平成１６年４月２２日
気 象 庁 観 測 部

# 配信資料に関する技術情報(気象編)第 162 号

～1km メッシュ全国合成レーダーGPV の提供等について～

現在、気象庁では 2.5 ㎞メッシュ分解能のレーダーデータを(財)気象業務支援センターを通じて提供しておりますが、平成16年6月から新たに 1km メッシュ分解能の全国合成レーダーデータの提供を開始します。

また、平成17年度の次期アデス整備にあわせ東日本(札幌、仙台、東京の各管区管内)のレーダーエコーデジタル化装置を更新する計画であり、2.5km メッシュ分解能の地方合成及び全国合成レーダーデータの配信をこのシステム更新にあわせて順次終了する計画です。

## 1. 新たに提供を開始するデータ

以下のデータを 1 ファイルにまとめ FTP 方式により提供します。

(1) 1km メッシュ全国合成レーダーエコー強度 GPV

(2) 2.5km メッシュ全国合成レーダーエコー頂高度 GPV

## 2. 提供開始日

平成 16 年 6 月 1 日(火)02 時(協定世界時)

なお、提供開始に先立ち試験配信を行なう予定です。

## 3. データの概要

(1) 1km メッシュ全国合成レーダーエコー強度 GPV

気象庁が保有する全国 20 台の気象レーダーで観測したエコー強度(レーダーで観測される換算降水強度)を 10 分間隔で提供します。水平分解能は 1km メッシュ相当の緯度 0.5 分・経度 0.75 分で、GPV の格子系は等緯度等経度格子です。提供する GPV データの領域は、従来の 2.5km メッシュ全国合成レーダーGPV と同じです。

(2) 2.5km メッシュ全国合成レーダーエコー頂高度 GPV

気象庁が保有する全国 20 台の気象レーダーで観測したエコー頂高度(レーダーで観測される降水エコーの高さ)を 10 分間隔で提供します。水平分解能は 2.5km メッシュ相当(従来は5kmメッシュ相当)の緯度 1.5 分・経度 1.875 分で、GPV の格子系は等緯度等経度格子です。提供する GPV データの領域は 1km メッシュ全国合成レーダーエコー強度 GPV と同じです。

## 4. ファイル形式

今回提供を開始するデータのファイル形式は、全て国際気象通報式 FM92 GRIB 二進形式格子点資料気象通報式(第 2 版)(以下、「GRIB2」という。)です。GRIB2 の詳細については国際気象通報式・別冊に詳しく記述されていますので、当該資料を参照願います。また、今回提供を開始するデータの GRIB2 各節の詳細については、それぞれ別添のとおりです。ファイルサイズはその日の状況により圧縮効率が変わり、エコー強度でおよそ 100～800KB、エコー頂高度でおよそ 10～80KB となります。また、当該データの解読(デコード)処理については、添付の「1km メッシュ全国合成レーダーGPV(GRIB2)の解読処理について」を参照し、サンプルデータが必要な場合は気象業務支援センターに提供しておりますのでご利用下さい。

5. ファイル名

1km メッシュ全国合成レーダーエコー強度 GPV と 2.5km メッシュ全国合成レーダーエコー頂高度 GPV の二つのファイルを 1 つの tar 形式ファイルにまとめ、以下のファイル名で提供します。

Z__C_RJTD_yyyyMMddhhmmss_RDR_JMAGPV__grib2.tar
（ZとCの間及びJMAGPVとgrib2の間にはアンダースコアーが2個設定されている点に注意、その他のアンダースコアーは1個。yyyyMMddhhmmssはデータの観測年月日時分秒をUTC（協定世界時）で設定。）

このtar形式のファイルを展開することで、それぞれ以下のファイル名の1kmメッシュ全国合成レーダーエコー強度GPVと 2.5kmメッシュ全国合成レーダーエコー頂高度GPVを得ることができます。

(1) 1kmメッシュ全国合成レーダーエコー強度GPV
Z__C_RJTD_yyyyMMddhhmmss_RDR_JMAGPV_Ggis1km_Prr10lv_ANAL_grib2.bin
（ZとCの間にはアンダースコアーが2個設定されている点に注意、その他のアンダースコアーは1個。yyyyMMddhhmmssはデータの観測年月日時分秒をUTC（協定世界時）で設定。）

(2) 2.5kmメッシュ全国合成レーダーエコー頂高度GPV
Z__C_RJTD_yyyyMMddhhmmss_RDR_JMAGPV_Gll2p5km_Phhlv_ANAL_grib2.bin
（ZとCの間にはアンダースコアーが2個設定されている点に注意、その他のアンダースコアーは1個。yyyyMMddhhmmssはデータの観測年月日時分秒をUTC（協定世界時）で設定。）

6. 1km メッシュ全国合成レーダーエコー強度GPVの特徴

(1) レーダーデータの精度向上

1km メッシュ全国合成レーダーエコー強度 GPV では、従来の補正方式(前 240 時間(10 日間)のアメダス雨量とレーダーデータの比較による補正処理)に加え、観測時間毎にその 10 分前からの前1時間におけるアメダス雨量とレーダーデータの比較による補正処理も行い、精度向上を図っています。

(2) 処理範囲の拡大

今回提供を開始するデータでは、大東島地方(沖縄県)を処理範囲に含めました。同地方における台風等の監視能力が向上しますが、レーダーの観測ビーム高度が高いため、他の地方と比べて、背の低い降水エコーに対しては補足率が低下しますので、その点に注意してご利用願います。

7. 次期気象レーダー観測処理システムの整備とこれに伴う配信データの変更

気象庁では、平成17年度の次期アデス整備にあわせ東日本(札幌、仙台、東京の各管区管内)のレーダーエコーデジタル化装置を更新し次期気象レーダー観測処理システム(以下、「次期レーダーシステム」という。)を整備する計画です。次期レーダーシステムでは、これまで管区ごとに行なってきたデータ処理を気象庁本庁に集約する予定です。

平成17年7月頃から各レーダーサイトの機器更新作業を順次行う予定ですが、機器

更新後は当該レーダーサイトの 2.5km メッシュデータを作成しません。このため従来配信していた東日本域の10領域の地方合成データ(「東北海道」「西北海道」「北北海道」「北部東北」「南部東北」「東部北陸」「西部北陸」「東部東海」「西部東海」「関東地方」)の配信を、更新作業の完了する平成17年10月までの間に順次廃止していく予定です。これにあわせて、2.5km メッシュ全国合成レーダーGPV も廃止します(添付のレーダーデータ配信計画を参照)。西日本の地方合成データについては、西日本の次期レーダーシステム整備までの間、配信を継続する予定です。

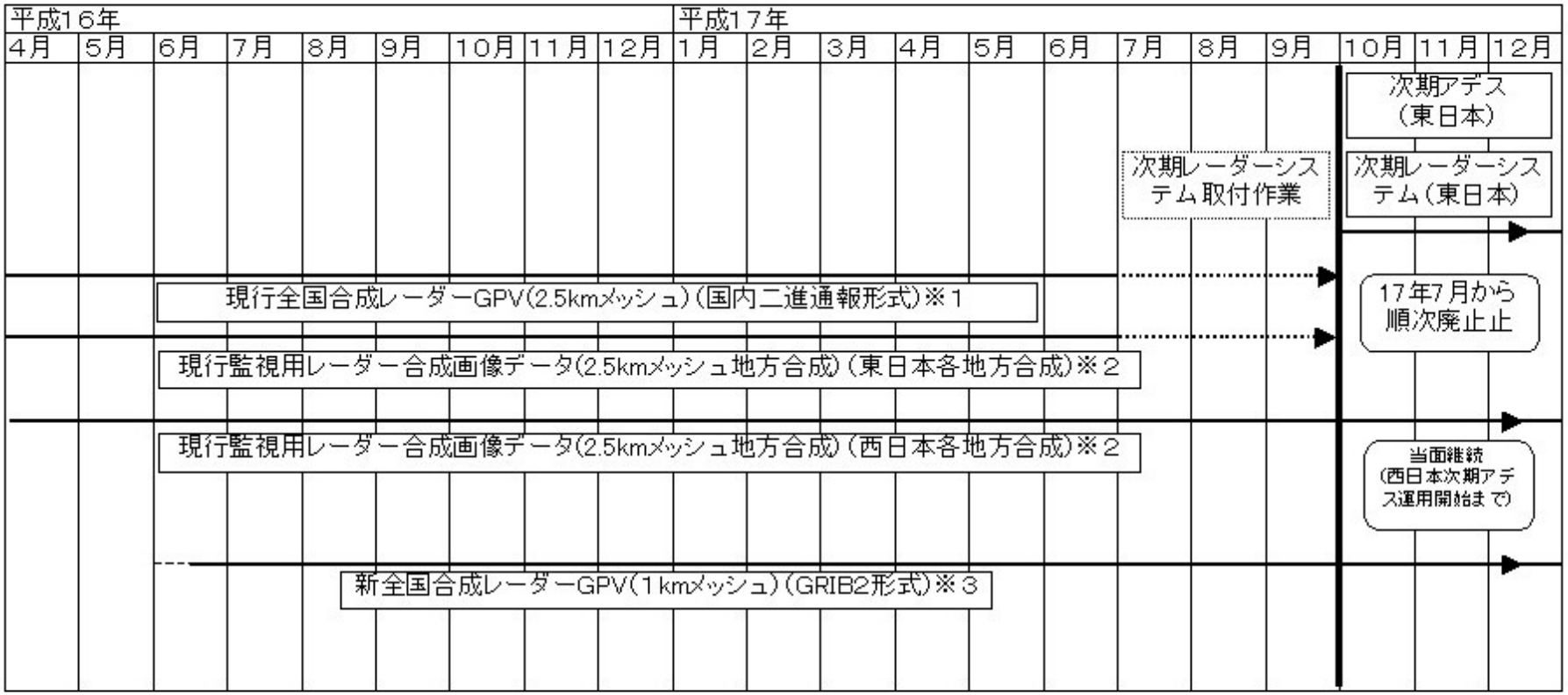
なお、次期レーダーシステムの機器更新作業では各レーダーで2週間程度、運用を休止する予定です。

詳細なスケジュールは、決定次第、お知らせいたします。

(添付資料)

- 1km メッシュ全国合成レーダーエコー強度 GPV フォーマット
- 2.5km メッシュ全国合成レーダーエコー頂高度 GPV フォーマット
- 1km メッシュ全国合成レーダーGPV (GRIB2)の解読処理について
- 次期レーダーシステムの整備と配信データの変更

次期レーダーシステムの整備と配信データの変更




※１ ：「5kmメッシュ全国合成レーダーエコー頂高度」をあわせて提供
※２ ：「5kmメッシュ各地方合成レーダーエコー頂高度」をあわせて提供
※３ ：「2．5kmメッシュ全国合成レーダーエコー頂高度」をあわせて提供

# 1kmメッシュ
# 全国合成レーダーエコー強度GPVフォーマット
# （GRIB2形式　Ver.1.07）

## 注意事項

・合成データの範囲は、東経118～150度、北緯20～48度の領域。この領域を、経度方向には2560格子、緯度方向には3360格子で区切る（合成範囲の図を参照）。経度方向の格子間隔は45秒、緯度方向の格子間隔は30秒（GIS第三次メッ
・データ圧縮にはランレングス圧縮を用いるが、圧縮に用いるレベルの最大値はそのファイル中の最大値を用いるのでファイルによって値が異なる点に注意。
・レーダー、雨量換算係数の運用情報の書式については※2の表を参照。
・換算雨量強度からレベル値への変換は※3の表を参照。
・ファイル名の命名法は下記様式の*yyyyMMddhhmmss*にデータの日時（年月日時分秒）を協定世界時で設定したものとする。

Z_C_RJTD_*yyyyMMddhhmmss*_RDR_JMAGPV_Ggis1km_Prr10lv_ANAL_grib2.bin

最初のZとCの間には半角のアンダースコアーを2個入れる点に注意。他のアンダースコアーは半角1個である。

例えば、日本標準時の2003年5月14日8:20:00のデータなら
Z_C_RJTD_20030513232000_RDR_JMAGPV_Ggis1km_Prr10lv_ANAL_grib2.bin

1kmメッシュ気象庁レーダ全国合成に用いるGRIB2のフォーマットおよびテンプレートの詳細

| 節番号 | 節の名称・該当テンプレート | オクテット（バイトと同じ） | 内容 | 表 | 値 | 備考 |
|---|---|---|---|---|---|---|
| 第0節 | 指示節 | 1～4 | GRIB | | GRIB | アスキーコードで設定する |
| | | 5～6 | 保留 | | missing | |
| | | 7 | 資料分野 | 符号表0．0 | 0 | 気象分野 |
| | | 8 | GRIB版番号 | | 2 | |
| | | 9～16 | GRIB報全体の長さ | | ****** | 第0節から第8節までのトータルのバイト数 |
| 第1節 | 識別節 | 1～4 | 節の長さ | | 21 | |
| | | 5 | 節番号 | | 1 | |
| | | 6～7 | 作成中枢の識別 | 共通符号表C-1 | 34 | 東京 |
| | | 8～9 | 作成副中枢 | | 0 | |
| | | 10 | GRIBマスター表バージョン番号 | 符号表1．0 | 2 | マスター表バージョン2 |
| | | 11 | GRIB地域表バージョン番号 | 符号表1．1 | 1 | 地域表バージョン1 |
| | | 12 | 参照時刻の意味 | 符号表1．2 | 0 | 解析 |
| | | 13～14 | 資料の参照時刻（年） | | ※1 | 協定世界時 |
| | | 15 | 資料の参照時刻（月） | | ※1 | 協定世界時 |
| | | 16 | 資料の参照時刻（日） | | ※1 | 協定世界時 |
| | | 17 | 資料の参照時刻（時） | | ※1 | 協定世界時 |
| | | 18 | 資料の参照時刻（分） | | ※1 | 協定世界時 |
| | | 19 | 資料の参照時刻（秒） | | ※1 | 協定世界時 |
| | | 20 | 作成ステータス | 符号表1．3 | 0 | 現業プロダクト |
| | | 21 | 資料の種類 | 符号表1．4 | 0 | 解析プロダクト |
| 第2節 | 地域使用節 | 不使用 | | | | |
| 第3節 | 格子系定義節 | 1～4 | 節の長さ | | 72 | |
| | | 5 | 節番号 | | 3 | |
| | | 6 | 格子系定義の出典 | 符号表3．0 | 0 | 符号表3．1参照 |
| | | 7～10 | 資料点数 | | 8601600 | 2560*3360 |
| | | 11 | 格子点数を定義するリストのオクテット数 | | 0 | |
| | | 12 | 格子点数を定義するリストの説明 | | 0 | |
| | | 13～14 | 格子系定義テンプレート番号 | 符号表3．1 | 0 | 緯度・経度格子 |
| | | 15 | 地球の形状 | 符号表3．2 | 4 | GRS80回転楕円体 |
| | | 16 | 地球球体の半径の尺度因子 | | missing | |
| | | 17～20 | 地球球体の尺度付き半径 | | missing | |
| | | 21 | 地球回転楕円体の長軸の尺度因子 | | 1 | |
| | | 22～25 | 地球回転楕円体の長軸の尺度付きの長さ | | 63781370 | |
| | | 26 | 地球回転楕円体の短軸の尺度因子 | | 1 | |
| | | 27～30 | 地球回転楕円体の短軸の尺度付きの長さ | | 63567523 | |
| | | 31～34 | 緯線に沿った格子点数 | | 2560 | |
| | | 35～38 | 経線に沿った格子点数 | | 3360 | |
| | | 39～42 | 原作成領域の基本角 | | 0 | |
| | | 43～46 | 端点の経度及び緯度並びに方向増分の定義に使われる基本角の細分 | | missing | |
| | | 47～50 | 最初の格子点の緯度 | 10**-6度単位 | 47995833 | 48N-2/3/8/10/2 |
| | | 51～54 | 最初の格子点の経度 | 10**-6度単位 | 118006250 | 118E+1/8/10/2 |
| | | 55 | 分解能及び成分フラグ | フラグ表3．3 | 48 | 0x30 |
| | | 56～59 | 最後の格子点の緯度 | 10**-6度単位 | 20004167 | 20N+2/3/8/10/2 |
| | | 60～63 | 最後の格子点の経度 | 10**-6度単位 | 149993750 | 150E-1/8/10/2 |
| | | 64～67 | i方向の増分 | 10**-6度単位 | 12500 | 1/8/10 |
| | | 68～71 | j方向の増分 | 10**-6度単位 | 8333 | 2/3/8/10 |
| | | 72 | 走査モード | フラグ表3．4 | 0 | |
| 第4節 | プロダクト定義 | 1～4 | 節の長さ | | 82 | |
| | | 5 | 節番号 | | 4 | |
| | | 6～7 | テンプレート直後の座標値の数 | | 0 | |
| | | 8～9 | プロダクト定義テンプレート番号 | 符号表4．0 | 50008 | 解析雨量と同じ |
| | ここからﾃﾝﾌﾟﾚｰﾄ | | | | | |
| | 4.50008 | 10 | パラメータカテゴリー | 符号表4．1 | 1 | 湿度 |
| | ↓ | 11 | パラメータ番号 | 符号表4．2 | 201 | 10分間降水強度（1時間換算値）レベル値 ※3 |
| | ↓ | 12 | 作成処理の種類 | 符号表4．3 | 0 | 解析 |
| | ↓ | 13 | 背景作成処理識別符 | JMA定義 | 201 | 全国気象庁レーダー合成 |
| | ↓ | 14 | 予報の作成処理識別符 | | missing | |
| | ↓ | 15～16 | 観測資料の参照時刻からの締切時間（時） | | 0 | |
| | ↓ | 17 | 観測資料の参照時刻からの締切時間（分） | | 5 | |
| | ↓ | 18 | 期間の単位の指示符 | 符号表4．4 | 0 | 分 |
| | ↓ | 19～22 | 予報時間 | | -10 | 0x8000000Aを設定する |
| | ↓ | 23 | 第一固定面の種類 | 符号表4．5 | 1 | 地面又は水面 |
| | ↓ | 24 | 第一固定面の尺度因子 | | missing | |
| | ↓ | 25～28 | 第一固定面の尺度付きの値 | | missing | |
| | ↓ | 29 | 第二固定面の種類 | 符号表4．5 | missing | |
| | ↓ | 30 | 第二固定面の尺度因子 | | missing | |
| | ↓ | 31～34 | 第二固定面の尺度付きの値 | | missing | |
| | ↓ | 35～36 | 全時間間隔の終了時（年） | | ※1 | 協定世界時 |
| | ↓ | 37 | 全時間間隔の終了時（月） | | ※1 | 協定世界時 |
| | ↓ | 38 | 全時間間隔の終了時（日） | | ※1 | 協定世界時 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | ↓ | 39 | 全時間間隔の終了時(時) | | ※1 | 協定世界時 |
| | | ↓ | 40 | 全時間間隔の終了時(分) | | ※1 | 協定世界時 |
| | | ↓ | 41 | 全時間間隔の終了時(秒) | | ※1 | 協定世界時 |
| | | | 42 | 統計を算出するために使用した | | 1 | |
| | | ↓ | | 時間間隔を記述する期間の仕様の数 | | | |
| | | ↓ | 43～46 | 統計処理における欠測資料の総数 | | 0 | |
| | | ↓ | 47 | 統計処理の種類 | 符号表4．10 | 1 | 積算 |
| | | | 48 | 統計処理の時間増分の種類 | 符号表4．11 | 2 | 同じ予報開始時刻を持ち、 |
| | | ↓ | | | | | 予報時間に増分が加えられる |
| | | ↓ | 49 | 統計処理の時間の単位の指示符 | 符号表4．4 | 0 | 分 |
| | | ↓ | 50～53 | 統計処理した期間の長さ | | 10 | |
| | | | 54 | 連続的な資料場間の増分に関する | | 0 | |
| | | ↓ | | 時間の単位の指示符 | | | |
| | | ↓ | 55～58 | 連続的な資料場間の時間の増分 | | 0 | 連続的な処理の結果 |
| | | ↓ | 59～66 | レーダー等運用情報 | | ※2 | |
| | | ↓ | 67～74 | 雨量換算係数運用情報 | | ※2 | |
| | | ここまで | | | | | |
| | | ﾃﾝﾌﾟﾚｰﾄ | | | | | |
| | | 4.50008 | 75～82 | 雨量計運用情報 | | missing | |
| | 第5節 | 資料表現節 | 1～4 | 節の長さ | | 519 | |
| | | | 5 | 節番号 | | 5 | |
| | | | 6～9 | 全資料点の数 | | 8601600 | 2560x3360（1km格子の場合） |
| | | | 10～11 | 資料表現テンプレート番号 | 符号表5．0 | 200 | 格子点資料－ランレングス圧縮 |
| | | ここから | | | | | |
| | | ﾃﾝﾌﾟﾚｰﾄ5.200 | 12 | 1データのビット数 | | 8 | |
| | | | 13～14 | 今回の圧縮に用いたレベルの最大値 | | V | Vは実際のデータ中に現れた最大のレベル |
| | | ↓ | | | | | 値（＜＝M） |
| | | ↓ | 15～16 | レベルの最大値 | | M | M=251 |
| | | ↓ | 17 | データ代表値の尺度因子 | | 2 | 10**2の意味 |
| | | ここまで | 16＋2xnn～ | | | | 各レベルnnに対する※3の雨量強度を100倍 |
| | | ﾃﾝﾌﾟﾚｰﾄ5.200 | 17＋2xnn | レベルnnに対応するデータ代表値 | | ※3 | した値を設定。nn=1のときは0とする。（nn＝1～M） |
| | 第6節 | ビットマップ節 | 1～4 | 節の長さ | | 6 | |
| | | | 5 | 節番号 | | 6 | |
| | | | 6 | ビットマップ指示符 | | 255 | ビットマップを適応せず |
| | 第7節 | 資料節 | 1～4 | 節の長さ | | ****** | 第7節のトータルのバイト数 |
| | | | 5 | 節番号 | | 7 | |
| | | ﾃﾝﾌﾟﾚｰﾄ7.200 | 6～ | ランレングス圧縮オクテット列 | | | 資料テンプレート7．200で記述された形式 |
| | 第8節 | 終端節 | 1～4 | 7777 | | 7777 | アスキーコードで設定する |

（注）

第0節最初の「GRIB」と第8節の「7777」のみアスキーコードで設定し、他は全て整数型のバイナリーで設定する。
バイナリー値は、ビッグエンディアンで設定する。
値欄が「missing」の場合そのデータは全ビット1の値、英数字の変数名や「******」は必要な値を設定する。
実際のデータは、ランレングス圧縮後第7節の6バイト目以降に設定する。
※1 第1節と第4節には、共に観測時刻を協定世界時で格納する。年月日時分秒で使用する数値は、
年：4桁の西暦年、月：1-12、日：1-31、時：0-23、分:0-59、秒:0-59 とする。

## ※2 レーダー等運用情報の詳細

### レーダー等運用情報

＜8バイト中の配置＞　（■は値を設定する2ビットを示し、□は0を設定する2ビットを示す）

□ □ □ □ □ □ □ □ □ □ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■

| 64 | 60 | 56 | 52 | 48 | 44 | 40 | 36 | 32 | 28 | 24 | 20 | 16 | 12 | 8 | 4 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | 沖縄SP | 名瀬SP | 石垣島 | 沖縄 | 名瀬 | 種子島 | 福岡 | 室戸岬 | 広島 | 松江 | 大阪 | 名古屋 | 福井 | 静岡 | 長野 | 東京 | 新潟 | 秋田 | 仙台 | 函館 | 釧路 | 札幌 |

＜各レーダーの運用情報2ビットの内容＞

0 . . . . 電文受信なし
1 . . . . 電文受信あり(エコーあり)
2 . . . . 電文受信あり(No Echo)
3 . . . . 電文受信あり(No Operation)

### 雨量換算係数運用情報

＜8バイト中の配置＞　（■は値を設定する2ビットを示し、□は0を設定する2ビットを示す）

□ □ □ □ □ □ □ □ □ □ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■

| 64 | 60 | 56 | 52 | 48 | 44 | 40 | 36 | 32 | 28 | 24 | 20 | 16 | 12 | 8 | 4 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | 沖縄SP | 名瀬SP | 石垣島 | 沖縄 | 名瀬 | 種子島 | 福岡 | 室戸岬 | 広島 | 松江 | 大阪 | 名古屋 | 福井 | 静岡 | 長野 | 東京 | 新潟 | 秋田 | 仙台 | 函館 | 釧路 | 札幌 |

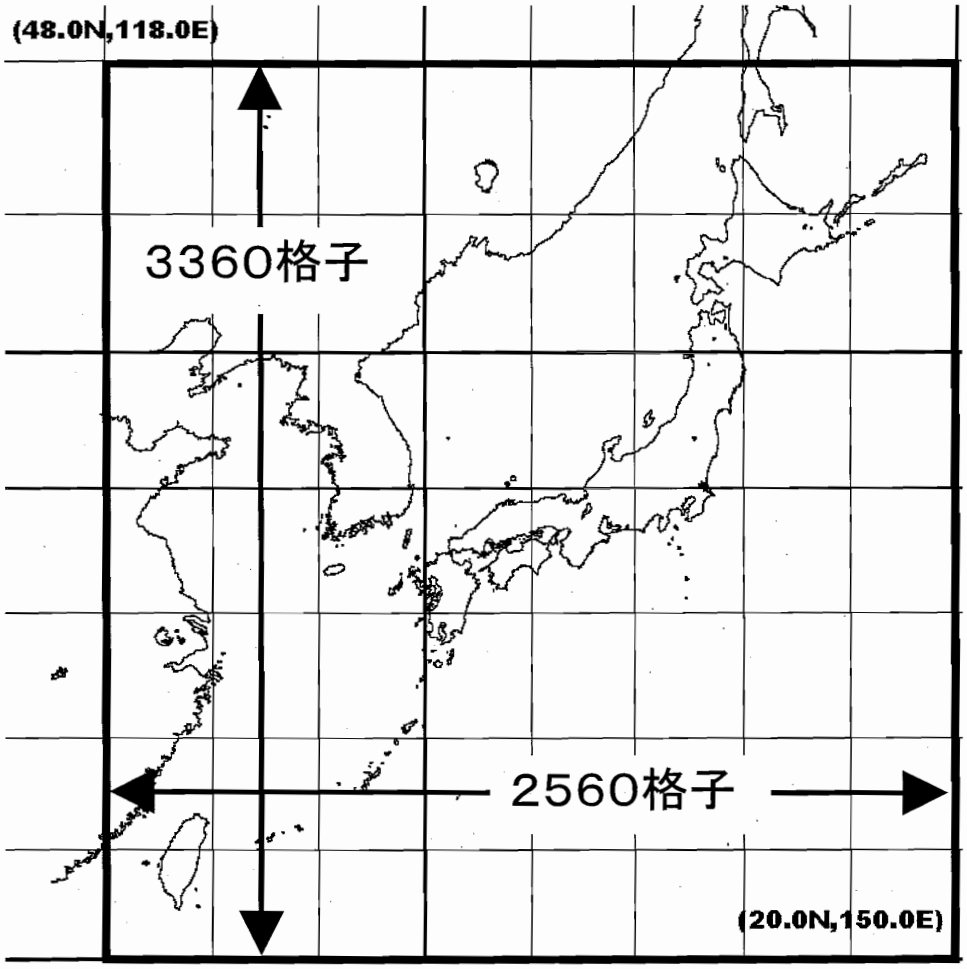
＜雨量換算係数の運用情報2ビットの内容＞

0 . . . . RAM0を使用
1 . . . . レーダー情報作成装置より受信した直近のRAM1を使用
2 . . . . レーダー情報作成装置より受信した過去のRAM1を使用
3 . . . . NAPSより受信したRAM1を使用

## ※3　1kmメッシュ気象庁レーダー全国合成のレベル値（0～251）

| 0～2mm/hは<br>0.1mm/h毎 | データ<br>代表値 | 2～5mm/hは<br>0.25mm/h毎 | データ<br>代表値 | 5～10mm/hは<br>0.5mm/h毎 | データ<br>代表値 |
|---|---|---|---|---|---|
| 0：観測範囲外<br>又は欠測 | | 21：2.0mm/h以上<br>2.25mm/h未満 | 2.13 | 33：5.0mm/h以上<br>5.5mm/h未満 | 5.25 |
| 1：No Echo | 0 | ・ | ・ | ・ | ・ |
| 2：0.2mm/h未満 | 0.1 | ・ | ・ | ・ | ・ |
| 3：0.2mm/h以上<br>0.3mm/h未満 | 0.25 | ・ | ・ | ・ | ・ |
| ・ | ・ | ・ | ・ | ・ | ・ |
| ・ | ・ | ・ | ・ | ・ | ・ |
| 20：1.9mm/h以上<br>2.0mm/h未満 | 1.95 | 32：4.75mm/h以上<br>5.0mm/h未満 | 4.88 | 42：9.5mm/h以上<br>10.0mm/h未満 | 9.75 |

| 10mm/h～180mm/hは<br>1.0mm/h毎 | データ<br>代表値 | 180mm/h以上は<br>2.0mm/h毎 | データ<br>代表値 |
|---|---|---|---|
| 43：10.0mm/h以上<br>11.0mm/h未満 | 10.5 | 213：180.0mm/h以上<br>182.0mm/h未満 | 181 |
| ・ | ・ | ・ | ・ |
| ・ | ・ | ・ | ・ |
| ・ | ・ | ・ | ・ |
| ・ | ・ | ・ | ・ |
| ・ | ・ | ・ | ・ |
| ・ | ・ | ・ | ・ |
| 212：179.0mm/h以上<br>180.0mm/h未満 | 179.5 | 250：254.0mm/h以上<br>256.0mm/h未満 | 255 |
| | | 251：256.0mm/h以上 | 260 |


(48.0N,118.0E)
3360格子
2560格子
(20.0N,150.0E)

# 2.5kmメッシュ
# 全国合成レーダーエコー頂高度GPVフォーマット
# （GRIB2形式　Ver.1.04）

## 注意事項

・合成データの範囲は、東経118～150度、北緯20～48度の領域。この領域を、経度方向には1024格子、緯度方向には1120格子で区切る（合成範囲の図を参照）。経度方向の格子間隔は1.875分、緯度方向の格子間隔は1.5分（2.5km相
・データ圧縮にはランレングス圧縮を用いるが、圧縮に用いるレベルの最大値はそのファイル中の最大値を用いるのでファイルによって値が異なる点に注意。
・レーダーの運用情報の書式については※2の表を参照。
・レベル値の意味は※3の表を参照。
・ファイル名の命名法は下記様式の*yyyyMMddhhmmss*にデータの日時（年月日時分秒）を協定世界時で設定したものとする。

Z_C_RJTD_*yyyyMMddhhmmss*_RDR_JMAGPV_Gll2p5km_Phhlv_ANAL_grib2.bin

最初のZとCの間には半角のアンダースコアーを2個入れる点に注意。他のアンダースコアーは半角1個である。

例えば、日本標準時で2003年5月14日8:20:00のデータなら
Z_C_RJTD_20030513232000_RDR_JMAGPV_Gll2p5km_Phhlv_ANAL_grib2.bin

2.5kmメッシュ気象庁レーダー頂高度全国合成に用いるGRIB2フォーマットおよびテンプレートの詳細

| 節番号 | 節の名称・該当テンプレート | オクテット（バイトと同じ） | 内容 | 表 | 値 | 備考 |
|---|---|---|---|---|---|---|
| 第0節 | 指示節 | 1～4 | GRIB | | GRIB | アスキーコードで設定する |
| | | 5～6 | 保留 | | missing | |
| | | 7 | 資料分野 | 符号表0．0 | 0 | 気象分野 |
| | | 8 | GRIB版番号 | | 2 | |
| | | 9～16 | GRIB報全体の長さ | | ****** | 第0節から第8節までのトータルのバイト数 |
| 第1節 | 識別節 | 1～4 | 節の長さ | | 21 | |
| | | 5 | 節番号 | | 1 | |
| | | 6～7 | 作成中枢の識別 | 共通符号表C-1 | 34 | 東京 |
| | | 8～9 | 作成副中枢 | | 0 | |
| | | 10 | GRIBマスター表バージョン番号 | 符号表1．0 | 2 | マスター表バージョン2 |
| | | 11 | GRIB地域表バージョン番号 | 符号表1．1 | 1 | 地域表バージョン1 |
| | | 12 | 参照時刻の意味 | 符号表1．2 | 0 | 解析 |
| | | 13～14 | 資料の参照時刻（年） | | ※1 | 協定世界時 |
| | | 15 | 資料の参照時刻（月） | | ※1 | 協定世界時 |
| | | 16 | 資料の参照時刻（日） | | ※1 | 協定世界時 |
| | | 17 | 資料の参照時刻（時） | | ※1 | 協定世界時 |
| | | 18 | 資料の参照時刻（分） | | ※1 | 協定世界時 |
| | | 19 | 資料の参照時刻（秒） | | ※1 | 協定世界時 |
| | | 20 | 作成ステータス | 符号表1．3 | 0 | 現業プロダクト |
| | | 21 | 資料の種類 | 符号表1．4 | 0 | 解析プロダクト |
| 第2節 | 地域使用節 | 不使用 | | | | |
| 第3節 | 格子系定義節 | 1～4 | 節の長さ | | 72 | |
| | | 5 | 節番号 | | 3 | |
| | | 6 | 格子系定義の出典 | 符号表3．0 | 0 | 符号表3．1参照 |
| | | 7～10 | 資料点数 | | 1146880 | 1024*1120 |
| | | 11 | 格子点数を定義するリストのオクテット数 | | 0 | |
| | | 12 | 格子点数を定義するリストの説明 | | 0 | |
| | | 13～14 | 格子系定義テンプレート番号 | 符号表3．1 | 0 | 緯度・経度格子 |
| | | 15 | 地球の形状 | 符号表3．2 | 4 | GRS80回転楕円体 |
| | | 16 | 地球球体の半径の尺度因子 | | missing | |
| | | 17～20 | 地球球体の尺度付き半径 | | missing | |
| | | 21 | 地球回転楕円体の長軸の尺度因子 | | 1 | |
| | | 22～25 | 地球回転楕円体の長軸の尺度付きの長さ | | 63781370 | |
| | | 26 | 地球回転楕円体の短軸の尺度因子 | | 1 | |
| | | 27～30 | 地球回転楕円体の短軸の尺度付きの長さ | | 63567523 | |
| | | 31～34 | 緯線に沿った格子点数 | | 1024 | |
| | | 35～38 | 経線に沿った格子点数 | | 1120 | |
| | | 39～42 | 原作成領域の基本角 | | 0 | |
| | | 43～46 | 端点の経度及び緯度並びに方向増分の定義に使われる基本角の細分 | | missing | |
| | | 47～50 | 最初の格子点の緯度 | 10**-6度単位 | 47987500 | 48N-1.5/60/2 |
| | | 51～54 | 最初の格子点の経度 | 10**-6度単位 | 118015625 | 118E+1.875/60/2 |
| | | 55 | 分解能及び成分フラグ | フラグ表3．3 | 48 | 0x30 |
| | | 56～59 | 最後の格子点の緯度 | 10**-6度単位 | 20012500 | 20N+1.5/60/2 |
| | | 60～63 | 最後の格子点の経度 | 10**-6度単位 | 149984375 | 150E-1.875/60/2 |
| | | 64～67 | i方向の増分 | 10**-6度単位 | 31250 | 1.875/60 |
| | | 68～71 | j方向の増分 | 10**-6度単位 | 25000 | 1.5/60 |
| | | 72 | 走査モード | フラグ表3．4 | 0 | |
| 第4節 | プロダクト定義 | 1～4 | 節の長さ | | 82 | |
| | | 5 | 節番号 | | 4 | |
| | | 6～7 | テンプレート直後の座標値の数 | | 0 | |
| | | 8～9 | プロダクト定義テンプレート番号 | 符号表4．0 | 50008 | 解析雨量と同じ |
| | ここからテンプレート | | | | | |
| | 4.50008 | 10 | パラメータカテゴリー | 符号表4．1 | 15 | レーダー |
| | ↓ | 11 | パラメータ番号 | 符号表4．2 | 192 | エコー頂高度レベル値 ※3 |
| | ↓ | 12 | 作成処理の種類 | 符号表4．3 | 0 | 解析 |
| | ↓ | 13 | 背景作成処理識別符 | JMA定義 | 201 | 全国気象庁レーダー合成 |
| | ↓ | 14 | 予報の作成処理識別符 | | missing | |
| | ↓ | 15～16 | 観測資料の参照時刻からの締切時間（時） | | 0 | |
| | ↓ | 17 | 観測資料の参照時刻からの締切時間（分） | | 5 | |
| | ↓ | 18 | 期間の単位の指示符 | 符号表4．4 | 0 | 分 |
| | ↓ | 19～22 | 予報時間 | | -10 | 0x8000000Aを設定する |
| | ↓ | 23 | 第一固定面の種類 | 符号表4．5 | 1 | 地面又は水面 |
| | ↓ | 24 | 第一固定面の尺度因子 | | missing | |
| | ↓ | 25～28 | 第一固定面の尺度付きの値 | | missing | |
| | ↓ | 29 | 第二固定面の種類 | 符号表4．5 | missing | |
| | ↓ | 30 | 第二固定面の尺度因子 | | missing | |
| | ↓ | 31～34 | 第二固定面の尺度付きの値 | | missing | |
| | ↓ | 35～36 | 全時間間隔の終了時（年） | | ※1 | 協定世界時 |
| | ↓ | 37 | 全時間間隔の終了時（月） | | ※1 | 協定世界時 |
| | ↓ | 38 | 全時間間隔の終了時（日） | | ※1 | 協定世界時 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | ↓ | 39 | 全時間間隔の終了時(時) | | ※1 | 協定世界時 |
| | | ↓ | 40 | 全時間間隔の終了時(分) | | ※1 | 協定世界時 |
| | | ↓ | 41 | 全時間間隔の終了時(秒) | | ※1 | 協定世界時 |
| | | | 42 | 統計を算出するために使用した | | 1 | |
| | | ↓ | | 時間間隔を記述する期間の仕様の数 | | | |
| | | ↓ | 43～46 | 統計処理における欠測資料の総数 | | 0 | |
| | | ↓ | 47 | 統計処理の種類 | 符号表4．10 | 1 | 積算 |
| | | | 48 | 統計処理の時間増分の種類 | 符号表4．11 | 2 | 同じ予報開始時刻を持ち、 |
| | | ↓ | | | | | 予報時間に増分が加えられる |
| | | ↓ | 49 | 統計処理の時間の単位の指示符 | 符号表4．4 | 0 | 分 |
| | | ↓ | 50～53 | 統計処理した期間の長さ | | 10 | |
| | | | 54 | 連続的な資料場間の増分に関する | | 0 | |
| | | ↓ | | 時間の単位の指示符 | | | |
| | | ↓ | 55～58 | 連続的な資料場間の時間の増分 | | 0 | 連続的な処理の結果 |
| | | ↓ | 59～66 | レーダー等運用情報その1 | | ※2 | |
| | | ↓ | 67～74 | レーダー等運用情報その2 | | missing | |
| | | ここまで | | | | | |
| | | ﾃﾝﾌﾟﾚｰﾄ | | | | | |
| | | 4.50008 | 75～82 | 雨量計運用情報 | | missing | |
| | 第5節 | 資料表現節 | 1～4 | 節の長さ | | 35 | |
| | | | 5 | 節番号 | | 5 | |
| | | | 6～9 | 全資料点の数 | | 1146880 | 1024*1120 |
| | | | 10～11 | 資料表現テンプレート番号 | 符号表5．0 | 200 | 格子点資料－ランレングス圧縮 |
| | | ここから | | | | | |
| | | ﾃﾝﾌﾟﾚｰﾄ5.200 | 12 | 1データのビット数 | | 8 | |
| | | | 13～14 | 今回の圧縮に用いたレベルの最大値 | | V | Vは実際のデータ中に現れた最大のレベル |
| | | ↓ | | | | | 値(＜＝M) |
| | | ↓ | 15～16 | レベルの最大値 | | M | M=9 |
| | | ↓ | 17 | データ代表値の尺度因子 | | 1 | 10**1の意味 |
| | | | | | | | 各レベルnnに対する※3のエコー頂高度の |
| | | ここまで | 16＋2xnn～ | | | | 値を10倍した値を設定。nn=1のときは0とす |
| | | ﾃﾝﾌﾟﾚｰﾄ5.200 | 17＋2xnn | レベルnnに対応するデータ代表値 | | ※3 | る。(nn＝1～M) |
| | 第6節 | ビットマップ節 | 1～4 | 節の長さ | | 6 | |
| | | | 5 | 節番号 | | 6 | |
| | | | 6 | ビットマップ指示符 | | 255 | ビットマップを適応せず |
| | 第7節 | 資料節 | 1～4 | 節の長さ | | ****** | 第7節のトータルのバイト数 |
| | | | 5 | 節番号 | | 7 | |
| | | ﾃﾝﾌﾟﾚｰﾄ7.200 | 6～ | ランレングス圧縮オクテット列 | | | 資料テンプレート7．200で記述された形式 |
| | 第8節 | 終端節 | 1～4 | 7777 | | 7777 | アスキーコードで設定する |

(注)

第0節最初の「GRIB」と第8節の「7777」のみアスキーコードで設定し、他は全て整数型のバイナリーで設定する。
バイナリー値は、ビッグエンディアンで設定する。
値欄が「missing」の場合そのデータは全ビット1の値、英数字の変数名や「******」は必要な値を設定する。
実際のデータは、ランレングス圧縮後第7節の6バイト目以降に設定する。
※1 第1節と第4節には、共に観測時刻を協定世界時で格納する。年月日時分秒で使用する数値は、
年:4桁の西暦年、月:1-12、日:1-31、時:0-23、分:0-59、秒:0-59 とする。

## ※2 レーダー等運用情報の詳細

### レーダー等運用情報その1

＜8バイト中の配置＞　（■は値を設定する2ビットを示し、□は0を設定する2ビットを示す）

□□□□□□□□□□■■■■■■■■■■■■■■■■■■■■■■■■■■

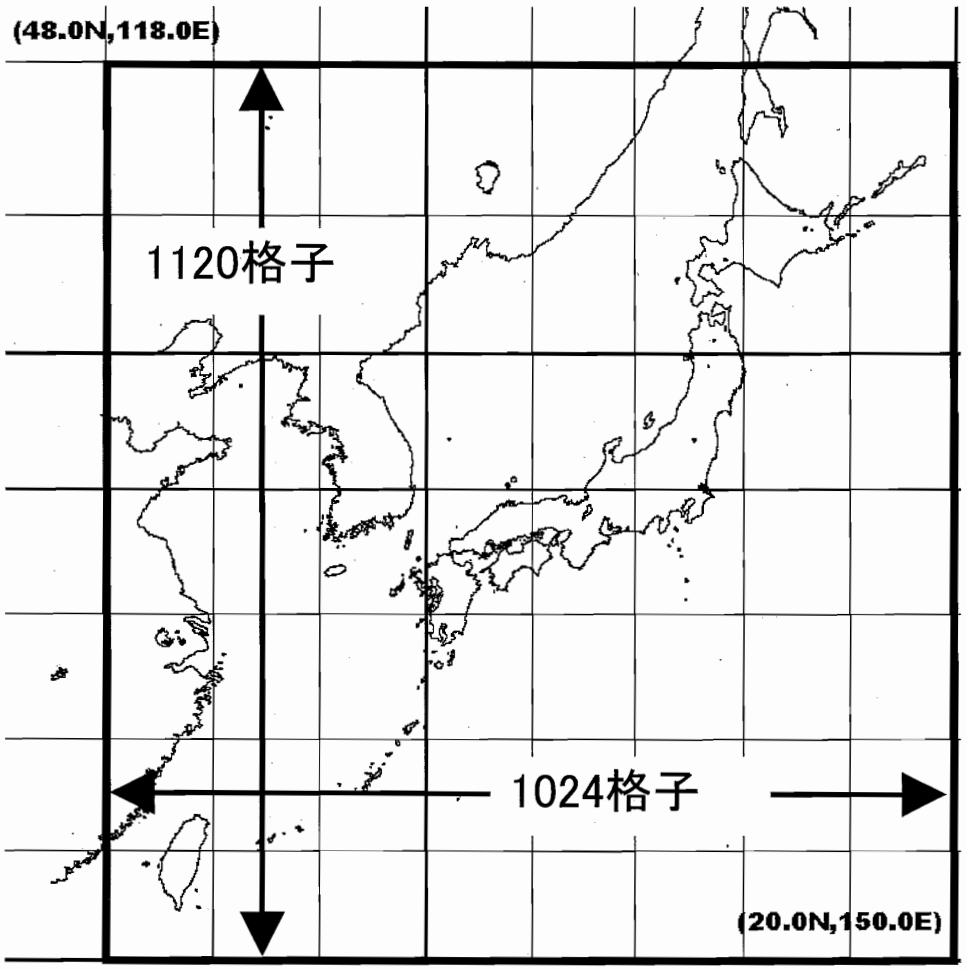
| 64 | 60 | 56 | 52 | 48 | 44 | 40 | 36 | 32 | 28 | 24 | 20 | 16 | 12 | 8 | 4 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | 沖縄SSP | 名瀬SP | 石垣島 | 沖縄 | 名瀬 | 種子島 | 福岡 | 室戸岬 | 広島 | 松江 | 大阪 | 名古屋 | 福井 | 静岡 | 長野 | 東京 | 新潟 | 秋田 | 仙台 | 函館 | 釧路 | 札幌 |

＜各レーダーの運用情報2ビットの内容＞

0 . . . . 電文受信なし
1 . . . . 電文受信あり(エコーあり)
2 . . . . 電文受信あり(No Echo)
3 . . . . 電文受信あり(No Operation)

※3　2.5kmメッシュ気象庁レーダー頂高度全国合成のレベル値(0～9)

| レベル値 | 意味 | データ代表値 |
|---|---|---|
| 0 | 観測範囲外 又は 欠測 | |
| 1 | No Echo | 0 |
| 2 | 2km未満 | 1 |
| 3 | 2km以上　4km未満 | 3 |
| 4 | 4km以上　6km未満 | 5 |
| 5 | 6km以上　8km未満 | 7 |
| 6 | 8km以上　10km未満 | 9 |
| 7 | 10km以上　12km未満 | 11 |
| 8 | 12km以上　14km未満 | 13 |
| 9 | 14km以上 | 15 |


(48.0N,118.0E)
1120格子
1024格子
(20.0N,150.0E)

# 全国合成レーダーGPV（GRIB2）の解読処理について

1kmメッシュ全国合成レーダーGPV を解読（デコード）処理するサンプルプログラムを提供します。

このサンプルプログラムの全部又は一部を利用することに問題はありませんが、利用したことによって利用者が被った直接的または間接的ないかなる損害についても、気象庁は一切責任を負いません。また、サンプルプログラムに関する個別の対応は行いかねますので、ご容赦願います。

## [ 利用方法 ]

1. 「make」コマンドによりコンパイルすることで実行ファイル「grib2_dec」が作成される。
   - ANSI 準拠の c コンパイラでコンパイルできます。ただし、拡張機能 unsigned long long を利用しています。UNIX（HP-UX，AIX）及び Linux（RedHat）での動作を確認しています。
   - リトルエンディアンマシンの場合、ヘッダーファイル「sample_decode.h」の 5 行目のコメントをはずしてください。

2. 次のコマンドを入力することにより、GRIB2各節の内容が端末に表示されると共に、xpm形式の画像イメージファイルが作成される。

   grib2_dec {データファイル名} -xpm

3. データを4バイト整数の配列で出力したい場合は、次のコマンドを実行する。

   grib2_dec {データファイル名}

Makefile

```
CC          = cc
CFLAGS = -O -Wall
MODULE              = grib2_dec
OBJS      = sample_grib2_dec.o rlencmp.o i2pix.o
HEADER   = sample_decode.h prr_template.h pmf_template.h

.c.o : $(HEADER)
        $(CC) $(CFLAGS) -c $< -o $@

$(MODULE) : $(OBJS)
        $(CC) $(OBJS) -o $(MODULE)

clean :
        rm $(OBJS) $(MODULE)
```

sample_decode.h

```
/* ************************************** */
/* sample grib2 decode program include file */
/* ************************************** */

/* #define IS_LITTLE_ENDIAN */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef struct sect {
     int num;
     int *len;
     unsigned char **v;
} ST_SECT;

int decode_rlen_nbit(void *udata, size_t utype, unsigned char *din,
                              int nin, int nout, int maxv, int nbit);
void i2pix_2(const int *fd, const ST_SECT ss[], FILE * fp);
```

pmf_template.h

```
/* ********************************************************************** */
/*   template for short range precipitation forecast                      */
/* ********************************************************************** */
char sc_pmf[][40]={
  "42118",
  "412211121111111",
  "411411211414144444444144441",
  "41221111121141141142111111411141488821n",
  "41421221n",
  "411",
  "41n",
  "4",
  ""
};
```

prr_template.h

```
/* ****************************************************** */
/*   template for analisys precipitation                  */
/* ****************************************************** */
char sc_prr[][40]={
  "42118",
```

```
    "412211121111111",
    "411411211414144444444144441",
    "41221111121141141142111111414111414888",
    "41421221n",
    "411",
    "41n",
    "4",
    ""
};
```

sample_grib2_dec.c

```
/* ---------------------------------------------------------------------- */
/*   Last Updated Ver.1.3      2004.04.14                                  */
/*          Copyright (C) 2003 Japan Meteorological Agency   All rights reserved */
/* ---------------------------------------------------------------------- */

#include "sample_decode.h"
#include "prr_template.h"
#include "pmf_template.h"

#ifdef IS_LITTLE_ENDIAN
#define Fread fread_little_endian
#else
#define Fread fread
#endif

/*----------------------------------------------------------------------*/
void fread_little_endian(void *d, int len, int num, FILE * fp)
{
    unsigned char uc[8], *ud;
    int i, j, k;

    ud = d;
    if (len == 1)
        fread(d, len, num, fp);
    else {
        for (i = 0, k = 0; i < num; i++, k += len) {
            fread(uc, len, 1, fp);
            for (j = 0; j < len; j++)
                *(ud + k + j) = uc[len - 1 - j];
        }
    }
}

/*----------------------------------------------------------------------*/
void init_sect(ST_SECT ss[], int af)
{
    int i, j, k;
    char c[2];

    if (af == 0) {
        k = 0;
        while (strlen(sc_prr[k]) != 0)
            k++;
        for (i = 0; i < k; i++) {
            ss[i].num = strlen(sc_prr[i]);
            ss[i].len = (int *) malloc(sizeof(int *) * ss[i].num);
            for (j = 0; j < ss[i].num; j++) {
                strncpy(c, &sc_prr[i][j], 1);
                *(ss[i].len + j) = atoi(c);
            }
            ss[i].v =
                (unsigned char **) malloc(sizeof(unsigned char *) *
                                          ss[i].num);
        }
```

```
55          } else {
56                  k = 0;
57                  while (strlen(sc_pmf[k]) != 0)
58                      k++;
59                  for (i = 0; i < k; i++) {
60                      ss[i].num = strlen(sc_pmf[i]);
61                      ss[i].len = (int *) malloc(sizeof(int *) * ss[i].num);
62                      for (j = 0; j < ss[i].num; j++) {
63                              strncpy(c, &sc_pmf[i][j], 1);
64                              *(ss[i].len + j) = atoi(c);
65                      }
66                      ss[i].v =
67                              (unsigned char **) malloc(sizeof(unsigned char *) *
68                                                                      ss[i].num);
69                      for (j = 0; j < ss[i].num; j++)
70                              *(ss[i].v + j) = NULL;
71                  }
72          }
73      }
74
75      /*--------------------------------------------------------------------*/
76      int read_sect(ST_SECT ss[], FILE * fp)
77      {
78          int i, nn, mm, si;
79          unsigned int slen;
80          unsigned short *us, ud;
81          unsigned char sn;
82
83          Fread(&slen, 4, 1, fp);
84          if (memcmp(&slen, "7777", 4) == 0) {
85                  si = 8;
86                  return (si);
87      #ifdef IS_LITTLE_ENDIAN
88          } else if (memcmp(&slen, "BIRG", 4) == 0) {
89      #else
90          } else if (memcmp(&slen, "GRIB", 4) == 0) {
91      #endif
92                  si = 0;
93                  slen = 16;
94                  Fread(&ud, 2, 1, fp);
95                  *(ss[si].v + 0) =
96                      (unsigned char *) realloc(*(ss[si].v + 0),
97                                                      sizeof(unsigned int));
98                  *(ss[si].v + 1) =
99                      (unsigned char *) realloc(*(ss[si].v + 1),
100                                                     sizeof(unsigned short));
101                 memcpy(*(ss[si].v + 0), &slen, 4);
102         } else {
103                 Fread(&sn, 1, 1, fp);
104                 si = (int) sn;
105                 if (si >= 3)
106                     si--;
107                 *(ss[si].v + 0) =
108                     (unsigned char *) realloc(*(ss[si].v + 0),
109                                                     sizeof(unsigned int));
110                 *(ss[si].v + 1) =
111                     (unsigned char *) realloc(*(ss[si].v + 1),
112                                                     sizeof(unsigned char));
113                 memcpy(*(ss[si].v + 0), &slen, 4);
114                 memcpy(*(ss[si].v + 1), &sn, 1);
115         }
116
117         for (i = 2; i < ss[si].num; i++) {
118                 if (*(ss[si].len + i) == 0) {
119                     us = (unsigned short *) (*(ss[si].v + i - 2));
120                     nn = (*(ss[si].len + i - 2) == 4) ? sizeof(unsigned char)
121                             : sizeof(unsigned short);
```

```
122                 mm = (*(ss[si].len + i - 2) == 4) ? slen - 5 : *us;
123             } else {
124                 nn = *(ss[si].len + i);
125                 mm = 1;
126             }
127             *(ss[si].v + i) =
128                 (unsigned char *) realloc(*(ss[si].v + i), (size_t) nn * mm);
129             Fread(*(ss[si].v + i), nn, mm, fp);
130         }
131
132         return (si);
133     }
134
135     /*---------------------------------------------------------------------*/
136     int dec_data(ST_SECT ss[], int **lv)
137     {
138         int nin, nout, maxv, nbit, rt;
139         unsigned int *ui;
140         unsigned short *us;
141
142         ui = (unsigned int *) *(ss[4].v + 2);
143         nout = *ui;
144         nbit = **(ss[4].v + 4);
145         us = (unsigned short *) *(ss[4].v + 5);
146         maxv = *us;
147         ui = (unsigned int *) *(ss[6].v + 0);
148         nin = *ui - 5;
149
150         *lv = (int *) malloc(sizeof(int) * nout);
151         rt = decode_rlen_nbit(*lv, sizeof(int), *(ss[6].v + 2), nin, nout,
152                                         maxv, nbit);
153
154         return (rt);
155     }
156
157     /*---------------------------------------------------------------------*/
158     void print_info(ST_SECT ss[], int sn)
159     {
160         int i, j, k;
161         unsigned long long *ull;
162         int *ii;
163         unsigned short *us, *n;
164
165         printf("========== SECTION %1.1d ==========¥n",
166                 (sn >= 2) ? sn + 1 : sn);
167         for (i = 0, j = 1; i < ss[sn].num; j += *(ss[sn].len + i), i++) {
168             if (*(ss[sn].len + i) == 1) {
169                 if (**(ss[sn].v + i) == 0xff)
170                     printf("      %3d      : 0x%2.2x¥n", j, **(ss[sn].v + i));
171                 else
172                     printf("      %3d      : %d¥n", j, **(ss[sn].v + i));
173             } else if (*(ss[sn].len + i) == 8) {
174                 if (sn == 0) {
175                     ull = (unsigned long long *) *(ss[sn].v + i);
176                     printf("%4d --%4d: %d¥n", j, j + *(ss[sn].len + i) - 1,
177                             (unsigned int) *ull);
178                 } else {
179                     printf("%4d --%4d: ", j, j + *(ss[sn].len + i) - 1);
180                     for (k = 0; k < 8; k++)
181                         printf("%2.2x", *(*(ss[sn].v + i) + k));
182                     printf("¥n");
183                 }
184             } else if (*(ss[sn].len + i) == 4) {
185                 ii = (int *) *(ss[sn].v + i);
186                 if (*ii >= 0 || (i == 12 && sn == 3))  /* only fcst_time is signed int */
187                     printf("%4d --%4d: %d¥n", j, j + *(ss[sn].len + i) - 1,
188                             *ii);
```

```
                else
                        printf("%4d --%4d: 0x%8.8x¥n", j,
                                j + *(ss[sn].len + i) - 1, *ii);
        } else if (*(ss[sn].len + i) == 2) {
                us = (unsigned short *) *(ss[sn].v + i);
                if (*us == 0xffff)
                        printf("%4d --%4d: 0x%4.4x¥n", j,
                                j + *(ss[sn].len + i) - 1, *us);
                else
                        printf("%4d --%4d: %d¥n", j, j + *(ss[sn].len + i) - 1,
                                *us);
        } else if (sn == 3 || sn == 4) {
                n = (unsigned short *) *(ss[sn].v + i - 2);
                us = (unsigned short *) *(ss[sn].v + i);
                for (k = 0; k < *n; k++) {
                        if (*(us + k) == 0xffff)
                            printf("%4d --%4d: 0x%4.4x¥n", j + 2 * k,
                                    j + 2 * k + 1, *(us + k));
                        else
                            printf("%4d --%4d: %d¥n", j + 2 * k, j + 2 * k + 1,
                                    *(us + k));
                }
        }
    }
    fflush(stdout);
}

/*---------------------------------------------------------------------*/
int main(int argc, char *argv[])
{
    ST_SECT ss[8];
    FILE *fp, *fpo;
    char fname[160], gname[160], suffix[160], fcs[160], ffm[160];
    int sn, *lv, gn, sff = 0, fcnt = 0, af, ll;

    if (argc == 1) {
        fprintf(stderr, "¥n¥nusage: grib2_dec ***_grib2.bin (-xpm)¥n¥n");
        fprintf(stderr,
                "    This program decodes the grib2 file named ***_grib2.bin, and prints the¥n
value of each section in GRIB2.  Also, this program puts out a raw (4 byte¥n  integer) data file
***_int.bin as a rectangle grid dimension.  In case of¥n  specifying -xpm options, an output file is to a
picture image file ***.xpm¥n  formatted as X-pixmap.¥n¥n");
        exit(1);
    } else if (argc == 3 && strcmp(argv[2], "-xpm") == 0) {
        strcpy(suffix, ".xpm");
        sff = 1;
    } else
        strcpy(suffix, ".bin");

    strcpy(fname, argv[1]);
    if ((fp = fopen(fname, "r")) == NULL) {
        fprintf(stderr, "grib2 file <%s> open error!!¥n", fname);
        exit(1);
    }
    af = (strstr(fname, "_ANAL") == NULL && strstr(fname, "_NOWC") == NULL ) ? 1 : 0;

    init_sect(ss, af);


    while ((sn = read_sect(ss, fp)) != 8) {
        if (sn == 6) {
            print_info(ss, sn);
            gn = dec_data(ss, &lv);
            if (gn > 0) {
                ll = strlen(fname) - strlen(strstr(fname, "_grib2.bin"));
                strncpy(gname, fname, ll);
                gname[ll] = '¥0';
```

```
256                         sprintf(fcs, "_%1d", fcnt);
257                         strcpy(ffm, (sff == 1) ? "" : "_int");
258                         strcat(gname, strcat(fcs, strcat(ffm, suffix)));
259                         if ((fpo = fopen(gname, "w")) == NULL) {
260                             fprintf(stderr, "output file <%s> open error!!¥n",
261                                     gname);
262                             exit(1);
263                         }
264                         if (sff == 0)
265                             fwrite(lv, sizeof(int), gn, fpo);
266                         else {
267                             i2pix_2(lv, ss, fpo);
268                         }
269                         fclose(fpo);
270                         fcnt++;
271                     }
272                     free(lv);
273                 } else {
274                     print_info(ss, sn);
275                 }
276             }
277             fclose(fp);
278             return 0;
279         }
```

rlencmp.c

```
1   /* ---------------------------------------------------------------------- */
2   /*  Last Updated Ver.1.3      2004.04.14                                  */
3   /*          Copyright (C) 2003 Japan Meteorological Agency  All rights reserved */
4   /* ---------------------------------------------------------------------- */
5
6   #include "sample_decode.h"
7
8   int ipow(const int i, const int j)
9   {
10      int k, l;
11      for (k = 0, l = 1; k < j; k++)
12          l *= i;
13      return (l);
14  }
15
16  void endian_conv4(void *idat)
17  {
18      unsigned char *uc, c;
19
20      uc = (unsigned char *) idat;
21      c = *(uc + 0);
22      *(uc + 0) = *(uc + 3);
23      *(uc + 3) = c;
24      c = *(uc + 1);
25      *(uc + 1) = *(uc + 2);
26      *(uc + 2) = c;
27
28  }
29
30  int nbit_unpack(unsigned char din[], int nin, unsigned int dout[],
31                  int nout, int nbit)
32  {
33
34      unsigned int wi, n_b2s;
35      int i, bitp, dp;
36
37      n_b2s = ipow(2, nbit) - 1;
38
39      i = 0;
```

```
        dp = 0;
        bitp = 0;
        while (dp < nin) {
                bitp += nbit;
                memcpy(&wi, &din[dp], 4);
#ifdef IS_LITTLE_ENDIAN
                dout[i] = wi & n_b2s;
#else
                dout[i] = (wi >> (32 - bitp)) & n_b2s;
#endif
                while (bitp >= 8) {
                    dp++;
                    bitp -= 8;
                }
                if (++i > nout)
                    return (-1);
        }

        return (i);
}

/* ------------------------------------------------------------------------ */
int decode_rlen_nbit(void *udata, size_t utype, unsigned char *din,
                                int nin, int nout, int maxv, int nbit)
/* ------------------------------------------------------------------------ */
/* decode ranlength compress ( nbit data version )                          */
/*    output:    udata   = user data for put                                */
/*    input:     utype   = user data type :sizeof(int or short or unsigned char) */
/*               din     = compressed data                                  */
/*               nin     = compressed data size (byte)                      */
/*               nout    = number of grid point                             */
/*               maxv    = maximum value of user data                       */
/*               nbit    = number of bit used for a compressed data         */
/*    return:    >=0     = number of decoded data                           */
/*               -4      = uncompressed data size exceeds nout              */
/*               -6      = first user data is out of the data range         */
/* ------------------------------------------------------------------------ */
{
        int i, j, k, l, m, n, v, p, cf = 0, *doi, ninb;
        short *dos;
        unsigned char *doc;
        unsigned int *wd, *ww;

        doi = (int *) udata;
        dos = (short *) udata;
        doc = (unsigned char *) udata;
        wd = (unsigned int *) malloc(sizeof(unsigned int) * nout);
        ww = (unsigned int *) malloc(sizeof(unsigned int) * nout);
        ninb = nbit_unpack(din, nin, wd, nout, nbit);
        if (ninb < 0)
                return (-4);
        l = ipow(2, nbit) - 1 - maxv;
        v = (int) (*wd);
        if (v < 0 || v > maxv)
                return (-6);

        i = 0;
        k = 0;
        p = -1;
        m = 1;
        n = 0;
        while (i < ninb) {
                v = (int) (*(wd + i++));
                if (v <= maxv) {
                    if (p >= 0) {
                            for (j = 0; j < m; j++) {
                                if (k == nout) {
```

```
107                                     cf = 1;
108                                     break;
109                                 }
110                                 *(ww + k++) = p;
111                             }
112                             if (cf == 1)
113                                 break;
114                         }
115                         p = v;
116                         m = 1;
117                         n = 0;
118                     } else {
119                         m += ipow(l, n++) * (v - maxv - 1);
120                     }
121                 }
122                 for (j = 0; j < m; j++) {
123                     if (k == nout)
124                         break;
125                     *(ww + k++) = p;
126                 }
127                 switch (utype) {
128                 case 1:
129                     for (j = 0; j < k; j++)
130                         *(doc + j) = (unsigned char) *(ww + j);
131                     break;
132                 case 2:
133                     for (j = 0; j < k; j++)
134                         *(dos + j) = (short) *(ww + j);
135                     break;
136                 case 4:
137                     for (j = 0; j < k; j++)
138                         *(doi + j) = (int) *(ww + j);
139                     break;
140                 }
141
142                 free(wd);
143                 free(ww);
144                 return (k);
145             }
```

i2pix.c

```
1   /* ---------------------------------------------------------------------- */
2   /*  Last Updated Ver.1.3     2004.04.14                                   */
3   /*          Copyright (C) 2003 Japan Meteorological Agency   All rights reserved */
4   /* ---------------------------------------------------------------------- */
5
6   #include "sample_decode.h"
7
8   #define COLOR_IE_NUM 16
9   #define COLOR_HE_NUM 10
10
11  static const unsigned char color_IE[COLOR_IE_NUM][3]=
12  {
13      {192,192,192},
14      {255, 255, 255},
15      {0,255,255},
16      {0,119,198},
17      {0,60,108},
18      {0,6,240},
19      {0,147,117},
20      {0,179,71},
21      {0,225,12},
22      {70,255,9},
23      {124,206,2},
24      {255,255,0},
```

```
        {255,128,0},
        {255,134,255},
        {254,69,162},
        {255,0,0},
};

static const unsigned char color_HE[COLOR_HE_NUM][3]=
{
        {192,192,192},
        {255,255,255},
        {0,255,255},
        {0,0,255},
        {0,147,17},
        {127,255,0},
        {255,255,0},
        {255,127,0},
        {255,0,255},
        {255,0,0}
};

static const int rank_IE[COLOR_IE_NUM - 1]=
{-1, 0, 1, 2, 4, 8, 12, 16,
 24, 32, 40, 48, 56, 64, 80};

static const int rank_HE[COLOR_HE_NUM - 1]=
{-1, 0, 2, 4, 6, 8, 10, 12, 14};



static int get_level(unsigned char level, const short ispc[],
                        short max_level,
                        const int rank[], int rank_num)
{
        short rain;
        int i;

        if(level > max_level){
                return -1;
        }

        rain = ispc[level];

        if(rain < 0){
                return 0;
        }
        else if(rain == 0){
                return 1;
        }

        for(i = 2; i < rank_num; i++){
                if(rain < rank[i]){
                        return i ;
                }
        }
        return rank_num;

}


void i2pix_2(const int *fd,          /* orignal data array      */
                const ST_SECT ss[],
                  FILE * fp                     /* output file pointer     */
        )
{
        int i, j, k;
        int r, g, b;
        char *line, *p;
```

```
92          unsigned char parm_num;         /* parameter number
93                                           echo intensity: 201, echo top: 192 */
94          unsigned char bg_id;           /* Background generating process identifier
95                                            JMA:201, ITGRAD: 200*/
96          int max_level, rank_num;
97          const int *org_rank;
98          int *rank;
99          unsigned char color[COLOR_IE_NUM][3];
100         short *ispc;
101         unsigned char scale_index;
102         int scale;
103
104         int cl;
105         int ix, iy;
106
107         short *stmp;
108         int *itmp;
109
110         parm_num = *(*(ss[3].v + 5));
111         bg_id = *(*(ss[3].v + 7));
112
113         itmp = (int *) *(ss[2].v + 14);
114         ix = *itmp;
115
116         itmp = (int *) *(ss[2].v + 15);
117         iy = *itmp;
118
119         if(parm_num == 192 && bg_id == 201){ /* echo top */
120             rank_num = COLOR_HE_NUM - 1;
121             memcpy(color, color_HE, (rank_num + 1) * 3);
122             org_rank = rank_HE;
123         }
124         else{
125             rank_num = COLOR_IE_NUM - 1;
126             memcpy(color, color_IE, (rank_num + 1) * 3);
127             org_rank = rank_IE;
128         }
129
130         stmp = (short*)*(ss[4].v + 6);
131         max_level = (short)*stmp;
132         if((ispc = (short*)malloc(sizeof(short) * (max_level + 1)))==NULL){
133             fprintf(stderr, "malloc error!¥n");
134             exit(-1);
135         }
136
137         stmp = (short*)*(ss[4].v + 8);
138         ispc[0] = -1;
139         memcpy(ispc + 1, stmp, max_level * sizeof(short));
140         scale_index = *(*(ss[4].v + 7));
141         scale = 1;
142         while(scale_index != 0){
143             scale *= 10;
144             scale_index--;
145         }
146         if((rank = (int*)malloc((rank_num + 1) * sizeof(int))) == NULL){
147             fprintf(stderr, "malloc error!¥n");
148             exit(-1);
149         }
150         for(i = 1; i < rank_num; i++){
151             rank[i] = org_rank[i] * scale;
152         }
153
154         line = (char *) malloc(sizeof(char) * ix + 1);
155         fprintf(fp, "static char *no_xpm[] = {¥n");
156         fprintf(fp, "¥"%d %d %d 1¥",¥n", ix, iy, rank_num + 2);
157         fprintf(fp, "¥"# c #000000¥",¥n");
158         for(i = 0; i < rank_num + 1; i++){
```

```
159             r = color[i][0];
160             g = color[i][1];
161             b = color[i][2];
162               fprintf(fp, "¥"%c c #%02X%02X%02X¥",¥n", i + '$', r, g, b);
163         }
164
165         for (j = 0; j < iy - iy / 50; j++) {
166               p = line;
167               for (i = 0; i < ix; i++) {
168                     k = i + j * ix;
169                 cl = get_level(fd[k], ispc, max_level, rank, rank_num);
170                 if(cl < 0){
171                       cl = 0;
172                 }
173                 *p++ = (char) ((int) '$' + cl);
174               }
175               *p = '¥0';
176               fprintf(fp, "¥"%s¥",¥n", line);
177         }
178
179     /* color bar */
180         for (j = iy - iy / 50; j < iy; j++) {
181               p = line;
182               for (i = 0; i < ix; i++){
183                     *p++ = (char) ((int) '$' + i * (rank_num + 1) / ix);
184             }
185               *p = '¥0';
186               fprintf(fp, "¥"%s¥",¥n", line);
187         }
188         fprintf(fp, "};¥n");
189         free(rank);
190         free(ispc);
191         free(line);
192         return;
193     }
```