

SEMI P39-0308

OASIS™ – オープン・アートワーク・システム・インターチェンジ・スタンダード (OPEN ARTWORK SYSTEM INTERCHANGE STANDARD)

本スタンドは、global Micropatterning Committee で技術的に承認されている。現版は 2008 年 1 月 18 日、global Audits and Reviews Subcommittee にて発行が承認された。2008 年 2 月に www.semi.org で、そして 2008 年 3 月に CD-ROM で入手可能となる。初版は 2004 年 3 月発行、前版は 2005 年 11 月に発行された。

1 目的

1.1 本仕様の目的は、階層構造を有する集積回路のマスクのレイアウト情報のための相互交換およびカプセル化のフォーマットを定義することである。

1.2 背景 — 2001 年の秋、SEMI の Data Path Task Force は古くからある GDSII ストリームフォーマットにとってかわるフォーマットを定義するためのワーキンググループを作った。GDSII ストリームフォーマットは、レイアウトを相互交換するための IC 業界のデファクトスタンダードとして 20 年以上も機能してきた。この古いフォーマットは、16 ビットと 32 ビットの内部整数フィールドに制限され、セルの幾何学図形の表現が不十分であること、および構造的オーバーヘッドが大きいことにより、最先端の設計に使用することが困難になってきており、ファイルサイズも場合によっては数十ギガバイトになるなど膨大なサイズになった。後継のフォーマットでは、次のような全体的な目標を与えた。

- GDSII ストリームよりも、ファイルサイズを少なくとも 1 桁小さくするように改善すること。
- 16 ビットと 32 ビットの整数幅のすべての制限を取り除き、完全な 64 ビットを可能にさせる新しいフォーマットを作成すること。
- フラットな幾何学図形を多数含むセルを効率的に表現すること。
- 設計と製造の間でレイアウト関連の情報の交換を容易にするために、より豊富な情報パレットを提供すること。

1.2.1 SEMI Data Path Task Force の形成に至る数ヶ月間、International Sematech は、マスク EDA の課題に焦点を当てた一連の会議を主催した。タスクフォースの多くの参加者は、これらの Sematech の会議にも関わり、これらのセッションからはるかに有用な情報により、この仕様の定義を進展させた。

2 範囲

2.1 本フォーマットは、主として EDA ソフトウェア、マスク描画装置、マスク検査／修正装置等のシステム間でやりとりする階層構造のマスクレイアウトを、カプセル化するために設計されている。

2.2 本フォーマットは、ハードウェアとソフトウェアのどちらにも依存しないように設計されている。

注意: このスタンドはその使用に関連したすべての安全問題を取り扱うことを意図していない。このスタンドの使用者は、その責任において、適切な安全および健康上実施すべき事柄を確立し、また使用前に法規制やその他の制限への適用性を判断するものである。

3 制限

3.1 XNAME, XELEMENT, XGEOMETRY 等の拡張レコードを使用すると、装置間の相互運用性を損ねる可能性がある。これらの拡張は主としてプロトタイピングに使用し、この仕様に拡張を正式に含めることによって相互運用性を維持するように推奨する。

4 参照スタンダードおよび文書

4.1 IEEEスタンダード¹

IEEE 754-1985 — IEEE Standard for Binary Floating-Point Arithmetic

4.2 ISOスタンダード²

ISO 646-IRV — “US-ASCII” Character Set

ISO 3309 — Information technology – Telecommunications and information exchange between systems – High-level data link control (HDLC) procedures – Frame structure

4.3 IETFスタンダード³

RFC 1951 — DEFLATE Compressed Data Format Specification Version 1.3

注意：別途指示のない場合、引用されたすべての文書は最新の出版バージョンである。

5 用語

5.1 略語と頭字語

5.1.1 BNF (Backus-Naur Form) — バッカス・ナウアー記法

5.1.2 EDA (Electronic Design Automation) — コンピュータによる設計の自動化

5.1.3 OASISTM (Open Artwork System Interchange Standard) — オープンアートワークシステムインターチェンジスタンダード⁴

5.2 定義

5.2.1 OASISに固有の用語のほとんどの定義は、それらを含む段落内の文に記載されている。

5.2.2 セル (cell) — レアウトの階層構造内の名前付きオブジェクト。固有の図形情報、注釈情報、他のセルの配置を格納している。

5.2.3 図形 (geometry) — 多角形、四角形、台形、パス、円等の二次元の幾何学的図形であり、レイヤとデータタイプの固有属性を有する。

5.2.4 配置 (placement) — セルのコピーが別のセルの座標空間内に特定の位置、向き、スケールで配置されることを基準として指定すること。セルの配置は、OASISファイル内の階層構造を可能にする基本的なメカニズムである。

5.2.5 プロパティ (property) — 名前、オプションである値のリストからなる注釈エレメントであり、ファイルまたはそのコンポーネントの1つを説明する情報を提供する。

5.2.6 レコード (record) — OASISファイル内の主要なデータの区切り。

5.2.7 テキストエレメント (text element) — 座標点 (x,y) および関連する文字列からなる注釈エレメント。

5.3 記号

5.3.1 “->” — 引数のその内容または意味へのマッピングを示す。

1 Institute of Electrical and Electronics Engineers, IEEE Operations Center, 445 Hoes Lane, P.O. Box 1331, Piscataway, New Jersey 08855-1331, USA. Telephone: 732.981.0060; Fax: 732.981.1721; <http://www.ieee.org>

2 International Organization for Standardization, ISO Central Secretariat, 1 rue de Varembe, Case postale 56, CH-1211 Geneva 20, Switzerland. Telephone: 41.22.749.01.11; Fax: 41.22.733.34.30; <http://www.iso.ch>

3 Internet Engineering Task Force; <http://www.ietf.org>

4 所有者の許諾により使用される。

6 OASISの基本

6.1 OASISファイルは、バイトシーケンスであり、レコードに分割される。レコードの長さは、その構造から認識でき、明示的なものではない(すべてのレコード長が明示的であるGDSIIと対照的)。

6.2 OASISファイルには、次の全体的な構文がある(§ 36 で説明する修正BNF記法を使用)。個別のレコードタイプは、太字の大文字で表記され、次の各項で詳細に説明する。

```
<oasis-file> -> <magic-bytes> START { CBLOCK | PAD | PROPERTY | <cell> | <name> } * END
<name> -> { CELLNAME | TEXTSTRING | LAYERNAME | PROPNAME | PROPSTRING | XNAME }
<cell> -> { CELL { CBLOCK | PAD | PROPERTY | XYRELATIVE | XYABSOLUTE | <element> } * }
<element> -> { <geometry> | PLACEMENT | TEXT | XELEMENT }
<geometry> -> { RECTANGLE | POLYGON | PATH | TRAPEZOID | CTRAPEZOID | CIRCLE | XGEOMETRY }
```

6.3 OASISファイルは、完全なレイアウト階層構造、一部のレイアウト階層構造、複数のレイアウト階層構造を表現できる。これらの解釈は、フォーマット固有のものではなく、アプリケーションの動作によってのみ支配される。それぞれのOASISファイルは、構文的に完全でなければならない。つまり<magic-bytes>で始まり、少なくとも 1 つのSTARTレコードおよびENDレコードを含んでいなければならない。

6.4 <magic-bytes>エレメントは、13 個のASCII文字のシーケンス、"%SEMI-OASIS<CR><NL>"であり、<CR><NL>は、ASCIIの16進数シーケンスの0D 0Aを表現する。OASISファイルをUNIXのfileユーティリティによって容易に識別できるようにするための認識シグネチャとして提供されている。(復改と改行の意図は、バイナリではないモードで動作しているFTPプログラムによる破壊の検出を助けることである。)

6.5 例外の取り扱い— OASISプロセッサは、本文書に記載されている構文と差違がある場合は致命的エラーとして取り扱わなければならない。OASISリーダーは、本仕様に準拠していると見なすために構文チェック前処理を実装する必要はない。例外が検出、報告されたシーケンスは、完全にアプリケーションに依存する。さらに、ファイル全体を解釈する必要がないアクセス要求(ファイル内の1つのセルやセルのサブセットの取り出し等)については、本仕様では、OASISリーダーがファイル全体の有効性を時間をかけてチェックする必要はない。

7 データ構造

7.1 バイト

7.1.1 バイトは8ビットの固定長の値である。バイトのビットパターンは最下位ビット(ビット0)を右にして示す。

7.2 整数

7.2.1 符号なし整数は、Nバイト($N > 0$)の整数値である。OASISフォーマットでは、低次のバイトが先に表記される。整数のバイト長は可変であり、整数は、バイトの連続で表現され、チェーン内の最後のバイトを除き、各バイトの最上位ビットが1であり、各バイトの残りの7ビットが連結して実際の整数値自体を形成する。整数のバイト長に制限はない(従って、大きさに制限はない)。



図 1
符号なし整数の表現

表 1 符号なし整数の例

符号なし整数の値	バイナリ表現
0	00000000
127	01111111
128	10000000 00000001
16,383	11111111 01111111
16,384	10000000 10000000 00000001

7.2.2 **符号付き整数**は、符号なし整数と同じバイトの連続の仕組みに従い、符号付きの大きさのフォームに格納され、仮数部は 1 ビット左にシフトし、符号ビットは低位(最初の)バイトの最下位ビットに格納される。符号ビットが 0 であれば正数を示し、1 であれば負数を示す。両方のゼロの表現(+0 と-0)は、比較を目的として数値的に等価に取り扱わなければならない。

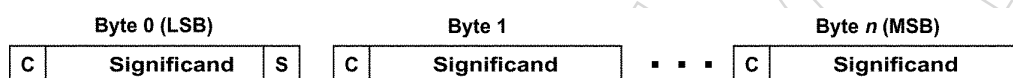


図 2
符号付き整数の表現

表 2 符号付き整数の例

符号付き整数の値	バイナリ表現
0	00000000
+1	00000010
-1	00000011
+63	01111110
-64	10000001 00000001
+8,191	11111110 01111111
-8,192	10000001 10000000 00000001

7.2.3 **例外的取り扱い** — 制限された空間(32 ビットの空間等)内の整数データのみをサポートするOASISプロセッサは、大きさがこの空間を超えると致命的エラーとして取り扱わなければならない。

7.3 実数

7.3.1 **実数**は、複数の有理数のフォームの 1 つあるいは単精度 4 バイト(ieee-4)または倍精度 8 バイト(ieee-8)の浮動小数点の値で格納できる。有理数のフォームは、通常は浮動小数点のフォームよりコンパクトであり、バイナリ浮動小数点表現によってのみ近似できる多数の値を正確に表現できるという利点がある。表現のタイプは、実数の仮数部の前に付ける符号なし整数で格納される。

表 3 実数のタイプ

形式	意味
'0' unsigned-integer	正の整数
'1' unsigned-integer	負の整数
'2' unsigned-integer	正の逆数
'3' unsigned-integer	負の逆数
'4' unsigned-integer unsigned-integer	正の分数
'5' unsigned-integer unsigned-integer	負の分数
'6' IEEE-4-byte-float	単精度浮動小数
'7' IEEE-8-byte-float	倍精度浮動小数

7.3.2 タイプ 0 と 1 では、実数は整数そのものであり、小数部はない。タイプ 2 と 3 では、符号なし整数の逆数の分母を表現し、暗黙の分子は 1 である。タイプ 4 と 5 は比であり、先に表記されるのが分子、後に表記されるのが分母である。タイプ 6 と 7 は、IEEE 754-1985 フォーマットのバイナリ浮動小数点数であり、仮数部の最下位のバイト(バイト 0)が最初に格納される。

7.3.3 例外の取り扱い — タイプ 2 ～ 5 については、分母がゼロの場合、致命的エラーとして取り扱わなければならない。タイプが 0 ～ 7 の範囲外の場合、致命的エラーとして取り扱わなければならない。

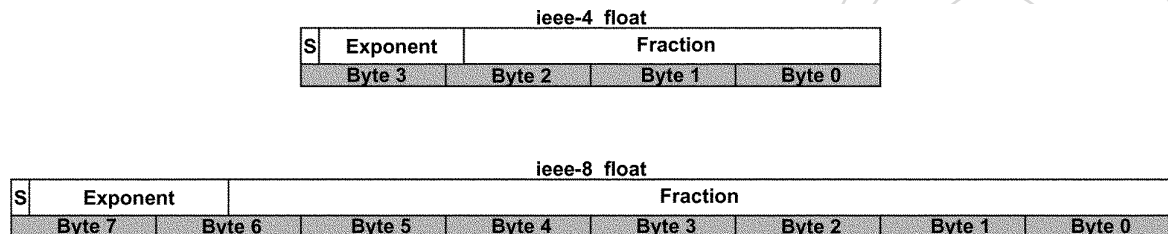


図 3
IEEE の浮動小数点フォーマット

表 4 実数の例

値	有理式	IEEE-4 形式
0.0	00000000 00000000	00000110 00000000 00000000 00000000 00000000
1.0	00000000 00000001	00000110 00000000 00000000 10000000 00111111
-0.5	00000011 00000010	00000110 00000000 00000000 00000000 10111111
0.3125	00000100 00001010 00010000	00000110 00000000 00000000 10100000 00111110
1/3	00000010 00000011	00000110 10101011 10101010 10101010 00111110
-2/13	00000101 00000010 00001101	00000110 11011001 10001001 00011101 10111110

7.4 文字列

7.4.1 文字列は、先頭に文字列内の文字数を表現する符号なし整数を付けたゼロ個以上のバイト(文字)のシーケンスである。

string -> length byte*

OASIS内の文字列は、意味によってさらにサブタイプに分けられる。**b-string** (“バイナリ文字列”)は、8 ビットの文字コードの組み合わせを任意の順序で格納できる文字列である。**a-string** (“ASCII文字列”)は、プリント可能なASCII文字コード(16 進数で 21 ～ 7E)およびスペース文字(16 進数で 20)のみを任意の順序で格納できる文字列である。**n-string** (“名前文字列”)は、プリント可能なASCII文字コード(16 進数で 21 ～ 7E)のみを格納でき、長さがゼロであってはならない。

7.4.2 プリント可能なASCII文字の集合は、16 進数の文字コード 21 ～ 7Eからなる。文字コードの昇順で、次の文字コードがある。

!"#\$%&'()*+,-./0123456789:;<=>?@	[21-40]
ABCDEFGHIJKLMN O PQRSTU VWXYZ [\]^_`	[41-60]
abcdefghijklmnopqrstuvwxyz{ }~	[61-7E]

これには、スペース(SP)、タブ(HT、VT)、すべての制御文字が除外される。

7.4.3 例外の取り扱い — OASISプロセッサは、**a-string**と**n-string**の不正な文字を致命的エラーとして取り扱わなければならない。長さがゼロの**n-string**は、致命的エラーとして取り扱わなければならない。

7.5 デルタ

7.5.1 **デルタ**は、幾何学的データ(座標、ベクトル、二次元のオフセット等)を表現する。

7.5.2 **1-delta**は、符号付き整数として格納され、水平または垂直の変位を表現する。符号ビットであるビット 0 は方向をコード化し、0 は東または北、1 は西または南である。残りのビットは大きさである。水平または垂直のアライメントは前後関係より暗黙的に定義される。

7.5.3 **2-delta**は、符号なし整数として格納され、水平または垂直の変位を表現する。ビット 0~1 は方向をコード化し、0 は東、1 は北、2 は西、3 は南である。残りのビットは大きさである。

7.5.4 **3-delta**は、符号なし整数として格納され、水平、垂直、45° の斜め方向の変位を表現する。ビット 0~2 は方向をコード化し、0 は東、1 は北、2 は西、3 は南、4 は北東、5 は北西、6 は南西、7 は南東である。残りのビットは(水平と垂直のデルタについては)大きさまたは(45° のデルタについては)x軸またはy軸への射影の大きさである。

7.5.5 **g-delta**には、二者択一のフォームがあり、単一の符号なし整数または符号無し整数のペアとして格納される。ビット 0 がゼロである場合は最初のフォームが示され、3-deltaと同様な方法で、水平、垂直、45° の斜め方向の変位を表現し、残りのビットは大きさを格納する。第二のフォームは汎用の(x,y)変位を表現し、符号なし整数のペアである。最初の整数のビット 0 は 1 である。最初の整数のビット 1 はx方向(0 が東、1 が西)である。最初の整数の残りのビットはx方向の大きさを表現する。二番目の整数のビット 0 はy方向(0 が北、1 が南)である。二番目の整数の残りのビットはy方向の大きさを表現する。両方のフォームは、g-deltaのリストに表記される。

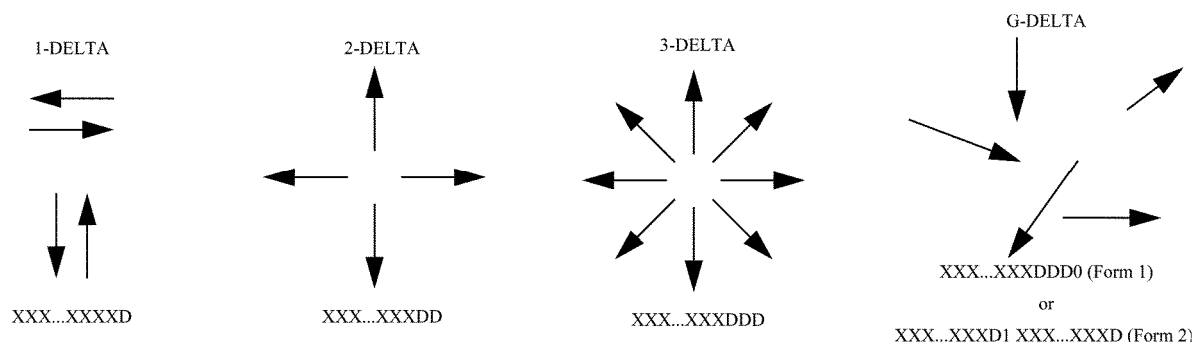


図 4
デルタのタイプ

表 5 デルタの例

ビットパターン	タイプ	意味
11111001 00100011	1-delta	$\Delta = -2300$
11111000 00100011	1-delta	$\Delta = +2300$
10011000 00101010	2-delta	$\Delta x = +1350$
10011011 00101010	2-delta	$\Delta y = -1350$
11001101 00000001	3-delta	$\Delta x = -25, \Delta y = +25$
11010111 00000111	3-delta	$\Delta x = +122, \Delta y = -122$
11101001 00000011 01111010	g-delta ₂	$\Delta x = +122, \Delta y = +61$
11101100 00000101	g-delta ₁	$\Delta x = -46, \Delta y = -46$
10111011 00000001 10110111 00001111	g-delta ₂	$\Delta x = -46, \Delta y = -987$

7.6 反復

7.6.1 **反復**は、セル配置、図形、テキストエレメントの「配列」を表現する。反復は、**PLACEMENT**, **<geometry>**, **TEXT**レコード自体の一部である。反復は、タイプをコード化する符号なし整数からなり、関連する反復パラメータがあればその後に付けられる。

表 6 反復のタイプ

タイプ	形式
0	<i>re-use the previous repetition definition</i>
1	x-dimension y-dimension x-space y-space
2	x-dimension x-space
3	y-dimension y-space
4	x-dimension x-space₁ ... x-space_{N-1}
5	x-dimension grid x-space₁ ... x-space_{N-1}
6	y-dimension y-space₁ ... y-space_{M-1}
7	y-dimension grid y-space₁ ... y-space_{M-1}
8	n-dimension m-dimension n-displacement m-displacement
9	dimension displacement
10	dimension displacement₁ ... displacement_{p-1}
11	dimension grid displacement₁ ... displacement_{p-1}

7.6.1.1 **x-dimension**, **y-dimension**, **x-space**, **y-space**, **dimension**, **n-dimension**, **m-dimension**, **grid**は、すべて符号なし整数である。**displacement**, **n-displacement**, **m-displacement**は、**g-delta**である。

7.6.2 **TYPE 0** は、モータル変数`repetition`に格納されている前の反復の記述を再使用することを示す(§ 10 を参照)。このタイプにはこれ以外の値は格納されない。

7.6.3 **TYPE 1** は、エレメント間に均一な水平スペースおよび垂直スペースがあるN列 ($N > 1$), M行 ($M > 1$) の配列である。**x-dimension**は $N - 2$ および**y-dimension**は $M - 2$ である。反復 ($i = 0, \dots, N-1$ および $j = 0, \dots, M-1$) のエレメント (i, j) の (**x-offset**, **y-offset**) ((水平, 垂直) 方向の累積スペーシング) は, ($i \times \mathbf{x-space}$, $j \times \mathbf{y-space}$) である。

7.6.4 **TYPE 2** は、エレメント間に均一な水平スペースがあるN列 ($N > 1$), 1 行のベクトルである。**x-dimension**は $N - 2$ である。反復 ($i = 0, \dots, N-1$) のエレメント i の (**x-offset**, **y-offset**) ((水平, 垂直) 方向の累積スペーシング) は, ($i \times \mathbf{x-space}$, 0) である。

7.6.5 **TYPE 3** は、エレメント間に均一な垂直スペースがある1列, M行 ($M > 1$) のベクトルである。**y-dimension**は $M - 2$ である。反復 ($j = 0, \dots, M-1$) のエレメント j の (**x-offset**, **y-offset**) ((水平, 垂直) 方向の累積スペーシング) は, (0, $j \times \mathbf{y-space}$) である。

7.6.6 **TYPE 4** は、エレメント間に均一ではない(可能性がある)水平スペースがあるN列 ($N > 1$), 1 行のベクトルである。**x-dimension**は $N - 2$ である。反復 ($i = 0, \dots, N-1$) のエレメント i の (**x-offset**, **y-offset**) ((水平, 垂直) 方向の累積スペーシング) は, ($\mathbf{x-space}_0 + \dots + \mathbf{x-space}_i$, 0) であり, $\mathbf{x-space}_0 = 0$ である。

7.6.7 **TYPE 5** は、反復の展開中にすべてのオフセット値に**grid**を乗算しなければならないことを除き**TYPE 4**と同じである。

7.6.8 **TYPE 6** は、エレメント間に均一ではない(可能性がある)垂直スペースがある1列, M行 ($M > 1$) のベクトルである。**y-dimension**は $M - 2$ である。反復 ($j = 0, \dots, M-1$) のエレメント j の (**x-offset**, **y-offset**) ((水平, 垂直) 方向の累積スペーシング) は, (0, $\mathbf{y-space}_0 + \dots + \mathbf{y-space}_j$) であり, $\mathbf{y-space}_0 = 0$ である。

7.6.9 **TYPE 7** は、反復の展開中にすべてのオフセット値に**grid**を乗算しなければならないことを除き**TYPE 6**と同じである。

7.6.10 **TYPE 8** は、エレメント間に均一で斜め方向(の可能性がある)変位があるN回 ($N > 1$), M回 ($M > 1$) の反復である。**n-dimension**は $N - 2$ および **m-dimension**は $M - 2$ である。**n-displacement**のコンポーネントである **nx-**

spaceおよびny-spaceを使用してn-displacementを定義すると(m-displacementも同様), 反復($i = 0, \dots, N-1$ and $j = 0, \dots, M-1$)の元素(i, j)の(x-offset, y-offset) ((水平, 垂直)方向の累積スペーシング)は, ($i \times nx\text{-space} + j \times mx\text{-space}, i \times ny\text{-space} + j \times my\text{-space}$)である。

7.6.11 **TYPE 9** は, エlement間に均一で斜め方向(の可能性がある)変位があるP元素($P > 1$)の反復である。**dimension**は $P - 2$ である。**displacement**のコンポーネントであるx-spaceおよびy-spaceを使用して**displacement**を定義すると, 反復($k = 0, \dots, P-1$)の元素kの(x-offset, y-offset) ((水平, 垂直)方向の累積スペーシング)は, ($k \times x\text{-space}, k \times y\text{-space}$)である。

7.6.12 **TYPE 10** は, Element間に均一ではない(可能性があり), 任意の二次元変位があるP元素($P > 1$)の反復である。**dimension**は $P - 2$ である。**displacement_k**のコンポーネントであるx-space_kおよびy-space_kを使用して**displacement_k**を定義すると, 反復($k = 0, \dots, P-1$)の元素kの(x-offset, y-offset) ((水平, 垂直)方向の累積スペーシング)は, ($x\text{-space}_0 + \dots + x\text{-space}_k, y\text{-space}_0 + \dots + y\text{-space}_k$), ここで $x\text{-space}_0 = y\text{-space}_0 = 0$)である。

7.6.13 **TYPE 11** は, 反復の展開中にすべてのオフセット値に**grid**を乗算しなければならないことを除き**TYPE 10**と同じである。

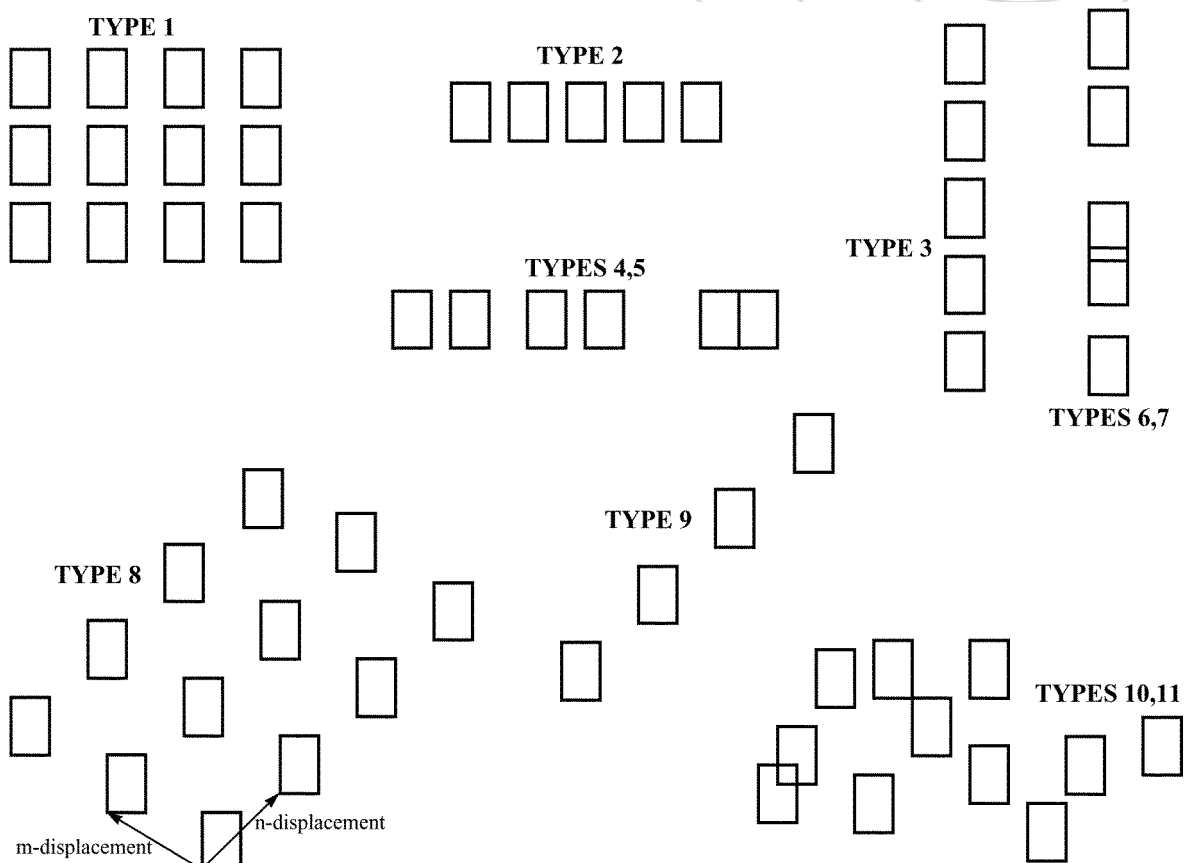


図 5
反復のタイプ

7.6.14 **例外の取り扱い** — 0 ~ 11 の範囲外の反復タイプは, 致命的エラーとして取り扱わなければならない。タイプが 0 の反復は, 1 つのセル内の最初の反復のタイプであってはならない。

7.7 ポイントリスト

7.7.1 **point-list**は、多角形やパスの幾何学座標のリストを表現し、タイプを示す符号なし整数からなり、複数のフォーマットの 1 つであるデルタのリストがその後に続く。(x,y)での初期頂点は、**POLYGON**レコードまたは**PATH**レコードによって提供され、**point-list**の一部ではなく、**vertex-count** (符号なし整数)は、初期頂点および暗黙の頂点を除外した点またはデルタの数である。

表 7 ポイントリストのタイプ

タイプ	形式	記述
0	vertex-count [<i>1-delta</i> [... <i>1-delta</i>]]	Implicit manhattan delta point-list (horizontal-first)
1	vertex-count [<i>1-delta</i> [... <i>1-delta</i>]]	Implicit manhattan delta point-list (vertical-first)
2	vertex-count [<i>2-delta</i> [... <i>2-delta</i>]]	Explicit manhattan delta point-list
3	vertex-count [<i>3-delta</i> [... <i>3-delta</i>]]	Explicit octangular delta point-list
4	vertex-count [<i>g-delta</i> [... <i>g-delta</i>]]	Explicit all-angle delta point-list
5	vertex-count [<i>g-delta</i> [... <i>g-delta</i>]]	Explicit all-angle double-delta point-list

7.7.2 タイプ 0 の**point-list**は、*1-delta*のリストからなり、最初の変位が暗黙的に水平であって水平と垂直が交互に切り替わる相対変位を表現する。このフォームで多角形の**point-list**を記述すると、最後の 2 つの変位は現在の点、直前のエッジ、開始点から一義に含意されるので省略される。多角形を記述する場合、**vertex-count**は 2 以上の偶数でなければならない。

7.7.3 タイプ 1 の**point-list**は、*1-delta*のリストからなり、最初の変位が暗黙的に垂直であって垂直と水平が交互に切り替わる相対変位を表現する。このフォームで多角形の**point-list**を記述すると、最後の 2 つの変位は現在の点、直前のエッジ、開始点から一義に含意されるので省略される。多角形を記述する場合、**vertex-count**は 2 以上の偶数でなければならない。

7.7.4 タイプ 2 の**point-list**は、*2-delta*のリストからなり、一連のマンハッタン相対変位を表現する。このフォームで多角形の**point-list**を記述すると、多角形は暗黙的に閉じていると仮定されるので、最後の変位は省略されるが、この最後の暗黙の変位はマンハッタン変位であって、 $\Delta x = 0$ または $\Delta y = 0$ でなければならない。

7.7.5 タイプ 3 の**point-list**は、*3-delta*のリストからなり、一連の八角形の相対変位を表現する。このフォームで多角形の**point-list**を記述すると、多角形は暗黙的に閉じていると仮定されるので、最後の変位は省略されるが、この最後の暗黙の変位は八角形の変位であって、角度は 45 度の倍数でなければならない。

7.7.6 タイプ 4 の**point-list**は、*g-delta*のリストからなり、一連の任意の角度の相対変位を表現する。このフォームで多角形の**point-list**を記述すると、多角形は暗黙的に閉じていると仮定されるので、最後の変位は省略される。

7.7.7 タイプ 5 の**point-list**は、*g-delta*のリストからなり、初期ベクトルを($\Delta x = 0$, $\Delta y = 0$)に設定した相対変位ベクトルに対する一連の調整量を表現する。各連続点の座標を計算するために、各連続*g-delta*のx成分とy成分が相対変位ベクトルに加算され、現在の点から次の点への相対変位を交互に記述する。このフォームで多角形の**point-list**を記述すると、多角形は暗黙的に閉じていると仮定されるので、最後の変位は省略される。このフォームの**point-list**は、曲率が極端ではない細かいグリッド上の大きな領域の曲線形状を近似した多角形やパスを、よりコンパクトに表現できるように意図されている。

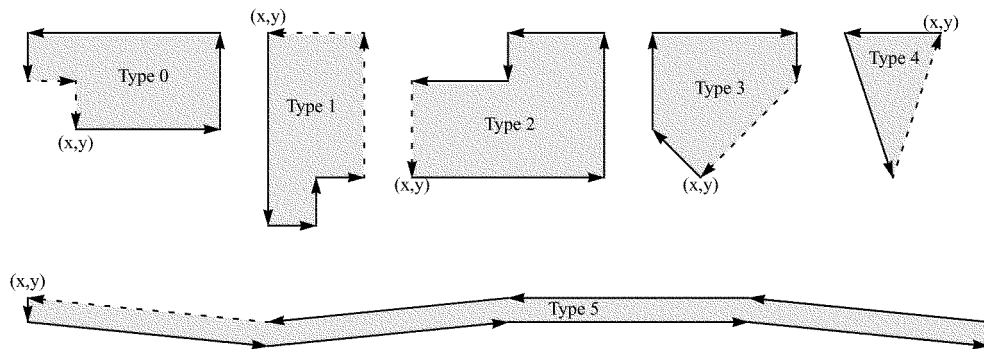


図 6
多角形を表現するポイントリスト

表 8 図 6 の多角形のポイントリスト

タイプ	ビットパターン
0	00000000 00000100 00001100 00001000 00010001 00000101
1	00000001 00000100 00010001 00000100 00000100 00000100
2	00000010 00000101 00100000 00011001 00010010 00001011 00010010
3	00000011 00000100 00010101 00100001 00110000 00010011
4	00000100 00000010 01000100 00001001 00001101
5	00000101 00001001 00000001 00000011 00101001 00000000 00000001 00000100 00000001 00000011 00000001 00000011 00101011 00000100 00101011 00000000 00000001 00000011 00000001 00000011

7.7.8 **例外の取り扱い** — 0 ～ 5 の範囲外のタイプの *point-list* は、致命的エラーとして取り扱わなければならない。0 ～ 1 のタイプの *point-list* については、連続する一致点や同一線形順序に並ぶ隣接したエッジは許されない。タイプ 2 の *point-list* を使用した多角形のマンハッタンではない暗黙の閉ベクトルまたはタイプ 3 の *point-list* を使用した多角形の八角形ではない暗黙の閉ベクトルは、致命的エラーとして取り扱わなければならない。0 ～ 1 のタイプの *point-list* を使用した多角形について、頂点数が奇数または 2 未満の場合は、致命的エラーとして取り扱わなければならない。

7.8 プロパティ値

7.8.1 **property-value** は、プロパティリストの 1 つの要素を格納する。タイプをコード化する符号なし整数からなり、その後には値自体または参照番号が続く。タイプ 0 ～ 7 は、表 3 で説明したスキームに従う実数である。

表 9 プロパティ値のタイプ

タイプ	形式
0-7	real (see Table 3)
8	unsigned-integer
9	signed-integer
10	a-string
11	b-string
12	n-string
13	propstring-reference-number (implied a-string)
14	propstring-reference-number (implied b-string)
15	propstring-reference-number (implied n-string)

7.8.2 **例外の取り扱い**— 0 ～ 15 の範囲以外のタイプのプロパティ値は、致命的エラーとして取り扱わなければならない。同じOASISファイル内に対応する**PROPSTRING**レコードがない**propstring-reference-number**を使用すると、致命的エラーとして取り扱わなければならない。

8 CELL参照

8.1 GDSIIストリームのようにOASIS内のセルは**名前**で識別される。**CELL**レコードはセルの定義を導入するだけでなく、その名前も定義する。**PLACEMENT**レコードは、配置されるセルを名前で参照する。GDSIIストリームのように、OASISには「無名」のセルはない。

9 レイヤ、データタイプ、テキストタイプ

9.1 GDSIIストリームのようにすべての<geometry>は、それに関連付けられるレイヤ番号とデータタイプ番号があり、すべてのテキストエレメントには、それに関連付けられるテキストレイヤ番号とテキストタイプ番号がある。

10 モーダル変数

10.1 圧縮を目的として、多くのOASISレコード内で選択されたデータエレメントは、モーダル変数または格納された状態を使用して暗黙に指定できる。ファイルの開始時および**CELL**レコードまたは<name>レコードに出会うと、*placement-x*, *placement-y*, *geometry-x*, *geometry-y*, *text-x*, *text-y*の例外を除き、すべてのモーダル変数は、未定義の状態に設定され、前述の例外は 0 に設定される。セルの記述にはさまざまなエレメントが現れるので、それらのエレメントで使用するモーダル変数はエレメントの定義による。これらのモーダル変数は、引き続きエレメントによって暗黙に使用することができる。モーダル変数は、*geometry-w*のような単一の値または*repetition*のような複数の変数構造を保持できる。

表 10 モーダル変数

モーダル変数	関連レコード
<i>Repetition</i>	PLACEMENT, TEXT, POLYGON, PATH, RECTANGLE, TRAPEZOID, CTRAPEZOID, CIRCLE, XGEOMETRY
<i>placement-x, placement-y, placement-cell layer, datatype</i>	PLACEMENT
<i>textlayer, texttype, text-x, text-y, text-string</i>	TEXT
<i>geometry-x, geometry-y</i>	POLYGON, PATH, RECTANGLE, TRAPEZOID, CTRAPEZOID, CIRCLE, XGEOMETRY
<i>xy-mode</i>	PLACEMENT, TEXT, POLYGON, PATH, RECTANGLE, TRAPEZOID, CTRAPEZOID, CIRCLE, XGEOMETRY, XYABSOLUTE, XYRELATIVE
<i>geometry-w, geometry-h</i>	RECTANGLE, TRAPEZOID, CTRAPEZOID
<i>polygon-point-list</i>	POLYGON
<i>path-halfwidth, path-point-list, path-start-extension, path-end-extension</i>	PATH
<i>ctrapezoid-type</i>	CTRAPEZOID
<i>circle-radius</i>	CIRCLE
<i>last-property-name, last-value-list</i>	PROPERTY

10.2 モーダル変数*xy-mode*は、表 10 に示すレコードタイプに関連するxフィールドとyフィールドについて、その解釈を支配する。2 つの解釈モード、絶対と相対が提供される。これらの 2 つのモードの動作の説明については、§ 21 を参照すること。

10.3 例外の取り扱い — 未定義の状態にあるモダル変数を暗黙に参照するOASISレコードは、致命的エラーとして取り扱わなければならない。

11 レコード

11.1 OASISファイル内の情報の基本単位は、**record**である。レコードは、**record-ID**をコード化する単一の符号なし整数からなり、その後にレコードの残りの部分である記述データが続く。この仕様では、**record-ID**の値は、アポストロフィで囲まれた 10 進数で表示される。

11.2 **CBLOCK**レコードは、通常の一連のレコードをバイト単位での圧縮形式によりカプセル化するので、特殊な場合である。OASISファイルを読み込み中に**CBLOCK**レコードに出会うと、まずデータを解凍する必要があり、それによって 1 つまたは複数の通常のレコードが生成され、順にデコードできる。**CBLOCK**レコードの詳細については、§ 35 を参照すること。

11.3 ほとんどのレコードには暗黙の長さがある。その長さを決定するためにはレコードを解析し、デコードしなければならない。**XNAME**レコード、**XELEMENT**レコード、**XGEOMETRY**レコードは、この例外である。これらは、そのすべてのユーザ定義のデータを単一の可変長の *b-string* にカプセル化するので、古いリーダでは読むことが出来ないOASISファイルを解釈しなくても単に文字列の長さに注目してそのレコード全体をスキップできるので、新しいレコードタイプのプロタイピング、埋め込まれた専有データの隠蔽、ローカルな相互運用不能な拡張のサポート等に使用できる。

11.4 例外の取り扱い — OASISプロセッサは、**CBLOCK**レコードが別の**CBLOCK**レコード内にネストしていると、致命的エラーとして取り扱わなければならない。

12 PADレコード

12.1 **PAD**レコードは、OASISファイル内にスペースを予約するシンプルな方法を提供する。次のフォーマットがある。

‘0’

12.2 **PAD**レコードは、他の 2 つのレコード間に挿入できる。

12.3 例外の取り扱い — **START**レコードの前または**END**レコードの後に**PAD**レコードが存在すると、致命的エラーとして取り扱わなければならない。

13 STARTレコード

13.1 **START**レコードは、OASISファイルの開始を識別し、¶6.4 で説明した<magic-bytes>シーケンスの直後に続く。次のフォーマットがある。

‘1’ version-string unit offset-flag [table-offsets]

13.2 **version-string**は、値がOASIS仕様のこのバージョンの値“1.0”の *a-string* である。バージョン“1.0”は、本文書で記述するOASISフォーマットに対応する。

13.3 **unit**宣言は、そのOASISファイルの座標システムのグローバルな精度をミクロンあたりのグリッドステップ数で指定する正の実数である。

13.4 **table-offsets**構造体が**START**レコードに格納される場合は**offset-flag** (符号なし整数)は 0 であり、**table-offsets**が**END**レコードに格納される場合は**offset-flag** (符号なし整数)は 1 である。**table-offsets**を**END**レコードに格納するオプションは、シークアンドアップデートのアクセスを必要とせずにOASISファイルをシーケンシャルに書き込み可能とし、その後のリーダがそのOASISファイルをセルレベルでランダムアクセス可能とするために提供されている。

13.5 **table-offsets**構造体は、6 ペアの符号なし整数からなる。それぞれのペアは、*flag*フィールドおよびそれに対応する*byte-offset*フィールドであり、次の順序となる。