TELEDYNE
WEBB RESEARCH
A Teledyne Technologies Company

82 Technology Park Drive
E. Falmouth, Massachusetts 02536
Phone: 508.548.2077
Fax: 508.540.1686
glidersupport@webbresearch.com

May 2010

Table of contents

This document is a field guide and reference documentation for use in preparation and deployment of Teledyne Webb Research Slocum Gliders.

Please also refer to the complete User Manual, Slocum Glider at:
ftp://ftp.glider.webbresearch.com/glider/windoze/production/src/doco/MANUAL/

The site above is an authorized user restricted site. To request access contact:
Glideraccess@webbresearch.com

For technical glider assistance contact:
Glidersupport@webbresearch.com

Qualified personnel
Only trained and qualified personnel should operate and maintain the glider. Teledyne Webb Research conducts regular training sessions several times a year. Glider users should attend a training session and understand basic glider concepts and terminology. Contact glidersupport@webbresearch.com for information regarding training sessions. Company policy is to fully support only properly trained individuals and groups.

Only personnel who have attended a Teledyne Webb Research training session should use this document.

Internet resources:

Sign into access restricted glider documentation
https://dmz.webbresearch.com

Software distribution
ftp://ftp.glider.webbresearch.com/glider/

Slocum User Manual
ftp://ftp.glider.webbresearch.com/glider/windoze/production/src/doco/MANUAL/

GMC user guide (Dockserver Manual)
ftp://ftp.glider.webbresearch.com/glider/windoze/production/src/gliderMissionControl/Documentation/gmcUserGuide.pdf

Windows executables (replace windoze with linux for linux)
ftp://ftp.glider.webbresearch.com/glider/windoze/production/windoze-bin/

Glider service bulletins:
ftp://ftp.glider.webbresearch.com/glider-service-bulletins/

Update glider code procedure:
ftp://ftp.glider.webbresearch.com/glider/windoze/production/src/doco/software-howto/updating-all-glider-software.txt

masterdata:
ftp://ftp.glider.webbresearch.com/glider/windoze/production/src/code/masterdata

Glider operations mission planning overview worksheet

Glider number		Prepared	
Payload instruments		by	
Deployment location	Surf Temp	Surf Sal	Density
Deployment Date			
Deployment notes			
Science collection notes			
	date	tech	notes
Ballast Complete			
Software check list complete			
Missions simulated			
Dockserver tested and updated			
Dockserver IP			
Pre-seal check list complete			
Post-seal check list complete			
All supplies packed			
Deployment details			
Cruise leaves			
Arrive on station			
Recovery details			
Cruise leaves			
Emergency recovery plans			
Pilots contact info	When	Phone	email

Mission notes

Pre Mission Seal Checklist (final seal)

– All ballasting complete and weights are adjusted (see page 9)

	date	tech	notes
Fore			
Pump lead screw clean and greased			
Pitch Lead screw clean and greased			
Leak detect in place batteries secure			
Ballast bottles secure			
O-ring inspected and lubed			
Exterior nose/bellow clean of debris			
Interior Clean of debris			
Reconstituted or Fresh Desiccant installed			
Payload			
Science serial numbers			
1			4
2			5
3			6
Wiring dressed			
O-ring inspected and lubed			
Payload weights properly secured			
CF card fully seated and loaded			See Software checklist (lab section)
Persistor button batteries checked			voltage
Interior clean of debris			
Aft			
Iridium Sim card installed			
Sim number			
Aft tray wiring dressed			
CF Card seated and loaded			See Software checklist (lab section)
Persistor button batteries checked			voltage
Ballast bottle secure			
O-ring inspected and lubed			
Battery voltages @ J13			
fore			voltage
pitch			voltage
aft			voltage
all			voltage
Battery voltage @ J31 (emergency)			
Anode to main tray continuity			
Threaded rod clean and greased			
Seal			
O-rings clean of debris			
15 in/lb torque			
All sections snug together			
Vacuum pulled			

Post seal Checklist

	date	tech	notes
General			
Pick-point installed			
Wing rails installed			
Wings and spares packed			
Hardware			
Exterior connectors secure and fastened			
Altimeter			
Aanderaa (if present)			
Burn wire			
MS plug seated			
Ejection weight assembly not seized			
Pressure sensors clear and clean			
Aft (flight)			
Payload (science)			
Continuity - Aft anode to Tail boom			
Bladder visual inspection			
Cowling installed			
Powered by battery **inside lab**			
Lab_mode_on			
Report ++ m_vacuum (6 in/Hg 7 for 1000m)			In/hg
Report ++ m_battery			volt
Lab_mode on Wiggle on			No errors for +5 minute
Verify time			
Verify science			
Put c_science_all_on 0 (off = -1)			
Put c_science_on 3 (off = 1)			
Put c_science_send_all 1 (off = 0)			
Powered by batt **Outside lab tests**			
3hrs Argos **put c_argos_on 3 (off = 1)**			
Confirm receipt of messages at Argos			
Confirm GPS			
Confirm Compass			
Dockserver comms- send and receive files			
Run status.mi			

Notes

Shipping Checklist

	date	tech	notes
Glider packed and secured w/ three straps			
Mobile computer packed			
Freewave and Freewave antenna			
Buoy with rope			
Glider evac tools			
Glider tools			
Red and green shorting plugs			
Wings packed			
Shipping address and details arranged			
Address	Contacts		details
Flights	Contacts		details

BALLASTING AND H-MOMENT

	date	tech	notes
Glider under vacuum			
Pick-point installed			
Wing rails installed			
Wings installed			
Exterior connectors secure and fastened			
Altimeter			
Aanderaa (if present)			
Burn wire			
MS plug seated			
Ejection weight assembly not seized			
Pressure sensors clear and clean			
Aft (flight)			
Payload (science)			
Bladder visual inspection			
Powered			
Report ++ m_vacuum (6 in/Hg 7 for 1000m)			
Report ++ m_battery			
Lab_mode on - **Wiggle on**			
ballast			
Cowling installed			
While in ballast tank			
Ensure no air in front or aft sections			
Note roll for potential adjustment			
Record weight adjustments necessary			
Rinse and dry after wettings			
Exit and power down glider when done			

Glider Ballast Worksheet

Glider Name: ___________ Date: ___________

Glider Displacement Disp (Liters): ___________ Technician:___________

TANK WATER:

Temperature (degrees C):___________

Conductivity(S/M):_______________

Salinity(PSU):___________________

Density(kg/cu m)_________________

TARGET WATER

Temperature (degrees C)___________

Conductivity(S/M):_______________

Salinity(PSU):___________________

Density(kg/cu m):________________

First Run

	Forward	Payload	Aft
Weight Removed			
Weights Added			

Roll:

Weight conversion constants

Stainless Steel = .875 * (weight added external)

Lead = .912 * (weight added external)

Second Run

	Forward	Payload	Aft
Weight Removed			
Weights Added			

Roll:

Third Run

	Forward	Payload	Aft
Weight Removed			
Weights Added			

Roll:

Final Weight Configuration As Shipped

Forward	weight	Payload	weight	Aft	weight
Port Bottle		Top FWD		Aft Bottle	
STBD Bottle		Bottom FWD			
Bottom Bottle		Top AFT			

Roll:

H-Moment

Software Checklist

	date	tech	notes
Flight CF Card contents archived			
Version updated			Version
Logs archived/deleted			
If new version			
Boot pico			
Load new app			
Install Autoexec.mi in config directory			
Burnapp			
Confirm App			
Boot app			
Payload CF Card contents archived			
Version updated			
Logs archived/deleted			
If new version			
Boot pico			
Load new app			
Install Proglets.dat in config directory			
Burnapp			
Confirm App			
Boot app			
Directory’s flight Persistor			
/Config			
Simul.sim deleted			
Configure sbdlist.dat and mbdlist.dat			
Autoexec.mi			
sensor: c_iridium_phone_num			number
sensor: F_MAX_WORKING_DEPTH(m)			Depth (m)
Confirm Installations			
Confirm calibration coefficients			Only necc if new hardware
/ma /missions			
Load custom .mi and .ma files			Files loaded
Sci>/proglets.dat			
Confirm desired sensors are installed			
Archive of all files locally			

Notes for Ballasting and lab tests

If the glider is not already closed up with a proper vacuum, you will need to do this before you can apply power to the glider. To do this pull the glider together with the tie rod using the long 24" T-handle provided Hex wrench until the hulls have come together. Set the torque to 15 in/lbs using the torque handle and long extension provided. With the vacuum tool and the long T-handle, put a vacuum on the glider. Your target is 6" hg (7 for 1000m), but it is best to pull a vacuum higher than this as you can bleed some off when the glider is powered on. Once this is accomplished, and the MS plug is in place, you may apply power. The glider will power on and go through its normal start up routine.

When you see:

SEQUENCE: About to run initial.mi on try 0
You have 120 seconds to type a control-C to terminate the sequence.
The control-P character immediately starts the mission.
All other characters are ignored.

Type **CTRL-C**. This will give you a GliderDos prompt. From the GliderDos prompt:

1. Type **callback 30**. This will hang up the iridium phone for 30 minutes. You can enter any value for callback from 1 to 30. Alternately you can type **use – iridium** to take the iridium out of service until you are done with your testing. NOTE: If you do this remember to type **use + iridium** when you are finished to put the iridium back into service.

2. Type **lab_mode on**. This puts the glider in lab mode and will prevent the glider from trying to run its default mission.

3. Type **ballast**. This will deflate the air bladder, put the pitch motor to zero and the ballast pump to zero.

4. Type **report ++ m_vacuum**. This will display the vacuum inside the glider every time the sensor updates. If the vacuum is already at 6" (7" for 1000m) hg you are done(+/- .2). If not you will need to adjust the vacuum.

5. Type **report clearall.** This will stop outputting the vacuum value.

Put the aft cowling on the glider. If you are connected via an external power supply you will need to power down by typing **exit** before installing the cowling. Re-power if necessary and follow steps 1-3. You are now ready to put the glider into the ballast tank.

You will need to get CTD data from the glider so that you can make your final weight adjustment calculations from ballast tank to real conditions. In order to do this:

6. Type **put c_science_all_on 0** **(off = -1)**. This will tell the science computer to sample all science sensors as fast as possible.

7. Type **put c_science_on 3** **(off = 1)**. This will display that data to the screen.
8. Type **put c_science_send_all 1** **(off = 0)** to send science to flight persistor.

Pick out the water temperature and conductivity and calculate your salinity and density. Enter this data into the ballasting and H-moment calculator sheet in the appropriate blocks. Enter the temperature, density and salinity for your target water into the appropriate blocks to get your total weight change from tank to real world conditions. It is important to remember that you need to make the glider neutral in the tank and do an H-moment calculation before you make this adjustment. To do the H-moment calculation, with the glider neutral in the tank:

8. Type **report ++ m_roll**. This will display the roll of the glider every time the Sensor updates, in radians.

Follow the instructions for calculating the H-moment on the ballasting and H-moment calculator spreadsheet.

Common Lab commands

While in **lab_mode on** (to exit **lab_mode off**) **(never launch the glider in lab_mode)**
Ballast zeros motors and deflates air bladder **(never launch the glider in ballast)**
Use – iridium or **callback 30** stops iridium phone calls
Report ++ (any_masterdata_sensor) Reports sensor as fast as possible
Report ++ m_battery
Put (any_masterdata_sensor)
report clearall turns off all reporting
example **Put c_fin 0** zeros fin after wiggle
Type **wiggle on**. This exercises the ballast pump, pitch motor, and fin motor
Type **wiggle off** to stop exercising the motors
Type **put c_science_all_on 0** **(off = -1)**. This will tell the science computer to sample all science sensors as fast as possible.
Type **put c_science_on 3** **(off = 1)**. This will display that data to the screen.
Type **put c_science_send_all 1** **(off = 0)** to send science to flight persistor.
If you need to apply power to the glider in an open state (no vacuum) you will need to do the following before powering down and opening the glider:

Type **exit pico**. This will bring you to a pico dos prompt.
Type **boot pico** to set the glider to boot into pico dos.
Type **boot –lab** from picodos to enter straight into **lab_mode on**
This will allow you to power up the glider without the fear of running the ballast pump. If the ballast pump is run on the bench without a vacuum, it can damage the forward rolling bellafram. When you are finished, close the glider back up, apply the vacuum and type **boot app** to set the glider to boot the application. You must always make sure the glider is set to boot app before doing any in the water tests.

Lab_mode off to exit lab mode
Exit reset to cycle to default settings
Exit and wait for prompt to remove green plug or power supply- Install red plug.

Pre mission check outs

These procedures should be followed for qualification of a glider for launch of a mission.

On the beach, deck and/or at lab with the glider outside with a clear view of the sky:

Power on glider and when prompted type **control-C** to exit to GliderDos.
From the GliderDos prompt,
Type **callback 30** to hang up the iridium phone.
Type **Lab_mode on**
Type **put c_gps_on 3** **Confirm GPS**
In the string like the following the highlighted A should turn from a V to and A.\
`gps_diag(2)cyc#:538|GPRMC,161908,A,5958.3032,N,`
`7000.5568,W,0.000,343.9,190808,0.3,W|`
After a number of A responses type **put c_gps_on 1** to stop screen display.

Type **wiggle on** and run for 3-5 minutes to check for any device errors or other abnormalities. Type **wiggle off** to stop wiggling.

Report ++ m_vacuum (remember vacuum can fluctuate with temperature)
Report ++ m_battery
report clearall

If no errors are found, type **lab_mode off** to return to the GliderDos prompt.

Note:
Make sure that the glider is not simulating or in boot Pico or Lab Mode before deployment.

Purge Log directory send logs over Freewave or **dellog** (this can take a long time if there are a large number of files and they will be lost). It is advised to purge and archive the log files in the lab.

Type **run status.mi** and confirm that all sensors are being read. Mission should end “mission completed normally”.

Let glider connect to Dockserver and send .sbd over Iridium files if not connected **Callback 1** to force iridium to call in one minute once connected.
Example of forcing iridium while Freewave is present:
GliderDos I -3 >**send -f=irid *.sbd -num=2** (this will send 2 most recent .sbd files over iridium – be patient as iridium is slow and there presently no positive feedback over Freewave).

Science sensor check out

Type **put c_science_all_on 0** (off = -1) This will tell the science computer to sample all science sensors as fast as possible.
Type **put c_science_on 3** (off = 1). This will display that data to the screen.
Type **put c_science_send_all_1** (off = 0) to send science to flight persistor.

Pack Glider ensuring use of all cart and crate straps and locks and/or load glider into the boat and proceed to the first waypoint or deployment location.

See deployment and recovery section

In the water:

Attach a line with flotation to the glider before putting it in the water. If you have great confidence in the glider's ballasting you may choose to not test on the line. Once the glider is in the water type **run status.mi** once again.

Run one or several of the following missions while on station until satisfied that the glider is ballasted and operating normally.

Run Ini0.mi	Does a single yo to max depth 3 meters, min depth 1.5 meters. Uses a fixed pitch battery and fin position
Run Ini1.mi	Does 3 yos to the north diving to 5 meters and climbing to 3 meters. Pitch should be +/- 20 degrees.
Run Ini2.mi	Goes to a waypoint 100 meters south of the dive point diving to 5 meters and climbing to 3 meters. Pitch should be +/- 20 degrees.
Run Ini 3.mi	Goes to a waypoint 100 meters north of the dive point diving to 5 meters and climbing to 3 meters. Pitch should be +/- 20 degrees.

Send files locally and/or by Iridium. Confirm flight data and desired flight characteristics of ini missions run, if necessary turn flight control over to Dockserver over Iridium.

If you have not removed buoy and the line from the glider, do it now.

From the GliderDos prompt type **exit reset**. This will force a re-initialization of all of the sensor values.

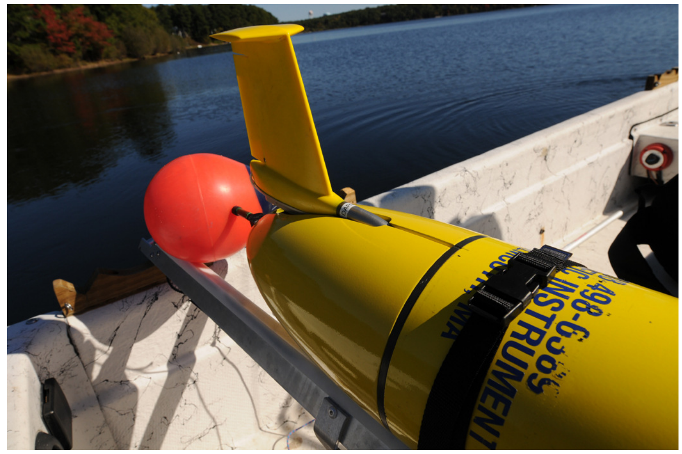
When the glider re-boots, when prompted type **control-C** to bring up the GliderDos prompt and type **loadmission waterclr.mi** to zero any built up water currents that are remembered long term.

Type "**run glmpc.mi**" or equivalent **.mi** to begin the desired mission.

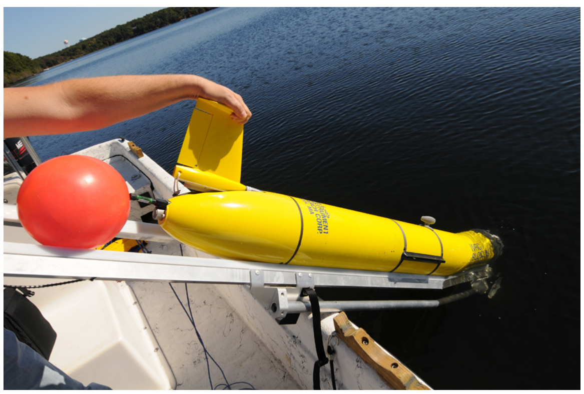
Glider Deployment

Deployment at sea can be dangerous, and the welfare of crew and glider handlers should be considered while at the rail of a ship. From a small boat the glider cart can be used to let the glider slip easily into the water. Remove the nose ring in the original cart design or when ready release the nose ring with handle bar release on newer carts.

For larger boats the pick point affixed to the payload bay should be used to lower and raise the glider with a crane or winch from the vessel to the water.



Glider with buoy and rope ready for first deployment



Note in the deployment sequence above that the Digifin can be handled.
The tail boom should be used for handling a glider not equipped with Digifin.

Large ship deployment

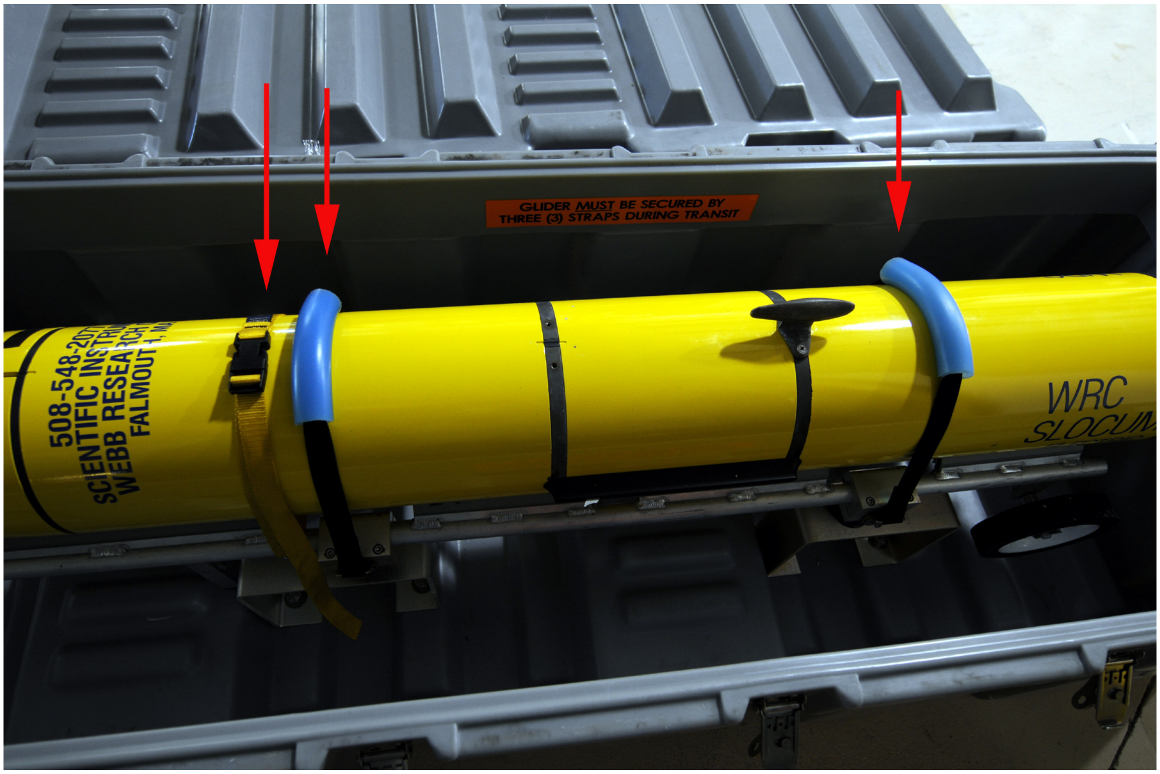
A quick release system utilizing the pick point can be fashioned from supplies found on most vessels, as illustrated in the following two images.

Glider Recovery

Note a boat hook can be used to manipulate the glider in the water. Care should be taken with non-Digifin gliders during deployment and recovery as the fin can be knocked out of calibration or damaged if handled too aggressively. Handle by tail boom or pick point only with non-Digifin designs.

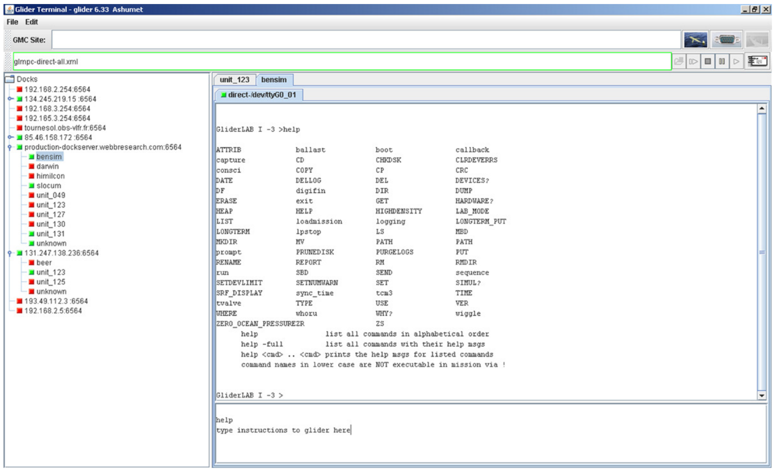
Lower the cart with nose ring into water and manipulate the glider by tail boom into position on cart. Lift and tilt the glider onto the ships deck.

Glider packing



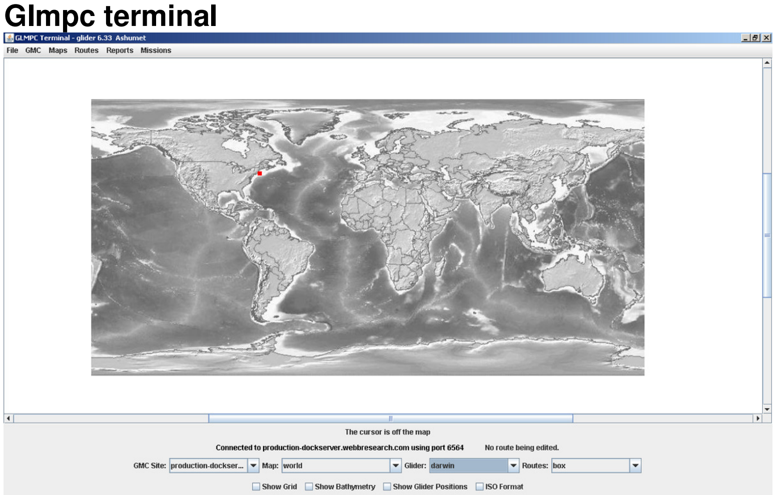
Ensure that all three straps are secure (2 crate straps and 1 cart strap). If extra supplies are included in crate, ensure that they will not interfere with the fin or become dislodged during transit.

Dockserver
Dockserver is the name of the laptop or rack mounted Linux Centos 4 based P.C. provided with a glider. The applications (also named Dockserver and Dataserver) must be launched from desktop icons to provide full Dockserver functionality.

Glider Terminal


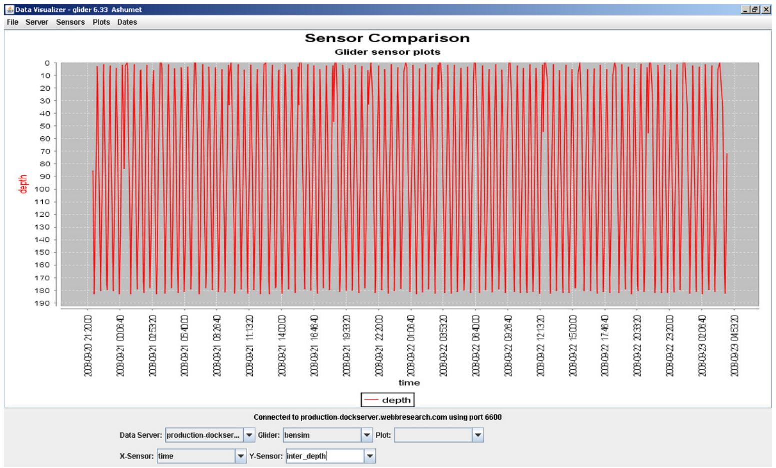
Primary interface through Dockserver to glider

Top panel – Dockserver site manual entry – Script functionality – Terminal and ports perspective toggle and remote glider notification tabs.
Left panel- active Docks and Gliders.
Middle right- communication from glider.
Bottom right- communication to glider.

Glmpc terminal


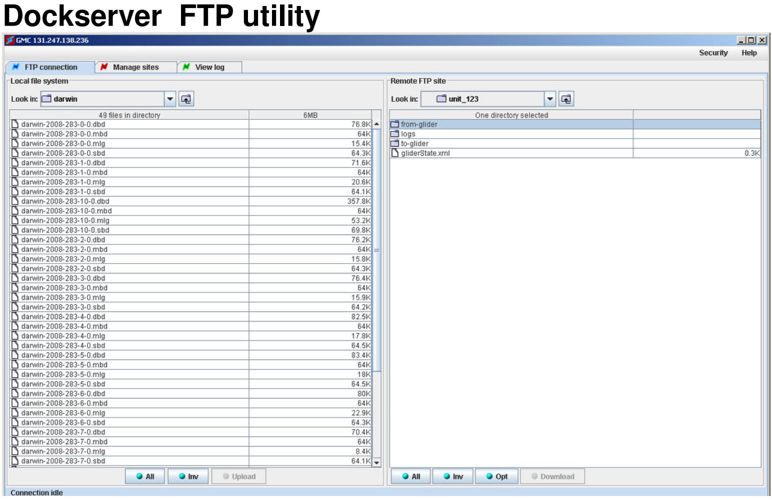
Real time Visual interface, which allows custom jpg maps and click through uploading of waypoints during live missions.

Data Visualizer



The Data Visualizer server must be running on Dockserver to view data remotely. Launch with desktop icon on Dockserver. Allows pilots to plot all glider data as it is received by Dockserver.

Dockserver FTP utility



**Whenever new files are sent to Dockserver you must disconnect and reconnect to refresh the file list.

.

Configure Comms with Terminal program (Procomm Plus).
Many users have decided to have a mobile Dockserver and a permanent installation Dockserver. If you do not have a mobile Dockserver the following settings will allow direct communications with a terminal program to the glider.

Connect powered Freewave to serial com port on computer with provided serial cable.
Open Procomm plus and select the following ProComm plus Terminal program settings
Select proper com port
Baud 115200
Parity N-8-1
Go to Options -> System Options -> Modem Connection.
Click on Modem Connection Properties.
If the Use hardware flow control check box is unchecked, check it and click OK.
Click on the Data tab.
Next to Receiver Crash Recovery Settings, click Change Settings.
Check If date/time match under Crash Recovery Options.
Check Overwrite if incoming newer under Overwrite Options.
Click OK.
Next to Sender Crash Recovery Settings, click Change Settings.
Check Crash recovery off under Crash Recovery Options.
Check Always overwrite under Overwrite Options.
Click OK.
Select Streaming from the Transmit method menu and uncheck Use local EOL convention.
Select 32 bit CRC from the Error detection menu and check Original file time stamp.
Click OK.

You're now ready to begin comms with glider and ZR/ZS testing.

Note there are know problems with using Hyperterminal and attempting to ZR/ZS.
Tera term is another viable terminal program.

Commonly used Glider Commands
From a GliderDos prompt the command **help** will list all commands available to the user:
Partial help menu and definitions
* see User Manual Appendix command examples for examples of this command

ballast	BALLAST ? ; for help	
boot	[PICO][PBM][APP]	
callback	<minutes til callback>	
capture	[d:][p]fn [/Dx/B/N/E]	*
CD	Change Directory	
CLRDEVERRS	zero device errs	
consci	[-f rf|irid] ; console to science	
COPY	source dest [/V]	
CP	<src_path> <dest_path> ; copy a file system branch	
DATE	[mdy[hms[a|p]]] /IEUMCP]	
DELLOG	ALL MLG DBD SBD	
DEL	[drv:][pth][name] [/P]	
DEVICES?	print device driver info	
DIR	[d:][p][fn] [/PWBLV4A:a]	*
exit	[-nofin] [poweroff|reset|pico|pbm]	
GET	GET <sensor name>	
HARDWARE?	[-v] ; Hardware config	
HEAP	Report Free Memory	
HELP	Print help for commands	
HIGHDENSITY	HIGHDENSITY ? ; for help	
LAB_MODE	[on|off]	
LIST	;display all sensor names	
loadmission	loads mission file	
logging	on|off ; during GliderDos	
LONGTERM_PUT	LONGTERM_PUT <sensor name> <new value>	
LONGTERM	LONGTERM ? ; for help	
LS	[path] ; list a file system branch	
MBD	MBD ? ; for help	
MKDIR	[drive:][path]	
MV	<src_path> <dest_path> ; copy a file system branch	*
PRUNEDISK	Prune expendable files to free space on disk	
PURGELOGS	Deletes sent log files	
PUT	PUT <sensor name> <value>	*
RENAME	[d:][p]oldname newname	*
REPORT	REPORT ? ; for help	
RMDIR	[drive:][path]	
run	[mission_file] ; runs it	
SBD	SBD ? ; ? for help	
SEND	[-f={rf}|{irid] [-num=<n>] [-t=<s>] [filespec ...]	*
sequence	SEQUENCE ? ; do this for help	
SETDEVLIMIT	devicename os w/s w/m	*

SETNUMWARN	[X] ; set max dev warnings to X
SIMUL?	print desc of what is simulated
SRF_DISPLAY	SRF_DISPLAY ? ; for help
sync_time	[offset] ; Syncs system time with gps time
TIME	[hh:mm:ss [a|p]] [/M/C]
TYPE	[drv:][pth][name]
USE	USE ? ; do this for help
VER	Firmware versions
WHERE	prints lat/lon
whoru	Vehicle Name:
WHY?	[abort#] ; Tells the reason for an abort
wiggle	[on|off] [fraction] ;moves motor
ZERO_OCEAN_PRESSURE	re-calibrate(zero) ocean pressure sensor
ZR	Zmodem Rec: zr ? for help
ZS	Zmodem Send: zs ? for help

Surface dialog

The following is an example of surface dialog.

```
Glider bensim at surface.
Because:Hit a waypoint [behavior surface_2 start_when = 8.0]
MissionName:initial.mi MissionNum:bensim-2010-123-2-0 (0103.0000)
Vehicle Name: bensim
Curr Time: Tue May  4 13:25:20 2010 MT:     316
DR  Location:  3342.801 N -11824.540 E measured      1.487 secs ago
GPS TooFar:   69696969.000 N 69696969.000 E measured     1e+308 secs ago
GPS Invalid :  3342.832 N -11824.533 E measured    252.734 secs ago
GPS Location:  3342.801 N -11824.540 E measured      3.994 secs ago
   sensor:m_battery(volts)=13.121562938211         3.926 secs ago
   sensor:m_iridium_signal_strength(nodim)=-1      1e+308 secs ago
   sensor:m_leakdetect_voltage(volts)=2.5          3.921 secs ago
   sensor:m_vacuum(inHg)=6.50223565323565          8.214 secs ago
devices:(t/m/s) errs:   0/   0/   0 warn:   0/   0/   0 odd:   0/   0/   0
ABORT HISTORY: total since reset: 0

   Hit Control-R to RESUME the mission, i.e. dive!
   Hit Control-C to END    the mission, i.e. GliderDos
   Hit Control-E to extend surface time by 5 minutes.
   Hit Control-W to get device warning reports.
   Hit Control-F to re-read yo, goto_l, sample, drift_ mafiles.
   Hit S [-f={rf}|{irid] [-num=<n>] [-t=<s>] [filespec ...] to send log files
   Hit ! <GliderDos cmd>  to execute <GliderDos cmd>
   Hit Control-T to consci to science computer when comms ready:
                                   ... communications NOT ready for consci.
                                   ... because: sci_m_science_on = 0

Water Velocity Calculations COMPLETE
Waypoint: (3342.8323,-11824.5333) Range: 58m, Bearing: 11deg, Age: -1:-1h:m
Drifting toward outer watch circle, centered on waypoint
Now 58.3 meters from middle, will dive at 100.0 meters
Time until diving is: 150 secs(estimated)
```

File manipulation quick tutorial
send *.XXX (works from GliderDos) - only .dbd .sbd .mdb .mlg files (30 most recent)
send *.* (works from GliderDos) sends all dbd .sbd .mdb .mlg files and .tbd, .nmd .ebd and .nlg files (30 most recent)

Do not use send *.* over iridium

s (works from surface dialog) while in mission - only .dbd .sbd .mdb files
s *.* (works from surface dialog while in mission) sends all dbd .sbd .mdb .mlg and .tbd, .nmd .ebd and .nlg files (30 most recent of each)

Do not use s *.* over iridium

zr (works form GliderDos) with terminal emulator all files types
dockzr (works from GliderDos) while using Dockserver* all file types
*file must be in to glider directory on Dockserver
zs (works from GliderDos) all file types
!zr (works while in mission) all file types from terminal emulator
!zs (works while in mission) all file types
!dockzr (works while in mission) from Dockserver
Care must be taken when sending files over Iridium. .dbd files should not be sent over iridium in normal conditions. .dbd files are prohibitively large (1 to 8 Mbytes is not uncommon) which results in large surface times and large expense to the user. Terms are from glider perspective: send = send from glider to shore. R = Receive from shore to glider.

Full file manipulation tutorial.
To send data files from the glider in GliderDos, to the Dockserver or a computer running a terminal emulator the command is **send.** The command **send *.*** will send the 30 most recent files of type .sbd, .mbd. dbd. .mlg and the sys.log. If Freewave and iridium are both present files will be sent over Freewave. The pilot can also specify a specific type of file, **send *.sbd** (30 most recent) or a specific file **send XXXXXX.sbd**. A pilot should never use the wildcard ***.*** when Freewave comms is not present.
If the glider is in a mission, the send command is truncated to **s**. All of the criteria above remain.

To send any other type of file .mi, .ma, .dat, etc from the glider, the command is **zs filename**. To send these types of files the pilot must first **cd** into the directory where the desired file resides. To send these types of files while in a mission during a surface dialog the command must be proceeded by **!** example: **!zs autoexec.mi**.

To send a file to the glider from a computer running a terminal emulator program to the glider, the command is **zr**. The proper upload path needs to be selected in the terminal program. When using Dockserver the command is **dockzr filename** or **dockzr *.*** and the desired file or files must be in the to glider directory, for the glider in question, on the Dockserver.
After sending data files from the glider the code will move the files from the logs directory to the sentlogs directory.

THINGS TO NEVER DO WITH A GLIDER

Never power up a shallow glider without a vacuum

Never run a simulation on a glider other than “on_bench”

Never deploy a glider in simulation

Never pick a glider up by the rudder/fin (digifin can be handled)

Never deploy a glider in “boot pico”

Never exit to pico during a deployment.

Never power on a glider with more than 15v DC from an external power supply.

Never deploy a glider in lab_mode

Never perform the top of a yo below 30 meters (with 100 or 200 meter glider)

Never secure the glider to the glider cart while over railing or in the water

THINGS TO DO WITH A GLIDER

Do secure it properly in crate with all 3 straps for shipping

Do use fresh desiccants on each deployment

Do monitor internal vacuum before launch (less vacuum indicates a leak; positive pressure may indicate dangerous gas accumulation)

Do simulate missions before launch

Do test Iridium and ARGOS telemetry before launch

.mi and .ma files

Default Webb Ashumet missions below, insert text of actual missions and ma files here if desired. Highlighted in yellow are sensors and arguments commonly changed by users.

```
# glmpc.mi
#
#  Retrieves waypoints from mafiles/goto_l10.ma (which is GLMPC generated)
#  Retrieves envelope  from     mafiles/yo10.ma
#  Retrieves climb to surface controls from mafiles/surfac10.ma
#
#  Surfaces:
#     if haven't had comms for an hour
#     mission done (finished all the waypoints)
#     Every waypoint
#     bad altimeter hit (yo finishes)
#     If requested by science
#
# All science sensors sample on only downcast
#
# 24-May-05 hfargher@DinkumSoftware.com Initial (based on gylov001.mi)
#


behavior: abend
   b_arg: overdepth_sample_time(s)       10.0 # how often to check

                                   # MS_ABORT_OVERTIME
   b_arg: overtime(s)                  -1.0 # < 0 disables

   b_arg: samedepth_for_sample_time(s)   30.0 # how often to check

   b_arg: max_wpt_distance(m)         3000 # MS_ABORT_WPT_TOOFAR
                                  # Maximum allowable distance to a waypoint
                                  # < 0 disables

# Come up if haven't had comms for a while, 20 minutes
behavior: surface
   b_arg: args_from_file(enum) 10 # read from mafiles/surfac10.ma
   b_arg: start_when(enum) 12     # BAW_NOCOMM_SECS 12, when have not had comms for WHEN_SECS secs
   b_arg: when_secs(sec)   1200   # 20 min, How long between surfacing, only if start_when==6,9, or 12
   b_arg: end_action(enum) 1      # 0-quit, 1 wait for ^C quit/resume, 2 resume, 3 drift til "end_wpt_dist"
   b_arg: keystroke_wait_time(sec) 300   # how long to wait for control-C



   # Come up when mission done
   # This is determined by no one steering in x-y plane (no waypoints)
behavior: surface
   b_arg: args_from_file(enum) 10 # read from mafiles/surfac10.ma
   b_arg: start_when(enum)   3   # 0-immediately, 1-stack idle 2-pitch idle 3-heading idle
                          # 6-when_secs, 7-when_wpt_dist
   b_arg: end_action(enum)  0     # 0-quit, 1 wait for ^C quit/resume, 2 resume
   b_arg: gps_wait_time(s) 300    # how long to wait for gps
   b_arg: keystroke_wait_time(s) 180   # how long to wait for control-C


   # Come up briefly if "yo" finishes
   # This happens if a bad altimeter hit causes a dive and climb to
   # complete in same cycle.  We surface and hopefully yo restarts
behavior: surface
   b_arg: args_from_file(enum) 10 # read from mafiles/surfac10.ma
   b_arg: start_when(enum)   2   # 0-immediately, 1-stack idle 2-pitch idle 3-heading idle
                          # 6-when_secs, 7-when_wpt_dist
   b_arg: end_action(enum)   1     # 0-quit, 1 wait for ^C quit/resume, 2 resume
   b_arg: gps_wait_time(s) 300    # how long to wait for gps
   b_arg: keystroke_wait_time(s) 15   # how long to wait for control-C
```

```
   # Come up every way point
behavior: surface
   b_arg: args_from_file(enum) 10 # read from mafiles/surfac10.ma
   b_arg: start_when(enum)    8   # 0-immediately, 1-stack idle 2-depth idle 6-when_secs
                         # 7-when_wpt_dist 8-when hit waypoint 9-every when_secs
   b_arg: when_wpt_dist(m)  10   # how close to waypoint before surface,

   b_arg: end_action(enum)   1     # 0-quit, 1 wait for ^C quit/resume, 2 resume
   # b_arg: report_all(bool)   0     # T->report all sensors once, F->just gps
   b_arg: gps_wait_time(s) 300    # how long to wait for gps
   b_arg: keystroke_wait_time(s) 300   # how long to wait for control-C


   # Come up when requested by science
behavior: surface
   b_arg: args_from_file(enum) 10 # read from mafiles/surfac10.ma
   b_arg: start_when(enum)     11 # BAW_SCI_SURFACE
   b_arg: end_action(enum)   1     # 0-quit, 1 wait for ^C quit/resume, 2 resume
   b_arg: report_all(bool)   0     # T->report all sensors once, F->just gps
   b_arg: gps_wait_time(s) 300    # how long to wait for gps
   b_arg: keystroke_wait_time(s) 300   # how long to wait for control-C


   # Come up every 10 minutes
#behavior: surface
#    b_arg: args_from_file(enum) 10 # read from mafiles/surfac10.ma
#    b_arg: start_when(enum)    9   # 0-immediately, 1-stack idle 2-depth idle 6-when_secs
#                           # 7-when_wpt_dist 8-when hit waypoint 9-every when_secs
#    b_arg: when_secs(s)     600   # How long between surfacing, only if start_when==6 or 9
#
#    b_arg: end_action(enum)   1     # 0-quit, 1 wait for ^C quit/resume, 2 resume
#    b_arg: report_all(bool)   0     # T->report all sensors once, F->just gps
#    b_arg: gps_wait_time(s) 300    # how long to wait for gps
#    b_arg: keystroke_wait_time(s) 300   # how long to wait for control-C



behavior: goto_list
   b_arg: args_from_file(enum) 10   # read from mafiles/goto_l10.ma
   b_arg: start_when(enum)      0   # 0-immediately, 1-stack idle 2-heading idle


behavior: yo
   b_arg: args_from_file(enum) 10   # read from mafiles/yo10.ma
   b_arg: start_when(enum) 2       #  0-immediately, 1-stack idle 2-depth idle
   b_arg: end_action(enum) 2       # 0-quit, 2 resume


  # Sample all science sensors only on downcast
behavior: sample
   b_arg: intersample_time(s)              0  # if < 0 then off, if = 0 then

behavior: prepare_to_dive
   b_arg: start_when(enum) 0     # 0-immediately, 1-stack idle 2-depth idle
   b_arg: wait_time(s)   720     # 12 minutes, how long to wait for gps

behavior: sensors_in       # Turn most input sensors off
```

goto10.ma

```
behavior_name=goto_list
#  Written by gen-goto-list-ma ver 1.0 on GMT:Tue Feb 19 18:56:54 2002
#  07-Aug-02 tc@DinkumSoftware.com Manually edited for spawars 7aug02 op in buzzards bay
#  07-Aug-02 tc@DinkumSoftware.com Changed from decimal degrees to degrees, minutes, decimal minutes
#  ??-Apr-03 kniewiad@webbresearch.com changed to ashument
#  17-Apr-03 tc@DinkumSoftware.com fixed comments
# goto_l10.ma
# Flies the box in ashumet
# Each leg about 200m

<start:b_arg>
b_arg: num_legs_to_run(nodim) -1 # loop
b_arg: start_when(enum) 0 # BAW_IMMEDIATELY
b_arg: list_stop_when(enum) 7 # BAW_WHEN_WPT_DIST
b_arg: initial_wpt(enum) -2 # closest
b_arg: num_waypoints(nodim) 4
<end:b_arg>
<start:waypoints>
-7032.0640   4138.1060
-7031.9200   4138.1090
-7031.9170   4138.0000
-7032.0610   4137.9980
<end:waypoints>
```

surfac10.a

```
behavior_name=surface
# surface-20deg.ma
# climb to surface with ballast pump full out
#                    pitch servo'ed to 26 degrees
# Hand Written
# 08-Apr-02 tc@DinkumSoftware.com Initial
# 01-Feb-03 tc@DinkumSoftware.com Renamed surfac20.ma
# 03-Mar-03 kniewiad@webbresearch.com Renamed surfac30.ma for Buzzards Bay Trials
# 09-Apr-03 kniewiad@webbresearch.com Adjusted for Ashumet. Pitch to 26 deg


<start:b_arg>
   # arguments for climb_to
   b_arg: c_use_bpump(enum)      2
   b_arg: c_bpump_value(X)  1000.0

   b_arg: c_use_pitch(enum)      3   # 1:battpos  2:setonce  3:servo
                                   #   in        rad        rad, >0 climb
   b_arg: c_pitch_value(X)     0.4528     # 26 deg

<end:b_arg>
```

yo10ma.

```
behavior_name=yo
# yo10.ma
# climb 3m   dive 12m alt 9m pitch 26 deg
# Hand Written
# 18-Feb-02 tc@DinkumSoftware.com Initial
# 13-Mar-02 tc@DinkumSoftware.com Bug fix, end_action from quit(0) to resume(2)
# 09-Apr-03 kniewiad@webbresearch.com Adjusted for Ashumet
<start:b_arg>
   b_arg: start_when(enum)      2   # pitch idle (see doco below)
   b_arg: num_half_cycles_to_do(nodim) -1   # Number of dive/climbs to perform
                                  # <0 is infinite, i.e. never finishes
   # arguments for dive_to
   b_arg: d_target_depth(m)     12
   b_arg: d_target_altitude(m)   3
   b_arg: d_use_pitch(enum)      3   # 1:battpos  2:setonce  3:servo
                                   #   in        rad        rad, <0 dive
   b_arg: d_pitch_value(X)   -0.4528     # -26 deg

   # arguments for climb_to
   b_arg: c_target_depth(m)      3
```

```
   b_arg: c_target_altitude(m)  -1
   b_arg: c_use_pitch(enum)      3   # 1:battpos  2:setonce  3:servo
                              #   in        rad       rad, >0 climb
   b_arg: c_pitch_value(X)     0.4538     # 26 deg
   b_arg: end_action(enum) 2     # 0-quit, 2 resume
<end:b_arg>

# NOTE: These are symbolically defined beh_args.h
# b_arg: START_WHEN     When the behavior should start, i.e. go from UNITIALIZED to ACTIVE
#    BAW_IMMEDIATELY    0   // immediately
#    BAW_STK_IDLE       1   // When stack is idle (nothing is being commanded)
#    BAW_PITCH_IDLE     2   // When pitch is idle(nothing is being commanded)
#    BAW_HEADING_IDLE   3   // When heading is idle(nothing is being commanded)
#    BAW_UPDWN_IDLE     4   // When bpump/threng is idle(nothing is being commanded)
#    BAW_NEVER          5   // Never stop
#    BAW_WHEN_SECS      6   // After behavior arg "when_secs", from prior END if cycling
#    BAW_WHEN_WPT_DIST  7   // When sensor(m_dist_to_wpt) < behavior arg "when_wpt_dist"
#    BAW_WHEN_HIT_WAYPOINT 8 // When X_HIT_A_WAYPOINT is set by goto_wpt behavior
#    BAW_EVERY_SECS     9   // After behavior arg "when_secs", from prior START if cycling
#    BAW_EVERY_SECS_UPDWN_IDLE 10  // After behavior arg "when_secs", from prior START AND
#                            //       updown is idle, no one commanding vertical motion
#    BAW_SCI_SURFACE    11  // SCI_WANTS_SURFACE is non-zero
#    BAW_NOCOMM_SECS    12  // when have not had comms for WHEN_SECS secs
# b_arg: STOP_WHEN
#   0   complete
#   1-N same as "start_when"
```