Quake Global 社製オーブコム衛星通信端末

（Q1200SG/SM、Q1400）

プログラミングマニュアル

Ver.　D

2005 年 6 月 1 日

オーブコムジャパン株式会社

目　次

序

本マニュアルは、Quake Base Application を利用して Quake 製モデムをプログラムする方法を説明します。基本アプリは、イベントに対応してモデムがレスポンスアクションするようにするソフトウェアモジュールです。イベントとは、アナログ入力値が閾値を超えるとかユーザーコマンド受信とか所定時間経過などを指します。レスポンスアクションとは、メッセージを送ったり GPS 測位を開始したり一定時間電源遮断をしたりといった動作を指します。イベントとレスポンスアクションの関連を示すものをアプリケーションイベントテーブルといいます。このテーブルは基本アプリの基礎となるもので、簡単にオーブコム端末を色々な目的にプログラムする方法を教えてくれます。アプリケーションイベントテーブルの他にも、メッセージをフォーマットする方法やセンサーの読み値を取り入れる方法を示すテーブルがあります。これらは各々メッセージテーブル、センサーテーブルと呼ばれています。

全ての Quake 製端末は、ロガーポートと呼ばれる RS-232 シリアルポートを持っています。このポートから、端末の運用状況や衛星受信状況、信号強度、メッセージのステータスを知ることが出来ます。ロガーポートはウィンドウズのハイパーターミナルをはじめとして全ての端末エミュレータソフトと接続する事が出来、端末ソフトを通じて端末内の色々なユーティリティやプログラミングメニューに簡単にアクセスする事が出来ます。

ロガーポートで使用するプログラミングユーティリティを使うことで Quake 製端末で次のような事が出来ます。

- 1 台の端末の中に、メッセージの宛先と本文中のデータの形式を規定するメッセージテーブルを 100 種類まで作る事が出来ます。
- Geo-Fencing に利用する 100 箇所の位置情報（緯度/経度）を持つ GPS センサーテーブルを 16 種類まで作る事が出来ます。
- 高/低のアラーム閾値を設定したアナログセンサーテーブルを 16 種類まで作る事が出来ます。
- 高/低のアラーム閾値と経過時間計測値を設定したデジタルセンサーテーブルを 16 種類まで作る事が出来ます。
- テキストでのメール送受が出来ます。

ロガーポートを使ってプログラムを作るには、先ず端末に内蔵されている基本ソフトウェ

アの基礎を理解する事が必要です。「基本ソフトウェア」とは工場で全ての Quake 製端末にインストールされるソフトウェアの事です。

1. 基本ソフトウェアの概観

本章では Quake 製端末の基本ソフトウェアにある機能グループを解説します。基本ソフトウェアは次のような機能グループから成り立っています。

- アプリケーション/イベントハンドラータスク
- アプリケーションイベントテーブル及び同ユーティリティ
- メッセージテーブル及び同ユーティリティ
- センサーテーブル及び同ユーティリティ
- その他の機能

A) アプリケーション/イベントハンドラータスク

アプリケーション/イベントハンドラータスクは、データの収集、メッセージをフォーマットして送る事、そして無線で伝えられて来るコマンドへの対応やセンサー入力の状況監視、GPS 位置情報監視、外部機器制御といったユーザーが定める機能を果たします。アプリケーション/イベント取扱い機能は端末内の不揮発性メモリーに保管されているイベントテーブルやメッセージテーブル、センサーテーブルを使用します。これらのテーブルは、色々なイベントや状況、メッセージのフォーマットや ORBCOMM 網を通じた伝送経路、そして各種外部機器からの測定値に対して各アプリケーションで求められる動作が記述されています。

Quake 製端末は、イベントテーブルやメッセージテーブル、センサーテーブルを作成したり、編集したりする為のユーティリティを簡単に使えるように設計してあります。ORBCOMM に馴染みのある技術者であれば複雑なアプリケーションでも簡単に作る事が出来ます。

注: アプリケーションとは工場出荷後にインストールされるソフトウェアファイル全般を意味し、端末に必ずしも存在するものではありません。もしもこのモジュールがインストールされていなければ端末は単なる ORBCOMM モデムとして動作します。また、複雑なアプリケーション用に C 言語で独自のプログラムを作成して端末に搭載する事が出来ます。その様なアプリケーション用に Quake の C 言語 API を利用した開発環境が用意されています。

B) アプリケーションイベントテーブル及び同ユーティリティ

アプリケーションイベントテーブル（以下「イベントテーブル」と略します）はイベントの見出しと、特定のイベントが発生した時に求められるレスポンスアクションとで構成されています。イベントテーブルはコンパクトで簡単に編集が出来、リモートコマンドによって変更が可能です。

Quake は以下のイベントタイプを定義しました。幾つかのイベントタイプは複数のイベントを含んでいます。テーブル（Q1200SM/SG 及び Q1400 では ANALOG_ALARM、ANALOG、J1780_MEAS、J1708_ALARM は使用できません）

- POWER_ON
- TIMER
- POSITION_FIX
- MSG_ACK
- ANALOG_ALARM
- DIGITAL_ALARM
- POSITION_ALARM
- SAT_IN_VIEW
- ANALOG
- DIGITAL
- COUNTER
- USER_CMD
- MTS_DTR
- SER_MEAS
- SER_ALARM
- J1708_MEAS
- J1708_ALARM
- DIG_0
- DIG_4
- SPEED_ALARM

イベントテーブルのいずれのイベントにもレスポンスアクションを指定する事が出来ます。Quake はレスポンスアクションタイプを次のように定義しました。（Q1200SM/SG 及び Q1400 では READ_ANALOG、READ_J1708 は使用できません）

- CFG_DIGITAL
- READ_ANALOG
- READ_DIGITAL

- SET_DIGITAL
- SET_TIMER
- GET_POSITION
- POWER_DOWN
- MSG_ENQ
- SEND_MSG
- START_DL_ACQ
- SET_COUNTER
- INCR_COUNTER
- DECR_COUNTER
- READ_SERIAL
- COMMS_CMD
- SET_GEOFENCE
- DEL_GEOFENCE
- READ_J1708
- SAVE_DATA_FILE

これらのイベントやレスポンスアクションについては後述します。これらのイベントとレスポンスアクションによってより複雑な機能を作り上げる事が出来ます。

基本ソフトウェアのユーティリティはイベントテーブルを作成し、編集したり不揮発メモリーに保存したりする為に使います。イベントテーブルは PC から端末のシリアルポートを使ってロードしたりアンロードしたりする事も出来ます。

C) メッセージテーブルとユーティリティ

端末の不揮発メモリーはメッセージテーブルを 100 種類まで保存できます。メッセージテーブルは、各メッセージに対してフォーマットされたパラメータの位置と使用するビット数が定義されています。Quake は現在 ORBCOMM システムで使用できる全てのメッセージタイプをサポートしています。基本ソフトウェアのユーティリティを使ってメッセージテーブルを作成し、編集したり不揮発メモリーに保存したり出来ます。また、PC から端末のシリアルポートを使ってロードしたりアンロードしたりする事も出来ます。

Quake は以下のメッセージパラメータタイプを定義しています。（Q1200SM/SG 及び Q1400 では ANALOG_IN、J1780_DATA は使用できません）

- ANALOG_IN

- DIGITAL_IN
- LATITUDE
- LONGITUDE
- ALTITUDE
- CURRENT_TIME
- ELAPSED_TIME
- COUNTER_VALUE
- USER_DATA_BYTE
- SERIAL_DATA
- SPEED
- HEADING
- LAST_USER_CMD_DATA
- LAST_OB_MSG_SUBJ
- J1708_DATA
- FILE_DATA

D) センサーテーブルとユーティリティ

端末は5つのセンサータイプそれぞれに対して最大16のセンサーテーブルを不揮発メモリーに保存できます。センサータイプは、アナログ入力、デジタル入力、GPS位置デバイス、シリアルポート、J1708バスの5つです。J1708バスはQ2000でサポートされています。センサーテーブルの各種パラメータによって測定条件や警報条件（最大/最小値）、平均値の取り方などを定義する事が出来ます。

例えば、センサーの読み値から機器の異常を警告する事が出来ます。イベントテーブルにPOWER_ONイベントとREAD_ANALOG1アクションを入力します。アナログセンサーテーブル1で､アナログチャネル0を1秒毎に5回読んで平均値を返すように設定します。読値が完了したらANALOG1イベントが発生し、イベントテーブルではANALOG1イベントに対してSEND_MSG2アクションを実行するように設定します。（もしも読値がセンサーテーブルの警報閾値を越えている場合、アナログ機能はANALOG_ALARM1イベントも発生させます。これによってアプリケーションは警報状態に対して異なるレスポンスアクションを取る事が出来ます。）

基本ソフトウェアのユーティリティを使ってセンサーテーブルを作成し、編集したり不揮発メモリーに保存したり出来ます。また、PCから端末のシリアルポートを使ってロードしたりアンロードしたりする事も出来ます。

E) その他の機能

基本ソフトウェアに含まれるその他の機能は、ORBCOMM 仕様に従った端末の機能や種々の下位のシステム機能の実行、アプリケーションのサポートなどに必要な重要で複雑な機能を果たしています。

F) ORBCOMM 設定パラメータ

Quake のソフトウェアは OSI プロトコルのセッション、トランスポート及びネットワーク層を構成する ORBCOMM ソフトを内含しています。Quake はこのソフトモジュールを単なる TL（トランスポート層）として参照しています。Quake モデムのユーザーはこれらの詳細を気にすることはありませんが、TL オペレーションは幾つかの設定パラメータで決定されている事だけを記憶しておいてください。

アプリケーションの設定パラメータは TL のものとは全く関係がありません。アプリケーションはイベントテーブルとメッセージテーブル、センサーテーブルで動きます。

アプリケーションと TL は端末内の独立した機能です。アプリケーションは TL との間でメッセージを送受し、TL は無線回線を通じてメッセージを送受します。アプリケーションをプログラムする際、最初に TL が正しく設定されているかを確認する事が重要です。さもないとアプリケーションは期待したように動作しません。例えば ob_route パラメータはアウトバウンド（OB）のメッセージやコマンドをアプリケーションへ渡すのか MTS シリアルポートに渡すのか、或いは両方かを規定します。ob_route を 1 と設定し、イベントテーブルに何も記入しなければ端末は単なる ORBCOMM モデムとして機能します。ob_route を 0 と設定するとアウトバウンドのメッセージとコマンドおよびインバウンドメッセージの Ack 全てをアプリケーションで受け取る事が出来、イベントテーブルに従って対応する事ができます。ob_route をデフォルトの 2 にしておけば、アプリケーションと MTS ポートを同時に使用する事が出来ます。

通常、TL 設定はデフォルトのままにしておきます。しかし、これらを変更する必要がある場合はロガーポートコンフィグパラメータユーティリティを使用します。このユーティリティについては後述します。

アプリケーションを設定するには ORBCOMM 網や ORBCOMM シリアルインターフェースプロトコル、TL コンフィギュレーションパラメータを理解している必要があります。これらは以下のドキュメントに解説されています。

- ORBCOMM System Overview, Rev. C or greater
- ORBCOMM Serial Interface Specification, Rev. F or greater

・ ORBCOMM Gateway Customer Access Interface Specification Rev. C or greater

これらのドキュメントは ORBCOMM システムの微妙な点を理解したり複雑なアプリケーションのプログラミングやデバッグを行う上で非常に重要なものです。**Quake 端末をプログラムする前に必ずこれらのドキュメントに目を通して下さい。**また、新しいバージョンが出されますので、お持ちのドキュメントが最新版である事を確認してください。

2. ロガーポートとの接続

ロガーポートから Quake 端末をプログラムするには TV エミュレーションプログラム（ハイパーターミナルなど）が必要です。エミュレータと端末のロガーポートを接続してください。DB9 コネクターで接続するには開発キットか専用のケーブルが必要です。ターミナルソフトを以下のように設定します。

・ Baud Rate: **19200 Baud**
・ Data Bits, Parity, Stop Bits: **8 Data Bits, No Parity, 1 Stop Bit**
・ Flow Control: **None**

ターミナルソフトを設定したら適当な直流電源を繋ぎます（Q1200 及び Q1400 の場合では 12V、2A 以上)。ASCII 文字がスクロールするはずです。文字が現れない場合はシリアルポートの設定が間違っているか端末との接続が正しくないかのいずれかです。

3. さわってみましょう

Quake 端末のロガーポートに接続して電源を入れたらプログラミングが可能です。電源を入れた後、ターミナルのスクリーンのスクロールが止まらない事があります。このような場合には”d”、”l”（小文字の L）とタイプして下さい。これによってほとんどのデータロギングが終了します。
（Quake ロガーデータについては Appendix D を参照ください）

プログラミングを開始する場合は”U”を入力します。するとターミナル画面には次のように表示されます。

```
***Utility Mode enabled***
Utility mode will expire in 20 seconds
```

全てのプログラミングテーブル、即ちアプリケーションイベントテーブル、メッセージテ

ーブル、センサーテーブル、はユーティリティモードで作成、編集されます。このモードでは先ず次のキーの意味を理解しておく必要があります。

‘?’ Lists menu options （オプションメニューの表示）
‘A’ Application Event Table Utilities （アプリケーションイベントテーブル）
‘M’ Message Table Utilities （メッセージテーブル）
‘G’ View/Edit Desired Gateway List （ゲートウェイリストの表示/編集）
‘a’ AIN Sensor Table Utilities （アナログ入力センサーテーブル）
‘d’ DIO Sensor Table Utilities （デジタルセンサーテーブル）
‘e’ Elapsed Times (ets) Utilities （経過時間ユーティリティ）
‘g’ GPS Sensor Table Utilities （GPS センサーテーブル）
‘j’ J1708 Sensor Tale Utilities （J1708 センサーテーブル）
‘s’ Serial Sensor Table Utilities （シリアルセンサーテーブル）
‘l’ last (et) alarm Utilities （最終経過時間アラーム）
‘F’ File Transfer Utilities （ファイル転送）
‘S’ Send Email （電子メール送信）
‘R’ Read Email （受信メールの表示）
‘C’ Modify Config Parameters （設定パラメータの変更）
‘P’ GPS Test （GPS テスト）

それでは簡単なプログラムを作って、電源オンされるたびに「Hellow World」というタイトルの電子メールを出させるようにしましょう。先ずアプリケーションイベントテーブルを出しますので、”U”とタイプした後に”A”とタイプしましょう。スクリーン上に次の表示があるはずです。

```
***Utility Mode enabled*** (will expire in 20 Secs)
A
==========  Application Event Table  ==========
EVENT                    RESPONSE ACTION
=========================================
CHOOSE FROM THE FOLLOWING:
‘a’ Add a New Event      ‘d’ Delete an Event      ‘e’ Edit an Event
‘q’ Quit
>
```

アプリケーションイベントユーティリティが起動されるとアプリケーションイベントテー

ブルがいつも表示されます。このケースではまだ何のイベントもリストされていませんので、端末には何もアプリケーションが無いことが分かります。もしもアプリケーションがある場合は'd'のキーを必要な回数選択して全てのイベントを消去してください。こうすることで新しいアプリケーションを書く用意が出来ました。

上の状態で"a"を選択します。スクリーンには次の表示が現れます。

```
SELECT EVENT:
'0'   POWER_ON          '2'   TIMER             '3'   POSITION_FIX
'4'   MSG_ACK           '5'   ANALOG_ALARM      '6'   DIGITAL_ALARM
'7'   POSITION_ALARM    '8'   SAT_IN_VIEW       '9'   ANALOG
'10' DIGITAL           '11' COUNTER           '12' USER_CMD
'13' MTS_DTR           '14' SER_MEAS          '15' SER_ALARM
'17' J1708_MEAS        '18' J1708_ALARM       '19' DIG_0
'20' DIG_4             '21' SPEED_ALARM
```

この例では、端末が電源オンされた時にメッセージを送りますので、「POWER-ON」イベントを選ぶ為に"0"を入力します。次のような表示が現れます。

```
SELECT RESPONSE ACTION:
'0'   CFG_DIGITAL       '1'   READ_ANALOG       '2'   READ_DIGITAL
'3'   SET_DIGITAL       '4'   SET_TIMER         '5'   GET_POSITION
'6'   POWER_DOWN        '7'   MSG_ENQ           '8'   SEND_MSG
'9'   START_DL_ACQ      '10' SET_COUNTER        '11' INCR_COUNTER
'12' DECR_COUNTER       '13' READ_SERIAL        '14' COMMS_CMD
'15' SET_GEOFENCE       '16' DEL_GEOFENCE       '17' READ_J1708
'18' SAVE_DATA_FILE
```

「POWER-ON」イベントに対してのレスポンスアクションを聞いていますので、"9"を選んでダウンリンク捕捉を開始します。これは端末のオーブコム受信機にオーブコム衛星からのダウンリンクをサーチさせるものです。メッセージを送ろうとする時には必ずダウンリンクが必要です。"9"を入力すると次の様に表示されます。

```
ENTER RESPONSE PRM1>
```

レスポンスアクション「START_DL_ACQ」のパラメータ 1 を聞いています。

「START_DL_ACQ」のパラメータ 1 は最初にサーチするダウンリンクチャネルです（Appendix B を参照して下さい)。”0”を入力することで全てのチャネルをサーチします。”0”を入力すると、次の様に表示されます。

ADD ADDITIONAL RESPONSES FOR THIS EVENT? (y/n)>

追加のレスポンスアクションを入力するかという問いです。「Hello World」というメッセージを送るので、”y”を入力します。画面は次のレスポンスアクションの入力を待っています。”8”と入力して「SEND_MSG」を選びます。画面は「SEND_MSG」の２つのパラメータ入力を促します。Appendix B を参照しながら、最初のパラメータとしてメッセージ数を、二つ目として送信不成功の場合に電源オフ時にはメッセージを不揮発メモリーに保存するかどうかを選択します。最初のパラメータには”0”（メッセージテーブル 0 を使用するという意味）を、二つ目のパラメータにも”0”（送信不成功の場合に不揮発メモリーに保存するという意味）を入力します。画面は再び追加のレスポンスアクション入力を問い合わせていますので、今回は”n”を選びます。新しく作られたイベントテーブルが画面に表示されます。

```
==========  Application Event Table  ===========
EVENT              RESPONSE ACTION
POWER_ON            START_DL_ACQ 0
-                  SEND_MSG 0 [0]
================================================
CHOOSE FROM THE FOLLOWING:
'a' Add a New Event   'd' Delete an Event   'e' Edit an Event
's' Save Event Table  'r' Revert to saved   'q' Quit (without Saving)
>
```

これでアプリケーションイベントテーブルが完成しました。”s”を入力してこのテーブルを不揮発メモリーにセーブし、”q”でこのアプリケーションイベントテーブルユーティリティとユーティリティモードから抜け出します。

アプリケーションイベントテーブルは完成しましたが、後少しやることが残っています。Message0（=メッセージテーブル 0）が定義されていません。メッセージはメッセージテーブルに作ります。メッセージテーブルに入力する為に、”U”そして”M”と入力します。画面は次の様になります。

```
CHOOSE FROM THE FOLLOWING:
```

```
'a' Add a Message Table          'd' Delete a Message Table
'v' Display a Message Table    'q' Quit
>
```

“a”を選んで新規メッセージテーブルを追加する事にし、”0”でメッセージテーブル 0 を指定します。するとメッセージタイプの入力画面になります。

```
SELECT MSG TYPE:
'0' REPORT            '1' MESSAGE           '2' GLOBALGRAM
'3' DEFAULT_REPORT     '4' DEFAULT_MESSAGE    '5' POSITION_REPORT
```

今回は Message を送りますので”１”を選んで下さい。次に自動ローミング、レポート自動変換オプション、その他のオーブコム特有のメッセージパラメータである gwy_id、polled、ack_level、priority、msg_body_type を聞いてきます。自動ローミングとレポート自動変換に就いては”disable”を選び、正しい gwy_id（米国なら”１”、日本なら”130”など）とその他のパラメータ値を入力します。画面が更に色々なパラメータを聞いてきますので、受信者数（1)、宛先アドレス（電子メールアドレスをフルに入力して下さい)、メッセージの表題（Hello World)、その他のパラメータ（0）を入力します。これでこのメッセージが完成し、メッセージテーブル 0 が確認の為に表示されます。

```
============= Msg 00 Message Table =============
Message Type        MESSAGE
Auto-Roaming        Disabled
Report Auto-Convert  Disabled
gwy_id             Default
polled             Default
ack_level          Default
priority           Default
msg_body_type       Default
Recipient Qty       1
Recipient List      john.doe@makeabuck.com
Subject            Hello World
================================================
```

これでこのアプリケーションは完成です。適切なアンテナがオーブコム端末に接続されている事、そのアンテナが充分に天空を見渡せることを確認して下さい。端末の電源を一旦切ってから再投入して下さい。数分後に「Hello World」メールを受信するでしょう。

4. アプリケーションイベントテーブル

上記の「Hello World」の例でアプリケーションイベントテーブルを紹介しましたが、Quake のオーブコム端末をプログラムする際に通常必要な作業は、ロガーポート画面の指示に従って入力する事と、Appendix A 及び B を参照してイベントやレスポンスアクションを熟知する事だけです。アプリケーションイベントテーブルをプログラムする際には以下の点に注意して下さい。

1. アプリケーションテーブルに記載されるイベントの順番は不同です。そのイベントがテーブルのどこに書かれていても、そのイベントが発生した時には対応する動作が常に行われます。

2. アプリケーションイベントテーブルを編集した時には必ず不揮発メモリーにセーブして下さい。

3. データを含むメッセージを送る場合、メッセージを送る前にデータを収集するようにアプリケーションを作成して下さい。例えば、GPS 位置データが求められた場合、GPS の測位終了（POSITION_FIX イベント）が得られてからこのデータ付きメッセージを送るようにして下さい。

4. 常にイベントテーブルには START_DL_ACQ レスポンスアクションがあることを確認して下さい。さもないと端末は衛星のダウンリンク信号を捕捉せず、結果として、メッセージの送受信が出来ません。START_DL_ACQ を POWER_ON イベントに対する動作として常に書き込む事を推奨します。

5. 何らかのイベントが発生してもアプリケーションイベントテーブルに記載が無い場合は、端末はそれを無視します。

5. メッセージテーブル

メッセージテーブルは、メッセージを作る為に必要なパラメータを列記する場所です。ユーティリティモードに入って”M”と入力してメッセージテーブルユーティリティに入ります。画面には次の様に表示されます。

```
M
```

```
CHOOSE FROM THE FOLLOWING:
'a' Add a Message Table      'd' Delete a Message Table
'v' Display a Message Table   'q' Quit
>
```

ここからは新規作成や編集などが出来ますが、以下の点に留意してください。

1. 新規テーブルを入力中にミスをした場合にはそのテーブルを削除して新たにやり直して下さい。
2. メッセージパラメータは、送信されるメッセージ中にメッセージテーブルに記された順番で並べられます。
3. 全てのメッセージデータは LSB が最初に来るようにフォーマットされるので、受信したメッセージのデコードも同様に行って下さい。
4. テーブルの入力が完了し、入力された値が適正な範囲にあれば、完成したテーブルが画面に表示され自動的に不揮発メモリーにセーブされます。
5. Quake 端末は最大 100 のメッセージテーブルをサポートします。

Quake Logger で表示されるメッセージテーブルの例を下記します。

```
============= Msg 00 Message Table =============
Message Type        MESSAGE
Auto-Roaming        Disabled
Report Auto-Convert  Disabled
gwy_id             Default
polled             Default
ack_level           Default
priority           Default
msg_body_type       Default
Recipient Qty       2
Recipient List      O/R 1
                  john.doe@makeabuck.com
Subject             [NONE]
Parameter 1         USER_DATA_BYTE 0  (8 bits)
Parameter 2         LATITUDE  (24 bits)
Parameter 3         LONGITUDE  (24 bits)
================================================
```

テーブル中に要求されるパラメータの多くはオーブコムに特有の、gwy_id、polled、priority、msg_body_type といったパラメータです。これらに就いてはオーブコム社のドキュメントに詳説されていますが、Quake 特有のものも含めて以下に概略を説明します。

Message Type：
選択肢として Report、Message、GlobalGram、Default Report、Default Message そして Position Report があります。Report はオーブコム特有のコンパクトなメッセージ形式で、６バイトのデータを送ることが出来ます。送るデータが６バイト以下で GlobalGram を必要としない場合には Report が良いでしょう。Default Report というのは gwy_id、polled、ack_level、priority そして msg_body_type パラメータが事前に指定されたデフォルト値を取る Report の事です。Message はオーブコム特有のメッセージ形式で、８K バイトまでのデータを送ることが出来ます。Default Message は Default Report と同様に指定されたオーブコムパラメータを使用するものです。GlobalGram はゲートウェイと繋がっていない衛星を使って送るメッセージです。

Auto-Roaming（自動ローミング）：
このパラメータに就いては Enabled（機能オン）或いは Disabled（機能オフ）を選択します。この機能は以下の条件のいずれかに当てはまるアプリケーションを開発する際に利用します。

1. 使用するオーブコム端末が、主に利用するゲートウェイのサービス範囲から外へ出ることが多い。
2. 使用するオーブコム端末が複数のゲートウェイに登録されており、いずれのゲートウェイからでもメッセージを送受出来るようにしたい。
3. 使用するオーブコム端末がゲートウェイから離れた場所にあって常時衛星経由で回線を繋ぐ事が難しく、ゲートウェイとの回線の状態によってメッセージを送る方法を GlobalGram 或いは Message/Report から自動的に選択したい。

GlobalGram 以外のメッセージ形式に対してこの機能がオンとなっている場合、端末はどのゲートウェイが「見えて」いるかによってメッセージの宛先ゲートウェイを自動的に修正します。但し、Message 或いは Report 形式のメッセージを GlobalGram にする事は出来ません。GlobalGram 形式の場合、この機能はもっとフレキシブルです。自動ローミング機能をオンにした GlobalGram メッセージは、ゲートウェイが一つも見えない状態であれば、GlobalGram として送られます。もしもこの端末が登録されている（Desired）ゲートウェイが見える場合には自動的に Message に変換されて送信されます。（Desired ゲートウェイリストの編集に就いては Quake ロガーユーティリティの章の Desired Gateway の確認/編集の項を参照して下さい。）自動ローミングの詳細に就いては Quake アプリケーションプ

ログラミング章の自動ローミングの項を参照して下さい。

Report Auto-Convert：
このパラメータに就いては Enabled（機能オン）或いは Disabled（機能オフ）を選択します。この機能がオンになっていると、6 バイト以下のデータを持つ Message は自動的に Report 形式に変換されて送信されます。この機能は、内含するデータ長が変化する Message に対して有効で、そのようなメッセージが常に最小のコストで送信されるように働きます。

gwy_id：
これはメッセージを受信するゲートウェイの ID です。このゲートウェイと繋がっていない衛星経由では、GlobalGram を除き、メッセージは送信できません。Quake 端末は出荷時に米国の gwy_id に設定されています。

polled：
これはオーブコムのメッセージパラメータの一つで、メッセージを直ちに (0) 送信するか、衛星からポーリングを受けた時に（1）送信するかを定めています。Quake 端末は出荷時にこの機能を（0）に設定されています。

ack_level：
これはオーブコムシステムからの Acknowledgement（受領確認）レベルです。オーブコムの定義では、0 = 受領確認無し；1 = ゲートウェイに不達の時のみ；2 = ゲートウェイへの送達及び不達を確認；3 = 宛先人へ不達の時のみ；4 = 宛先人への送達及び不達を確認、となっています。Quake 端末は出荷時にこの機能を（2）に設定されており、それ以外の設定を使用することはお勧め出来ません。これ以外の設定をした場合、MSG_ACK イベントがアプリケーション上では発生しなかったり、その結果メッセージを消失する可能性があります。

priority：
これはメッセージの優先度（priority）です。オーブコムの定義では、0 = Non-urgent ; 1 = Normal ; 2 = Urgent ; 3 = Special delivery (highest priority)となっています。Quake 端末は出荷時に 1（Normal）に設定されています。

msg_body_type：
これはメッセージボディの形式です。オーブコムは 16 のボディ形式を定めていますが、その多くは未だ実用化されていません。その多くは X.400 網で使用するもので、実際に使用できるものは ASCII テキスト（msg_body_type = 0）とバイナリー（同 14）です。。Quake

端末は出荷時にデフォルトとして 14 を設定しています。バイナリ-データの詳細は Appendix E を参照して下さい。

serv_type：
これは機能化の為に ack_level と priority を複合したものです。このパラメータは Report に使用します。オーブコムは 16 の有り得る ack/priority 組み合わせを定義しています。Quake 端末は出荷時にこの機能を（2）に設定されており、それ以外の設定を使用することはお勧め出来ません。これ以外の設定をした場合、MSG_ACK イベントがアプリケーション上では発生しなかったり、その結果メッセージを消失する可能性があります。

Recipient Qty：
これはメッセージの宛先人の数です。受領者の指定方法は 2 種類、電子メールアドレスと O/R インディケータ(短縮ダイアル)、が有ります。オーブコムは端末各々の登録データとして、1 から 7 までの O/R インディケータに電子メールアドレスを登録する事が出来ます。これらの O/R インディケータを使うことで Message のバイト数を減らす事が出来ます。送り先のメールアドレス全部を送る代わりに 1 バイトの O/R インディケータを送ることでメッセージを遥かにコンパクトに出来ます。もしも送り先メールアドレスが O/R インディケータとして登録されていない時は、メッセージ中に送り先のメールアドレスを記述する必要があります。

Report 形式の場合、メールアドレスを 6 バイトに納めることは出来ませんので、1 つの O/R アドレスのみへの送信となります。GlobalGram も同様に 1 つの O/R インディケータのみが許されます。

Subject：
メールの件名です。必須事項ではありません。

Parameters：
これは端末によって収集されるデータです。メッセージテーブルのエディター画面では以下のような表示でメッセージに含むデータやパラメータを聞いてきます。

```
SELECT PARAMETER 1 TYPE:
'0'  ANALOG_IN          '1'  DIGITAL_IN         '2'  LATITUDE
'3'  LONGITUDE          '4'  ALTITUDE           '5'  CURRENT_TIME
'6'  ELAPSED_TIME       '7'  COUNTER_VALUE      '8'  USER_DATA_BYTE
'9'  SERIAL_DATA        '10' SPEED              '11' HEADING
```

```
'12' LAST_USER_CMD_DATA  '13' LAST_OB_MSG_SUBJ    '14' J1708_DATA
'15' FILE_DATA
```

メッセージ中に含ませるデータを選び、パラメータのビット数を記入して下さい。パラメータの詳細及びパラメータデータを希望するビットサイズに縮小する方法に就いてはAppendix C に述べられています。

6. センサーテーブル

センサーテーブルは特定のセンサーからどの様にデータを読むかを規定するものです。ユーティリティモードから以下の様に各センサーテーブルの編集モードに入れます。（Q1200SM/SG 及び Q1400 では Analog Input Sensor Table Utilities 及び J1780 Sensor Table Utilities は使用できません）

- 'a' to enter Analog Input Sensor Table Utilities
- 'd' to enter Digital I/O Sensor Table Utilities
- 'g' to enter GPS Sensor Table Utilities
- 's' to enter Serial Sensor Table Utilities
- 'j'　to enter J1708 Sensor Table Utilities

A) アナログセンサーテーブル（Q1200SG/SM 及び Q1400 では使用できません）

Quake ロガーは 0 から 15 までのテーブルを持つことしか出来ません。しかし、どれだけ複雑なアプリケーションでもこれで充分なはずです。アプリケーションイベントテーブルがレスポンスアクションとして、例えば、READ_ANALOG5 と定めている場合、これはアナログセンサーテーブル 5 に定められたアナログチャネルの値を同テーブル 5 に定められた方法で読み取る事を意味します。READ_ANALOG5 はアナログチャネル 5 を意味しませんので注意が必要です。

アナログセンサーテーブルのプログラミング例を示します。アナログチャネル 7 を一日一回読むことにします。読値方法としては、1 秒間隔で 10 回の読み値を取って平均し、警報閾値の最大と最小と比べるようにします。

これをアナログセンサーテーブル 0 に書き込む事にします。アナログセンサーテーブルを編集する為、先ず"U"そして"a"と入力します。次の画面が現れます。

```
ADC: If you don't want to change a field, input an 'x'.
ADC utils
```

```
Enter m(measurements) c(conversions) l(log)
```

“m”を入力して下さい。画面には次の様に表示されます。

```
adc utils, enter:
v(view) d(defaults) m(modify) s(save) q(quit)
```

“m”を入力して下さい。画面には次の様に表示されます。

```
Enter the number of the measurement to modify<CR>
```

ここで”0”を入力してアナログセンサーテーブル 0 の準備を始めます。画面の質問や入力要求に回答を入力して下さい。テーブルが完成すると次の様な概要が表示されます。

```
Channel 7 will be read 10 times, at 1 sec intervals
The measurements will have 10 bit resolution
Alarms will be reported on completion of samples
Low alarm triggered if mean is less than 10000.
High alarm triggered if mean greater than 50000.
Satisfied?(y/n)
```

幾つかの点に就いてここで説明します。端末の A/D コンバータ解像度は 10 ビット或いは 12 ビットです。どのような場合でも外部ノイズによってこの変換精度は 8 ビット以下になってしまうため、この選択肢にはあまり意味がありません。端末内では､各サンプルは 16 ビットの整数に読み替えられます。つまり、サンプルの最小値はゼロと記録され、想定されるサンプルの最大値は 65535 となります。この値を必要に応じて使用されるアナログ機器の単位に変換することが出来ます。変換はリニアに変換します。しかし、出来るだけ生の値を使用することを推奨します。コンバージョンが使用される場合でもメッセージ中のデータは常に生のアナログデータです。

このアプリケーション例を完成するにはアナログチャネル 7 を一日に一回読むということが必要です。アナログセンサーテーブルは読値の間隔は定義しません。これはアプリケーションイベントテーブルで行います。一日一回”READO_ANALOG0”というレスポンスアクションを行うようにアプリケーションイベントテーブルを書きます。（タイマーに就いては「アプリケーションソフトウェアタイマー」の項を参照してください。）

アナログセンサーテーブルユーティリティで”v”(view)を選択すると16のアナログセンサーテーブル全てが画面にスクロールします。これら全てのテーブルはデフォルトデータですが、アプリケーションイベントテーブルでセンサーを読むようにプログラムされない限りいずれのテーブルも使われません。

アナログセンサーテーブルユーティリティで”l”(log)を選択すると、各アナログデータ値が測定された直後に画面に表示されます。この機能はデバッグやアナログ入力のセットアップなどで使用します。

B) デジタルセンサーテーブル

Quake ロガーは 0 から 15 までのテーブルを持つことしか出来ません。しかし、どれだけ複雑なアプリケーションでもこれで充分なはずです。アプリケーションイベントテーブルがレスポンスアクションとして、例えば、READ_DIGITAL3 と定めている場合、これはデジタルセンサーテーブル 3 に定められたデジタルチャネルの値を同テーブル 3 に定められた方法で読み取る事を意味します。READ_DIGITAL3 はデジタルチャネル 3 を意味しませんので注意が必要です。

デジタルセンサーテーブルのプログラミング例を示します。デジタルチャネル 3 がドアスイッチに接続され、ドアが開いている時に「高」、閉じている時に「低」となるとします。ドアが開いた時には端末内で警報が出されるようにします。

これをデジタルセンサーテーブル 1 に書き込む事にします。デジタルセンサーテーブルを編集する為、先ず”U”そして”d”と入力します。次の画面が現れます。

```
DIO: Enter m(measurements) i(intialization)
```

“m”を入力して下さい。画面には次の様に表示されます。

```
DIO: If you don't want to change a field, input an 'x'.
dio utils, enter:
v(view) d(defaults) m(modify) s(save) q(quit)
```

ここで”m”を入力して”1”とタイプしデジタルセンサーテーブル 1 の編集を始めます。画面の質問や入力要求に回答を入力して下さい。テーブルが完成すると次の様な概要が表示されます。

```
Channel  3 will be read 3600 times, at 1 sec intervals
DIO alarms will be reported immediately
Low alarm will NEVER trigger.
High alarm triggered if any value greater than 0x10.
Elapsed times will NOT be recorded or alarmed.
Satisfied?(y/n)y
```

この例では、デジタルチャネル 3 を 1 秒間隔で読むようにテーブルがプログラムされており、もしもドアが開いていれば 255 という値に読み替えられるデジタルの「高」の読み値が得られます。もしもドアが閉じていればデジタルの「低」の読み値が得られます。このテーブルではチャネルの読みを 3600 回繰り返すように掛かれていますが、この数値は 10 でも 1000 でも構いません。重要な事はチャネルを永久に読み続け、ドアが開けられたらアラームを鳴らす様にすることです。アプリケーションイベントテーブルにその様に書き込む必要があります。POWER_ON イベントと DIGITAL1 イベントに対して READ_DIGITAL1 というレスポンスアクションを定義させる事でチャネルを読み続けます。

「高」の読み値が 0x10（十進数の 16）以上の場合にアラームが起動するようになっていることに注意して下さい。この値もいい加減で結構です。ドアが閉まっている状態でのデジタル読み値は 0 です。ドアが開けられれば読み値は 255 になります。アラーム起動のための数値は 1 でも良いのです。これらの読み値は 16 進数で入力して下さい。

デジタルセンサーテーブルではセンサー上の経過時間の積算が可能です。これはこのセンサーが「高」或いは「低」であった累積時間です。もしも経過時間値が必要であれば下記の様な問い掛けに”y”（イエス）と応えて下さい。

```
Record Elapsed Times on this Channel?
```

そうすると画面は次の様な質問を投げてきます。

```
Enter sensor active high or low?(h/l)
Elapsed time alarm interval 100.00 hours:
Elapsed time write interval  5.00 hours:
```

この alarm interval（アラーム間隔）を超過するとアプリケーションは DIGITAL_ALARM イベントを発生させます。さらに、定められた間隔ごとに経過時間を不揮発メモリーに書

き込みます。経過時間は RAM に記録されており、通常の電源オフ時及び定められた間隔毎に不揮発メモリーに書き込まれています。この機能によって、不測の電源オフの際にある程度のデータ保護が期待出来ます。

デジタルセンサーテーブルユーティリティで"v"(view)を選択すると 16 のデジタルセンサーテーブル全てが画面にスクロールします。これら全てのテーブルはデフォルトデータですが、アプリケーションイベントテーブルでセンサーを読むようにプログラムされない限りいずれのテーブルも使われません。

デジタルセンサーテーブルユーティリティで"i"(initialization)を選択すると、デジタル I/O の設定ユーティリティにアクセスする事が出来ます。しかし、このユーティリティはデジタルセンサーを入力で使用するか出力で使用するかの設定だけに使用することを推奨します。出力に設定されているデジタル I/O チャネルを入力で使用すると障害を起こす惧れがあります。Quake 端末は出荷時に全てのデジタルセンサーを入力として設定してあります。

C) GPS センサーテーブル

Quake ロガーは 16 種類の GPS センサーテーブルを作成、編集する事が出来ます。各テーブルは 100 個所までの地点（緯度/経度）を記憶する事が出来、この地点の周囲の円形境界に対して端末がその内側或いは外側に居る事でアラームを鳴らすことが出来ます。

GPS センサーテーブルのプログラミング例を示します。端末が特定の 4 地域のいずれかに入った場合に、その旨のメッセージを発するようにします。

この地域への進入を GPS センサーテーブル 0 で、同じく地域からの退出を GPS センサーテーブル 1 で管理することにします。警告範囲の半径を、進入の際には 500 メートル、退出の際には 1000 メートルとします。端末が境界近傍に止まる事を考慮して、進入と退出の警告半径を異なる数値としてそれぞれの境界線を充分に離すようにします。さもないと GPS の誤差によって同地域に端末が出たり入ったりしているかの如くメッセージが多発されてしまいます。

GPS センサーテーブルを編集する為、"U"とタイプしてユーティリティモードに入り、"g"とタイプして編集モードに入ります。すると以下のような画面が現れます。

```
gps utils, enter:
  d(dummy position)  f(fix) t(BIT test) p(toggle power mode) I(init)
  m(measurement) D(debug) a(toggle all messages) c(cold start) L(Locations)
```

```
 l(log gps) s(log some) r(DEASSERT UART0 reset) R(ASSERT UART0 reset)
 i(ivals) c(cold start now) w(warm start now) C(cold start sequence)
q(quit)
  x(dgn_txfr_apl_table(GPS))
```

多くの選択肢が出てきますが、重要なのは主として”m”（measurement）、”L”（location）そして”l”（log gps）でしょう。

“m”を入力して GPS センサーテーブルのエディターを起動し、”0”を入力して GPS センサーテーブル 0 の作成を開始します。画面に示される質問に答えていって下さい。テーブルのプログラミングが終了すると以下の様な表示がされます。

```
Alarm is set to trigger only on completion.
Position will be taken 10 times, at 1 sec intervals
Alarms will be reported on completion of samples.
Low alarm triggered if:
  distance from mean measured position and any location in set 1
  is less than 500 meters.
High alarm will NEVER trigger.
The reference position is
Satisfied?(y/n)
```

他のセンサーと同様に、アラームを平均値に対して設定するか或いは一つでも条件に合致した際にと設定する事も出来ます。今回は誤りの可能性を低くする為に 1 秒間隔の 10 回計測値の平均を選択しました。このテーブルは、Location Set1 の情報を入力すれば完成です。これは他のエディターで行いますが、その点は後述します。また、参照地点という機能がありますが、これは使用していません。さて、”m”（modify）を選び”1”を入力して GPS センサーテーブル 1 の準備に掛かります。画面に示される質問に答えていって下さい。テーブルのプログラミングが終了すると以下の様な表示がされます。

```
Alarm is set to trigger only on completion.
Position will be taken 10 times, at 1 sec intervals
Alarms will be reported on completion of samples.
Low alarm will NEVER trigger.
High alarm triggered if:
  distance from mean measured position and all locations in set 1
```

```
  is greater than 1000 meters.
The reference position is
Satisfied?(y/n)
```

これでテーブル 0 とテーブル 1 が完成しました。後は Location Set1 を追加するだけです。この為、GPS センサーテーブルエディターを抜けてユーティリティモードから”g”を入力した状態へ戻ります。多くの選択肢から”L”（location）を選びます。画面は次の様に表示されます。

```
gps locations, enter:
 l(list location sets)  m(modify) v(view locations in a set)
 n(new) r(remove location from set) d(delete set) s(scan) q(quit)
```

“n”を選んで新しい地点の入力をします。画面の指示に従って新しい地点を入力して下さい。各々の地点の入力が終わったら Location Set1 にそれを追加して下さい。同様にして 4 つの地点を入力し終わったら、”v”そして”1”とタイプして下さい。Location Set1 が画面に次の様に表示されます。

```
Location   0:       Joe lat 41.549843000, lon -117.239873000
Location   1:      Bill lat 42.609430000, lon -117.670930000
Location   2:       Tom lat 42.109870000, lon -116.098340000
Location   3:      Dick lat 39.098300000, lon -114.000320000
```

これでこのアプリケーションの GPS センサーテーブルと地点リストが完成しました。次にアプリケーションイベントテーブルで、端末が指定区域内にあるときには GET_POSITION0 を、指定区域外にあるときには GET_POSITION1 をレスポンスアクションとして採るようにプログラムします。例えばこのアプリケーションの場合は次の様になります。

```
==========  Application Event Table  ===========
EVENT              RESPONSE ACTION
POSITION_FIX 0      GET_POSITION 0
POSITION_ALARM 0    SEND_MSG 0 [0]
-                 GET_POSITION 1
POSITION_FIX 1      GET_POSITION 1
POSITION_ALARM 1    SEND_MSG 1 [0]
```

```
-                  GET_POSITION 0
POWER_ON            GET_POSITION 0
-                  START_DL_ACQ 0
================================================
```

今回の場合、GET_POSITION 動作によって返されるものは POSITION_FIX か POSITION_ALARM のいずれかで、両方が返る事はありません。POSITION_FIX イベントが返されるということはアラームは発生していません。アラームが発生していないので、アプリケーションは POSIITON_FIX イベントに対応して GET_POSITION 動作を繰り返します。この様にして端末の位置を絶えず更新しながらアラーム状態になっているかを確認し続けます。アラーム状態になった場合、端末が指定範囲内に入ったときには Message0 を、出て行った場合には Message1 という異なるメッセージを送ります。

D) シリアルセンサーテーブル

Quake 基本コード及び基本アプリケーションにシリアルセンサーの監視機能が追加され、非同期シリアルインターフェース（RS-232）を通じて種々の機器を監視出来るようになりました。シリアルポートにはモデムの他に、計量や温度センサー外部 CPU などが接続されるでしょう。これらの外部機器は色々なメーカーによって作られている為、多くの種類の通信プロトコルが用いられます。Quake 端末では可能な限り色々なプロトコルをサポートできるように考えられており、次の様な端末とシリアルデバイス間に起こる種々のやり取りをサポートしています。

1. 端末から任意に設定された問合せコマンドを送ることによってデータを要求する。
2. 或いは、端末からの要求無しに、シリアル機器側からデータを送り込む。
3. データの受信に際しては任意に設定した「送信開始」信号でデータの到着を知らせる；この信号以前のものは無視され、有効なデータは「送信開始」信号以降に受信される。
4. 有効データにはデータ開始信号を付けない；先頭から有効データとなる。
5. 「送信開始」信号到着からの一定時間経過或いは有効データの最初の信号到着が必要。
6. 受信するデータ長は固定或いは可変。
7. 受信データは、バイナリー、キャラクター或いはキャラクターで表示される数値。
8. データ列の終わりは何らかの形で表示されなければならない。これは任意の「送信終了」信号を使用する方法、データ長で規定する方法、或いは受信時間で制限する方法などがある。
9. ボーレートなどの非同期シリアルポートパラメータは外部機器との間で整合してなければならない。

種々のシリアルセンサーをサポートするには幾つか複雑化する点があることに留意すれば、

他の測定と基本的に同じ様にシリアル測定をする事が出来ます。0 番から 15 番までの合計 16 のシリアルセンサーテーブルに色々なシリアル測定方法を作る事が出来ます。イベントテーブルで READ_SERIAL 動作が呼び出されると、そこに指定されている番号のシリアルセンサーテーブルに記述された測定の方法が実行されます。シリアル測定に就いて幾つかの例を以下に紹介します。

次の様な特性をもったシリアルセンサーを想定しましょう。

1. このセンサーはトレーラーに積まれている荷物の色々な情報、例えば温度や重量など、を測ることができます。
2. このセンサーはオーブコム端末の MTS ポートに 19200baud で接続されています。
3. センサー側ではどのデータが欲しいかという具体的な要求を受けてそれを出すものとします。今回は温度データを貰う為に"SEND_TEMP"という命令を送ってセンサーに温度を読ませる、という事にします。
4. センサーが温度計測の要求を受けてから測定に 10 秒ほどかかり、その後に回答を返します。
5. センサーは測定値を ASCII の浮動小数点で表示し、データの前に">>"、後ろに"<<"を表示します。回答を返す時間は送信開始から 5 秒以内に終わります。

端末とセンサー間のやり取りの代表例は次の様になります。

端末がセンサーに送信　　SEND_TEMP

センサーが温度を測定するのに 10 秒ほど掛かります。

センサーが端末に送信　　>>2.1<<

送信完了までの時間は 5 秒以内。

ここでは外部機器からの温度測定値一つだけを採ると仮定します。

このシリアルセンサーテーブルは、ロガーポートユーティリティ或いは QUAKETools で作成します。どちらの場合も、シリアル測定設定パラメータは下記の様に設定します。

samples = 1　　測定結果には一つの読み値が有ります。

interval = 1　　読み値が一つしかないので実質無関係です。

port_nem = 2　　1 がロガーポートの意味で、2 は MTS ポートを指します。

baud=19200	必要なボーレートを設定します。
alarm_type= ALARM_IMMEDIATELY	アラームの状態が生じたら測定を中止して即刻報告する。
timeout=5	センサーからのデータは 5 秒以内に送信を完了する。完了しない場合は何か間違いがある。
termination_count = 80	センサーからのデータは ASCII による浮動小数点で表される為、データ長が変化する可能性がある（例えば-23.716 とか 1.0 とか）。しかし、最大でも 80 文字を超える事はないと考えられる。
term_len = 2	「送信終了」信号は 2 文字（“<<”）です。
term_bytes = “<<”	データ列の後に送られる「送信終了」信号です。
send_len = 9	データ要求命令文は 9 文字（“SEND TEMP”）です。
send_bytes = SEND TEMP	センサーに対して温度読み値を要求する命令文です。
hi_alarm_type = ASCII_FLOAT	入って来るデータは ASCII で表現され浮動小数点数値に変換され、アラームは変換されたデータに対して発せられる。.
hi_len = 0	低位点を表すデータのバイト数。この変数はテキストにのみ適用されるので、この場合は使われない。
hi_alm_bytes = 10.0	温度がこの値以上になったらアラームを発する。
low_len = 0	高位点を表すデータのバイト数。上記の hi_len.を参照。
low_alm_bytes = 0.0	温度がこの値以下になったらアラームを発する。
Start_len = 2	「送信開始」信号は 2 文字（“>>”）です。
Start_bytes = “>>”	「送信開始」信号です。文字列の最後は null 以外でなければな

りません。

Start_timeout = 10　　“SEND TEMP” を送った後 10 秒間の内に“>>” を受信しない場合、エラーと判定して測定を中止する。

B_mode　　この機能は使いません。B_mode に就いては後述します。

Exact terminator　　この機能は使いません。詳細は後述します。

上記の設定をする為に、ロガーユーティリティを使用します。”U”に続いて”s”をタイプします。画面には次の様に表示されます。

```
Serial Sensor Utils, enter:
  l(log serial) v(view) d(defaults) g(get serial) e(echo)
  r(retrieve) m(modify) s(save) w(wipe file) q(quit)
```

シリアルセンサーテーブル 0 を編集する為、”m”に続いて”0”を入力します。現在の設定一覧が出てきます。画面の指示やメニューに従って各設定を上記の値にセットして下さい。最初の設定画面では受信するデータの形式をセットしますので、次の様に基本データ形式を ASCII_NUMERIC の浮動小数点に変更して下さい。

```
Modify the fundamental data type:
  (binary/char/ascii_numeric, fixed/float)?(y/n)y

Please enter the type of data:
  b(binary) c(character) a(ascii numeric)<CR>a

Please enter ascii numeric type: i(fixed) f(floating)<CR>f
High alarm is of type ASCII_FLOAT.
Low alarm is of type ASCII_FLOAT.
Satisfied?(y/n)y
```

残りの設定をセットし終わった時点で、view オプション（”v”）でセンサーテーブルの設定を表示させます。画面上には次の様になります。

```
Measurement 0:
```

```
 Measurement is set to normal mode
 Samples 1, interval 1 ALARM IMMEDIATLY
 port 2,  baud 19200
 Termination sequence may occur after variable number of data bytes.
 Methods used to terminate input:
 timeout 5 sec.    byte count 80
 Incoming data are of CHARACTER type. input may also terminate on:
 A termination sequence  exists, of length 2, as follows:
0x  3c 3c
   <  <
    An initiation sequence  exists, of length 2, as follows:
0x  3e 3e
   >  >
    and has an associated timeout of 10 seconds.
 The incoming characters will be interpreted as an ASCII float number.

Send bytes:
0x  53 45 4e 44 20 54 45 4d 50
   S  E  N  D    T  E  M  P
  High alarm value = float 10   Low Alarm value = float 0
```

この場合、タイムアウトが二種類あることに注意して下さい。一つはセンサーからデータが入り始めてからデータ取得完了までのタイムアウトで小文字の”o”を入力して選択します。大文字の”O”を入力すると、センサーからのデータ受信が始まるまでの時間のタイムアウト操作になります。また、他の測定方法同様に、シリアルユーティリティメニューから”l”キーを使ってシリアル測定のログを採る事が出来ます。ログ機能は測定方法の正しい設定や各機能の働き方を理解する事に役立ちます。

多分シリアル測定の働きをより良く理解する最も効率的な方法は、ハイパーターミナルの様なエミュレータを端末の MTS ポートに接続して、シリアルセンサーと端末間のやり取りを手動でシミュレートする事でしょう。以下は上記の測定法を用いた場合のセッションの例です。

端末ロガーからの出力

```
***Utility Mode enabled*** (will expire in 20 Secs)
r
```

```
SELECT RESPONSE ACTION:
'0'  CFG_DIGITAL        '1'  READ_ANALOG        '2'  READ_DIGITAL
'3'  SET_DIGITAL        '4'  SET_TIMER          '5'  GET_POSITION
'6'  POWER_DOWN         '7'  MSG_ENQ           '8'  SEND_MSG
'9'  START_DL_ACQ       '10' SET_COUNTER        '11' INCR_COUNTER
'12' DECR_COUNTER       '13' READ_SERIAL        '14' COMMS_CMD
'15' SET_GEOFENCE       '16' DEL_GEOFENCE       '17' READ_J1708
> 13
ENTER RESPONSE PRM1> 0
Calling APL Response Action Processing Routine...
Exiting Utility mode
use MTS semaphores
Send_len = 9.
In ser_has_started; wait for data...exit ser_has_started!
Update ser stats
Received f_val = 3.100000
Last value  3.1; Mean  3.1; Max  3.1; Min  3.1; Set #0 struct to AVAIL
APL: Processing SER_MEAS 0 Event
```

ハイパーターミナル上の MTS I/O（ローカルエコー有り）

```
SEND TEMP>>3.1<<
```

(ア) シリアル測定モード

シリアルデータの送受信を制御する方法に就いて説明します。送信終了に就いては、普通、終了条件に就いて論理式の OR を使います。通常、「必要バイト数のデータを受領した」OR、「(”o”で設定される方の）タイムアウト秒数を経過してデータを受信している」OR、「(設定されているのであれば)「送信終了」信号を受信した」場合に測定を終了します。これによって可変長のデータを扱う事が可能になります。例えば、<CR>を「送信終了」信号に設定したとします。何バイトでも入力する事が出来、いつでも<CR>を入力して終了する事が出来ます。「送信終了」信号を使うモードでは、「送信終了」信号のバイト数及びその信号を具体的に規定しておく必要があります。これらと異なる信号はエラーとなり、SERIAL_EVENT というイベントは発生しないことになります。このモードはシリアルユーティリティメニューの編集サブメニューで”E”を入力することで切り替わります。

通常の測定では端末内のアプリケーションが、問合せ文字列をセンサーに送るなどして、測定開始の任を負っています。このような場合、複数の測定作業を短時間に行おうとする

ことは、CPU の負荷を増加し、お勧め出来ません。この為、測定作業は 1 秒に 1 件以下とされています。しかしながら、ある種のシリアルセンサーは随時測定値を送って来ることが考えられます。b_mode の主目的はその様な前触れのないデータ送信を可能にすることです。b_mode が選択されるとインターバルは 0 に設定されます。一つの測定が終了すると直ちに次の測定に対応出来るようになります。さらに、b_mode では「送信開始」信号ではなくいきなりデータ信号で開始することが出来ます。b_mode では「送信開始」信号が定められていなくとも、最初のタイムアウト（”O”で設定する開始までの方）によってデータ受信開始までの時間を表します。データを受信し始めると二つ目のタイムアウト（”o”で設定する方）がデータ受信完了までの時間を表します。通常の、b_mode ではないオペレーションでは、端末は問合せ文字列を送って測定を開始し、データはほとんど瞬時に受領されます。従って、「送信開始」信号がない状態では開始時間タイムアウトは使用されません。

(イ) シリアル測定アラーム

シリアルの測定では数値変換されるデータを与えられることがあります。これは測定の設定でバイナリーデータを採るか ASCII に変換された数値データを採るかで発生する問題です。いずれの場合でも、数値シリアルデータは ADC データの様な他の全ての数値データと同様に扱われます。測定が行われるごとに累積平均が計算され、最大/最小値が記録され、そして最新値は高位/低位のアラーム閾値に参照されます（ALARM_TYPE が ON_COMPLETION となっている場合は測定が終了した後)。シリアル測定では数値変換を伴なわないデータもありえます。”Error”、”Over Temp”或いは外部からのディレクトリ一覧などがそれらです。Quake の基本コードではこの種のシリアル/テキストデータに対するアラームが可能です。これは高位或いは低位アラームを、受信するテキストと高位或いは低位のアラーム閾値として設定するテキストと比較して行います。アラームは、EXACT_MATCH（完全一致)、或いは受信するテキスト内のどこかに設定された閾値テキストが含まれている場合に発生するような設定が出来ます。閾値テキストの設定は問合せ文字列の設定と同じ様に行います。EXACT_MATCH 或いは CONTAINED_WITHIN の選択は、基本データ形式を変更した後に行います。文字データに対するシリアル測定アラーム設定の例を示します。

ロガーユーティリティで見た測定の設定

```
Measurement 1:
Measurement is set to normal mode
Samples 1, interval 1 ALARM IMMEDIATLY
port 2,  baud 19200
Termination sequence may occur after variable number of data bytes.
Methods used to terminate input:
```

```
 timeout 20 sec.   byte count 80
 Incoming data are of CHARACTER type. input may also terminate on:
 A termination sequence has NOT been specified.
 An initiation sequence  has NOT been specified.
 there is a timeout of 65535 seconds for start of data.

Send bytes:
0x  2a
   *

High alarm triggers if read bytes are CONTAINED_WITHIN alarm bytes
High alarm bytes:
0x  77 6f 72 6c 64
   w  o  r  l  d

Low alarm triggers if read bytes are EXACT_MATCH alarm bytes
Low alarm bytes:
0x  45 52 52 4f 52
   E  R  R  O  R
```

シリアルセンサーテーブル１の開始

```
***Utility Mode enabled*** (will expire in 20 Secs)
r
SELECT RESPONSE ACTION:
'0'  CFG_DIGITAL        '1'  READ_ANALOG        '2'  READ_DIGITAL
'3'  SET_DIGITAL        '4'  SET_TIMER          '5'  GET_POSITION
'6'  POWER_DOWN         '7'  MSG_ENQ           '8'  SEND_MSG
'9'  START_DL_ACQ       '10' SET_COUNTER        '11' INCR_COUNTER
'12' DECR_COUNTER       '13' READ_SERIAL        '14' COMMS_CMD
'15' SET_GEOFENCE       '16' DEL_GEOFENCE       '17' READ_J1708
> 13
ENTER RESPONSE PRM1> 1
```

MTS ポートに接続したハイパーターミナル画面（ローカルエコーあり）

```
*Hello, world
```

端末ロガーポートからの回答（シリアルログをオン）

```
use MTS semaphores
Send_len = 1.
Update ser stats
Last value >>>High serial alarm.  terminate, status = 0<<<
Set #1 struct to AVAIL
APL: Processing SER_ALARM 1 Event
```

シリアルセンサーテーブル１が完了した状態でのロガーポート画面

```
Receive_bytes:
0x  48 65 6c 6c 6f 2c 20 77 6f 72 6c 64 0d
   H  e  l  l  o  ,     w  o  r  l  d  ^M


Receive_len = 13
```

注:　入力された文字列は<CR>を含めて全てデータとして受信され、高位のアラーム文字列である”world”が入力文字列に含まれていた事から、このイベントはアラーム状態で終了しています。この場合、データ入力は 20 秒間のタイムアウトで終了しました。

(ウ) シリアルデータ保存

各種（浮動小数点形式、整数値、等）のシリアル数値データとテキストシリアルデータは端末のプロセッサーによって端末内の異なる場所に保存されます。しかし、全ての形式のシリアルデータを含むメッセージ作成を簡単な統一方式とする為、シリアルデータはシリアル構造の中の receive_bytes 列に保存されます。数値データはリトルエンディアン形式で、ASCII_NUMERIC は変換されて IEEE４バイト浮動小数点数値として保存されます。テキストデータは最初に受信されたバイトから低位メモリーに保存されます。シリアルデータを含むメッセージを解読する際にはこのデータ保存方式に注意して下さい。

E)　J1708 センサーテーブル

（Q2000 用：略）

7. Quake ロガーユーティリティ

アプリケーションやメッセージ、センサーテーブルの作成及び表示、編集の他、Quake ロガーにはデバッグアプリケーション、電子メール送受信及び運用性能の監視の機能があります。本章ではこれらの機能を簡単に説明します。各機能を使うには先ずユーティリティ

モードに入り、下記のいずれかのキーを押します。

```
AVAILABLE COMMANDS:
'?' Lists menu options
'A' Application Event Table Utilities
'M' Message Table Utilities
'G' View/Edit Desired Gateways List
'a' AIN Sensor Table Utilities
'0' Area0 Utilities
'd' DIO Sensor Table Utilities
'e' Elapsed Times (ets) Utilities
'g' GPS Sensor Table Utilities
'j' J1708 Sensor Table Utilities
's' Serial Sensor Table Utilities
'l' last (et) alarm Utilities
'F' File Transfer Utilities
'S' Send Email
'R' Read Email
'C' Modify Config Parameters
'H' Set Service Meter Hours
'P' GPS test
```

A) View/Edit Desired Gateways List

（接続ゲートウェイリストの表示と編集）

このユーティリティは接続ゲートウェイのリストを表示/編集します。このリストはこの端末が通信を許される相手ゲートウェイを定めます。もしもご使用になる端末が一つのゲートウェイのサービス範囲でしか使われない、或いは単一のゲートウェイにしか登録されない場合はこのユーティリティは不要です。しかし複数のゲートウェイに登録される場合、それらのゲートウェイをこのリストに入力しておく必要があります。これにより、御使用になる端末はこのリスト上のゲートウェイとしか通信をしなくなります。もしも接続ゲートウェイリストが入力されていないと、端末はあらゆるゲートウェイと通信する事が可能ですが、端末が登録されていないゲートウェイと通信しようとすることで、結果として通信の効率が低くなってしまいます。

接続ゲートウェイリストの入力はシンプルです。このユーティリティに入ると現在のリストが表示されます。（注：接続ゲートウェイリストには使おうとするゲートウェイを入力す

るか或いは使わないゲートウェイを入力するかのいずれかが可能です。どのゲートウェイに登録されているか不確かだが特定のゲートウェイを使用することは避けたい、という際に後者が便利です。）必要に応じて現状のリストから削除するなり追加して下さい。もしもリストはない、或いはリストそのものを削除した場合、端末は全てのゲートウェイを接続対象と見なします。新しいリストを作成する場合は”n”を押して下さい。画面は「利用するゲートウェイ」を入力するのか「利用しないゲートウェイ」を入力するのかを聞いてきます。この問いに答えた後、入力すべきゲートウェイ ID をコンマかスペースで区切って入力して下さい。終わりましたら<Enter>を押して下さい。リストは自動的に保存されます。

B) Execute Application Response Action

（アプリケーション上のレスポンスアクションの実行）

このユーティリティはターミナル側からレスポンスアクションを実行させるものです。デバッグや端末の運用性能を見ようとする時に便利です。ユーティリティモードから”y”とタイプして入ります。画面にはレスポンスアクションのテーブルが示されその中から一つを選択します。動作が選択されると、あたかも何らかのイベントが発生しそれによってその動作がアプリケーションイベントテーブルで指定されているかのように指定動作が実行されます。

C) Elapsed Times Utilities

（経過時間ユーティリティ）

このユーティリティは経過時間をリアルタイムで読んだり、初期値に戻したりする事が出来ます。経過時間の読み値は、エンジンや機械の様なあるものがオン或いはオフ状態にどの位の時間あったかを知りたいというアプリケーションで利用されます。このユーティリティには”U”をタイプしてユーティリティモードに入り更に”e”とタイプします。画面には次の様に表示されます。

```
ets_util, enter:
v(view) m(modify) s(save) l(log) d(DIOinit) u(user cmd) q(quit)
```

“d(DIOinit)”及び”u(user cmd)”は使用しないで下さい。”l(log)”は現在進んでいる経過時間の読み値を表示させます。”m(modify)”は 16 の経過時間テーブルのどれでも編集する事を可能にします。”v(view)”は 16 の経過時間テーブル全てを表示します。経過時間テーブルの編集をする場合、このユーティリティを終える前に”s(save)”する事を忘れないように注意して下さい。

D) Last Alarm Utilities

このユーティリティは 16 全ての経過時間の最終時刻を表示、編集します。

E) File Transfer Utilities

このユーティリティはファイルを検索して端末のファイルシステムにダウンロードするものです。この機能は複雑な運用の為にあり、通常利用を想定していません。この機能を利用されたい場合は Quake にお問合せ下さい。

F) Send Email Utility

これは電子メールを作ってオーブコムシステムを使って送信する簡単なユーティリティです。ユーティリティモードから”S”とタイプしてこのモードに入ります。画面の問い掛けに従ってメールを作成します。オーブコム端末の受信機がオンとなっていることとアンテナが広く天空を見渡せる事を確認して下さい。Quake ロガー画面で Acq とか Rx とかが付けられたデータが連続してスクロールしていれば受信機はオンになっています。もしもそうなっていなければ、”d”とタイプして”w”と続けるとオーブコム受信機をスタートさせる事が出来ます。

G) Read Email Utility

これは電子メールを読む簡単なユーティリティです。電子メールが送られてきてそれを読みたい時にはユーティリティモードに入って”R”とタイプします。これでメールを選んで表示する事が出来ます。基本ソフトウェアの Read Email 機能では現状 10 通までの保存が可能です。アウトバウンドメッセージのみが保存されます（アウトバウンドとはインターネットで作成されたメッセージが端末へ流れるトラヒックを指します)。インバウンドメッセージは保存されません（イバウンドとは端末から送出されるトラヒックを指します)。

H) QUAKE Debug Utilities

下記のユーティリティは主として QUAKE 社内で端末の機能を試験するために使用されます。これらの大部分は利用者にとって余り意味の無いものです。しかし幾つかのものは多少の利用価値がありますのでここで概略を説明しています。但し、**Warning** と下記されているコマンドは絶対に使わないで下さい。これらのいずれもが端末を使用不能にしてしまう可能性をもっています。デバッグモードに入るには”d”とタイプして下さい。画面に次の様にコマンドが示されます。

```
AVAILABLE COMMANDS:
'I' Read Analog and Digital Input Values
'S' Serial Port Utilities
```

```
'E' Send Event to Application
'C' Send Comms Cmd to TL
'c' RTC Utilities  (Warning: Do not use)
'f' Flash Utilities  (Warning: Do not use)
'i' Simple DIO utilities  (Warning: Do not use)
'T' TAT Utilities  (Warning: Do not use)
'v' VxWorks Utilities  (Warning: Do not use)
'R' Reboot (utl_pwr_down(0))  'r' st_reboot()
'l' Enable Downlink logging
'e' Set Log Debug Level (now = LOG_NORMAL)
'L' TL Logging Utilities
'P' Toggle logging priority
'w' Wide band search  's' Get dsp samples  'x' Transmit  (Warning: Do not
use)
't' Tune synthesizer to channel 271
'b' Test 1708 bus
```

I) Read Analog and Digital Input Values

（アナログ及びデジタル入力値を読む）

このユーティリティは端末に対してアナログ及びデジタルの各入力値を読むように指示します。外部機器とのインターフェースを設定するのに有用です。デバッグモードから”I”（I の大文字）をタイプしてこのユーティリティに入ります。

J) Serial Port Utilities

（シリアルポートユーティリティ）

このユーティリティは MTS シリアルポート上で個々のデジタル信号を読み取ります。デバッグモードから”S”とタイプして下さい。

K) Send Event to Application

このユーティリティはエミュレータからアプリケーションに対してイベントを送ることが出来、プログラムのデバッグや端末の運用能力を調べる事に有用です。デバッグモードから”E”とタイプして入ります。イベントが選択されると直ちに端末へ送られ、アプリケーションイベントテーブルの内容に従って実行されます。

L) Send Comms Cmd to TL

このユーティリティは、エミュレータから通信コマンドを TL（トランスポート層）へ送り

ます。デバッグモードから"C"をタイプして入ります。通信コマンドに関しては、オーブコムのシリアルインターフェース仕様書を必ず参照して下さい。

M) Reboot

このユーティリティはエミュレータから端末を再起動させるものです。デバッグモードから"R"をタイプして下さい。

N) Enable Downlink Logging

このユーティリティはダウンリンクのログ採りをエミュレータから止めるものです。受信機がオン状態の場合、画面上には衛星からのダウンリンク状況がスクロールします。Quake ロガーからプログラムをタイプしている時には煩わしく感じる事があります。このユーティリティでデータストリームを停止させます。デバッグモードで"l"（小文字のエル）をタイプして下さい。

O) Wide Band Search

このユーティリティはエミュレータから端末の受信機をオンするものです。デバッグモードから"w"をタイプします。この場合、受信機は全ての衛星チャネルをサーチし（これには数秒しか掛かりません)、衛星を補足するまで継続します。

8. Quake アプリケーションプログラミング

これまでの章では Quake アプリケーションについて、アプリケーションイベントテーブルやメッセージテーブル、センサーテーブルなどの作成と編集に関して記述してきました。また、Quake ロガーユーティリティの使い方や、そのデバッグにおける利用法、外部機器とのインターフェースなども記述してきました。ここからは Quake 端末のプログラミングにおけるより詳細な部分と幾つかのサンプルプログラムを紹介していきます。

基本ソフトウェア概観の章で述べましたが、アプリケーションには既に定義されている多くのイベントやレスポンスアクションがあります(それらについては Appendix の A 及び B に説明されています)。イベントやレスポンスアクションの幾つかは、その背景にある概念を説明する必要があり、以下にそれらを説明しています。

A) アプリケーションソフトウェアタイマー

ほとんどのアプリケーションは、特定の動作を一定の間隔やタイミングで行う事が求められます。イベントとレスポンスアクションのタイミングを調整する為、アプリケーションイベントテーブルではソフトウェアタイマーが用意されています。次のテーブルは Quake

端末に用意されているタイマーの種類とその説明及び注意を表しています。

タイマー番号		
0～9	揮発性	端末が電源断になると設定が初期化されます。通常使用するタイマーです*。電源入で全ての揮発性タイマーはオフ状態になります。
10~19	不揮発性	タイマーの状況は不揮発メモリーに保持されます。端末のモードに関係無く時間管理が必要な場合にのみこのタイマーを使用します。スリープ状態でこのタイマーが作動しても端末を起動することはなく、次に起動された時に「タイマー」イベントをアプリケーションへ送ります。
20~29	ｳｪｲｸｱｯﾌﾟ	不揮発性タイマーと同一ですが、端末がスリープ状態にある場合には端末を起動します。
30~39	24 時間時刻	ウェイクアップタイマーと同一ですが、作動時間を一日の絶対時間で設定します。

* 可能な限り不揮発性タイマー及びウェイクアップタイマーは使用を避けてください。電源の断入が激しいアプリケーションでこれらのタイマーを使用すると予期せぬイベントを発生させ（その結果として予期せぬ動作も発生させ）る可能性があります。これらのタイマーを使用する際は充分な検討と試験によって不具合が発生しないこと及び電源の断入による影響がないことを確認して下さい。

B) アプリケーションソフトウェアカウンター

多くのアプリケーションではカウンター機能を要求される場合があります。この様な要求の為にアプリケーションイベントテーブル及びメッセージテーブルにソフトウェアカウンターを提供しています。

ソフトウェアカウンターには 2 つの大きな機能があります。

1．そのデータを保存したりメッセージに応用したりする為にイベントの発生回数を数える（もしもタイマーと一緒に使えば経過時間を記録できます)。
2．複数のイベントの発生をきっかけとして動作が要求される場合に、その様なイベントの合算をする。

イベントテーブルでは一つのイベントが複数のレスポンスアクションを指定する事はあっても､逆はあり得ないため、アプリケーションイベントテーブルにおいて複数のイベントを合算させる機能は、これ以外にはありません。

イベントを発生させる為にカウンターを使う事もあります。カウンターを一定値に設定してイベントの発生毎に減算させ、カウンターがゼロになった時に「COUNTER」イベントをアプリケーションに送ります。その様にしてカウンターをゼロにするようなイベントと組み合わせて特定の動作を起こさせます。

次のテーブルは Quake 端末に用意されているカウンターの種類とその説明及び注意を表しています。

カウンター番号		
0～9	揮発性	端末が電源断になると設定が初期化されます。通常使用するカウンターです*。電源入で全ての揮発性カウンターは 0 にセットされます。
10~19	不揮発性	カウンターの値は不揮発メモリーに保持されます。端末のモードに関係無くカウントデータの維持が必要な場合にのみこのカウンターを使用します。

* 可能な限り不揮発性カウンターは使用を避けてください。電源の断入が激しいアプリケーションでこれらのカウンターを使用すると予期せぬイベントを発生させ（その結果として予期せぬ動作も発生させ）る可能性があります。また、不揮発性カウンターは都度フラッシュメモリーへの書き込みを行います。従い、不揮発性カウンターを一日に数百回も動作させるような場合は端末の不揮発メモリーの寿命に影響が出る可能性があります。

C) ユーザーコマンド

ユーザーコマンドは、オーブコムが仕様書で定義しているオーブコムネットワークから端末に送られるデータ/コマンドメッセージの様式の一つで、本文は 5 バイトです。最初の 1 バイトはどの様なアクションを取るかアプリケーションが決めるために使われます。

全てのユーザーコマンドが「USER_CMD」イベントと見なされるわけではありません。基本アプリケーションは先ずユーザーコマンドの中身を検証し、もしも最初の 1 バイトが所定の値である時にのみ USER_CMD イベントを発生させます。次のテーブルに種々の機能を表示します。

コマンドタイプ	制御バイト値 (Param1)	所要パラメータ (Param2~5)	説明
USER_CMD イベント	1~4 の一つ	(0~255)の数値を 1~4 個	最大 4 つまでの USER_CMD イベントをアプリケーションに送ります。制御バイトは幾つのイベントが送られるかを示します。各イベントは制

			御バイトに続くバイトに示される数値をパラメータとして持ちます。
イベントテーブル上のバイト修正	5	(0~255)の数値を4つ	パラメータは各々、イベントテーブル上の行番号、列番号、旧値、新値、を示します。旧値を検証して間違いの無い事を確認した後、値を修正します。
	6	(0~255)の数値を4つ	

ユーザーコマンドは、全ての電子メールソフトウェアから ORBCOMM ネットワークを通じて端末へ送ることが出来ます。ユーザーコマンドは件名欄に[COMMAND:ACK]と記入し、本文は空白のまま 5 バイトのバイナリーファイルを添付します。詳細については、ORBCOMM の ORBCOMM Gateway Customer Access Interface Specification を参照願います。

例１： USER_CMD0 イベントを送ってみましょう

本例では、端末の電子メールアドレスは mysc@orbcomm.net としています。「Hex Edit」や「UltraEdit」といった 16 進数エディターを使って、01 00 00 00 00 というバイナリーファイルを作って端末へ送ります。最初の 01 は一つのイベントの USER_CMD が送られることを意味しています。次の 00 はこの USER_CMD の番号を示します。次の様にメールを作って見ましょう。

宛先： mysc@orbcomm.net
送信者：「あなたのメールアドレス」
件名： [COMMAND:ACK]
本文： （空欄）
添付ファイル：(5 バイトのバイナリーファイル)

注：ユーザーコマンドを正しく受信するには、端末側が以下の条件を満足している必要があります。

1. 端末がオンになっていて、衛星を受信中であること
2. 端末がオフの場合は、次回オンになった時にネットワークに対して送信し、且つ、送信後少なくとも数分間オンのままである事。こうすることで ORBCOMM ネットワーク側は端末がメッセージを受信できる状態であることを認識して、当該コマンドを送信し端末からの回答を得る事が出来る。

3. ob_rout パラメータが、OB メッセージ及びコマンドをアプリケーションに渡すように設定されている事。
4. 使用する電子メールソフトウェアが、本文中に HTML、XML 或いは署名ファイルなどいかなる文字も挿入しない事。現在使用している電子メールソフトウェアをこの様に本文中に何も書き込まないように設定する事が難しい場合は、他のソフトウェアを使用して下さい。
5. 送信者が端末の短縮ダイアルに登録されている事。

端末のアプリケーションイベントテーブルに USER_CMD0 というイベントがあれば、端末は上記のコマンドに対してテーブルに定められたレスポンスアクションをとります。

D) 自動ローミング

自動ローミングは、各メッセージテーブルで指定されるオプションです（注：メッセージ毎に指定する必要があります)。メッセージテーブルでこのオプションを enabled とすると、当該メッセージは自動的にこの端末が登録されているゲートウェイうち、その場で通信が出来る所に送られます。

もしも以下のような条件である場合、メッセージテーブルの自動ローミングを enabled にして下さい。

1. 当該端末が主ゲートウェイのサービス範囲から外に出がちである場合。
2. 当該端末が複数のゲートウェイに登録されており、メッセージをそれらのいずれかから速やかに受信したい場合。
3. 当該端末のある場所では衛星を介してゲートウェイと通信することが常時は叶わない場合で、メッセージをゲートウェイ経由に拘らずグローバルグラムででも受信したい場合。

自動ローミングに設定されたメッセージが送信されるときは、送信が完了するまで基本ソフトウェアがメッセージ送信作業の進捗と視野内の衛星及びゲートウェイの状況を注意深く監視しています。もしも衛星やゲートウェイの状況に変化があった場合は、自動的に当該メッセージの送信ルートが修正されます。この作業はメッセージの送信が完了するまで続けられます。

自動ローミングには、グローバルグラム用とその他のメッセージ用の 2 つの機能があります。グローバルグラムに対して自動ローミングを設定すると、そのメッセージはその時利用可能なゲートウェイ或いは衛星へ送られます。つまり、メッセージテーブルを作成する際にメッセージタイプをグローバルグラムとして自動ローミングを設定すれば、そのメッ

セージはその時に利用可能な手段で（グローバルグラムに拘泥せずに）送信されます。メッセージが作成された時点でゲートウェイが視野内に無ければグローバルグラムとして送信されますし、利用可能なゲートウェイが見えていればグローバルグラムは自動的にメッセージに（もしもデータが 6 バイト以下ならレポートに）変換されて送信されます。この様に、自動ローミングを enable としたグローバルグラムは、通常、端末の所在地に関係なく可能な範囲で最も迅速な通信を提供します。

その他のメッセージ用の自動ローミングは、上記のグローバルグラム用自動ローミングによく似ています。一点、異なるのはそれらのメッセージがグローバルグラムに変換されることが無いことです。メッセージやレポートを自動ローミング enabled として作成すると、視野内にある利用可能なゲートウェイのいずれかに自動的に送信され、もしも送信完了前に当該ゲートウェイの状況が変化した場合には、自動的にメッセージ（宛先ゲートウェイ）を修正します。しかし、ゲートウェイが一つも視野内に無い場合でも、グローバルグラムに変換されることはありません。場所によってはグローバルグラムで送ることによってかえって送達遅れが大きくなることがあり、利用可能なゲートウェイが視野に入るまで待ったほうが遅れが小さくて済む場合などに利用します。

ローミングのアプリケーションで極力送達遅れを小さくしたい様な場合には、グローバルグラムとメッセージあるいはレポートの二通りで送ることを推奨します。通信費用はかさみますが、これらの両方のルートを確実に使うことが出来ます。

なお、「利用可能なゲートウェイ」は、端末の不揮発メモリーにある Desired Gateway List を参照します。端末に同リストが全く入力されていない場合にはすべてのゲートウェイを利用可能とみなします。同リストが入力されている場合は、リストされているゲートウェイとのみ通信が可能（或いは不可能;リスト入力時の設定によります）です。Desired Gateway List の編集等については、QUAKE ロガーユーティリティの章を参照下さい。

E) アプリケーション例

本章はアプリケーション例をいくつか示し、それらがどの様にイベントテーブル、メッセージテーブル及びセンサーテーブルを使用するかを見て頂きます。これらのアプリケーションを検証し、本書の色々な記述を参照頂く事で、さらに複雑なアプリケーションを容易に実現できることと思います。

例 1: アナログ入力測定値を定期的に送る

本例ではアナログ入力チャネル 0、2 及び 5（解像度は 12 ビットとします）の測定値をデフォルトで設定されたアドレスへ毎日 3 回送るレポートを作るようにします。端末は、レポートがゲートウェイに受信されるか或いは 20 分経過したかいずれか早い時点で電源断となるようにします。このアプリケーションのイベントテーブルとメッセージテーブルを以下に示します。

```
==========  Application Event Table  ===========
EVENT              RESPONSE ACTION
POWER_ON            START_DL_ACQ 0
-                  SET_TIMER 0 [0 0 20 0]
-                  SET_COUNTER 0 [0 0 0 3]
-                  READ_ANALOG 0
-                  READ_ANALOG 2
-                  READ_ANALOG 5
ANALOG 0            DECR_COUNTER 0 [0]
ANALOG 2            DECR_COUNTER 0 [0]
ANALOG 5            DECR_COUNTER 0 [0]
COUNTER 0           SEND_MSG 1 [0]
MSG_ACK 0           SET_TIMER 0 [0 0 5 0]
TIMER 0             POWER_DOWN 1 [0 8 0 0]
================================================


============= Msg 01 Message Table =============
Message Type        DEFAULT_REPORT
Auto-Roaming        Disabled
Parameter 1         ANALOG_IN 0  (12 bits)
Parameter 2         ANALOG_IN 2  (12 bits)
Parameter 3         ANALOG_IN 5  (12 bits)
================================================
```

電源投入を受けて端末は START_DL_ACQ を実行し、揮発性タイマー（Timer 0）を 20 分にセットし、揮発性カウンター（Counter 0）を 3 にセットし、アナログ入力 0、2 及び 5 の値を読むように要求します。各アナログ入力の値が読まれるとカウンターが 1 づつ減っていきます。カウンターがゼロに達する、即ち 3 つすべての読値が完了すると、端末はメッセージを送信します。メッセージの受領確認を受信するか 20 分間が経過すると端末は 8 時間スリープに入ります。

例 2:　指定時刻に位置情報を送る

このアプリケーションでは GPS の緯度･経度情報をグリニッジ標準時の 0530 と 1730 にデフォルトの宛先へ送ります。レポートの受領確認を受信するか 15 分経過すると端末はスリープに入ります。このアプリケーションのイベントテーブルとメッセージテーブルを以下に示します。

```
==========  Application Event Table  ===========
EVENT              RESPONSE ACTION
POWER_ON            START_DL_ACQ 0
-                  SET_TIMER 30 [0 5 30 0]
-                  SET_TIMER 31 [0 17 30 0]
-                  SET_TIMER 0 [0 0 15 0]
TIMER 30            GET_POSITION 0
TIMER 31            GET_POSITION 0
POSITION_FIX 0       SEND_MSG 1 [0]
MSG_ACK 1           SET_TIMER 0 [0 0 5 0]
TIMER 0             POWER_DOWN 3 [0 0 0 0]
================================================


============= Msg 01 Message Table =============
Message Type        DEFAULT_REPORT
Auto-Roaming        Disabled
Parameter 1         LATITUDE  (24 bits)
Parameter 2         LONGITUDE  (24 bits)
================================================
```

電源投入を受けて端末は START_DL_ACQ を実行し、24 時間時刻タイマー（Timer 30 及び 31）を各々0530GMT と 1730GMT にセットし、揮発性タイマー（Timer 0）を 15 分にセットします。Timer30 か 31 によって電源がオンとなると端末は GET_POSITION 0 動作を行います。GPS 測位が確定すると、POSITION_FIX 0 イベントが発生し、メッセージ 1 を送信します。メッセージ 1 の受領が確認されるか Timer 0 が切れると、次に 24 時間時刻タイマーが働くまで端末は電源オフとなります。

例 3:　アラーム入力を監視しながら週に一度位置情報を送る

このアプリケーションは週に一度 GPS の緯度･経度情報を送り、デジタル入力を 30 分に一度監視し、もしもデジタル入力が高位となった場合にはアラームレポートを送ります。ア

ラームレポートは一日当たり一通だけです。位置情報を送った後、メッセージの受領が確認されるか 10 分間経過すると端末の電源がオフになります。アラームレポートを送った場合は同様に受領が確認されるか、或いは 30 分が経過した時点で端末の電源がオフになります。このアプリケーションのイベントテーブルとメッセージテーブルを以下に示します。

```
==========  Application Event Table  ===========
EVENT              RESPONSE ACTION
POWER_ON            START_DL_ACQ 0
-                  READ_DIGITAL 0
DIGITAL_ALARM 0     GET_POSITION 0
DIGITAL 0           DECR_COUNTER 10 [0]
-                  SET_TIMER 1 [0 0 5 0]
COUNTER 10          SET_COUNTER 10 [0 0 1 80]
-                  GET_POSITION 1
POSITION_FIX 0      SEND_MSG 1 [1]
-                  SET_TIMER 0 [0 0 30 0]
POSITION_FIX 1      SEND_MSG 2 [0]
-                  SET_TIMER 1 [0 0 10 0]
MSG_ACK 1           SET_TIMER 0 [0 0 5 0]
MSG_ACK 2           SET_TIMER 1 [0 0 5 0]
TIMER 0             POWER_DOWN 1 [0 24 0 0]
TIMER 1             POWER_DOWN 1 [0 0 30 0]
================================================

============= Msg 01 Message Table =============
Message Type        DEFAULT_REPORT
Auto-Roaming        Disabled
Parameter 1         LATITUDE  (24 bits)
Parameter 2         LONGITUDE  (24 bits)
================================================

============= Msg 02 Message Table =============
Message Type        REPORT
Auto-Roaming        Disabled
gwy_id             Default
polled             Default
```

```
serv_type          Default
Recipient          O/R 2
Parameter 1         LATITUDE  (24 bits)
Parameter 2         LONGITUDE  (24 bits)
================================================
```

メッセージ 02 が DEFAULT REPORT ではなく、REPORT である点に注意して下さい。(無線回線で使われる Default Report のヘッダーの中身は端末の設定パラメータが使用されている) このメッセージ 02 はメッセージ 01 とは異なる宛先に送られるようになっています。このためにデフォルトではない O/R アドレスを指定するためにメッセージタイプを REPORT にしています。(異なる種類のメッセージの宛先を違えることによってメッセージ番号や内容をメッセージデータフィールドに明示する必要が無くなり、より多くのデータを送ることが出来るようになります。) なお、このアプリケーション例ではデフォルトの O/R アドレスは 1 であると仮定しています。

電源投入を受けて端末は START_DL_ACQ を実行し、デジタルセンサーテーブル 0 に従って値を読むように要求します。デジタル測定によって DIGITAL_ALARM が返された場合、端末は GPS 位置情報を読み、メッセージ 2 を送るとともに Timer 0 を 30 分にセットします。端末が MSG_ACK を受領するか Timer 0 が 0 になると (いずれか早い方)、端末は電源断となり、24 時間スリープします。デジタル測定から DIGITAL イベントが返された場合は、端末はメッセージを送らず不揮発カウンター (Counter 10) を 1 つ減算します。Counter 10 が 0 になった時、端末は COUNTER 10 イベントを発生させて GPS 位置情報を読み、メッセージ 1 を送るとともに Timer 1 を 10 分にセットします。端末が MSG_ACK を受領するか Timer 0 が 0 になると (いずれか早い方)、端末は電源断となり、30 分間スリープします。

本例で使用しているパラメータは、位置情報を送る間隔を電源オン/オフサイクルの長さで除したものです (7 日間÷30 分、或いは (7x24) 時間÷0.5 時間 = 336)。SET_COUNTER の設定可能範囲は 0~255 ですので、336 を実現するために PRM4 (x256) = 1、PRM5 = 80 としました。

F) アウトバウンドメッセージプロセスと書式

アウトバウンド (OB) メッセージとは ORBCOMM が定義しているメッセージタイプで、データやコマンド情報を端末へ送るためのものです。OB メッセージは最大 8,000 バイトのデータを送ることが出来ます。通常の OB メッセージにテキストコマンドを入力すること

でアプリケーションコマンドを送ることが出来ます。

アウトバウンドグローバルグラム（データグラムとも言います）は OB メッセージに似ていますが、ゲートウェイの無い地域にある端末にデータを届けることが出来るように衛星の蓄積転送機能を使います。グローバルグラムは最大で 182 バイトに制限されています。現在のところ、アプリケーションコマンドをグローバルグラムで送ることは出来ません。

アプリケーションコマンドは OB メッセージの件名欄に以下の書式で記入します。

Command[Filename][Param1][Param2]Password[Ack]

全てのテキストとパラメータは ASCII で入力してください。OB メッセージを使ってアプリケーションに送れるコマンドは 4 種類です。これらは端末の不揮発メモリーにあるデータファイルのアップロードやダウンロード、修正に使用します。（コンフィギュレーション情報やアプリケーションテーブルも全て端末の不揮発メモリーに保存されています。従って、これらのコマンドを使って端末のコンフィギュレーションを変更することも可能です。）

コマンド	コマンドライン	必要なパラメータ	概要/注意
不揮発ファイルを送る	SEND	ファイル名	ファイル名（例:"APL.NVM"）で指定されたファイルをインバウンドメッセージの添付バイナリーファイルとして送信者のアドレスへ送り返します。
不揮発ファイル全てのリストを送る	SEND DIR		全ての不揮発ファイルの名前とサイズを送信者のアドレスへ送り返します。
不揮発ファイルのロード	LOAD	ファイル名、添付バイナリーファイル	添付ファイルは指定されたファイル名で不揮発メモリーに書き込まれます。
不揮発ファイルの修正	MDFY	ファイル名、ファイルインデックス、データ長、添付バイナリーファイル	ファイル名で指定された不揮発ファイルに、添付バイナリーファイルに含まれるデータバイトを書き込みます。ファイルインデックスパラメータ（Param1）は書き込み開始場所のインデックスを表し、データ長パラメータ（Param2）は書き込まれるデータのバイト数を表します。

イベントをアプリへ送る	AEVT	イベント名、（必要なら）イベントパラメータ	アプリケーションへイベントを送り、アプリケーションイベントテーブル中の対応する動作を実行させます。イベントパラメータはイベント名の後にコロン（":"）をいれてパラメータ値を指定します。
GPS コマンド	GPSC	種々	「GPS リモートコマンド」の項を参照ください。

リモートコマンドで修正されたアプリケーションテーブルは不揮発メモリーにセーブされる前にその書式が正しいかどうか、検証されます。トランスポート層のコンフィギュレーションパラメータファイルが無線で修正された場合、RAM のコンフィギュレーションパラメータもアップデートされます。

端末のコンフィギュレーションに及ぼすこれらコマンドの影響を考えて、各コマンドにはパスワードが設定されています。このパスワードは特定の人間にのみ公開されており、安全を考えて本書では明らかにしていません。なお、（4 桁の数字の）PIN が設定されている端末ではその PIN がパスワードになっています。

端末に送られたコマンドの検証のため、オプションとして受領確認メッセージを要求することが出来ます。これはコマンドの最後に"ACK"というテキストを追加すれば可能になります。このオプションは"LOAD"と"MDFY"コマンドのみで実行可能です。("SEND"コマンドでは一種予備的な機能になります。)

例 1:　端末のアプリケーションイベントテーブルを要求する

この例では端末のメールアドレスを mysc@orbcomm.net、PIN を 0345 と仮定しています。通常の電子メールソフトを使って次のような電子メールを作ります。

TO:　mysc@orbcomm.net
FROM: [your email address]
SUBJECT: **SEND APL.NVM 0345**
MSG BODY: [blank]

このメッセージが送られると数分後に APL.NVM というファイルが添付された返信を端末から受け取れます。

但し、端末は OB メッセージを受信するための以下の条件を満たしていなければなりません。

1. 端末がオンになっていて、衛星を受信中であること
2. 端末がオフの場合は、次回オンになった時にネットワークに対して Message Enquiry を送信し、或いは、メッセージを送信後少なくとも数分間オンのままである事。こうすることで ORBCOMM ネットワーク側は端末がメッセージを受信できる状態であることを認識して、当該コマンドを送信し端末からの回答を得る事が出来る。
3. ob_rout パラメータが、OB メッセージ及びコマンドをアプリケーションに渡すように設定されている事。
4. 使用する電子メールソフトウェアが、本文中に HTML、XML 或いは署名ファイルなどいかなる文字も挿入しない事。現在使用している電子メールソフトウェアをこの様に本文中に何も書き込まないように設定する事が難しい場合は、他のソフトウェアを使用して下さい。
5. 送信者が端末の短縮ダイアルに登録されている事。

例 2: 端末内の全ての不揮発ファイルリストを要求する

本例でも例１と同じ端末アドレスと PIN を持つものとします。通常の電子メールソフトを使って次のような電子メールを作ります。

TO: mysc@orbcomm.net
FROM: [your email address]
SUBJECT: **SEND DIR 0345**
MSG BODY: [blank]

このメッセージが送られると数分後にメッセージ本文にファイル名とサイズのリストを記した返信を端末から受け取れます。

例３: 新しいアプリケーションイベントテーブルを端末に送り Ack を要求する

本例でも例１と同じ端末アドレスと PIN を持つものとします。通常の電子メールソフトを使って次のような電子メールを作ります。

TO: mysc@orbcomm.net
FROM: [your email address]
SUBJECT: **LOAD APL.NVM 0345 ACK**
MSG BODY: [blank]
ATTACHED FILES: [attach new APL.NVM file here]

端末は受信された新しい APL.NVM ファイルを不揮発メモリーにロードします。コマンドに「ACK」オプションが指定されているので、端末はあなたのメールアドレス宛に「LOAD APL.NVM ACK」という件名のメールを返信してきます。これによってコマンドが成功裏に端末に受信されたことが分かります。

メッセージテーブルファイルの様ないくつかの不揮発ファイルはルートディレクトリ下のディレクトリに保存されています。これらのファイルを無線回線で書き換えるには、パスを指定する必要があります。例えば、下記のメールはメッセージテーブル 0 を書き換えて、ACK を要求するものです。

TO:　mysc@orbcomm.net
FROM: [your email address]
SUBJECT: **LOAD /MSGTBL/MSGTBL00.NVM 0345 ACK**
MSG BODY: [blank]
ATTACHED FILES: [attach new MSGTBL00.NVM file here]

ファイルの LOAD コマンドは極力使用を避けて下さい。端末の不揮発ファイルを不正に書き換えてしまうと、端末が動作しなくなる恐れがあります。このコマンドの使用にあたっては、事前に検討と試験を充分に行ってください。

例 4:　TIMER 0 イベントをアプリケーションに送る
本例でも例 1 と同じ端末アドレスと PIN を持つものとします。通常の電子メールソフトを使って次のような電子メールを作ります。

TO:　mysc@orbcomm.net
FROM: [your email address]
SUBJECT: **AEVT TIMER:0 0345**
MSG BODY: [blank]

イベントをアプリケーションに送るコマンドは可能な限り使用を避けて下さい。アプリケーションにイベントを送ると予想外の結果を生じ、ひいては端末が作動しなくなる恐れがあります。このコマンドの使用にあたっては、事前に検討と試験を充分に行ってください。

G)　GPS リモートコマンド

GPS の位置情報セットや指定区域の設定を可能な限り簡単に行えるように数々のコマンド

が用意されており、これによって端末の設定を無線で速やかに行うことが出来ます。GPS リモートコマンドの一つの特徴は、幾つものコマンドを一つのメッセージタイトルに列記することが出来ることです。例えば、既存のロケーションセットに新たな場所を加えたり、他のロケーションセットの指定区域の指定半径を変更したり、また他のロケーションセットを削除したりといったことを一つのメッセージで行うことが出来ます。

GPS リモートコマンドの入力は先ず「GPSC」という文字列を件名欄に入力します。これに続けて下記のコマンドを列記します。書き込まれたコマンドは左から右へ順次実行されます。（件名欄は８０文字に制限されていますので、この制限を越えないように注意して下さい。）

コマンド	コマンドライン	必要なパラメータ	概要/注意
新たな地点を Location Set に加える	ADDLOC	Location Set 番号、緯度、経度、「地点名」	地点名はオプションで、英字１０文字以内。
地点を書き換える	OVRLOC	Location Set 番号、緯度、経度、「地点名」	指定された Location Set の最後の地点を指定された地点データで上書きします。地点名はオプションで、英字１０文字以内。
Location Set から地点を削除する	DELLOC	Location Set 番号、Set 中の地点番号	もしも最後の地点を削除しようとしていて、Set 中に幾つの地点が登録されているか分からない場合は、９９９といった大きな地点番号を指定してください。
Location Set を削除する	DELSET	Location Set 番号	
円形境界の半径を設定する	SETRAD	センサーテーブル番号、半径（m）、High/Low	全てのセンサーテーブルの半径を設定するにはセンサーテーブル番号を”A”と指定してください。High/Low の指定は”H”或いは”L”と略しても構いません。

例:　（混乱を避けるために件名欄の最後に付けるべきパスワードを省略しています）

１．Location Set ３に南緯２２.３°、北緯 10.2°の地点を追加する。
Subject: **GPSC ADDLOC=3:-22.3:10.2**

２．Location Set １に北緯 32.5°、西経 117.5°の地点を「SAN_DIEGO」という地名で追加する。

Subject: **GPSC ADDLOC=1:32.5:-117.5:SAN_DIEGO**

3．Location Set 2 の最後の地点を北緯 38.5°、西経 105.0°の地点に上書きする。地名を「DENVER」とする。

Subject: **GPSC OVRLOC=2:38.5:-105.0:DENVER**

4．Location Set 4 の最後の地点を北緯 36.1°、西経 101.3° 「YARD1」という地点に上書きし、同 Set 4 に「YARD 2」という北緯 36.2°、西経 101.4°の地点を追加する。

Subject: **GPSC OVRLOC=4:36.1:-101.3:YARD1 ADDLOC=4:36.2:-101.4:YARD2**

5．GPS センサーテーブル 0 の半径を 5km に設定し、高位（Set 中の全ての地点からも指定距離以上の場所）となった場合にアラームを出させる。

Subject: **GPSC SETRAD=0:5000:H**

6．全ての GPS センサーテーブルの半径を 10km に設定し、低位（Set 中のいずれかの地点から指定距離内の場所）のアラームを設定する。

Subject: **GPSC SETRAD=A:10000:L**

7．Location Set 1 の最終地点を削除する。

Subject: **GPSC DELLOC=1:999**

8．Location Set 2 を削除する。

Subject: **GPSC DELSET=2**

H) アプリケーションメッセージの保存

アプリケーションによってはメッセージを不揮発メモリーに保存し、そのメッセージが送達されたことを確認する必要があります。特に端末の電源が不安定な状態ではこの様な機能が有効です。メッセージの保存は、アプリケーションイベントテーブルで各メッセージの SEND_MSG レスポンスアクションのパラメータとして設定することが出来ます。

メッセージ保存は以下のような場合に有効です:

1．電源の不測の遮断が予想される場合、

2．メッセージの逸失が絶対に許されない場合、

3．メッセージが送信される前に電源遮断が起きる可能性があり、次に電源入となった時点で当該メッセージを送信したい場合

上記のような事情が無い場合はメッセージの保存は推奨されません。

1． 予想しない時点でメッセージが送られてきて混乱を生じる可能性がある

2． 送信できなかったメッセージを保存するのは通常次善の策になる。一回の電源入セッションの間にあるメッセージが送信完了せず保存され、次のセッションで新しいメッセージと一緒に送信された場合、古いメッセージはもはや意味を持たないケースがあ

る

メッセージ保存機能は、メッセージ番号毎に一つのメッセージを保存します。例として、Message 0（メッセージテーブル 0 に従って作られたメッセージ）がある電源入セッションで送信され、受領確認前に不測の電源断が発生した場合には、次回の電源入セッションで再度送信が行われます。しかし、このセッションでもしも新しい Message 0 が作られてトランスポート層に送られると、IB キューには二つの Message 0 が存在し、また電源断が起きると古いセッションの Message 0 は失われます。

9. APPENDIX

A) Appendix A　アプリケーションイベント

イベント名	必要なパラメータ	概要
POWER_ON	無し	方法の種を問わず（DTR 信号、RTC 起動信号、電源接続等）本体電源が投入された場合に発生する。
TIMER	0 – 39	アプリケーションソフトウェアタイマーのいずれかがタイムアップとなった場合に発生する。パラメータはどのタイマーかを表す
POSITION_FIX	0 – 15	GPS 測定が完了したことを表す。パラメータはどのセンサーテーブルを用いて測定したかを示す。
MSG_ACK	0 – 99	トランスポート層から受領確認を受信したことを示す。パラメータは MHA 番号。
ANALOG_ALARM	0 – 15	アナログ測定が完了し測定値が設定されたアラーム閾値超であることを示す。パラメータはどのセンサーテーブルを用いて測定したかを示す。
DIGITAL_ALARM	0 – 15	デジタル測定が完了し測定値が設定されたアラーム閾値超であることを示す。パラメータはどのセンサーテーブルを用いて測定したかを示す。
POSITION_ALARM	0 – 15	GPS 測定が完了し測定値が設定されたアラーム閾値超であることを示す。パラメータはどのセンサーテーブルを用いて測定したかを示す。
SAT_IN_VIEW	0 又は 1	トランスポート層の satellite in view フラッグの状態が変化したことを表す。0 は衛星が視野に無いことを表す。
ANALOG	0 – 15	アナログ測定が完了したことを表す。パラメータはどのセンサーテーブルを用いて測定したかを示す。

DIGITAL	0 – 15	デジタル測定が完了したことを表す。パラメータはどのセンサーテーブルを用いて測定したかを示す。
COUNTER	0 – 19	アプリケーションソフトウェアカウンターのいずれかが 0 となる或いは閾値を超えた場合に発生する。パラメータはどのカウンターかを表す
USER_CMD	0 – 255	衛星からユーザーコマンドを受信し、アプリケーションにパラメータで定められたイベントが渡されたことを示す。
MTS_DTR	0 – 1	MTS ポートの DTR 信号の状態が変化したことを示す。パラメータ 1 は、ラインが低位（負電圧）から高位（正電圧）へ変化したことを表す。
SER_MEAS	0 – 15	シリアルセンサー測定が完了したことを表す。パラメータはどのセンサーテーブルを用いて測定したかを示す。
SER_ALARM	0 – 15	シリアルセンサー測定が完了し測定値が設定されたアラーム閾値超であることを示す。パラメータはどのセンサーテーブルを用いて測定したかを示す。
J1708_MEAS	0 – 15	使用しません
J1708_ALARM	0 – 15	使用しません
SPEED_ALARM	0 - 15	GPS 測定が完了し測定されたスピード値が設定されたアラーム閾値超であることを示す。パラメータはどのセンサーテーブルを用いて測定したかを示す。

SAT_IN_VIEW イベントは、通常、イベントテーブルでは使用しません。Sat_In_View 状況は、特に受信状態が弱い場合には頻繁に変化します（例えば衛星が低仰角にある場合や、雑音或いは遮蔽物がある場合など）。これにより、Sat_In_View イベントが複数発生し、レスポンスアクションも重複して実行されてしまいます。

B) Appendix B　アプリケーションレスポンス

レスポンスアクション	必要なパラメータ	概要
CFG_DIGITAL	0 – 8 0 – 255	デジタル I/O の出力或いは入力の設定をする。最初のパラメータはチャネル番号（0－7、8 は全チャネルの意味）。二つ目のパラメータは設定（0 は入力に設定、1 は出力に設定）で、全チャネル設定の場合は LSB から各ビットが順にチャネル 0 － 7 の設定を表す。
READ_ANALOG	0 – 15	アナログ測定を命令する。パラメータはどのセンサー

		テーブルを使用するかを示す。
READ_DIGITAL	0 – 15	デジタル測定を命令する。パラメータはどのセンサーテーブルを使用するかを示す。
SET_DIGITAL	0 – 8 0 – 255	デジタルラインの状態を高/低に設定する。最初のパラメータはチャネル番号（0−7、8 は全チャネルの意味）。二つ目のパラメータは設定（0 は入力に設定、1 は出力に設定）で、全チャネル設定の場合は LSB から各ビットが順にチャネル 0 − 7 の設定を表す。 当該ラインが出力に設定されていない場合には、このアクションは意味を持たない。
SET_TIMER	0 – 39 4x(0 – 255)	ソフトウェアタイマーを設定する。最初のパラメータはタイマー番号を表し、二つ目のパラメータは 4 つの数字で設定時間の日、時間、分、秒を表す（24 時間時刻タイマーの場合は日は無視され、後の 3 つで時、分、秒、を意味する。）。
GET_POSITION	0 – 15	GPS 測位を命令する。パラメータはどのセンサーテーブルを使用するかを示す。
POWER_DOWN	0 – 3 4x(0 – 255)	端末の電源を遮断する。最初のパラメータは休止時間の計算方法を表す。 0: 5 分間（後続のパラメータは無視される） 1: 後続の 4 つの数字で表される時間（日、時間、分、秒）の間休止する 2: 後続の 3 つの数字で表される時刻（{日に相当する数値は無視}、時、分、秒）まで休止する 3: 次のウェイクアップタイマーや 24 時間時刻タイマー、或いは DTR 信号によって起動されるまで休止する 注) POWER_DOWN で設定した休止期間終了以前にウェイクアップ或いは 24 時間時刻タイマーが動作した場合はタイマーの動作が優先します。
MSG_ENQ	無し	全ての OB メッセージを要求する通信コマンドを送信する。
SEND_MSG	0 – 99 0 – 1	メッセージを作成、送信する。最初のパラメータはどのメッセージテーブルを使用するかを表し、二つ目のパラメータはメッセージを不揮発メモリーに保存するかどうか（0 は保存、1 は保存しない）を表す。

START_DL_ACQ	0 – 3	衛星のダウンリンク補足を開始する。 注）パラメータはサーチ方法を指定しますが、通常は 0（Wide Band Search）を使用してください。
SET_COUNTER	0 – 19 4x(0 – 255)	ソフトウェアカウンターを設定する。最初のパラメータはカウンター番号を表し、二つ目のパラメータは 4 つの数字で度数を設定する。後方から順に低い位を表す（各パラメータは 0~255 の範囲で設定可能です。0 は 1 を表します。255 よりも大きな数字は 256 進数として 4 つのパラメータで表現してください。）。
INCR_COUNTER	0 – 19 0 – 255	ソフトウェアカウンターのカウントを増加させる。最初のパラメータはカウンター番号を表し、二つ目のパラメータは増加量（0 は 1 カウントの意味です）を表す。
DECR_COUNTER	0 – 19 0 – 255	ソフトウェアカウンターのカウントを減算させる。最初のパラメータはカウンター番号を表し、二つ目のパラメータは減算量（0 は 1 カウントの意味です）を表す。
READ_SERIAL	0 – 15	シリアルセンサー測定を命令する。パラメータはどのセンサーテーブルを使用するかを示す。
COMMS_CMD	0 – 29 0 – 255	トランスポート層に通信コマンドを送る。 コマンド詳細については ORBCOMM の Serial Interface Sepcificaiton、Communications Command Packet Definition を参照してください。
SET_GEOFENCE	0 – 15 0 – 1	最初のパラメータで示される Location Set の最後に最新の GPS 測定位置を追加する。二つ目のパラメータが 0 の場合は単純に追加し、0 以外の場合は Set 中の最後のものに上書きされる。（指定された Set が無い場合は新規に作成されます。）
DEL_GEOFENCE	0 – 15 0 – 255	最初のパラメータで示される Location Set 中の、二つ目のパラメータで示される地点を削除する。二つ目のパラメータが、Set 中の地点の数よりも大きい場合は最後のものが削除される。
SAVE_DATA_FILE	0 – 255 0 – 99 0 - 1	最初のパラメータは書き込むファイルの番号で、二つ目はどのメッセージテーブルに指定されたデータをセーブするかを指定し、最後のパラメータは、(0) 上書きするか、(1) 書き足すか、を指定します。File は

		SCO_MSGS/DATAxxx.NVM という名前（xxx はファイル番号）でセーブされます。

注意事項:

1．CFG_DIGITAL で、出力に設定したデジタルチャネルを低インピーダンス回線に接続すると端末ハードウェアに障害を起こす可能性があります。このコマンドの使用にあたっては事前に検討と試験を充分に行ってください。

2．START_DL_ACQ はどんなアプリケーションでも必要です。この動作を行わないと全てのメッセージの受発信ができません。START_DL_ACQ を POWER_ON イベントに対応する動作としてイベントテーブルに入れておくことを推奨します。

3．POWER_DOWN 動作は、端末が IB メッセージ送信後少なくとも 3~5 分間は衛星との通信状態にあった後、実行されるようにして下さい。もしも電源遮断が IB メッセージの受領確認直後に行われると、ユーザーコマンド、アプリケーションコマンド、ORBCOMM システムコマンドなどを端末が受領できません。この場合、端末のアップデートが出来なくなってしまいます。

C) Appendix C　メッセージテーブルのパラメータ

パラメータ	必要なﾊﾟﾗﾒｰﾀ	初期設定長(bit)	形式	概　要
ANALOG_IN	0 – 7	8	N	アナログ入力の最新値。パラメータはチャネル番号。
DIGITAL_IN	0 – 8	8	N	デジタル入力の最新値。パラメータはチャネル番号(0-7 のみ。8 は全チャネルの意味で、ビットマップ上の 0x01 がチャネル 0 をあらわす)。
LATITUDE	無し	24	N	ORBCOMM 書式による最新の緯度。
LONGITUDE	無し	24	N	ORBCOMM 書式による最新の経度。
ALTITUDE	無し	24	N	32 ビット整数で表される最新の高度（単位: m）。
CURRENT_TIME	無し	32	N	JS 書式(80 年 1 月 6 日 0 時からの経過秒数)による現在時間（GMT）。
ELAPSED_TIME	0 – 15	32	N	パラメータで指定されるデジタルセンサーテーブルに記されているタイマーの現在経過時間。

COUNTER_VALUE	0 – 19	32	N	パラメータに指定されるカウンターの現在値。
USER_DATA_BYTE	0 – 255	8	N	入力されたパラメータの値をそのままメッセージに挿入します。
SERIAL_DATA*	0 – 15	無し	T	シリアルセンサーの最新読値。パラメータはシリアルセンサーテーブル(データ長指定は無視されます)。
SPEED	無し	16	N	GPS が計測した速度の最新値(km/h)
HEADING	無し	16	N	GPS が計測した進行方向の最新値(度)。真北を 0 とし、北→東→南と表示する。
LAST_USER_CMD_DATA	0 – 4	1 ﾊﾞｲﾄ	T	その時点で記録されている最新の User Command データ(電源断があると記録は無くなります)。入力されるパラメータは先頭から何バイト目からコピーするかを示します（0 は先頭からの意味)。データ長パラメータは、メッセージにコピーされるバイト数を表す。
LAST_OB_MSG_SUBJ*	0 – 255	無し	T	その時点で記録されている最新の OB メッセージの件名(電源断があると記録は無くなります)。入力されるパラメータは先頭から何バイト目からコピーするかを示します（0 は先頭からの意味)。データ長パラメータは、メッセージにコピーされるバイト数を表す(0 を入力すると件名全体をコピーします)。末尾の Null はコピーされない。
FILE_DATA*	0 – 255	無し	T	SAVE_DATA_FILE で保存されたファイルのデータ。

注）

* 一つのメッセージテーブルには可変長メッセージパラメータを一つだけを使用することが出来、そのパラメータは必ずテーブルの最後のパラメータである必要があります。これは端末から送られるデータをデコードする際の混乱を排除するためです。

** N = 単一値の数値データ

T = マルチバイトのテキストデータ

メッセージデータのフォーマット

メッセージテーブルの全てのパラメータは以下のいずれかになります:

- 単一値の数値データ
- マルチバイトを含むテキストデータ

単一地数値データは常に LSB が先頭となるようにフォーマットされますので、低位バイトから着信します。一例として、アナログ入力値が 1,000（@16 ビット）だった場合を考えます。十進数の 1,000 は 16 進数では 0x03E8 となり、これは（0x03 x 256）+ 0xE8 です。LSB は 0xE8 で、MSB は 0x03 ですので、このパラメータはメッセージ中に 0xE8 が最初のバイトとして、0x03 が二つ目のバイトとして挿入されることになります。この点は端末から送信されてくるデータを解読する際に留意してください。また、先に着信するデータ長が 8 ビットの倍数ではない場合には、当該データの先頭がバイト境界に合致しないケースがあることも留意してください。

テキストデータは、ストリングのまま着信します。例えば"HELLO"という文字列は H が最初に、その次に E が、という具合に挿入されます。

メッセージパラメータは、メッセージテーブルに記載された順番でメッセージに挿入されます。

メッセージテーブルのメッセージパラメータには必ず同パラメータ長（= データ長）を示す数字がついています。このデータ長については以下に述べるようなルールがあります。

単一値数値メッセージパラメータの種類

数値パラメータ（= 数値データ）はいかなるサイズにもすることが出来ます。これにより、固定長メッセージに、より多くの情報を入れることが出来るようになります。データ長は実際の当該数値パラメータの長さと高位或いは低位ビット省略などの圧縮の有無などによって決まります。

データ長の単位はビットです。この種のパラメータは単一の数値として符号無しの最大 32 ビットの整数で表されます。このデータ長を示す「パラメーターデータ長バイト」という指数があります。この指数はメッセージテーブルで各パラメータに示されています。この指数の意味を以下に示します。

パラメータデータ長バイト	意　　味
0	各パラメータのデフォルトデータ長が使用されています。
1~32	指定された数値のビット長がデータ長として割り当てられています。但し、

	COUNTER 数値でデフォルト値を超えているものは、MSB 側から超過分相当のビットを省いて所定データ長に調整されます。
33~255	所定の方法によって、MSB 及び LSB 側のビットを省いてデータ長を調整されます。

マルチバイトテキストメッセージパラメータの種類
テキストデータは、データ長をバイト単位で指定されています。この場合の「パラメータデータ長バイト」の意味は以下の通りです。

パラメータデータ長バイト	意　　味
0	各パラメータのデフォルトデータ長が使用されています。
1-254	指定されたバイト数を使用します。実データ長が指定値を越えている場合は、指定データ長以上の部分は省かれます。
255	データ長は可変です。

シリアルデータの場合、得られたデータ全てを送ることが求められる場合で、データ取得前の時点でデータ長が分からない場合があります。この場合にはパラメータデータ長バイトを 255（可変長）と指定します。また、取得データの長さに拘わらず送信バイト数を、例えば 10 バイトに一定にしたい場合は、パラメータデータ長バイトを 10 と指定します。この場合、得られたデータが 5 バイトの場合は 5 バイトのみが送られますが、50 バイトのデータが得られても最初の 10 バイトのみが送信されます。

D) Appendix D　ダウンリンクロガーデータ

端末の ORBCOMM 受信機が動作すると、QUAKE ロガーモニターに衛星からのダウンリンク情報が表示されます。ORBCOMM 受信機は通常、Acquisition と Receive という二つのモードのいずれかの状態にあります。Acquisition とは受信機が衛星のダウンリンクを探している状態です。この状態では以下のような情報がモニターに送られます:

```
Aq[26Jun02 22:29:48] chan 265 (WB Search) estDplr -2300 Pwr -115 Ebno 5.1
```

ここに示された情報の意味は以下のとおりです:
Aq:　受信機が Acquisition モードであることを示す。
26Jun02 22:29:48:　内部時計の現在時刻(GMT)。
chan 265:　衛星のダウンリンクチャネル。

(WB Search): ワイドバンドサーチが実行されたことを示す。
estDplr –2300: 計算上の周波数ドップラー量(Hz)。
Pwr –115: 計算上の受信電力(dBm)。
Ebno 5.1: 計算上の EbNo(信号/雑音比)(dB)。

Receive は端末が衛星を補足している状態です。この場合でモニターに送られる情報の例を示します:

```
Rx[26Jun02 22:30:06|11.00]Sync(23* 285 04): Dplr -2448 Pwr -103 Ebno 13.9
0/50
```

ここに示された情報の意味は以下のとおりです:
Rx: 受信機が Receive モードであることを示す。
26Jun02 22:30:06: 衛星が示している現在時刻(GMT)。
11.0: 現在の衛星を補足してからの経過秒数。
Sync: Sync セグメントを示す。
(23* 285 04): 現在補足している衛星番号(23)、ダウンリンクチャネル(285)及びダウンリンクフレーム番号(04)。アスタリスク(*)は現在の衛星が通信可能であることを示す。
Dplr –2448: 周波数ドップラー量(Hz)
Pwr –103: 受信電力(dBm)
Ebno 13.9: EbNo(信号/雑音比)(dB)
0/50: 受信セグメント中の不良セグメントと総セグメント数。0/50 とは直前のフレーム中の 50 のダウンリンクセグメント全てがチェックサムエラー無しに受信されたことを表す。

受信状態の端末は、これらの他にも色々なメッセージをロガーへ送ります。その多くは、衛星軌道情報であるとかダウンリンクパケット情報であるとか、ユーザーにとって余り意味のあるものではありません。上に説明しましたメッセージは毎秒ロガーへ送られるもので、ダウンリンクの統計的情報を示すものです。

E) Appendix E　QUAKE 端末の電子メール添付ファイル解析

メッセージテーブルのデフォルト設定を使用する場合、msg_body_type は 14、即ち、データはバイナリー添付ファイル付き電子メールとして受信されます。データは添付バイナリーファイルに格納されていますので、Hex Edit や UltraEdit、Hex View といったソフトを使用されることをお勧めします。データの解析の例を 3 例示します。これらの例では、緯度、経度、現在時刻、経過時間、速度、方向等々のデータの読み方を説明しています。

例 1: 緯度/経度の読み方

通常、緯度/経度は 6 バイトの添付ファイルとして受信されます。最初の 3 バイトが緯度で次の 3 バイトが経度です。

CB 4C 51 D1 B5 AC

緯度は: **51 4C CB** です。順番が転倒することに注意して下さい。
同様に経度は: **AC B5 D1** です。

緯度/経度の計算方法は次のようになります。

ORBCOMM の Codes to Geodeic:

```
double lat, lon, lon_code, lat_code;
lat = 90.-((unsigned long) lat_code/ ((double) 0xFFFFFF)) * 180.;
long= ((unsigned long) lon_code/ ((double) 0xFFFFFF)) * 360.;
if (lon > 180.)
lon = lon –360.;
```

表現を変えると、

```
Latitude =  (lat_code * (-180/16777216)) + 90
Longitude =  (lon_code * (360/16777216)) - 360
```

上記の例を計算してみます。

```
Latitude = (lat_code * (-180/16777216)) +90
        = (51 4C CB * (-180/16777216)) +90
        = (5328075  * (-0.000010728836059570312 5) ) + 90
        = -57.164043 +90
        = 32.835956
Longitude = (lon_code * (360/16777216)) –360
         = (AC B5 D1 * (360/16777216)) –360
         = (11318737 * (0.000021457672119140625) ) - 360
         = 242.873747 - 360
         = -117.126252
```

例 2: 現在時刻

デフォルトのパラメータ長を使用すると、Current_Time は 4 バイトで下記の様の表されま

す。十進数に読み替えられた値は、1980 年 1 月 6 日零時 (GPS タイムがスタートした瞬間) からの秒数を表しています。

28 96 C2 27

LSB MSB

27 C2 96 28 (Hex) = 667063848

以下はユリウス秒を日、月、時、分、秒に換算する関数です。

```
/* utl_convert_JS_to_DMYHMS - Converts a Julian Seconds (Seconds since
*   1/6/80) format time value into an array containing, in order, the
*   corresponding time value in Day, Month, Year, Hour, Minute, Second.
*
* INPUTS: UINT32 JS value
*
* OUTPUTS:
*   bytes[0] = day (1-31)    bytes[1] = month (1-12)   bytes[2] = year (0-99)
*   bytes[3] = hrs (0-23)    bytes[4] = min (0-59)     bytes[5] = secs (0-59)
*
*/
void utl_convert_JS_to_DMYHMS
    (
    UINT32 timeJS,
    UINT8 bytes[]  /*byte array to write to*/
    )
{
    UINT8 month_days[12] = {31,28,31,30,31,30,31,31,30,31,30,31};

    bytes[5] = timeJS % 60; /*seconds*/
    timeJS /= 60;
    bytes[4] = timeJS % 60; /*minutes*/
    timeJS /= 60;
    bytes[3] = timeJS % 24; /*hours*/
    timeJS /= 24; /*now timeJS is integer JD since 00:00:00 Jan. 6 1980*/
    timeJS += 6; /*let's make it since 00:00:00 Jan. 1 1980, and +1 since
```

```
                days start at 1, not 0 like JD.*/
    bytes[2] = 80; /*1980*/

    while(timeJS > (UINT32)(365 + ((bytes[2]%4)==0) )) /*365 (+ 1 if leap year)*/
    {
        timeJS -= (365 + ((bytes[2]%4)==0) ); /*year++ while days > 365/366*/
        bytes[2]++;
    }
    if(bytes[2] >= 100)
        bytes[2] -= 100;
    bytes[1] = 1; /*January*/
    if(!(bytes[2] % 4))
        month_days[1] = 29;
    while(timeJS > month_days[(bytes[1] - 1)])
        timeJS -= month_days[(bytes[1]++ - 1)]; /*month++ while day > days in month*/
    bytes[0] = timeJS; /*day*/
    return;
} /*end utl_convert_JS_to_DMYHMS()*/
```

また、別の方法として以下があります。例題として、現在時刻を **27 C2 96 28** とします。

1．まずこの **27 C2 96 28** を十進数に換算し、667063848 を得ます（データの並び向きに注意して下さい)。

2．この 667063848 から 662256000（これは 80 年 1 月 6 日から 01 年 1 月 1 日午前 0 時までの秒数）を減じます。

667063848 - Your reading
- 662256000 - Seconds from Jan 6,1980 to Jan 1,2001

4807848 – 01 年 1 月 1 日からの秒数

3．この数字をもとに DAYS を 4807848÷60÷60÷24=55.6464 と計算します。ユリウス日は 55 日で、これは 2 月 24 日に相当します。

4．HOURS として 0.6464×24=15.5133 と計算します。これは 15 時を意味します。

5．同様にして、MINUTES を 0.5133×60=30.8、SECONDS を 0.8×60=48 と計算します。

6．この様にして、2001 年 2 月 24 日 15:30:48 を得ます。

例 3: 経過時間、速度、方位、高度等々

本例ではメッセージテーブルに 6 つのパラメータがあるものとします。メッセージテーブルの例を以下に示します。

```
============= Msg 00 Message Table =============
Message Type        MESSAGE
Auto-Roaming        Disabled
gwy_id             Default
polled             Default
ack_level          Default
priority           Default
msg_body_type       Default
Recipient Qty       1
Recipient List      O/R 1
Subject            [NONE]
Parameter 1         ELAPSED_TIME 0  (32 bits)
Parameter 2         SPEED  (16 bits)
Parameter 3         HEADING  (16 bits)
Parameter 4         ALTITUDE  (16 bits)
Parameter 5         ANALOG_IN 6  (8 bits)
Parameter 6         DIGITAL_IN 3  (8 bits)
================================================
```

デフォルトのデータ長であれば、添付データは次のようになります（合計 12 バイト）。

23 FC 4A 00 5B 00 23 00 7D 00 7D FF

これを各項目に分けると次のようになります。

23 FC 4A 00	**5B 00**	**23 00**	**7D 00**	**72**	**FF**
Elapsed Time	Speed	Heading	Altitude	Analog	Digital

経過時間の計算

ELAPSED_TIME パラメータは十進数に換算した数値を 3600 で除します。答えの単位は「時間」です。

受信データ: **23 FC 4A 00**

十進数への換算: **00 4A FC 23** = 4914211

3600 で除する: 4914211÷3600

回答:　1365.058611111 時間

速度の計算

SPEED パラメータは十進数に換算した数値で km/h を表します。

受信データ:　5B 00
十進数への換算:　00 5B = 91
回答:　91km/h

方位の計算

HEADING パラメータは十進数に換算した数値で北からの度数で表します。

受信データ:　23 00
十進数への換算:　00 23 = 35
回答:　035 度

高度の計算

ALTITUDE パラメータは十進数に換算した数値で高度をメーターで表します。

受信データ:　7D 00
十進数への換算:　00 7D = 125
回答:　125 m

アナログ値の計算

ANALOG パラメータは十進数に換算します。

受信データ:　72
十進数への換算:　72 = 114
回答:　114

デジタル値の計算

DIGITAL パラメータは十進数に換算します。

受信データ:　FF
十進数への換算:　FF = 255
回答:　255

F) Appendix F　QUAKE ファームウェアのインストール

QUAKE 端末は出荷時に最新のファームウェアを搭載して出荷されます。しかし、Quake 社では随時新しいファームウェアをウェブ上でリリースしています。この新しいファームウェアをインストールする方法を下記します。

1．先ず、Quake Firmware Loader を入手します。このソフトは Quake のウェブや QuakeTools の CD から入手できます。
2．COM 1 からシリアルケーブルを QUAKE 端末の MTS ポートに繋ぎます。
3．DOS プロンプトから新しいファームウェアと Loader の置いてあるディレクトリへ進みます。
4．DOS プロンプトで”ldfast arm.hex x”と入力します。
5．端末に電源を入れるようにプロンプトが要求しますので、電源を投入する、或いは既に電源が入っている場合は電源の入/切を行ってください。
6．ダウンロードが開始され、画面がスクロールします。
7．ダウンロードが完了するまで PC を操作しないで下さい。
8．ダウンロードが完了しない場合、使用する PC 或いは PC と端末間の回線のスピードが遅い可能性があります。”ldslow arm.hex x”と入力し、上記の 5 からやり直してください。
9．それでもダウンロードできない場合は Quake へご連絡ください。