USB_H

VIDEO OUTPUT:

1.1 J4: VIDEO OUT (2mm 2x6 Connector)			
Pin	Description	Pin	Description
1	RED	2	SCL
3	GREEN	4	SDA
5	BLUE	6	Gnd
7	VSYNC	8	Gnd
9	HSYNC	10	Gnd

11	3.3V	12	Video Power (3.3V)
----	------	----	--------------------

*Pins 11 and 12 of J4 are shorted.

POWER IN:

J7: POWER IN (.100 2x5 Connector)			
Pin	Description	Pin	Description
1	ATX PS_ON_L	2	Gnd
3	Gnd	4	3.3V
5	12V	6	5V
7	5V	8	5V
9	POWER OK	10	Gnd

J3: IDE Disk (2x22 Connector)			
Pin	Description	Pin	Description
1	Buf_Rst_Out_L	2	GND
3	PDATA7	4	PDATA8
5	PDATA6	6	PDATA9
7	PDATA5	8	PDATA10
9	PDATA4	10	PDATA11
11	PDATA3	12	PDATA12
13	PDATA2	14	PDATA13
15	PDATA1	16	PDATA14
17	PDATA0	18	PDATA15
19	GND	20	NC_PATA_KEY
21	DDRQ	22	GND
23	IO_WRITE_L	24	GND
25	IO_READ_L	26	GND
27	IOCHRDY	28	CABLE_SEL
29	DDACK_L	30	GND
31	PATA_IRQ	32	NC
33	ADDR1	34	DMA66_DET
35	ADDR0	36	ADDR2
37	PATA_CS1	38	PATA_CS3
39	NC_PATA_ACTIVITY	40	GND
41	5V	42	5V
43	GND	44	GND

RESERVED CONNECTORS

- J6, J8, J19, J20, J21, J23, J24 are all Reserved Connectors

