

QUEphone manual (Rev. 170822)

Table of Contents

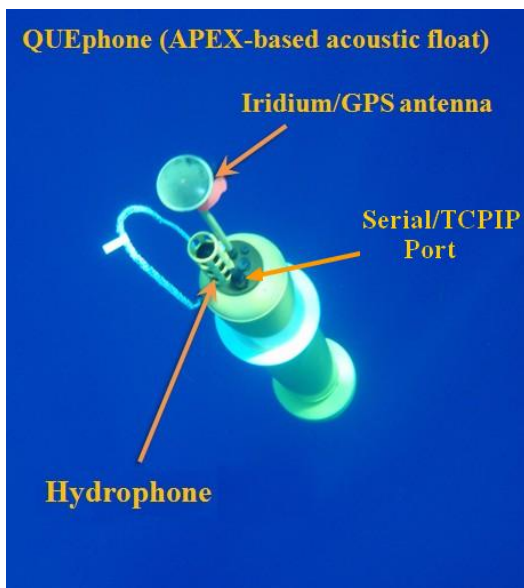
<i>I. Introduction</i>	2
<i>II. Principle</i>	3
<i>III. QUEphone operating manual (Short version)</i>	8
1. Deployment procedure.....	8
2. Recovery procedure	9
3. QUEphone local commands	10
4. QUEphone remote commands via NOAA/PMEL Iridium server	12
5. NOAA/PMEL Iridium/Rudics server	15
<i>IV. Preparing the QUEphone</i>	16
1. APEX float.....	16
a. Bench testing.....	16
b. Example of actual mission parameters	18
2. WISPR board (EOS, WA)	19
3. Getting ready.....	19
<i>V. System</i>	22
1. Apf9 hardware modifications	22
2. PAM system (WISPR DSP board and pre-amp system)	24
a. WISPR board mounted inside the QUEphone below the endcap	24
b. Pre-amp and hydrophone.....	27
3. WISPR program commands (ver 2).....	29
4. Example of start file on CF card	30
5. Communicating with QUEphone.....	31
a. With Apf9 board	31
b. With WISPR board.....	32
<i>VI. A day before deployment</i>	35
<i>VII. Programming apf9 (motherboard of APEX float)</i>	36
1. Your PC	36
2. Programming Apf9 board	37
3. Hardware setup	37
<i>VIII. Example of QUEphone record received via satellite (bench test)</i>	38
<i>IX. Frequently asked question (FAQ)</i>	45
<i>X. MissionTime.m (Matlab code)</i>	48

1. Introduction

A profiler float has been a tool of choice for ocean climate studies since the late 2000s. A single standard profiler floats in its life can repeat hundreds of descent and ascent cycles between the surface and the parking depth up to 2000 m by adjusting its buoyancy. Although it drifts with ocean current, it is relatively inexpensive as compared to another buoyancy-driven platform, underwater glider. As of today (Aug, 2017), ~3700 floats are profiling in the global ocean reporting the water column data to the land stations. For a remotely operated acoustic platform, it is quiet, simple and inexpensive to operate and suitable for a short-term acoustic monitoring in a small area.

With a support from the ONR marine mammal program, Oregon State University (OSU) has successfully converted the APEX floats from Teledyne Webb Research (MA) to passive acoustic profiler platforms. It is called QUEphone. With a QUEphone, the standard CTD sensor was replaced by a hydrophone and a depth sensor and by integrating the OSU-developed PAM (Passive Acoustic Monitoring) module called WISPR, which became available from EOS, WA. It is capable of recording the sound signal for an extended period at 125-kHz sampling rate while running a sophisticated detection algorithm for marine mammal calls such as beak whale clicks. Running an acoustic monitoring mission, buoyancy control, time management, telemetry, and operating the WISPR are all controlled by the energy-efficient Apf9 processor of APEX float.

To control the float remotely, a user sends the commands same as the Teledyne Webb APEX in addition to the acoustic commands to the WISPR. The standard unit comes with the alkaline battery which lasts approximately for 20 days and with a high power lithium battery it runs approximately for 45 days while running once-a-day descent-park-ascent-telemetry cycles. The acoustic data are stored in a 512 GB CF card, which lasts approximately 1.5 months of continuous recording. The instrument is recovered at the end of a mission in order to analyze the acoustic data stored in the memory.



Picture 1. QUEphone: an acoustic platform for high frequency marine animals.

This manual describes a principle and procedures of QUEphone operation including deployment and recovery and remote operation using NOAA PMEL Iridium Rudics web site. Pre-launch preparation in the lab, local and remote commands, data file structure, the web interface for satellite communication are described. It is recommended for novice users to familiarize themselves first to the APEX float manuals prepared by Teledyne Webb and the University of Washington. Both explain mission parameters and standard parameter setups, and local and remote commands in detail. Most users can skip the programming procedure (chapter VI of this manual).

II. Principle

Figure 1 shows a sample mission of the QUEphone. It repeats a typical telemetry-descent-park-ascent-telemetry cycle for three consecutive days while surfacing at pre-programmed time of day (ToD) for telemetry once a day. There is another phase called deep-profile, which makes a short profile from the park-depth to deeper water. This feature is not used for acoustic mission and is not discussed here. Users are encouraged to read the APEX manual from Webb and experiment different parameters by themselves.

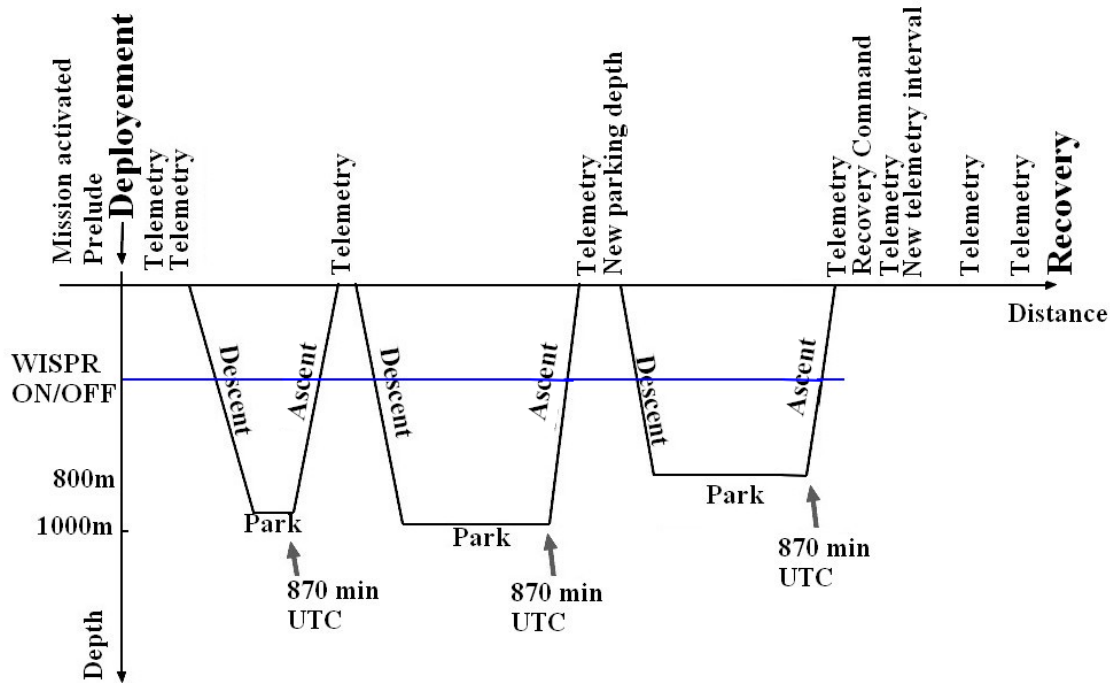


Figure 1. A typical QUEphone mission and operational phases

OSU's QUEphone adopted web-based a two-way telemetry protocol developed by the NOAA/PMEL Rudics server which handles hundreds of offshore NOAA buoy telemetries reliably every day. Be aware that file uploading from the floats and sending commands to the float are handled differently from the standard Teledyne Web protocols. However operating principle, including which data files are to be uploaded and all the commands used locally or remotely via satellite are the same as the APEX float with a few exceptions.

In the field, the QUEphone can be activated locally either via a serial COM port (requires PC) or by swiping a magnet. If it is properly prepared, during the prelude phase, there are two telemetries to NOAA Rudics Iridium server before starting to descend. The subsequent dive is a test dive. After telemetry it starts a regular once-a-day dive-ascent mission. At the surface, it sends two files: message file and engineering log file. The log file contains temp-depth (CTD data without conductivity), GPS, engineering data. The engineering data contains system engineering status as well as detection results if there is any. Due to a limited bandwidth of Iridium (2400 bps), it does not send raw acoustic data.

On the land, at Rudics server site all files received are concatenated into one file, if the data transmission is completed in one connection. If the telemetry is interrupted because of the poor satellite connection, multiple files will be created, and it is up to the remote operator to edit and concatenate proper sections to make one file.

In the following example, a simple 0-1000 m profile mission list is explained. This mission keeps the QUEphone at 1000 m (parking depth) near the sound channel for ~17 hours per day where the noise interference from the surface is low and acoustic detections more likely occur than any other depth because of low propagation loss.

You can get this list by connecting a 6-pin AG206-6 connector at the top QUEphone end cap (Picture 4) and typing “L” (not case sensitive). It requires a 2-wire SAIL-to-serial RS232 converter. Teledyne provides the converter which needs an external +12-V. Baud rate is 9600.

The first line is the program version followed by APEX serial number.

User status G3F999999P0450 means that it sets gain = 3 and with maximum detections of 99999 and turn-on/off threshold depth of WISPR at 450 m. In each descent, at a user-specified threshold depth (450 m), the WISPR power is turned on and acoustic monitoring begins. Likewise, during ascents (referred to as profile in the APEX definition) the WISPR is turned off when it crosses the threshold depth.

Pwd is the QUEphone ID followed by a primary and an alternate Iridium phone number of the Rudics server.

In this example, it is operated with ToD mode (Time of Day). Every day at 870 min UTC (14:30 UTC or 6:30 PST), it ends parking mode at 1000 m and starts ascending. It should not be confused with the time that QUEphone surfaces to transmit files. Since the descent and ascent speeds of APEX float are constant and ~0.08m/s, it takes approximately 3.5 hours to make a 1000-m vertical descent or ascent. 3.5 hours after the ToD=870 means that at ~18:00 (UTC) or ~10:00 (PST) it surfaces and starts sending one-day log files, which include GPS positions, engineering, temp-depth and detection results, via Iridium satellite. The ToD profiling operation with 1000-m parking depth, therefore maximizes the chance of acoustic detections by staying at the sound channel depth for a long period (~17 hours) while minimizing current drift.

At the surface, it not only transmit the data files but also receive commands from land at the end of each file upload. **It is important to know that commands must be submitted to the Rudics server before the QUEphone surfaces.** For example, on the 2-nd day, it receives new commands from the land to change the parking depth to 800 m. On the 3-rd day, it receives a recovery command (ActivateRecoveryMode()) from the land. As a result, it suspends the mission and stays at the surface until it is picked up by a field operator or the mission is reactivated remotely. During recovery mode, the QUEphone repeats transmissions of its GPS positions at a fixed telemetry interval, which can be changed by sending a new command (TelemetryRetry(15)) to shorten the telemetry interval, e.g., 15 min to help the field crew to find the QUEphone by providing more up-to-date positions.

The remote operator should examine the data files uploaded every day. As the mission progresses, the operator must pay attention to the battery voltage level which may look like the followings:

```
(Jan 22 2015 18:08:54,367sec)AirSystem()BatV[185cnt,14.3V] Amp[72cnt,29.0mA]
BrmtrP[130cnt,7.7"Hg]Run-Tm[4s]
```

If the battery voltage becomes too low, the QUEphone may not come back to the surface or unable to transmit the data. Critical voltage is ~9 V.

The remote operator also needs to pay attention to free data space. Look for a message after “DSP ON”. The following line is the data storage info.

```
(Jan 22 2015 00:32:28, 170 sec) MissionControlAgent Data storage available 99.40
percent
```

As the mission continues, data storage space becomes critically low or battery voltage falls below 10 V and you may want to stop logging in all subsequent profiles. During normal hibernation mode, APEX’s power consumption is ~ 15 mW. Whereas WISPR requires approximately 1 Watts. So you may want to stop data logging to save the battery power for the recovery operation or extend the mission for several days without using battery too much until the weather becomes suitable for a small-boat recovery operation. To stop logging in the subsequent profiles, send the following command

```
User(G1D999999P5000) [CRC]
```

The number after P specifies the depth in meter at which WISPR’s power is on/off. In this case, it sets the WISPR’s turn on/off depth to 5000 m. CRC is the cyclic redundancy check generated by CRC-CCITT (0x1D0F) <https://www.lammertbies.nl/comm/info/crc-calculation.html>. Since APEX float’s maximum operating depth is 2000 m, it never turns on the WISPR’s power for the rest of mission.

If beaked whale detections occur, the QUEphone data at the NOAA/PMEL Rudics web site may look like the followings:

```
(Jan 22 2015 18:27:49, 906 sec) QuecomGetDTX() Ndtx=5,Nclk=928,Acm=8,MTK=0,MTHR=85,
MRT=2416,MICI=0.100, P=520.7dB
```

Where Ndtx is number of detections since the last detection, Nclk is a number of clicks, Acm is accumulated detections, MTK is mean Teager-Keiser, MTHR is mean threshold, MRT is mean ratio, MICI is mean inter clicking interval in sec and P is the depth that detections occurred.

APEX float operating parameters listing

```
APEX version 100305b  sn 4693          (version num- date, sn = APEX hull number)
User: G3D999999P0450          (Gain=3, Max detec=999999, WISPR On at 450m)
Pwd:  Q003                      (QUEphone ID)
Pri:  ATDTxxxxxxxxxxxxx          Mhp (Rudics phone #)*
Alt:  ATDTxxxxxxxxxxxxx          Mha
      0870 ToD for down-time expiration. (Minutes) Mtc
      00370 Down time. (Minutes) Mtd (>descent+2hours)
      00480 Up time. (Minutes) Mtu (>ascent+2hours)
      00280 Ascent time-out. (Minutes) Mta (>ascent+1hours)
      00000 Deep-profile descent time. (Minutes) Mtj (we don't use)
      00230 Park descent time. (Minutes) Mtk (<1.5*descent+1hour)
      00015 Mission prelude. (Minutes) Mtp
      00015 Telemetry retry interval. (Minutes) Mhr
      00060 Host-connect time-out. (Seconds) Mht
      0 Continuous profile activation. (Decibars) Mc
```

```

1000 Park pressure. (Decibars)           Mk  (~m)
1000 Deep-profile pressure. (Decibars)    Mj  (parking depth ~ m)
034 Park piston position. (Counts)       Mbp (call Teledyne)
000 Compensator hyper-retraction. (Counts) Mbh (don't change)
016 Deep-profile piston position. (Counts) Mbj (na)
010 Ascent buoyancy nudge. (Counts)      Mbn (don't change)
022 Initial buoyancy nudge. (Counts)     Mbi (don't change)
254 Park-n-profile cycle length.         Mn  (don't change)
124 Maximum air bladder pressure. (Counts) Mfb (don't change)
096 OK vacuum threshold. (Counts)        Mfv (consult Teledyne)
226 Piston full extension. (Counts)      Mff (don't change)
016 Piston storage position. (Counts)     Mfs (don't change)
    2 Logging verbosity. [0-5]           D   (keep it 2)
0002 DebugBits.                          D
55ad Mission signature (hex)
*Rudics phone number is masked intentionally.

```

The listing above is the serial output of APEX float processor, Apf9, when you type “L”. To access the processor, you must first connect the SAIL (RS232 Current Loop interface) with 6-pin Impulse connector. Hitting “enter” key to wake up QUEphone. Typing ‘L’ gets this list.

These parameters follow the strict rules set of the Webb APEX float mission agent. They are dictated primarily by descent and ascent speeds (0.08 m/s) and parking depth with large safety margins for a successful operation. Before each mission begins, users must validate on the bench by typing ‘z’ (in mission mode) for sanity check. If it does not pass the sanity check, the float is not deployable (would not start the mission). Remotely changing these parameters, therefore could be dangerous. If the new settings sent remotely and do not pass the self-sanity check, it does not take any changes or in worst case, it may become inoperable. Therefore, it is recommended to test the new set of mission parameters on another float in the lab or Matlab simulator (MissionTime.m in X).

Park piston position is not only dependent on the depth but also the weight of the housing. Each housing is slightly differently made, and salinity and temperature of the water is also different for each site, users need to contact Teledyne Webb for the optimum piston position and weight of the float by giving them the 1) serial number (i.e., sn 4693, which is also written on the case or internal cage) and 2) the parking depth and vertical temperature-salinity profile of the area which is season dependent.

During the mission if you change the parking depth by remote command (ParkPressure), it requires for the remote operator to send up to 8 new parameters including ParkPresssure(), DeepProfilePressure(), ParkPistonPos(), UpTime(), DownTime(), ParkDescentTime(), DeepProfilePressure() and TimeOfDay(). These mission time-related parameters have to meet the strict APEX float mission rules as described earlier. See 3.1.2. Mission insane in the Iridium Apex manual (UW version 1.3.2.2) page 13 and 15 for details.

General rule of thumb for the piston position P_s is

$$P_s \approx P_p + \nabla d / 12.5 \quad (1)$$

where P_p is the initial park piston position given by Teledyne Webb and ∇d is the pressure difference in meters from the initial parking depth ($\nabla d =$ initial parking depth -

new parking depth). The lower piston position is the float becomes heavier, therefore dives deeper. The higher piston position becomes, it becomes lighter in water and dives shallower depth. For example, if the initial parking depth is 800 m and piston position is 96, in order to increase the depth to 1000 m, $\nabla d = -200$ and the new piston position is ~ 80 . In other words, 1 piston position is equivalent to the depth change of 12.5 m.

The float depth versus weight is 20 m/gram. For example, if you make the float weight 10 grams heavier than the Webb recommended weight, it goes 200 m deeper than your intended parking depth.

$$D_w = D_p + W * 20 \quad (2)$$

Where D_w is the actual parking depth, D_p is your intended depth, and W is the excess weight.

If your scale is not too accurate, the float may park at a depth slightly off from the target depth. However, the float is smart enough to adjust the piston position by itself a little by little, so it eventually settles at the designated parking depth as long as the depth does not exceed 2000 m or it hits the bottom.

As the mission progresses, the remote operator must keep an eye on the seafloor depth. If the bottom depth becomes shallower, adjustment of the parking depth and associated parameters may become necessary. If the seafloor becomes shallower than the parking depth, the QUEphone does not reach the intended depth, and when the “park descent time” expires (remote command `ParkDescentTime`), it aborts the mission and ascends back to surface. You can increase `ParkDescentTime` to a large number to avoid the mission abort.

For more details, see “Iridium Apex Manual” by Dana Swift, University of Washington.

III. QUEphone operating manual (Short version)

1. Deployment procedure

Preparation 1: Near the site on land within 24 hours before the deployment

Move the QUEphone on the deck with clear sky view for satellite com.

1. Connect the SAIL (RS232 Current Loop interface) with 6-pin Impulse connector. Hit “enter” key to wake up QUEphone. It responds with > prompt. (SAIL needs 12-V external power or battery! There are two clamp connectors: one to the blue cable and the other to the Zinc anode on the end cap.)
2. Type ‘ga’ to download a new GPS almanac. It takes 15 minutes.
3. Type ‘gc’ to configure the GPS.
4. Type ‘gt’ to synchronize the APEX float time to GPS.
5. Type ‘id’ to display the piston position. If it is not 226, type ‘ig 226’ to move the piston. You will hear a faint motor sound. It may take as long as 15 min.
6. Type ‘L’ to display the mission parameters. Make sure they are consistent with the logging sheet.
7. Type ‘st’ to check the ambient pressure and temp (near 0 dB and room temp)
8. Type ‘c’ to check the battery voltage (14.5 to 15.5 V) and internal pressure (-6 to -7.2 dB)
9. Type ‘q’ to put the QUEphone hibernate mode.
10. Put the dummy plug back.

Preparation 2: At the site 15 minutes before deployment

Move the QUEphone on the deck with clear sky view for satellite com.

Method 1 (if you have satellite phone or in cell-phone range and no laptop)

1. Swipe the magnet. You should hear motor sound. Preferably piston position is set to 226 on land so that you do not have to wait too long.
2. Notify the remote operator that magnet is swiped. They can monitor the QUEphone status through Rudics web site on their end. If the remote operator sees status files uploaded and all OK, they notify the field crew to give a go-ahead.
3. If no file upload takes place, there is a critical problem with Iridium. The land operator will contact you. **Abort the deployment.**
4. When Iridium transmission is confirmed, deploy. Pick up the instrument by the nylon loop.
5. 2-nd Iridium file upload completed. Stay near the QUEphone until start diving. The float sends the last file (including GPS pos) and in the next 10 minutes, it should submerge completely.

Method 2 (no remote operator’s support but you have a laptop. This is the preferred way.)

6. On the deck connect the 6-pin connector on the end cap. Open serial communication program of your laptop. On com terminal screen, hit return to wake up the QUEphone. Screen display looks like:
Asynchronous wake-up detected (i.e., wake-up not initiated by alarm signal).
Entering Command Mode [ApfId 4694, FwRev 150115]
7. Type “e” to execute
8. It should start self-test and display actions taking place inside the QUEphone
9. QUEphone must have a clear sky view to get the GPS and Iridium connection for the next 20 min. You have a 20-min window to deploy
10. **If the first telemetry is not successful, abort the deployment.**
11. **If the telemetry is successful, proceed deployment.**
12. Unplug the serial cable and **put the dummy plug back on the end cap.**
13. In the next 20 min, QUEphone will execute the mission in the following sequences:
0 min: Mission activated
2 min: starts searching GPS

3 min: 1-st GPS position & status file upload via Iridium to the Rudics site

If the first telemetry is not successful, abort the deployment.

Idle for 9 min. **This is the best deployment time window!!!**

16 min: starts 2-nd GPS & Iridium TX

20 min: ends 2-nd Iridium TX

QUEphone start moving the piston and losing buoyancy.

14. In 10 min it submerges completely.

2. Recovery procedure

1. **Remote operator:** 1 to 23 hours before the recovery, paste ActivateRecoveryMode() [0x9849] and two Senddata() [0xC9C9] in the next_CALL commands list to the NOAA/PMEL Rudics web site.
2. **Remote operator:** Waits for the QUEphone float to surfaces and a new file is uploaded in the Dial-in log. Once the file is up, look for the **latest** GPS position in the file, which may look like

```
# GPS fix obtained in 40 seconds.
#      lon      lat mm/dd/yyyy hhmmss nsat
Fix: -124.0439  44.6208 01/22/2015 202600   4
```

The file may contain two GPS positions. Notify the field party the latest GPS position with time stamp via an e-mail or SMS.

Once QUEphone is at the surface, it sends GPS location every 15 min (default).

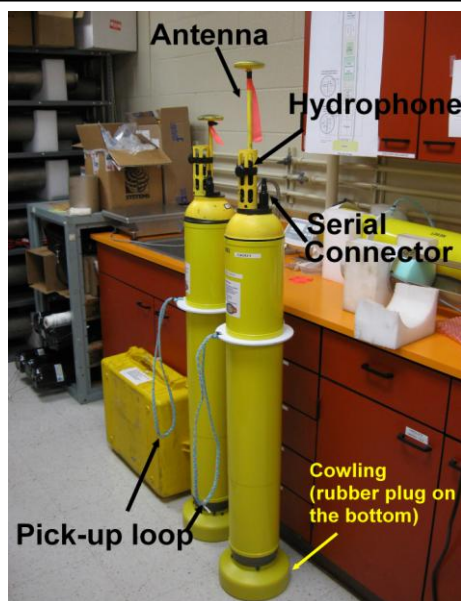
Remote operator can send TelemetryRetry(5) [0x09C5] to make the interval 5 min (if there is enough battery power left). Keep sending the newest GPS position to the field party. If the field party is still far, and it takes several hours to reach the recovery position, send TelemetryRetry(60) [0x01D9] to make the interval longer to save the battery. Note that there is ~5 min time lag between actual transmission and the file is posted on the Rudics site.

3. **Field Party:** Search for the QUEphone. To pick up the float, grab by the polypro loop (**Do not grab antenna or hydrophone**). Preferably to be picked up by RHIB or low-draft boat weather permitting. If it is from a low-draft ship, a long pole with a hook (with rope on one end) would do.
4. After recovery, secure the QUEphone with the upright position where it has a clear view of the sky for GPS and Iridium communication.
5. Let the land operator know that the QUEphone is recovered.
6. **Remote operator** can send TelemetryRetry(360) [0xCB64]. On the next telemetry, it extends the telemetry cycle 6 hours.
7. **Field Party:** Once on land or at a secure location, connect the serial connector to the QUEphone and hit return. It may not respond if it is in the middle of GPS or Iridium telemetry. Wait until it finishes.
8. When it is ready to take a command, it should respond with
(Dec 15 2010 19:59:10, 8437 sec) Apf9Init() Asynchronous wake-up detected
(ie., wake-up not initiated by alarm signal).
Entering Command Mode [ApfId 4694, FwRev 101215]
>
8. When you see a prompt >, type **k** to kill the mission.
It responds as

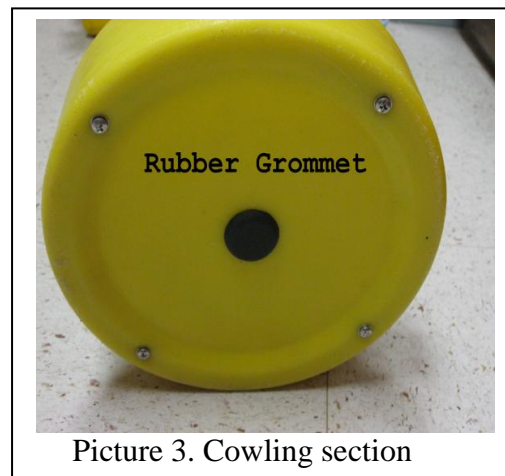
```
WARNING: Deactivating the mission renders the float nondeployable
until the mission is reactivated. Kill the mission anyway?
```

9. Type 'Y' and carriage return to kill the mission.
WARNING: Mission deactivated; float is not deployable until the mission is reactivated.

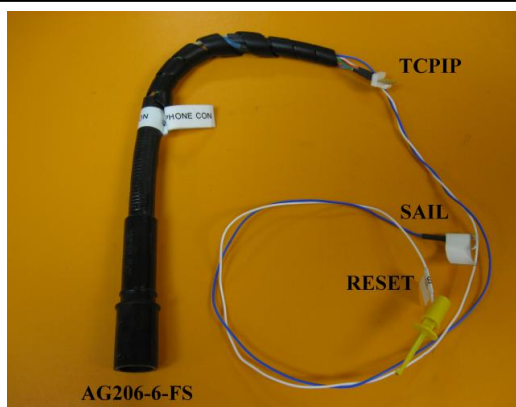
10. Move the piston to 110 for shipping by typing a command **“ig 110”**. It takes about 10 min.
11. When the motor stops, type **“q”** to quit. This makes the QUEphone low-power hibernate mode. Put the dummy plug back. Wash with fresh water. Remove the rubber grommet on the bottom and wash well inside the cowling. Be careful not to poke inside with a sharp object such as screw driver. Wash top section where the pressure sensor and hydrophone are. Dry it for shipping.
12. Antenna and hydrophone are the weakest parts, and can easily break off at the stem. For shipping, support the antenna with polyurethane foam.



Picture 2. QUEPhone



Picture 3. Cowling section



Picture 4. AG206-6-pin connector for apf9 and WISPR communication



Picture 5. SAIL (current-loop RS232) interface. You need a 12 V battery.

3. QUEphone local commands

To set the mission parameters before deployments, the operator must use SAIL (Picture 5) to communicate with apf9 processor board of the QUEphone. SAIL is a 2-wire current

loop serial communication converter for RS232. Connect the 6-pin AG206-6 connector on the end cap. One SAIL interface line should be connected to the SAIL signal (which is connected to the apf9 board inside, Picture 8), and the other is connected to the ground. There is no polarity. A 12-15 V battery or power supply is needed for the SAIL. The apf9 is the main controller board of the APEX float which stores the mission parameters and executes the mission by controlling the hydraulic motor, reading the pressure sensor, communicating Iridium/GPS and WISPR. QUEphone uses the same commands as the APEX to set all the mission parameters. In addition, QUEphone has own WISPR related commands (i* commands). Here are some commands often used in the lab. They are not case sensitive. For a complete list of APEX commands, users should refer to Iridium Apex Manual.

APEX main menu (not case sensitive)

C	Calibrate; Battery, current, & vacuum level (useful when pulling air out to vacuum)
D	Set verbosity level (typically 2) - e.g., d2 set verbosity 2
E	Execute (activate) mission -start a new mission
G	GPS module agent
K	Kill mission -Use it to end the mission
L	List mission parameters
M	Mission programming agent -see the APEX manual
N	Display float serial number
Q	Exit command mode - put it low-power sleep mode
R	Activate recovery mode -you really do not need this on the bench
S	Sensor module agent
T	Get RTC time

GSP menu

Ga	Upload almanac (do this before the mission near the deployment site)
Gt	Synchronize the apf9 clock with GPS
Gf	Get fix
Gc	Configure GPS almanac.
Gt	Synchronize APEC float clock to GPS.

Sensor menu

Sp	Read the external pressure and temp
----	-------------------------------------

FLASH file system

Jl	Display directory
Je	Erase all the files (file system has to be clean before deployment)

System menu

Id	Display the hydraulic piston position
Ig #	Go to specified position - make it full extension 226 before deployment
Il #	Max engineering file size in kB -5 to 63kB (set to 40 kB)

QUEphone specific commands

These are additional commands for QUEphone only to control the WISPR. There are more commands for APEX. See the APEX manual.

i*p	QUEphone ID (e.g., Q001)
i*u	User command to set gain, max detections/dive, and WISPR power on/off depth

User(G1D999999P0400) gain=1, D=999999, Depth=400m
i*o WISPR power ON (useful to test WISPR board)
i*x End the WISPR program (this is the only way to properly end the program)
i*k kill the WISPR power off (you must issue i*x first)

Cautions!

- 1) You may experience that apf9 board does not respond to your command when it is in “mission programming mode.” You need to exit out the mode by typing ‘q’.
- 2) During bench tests, do not pull the power off while WISPR is running logging program. By doing so, there is a risk of corrupting the file system of WISPR’s CF card. If the WISPR’s program is still running, a proper way to terminate the program is typing i*x. Apf9 sends a pulse to terminate the WISPR’s program, close all the files and unmount the file system. Wait for ~15 sec or until an Apf9 prompt appears, then type i*k to kill the power. After the WISPR’s logging program properly ends, you can remove CF card or unplug the WISPR power plug safely. The operating system of WISPR is uClinux. If the WISPR’s power is still on and logging program is not running, you can communicate with WISPR by either connecting the COM0 console port (115200 bps) on the WISPR (if the housing is open) or TCPIP (Picture 4). If you are on console port, type cd / to go back to the root and type umount /mnt. Type i*k at apf9 SAIL com, or push the “OFF button (red)” on the WISPR. After this you can pull the power off.
- 3) By unplugging the WISPR’s power off, you lose the real-time clock time. Be aware also that if you pull the apf9 power off, it loses the memory of the last GPS almanac. When Apf9 is repowered, it takes 15 minutes to download the recent GPS almanac from the antenna.

4. QUEphone remote commands via NOAA/PMEL Iridium server

Although the QUEphone uses the same commands as the APEX float, sending commands remotely via satellite is a quite different procedure from the standard APEX local command procedure. To send commands or receive data files to/from the QUEphone, OSU uses the NOAA/PMEL buoy data file/command interface web site. During the mission, the operator may decide to change a few mission parameters or ultimately to recover the QUEphone. They can do so by sending commands remotely through the NOAA/PMEL Rudics server. Users can paste the commands to QUEphone dial-out web site on the PMEL Rudics sever. Here is how the web interface page looks like. First, you must pick the appropriate QUEphone number (e.g., Q001).

http://intranet.pmel.noaa.gov/tempe_dialout/store_dialin_cmds/save_cmds.php



Figure 2. Choosing platform ID

By clicking “submit”, it takes you to another command page (Figure 3). The top table allows you to enter up to 10 commands. In this example, only the first command is valid, which is to activate the recovery mode (ActivateRecoveryMode() [0x9848]). Each command line consists of one command and the 4-character long hex CRC inside the square brackets which follows the rule of CRC-CCITT (0x1D0F). CRC of ActivateRecoveryMode() is 0x9848. QUEphone uses CRC to check for data transmission error. You can obtain the corresponding CRC by inserting the command in the window of the following web site.

<http://www.lammertbries.nl/comm/info/crc-calculation.html>

NEXT_CALL commands

NEXT_CALL commands will be sent to QUEHQ001 during the next successful dial-in connection.

NEXT_CALL commands will be cleared once they have been sent successfully during a dial-in call from QUEHQ001

Enter, Edit, or Clear up to 5 NEXT_CALL commands:

1:	ActivateRecoveryMode() [0x9848]
2:	Senddata() [0xC9C9]
3:	Senddata() [0xC9C9]
4:	
5:	
6:	
7:	
8:	
9:	
10:	

DEFAULT commands

DEFAULT commands will be sent if no NEXT_CALL commands are set.

DEFAULT commands will not be cleared and will be sent for any dial-in connections when no NEXT_CALL commands are set.

Enter, Edit, or Clear up to 5 DEFAULT commands:

1:	Senddata() [0xC9C9]
2:	Senddata() [0xC9C9]
3:	Senddata() [0xC9C9]
4:	
5:	
6:	
7:	
8:	
9:	
10:	

Figure 3. Recovery command

Bottom table is a default commands list. By default, QUEphone sends at least two files during a normal profile mission and only one during recovery mode. The file size is limited to 20 kB and total data size is 62 kB. QUEphone sends one file per one command. For example, in order to upload 3 data files during the telemetry session, the PMEL Rudics has to send 3 commands of either functional or non-functional kind. If QUEphone has no files to send, the extra Senddata() [0xC9C9] becomes non-functional dummy commands. If you have no specific command to send, then just fill in the rows with

multiple “Senddata() [0XC9C9]” commands. After each successful command download, PMEL Rudics server erases all the commands in the top table but not the command in the default list.

It is a useful feature since data uploading from the QUEphones may take place during night possibly at a most inconvenient time.

NEXT_CALL commands

NEXT_CALL commands will be sent to QUEHQ001 during the next successful dial-in connection.

NEXT_CALL commands will be cleared once they have been sent successfully during a dial-in call from QUEHQ001

Enter, Edit, or Clear up to 5 NEXT_CALL commands:

1:	DownTime(450) [0x220B]
2:	UpTime(500) [0x8155]
3:	AscentTimeOut(370) [0xE70F]
4:	ParkDescentTime(320) [0xD6E0]
5:	ParkPistonPos(72) [0x510B]
6:	ParkPressure(1200) [0x64F4]
7:	DeepProfilePressure(1200) [0xCDDD]
8:	TimeOfDay(770) [0x7C26]
9:	
10:	

DEFAULT commands

DEFAULT commands will be sent if no NEXT_CALL commands are set.

DEFAULT commands will not be cleared and will be sent for any dial-in connections when no NEXT_CALL commands are set.

Enter, Edit, or Clear up to 5 DEFAULT commands:

1:	Senddata() [0xC9C9]
2:	Senddata() [0xC9C9]
3:	Senddata() [0xC9C9]
4:	
5:	
6:	
7:	
8:	
9:	
10:	

E-mail address:

Save data received as:

Figure 4. Changing mission parameters and sending commands

The followings are some of the commands often used.

ActivateRecoveryMode() [0x9848] – Activate recovery. QUEphone stops mission and it stays at surface reporting status including GPS position. It also takes new commands.

TelemetryRetry(5) [0x09C5] –Telemetry interval of 5 min after the recovery is activated. It transmits status report including GPS position.

User(G2D999999P0650) [0x3508] - Set gain=2, max detection =9999999, WISPR’s turn on/off depth 650m.

ParkPressure(####) [CRC] –Park pressure depth in decibars (0-2000 decibars).

If you change the parking depth, you may have to change seven other mission parameters including AscentTimeOut, Uptime, ParkDescentTime, DownTime, TimeOfDay, ParkPistonPosition, and DeepProfilePressure. These parameters have to be tested using simulator or another APEX float in the lab. ParkPistonPosition also should be set to a new value, although it adjusts the piston position slowly by itself.

DeepProfilePressure(####) [CRC] – Has to be the same as ParkPressure.

ParkPistonPosition (###) [CRC] -Park piston position. Contact Teledyne Web for the appropriate piston position as well as the optimum weight for the mission. Users have to provide the vertical profile of Salinity and temp of the area and intended parking depth. Each float is made differently and varies by the sea water density. Users still need to do a minor adjustment to the park piston position. General rule of piston position = $P_i + \nabla d / 12.5$, where P_i is some initial piston position and ∇d is the pressure difference in meter. If the intended parking depth was 1000 m with recommended piston position of 24, but it parked at 1300 m, you need to adjust the piston by $300 / 12.5 = 16$. New piston position is $24 + 16 = 40$. Bear in mind that this is just an approximation.

AscentTimeout(####) [CRC]-Ascent time in min sufficient from parking depth to surface.
 $\geq D / 0.08 + 1 * \text{Hour}$ where D is parking depth.

UpTime(####) [CRC] $\geq \text{AscentTimeout} + 2 * \text{Hour}$

$D / 0.08 \leq \text{ParkDescentTime(####) [CRC]} \leq 1.5 * D / 0.08 + 1 * \text{Hour}$

DownTime(####) [CRC] $\geq \text{ParkDescentTime} + 2 * \text{Hour}$

TimeOfDay(####) [CRC] -Time of day for QUEphone to stop park and start ascending.
 #### is in min in GMT. Add the ascent time to estimate the surface time.

5. NOAA/PMEL Iridium/Rudics server

To see files received from QUEphone, go to the NOAA/PMEL Rudics web site
http://intranet.pmel.noaa.gov/rudics/ALL_RUDICS/

The web page may look like below. Click the date you wish to see (Figure 5).

Iridium/RUDICS Files				
• Dial-in Logs		2 Week Dial-In Call Summary		
01-24-15	01-23-15	01-22-15	01-21-15	01-20-15
01-19-15	01-18-15	01-17-15	01-16-15	01-15-15
01-14-15	01-13-15	01-12-15	01-11-15	01-10-15
01-09-15	01-08-15	01-07-15	01-06-15	01-05-15
01-04-15	01-03-15	01-02-15	01-01-15	12-31-14
12-30-14	12-29-14	12-28-14	12-27-14	12-26-14
12-25-14	12-24-14	12-23-14	12-22-14	12-21-14
12-20-14	12-19-14	12-18-14	12-17-14	12-16-14
12-15-14	12-14-14	12-13-14	12-12-14	12-11-14
12-10-14	12-09-14	12-08-14	12-07-14	12-06-14
12-05-14	12-04-14	12-03-14	12-02-14	12-01-14
• Dial-out Logs				

Figure 5. Rudics dial-in logs: pick the date you want to see

Look for QUEH Q####, and click “SEE CALL”. It contains details of configuration, mission, time, system status, detections, GPS, and Iridium telemetry results.

RUDICS Dial-In Log for 01-22-2015

[001]	00:02:05	FLEX 9999	2265 bytes received in	16 seconds	Good	SEE CALL	DATA FILE
[002]	00:04:43	QUEH Q003	5312 bytes received in	54 seconds	Good	SEE CALL	DATA FILE
[003]	00:09:00	QUEH Q003	5732 bytes received in	54 seconds	Good	SEE CALL	DATA FILE
[004]	00:16:03	DART 413P	440 bytes received in	6 seconds	Good	SEE CALL	DATA FILE
[005]	00:18:36	DART 430p	440 bytes received in	6 seconds	Good	SEE CALL	DATA FILE
[006]	00:19:06	DART 430S	607 bytes received in	9 seconds	Good	SEE CALL	DATA FILE
[007]	00:20:28	pCo2 0007	10488 bytes received in	42 seconds	Good	SEE CALL	DATA FILE
[008]	00:20:35	pco2 0017	10488 bytes received in	48 seconds	Good	SEE CALL	DATA FILE
[009]	00:20:57	pco2 0109	13368 bytes received in	137 seconds	Good, blocks re-sent	SEE CALL	DATA FILE
[010]	00:20:59	pco2 0154	17256 bytes received in	66 seconds	Good	SEE CALL	DATA FILE
[011]	00:21:51	pco2 0129	15832 bytes received in	62 seconds	Good	SEE CALL	DATA FILE
[012]	00:22:06	pco2 0112	15896 bytes received in	57 seconds	Good	SEE CALL	DATA FILE
[013]	00:23:16	pco2 0002	14611 bytes received in	54 seconds	Good	SEE CALL	DATA FILE

Figure 6. Rudics dial-in log with ID, date and time

An example of the QUEphone record is at the end of this manual (IX). Be aware that there is a delay of approximately 5 minutes after the Rudics receives a file and the file is posted on the web. As a result, the GPS position could be as much as ~10 minutes old. If you are recovering the QUEphone, it may drift off from the last reported position but should be within a few hundreds meters.

IV. Preparing the QUEphone

1. APEX float

The standard APEX float comes with four alkaline battery packs. However, because of the weight limitation, the battery pack 1 (900410-10) has to be removed. Only one type 2 and two 3 are used for QUEphone (total of four). All three types of alkaline battery packs are available from Teledyne Webb. The followings are the Alkaline battery part numbers and approximate weights of the battery packs (available from Teledyne Webb)

1) 900410-10 x 1 C-cell (~740 gr) - This pack has to be removed to make the float lighter to compensate for the weight of HTI92WB hydrophone.

2) 900409-10 x 1 D-cell (~1485 gr, 18 AHr ea)

3) 900408-10 x 2 D-cell (~1485 gr, 18 AHr ea)

Total amp hour 54 AHr (54 AHr 20 deg C, 45 AHr at 5 deg C)

The output voltage of each pack should be ~15-16 V. For profiling once a day between 0 and 1000 m, running WISPR continuously with three D-cell packs, the battery should last at least 14 days. For profiling 0-500m, it should last ~ 20 days.

Lithium battery option is possible. Lithium battery part number is

3PD1243 Rev B from Electrochem (950 gr ea. 4 DD cells.15.6 V, 40 AHr each)

3 Lithium battery packs and one Alkaline 900409 fit. Total 138 AHr.

For 0-1000 profiling, it lasts 35 days. For 0-500 m profiling, ~ 50 days.

a. Bench testing

The user can bench test the QUEphone in the lab and simulate a shallow-water dive by applying high pressure (such as Argon tank gas) to the pressure sensor located at the end cap. It requires an external Iridium/GPS antenna, Argon gas tank, copper tubes, a custom fitting for the pressure sensor, pressure regulators and gauges and valves to control the pressure. Normally the QUEphone's Iridium/GPS antenna cable is connected to the SMA connector of the antenna on the end cap, but for bench tests, it has to be connected to an

external roof-top antenna through a long cable. If you use external antenna such as APEX float antenna or Seaglider antenna (available from Sound Ocean Systems, WA) with low-loss cable, the antenna cable can be as long as 10 m. The high-pressure gas tube is attached to the pressure sensor to simulate the shallow water dives up to 100 m by manually controlling the pressure. WISPR's COM1 is connected to COM1 (9600 bps) of apf9. WISPR's COM0 is for PC to monitor WISPR's activity on the bench at 115200 bps (refer to Figure 9). Because of a small leak at the plastic pressure fitting, using high-pressure gas is not appropriate for a deep water simulation. We recommend a 50 m profiling.

Below is the basic mission parameter list for a bench test with a parking depth set at 50 m. Note that it takes a minimum of 2.5 hours even for a 50-m dive test to complete one mission including prelude, telemetry test, moving the piston to reach to the park piston position, stay at the parking depth for a minimum of 1 hour, moving the piston to surface and GPS/Iridium communication. These parameters include safety margins (dictated by the apf9 mission configurator) for descent and ascent time. Durations any shorter than these (i.e., shorter Down time, Up time, and Ascent time-out), the sanity check fails and it would not start a mission. See the details on how these parameters should be set in the APEX manual provided by Teledyne Webb.

```

APEX version 15010814  sn 4693 (version num-date time, sn = APEX hull number)
User: G1D099999P0020          (Gain=1, max detect=99999, WISPR on at 20m)
Pwd:  Q003                    (QUEphone ID)
Pri:  ATDTxxxxxxxxxxxxx      Mhp (Rudics phone #)
Alt:  ATDTxxxxxxxxxxxxx      Mha
INACTV ToD for down-time expiration. (Minutes)  Mtc
00132 Down time. (Minutes)          Mtd
00192 Up time. (Minutes)            Mtu
00072 Ascent time-out. (Minutes)     Mta
00000 Deep-profile descent time. (Minutes)  Mtj
00011 Park descent time. (Minutes)    Mtk
00005 Mission prelude. (Minutes)      Mtp
00005 Telemetry retry interval. (Minutes) Mhr
00060 Host-connect time-out. (Seconds) Mht
    0 Continuous profile activation. (Decibars) Mc
    50 Park pressure. (Decibars)       Mk (parking depth)
    50 Deep-profile pressure. (Decibars) Mj (not used)
210 Park piston position. (Counts)     Mbp (arbitrary #)
000 Compensator hyper-retraction. (Counts) Mbh
016 Deep-profile piston position. (Counts) Mbj
010 Ascent buoyancy nudge. (Counts)    Mbn
022 Initial buoyancy nudge. (Counts)    Mbi
254 Park-n-profile cycle length.       Mn
124 Maximum air bladder pressure. (Counts) Mfb
110 OK vacuum threshold. (Counts)      Mfv (trick float)
226 Piston full extension. (Counts)     Mff
016 Piston storage position. (Counts)    Mfs
    2 Logging verbosity. [0-5]         D (keep it 2)
0002 DebugBits.                      D
70eb Mission signature (hex).

```

“User” is to set the gain, a number of max detections per dive and the depth to turn on/off the WISPR program which can be set by i*u command. Note that “OK vacuum threshold” is specified high (normally 96) so that we can conduct the bench test while the system is opened at ~1 atm. “Park piston position” is also set to high so that the excursion

does not take too long from the surface piston position which is close to the full extension (226).

To begin a mission, type 'e' from a console. You can leave the SAIL serial term on and watch the progress of mission. During the prelude and telemetry mode, the operator loses control of apf9 and Apf9 does not respond. During descend, park and profile mode, apf9 responds to the operator by hitting 'Enter' key. It responds with > prompt and operator can type 'k' to kill the mission and 'y' to confirm. To continue the mission, operator can type 'q'. It goes to low-power hibernate mode (<1mA @15V) immediately. Doing nothing also makes the QUEphone hibernate mode in a few minutes.

To end the mission on bench, you can either hard reset the apf9 board (see the picture 8), or hit return during descent, park and profile mode and kill the mission. Note that apf9 does not respond to any key stroke during the prelude and telemetry. Removing the power to reset is the last resort since it erases the GPS almanac and it takes 15 minutes to download the update. It may also corrupt the file system of the WISPR's CF card if it is writing data. The board is static sensitive, so touch ground before connecting or disconnecting connectors.

b. Example of actual mission parameters

Below is the list of actual mission parameters for the water off Washington, in March 2015. With this setting, Q003 repeats the descent-park-ascent-telemetry cycle once a day. At 870 min (14:30 UTC or 6:30 PST) Q003 changes from parking to profile phase mode and starts ascending. It takes about 3.5 hours (at 8 cm/s) for the QUEphone to ascend 1000 m, therefore the expected surface time is ~10AM (PST).

```

APEX version 15030511  sn 4693      (version num - date, sn = APEX hull number)
User: G3D999999P0450      (Gain=3, Max detec=999999, WISPR On at 450m)
Pwd: Q003                  (QUEphone #)
Pri: ATDTxxxxxxxxxxxxx    Mhp (Rudics phone #)
Alt: ATDTxxxxxxxxxxxxx    Mha
    0870 ToD for down-time expiration. (Minutes)    Mtc (Time to ascend)
    00370 Down time. (Minutes)    Mtd
    00480 Up time. (Minutes)    Mtu
    00280 Ascent time-out. (Minutes)    Mta
    00000 Deep-profile descent time. (Minutes)    Mtj
    00230 Park descent time. (Minutes)    Mtk
    00015 Mission prelude. (Minutes)    Mtp
    00015 Telemetry retry interval. (Minutes)    Mhr
    00060 Host-connect time-out. (Seconds)    Mht
    0 Continuous profile activation. (Decibars)    Mc
    1000 Park pressure. (Decibars)    Mk
    1000 Deep-profile pressure. (Decibars)    Mj
    034 Park piston position. (Counts)    Mbp (call teledyne)
    000 Compensator hyper-retraction. (Counts)    Mbh
    016 Deep-profile piston position. (Counts)    Mbj
    010 Ascent buoyancy nudge. (Counts)    Mbn
    022 Initial buoyancy nudge. (Counts)    Mbi
    254 Park-n-profile cycle length.    Mn
    124 Maximum air bladder pressure. (Counts)    Mfb
    096 OK vacuum threshold. (Counts)    Mfv
    226 Piston full extension. (Counts)    Mff
    016 Piston storage position. (Counts)    Mfs
    2 Logging verbosity. [0-5]    D    (keep it 2)
0002 DebugBits.    D
55ad Mission signature (hex).
```

It turns on/off the WISPR at 450 m with the preamp gain of 3. If the number of the accumulated detections per dive exceeds 999999, the WISPR stops the detection and returns to the surface for telemetry.

The park-piston position is where the hydraulic pump stops to reach the desired parking depth. It depends on the parking depth and sea water density of the area. It is also system dependent which varies by the weight of each system in water. The user must contact Teledyne Webb for the optimum piston position for the target parking depth. They will give you the piston position and weight of the float.

2. WISPR board (EOS, WA)

Before each mission the CF card must be formatted (FAT32) and three program files must be installed (e.g., wispr bw, console prompt and script file, 'start'). Copy the WISPR program and associated files from your PC through TCPIP. Erase any data files including *.flac, *.log and *.dtx. To test WISPR, follow the steps below.

1. Connect COM0 of WISPR to PC com port. Baud rate is 115200.
2. Turn on WISPR from apf9 console (SAIL 9600 baud) by giving a command i*o.
3. Watch the COM0 console as the Blackfin WISPR processor boots up. When prompted, type 'q' to interrupt the progress of start script and stop the program. Otherwise WISPR program would start.
4. Data files and programs are in /mnt. Clean the CF cards by erasing all the *.dat, *.flac, *.dtx, *.log files or reformat the card in FAT32 if it has not been done so.
5. If reformatted, copy the WISPR program and associated files to /mnt.
6. Type 'cd /' and unmount the file system by typing 'umount /mnt' on COM0.
7. Vi edit /mnt/start file and change some of the logging parameters to suite your mission. (see the Wispr program descriptions)
8. If it is necessary to re-mount the file system,
/bin/mount /dev/sda1 /mnt
9. If the program has started, you can end it from apf9 console by typing "i*x".
10. At apf9 console, you can turn off the WISPR by typing "i*k". **You should not turn off the WISPR without ending the program properly and also unmounting the file system.**

For more details on the WISPR system go to EOS web site.

<http://embeddedocean.com/passive-acoustics-2/wispr-v1-0/>

3. Getting ready

The weight of QUEphone in the air is approximately 25 kg and it varies depending on the construction. Before the new mission, it is necessary to change the batteries, desiccants, and adjust the weight to the manufacturer's recommended value within a few grams. It requires a high precision scale (such as <http://www.arlynsscales.com/Ultra-Precision-Scales-with-Super-Sensitivity-and-p/ultra-precision-scale.htm> Also purchase of a 25 kg calibration weight is recommended.).

After changing the battery and desiccants, first measure the weight of the QUEphone without pulling air (opened but everything is in it). You may have to add or remove lead pellets inside the housing so that it would become ~3 grams heavier than the

manufacturer's recommended value (~3 grams is the air weight inside the tube). After pulling the air by the vacuum pump with a custom fitting with pressure gauge (Teledyne Webb provides), make the weight measurement again. The internal vacuum level should be approximately 6.6 inch Hg at 20 to 24 °C. This vacuum level must be less than "OK vacuum count" in the mission list (96). The command to use the internal pressure is "ic".

For shipping purpose, the piston position should be ~110. Type "id" to display the piston position. To change the piston position, use a command "ig 110". In the field, before deploying you must move the piston to 220 by giving a command "ig 220". This ensures the QUEphone to float in the sea water and stay at the surface so that it would start telemetry in 5 min to send GPS position and the status before it start diving.

Opening, closing the top end cap and accessing WISPR board

Opening:

The end cap and pressure tube are held together by one connecting mating rod at the top of cage and mating nut (hex head) of the end cap. To access the mating nut, open a seal plug, which can be loosened by inserting a long (10") 3/16 hex wrench. The pressure inside the tube is negative, so it makes hissing sound as you rotate the seal plug and open the hole. Max torque for this seal plug is 18 lbs in. To separate the end cap, you must insert a 3/16 10" long hex wrench through the service hole to access the hex head of the internal cage. As you loosen the connecting rod, wiggle the end cap to make it free.

The end cap and main electronics are connected by four cables. After separating the end cap, carefully disconnect these four cables. Connecting and disconnecting the four cables must be done through a small space between the pressure tube and end cap, and it is tricky. Ask someone to help you on this.

Disconnect the four cables in the following order:

- (1) 2-pin power connector (red and black wires),
- (2) 10-pin (Seascan sensor)
- (3) 7-pin (WISPR) connectors.
- (4) GPS/Iridium antenna cable

The WISPR data are stored on CF card which is on the back side of the WISPR board which requires unscrewing the four mounting bolts (see the Picture 11). When removing the CF card, make sure that WISPR power is not on.

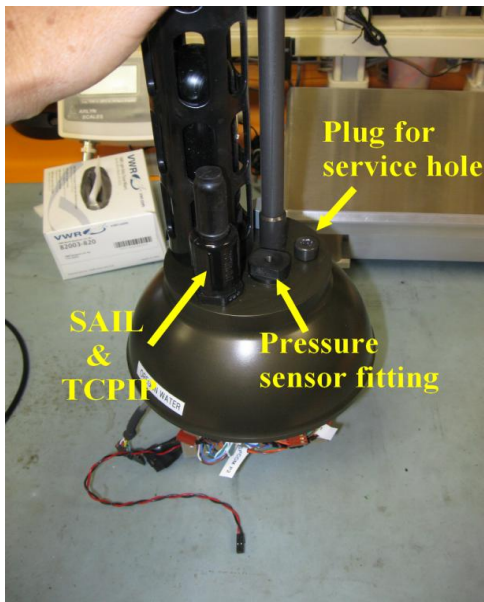
Closing:

Connecting the cables is by the following order.

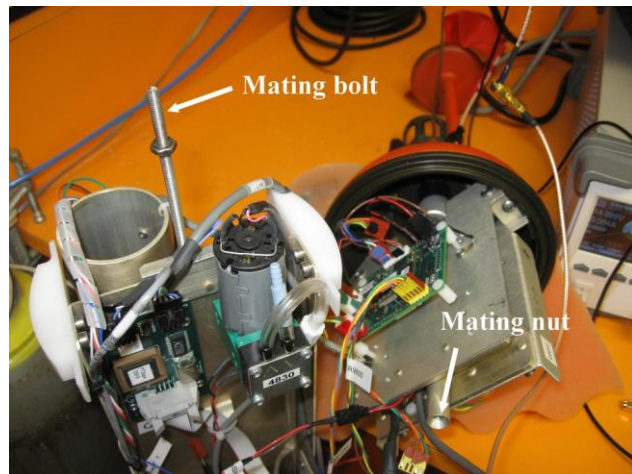
- (1) GPS/Iridium antenna cable
- (2) 10-pin (Seascan sensor)
- (3) 2-pin power connector (red and black wires),
- (4) 7-pin (WISPR) connectors.

This order seems to reduce the chance of starting up the WISPR's program accidentally. After connecting the cable and closing the end cap, connect the SAIL to the QUEphone and check the status. On the serial terminal, type two commands in the following sequence: "i*x" and "i*k". The first command (i*x) stops the WISPR program in case it accidentally started. i*k kills the power to the WISPR board.

Clean the mating surface and O-rings. You must align the marks of the bottom end cap, pressure tube, and end cap. This is to align the internal magnet sensor to the magnet swipe area of the housing. After installing the end cap and tightening the mating hex nut and connecting rod (18 lbs torque) by the wrench, the same service hole is used to pull air out by a vacuum pump with a custom fitting with O-ring at the end. Instead of tightening the plug, leave a few turns off so that the internal air can be pulled off. The vacuum should be -6.2 to -7.2 in Hg. Be aware that the pressure gauge is not too accurate. Check the APEX internal pressure sensor reading while pulling the air. The SAIL still connected type “c” to check the vacuum level. You need a second person to type “c” while keeping the custom vacuum fitting pressed flat against the service hole surface and tightening the plug at the same time.



Picture 6. End cap



Picture 7. Connecting rod and mating nut



Accessing the hex head of the mating nut through vacuum hole. A long Allen wrench is inserted diagonally to reach the head of mating nut.

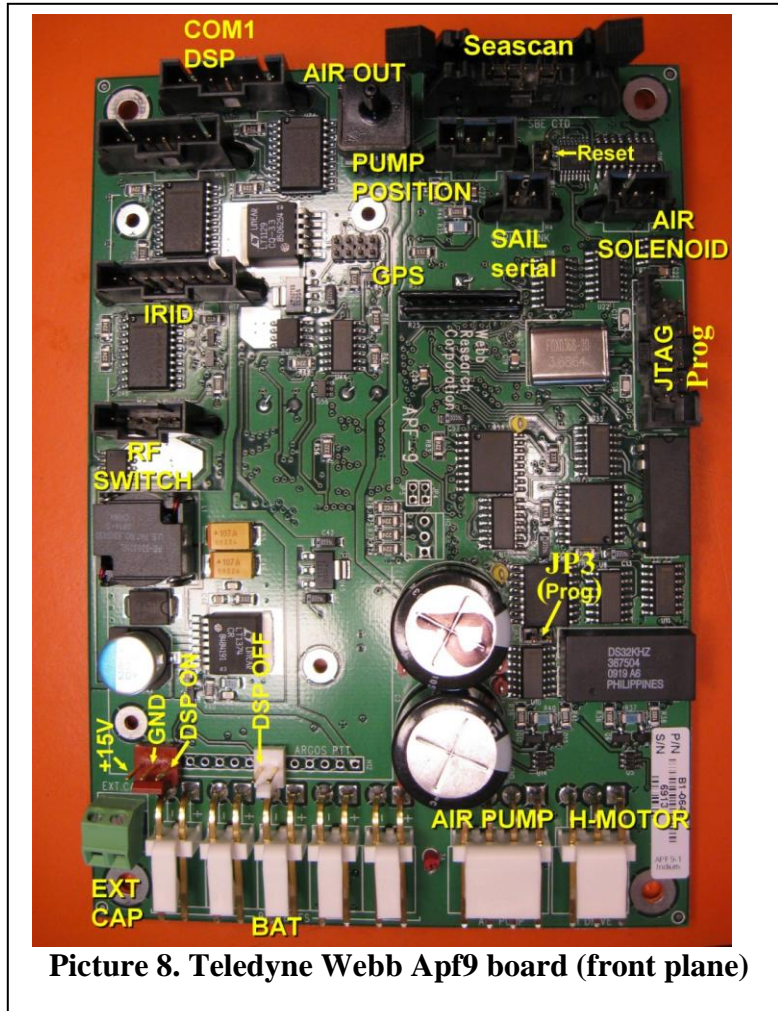


Vacuum fitting. Target pressure ~ -6.7 inHg.

V. System

1. Apf9 hardware modifications

Five modifications on the Teledyne Webb apf9 near ARGOS PIT mark are necessary in order to make it work with WISPR. Three modifications are on the component side at the unused slot. Two are on the back plane (solder side).



Picture 8. Teledyne Webb Apf9 board (front plane)

Front plane (component side)

- +15V power to WISPR (on Molex connector) (pin 15)

- Power On digital line (pin 13)

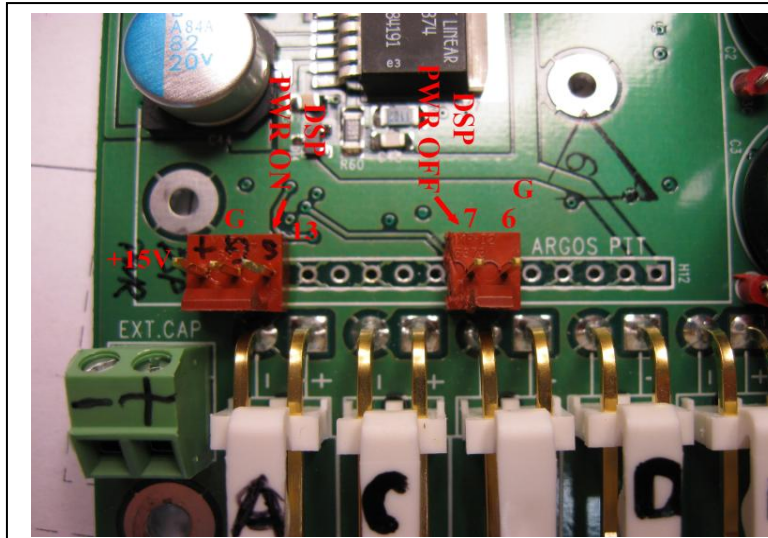
- Power off digital line (pin 7)

Backplane (solder side)

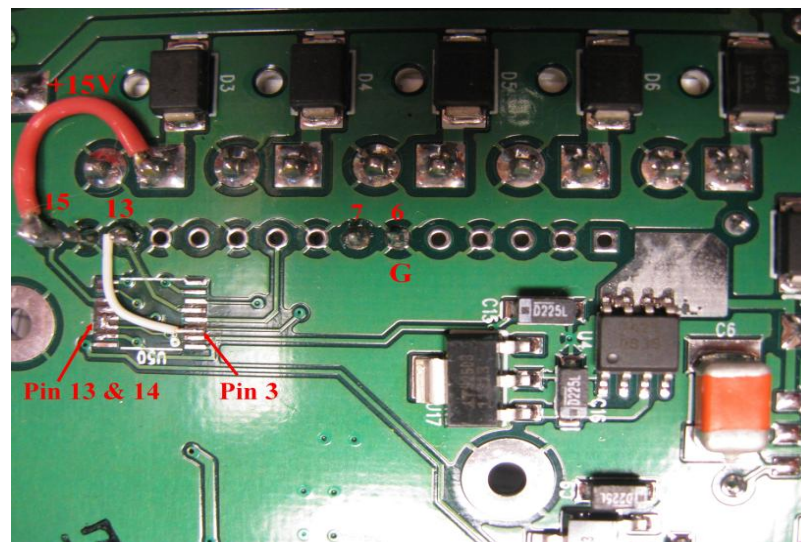
- Empty slot for U50 IC (see back plane).

- Connect pin 13 and 14 of U50 IC empty slot.

Front plane (WISPR DSP power on/off pins)



Picture 9. Apf9 front plane (WISPR power on/off pins and power pins)



Picture 10. Apf9 back plane

WISPR power is connected to the D3 diode output. Turning on/off the WISPR is controlled by two digital IO lines from apf9. It is turned on when the apf9 program sends a pulse (2 ms long) from pin 13 of ARGOS PIT to the WISPR's CONTROL connector (pin 3). When WISPR is in dormant mode, it still draws ~1 mA to keep the onboard real-time clock subsystem active. When it receives a pulse at CONTROL pin 3, it powers up the main system and launches the logging/detection program. To turn off, Apf9 board sends a pulse from pin 7 of ARGOS PIT to CONTROL pin 4 on the WISPR. There are

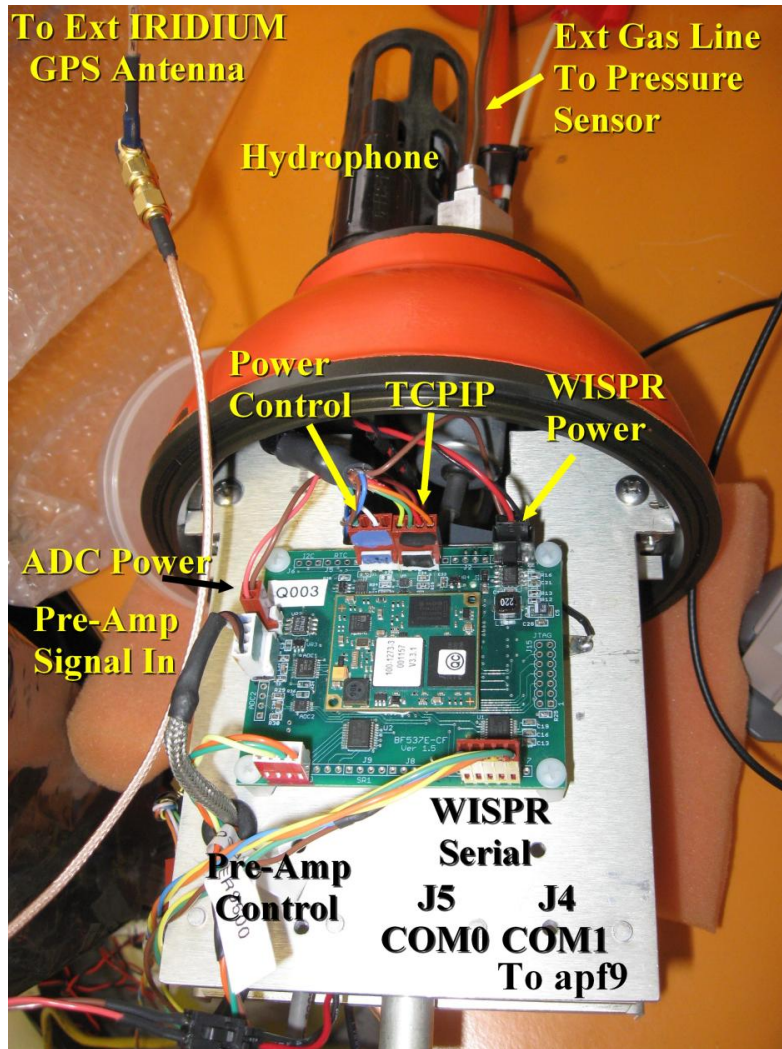
also two manual switches on the WISPR board (red and black push buttons) to turn on/off its power.



2. PAM system (WISPR DSP board and pre-amp system)

a. WISPR board mounted inside the QUEphone below the endcap

Both pre-amp and WISPR board are mounted where in the standard APEX Seabird CTD board is below the end cap. For QUEphone, CTD board is removed. After the end cap is closed, WISPR can still be controlled and monitored by TCPIP and apf9 serial com (SAIL). The two communication lines are available at the 6-pin Impulse connector.



Picture 11. WISPR board Rev 0 (with Molex connectors)

On the WISPR board, there are two separate power inputs: one for ADC and the other for WISPR's DSP. With the QUEphone, apf9 board and WISPR share the same battery power line (15 - 16V DC). "ADC input" takes the pre-amp differential outputs with a shielded cable (0-5V). Powering on/off the WISPR is controlled by apf9 by software. During the dive, when it passes a certain depth threshold, it turns on the power of WISPR. Once the power is on, Apf9 communicates with WISPR over a 9600 bps serial COM1 (J4). During the prelude stage, Apf9 sends inquiries to WISPR for the CF card usage (% of free space), GPS time, and pre-amp gain. WISPR controls the power the pre-amp and controls the gain by its GPIO. If WISPR is inquired by Apf9 for click detections, it sends brief detection results. 115200 baud rate COM0 is reserved for debugging only and not used for missions. The power ground is connected to the aluminum frame.

Picture 11 shows the WISPR board cabling. Pre-amp signal cable to the A/D input must be kept away from noise source particularly digital cables. To utilize the Ethernet

capability, set IP address of WISPR board 192.168.0.20. Your laptop PC's IP address also has to be 192.168.0.XX.

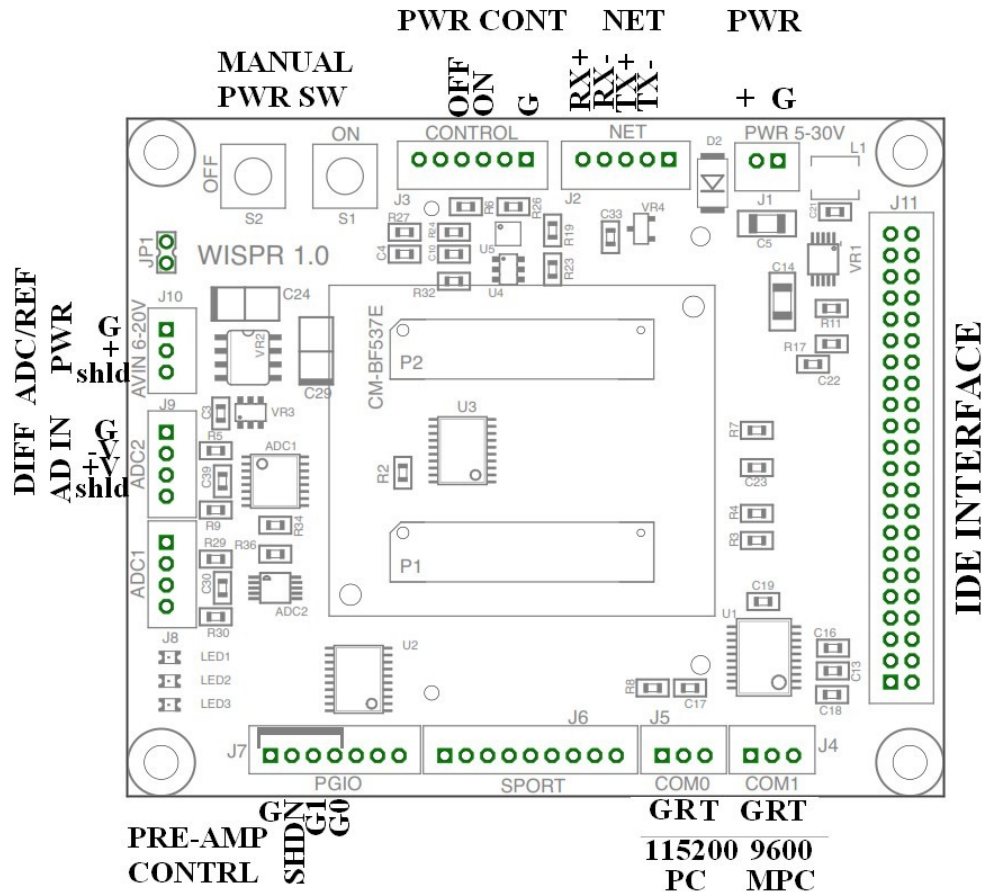


Figure 7. WISPR Rev 1

WISPR uses 24-bit analog-digital converter (ADC by J10 in Figure 7) with a differential input with a range of 0-5 V per channel (total input range is 10V). The ADC has a built-in Sigma-Delta low-pass filter, which eliminates a need for anti-aliasing low-pass filter at the pre-amp. When it is running at CPU clock speed of 250 MHz at 125 kHz sampling rate, it draws approximately 70 mA @ 15V.

WISPR has one CF card slot on the backside. Currently (January 2015) 512 GB is the largest card on the market which holds approximately 23 days of continuous acoustic record sampled at 125 kHz without file compression (FLAC). With FLAC 2 compression, the same card holds ~47-day data.

QUEphone has a payload capacity for three D-cell alkaline battery packs which keep the system electronic running for 14 days for a 0-1000 m and 20 days for 0-500 m profile mission. It also can take three DD-cell Lithium packs and one D-cell alkaline pack, which increases the battery capacity to last a 1 to 1.5 month mission.

The operating system of WISPR is uClinux. It executes the programs stored in the CF card by loading and executing the commands and programs on 'start' script file. An example of 'start' file is described in the following section. WIRPS communicates with other devices through its two COM ports. COM0 is 115kbps which is used for console

monitoring when it is connected to a PC. COM1 (9600 bps) is used to communicate with other devices including the Apf9 board of the APEX float.

It has one Ethernet port (J2), however, the data speed is limited to 10 Mbps. Therefore, the Ethernet is not suitable for downloading the entire CF card contents.

b. Pre-amp and hydrophone

EOS Pre-amp including the HTI92WB hydrophone draws approximately 4.5 mA @ 15V with a band width of 1 Hz to 62.5 kHz with a programmable gain (0-3 steps with +6 dB per step). It is installed on the opposite side of the aluminum bracket that WISPR board is installed.

The WISPR board controls the pre-amp power and gain by three digital IOs. When the power is not on (~~SHD~~ is on at J4), the EOS pre-amp draws ~100 μ A. When the power is on, the pre-amp draws approximately 3 mA at 15 V. Its differential output (J3 0-5V) is connected to the WISPR's 1-channel ADC differential input (J9).

Figure 8 shows the pre-amp gain when the gain = 0 (minimum gain). Use can choose the gain digitally by setting two digital IO lines 0 or 1 (G0 and G1 of J4). One increment of the setting increases the system gain by 6 dB in all frequency (0, 6, 12, and 18 dB).

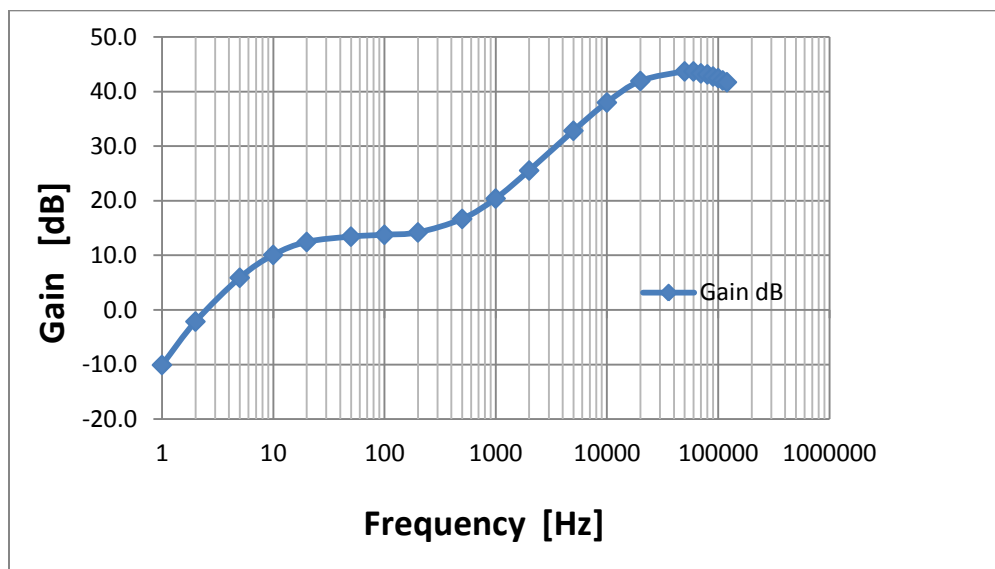
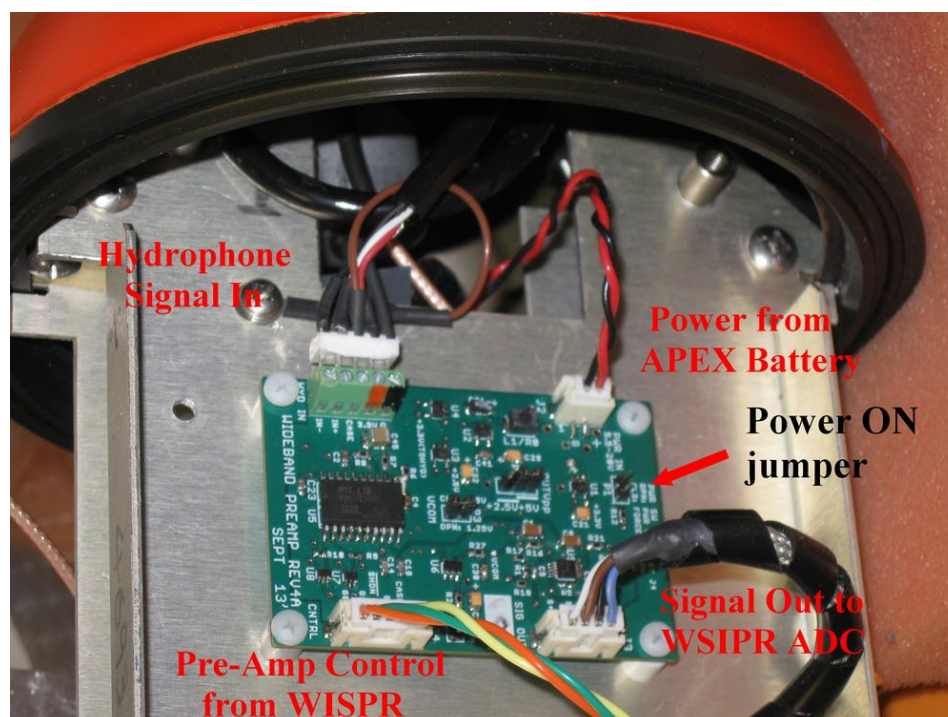
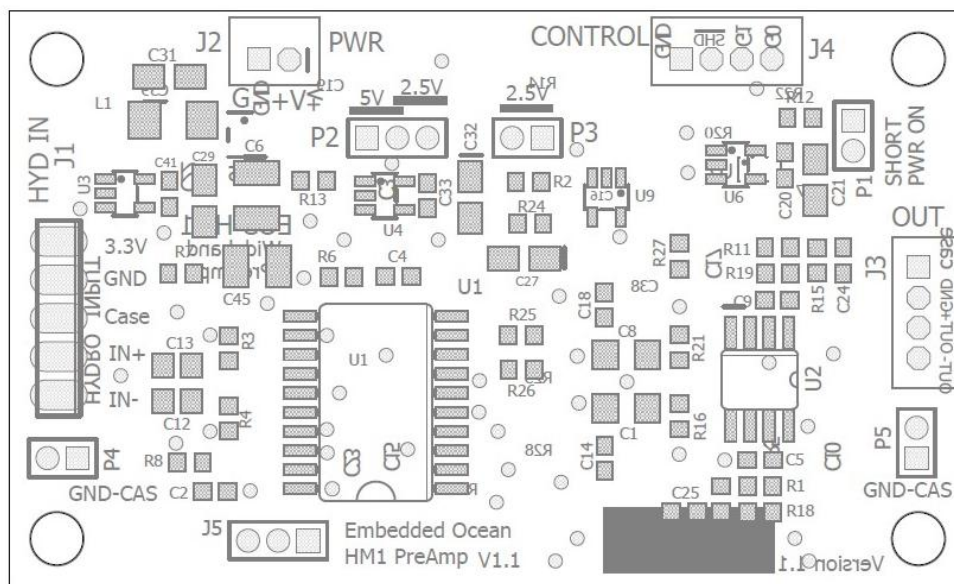


Figure 8. Gain curve of the EOS HM1 pre-amp with gain = 0.

High Tech Incorporated <http://www.hightechincusa.com/Main.html> manufactures HTI92WB. It has a sensitivity of -165 or -175 dB re 1V/ μ Pa (user choice) with a 1-pole high pass at 25 Hz or 50 Hz (user choice). Users must be aware that to calculate the total system sensitivity, the first-order high pass filter effect at -6dB/octave of the user-specified high pass filter frequency of the hydrophone. The users also should be aware that the HTI provides only one nominal sensitivity measured at 3 kHz and it is usually not constant across all the frequency band.

Picture 12 shows the cablings on the pre-amp side. Because of its high sensitivity, it may pick up digital noise from the processor or serial communication, and data may become contaminated. The hydrophone cable must stay away from any digital or communication cables and must not touch any other cables. P1 jumper on the pre-amp is used for bench test only, and must be off for normal operations. P2 defines the range of output, which should be on 5 V (P2). P3 defines the mean of the output voltage which should be 2.5V for WISPR.



Picture 12. Pre-amp

3. WISPR program commands (ver 2)

WISPR runs a logging program for the acoustic data and runs a Teager-Kiser detection algorithm for beaked whale simultaneously. You can include options below in the command line of the 'start' file of the CF card. The followings are descriptions of the command line parameters.

Options:	DESCRIPTION	DEFAULT
-M {mode}	Processing mode number	[1]*
-T {secs}	Size of ADC data buffers in seconds	[up to 8 seconds]
-F {level}	Sets flac compression level	[2, 3]
-s {bitshift}	Sets data bitshift	[8]
-o {organization}	Organization in FLAC metadata string	[none]
-b {number}	Number of data buffers per file	[10]
-n {nclick}	Min number of click for detection	[10]
-q {nbufs}	data snippet option (not implemented yet)	[?]
-i {number}	Number buffers to skip between file	[10]
-p {prefix}	Data file name prefix	[wispr_]
-l {filename}	Log file name	[no log file]
-v {level}	Verbose level (0=none)	[0]
-L	Enable LEDs	[disabled]
-r	Request GPS time, lat/lon and gain at start	[disabled]
	wait GPS signal for 17 sec, for free disk space	57 sec
-W	Simulated test detection mode	[disabled]
-f	sampling frequency 62500, 93750, 125000	[125000]
	If wispr_kw (Killer whale) 93750 is default	
-g	gain 0, 1, 2, 3 (additonal gain with 6dB incr)	[0]
-h	print help	
-C	number of files to record	
x	cpu usage time	

One single program can be operated in the following 5 different modes.

*Modes

- Mode 1: Record continuously, no detection functionality
 - Mode 2: Record continuously with detection functionality
 - Mode 3: Record intermittently.
Skip specified number of buffers between files.
 - Mode 4: Run a beak whale detection function only.
Process incoming data continuously with the detection function but only write a file when a detection appeared.
 - Mode 5: Run beak whale detection function and record data intermittently.
Same idea as mode 4 + record data intermittently.
Used to get an idea how many encounters were missed or to monitor noise levels regularly).
- **-r Send interrupt to ext processor to receive GPS time, lat/lon and gain during the first 17-second start up, and wait for 5 sec. Default is to receive the GPS time, lat/lon and gain without interrupt. Send % of free space of CF card 40 sec later (57 sec after start up).

***Simulated test detection mode (bench test only)

Test the detection performance by overwriting AD buffer content with a short wave file of a specific species clicks/calls before processing the buffer. File name has to be **bw_test.wav**.

Detection results are stored in **detections.dtx**.

Note

When the Apf9 board turns on the WISPR's power, in ~7 seconds it starts uClinux, mount the file system, TCPIP and executes the logging/detection program (wispr). The command line defines the pre-amp gain (-g option) and if GPS time synchronization is requested. With '-r' option, the wispr program synchronizes the board RTC (real-time-clock) to GPS time. It also calculates a free storage space available on the CF card. The free space calculation of the Lexar 512 GB card takes ~55 sec. WISPR then sends a free space in % to apf9 board.

Clock speed of the WISPR is slowed down to 250 MHz to save battery power (default is 500 MHz). By slowing down the CPU speed a half, it saves the power by 25% (~100 mA at 500 MHz). These two commands should be in start script.

```
echo userspace > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
echo 250000 > /sys/devices/system/cpu/cpu0/cpufreq/scaling_setspeed
```

To know what CPU speed it is running, type

```
cat /proc/cpuinfo
```

The WIPSR board comes with 64 MB RAM. To use all 64 MB and to avoid memory leak, you need to set the memory space for the kernel and user. If you have a new WISPR from EOS, you need to do this once. Here is how.

```
>setenv bootargs root=dev/mtdblock0 rw clk-in-hz=25000000
earlyprintk=serial.ports0,115200 mem=48M max_mem=64M$#
console=ttyBF0,115200
>save
>boot
```

This needs to be done only once.

4. Example of 'start' file on CF card

```
/bin/ifconfig eth0 192.168.0.20
#dhcpcd &

cp /mnt/console_prompt /bin
chmod 777 /bin/console_prompt
count=10
quit='q'
echo " "
echo "--- Embedded Ocean Systems WISPR V1.0 ---"
echo "Starting recording and detection."
echo "Enter 'q' to stop."
while [ "$count" -gt 0 ]
do
    val=`console_prompt -t 100`
    if [ "$val" = "$quit" ]; then
```

```

        exit 1
    fi
    count=`expr "$count" - 1`
done

# Uncomment these to cut cpu freq in half
echo userspace > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
echo 250000 > /sys/devices/system/cpu/cpu0/cpufreq/scaling_setspeed

sleep 1

# clear cache to free memory
sync; echo 3 > /proc/sys/vm/drop_caches
sleep 1

# Start detection/recording
#echo "WISPR recording and detection started."

cp /mnt/wispr /bin
chmod 777 /bin/wispr
#/bin/wispr -l wispr.log -v2 -T8 -M2 -W -F2 -r
/bin/wispr -l wispr.log -v2 -T8 -M2 -F2 -r

```

5. Communication with QUEphone

a. With Apf9 board

You can serially communicate with the apf9 board in the QUEphone via 2-wire SAIL-to-serial RS232 converter. Teledyne provides the converter which needs an external +12-V. Baud rate is 9600.

Apf9 board, when it is idle (hibernate mode) during a mission, draws less than 1 mA (@15V). When it is not in mission and connected by serial (e.g., on the lab bench), it draws ~15 mA. When it is connected to PC via a serial term program, salutation and prompt (>) should appear on the console. If no prompt (>) appears, hit the enter key. Type 'L' (case insensitive) to list 28 mission parameters of APEX float.

For the details of these mission parameters, refer to the Teledyne Webb APEX manual. You must be aware that all the time durations, parking depth, and piston positions have to be consistent. Always run mission sanity check ('z') whenever the mission parameters are changes. For programmers, be aware that when a new program is burned in, it often changes the verbose level. Make sure Verbose level is 2 (command d2). Verbose 1 is not detail enough and >2 is too detail and transmitted file size becomes too large for Iridium operation. Maximum file size is set 62 kB (can be changed by i*1 command). Verbose=3, 4 are used only when testing the Iridium code on the bench for transmitting a large file (<60kB).

Figure 9 shows how QUEphone's Apf9 board communicates with outside locally via SAIL serial and WISPR, and remotely by Iridium including baud rate.

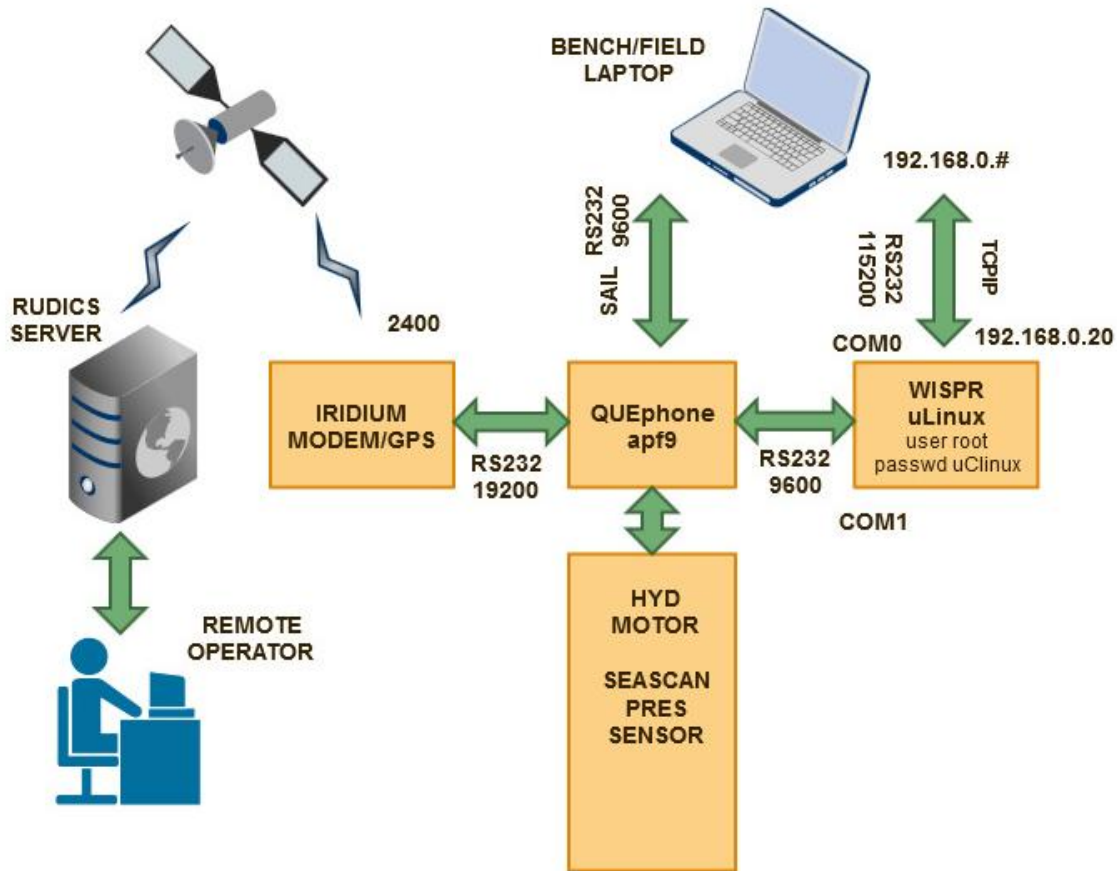


Figure 9. QUEphone local and remote communication

b. With WISPR board

If QUEphone is not closed yet, you can communicate with the WISPR via RS232 serial COM0 port (115k baud rate) on the WISPR board. COM0 displays useful bootup info but slow. Once the housing is closed, however, the only way to communicate with the WISPR is via a 4-pin Ethernet connector on the end cap. Ethernet is fast but does not show bootup message and cannot interrupt the boot up process.

For the bench test, you can start the WISPR program manually on the Apf9 serial terminal by typing `i*o`. It powers up the WISPR and if you are connected to WISPR's COM0 terminal, bootup messages start appearing. After 10 lines or so, `bfin_mac` shows up. Type any key to stop "auto boot." If you are able to interrupt the autoboot process in time, it only loads up the basic kernel in its RAM. A full uClinux system is not loaded yet.

If you wish to set date and time of the WISPR manually at this point, type

date 050511152017
(Date Time year)

As you notice, it can set time only to the minute. Once time is set, WISPR retains the time in RTC until the battery power is unplugged. Now reboot the WISPR by typing 'reset' and this time let it boot up until the penguin figure shows up and the logging program partially starts. When it starts, it gives ~5 sec for you to decide to stop or launch the logging program. If you type 'q', the logging program stops. If you do nothing (default), timer expires and program begins.

If you can stop the program in time, you can use some Unix commands to check the system. A better way to set the clock is after uClinux is fully loaded is by date command.

```
date -s 2017.07.10-22:20:10
```

This is a software clock, which allows to set the Unix clock up to second. To transfer software time to the internal hardware clock, you need to type

```
hwclock -w
```

Now rebooting the WISPR by typing 'reset' would not lose the current time.

Try other Unix commands. For example, type 'ps' to see if any program is running. You can kill the process by Unix kill command.

At this point, you can connect to the WISPR by Ethernet. Open Cygwin or MSDOS window to telnet WISPR by

```
telnet 192.168.0.20
```

No login user/password is required. (or if it asks, user name is root and password is 'uClinux')

After making sure that logging/detection program is not running, change the directory to /mnt to edit file or remove the data files from the previous mission. There is 'start' script file in /mnt (CF card). You can vi edit start file. In the 'start' file, examine the last command line very carefully to see if it does the job you want. After exiting out vi, move back to the root directory and unmount the file system in preparation for powering down.

```
umount /mnt
```

Before unmounting the file system, check if the logging program is still running. If so, a proper way to kill the program is from the Apf9 terminal window by typing

```
i*x to exit out the WISPR logging program (Apf9 send $EXI*)
```

```
i*k to kill the WISPR power. (it physically turns off WISPR). Wait for 20 seconds.
```

```
To power up
```

```
i*o
```

Apf9 board sends commands to the WISPR's COM1 (9600 bps) to control. It can also change the GPIO lines to execute the other commands. Details of character strings that Apf9 sends to WISPR's COM1 are described in the following section.

In case you need to re-mount the CF card file system, do the followings from the COM0 serial term (115200 bps)

```
/bin/mount /dev/sda1 /mnt
```

If you need to fdisk a new CF card in preparation for a new deployment, do the following on WISPR from COM0 or log in through TCPIP Ethernet (10MB/s).

/bin/fdisk /dev/sda

p to print partitions

If it is 512 GB is formatted properly, it should look like this.

Device	Boot	Start	End	Blocks	if	System
/dev/sda1	*	1	622268	500167678*	83	Linux

A new CF card partition may be different from these, and if it is different, delete the existing partition by '**d**'

Partition number (1-4) : 1

Add a new partition by '**n**'

e extended

p primary (1-4) :

Type '**p**' for primary

Partition (1-4): 1

First cylinder (1-622268, default 1) carriage return for default

Last cylinder : carriage return for default

p to print the partition again

w to write the partition

After fdisk you need to reformat the disk in FAT32. If the file system is still mounted, unmount it first then

/bin/mkfs.vfat -F32 /dev/sda1

To power off the WISPR, be sure to end the wispr program and unmount /mnt first.

The 4-pin TCPIP and 2-pin SAIL ports are available at the 6-pin AG206 connector. You can download a few data files from WISPR to your PC by TCPIP. For example use cygwin window and type (the last '.' means the current PC directory)

scp root@192.168.0.20:/mnt/filename . or

scp filename root@192.168.0.20:/mnt

For file transfer, it may ask for user name and password. User name is root and password is "uClinux". WinSCP is another way of doing file transfers and may be more useful than cygwin. Note that WISPR's TCPIP speed is limited 10 Mbps which is too slow to download the entire directory of the 512 GB CF card (~6 days). It also uses battery power. So transferring the entire directory to PC by a slow TCPIP is not recommended.

If the QUEphone is open and the WISPR board is exposed, you can start, interrupt and kill the program by connecting the PC COM port to WISPR's COM1. The followings are the character strings at COM1 WISPR expects to see or sends to/from Apf9 board. Cr is a carriage return.

\$GPS ,%ld, %8.3f,%7.3f*	Cr GPS time, long and lat	apf9->WISPR
\$DX?,%ld,%ld*	Cr Inq detections	apf9->WISPR
\$DXN ,%d*	Cr Num of detections	WISPR->apf9
\$ACK*	Cr Send ACK for each line	WISPR->apf9

\$NGN,%d*	Cr New gain (0-3)	apf9->WISPR
\$EXI*	Cr End logging	apf9->WISPR
\$DET,%d*	Cr Detection parameter	WISPR->apf9
\$DFP*	Cr Inq disk space	apf9->WISPR
\$DFP,%5.2f*	Cr Reply disk space avail %	WISPR->apf9

If the system is open, and you have an access to the WISPR board, pushing S1 (black on button) starts the WISPR kernel and executes 'start' script on CF card. It is equivalent to 'i*o' from Apf9. Pushing S2 (red off button) is equivalent to 'i*k', which is to pull the power off from WISPR. Caution! Once the logging program starts running, pushing OFF button is strongly discouraged. Because CF card is still mounted and the program may be a writing file, pulling the power off means crashing the program without closing the file. Too many of these eventually may damage the file system and data recovery will be difficult. A proper way to end the program is to send \$EXI* command to COM1 from your computer's COM port, which is equivalent to 'i*x' from Apf9. Wait several seconds and type

```
umount /mnt
```

to make sure the file system is no longer mounted and safe to pull the power off.

VI. A day before deployment

Most likely you are shipping the QUEphone to a location where the coordinates are significantly different from your lab. GPS receiver uses a time schedule (called GPS almanac) from satellites about which GPS satellites up and it differs by coordinates and date. When GPS receiver moves to a new location by a degree or so, or if it has not been on for a long time, it needs to download a new almanac before it can figure out its coordinates. The downloading takes ~15 min, which not only slows down the deployment but also the ocean current drifts QUEphones where it is deployed. For these reasons, it is recommended to take the QUEphone out a day before and operate in prelude mode to capture the new almanac at a new location by the pier or on the boat. To do this, we recommend to run a short preparation mission outside, which would take 30 min.

A day before the deployment

1. Move the QUEphone outside with a clear sky view for satellite com.
2. Plug in the 6-pin Impulse connector on the QUEphone end cap.
3. Connect the SAIL com to the Impulse connector and open serial communication program (9600, N, 8, 1). SAIL needs 12-V external power! No polarity with SAIL. One to clamp on to the blue cable and the other to the case ground (use Zinc anode).
4. On your laptop serial com screen, hit return key (9600, N, 8, 1). This wakes up the QUEphone). Screen display should look like:
Asynchronous wake-up detected (ie., wake-up not initiated by alarm signal).
Entering Command Mode [ApfId 4694, FwRev 101215]
5. Type "L" to list the mission parameters and check against the logging sheet.
6. Move the piston to 226 by typing "**ig 226**" and wait. This inflates the oil bladder to be positively buoyant in water. QUEphone is shipped typically with position 110. It takes ~15 min.

7. Type “**Ga**” to download the recent GPS almanac at your location now. It takes 15 min.
8. Type “**Gc**” to configure the GPS. It takes 12 sec. GPS almanac is updated.
9. Type “**Gt**” to synchronize the QUEphone internal clock.
10. Type “q” to quite. Disconnect the SAIL and replace the rubber plug.
11. QUEphone is ready to deploy for the next day.

25 min before the deployment

1. On the deck, move the QUEphone to where it has a clear sky view for GPS and Iridium.
2. Inform crew that QUEphone is ready to deploy in 25 min.
3. Open the serial com program and connect the SAIL and hit return to wake up the QUEphone. Screen display should look like:
Asynchronous wake-up detected (ie., wake-up not initiated by alarm signal).
Entering Command Mode [ApfId 4694, FwRev 15112511]
4. Type “**L**” to list the mission parameters and compare with the logging sheet.
5. When you are ready, type “**e**” to execute the mission. The mission begins.
6. Self-test starts displaying system status.
7. Keep the SAIL connected and watch the serial terminal as the mission preparation progresses.

MIN What is happening

- 0 -Mission activated
- 2 -Starts searching GPS
- 3 -Starts 1-st Iridium TX
- 7 -If the 1-st Iridium TX is successful, the following message appears:
(Nov 17 2010 00:11:31, 1215 sec) Telemetry() Tlmtry cycle done:PrfId=0 ConnAtmpts=2 Conn_St=2

If the first Iridium TX try is not successful, it will retry a few more times. Wait until this 1-st Iridium com is successful. If the Iridium com is unsuccessful, do not deploy this QUEphone. When you see this, sat com was successful and QUEphone is ready to deploy.

Deploy in the next 15 min! Replace the rubber plug back.

- ↓ The system rests (9 min). **This is the best time to deploy.**
- 16 -Starts 2-nd GPS and Iridium TX
 - 20 -Ends 2-nd Iridium TX
 - 30 -QUEphone starts descending (takes about 10 min)**
8. Send text or e-mail to the land operator.

VII. Programming apf9 (motherboard of APEX float)

This chapter describes the procedure to compile a new program for apf9 board. Skip this chapter if you are not a programmer. If you need to modify apf9, leave it to a manufacture (OSU or Teledyne Webb). Current version is apf9_OSU6WISP. C source code are open to public and *.obj are available from OSU upon request.

1. Your PC

- Install cygwin (needs an internet connection to download)

- Install gnu make for win-32.
- Place gnu make in a directory that is included in the system PATH variable. rename it to just “make”
- Do the same thing for gcc, g++, perl, ln, and tr.
- Install PSDsoft Express by downloading from the SD microelectronics website. **At this point, PSDsoft driver is compatible with Window XP only.**

2. Programming Apf9 board

Currently OSU uses Cygwin terminal window to run Unix-like environment and compile. Compiler is HI-TECH XA. C programs and h files are in apf9**/src, /lib and /apf9lib. In the main program, apf9main.c, change the FwRev (about line 11) to the current date. This is the revision number.

1. Open cygwin window.
2. cd to the apf9_new directory. Typical directory is cygdrive/c/apf9_XXX.
3. Source codes are in src, lib, and apf9lib.
4. After changing any *.c or *.h files. Type ‘make’ to compile there.
5. The same applies to source codes in /lib and apf9lib
6. *An error about FwRev.exe is expected, ignore this. Do not ignore any compiler errors that appear*

Original apf9 source codes were written by Dana Swift, Univ. of Washington for direct modem line. OSU modified it substantially to interface with WISPR board and to communicate via NOAA/PMEL Rudics Iridium server, where ~100 buoys, remote stations, and autonomous sensor platforms send data to. A new compiled, linked, and mapped binary file is apf9xxx\psd\app.obj. The executable code can be written to apf9 board by PSDExpress software with Raisonance RLink with JTAG programmer.

3. Hardware setup

If you are not burning a new program to apf9 board, skip this section.

You need to install PSDsoft and RLink (STX-RLINK) from Raisonance on your PC. You must open the APEX float pressure housing to access the apf9 board. Need a hex wrench to access through a pressure plug and remove the top end cap (with antenna and hydrophone). During programming and testing to save battery life, you may want to change to an external power. Place a jumper on JP3 on apf9 board (see apf9 board picture). Plug in the JTAG connector to apf9. Watch the polarity of the connector.

1. Open PSDsoft Express
2. Press Cancel then OK.
3. Click on JTAG ISP in the PSDSoft Express Windows.
4. Choose one device in the chain in the JTAG-ISP operation window.
5. Select the psd/app.obj file.
6. The device is PSD835G2V
7. Other options: Program/Verify, Region: All, Package: U (80-Pin TQFP), 4 pins.
8. Press execute, then yes.
9. After ~32 sec a new program is burned. You can run the program while RLINK JTAG connected, but the JP3 jumper must be removed.

10. Do not save JTAG setup. (Save only once when a new project is created with a new subdirectory).
11. Unplug the JTAG connector and JP3 jumper.
12. In APEX float com screen type 'L' to list the APEX mission parameters on screen. Make sure verbose level is set to 2. To change to 2, type D2.

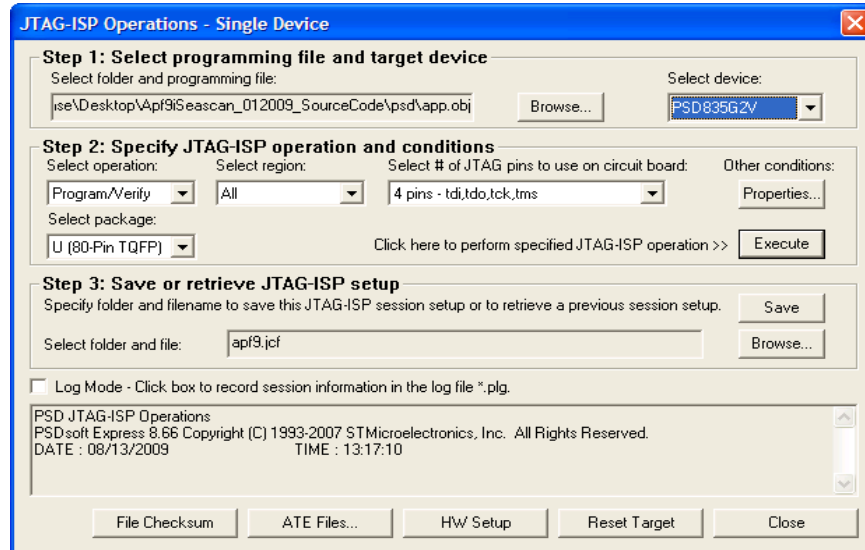


Figure 10. Apf9 programming- burning a new program

VIII. Example of QUEphone record received via satellite (bench test)

```

QUEphone version 15012114  sn 4693
User: G1D999999P0020
Pwd: Q003
Pri: ATDTxxxxxxxxxxxxxxx Mhp
Alt: ATDTxxxxxxxxxxxxxxx Mha
INACTV ToD for down-time expiration. (Minutes) Mtc
00132 Down time. (Minutes) Mtd
00192 Up time. (Minutes) Mtu
00072 Ascent time-out. (Minutes) Mta
00000 Deep-profile descent time. (Minutes) Mtj
00011 Park descent time. (Minutes) Mtk
00005 Mission prelude. (Minutes) Mtp
00005 Telemetry retry interval. (Minutes) Mhr
00060 Host-connect time-out. (Seconds) Mht
0 Continuous profile activation. (Decibars) Mc
50 Park pressure. (Decibars) Mk
50 Deep-profile pressure. (Decibars) Mj
210 Park piston position. (Counts) Mbp
000 Compensator hyper-retraction. (Counts) Mbh
016 Deep-profile piston position. (Counts) Mbj
010 Ascent buoyancy nudge. (Counts) Mbn
022 Initial buoyancy nudge. (Counts) Mbi
254 Park-n-profile cycle length. Mn
124 Maximum air bladder pressure. (Counts) Mfb
110 OK vacuum threshold. (Counts) Mfv
226 Piston full extension. (Counts) Mff
016 Piston storage position. (Counts) Mfs
2 Logging verbosity. [0-5] D
0002 DebugBits. D
57c3 Mission signature (hex).
> e Executing mission activation sequence.
(Jan 22 2015 00:17:40, 461 sec) SelfTest() Passed: int P [105,0.5"Hg]
<thrshld [110,1.9"Hg].

```

(Jan 22 2015 00:17:41, been initialized.	462 sec) SelfTest()	Passed: FLASH file system has
(Jan 22 2015 00:17:44, been reformatted.	465 sec) SelfTest()	Passed: FLASH file system has
(Jan 22 2015 00:17:47, Seascan PT module.	467 sec) SelfTest()	Passed: Response received from
(Jan 22 2015 00:17:48, [A183]	469 sec) SelfTest()	Passed: Seascan Serial Number
(Jan 22 2015 00:17:49,	470 sec) SelfTest()	Passed: LBT model: 9522A
(Jan 22 2015 00:17:50, ISO6004	471 sec) SelfTest()	Passed: LBT firmware revision:
(Jan 22 2015 00:17:51, (19200) baud rate.	472 sec) IrModemConfigure()	Modem configured for fixed
(Jan 22 2015 00:17:51, NVRAM.	472 sec) IrModemConfigure()	Configuration saved to modem's
(Jan 22 2015 00:17:51,	472 sec) IrModemConfigure()	Modem configuration successful.
(Jan 22 2015 00:17:52, successful.	472 sec) SelfTest()	Passed: LBT configuration was
(Jan 22 2015 00:17:52, 300224010449370	473 sec) SelfTest()	Passed: LBT IMEI:
(Jan 22 2015 00:18:37, ICCID:8988169414000461440	518 sec) SelfTest()	Passed: SIM card
(Jan 22 2015 00:18:38, initiated.	MSISDN:881693405479 519 sec) ConfigGarmin15()	Garmin GPS15 configuration
(Jan 22 2015 00:18:45, configured.	526 sec) ConfigGarmin15()	Garmin GPS15 successfully
(Jan 22 2015 00:18:46,	527 sec) PreludeInit()	Normal mission activation
(Jan 22 2015 00:18:47, itime=527]	1 sec) PreludeInit()	Mission restarted. [pid=1,
(Jan 22 2015 00:18:47, CPT:76s]	1 sec) PistonMoveAbsWTO()	210->226 [76s, 14.3V, 0.137A,
Mission activated.		
(Jan 22 2015 00:20:12, telemetry.	85 sec) Prelude()	Logging config and initiate
(Jan 22 2015 00:20:12, FwRev 15012114:	86 sec) LogConfig..()	Mission configuration for Apf9i
(Jan 22 2015 00:20:12,	86 sec) LogConfig..()	AscentTimeOut(72) [min]
(Jan 22 2015 00:20:12, [primary]	86 sec) LogConfig..()	AtDialCmd(ATDTxxxxxxxxxxxxxx)
(Jan 22 2015 00:20:13, [alternate]	87 sec) LogConfig..()	AtDialCmd(ATDTxxxxxxxxxxxxxx)
(Jan 22 2015 00:20:13,	87 sec) LogConfig..()	BuoyancyNudge(10) [count]
(Jan 22 2015 00:20:13, [count]	87 sec) LogConfig..()	BuoyancyNudgeInitial(22)
(Jan 22 2015 00:20:13,	87 sec) LogConfig..()	ConnectTimeOut(60) [sec]
(Jan 22 2015 00:20:14,	88 sec) LogConfig..()	CpActivationP(0) [dbar]
(Jan 22 2015 00:20:14,	88 sec) LogConfig..()	DeepProfileDescentTime(0) [min]
(Jan 22 2015 00:20:14, [count]	88 sec) LogConfig..()	DeepProfilePistonPos(16)
(Jan 22 2015 00:20:15,	88 sec) LogConfig..()	DeepProfilePressure(50) [dbar]
(Jan 22 2015 00:20:15, [count]	89 sec) LogConfig..()	CompensatorHyperRetraction(0)
(Jan 22 2015 00:20:15,	89 sec) LogConfig..()	DownTime(132) [min]
(Jan 22 2015 00:20:15,	89 sec) LogConfig..()	FloatId(4693)
(Jan 22 2015 00:20:16,	89 sec) LogConfig..()	FullExtension(226) [count]
(Jan 22 2015 00:20:16,	90 sec) LogConfig..()	FullRetraction(9) [count]
(Jan 22 2015 00:20:16,	90 sec) LogConfig..()	MaxAirBladder(124) [count]
(Jan 22 2015 00:20:16,	90 sec) LogConfig..()	MaxLogKb(50) [KByte]
(Jan 22 2015 00:20:17,	90 sec) LogConfig..()	MissionPrelude(5) [min]
(Jan 22 2015 00:20:17,	91 sec) LogConfig..()	OkVacuum(110) [count]
(Jan 22 2015 00:20:17,	91 sec) LogConfig..()	ParkDescentTime(11) [min]
(Jan 22 2015 00:20:17,	91 sec) LogConfig..()	ParkPistonPos(210) [count]
(Jan 22 2015 00:20:17,	91 sec) LogConfig..()	ParkPressure(50) [dbar]
(Jan 22 2015 00:20:18,	92 sec) LogConfig..()	PnPCycleLen(254)
(Jan 22 2015 00:20:18,	92 sec) LogConfig..()	Pwd(Q003)
(Jan 22 2015 00:20:18,	92 sec) LogConfig..()	TelemetryRetry(5) [min]
(Jan 22 2015 00:20:18,	92 sec) LogConfig..()	TimeOfDay(DISABLED) [min]
(Jan 22 2015 00:20:19,	92 sec) LogConfig..()	UpTime(192) [min]
(Jan 22 2015 00:20:19,	93 sec) LogConfig..()	User(G1D999999P0020)
(Jan 22 2015 00:20:19,	93 sec) LogConfig..()	Gain=1, Max # of
Detect/Prof=999999, WISPR ON/OFF Depth =20		

```

(Jan 22 2015 00:20:20,      94 sec) LogConfig..()      Verbosity(2)
(Jan 22 2015 00:20:20,      94 sec) LogConfig..()      DebugBits(0x0002)
(Jan 22 2015 00:21:25,     159 sec) AirSystem()
BatV[185cnt,14.3V]Amp[71cnt,28.6mA]BrmtrP[130cnt,7.7"Hg]Run-Tm[25s]
(Jan 22 2015 00:21:59,     193 sec) GpsServices()      GPS almanac is current.
(Jan 22 2015 00:22:34,     227 sec) GpsServices()      Profile 0 GPS fix obtained in 34
sec.
(Jan 22 2015 00:22:34,     228 sec) GpsServices()      Fix: -124.0439  44.6210
01/22/2015 002210      4
(Jan 22 2015 00:23:09,     263 sec) GpsServices()      APF9 RTC skew (-1s) OK.
(Jan 22 2015 00:23:10,     263 sec) GpsServices()      GPS srvc complete.
(Jan 22 2015 00:23:34,     288 sec) WriteVitals()      Writing vitals to
"4693.1501220020".
(Jan 22 2015 00:23:41,     295 sec) UpLoad()           5260B 2 files 3 blks of max
4000B Fid 0-2
(Jan 22 2015 00:23:42,     295 sec) CLogin()           Connecting to primary host.
(Jan 22 2015 00:23:57,     311 sec) CLogin()           Connection 1 established in 15
sec.
(Jan 22 2015 00:24:15,     329 sec) login()            ACK rcvd.
(Jan 22 2015 00:24:15,     329 sec) login()            TX of platform ID (Q003)
success.
(Jan 22 2015 00:24:15,     329 sec) CLogin()           Logged to host. [in 18 seconds]
(Jan 22 2015 00:24:17,     331 sec) UpLoadFile()        4693.1501220020. 725B blk 1/1
total.
(Jan 22 2015 00:24:18,     332 sec) ChkResendReq()      Land rsp: \nccmds
(Jan 22 2015 00:24:22,     336 sec) UpLoad()           Reading msg from land.
\n@@@[0x17][0xe4][0x00][0x18]C[0x01][0x01]Senddata() [0xC9C9]
(Jan 22 2015 00:24:23,     337 sec) UpLoad()           Removing uploaded file
4693.1501220020
(Jan 22 2015 00:24:24,     338 sec) RemoveFile()        Upload OK. Remove file
4693.1501220020
(Jan 22 2015 00:24:24,     338 sec) UpLoad()           CRC chk valid. Rcvd blk 1/1,
length 24, Files left =1
(Jan 22 2015 00:24:24,     338 sec) UpLoad()           Still 1 more files files. 'data'
sent.
(Jan 22 2015 00:24:25,     339 sec) UpLoad()           4567B 1 files 2 blks of max
4000B Fid 0-0
(Jan 22 2015 00:24:29,     343 sec) UpLoadFile()        4693.000.log. 4029B blk 1/2
total.
(Jan 22 2015 00:24:46,     360 sec) UpLoadFile()        4693.000.log. 572B blk 2/2
total.
(Jan 22 2015 00:24:47,     361 sec) ChkResendReq()      Land rsp: ccmds
(Jan 22 2015 00:24:50,     364 sec) UpLoad()           Reading msg from land.
\n@@@[0x17][0xe4][0x00][0x18]C[0x01][0x01]Senddata() [0xC9C9]
(Jan 22 2015 00:24:51,     365 sec) UpLoad()           Removing uploaded file
4693.000.log
(Jan 22 2015 00:24:52,     366 sec) RemoveFile()        Upload OK. Remove file
4693.000.log
(Jan 22 2015 00:24:52,     366 sec) UpLoad()           CRC chk valid. Rcvd blk 1/1,
length 24, Files left =0
(Jan 22 2015 00:24:52,     366 sec) UpLoad()           'done' sent. No more file to
send 0
(Jan 22 2015 00:24:53,     367 sec) ChkResendReq()      Land rsp: ccmds
(Jan 22 2015 00:24:55,     369 sec) UpLoad()           Reading msg from land.
\n@@@[0x17][0xe4][0x00][0x18]C[0x01][0x01]Senddata() [0xC9C9]
(Jan 22 2015 00:24:56,     370 sec) UpLoad()           CRC chk valid. Rcvd blk 1/1,
length 24, Files left =0
(Jan 22 2015 00:24:57,     371 sec) UpLoad()           'done' sent. No more file to
send 0
(Jan 22 2015 00:24:57,     371 sec) ChkResendReq()      Land rsp: ccmds
(Jan 22 2015 00:24:59,     373 sec) UpLoad()           Reading msg from land.
\n@@@[0x17][0xe4][0x00][0x18]C[0x01][0x01]Senddata() [0xC9C9]
(Jan 22 2015 00:25:00,     374 sec) UpLoad()           CRC chk valid. Rcvd blk 1/1,
length 24, Files left =0
(Jan 22 2015 00:25:01,     375 sec) UpLoad()           'done' sent. No more file to
send 0
(Jan 22 2015 00:25:01,     375 sec) ChkResendReq()      Land rsp: ddone
Hung up. 0/3
(Jan 22 2015 00:25:11,     385 sec) UpLoad()           Total files uploaded: 1
(Jan 22 2015 00:25:12,     386 sec) Telemetry()        Tlmtry cycle done:PrfId=0
ConnAtmpts=1 Conn_St=1

```



```

(Jan 22 2015 00:25:12,      386 sec) Apf9PowerOff()      Sleep period (-80sec) not in
valid range [5sec,7Hr].      Resetting to 5sec.
(Jan 22 2015 00:25:19,      392 sec) MissionControlAgent Go descend after one last
telemetry.
(Jan 22 2015 00:25:25,      399 sec) TelemetryInit()      Profile 0. (Apf9i FwRev:
15012114)
(Jan 22 2015 00:25:39,      413 sec) AirSystem()
BatV[185cnt,14.3V]Amp[73cnt,29.4mA]BrmtrP[132cnt,8.3"Hg]Run-Tm[5s]
(Jan 22 2015 00:26:14,      448 sec) GpsServices()      GPS almanac is current.
(Jan 22 2015 00:26:54,      487 sec) GpsServices()      Profile 0 GPS fix obtained in 39
sec.
(Jan 22 2015 00:26:54,      488 sec) GpsServices()      Fix: -124.0439  44.6209
01/22/2015 002630  6
(Jan 22 2015 00:27:29,      523 sec) GpsServices()      APF9 RTC skew (-1s) OK.
(Jan 22 2015 00:27:30,      523 sec) GpsServices()      GPS srvc complete.
(Jan 22 2015 00:27:55,      548 sec) WriteVitals()      Writing vitals to
"4693.000.msg".
(Jan 22 2015 00:28:01,      555 sec) UpLoad()      4812B 2 files 3 blks of max
4000B Fid 1-3
(Jan 22 2015 00:28:02,      555 sec) CLogin()      Connecting to primary host.
(Jan 22 2015 00:28:21,      575 sec) CLogin()      Connection 1 established in 19
sec.
(Jan 22 2015 00:28:39,      593 sec) login()      ACK rcvd.
(Jan 22 2015 00:28:39,      593 sec) login()      TX of platform ID (Q003)
success.
(Jan 22 2015 00:28:39,      593 sec) CLogin()      Logged to host. [in 18 seconds]
(Jan 22 2015 00:28:41,      595 sec) UpLoadFile()      4693.000.msg. 725B blk 1/1
total.
(Jan 22 2015 00:28:42,      596 sec) ChkResendReq()      Land rsp: \nccmds
(Jan 22 2015 00:28:46,      600 sec) UpLoad()      Reading msg from land.
\n@@@[0x17][0xe4][0x00][0x18]C[0x01][0x01]Senddata() [0xC9C9]
(Jan 22 2015 00:28:47,      601 sec) UpLoad()      Removing uploaded file
4693.000.msg
(Jan 22 2015 00:28:48,      601 sec) RemoveFile()      Upload OK. Remove file
4693.000.msg
(Jan 22 2015 00:28:48,      602 sec) UpLoad()      CRC chk valid. Rcvd blk 1/1,
length 24, Files left =1
(Jan 22 2015 00:28:48,      602 sec) UpLoad()      Still 1 more files files. 'data'
sent.
(Jan 22 2015 00:28:49,      603 sec) UpLoad()      4116B 1 files 2 blks of max
4000B Fid 1-1
(Jan 22 2015 00:28:53,      607 sec) UpLoadFile()      4693.1501220020.log. 4036B blk
1/2 total.
(Jan 22 2015 00:29:10,      624 sec) UpLoadFile()      4693.1501220020.log. 121B blk
2/2 total.
(Jan 22 2015 00:29:11,      625 sec) ChkResendReq()      Land rsp: ccmds
(Jan 22 2015 00:29:13,      627 sec) UpLoad()      Reading msg from land.
\n@@@[0x17][0xe4][0x00][0x18]C[0x01][0x01]Senddata() [0xC9C9]
(Jan 22 2015 00:29:17,      631 sec) UpLoad()      Removing uploaded file
4693.1501220020.log
(Jan 22 2015 00:29:17,      631 sec) RemoveFile()      Upload OK. Remove file
4693.1501220020.log
(Jan 22 2015 00:29:18,      631 sec) UpLoad()      CRC chk valid. Rcvd blk 1/1,
length 24, Files left =0
(Jan 22 2015 00:29:18,      632 sec) UpLoad()      'done' sent. No more file to
send 0
(Jan 22 2015 00:29:19,      632 sec) ChkResendReq()      Land rsp: ccmds
(Jan 22 2015 00:29:20,      634 sec) UpLoad()      Reading msg from land.
\n@@@[0x17][0xe4][0x00][0x18]C[0x01][0x01]Senddata() [0xC9C9]
(Jan 22 2015 00:29:21,      635 sec) UpLoad()      CRC chk valid. Rcvd blk 1/1,
length 24, Files left =0
(Jan 22 2015 00:29:21,      635 sec) UpLoad()      'done' sent. No more file to
send 0
(Jan 22 2015 00:29:22,      636 sec) ChkResendReq()      Land rsp: ccmds
(Jan 22 2015 00:29:24,      638 sec) UpLoad()      Reading msg from land.
\n@@@[0x17][0xe4][0x00][0x18]C[0x01][0x01]Senddata() [0xC9C9]
(Jan 22 2015 00:29:25,      639 sec) UpLoad()      CRC chk valid. Rcvd blk 1/1,
length 24, Files left =0
(Jan 22 2015 00:29:26,      639 sec) UpLoad()      'done' sent. No more file to
send 0
(Jan 22 2015 00:29:26,      640 sec) ChkResendReq()      Land rsp: ddone

```

Hung up. 0/3

```

(Jan 22 2015 00:29:36,      650 sec) UpLoad()          Total files uploaded: 1
(Jan 22 2015 00:29:37,      650 sec) Telemetry()       Tlmtry cycle done:PrfId=0
ConnAttempts=1 Conn St=1
(Jan 22 2015 00:29:43,        6 sec) DescentInit()      Park profile 1 initiated at
651sec.
(Jan 22 2015 00:29:46,        8 sec) DescentInit()      Surface press: 52.2dB.
(Jan 22 2015 00:29:50,       13 sec) PistonMoveAbsWTO()  226->210 [77s, 14.3V, 0.129A,
CPT:77s]
(Jan 22 2015 00:31:10,       93 sec) MissionControlAgent  Descent p=52.2dB:  WISPR ON
(Jan 22 2015 00:32:28,       170 sec) MissionControlAgent  Data storage available 99.40
percent
(Jan 22 2015 00:32:35,       178 sec) Descent()          Press: 52.2 dB
(Jan 22 2015 00:34:44,       306 sec) QuecomGetDTX()
Ndtx=2,Nclk=360,Acm=2,MTK=1,MTHR=85,MRT=2339,MICI=0.100,P=52.1dB
(Jan 22 2015 00:34:47,       309 sec) Descent()          Press: 52.2 dB
(Jan 22 2015 00:39:44,       606 sec) QuecomGetDTX()
Ndtx=5,Nclk=872,Acm=7,MTK=1,MTHR=85,MRT=2262,MICI=0.100,P=52.2dB
(Jan 22 2015 00:39:47,       609 sec) Descent()          Press: 52.2 dB
(Jan 22 2015 00:44:44,       906 sec) QuecomGetDTX()
Ndtx=5,Nclk=928,Acm=12,MTK=0,MTHR=85,MRT=2416,MICI=0.100,P=52.2dB
(Jan 22 2015 00:44:47,       909 sec) Descent()          Press: 52.2 dB
(Jan 22 2015 00:49:44,      1206 sec) QuecomGetDTX()
Ndtx=5,Nclk=872,Acm=17,MTK=1,MTHR=85,MRT=2262,MICI=0.100,P=52.2dB
(Jan 22 2015 00:49:47,      1209 sec) Descent()          Press: 52.2 dB
(Jan 22 2015 00:54:44,      1506 sec) QuecomGetDTX()
Ndtx=5,Nclk=928,Acm=22,MTK=0,MTHR=85,MRT=2416,MICI=0.100,P=52.2dB
(Jan 22 2015 00:54:47,      1509 sec) Descent()          Press: 52.2 dB
(Jan 22 2015 00:59:44,      1806 sec) QuecomGetDTX()
Ndtx=5,Nclk=872,Acm=27,MTK=1,MTHR=85,MRT=2262,MICI=0.100,P=52.2dB
(Jan 22 2015 00:59:47,      1809 sec) Descent()          Press: 52.2 dB
(Jan 22 2015 01:04:44,      2106 sec) QuecomGetDTX()
Ndtx=5,Nclk=928,Acm=32,MTK=0,MTHR=85,MRT=2416,MICI=0.100,P=52.2dB
(Jan 22 2015 01:04:47,      2109 sec) Descent()          Press: 52.2 dB
(Jan 22 2015 01:09:44,      2406 sec) QuecomGetDTX()
Ndtx=5,Nclk=872,Acm=37,MTK=1,MTHR=85,MRT=2262,MICI=0.100,P=52.2dB
(Jan 22 2015 01:09:47,      2409 sec) Descent()          Press: 52.2 dB
(Jan 22 2015 01:14:44,      2706 sec) QuecomGetDTX()
Ndtx=5,Nclk=928,Acm=42,MTK=0,MTHR=85,MRT=2416,MICI=0.100,P=52.2dB
(Jan 22 2015 01:14:47,      2709 sec) Descent()          Press: 52.2 dB
(Jan 22 2015 01:19:44,      3006 sec) QuecomGetDTX()
Ndtx=5,Nclk=872,Acm=47,MTK=1,MTHR=85,MRT=2262,MICI=0.100,P=52.2dB
(Jan 22 2015 01:19:47,      3009 sec) Descent()          Press: 52.2 dB
(Jan 22 2015 01:24:44,      3306 sec) QuecomGetDTX()
Ndtx=5,Nclk=928,Acm=52,MTK=0,MTHR=85,MRT=2416,MICI=0.100,P=52.2dB
(Jan 22 2015 01:24:47,      3309 sec) Descent()          Press: 52.2 dB
(Jan 22 2015 01:29:41,      3604 sec) Descent()          Press: 52.2 dB
(Jan 22 2015 01:29:44,      3606 sec) ParkInit()
(Jan 22 2015 01:29:51,      3613 sec) QuecomGetDTX()
Ndtx=5,Nclk=872,Acm=57,MTK=1,MTHR=85,MRT=2262,MICI=0.100,P=52.2dB
(Jan 22 2015 01:34:44,      3906 sec) QuecomGetDTX()
Ndtx=5,Nclk=928,Acm=62,MTK=0,MTHR=85,MRT=2416,MICI=0.100,P=52.2dB
(Jan 22 2015 01:39:44,      4206 sec) QuecomGetDTX()
Ndtx=5,Nclk=872,Acm=67,MTK=1,MTHR=85,MRT=2262,MICI=0.100,P=52.2dB
(Jan 22 2015 01:44:44,      4506 sec) QuecomGetDTX()
Ndtx=5,Nclk=928,Acm=72,MTK=0,MTHR=85,MRT=2416,MICI=0.100,P=52.2dB
(Jan 22 2015 01:49:44,      4806 sec) QuecomGetDTX()
Ndtx=5,Nclk=872,Acm=77,MTK=1,MTHR=85,MRT=2262,MICI=0.100,P=52.2dB
(Jan 22 2015 01:54:44,      5106 sec) QuecomGetDTX()
Ndtx=5,Nclk=928,Acm=82,MTK=0,MTHR=85,MRT=2416,MICI=0.100,P=52.2dB
(Jan 22 2015 01:59:44,      5406 sec) QuecomGetDTX()
Ndtx=5,Nclk=872,Acm=87,MTK=1,MTHR=85,MRT=2262,MICI=0.100,P=52.2dB
(Jan 22 2015 02:04:44,      5706 sec) QuecomGetDTX()
Ndtx=5,Nclk=928,Acm=92,MTK=0,MTHR=85,MRT=2416,MICI=0.100,P=52.3dB
(Jan 22 2015 02:09:44,      6006 sec) QuecomGetDTX()
Ndtx=5,Nclk=872,Acm=97,MTK=1,MTHR=85,MRT=2262,MICI=0.100,P=52.3dB
(Jan 22 2015 02:14:44,      6306 sec) QuecomGetDTX()
Ndtx=5,Nclk=928,Acm=102,MTK=0,MTHR=85,MRT=2416,MICI=0.100,P=52.3dB
(Jan 22 2015 02:19:44,      6606 sec) QuecomGetDTX()
Ndtx=5,Nclk=872,Acm=107,MTK=1,MTHR=85,MRT=2262,MICI=0.100,P=52.3dB

```

```

(Jan 22 2015 02:24:44,      6906 sec) QuecomGetDTX()
Ndtx=5,Nclk=928,Acm=112,MTK=0,MTHR=85,MRT=2416,MICI=0.100,P=52.3dB
(Jan 22 2015 02:29:50,      7213 sec) ParkTerminate()      Pist Pos:210 Vac:105 Vq:185 Aq:8
VSeascan:185 ASeascan:9
(Jan 22 2015 02:29:53,      7215 sec) ParkTerminate()      PT: 52.4dB 23.9500C
(Jan 22 2015 02:29:56,      7218 sec) ProfileInit()        PrfId:001 Press:52.4dbar
pTable[65]:50dbar
(Jan 22 2015 02:30:01,      7223 sec) QuecomGetDTX()
Ndtx=5,Nclk=872,Acm=117,MTK=1,MTHR=85,MRT=2262,MICI=0.100,P=52.4dB
(Jan 22 2015 02:30:04,      7226 sec) PistonMoveAbsWTO()    210->226 [30s, 14.2V, 0.141A,
CPT:107s]
(Jan 22 2015 02:30:49,      7271 sec) QuecomGetDTX()
Ndtx=1,Nclk=208,Acm=118,MTK=0,MTHR=85,MRT=2724,MICI=0.100,P=27.7dB
(Jan 22 2015 02:30:54,      7276 sec) Profile()            Sample 0 at 25.1dB for bin 65
[50dB]. PT:23.4dB 24.0130C
(Jan 22 2015 02:30:55,      7277 sec) PistonMoveAbsWTO()    216->226 [15s, 14.2V, 0.141A,
CPT:122s]
(Jan 22 2015 02:31:19,      7302 sec) MissionControlAgent  Crossed WISPR-PWR-OFF depth
p=11.1dB: WISPR OFF
(Jan 22 2015 02:31:41,      7324 sec) SurfaceDetect()      SurfacePress:20.0dB Press:6.2dB
PistonPos:219
(Jan 22 2015 02:31:42,      7324 sec) PistonMoveAbsWTO()    219->226 [34s, 14.3V, 0.137A,
CPT:156s]
(Jan 22 2015 02:32:23,      7366 sec) TelemetryInit()      Profile 1. (Apf9i FwRev:
15012114)
(Jan 22 2015 02:38:41,      7744 sec) AirSystem()
BatV[185cnt,14.3V]Amp[71cnt,28.6mA]BrmtrP[130cnt,7.7"Hg]Run-Tm[25s]
(Jan 22 2015 02:39:10,      7773 sec) GpsServices()        GPS almanac is current.
(Jan 22 2015 02:39:44,      7806 sec) GpsServices()        Profile 1 GPS fix obtained in 32
sec.
(Jan 22 2015 02:39:44,      7806 sec) GpsServices()        Fix: -124.0439 44.6210
01/22/2015 023920 3
(Jan 22 2015 02:40:19,      7842 sec) GpsServices()        APF9 RTC skew (-1s) OK.
(Jan 22 2015 02:40:20,      7842 sec) GpsServices()        GPS srvc complete.
(Jan 22 2015 02:40:44,      7867 sec) WriteVitals()        Writing vitals to
"4693.001.msg".
(Jan 22 2015 02:40:51,      7874 sec) UpLoad()            11251B 2 files 4 blks of max
4000B Fid 0-2
(Jan 22 2015 02:40:52,      7874 sec) CLogin()            Connecting to primary host.
(Jan 22 2015 02:41:54,      7936 sec) chat()              Expected string [CONNECT] not
received.
(Jan 22 2015 02:41:54,      7936 sec) modem_connect()      Modem command
[ATDTxxxxxxxxxxxxxx] failed.
(Jan 22 2015 02:42:03,      7946 sec) CLogin()            Connection 1 established in 71
sec.
(Jan 22 2015 02:42:34,      7976 sec) expect()            Attempt to read prompt string
[ACK] failed.
(Jan 22 2015 02:43:04,      8006 sec) expect()            Attempt to read prompt string
[ACK] failed.
(Jan 22 2015 02:43:04,      8006 sec) CLogin()            Attempt to login to host failed.
Connect attempt failed. Try again. Power off/on modem.
(Jan 22 2015 02:43:18,      8020 sec) chat()              Expected string [OK] not
received.
(Jan 22 2015 02:43:18,      8020 sec) modem_hangup()      Modem command [+++~~~ATH0]
failed.
Retry
(Jan 22 2015 02:44:00,      8062 sec) CLogin()            Connecting to primary host.
(Jan 22 2015 02:44:12,      8075 sec) CLogin()            Connection 2 established in 12
sec.
(Jan 22 2015 02:44:43,      8105 sec) expect()            Attempt to read prompt string
[ACK] failed.
(Jan 22 2015 02:45:13,      8135 sec) expect()            Attempt to read prompt string
[ACK] failed.
(Jan 22 2015 02:45:13,      8135 sec) CLogin()            Attempt to login to host failed.
Connect attempt failed. Try again. Power off/on modem.
(Jan 22 2015 02:45:27,      8149 sec) chat()              Expected string [OK] not
received.
(Jan 22 2015 02:45:27,      8149 sec) modem_hangup()      Modem command [+++~~~ATH0]
failed.
Retry
(Jan 22 2015 02:46:09,      8191 sec) CLogin()            Connecting to primary host.

```

```

(Jan 22 2015 02:46:25,      8207 sec) CLogin()           Connection 3 established in 16
sec.
(Jan 22 2015 02:46:41,      8223 sec) login()           ACK rcvd.
(Jan 22 2015 02:46:41,      8223 sec) login()           TX of platform ID (Q003)
success.
(Jan 22 2015 02:46:41,      8224 sec) CLogin()           Logged to host. [in 16 seconds]
(Jan 22 2015 02:46:43,      8226 sec) UploadFile()       4693.001.msg. 1863B blk 1/1
total.
(Jan 22 2015 02:46:45,      8228 sec) ChkResendReq()      Land rsp: \nccmds
(Jan 22 2015 02:46:52,      8234 sec) Upload()           Reading msg from land.
\n@@@[0xb0][0x92][0x00]$C[0x01][0x01]ActivateRecoveryMode() [0x9848]
(Jan 22 2015 02:46:53,      8236 sec) Upload()           Removing uploaded file
4693.001.msg
(Jan 22 2015 02:46:54,      8236 sec) RemoveFile()       Upload OK. Remove file
4693.001.msg
(Jan 22 2015 02:46:54,      8236 sec) Upload()           CRC chk valid. Rcvd blk 1/1,
length 36, Files left =1
(Jan 22 2015 02:46:54,      8237 sec) Upload()           Copying good command. length 36
Command # 1
(Jan 22 2015 02:47:00,      8243 sec) Upload()           Still 1 more files files. 'data'
sent.
(Jan 22 2015 02:47:01,      8244 sec) Upload()           9417B 1 files 3 blks of max
4000B Fid 2-2
(Jan 22 2015 02:47:05,      8248 sec) UploadFile()       4693.001.log. 4029B blk 1/3
total.
(Jan 22 2015 02:47:25,      8267 sec) UploadFile()       4693.001.log. 4005B blk 2/3
total.
(Jan 22 2015 02:47:43,      8285 sec) UploadFile()       4693.001.log. 1422B blk 3/3
total.
(Jan 22 2015 02:47:45,      8287 sec) ChkResendReq()      Land rsp: ccmds
(Jan 22 2015 02:47:50,      8292 sec) Upload()           Reading msg from land.
\n@@@[0x17][0xe4][0x00][0x18]C[0x01][0x01]Senddata() [0xC9C9]
(Jan 22 2015 02:47:51,      8294 sec) Upload()           Removing uploaded file
4693.001.log
(Jan 22 2015 02:47:52,      8294 sec) RemoveFile()       Upload OK. Remove file
4693.001.log
(Jan 22 2015 02:47:52,      8294 sec) Upload()           CRC chk valid. Rcvd blk 1/1,
length 24, Files left =0
(Jan 22 2015 02:47:52,      8295 sec) Upload()           'done' sent. No more file to
send 0
(Jan 22 2015 02:47:53,      8295 sec) ChkResendReq()      Land rsp: ccmds
(Jan 22 2015 02:47:55,      8297 sec) Upload()           Reading msg from land.
\n@@@[0x17][0xe4][0x00][0x18]C[0x01][0x01]Senddata() [0xC9C9]
(Jan 22 2015 02:47:56,      8299 sec) Upload()           CRC chk valid. Rcvd blk 1/1,
length 24, Files left =0
(Jan 22 2015 02:47:56,      8299 sec) Upload()           'done' sent. No more file to
send 0
(Jan 22 2015 02:47:57,      8300 sec) ChkResendReq()      Land rsp: ddone
Hung up. 0/3
(Jan 22 2015 02:48:15,      8317 sec) Upload()           Creating mission.cfg
(Jan 22 2015 02:48:21,      8323 sec) Upload()           Total files uploaded: 1
(Jan 22 2015 02:48:21,      8323 sec) Telemetry()        Tlmtry cycle done:PrfId=1
ConnAtmpts=4 Conn St=3
(Jan 22 2015 02:48:21,      8324 sec) TelemetryTerminate() Parsing new mission config.
(Jan 22 2015 02:48:23,      8325 sec) configure()        Parsing configurators in
"mission.cfg".
(Jan 22 2015 02:48:26,      8328 sec) configure()        ActivateRecoveryMode() [0x9848]
[ActivateRecoveryMode()].
(Jan 22 2015 02:48:26,      8329 sec) configure()        Configuration CRCs and syntax
OK.
(Jan 22 2015 02:48:27,      8329 sec) ConfigSupervisor() Configuration accepted.
(Jan 22 2015 02:48:27,      8330 sec) configure()        Recovery mode activated.
[itimer:8329, alarm:7667]
(Jan 22 2015 02:48:39,      8341 sec) Apf9Init()          Asynchronous wake-up detected
(ie., wake-up not initiated by alarm signal).
Entering Command Mode [ApfId 4693, FwRev 15012114]
> k
WARNING: Deactivating the mission renders the float nondeployable
        until the mission is reactivated. Kill the mission anyway?

Type 'Y' to kill the mission or any other key to abort. y

```

WARNING: Mission deactivated; float is not deployable until the mission is reactivated.

IX. Frequently asked question (FAQ)

How to start the QUEphone mission; magnet swipe or serial communication?

- 1) Magnet is convenient in the field, but the field operator has no idea what QUEphone is doing unless the remote operator informs him/her. If there is no communication with the land and not serial communication with Apf9, only way to tell if it is working or not is to listen to the hydraulic motor, which could be difficult on the noisy small boat. Also the field operator can not tell if the telemetry is working or not.
- 2) The best way is to connect to the QUEphone's serial port by SAIL and observe what it is doing as the pre-launch choir progresses on the boat.

How to recover the QUEphone?

Remote operator sends a recovery command by pasting a recovery activation and tow Senddata() command at Rudics site

```
ActivateRecoveryMode() [0x9848]
Senddata() [0xC9C9]
Senddata() [0xC9C9]
```

QUEphone typically sends two files (and possibly three if the log file is long). The last two commands ensure that no files would be left unsent. After receiving a recovery command, QUEphone stays at the surface and transmits its GPS location at the interval specified in the mission profile. After successful recovery, the field operator connects the SAIL and PC, and type carriage return until it responds. When the prompt appears, type 'k' to kill the mission and 'y' to confirm.

Can we restart mission after recovery mode is activated?

There are two ways.

- 1) By connecting to the SAIL serial port of QUEphone. Type 'k' to the kill the mission and type 'e' to restart the new mission. This method is fast and can be repeated as many as you need, but needs a PC and SAIL. This is the safest way we recommend.
- 2) By remotely activating from Rudics. An remote operator can paste a mission command on Rudics site such as


```
ParkPistonPos(47) [0xF7AE]
```

 and send it. When it connects to Rudics next time, it restarts a new mission. Another command can be used is


```
User(E1D99999P0050) [0xD2EB]
```

 This command enables (letter "E") the mission. It also sets the gain 1 and WISPR turn on/off depth to 50. You can change these gain and depth but CRC number must be consistent. Method 2 is slower because it must wait for the telemetry cycle and also works only once. If this method is used, it restarts mission only once.

Can I change the parking depth remotely during the mission?

Yes, but you will have to change seven additional mission parameters including the park position and six time-related parameters. For example, to change the parking

depth from 500 to 1000 m, the following seven commands have to be sent. To make the surface time to be the same 17 UTC, it also needs a new TimeOfDay(780) command as well.

ParkPistonPos(34) [0x276D]
 ParkPressure(1000) [0x899C]
 DownTime(405) [0x360E]
 UpTime(445) [0xD4D4]
 AscentTimeOut(320) [0x0CFF]
 ParkDescentTime(280) [0x6795]
 DeepProfilePressure(1000) [0x20B5]
 TimeOfDay(780) [0x5017]

The reason for all these changes is that the length of time requires to descent from 0 to 1000 m and ascend from 1000 m to 0 m are different (based on 0.08 cm/s speed) from the 500-m parking depth, which changes the duration of each phase. You must be careful changing the parking depth because if one parameter is not right, Apf9's sanity check program rejects all and it may stop the mission. Matlab code, "MissionTime.m" helps you to check the new mission parameters. The best way to check the new configuration is to test on another APEX system in your lab and run sanity check (by typing 'z') before sending the commands.

Be aware that the parking piston position is system independent. General rule of thumb for the new piston position Ps is

$$P_s \approx P_p + \nabla d / 12.5$$

where Pp is the current park piston position given by Teledyne Webb and ∇d is the pressure difference in meter from the initial parking depth (∇d = initial parking depth - new parking depth). For example if the current parking depth 800 m and piston position is 35, $P_s = 35 - 200/12.5 = 19$.

How can I calculate the CRC for the command?

Go to <https://www.lammerbries.nl/comm/info/crc-calculation.html> and use the output of generated by CRC-CCITT (0x1D0F).

How can I make the battery last longer?

1. Make the parking depth shallower. APEX uses a lot of power for the hydraulic motor. Making the parking depth shallow not only uses less battery power but the water is warmer there which makes the alkaline battery life longer. For example by making the parking depth from 1000 m to 500 m makes the alkaline battery life from 14 days to 20 days.
2. Change the battery from alkaline to lithium. Lithium battery has an energy/weight ratio of 2.5 of the alkaline. If it is a 500-m mission, it would last the mission to ~45 days. Part number is 3PD1243 Rev B available from Teledyne Webb. Three Lithium packs and one D-cell alkaline battery pack fit in QUEphone.

How accurate the weight measurement must be?

QUEphone weighs approximately 25.5 kg. Teledyne recommends to adjust its air weight to the recommended value within ± 2 grams. However a typical parking depth

of the QUEphone is 1000 m (APEX float is rated at 2000 m). Therefore even if it is 10 grams off, the parking depth is by 200 m (800 or 1200 m). 1200 m would not crash the hull. Since the float will eventually adjust the depth to the programmed depth by itself, weight off by 5 grams or so is not a big deal.

What is the daily drift of the QUEphone by the ocean current?

It depends on the area and season. Typical drifts we have seen in AUTC in June 2010 were 2-3 km/day. QUTR range (WA) in the spring of 2015 was 2 to 7 km/day. At SCORE range (CA) in the winter of 2011 was approximately 10 km/day. At the same range in 2015 it was approximately 5 km/day. Kona off Hawaii in 2009 was approximately 4 km/day.

What is the recommended spacing of multiple QUEphones?

Maximum detection range of beak whale is approximately 5 km. If multiple QUEphones are deployed to monitor a designated area, optimum spacing between the QUEphones would be between 7.5 to 10 km. Ocean current is complicated especially if the islands are nearby. They may drift together in parallel for a few days but will eventually separated or get closer and you may need to pick them up and redeploy after several days in order to maintain the distance.

How long the data storage lasts?

WISPR stores the acoustic data in FLAC format on a single 512 GB CF card. At 125 kHz sampling rate with FLAC 2 compression, it would last 45 days.

X. MissionTime.m (Matlab code)

```
%MissionTime.m
%Compute time parameters for QUEPhone
prompt='What is the parking depth? ';
ParkDepth=input(prompt);
vspeed=0.08;
Hour=60; %minutes

fprintf('\nParking depth      = %d m\n', ParkDepth);
AscentTimeOut=round(ParkDepth/vspeed/60 + 1*Hour+0.4);
fprintf('AscentTimeOut    >= %d min\n', AscentTimeOut);

UpTime=round(AscentTimeOut + 2*Hour);
fprintf('UpTime              >= %d min\n', UpTime);

ParkDescentTimeL=round(ParkDepth/vspeed/60);
DownTime=round(ParkDescentTimeL+2*Hour+0.4);
fprintf('DownTime            > %d min\n', DownTime);

ParkDescentTimeH=round(1.5*ParkDepth/vspeed/60)+1*Hour;
fprintf('%d min <= ParkDescentTime <= %d min\n', ParkDescentTimeL,
ParkDescentTimeH);

prompt='What is the redefined AscentTimeout? ';
AscentTimeOut=input(prompt);
UpTime=round(AscentTimeOut + 2*Hour+0.4);
fprintf('UpTime              >= %d min\n', UpTime);

%ParkDescentTimeL=round(ParkDepth/vspeed/60);
%ParkDescentTimeH=round(1.5*ParkDepth/vspeed/60)+1*Hour;
%fprintf('%d min <= ParkDescentTime <= %d min\n', ParkDescentTimeL,
ParkDescentTimeH);

prompt='What is the redfined ParkDescentTimeout? ';
ParkDescentTime=input(prompt);
DownTime=round(ParkDescentTime+2*Hour+1+0.4);
fprintf('DownTime            >%d min\n', DownTime);
%clear all;

prompt='What time zone in hour? (0-24, e.g., PDT=7) ';
TimeZone=input(prompt);

prompt='What local time do you want QUEphone to come up? (0-24)';
SurfaceTime=input(prompt);
Ascent=round(ParkDepth/vspeed/60+0.4);
ToD=SurfaceTime*Hour-Ascent+TimeZone*Hour;
fprintf('ToD                = %d min\n', ToD);
clear all;
```