M106V03

Debug Solutions Power Debugger

Debug System for Boundary Scan Board

コマンドリファレンス　付録－Ｂ

ＢＳＤＬ解説

Debug Solutions Power Debugger

ご注意

１．このソフトウェアの著作権は、Ｄｅｂｕｇ　Ｓｏｌｕｔｉｏｎｓ社にあります。
２．このソフトウェアおよびマニュアルの一部または全てを無断で使用、複製することはできません。
３．ソフトウェアは、コンピュータ１台につき１セット購入が原則となっております。
４．このソフトウェアおよびマニュアルは、本製品の使用許諾契約書のもとでのみ使用可能です。
５．このソフトウェアおよびマニュアルを運用した結果の影響については、いっさい責任をおいかねますのでご了承下さい。

目次

１．ＢＳＤＬ概要

BSDL（Boudary Scan Described Language）はデバイス内のバウンダリスキャン回路の特性を表現したテキストファイルで、VHDL（IEEE Std. 1076－1993）のサブセットとしてIEEE Std. 1149. 1b－1994（IEEE Standard Test Access Port and Boundary－Scan Architecture） に規定されています。

BSDLは最初、1990年にドラフトバージョンとして最初の提案がなされました。その後、いくつかの変更がなされ、1994年にStd. 1149. 1bとして承認されました。

２．ＢＳＤＬで使用される単語について

２．１　キャラクタセット

BSDLでの単語表記は大文字と小文字の認識はおこないません。又使用できる文字は以下のとおりです。

1）大文字、小文字　　:A, B, C･･･Z ，　a, b, c･･･z
2）数字　　:０－９
3）特殊文字　　:“　＆　‘（　）　＊　，　－　．　：　；　＜　＝　＞　＿　その他の特殊文字はコメントとしては使用可能です。
4）区切り文字　　:空白文字（スペース）　Tab文字　　改行　ラインフィード　フォームフィード

２．２　単語

BSDLで使用される単語は、１文字または複数のアルファベット文字、数字或いはアンダースコア（__）から構成されます。又、単語の文字数の上限はありません。以下の文字は有効な単語とみなされます。

```
BSDL                          -- 有効な単語
IEEE__STD__1149__1            -- 有効な単語
```

アンダースコア（__）はVHDLの規定により、最後の文字には使用できません。以下の文字は不正な単語と認識されます。

```
IEEE__STD__1149__             -- 不正な単語
```

２．３　ＢＳＤＬ予約語

以下に示す単語はBSDLの予約語となっており、ユーザーが使用する識別子としては使用できません。但しコメント内で使用することは可能です。

" BC__0 "	" BC__1 "	" BC__2 "
" BC__3 "	" BC__4 "	" BC__5 "
" BC__6 "	" BC__7 "	" BC__8 "
" BC__9 "	" BC__10 "	" BC__11 "
" BC__12 "	" BC__13 "	" BC__14 "
" BC__15 "	" BC__16 "	" BC__17 "
" BC__18 "	" BC__19 "	" BC__20 "
" BC__21 "	" BC__22 "	" BC__23 "

" BC__24 "	" BC__25 "	" BC__26 "
" BC__27 "	" BC__28 "	" BC__29 "
" BC__30 "	" BC__31 "	" BC__32 "
" BC__33 "	" BC__34 "	" BC__35 "
" BC__36 "	" BC__37 "	" BC__38 "
" BC__39 "	" BC__40 "	" BC__41 "
" BC__42 "	" BC__43 "	" BC__44 "
" BC__45 "	" BC__46 "	" BC__47 "
" BC__48 "	" BC__49 "	" BC__50 "
" BC__51 "	" BC__52 "	" BC__53 "
" BC__54 "	" BC__55 "	" BC__56 "
" BC__57 "	" BC__58 "	" BC__59 "
" BC__60 "	" BC__61 "	" BC__62 "
" BC__63 "	" BC__64 "	" BC__65 "
" BC__66 "	" BC__67 "	" BC__68 "
" BC__69 "	" BC__70 "	" BC__71 "
" BC__72 "	" BC__73 "	" BC__74 "
" BC__75 "	" BC__76 "	" BC__77 "
" BC__78 "	" BC__79 "	" BC__80 "
" BC__81 "	" BC__82 "	" BC__83 "
" BC__84 "	" BC__85 "	" BC__86 "
" BC__87 "	" BC__88 "	" BC__89 "
" BC__90 "	" BC__91 "	" BC__92 "
" BC__93 "	" BC__94 "	" BC__95 "
" BC__96 "	" BC__97 "	" BC__98 "
" BC__99 "		
" AT__PINS "	" BIDIR "	" BIDIR__IN "
" BIDIR__OUT "	" BORTH "	" BOUNDARY "
" BOUNDARY__LENGTH "	" BOUNDARY__REGISTER "	" BSCAN__INST "
" BSDL__EXTENSION "	" BYPASS "	" CAP "
" CAP__DATA "	" CAPUTURES "	" CELL__DATA "
" CELL__INFO "	" CELL__TYPE "	" CLAMP "
" CLOCK "	" CLOCK__INFO "	" CLOCK__LEVEL "
" COMPLIANCE__PATTERN "	" COMPONENT__CONFORMANCE "	
" CONTROL "	" CONTROLR "	" DESIGN__WARNING "
" DEVICE__ID "	" DIFFERENTIAL__CURRENT "	
" DIFFERENTIAL__VOLTAGE "	" EXPECT__DATA "	
" EXTEST "	" HIGHZ "	" ID__BITS "
" ID__STRING "	" ID__CODE "	" ID__CODE__REGISTER "
" INPUT "	" INSTRUCTION__CAPTURE "	
" INSTRUCTION__LENGTH "	" INSTRUCTION__OPCODE "	
" INSTRUCTION__PRIVATE "	" INTERNAL "	" INTEST "

" INTEST__EXECUTION "	" KEEPER "	" LOW "
" OBSERVE__ONLY "	" OBSERVING "	" ONE "
" OUTPUT2 "	" OUTPUT3 "	PHYSICAL__PIN__MAP "
" PI "	" PIN__MAP "	" PIN__MAP__STRING "
" PO "	" PORT__GROUPING "	" PULL0 "
" PULL1 "	" REGISTER__ACCESS "	" RUNBIST "
" RUNBIST__EXECUTION "	" SAMPLE "	" STD__1149__1__1990 "
" STD__1149__1__1993 "	" STD__1149__1__1994 "	" TAP__SCAN__CLOCK "
" TAP__SCAN__IN "	" TAP__SCAN__MODE "	" TAP__SCAN__OUT "
" TAP__SCAN__RESET "	" UPD "	" USERCODE "
" USERCODE__REISTER "	" WAIT__DURATION "	" WEAK0 "
" WEAK1 "	" X "	" Z "
" ZERO "		

２．４　ＶＨＬＤ予約語

以下に示す単語はVHDLの予約語となっており、BSDLの中でユーザーが使用する識別子としては使用できません。
但しコメント内で使用することは可能です。

" ZERO "	" ABS "	" ACCESS "
" AFTER "	" ALIAS "	" all "
" AND "	" ARCHITECTURE "	" array "
" ASSERT "	" attribute "	" BEGIN "
" bit "	" bit__vector "	" BLOCK "
" body "	" buffer "	" BUS "
" CASE "	" COMPONENT "	" CONFIGURATION "
" constant "	" DISCONNECT "	" downto "
" ELSE "	" ELSEIF "	" end "
" entity "	" EXIT "	" FALSE "
" FILE "	" FOR "	" FUNCTION "
" GENERATE "	" generic "	" GUARDED "
" IF "	" IMPURE "	" in "
" INERTIAL "	" inout "	" is "
" LABEL "	" LIBRARY "	" linkage "
" LITERAL "	" LOOP "	" MAP "
" MOD "	" NAND "	" NEW "
" NEXT "	" NOR "	" NOT "
" NULL "	" of "	" ON "
" OPEN "	" OR "	" OTHERS "
" out "	" package "	" port "
" positive "	" POSTPONED "	" PROCEDURE "

" PROCESS "	" PURE "	" range "
" record "	" REGISTER "	" REJECT "
" REM "	" REPORT "	" RETURN "
" ROL "	" ROR "	" SELECT "
" SEVERITY "	" SHARED "	" signal "
" SLA "	" SLL "	" SRA "
" SRL "	" string "	" subtype "
" THEN "	" to "	" TRANSPORT "
" true "	" type "	" UNAFFECTED "
" UNITS "	" UNTIL "	" use "
" VARIABLE "	" WAIT "	" WHEN "
" WHILE "	" WITH "	" XNOR "
" XOR "		

２．５　文字列

BSDL内の文字列はダブルクォーテションマーク(")で囲まれていなければなりません。又ダブルクォーテションマーク(")は文字列としては使用できません。

```
" This  is  String "                    -- 文字列として認識されます。
" This  is  String " " "                -- 不当な文字列となります。
```

文字列は　&　マークを使用して連結することが可能です。

```
" This  is  First  String. "  &
" This  is  concatenated  previous  String. "
```

この2行の文字列は　" This is First String. This is concatenated previous String. "　と同じです。

２．６　コメント

２つのハイフン（--）から行の最後までがコメントとしてみなされます。2．1　キャラクタセットで定義されている特殊文字のほかに、VHDLで定義されている特殊文字も使用できます。

３．文法定義の表記方法について

３．１　表記記号について

（1）“＜”，“＞”で囲まれた単語　　:BSDLで使用される文法要素
（2）“{”，“}”で囲まれた要素　　:0個以上の要素を含みます（要素をふくまない場合もあります）。
（3）“[”，“]”で囲まれた要素　　:0又は1個の要素を含みます（要素をふくまない場合もあります）。
（4）“「”，“」”で囲まれた要素　　:順序にかかわらず表記可能です。
（5）“｜”　　:前後の要素は二者択一で使用されます。
（6）“::＝”　　:前の要素は後ろの要素で定義されます。

３．２　最小構成要素

（1）＜VHDL identifiier＞　:“2．2 単語” で定義される有効な識別子
（2）＜integer＞　:符号無し数字から構成されるVHDL integer
（3）＜real number＞　:＜integer＞．＜integer＞ 又は ＜integer＞．＜integer＞E＜integer＞ で表記されるVHDL real number。全ての文字列に空白文字を含んではいけません。
（4）＜pattern＞　:1つ以上の 0，1，X の文字から構成されるステートパターンを示す表記です。全ての文字列に空白文字を含んではいけません。0は“Low State”を、1は“High Stateを、Xは”Don‘t Care“を示します。
（5）＜32－bit pattern＞　:32bit ＜pattern＞ 文です。

３．３　共通要素

（1）＜port ID＞
port ID は外部信号とインターフェスを行う信号の名称を定義します。＜port ID＞は以下の構文で定義されます。

＜port ID＞::＝＜port name＞｜＜subscripted port name＞
＜port name＞::＝＜VHDL identifiier＞
＜subscripted port name＞::＝＜VHDL identifiier＞（ ＜subscript＞ ）
＜subscript＞::＝＜integer＞

（2）＜instruction name＞
instruction name はこのスタンダードで定義されたインストラクション名又はデバイスメーカーによってつけられたインストラクションを示します。

＜instruction name＞::＝　BYPASS｜CLAMP｜EXTEST｜HIGHZ｜IDCODE｜INTEST
RUNBIST｜SAMPLE｜USERCODE｜＜VHDL identifier＞

４．文法定義

４．１　ＢＳＤＬの構造について

BSDLの構造は以下のような構造をとらなければなりません。

```
＜BSDL description＞ ::＝
    entity ＜component name＞ is
                                ＜generic parameter＞
                                ＜logical port description＞
                                ＜standard use statement＞
                                {＜use statement＞}
                                ＜component conformance statement＞
                                ＜device package pin mappings＞
                                [＜grouped port identificeation＞]
                                ＜scan port identification＞
                                [＜compliance enable description＞]
                                ＜instruction register description＞
                                [＜optional register description＞]
                                [＜register access description＞]
                                ＜boundary－scan register description＞
                                [＜runbist description＞]
                                [＜intest description＞]
                                {＜BSDL extentions＞}
                                [＜design warning＞]
    end ＜component name＞;
```

※BSDL表記においては、その要素順序は上記の構造であらわされる順序をまもらなければなりません。

☆注意点

（1）entity 文を構成する各属性内の全ての component name はentity文の component name と一致しなければなりません。

４．２ Ｇｅｎｅｒｉｃ ｐａｒａｍｅｔｅｒ文

generic parameter 文は複数コンポーネント表記を行う(複数の物理ポートが定義される)場合、論理ポートに物理ポート名を割りつけます。

```
<generic parameter>     ::= generic ( PHYSICAL_PIN_MAP : string );
                          | generic ( PHYSICAL_PIN_MAP : string
                                        := <default device package type> );

<default device package type> ::= " <VHDL identifier> "
```

<default device package type>は“4.7 Deivce package pin mappings”で記述される <pin mapping name> で定義されます。

☆例:

```
generic ( PHYSICAL_PIN_MAP : string );
```

又は

```
generic ( PHYSICAL_PIN_MAP : string := "JT_PACKAGE" );
```

☆注意点

(1) <default device package type> は“4.7 Deivce package pin mappings”で記述される <pin mapping name> で一意に定義されていなければなりません。

４．３ Ｌｏｇｉｃａｌ ｐｏｒｔ ｄｅｓｃｒｉｐｔｉｏｎ文

BSDLのポート定義がVHDLのポート定義とほぼ同じです。ポート定義はデバイスの物理的なピンに対して、分かりやすい名前をつけるために用いられます。IEEE Std. 1149.1ではデジタル信号でないピン(電源、GND, ノンコネクト、アナログ信号等)は BSDLの中で <pin type>を linkage の属性で定義することを強く推奨しています。

```
<logical port description> ::= port( <pin spec> { ; <pin spec> } );
<pin spec>                 ::= <identifier list> : <pin type> <port dimension>
<identifier list>          ::= <port name> { , <port name>}
<pin type>                 ::= in | out | buffer | inout | linkage
```

```
<port dimension>        ::= bit | bit_vector( <range> )
<range>                 ::= <integer_1> to <integer_2>
                          | <integer_2> downto <integer_1>
<integer_1>             ::= <integer>
<integer_2>             ::= <integer>
```

<pin type> の 定義

in	:入力ピン
out	:出力ピン
buffer	:2ステート出力ピン
inout	:双方向ピン
linkage	:そのほかのピンタイプ(電源、GND, アナログ、ノンコネクト信号等)

<port dimension> はポートの信号数を示しています。<port dimension>が bit ならばその信号は1bit の信号を示しており、bit_vector であれば複数の信号(バス)を示しています。

※ 最後の<pin spec>定義の後には セミコロン(;) がつかないことに注意して下さい。

☆例:

```
port (  OE_NEG1:        in      bit;
        Y1:             out     bit_vector(1 to 4);
        Y2:             out     bit_vector(1 to 4);
        A1:             in      bit_vector(1 to 4);
        A2:             in      bit_vector(1 to 4);
        OE_NEG2:        in      bit;
        GND, VCC:       linkage bit;
        TDO:            out     bit;
        TDI, TMS, TCK:  in      bit       );
```

☆注意点

(1) <bit_vector> のバス表記を示す <integer_1> の値は <integer_2>の値に等しいか大きな値でなければなりません。
(2) Logical port description の <port name> は同じ名前を複数回定義してはいけません。

４．４ Ｓｔａｎｄａｒｄ ｕｓｅ 文

standard use 文は一つの Standard VHDL Package を識別します。この Standard VHDL Package には BSDL の中から参照する属性、型、定数等を定義します。

```
＜standard use statement＞            ::＝ use ＜standard VHDL package identifier＞. all;
＜standard VHDL package identifier＞  ::＝ STD__1149__1__1994
                                        | ＜other package identifier＞
＜other package identifier＞          ::＝ ＜VHDL identifier＞
```

現在のStandard Packageには STD__1149__1__1994 と STD__1149__1__1990 があります。

☆例:

```
use STD__1149__1__1994. all;
```

又は

```
use STD__1149__1__1990. all;
```

☆注意点

standard use 文は、BSDLが適用するスタンダードのバージョンも示しています。STD__1149__1__1990であれば IEEE Std. 1149. 1－1990 に対応し、STD__1149__1__1990 であればIEEE Std. 1149. 1b－1994 に対応しています。

1990年バージョンに対応したBSDLでは、以下の構文をサポートしていません。

＜compornent conformance statement＞
＜grouped port identification＞
＜compliance enable description＞
＜runbist description＞
＜intest description＞
＜BSDL extenstion＞

４．５ Ｕｓｅ 文

use 文は BSDL の中から参照する属性、型、定数等を定義した VHDL Package を指定します。＜use＞文はオプションです。

＜use statement＞ ::＝ use ＜user VHDL package identifier＞. all;
＜user VHDL package identifier＞ ::＝ ＜VHDL identifier＞

＜user VHDL package identifier＞はユーザーが指定する属性、型、定数等の情報を定義したVHDL package名です。 “. all” のサフィックスはそのpackageの全ての項目が有効であることを意味します。VHDLでは“. all” 以外の指定も可能ですが、BSDLでは“. all” のサフィックスのみが使用できます。

☆例:

```
use My_Package. all;                    -- ユーザー指定のVHDL Packageです。
```

４．６ Ｃｏｍｐｏｒｎｅｎｔ ｃｏｎｆｏｒｍａｎｃｅ 文

compornent conformance 文は対応するデバイスの回路が対応するスタンダードの版数を示します。

＜compornent conformance statement＞ ::＝ attribute COMPORNENT_CONFIRMANCE
of ＜compnent name＞ : entity is
＜conformance string＞;
＜conformance string＞ ::＝ " ＜conformance identification＞ "
＜conformance identification＞ ::＝ STD_1149_1_1990 | STD_1149_1_1993

＜conformance identification＞が STD_1149_1_1990 であれば IEEE Std. 1149. 1－1990に対応しており、STD_1149_1_1993 であれば IEEE Std. 1149. 1a－1993 に対応しています。

☆例:

```
attribute COMPORNENT_CONFIRMANCE My_LSI is STD_1149_1_1993;
```

☆注意点

1990年バージョンに対応したBSDLでは、compornent conformance 文をサポートしていません。

４．７ Ｄｅｖｉｃｅ ｐａｃｋａｇｅ ｐｉｎｍａｐｐｉｎｇｓ 文

device package pin mappings 文はデバイスの論理的な信号名(ピン名)と物理的な信号名(ピン番号)とを結びつける働きをします。

```
<device package pin mappings> ::= <pin map statement> <pin mapppings>
<pin map statement>   ::= attribute PIN_MAP of <component name> : entity is
                          PHYSICAL_PIN_MAP;
<pin mappings>        ::= <pin mapping> { <pin mapping> }
<pin mapping>         ::= constant <pin mapping name> : PIN_MAP_STRING :=
                          <map string>;
<pin mapping name>    ::= <VHDL identifier>
<map string>          ::= " <port map> {, <port map> } "
<port map>            ::= <port name>:<pin list>
<pin list>            ::= <pin ID> | ( <pin ID> { , <pin ID> } )
<pin ID>              ::= <VHDL identifier> | <integer>
```

☆例:

```
attribute PIN_MAP of xx74bct8244a : entity is PHYSICAL_PIN_MAP;

constant JT : PIN_MAP_STRING := "OE_NEG1:1, Y1:(2, 3, 4, 5), " &
                                "Y2:(7, 8, 9, 10), A1:(23, 22, 21, 20), " &
                                "A2:(19, 17, 16, 15), OE_NEG2:24, GND:6, " &
                                "VCC:18, TDO:11, TDI:14, TMS:12, TCK:13";

constant DW : PIN_MAP_STRING := "OE_NEG1:1, Y1:(2, 3, 4, 5), " &
                                "Y2:(7, 8, 9, 10), A1:(23, 22, 21, 20), " &
                                "A2:(19, 17, 16, 15), OE_NEG2:24, GND:6, " &
                                "VCC:18, TDO:11, TDI:14, TMS:12, TCK:13";
```

attribute PIN_MAP of に続く component name は“4. 1 generic parameter 文”で定義された component name が使用されます。

constant に続く pin mapping 宣言文は、デバイスのパッケージ毎のピン配置を定義しています。機能は同じで、複数のPKGの種別(DIP, SOP, BGA等)がある場合、pin mapping 文を複数宣言することで、一つのBSDLで表記することができます。<port name>:<pin list> で信号名をピン番号に結びつけます。<port name>はLogical port description文で宣言された信号名が使用され、bit_vector でバス宣言された信号は<pin list>でバスの数だけピン番号が定義されます。

☆注意点

（1） ＜pin mapping＞内の＜pin ID＞は複数回の定義はできません。
（2） Logical port description文で ＜pin type＞ が linkage 属性以外の＜port name＞は＜pin mappings＞で定義されていなければなりません。又＜pin mappings＞で定義されている＜port name＞はLogical port description文で定義されていなければなりません。
（3） ＜pin list＞の要素の数はLogical port description文定義されている bit_vector の要素の数と一致しなければなりません。

４．８ Ｇｒｏｕｐｅｄ ｐｏｒｔ ｉｄｅｎｔｉｆｉｃａｔｉｏｎ 文

grouped port identification 文は1ビットのデータを複数の信号線を使って表現するような場合に使用されます。具体的には、差動電圧や差動電流を用いた信号に対応するために用意されています。差動信号の個々の信号はアナログ信号ですが、2本をひとまとめとして考えればデジタル信号といえます。grouped port identification 文はこのような信号に対応します。

```
＜grouped port identification＞ ::＝ attribute PORT_GROUPING of ＜component name＞
                                        : entity is ＜group table string＞;
＜group table string＞    ::＝ " ＜group table＞ "
＜group table＞           ::＝ ＜twin group entry＞ { , ＜twin group entry＞ }
＜twin group entry＞      ::＝ ＜twin group type＞ ( ＜twin group list＞ )
＜twin group type＞       ::＝ DIFFERENTIAL_VOLTAGE | DIFFERENTIAL_CURRENT
＜twin group list＞       ::＝ ＜twin group＞ { , ＜twin group＞ }
＜twin group＞            ::＝ ( ＜representative port＞ , ＜associated port＞ )
＜representative port＞   ::＝ ＜port ID＞
＜associated port＞       ::＝ ＜port ID＞
```

＜representative port＞は差動ペアのうちの正極性（＋）信号に対応し、＜associated port＞は負極正（－）に対応します。

☆例:

```
entity sadou is
generic( Physical_Pin_Map : string := "SOP" );

port (   CLK : in bit;
         Datao0P, Datao1P, Datao2P, Datao3P : out bit;
         Datao0N, Datao1N, Datao2N, Datao3N : out bit;
```

```
            Datai0P, Datai1P, Datai2P, Datai3P  :  in  bit;
            Datai0N, Datai1N, Datai2N, Datai3N  :  in  bit;
            GND  ,  Vcc  :  linkage  bit;
            xEn  ,  TCK  ,  TMS  ,  TDi  :  in  bit;
            TDo  :  out  bit  );

    use  STD__1149__1__1994. all;

    attribute  PIN__MAP  of  xx74bct8244a  :  entity  is  PHYSICAL__PIN__MAP;
    constant  SOP  :  PIN__MAP__STRING  :=        "CLK  :1  ,  xEn  :6  , " &
                      "Datao0P:2  ,  Datao1P:3  ,  Datao2P:4  ,  Datao3P:5  , " &
                      "Datao0N:7  ,  Datao1N:8  ,  Datao2N:9  ,  Datao3N:10  , " &
                      "Datai0P:15  ,  Datai1P:16  ,  Datai2P:17  ,  Datai3P:18  , " &
                      "Datai0N:19  ,  Datai1N:20  ,  Datai2N:21  ,  Datai3N:22  , " &
                      "GND:12  ,  VCC:24,  TDO:14,  TDI:13,  TMS:12,  TCK:11 ";

    attribute  PORT__GROUPING  of  sadou  :  entity  is
                      "Differential__Voltage(       (  Datao0P  ,  Datao0N  )  ,  " &
                                                    (  Datao1P  ,  Datao1N  )  ,  " &
                                                    (  Datao2P  ,  Datao2N  )  ,  " &
                                                    (  Datao3P  ,  Datao3N  )  )  ,  " &
                      "Differential__Current(       (  Datai0P  ,  Datai0N  )  ,  " &
                                                    (  Datai1P  ,  Datai1N  )  ,  " &
                                                    (  Datai2P  ,  Datai2N  )  ,  " &
                                                    (  Datai3P  ,  Datai3N  )  )  ,  " &

                    (  --------------  中略  --------------  )

    attribute  BOUNDARY__REGISTER  of  sadou  :  entity  is
              "9  (  BC__1  ,  CLK        ,  input  ,  X  )  ,  " &
              "8  (  BC__1  ,  xEN        ,  input  ,  X  )  ,  " &
              "8  (  BC__1  ,  *          ,  control  ,  1  )  ,  " &
              "7  (  BC__1  ,  Data0iP    ,  input  ,  X  )  ,  " &
              "6  (  BC__1  ,  Data1iP    ,  input  ,  X  )  ,  " &
              "5  (  BC__1  ,  Data2iP    ,  input  ,  X  )  ,  " &
              "4  (  BC__1  ,  Data3iP    ,  input  ,  X  )  ,  " &
              "3  (  BC__1  ,  Data0iP    ,  output3  ,  X  ,  8  ,  1  ,  Z  )  ,  " &
              "2  (  BC__1  ,  Data1iP    ,  output3  ,  X  ,  8  ,  1  ,  Z  )  ,  " &
              "1  (  BC__1  ,  Data2iP    ,  output3  ,  X  ,  8  ,  1  ,  Z  )  ,  " &
              "0  (  BC__1  ,  Data3iP    ,  output3  ,  X  ,  8  ,  1  ,  Z  )";

    end  sadou;
```

☆注意点

（1） 1990年バージョンに対応したBSDLでは、Grouped port identification 文をサポートしていません。
（2） ＜representative port＞, ＜associated port＞で定義されている＜port name＞はLogical port description文で定義されていなければなりません。
（3） ＜twin group＞内の＜representative port＞, ＜associated port＞がDevice package pinmappingsでバス定義されている場合、要素の数はLogical port description文定義されている bit_vector の要素の数と一致しなければなりません。
（4） ＜twin group＞内の＜representative port＞, ＜associated port＞は同じ pin type でなければなりません。

４．９ Ｓｃａｎ ｐｏｒｔ ｉｄｅｎｔｉｆｉｃａｔｉｏｎ 文

scan port identification はデバイスのバウンダリスキャン制御信号の定義を行います。

＜scan port identification＞ ::＝「＜TCK stmt＞ ＜TDI stmt＞ ＜TMS stmt＞ ＜TDO stmt＞
[＜TRST stmt＞] 」

＜TCK stmt＞ ::＝ attribute TAP_SCAN_CLOCK of ＜port ID＞ : signal is
(＜clock record＞);
＜TDI stmt＞ ::＝ attribute TAP_SCAN_IN of ＜port ID＞ : signal is true;
＜TMS stmt＞ ::＝ attribute TAP_SCAN_MODE of ＜port ID＞ : signal is true;
＜TDO stmt＞ ::＝ attribute TAP_SCAN_OUT of ＜port ID＞ : signal is true;
＜TRST stmt＞ ::＝ attribute TAP_SCAN_RESET of ＜port ID＞ : signal is true;
＜clock record＞ ::＝ ＜real number＞ , ＜halt state value＞
＜halt state value＞ ::＝ LOW | BOTH

＜TCK stmt＞ , ＜TDI stmt＞ , ＜TMS stmt＞ , ＜TDO stmt＞ , ＜TRST stmt＞ 定義文は各バウンダリスキャン制御信号のピン名定義を行います。
＜TCK stmt＞ 文の ＜clock record＞ の要素である ＜real number＞ はTCKの最高動作周波数を示しています。又、＜halt state value＞ はテスト信号を保持できるTCKの状態を示しており、BOTH であれば “L” 又は “H” のどちらの状態でもTCKを停止できます。

☆例:

```
attribute TAP_SCAN_IN      of TDI : signal is true;
attribute TAP_SCAN_MODE    of TMS : signal is true;
attribute TAP_SCAN_OUT     of TDO : signal is true;
attribute TAP_SCAN_CLOCK of TCK : signal is (20.0e6, BOTH);
```

☆注意点

（1） ＜TDO stmt＞ で定義される ＜port ID＞ は“4．3 Logical port description文”で定義された名前でポート名でなければなりません。又その ＜pin type＞ は out でなければなりません。
（2） ＜TCK stmt＞ ， ＜TDI stmt＞ ， ＜TMS stmt＞ ， ＜TRST stmt＞ は“4．3 Logical port description文”で定義された名前でポート名でなければなりません。又その ＜pin type＞ は in でなければなりません。
（3） ＜scan port identification＞ の ＜port ID＞ は grouped port で定義されているポート名は使用できません。

４．１０ Ｃｏｍｐｌｉａｎｃｅ ｅｎａｂｅｌ ｄｅｓｃｒｉｐｔｉｏｎ 文

Compliance enable description は、バウンダリスキャン試験に影響を与える信号に対する安全なパターンを示します。Compliance enable description で定義された信号は、バウンダリスキャン試験を行う場合、設定されたパターンを入力し続けなければなりません。

```
＜compliance enable description＞ ::＝ attribute COMPLIANCE_PATTERNS of
                                        ＜component name＞ : entity is
                                        ＜compliance pattern string＞;
＜compliance pattern string＞  ::＝ " （ ＜compliance port list＞ ）（ ＜pattern list＞ ） "
＜compliance port list＞       ::＝ ＜port ID＞ { ， ＜port ID＞ }
＜pattern list＞               ::＝ ＜pattern＞ { ， ＜pattern＞ }
```

＜compliance port list＞に複数の＜port ID＞が定義される場合、＜port ID＞の並びの順に＜pattern list＞のビットが対応します。

☆例:

```
attribute COMPLIANCE_PATTERNS of hogeLSI : entity is
                                  "（xEN1 ， xEN2 ， xEN3 ， xEN4），（1111）" ;
```

Std．1149．1bに準拠した試験を実行するソフトウェアは、デバイスの試験を実行する前に、Compliance enable状態を保証する必要があります。

☆注意点

（1） 1990年バージョンに対応したBSDLでは、Compliance enabel description 文をサポートしていません。
（2） ＜compliance port list＞ の ＜port ID＞ の数と ＜pattern＞ のビットの数は一致しなければなりません。
（3） ＜compliance port list＞ の ＜port ID＞ は“4．9 scan port identification”で定義されている

ポート名は使用できません。

（4） ＜compliance port list＞ の ＜port ID＞ は “4．3 Logical port description文” で定義された名前でポート名でなければなりません。又その ＜pin type＞ は in でなければなりません。

（5） ＜compliance port list＞ の ＜port ID＞ は “4．8 grouped port identification” で定義されているポート名は使用できません。

４．１１ Ｉｎｓｔｒｕｃｔｉｏｎ ｒｅｇｉｓｔｅｒ ｄｅｓｃｒｉｐｔｉｏｎ 文

Instruction register description はデバイスに依存するインストラクションレジスタの特性を記述します。デバイスに実装されるインストラクションレジスタの特性を示すものとして以下のものがあります。

Length	:インストラクションレジスタは最低2ビットの長さを持ちます。最大長に制限はありません。
Instructions	:インストラクションをサポートするためには対応するレジスタが必要です。デバイスの設計者は IEEE Std. 1149．1 に規定されているオプションインストラクションの一部或いは全てを実装することが可能です。又プライベートインストラクションも実装できます。プライベートインストラクションは、試験の実行時、予期しない動作を防ぐためにアプリケーションに対してプライベートインストラクションであることを明示しなければなりません。
Instruction codes （opcodes）	:EXTEST と BYPASS インストラクションのビットパターンは IEEE Std. 1149．1 によってあらかじめ定義されています（EXTEST：All“0”, BYPASS：All“1”）。他のインストラクションのビットパターンはデバイスの設計者によって定義されなければなりません。
Instruction capture	:Capture－IR の状態遷移を通過すると、インストラクションレジスタに入力したデータがロードされます。

BSDLのインストラクションレジスタの特性とはその長さ、オペコード、Capture－IR の状態遷移でロードされるパターン、そして与えられたインストラクションがパブリックかプライベートかの情報です。

```
＜instruction register＞       ::＝  ＜instruction length stmt＞
                                     ＜instruction opcode stmt＞
                                     ＜instruction capture stmt＞
                                     [ ＜instruction private stmt＞ ]
＜instruction length stmt＞    ::＝ attribute INSTRUCTION_LENGTH of
                                    ＜component name＞ : entity is ＜integer＞;
＜instruction opcode stmt＞    ::＝ attribute INSTRUCTION_OPCODE of
                                    ＜component name＞ : entity is ＜opcode table string＞;
＜instruction capture stmt＞   ::＝ attribute INSTRUCTION_CAPTURE of
                                   ＜component name＞ : entity is ＜pattern list string＞;
＜instruction private stmt＞   ::＝ attribute INSTRUCTION_PRIVATE of
                                   ＜component name＞ : entity is ＜instruction list string＞;

＜opcode table string＞    ::＝ " ＜opcode description＞ { , ＜opcode description＞ } "
＜pattern list string＞    ::＝ " ＜pattern list＞ "
```

```
<pattern list>             ::= <pattern> { , <pattern> }
<instruction list string>  ::= " <instruction list> "
<instruction list>         ::= <instruction name> { , <instruction name> }
<opcode description>       ::= <instruction name> ( <pattern list> )
```

☆例:

```
attribute INSTRUCTION_LENGTH of xx74bct8244a : entity is 8;
attribute INSTRUCTION_OPCODE of xx74bct8244a : entity is
        "EXTEST   (00000000, 10000000), " &
        "BYPASS   (11111111, 10000100), " &
        "SAMPLE   (00000010, 10000010), " &
        "INTEST   (00000011, 10000011), " &
        "HIGHZ    (00000110, 10000110), " &  -- Bypass with outputs high-z
        "CLAMP    (00000111, 10000111), " &  -- Bypass with bs value
        "RUNT     (00001001, 10001001), " &  -- Boundary run test
        "READBN   (00001010, 10001010), " &  -- Boundary read normal mode
        "READBT   (00001011, 10001011), " &  -- Boundary read test mode
        "CELLTST  (00001100, 10001100), " &  -- Boundary selftest normal mode
        "TOPHIP   (00001101, 10001101), " &  -- Boundary toggle out test mode
        "SCANCN   (00001110, 10001110), " &  -- BCR scan normal mode
        "SCANCT   (00001111, 10001111)  " ;  -- BCR scan test mode

attribute INSTRUCTION_CAPTUREof xx74bct8244a : entity is "10000001";
attribute INSTRUCTION_DISABLE of xx74bct8244a : entity is "HIGHZ";
attribute INSTRUCTION_GUARD  of xx74bct8244a : entity is "CLAMP";
```

この例は1990年版BSDL表記です。1994年版BSDLでは INSTRUCTION_DISABLE 及び INSTRUCTION_GUARD は削除されています。

各attribute の目的を以下に示します。

INSTRUCTION_LENGTH :<instruction length stmt>はインストラクションレジスタの長さを定義します。オペコードのパターンのビット長は<instruction length stmt>で定義されたレジスタ長と一致しなければなりません。

INSTRUCTION_OPCODE :<instruction opcode stmt>はインストラクション名とそれに対応するビットパターンを定義します。ビットパターンの右端のビットが、シフトレジスタでTDoに近い(最初に出力される)ビットになります。IEEE Std. 1149. 1ではEXTEST, BYPASSインストラクションのコードの表記は省略可能です。この場合EXTESTのパターンはAll“0”, BYPASSのパターンはAll“1”となります。又他のビットパター

ンを追加で定義することも可能です。使用されないビットパターンはBYPASSインストラクションにデコードされます。

INSTRUCTION_CAPTURE : ＜instruction capture stmt＞はTAPコントローラがCapture－IRの状態遷移を通過するときに、インストラクションレジスタにロードされるビットパターンを定義します。IEEE Std. 1149. 1ではLSBの2ビットは“01”と決められています。残りのビットは設計者が定義します。

INSTRUCTION_PRIVATE :＜instruction private stmt＞はプライベートインストラクションを定義するもので、オプションです。プライベートインストラクションはデバイスの設計者によって使用されるもので、その動作は正常動作が保証されないため、通常動作ではこのインストラクションの使用は避けるべきです。

☆注意点

（1） INSTRUCTION_LENGTH の ＜integer＞ は2以上でなければなりません。
（2） ＜pattern list＞ で定義される全てのオペコードのビット長はインストラクションレジスタ長と同じでなければなりません。
（3） All“1” のパターンをもつオペコードはBYPASSに定義されるか、又はその表記は省略されていなければなりません。
（4） All“0” のパターンをもつオペコードはEXTESTに定義されるか、又はその表記は省略されていなければなりません。
（5） SAMPLE／PRELOADの為のオペコードが定義されなければなりません。そしてそのオペコードは＜opcode description＞の中でSAMPLEとして定義されなければなりません。
（6） オペコードのパターンにはXを含むことはできません。
（7） ＜instruction capture stmt＞ の ＜pattern＞ の値はインストラクションレジスタ長と等しくなければなりません。又そのLSBの2ビットは“01”でなければなりません。
（8） ユーザー定義のインストラクションのみがプライベートとして定義できます。

４．１２ Ｏｐｔｉｏｎａｌ ｒｅｇｉｓｔｅｒ ｄｅｓｃｒｉｐｔｉｏｎ 文

optional register descriptionはデバイスが実装するオプションレジスタの定義を行います。オプションレジスタには、IDCODEレジスタとUSERCODEレジスタがあります。

＜optional register description＞ ::＝「＜idcode statement＞ [＜usercode statement＞]」

＜idcode statement＞ ::＝ attribute IDCODE_REGISTER of ＜component name＞
: entity is ＜32－bit pattern list string＞;

＜usercode statement＞ ::＝ attribute USERCODE_REGISTER of ＜component name＞
: entity is ＜32－bit pattern list string＞;

＜32－bit pattern list string＞ ::＝ " ＜32－bit pattern list＞ "

＜32－bit pattern list＞ ::＝ ＜32－bit pattern＞ { , ＜32－bit pattern＞ }

☆例:

```
attribute IDCODE_REGISTER of xx74bct8244a: entity is
        "0000" &                        -- Version
        "0101000000100100" &            -- Part number
        "00000000001" &                 -- Manufacturer's ID
        "1";                            -- Required by IEEE Std. 1149.1b

attribute USERCODE_REGISTER of xx74bct8244a: entity is
                                "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX";
```

Xは未定義を示し、＜32－bit pattern list＞は、All“X”の定義も可能です。

☆注意点

（1） ＜idcode statement＞でIDCODEレジスタが定義されている場合、＜idcode statement＞の＜32－bit pattern＞のLSB1ビットは “1” でなければなりません。

４．１３ Ｒｅｇｉｓｔｅｒ ａｃｃｅｓｓ ｄｅｓｃｒｉｐｔｉｏｎ 文

全てのインストラクションはテストデータをTDIとTDOの間のレジスタに配置しなければばりません。ユーザー定義のインストラクションはIEEE Std. 1149. 1で定義されているテストデータレジスタ、或いはデバイス設計者が定義したレジスタにアクセスする可能性があります。IEEE Std. 1149. 1はデバイス設計者がデバイスにユーザー定義のレジスタとして参照できるレジスタを追加することを許しています。これらの試験に関連するインストラクションの名前やデータレジスタのビット長を定義します。

```
＜register access description＞ ::＝ attribute REGISTER_ACCESS of
                                ＜component name＞ : entity is ＜register string＞ ;

＜register string＞    ::＝ " ＜register association＞ { , ＜register association＞ } "
＜register assocoation＞    ::＝ ＜register＞ ( ＜instruction capture list＞ )
＜instruction capture list＞ ::＝ ＜instruction capture＞ { , ＜instruction capture＞ }
＜instruction capture＞     ::＝ ＜instruction name＞ [ CAPTURES ＜pattern＞ ]
＜register＞   ::＝ BOUNDARY | BYPASS | DEVICE_ID
             | ＜VHDL identifier＞ ＜ left bracket＞ ＜register length＞ ＜right bracket＞
＜register length＞         ::＝ ＜integer＞
```

☆例:

```
attribute REGISTER_ACCESS of xx74bct8244a : entity is
"BOUNDARY (EXTEST, INTEST, SAMPLE, READBN, READBT, CELLTST), " &
"BYPASS   (BYPASS, HIGHZ, CLAMP, RUNT, TOPHIP), " &
"BCR[2]   (SCANCN, SCANCT CAPTURES 0X)" ;
```

上記例において EXTEST, INTEST, SAMPLE, READBN, READBT, CELLTST, BYPASS, HIGHZ, CLAMP, RUNT, TOPHIP, SCANCN, SCANCT は“4. 11 Instruction register description”の <instruction opcode stmt> で定義されなければなりません。最初の6個のインストラクションはBoundary－Scanレジスタを選択し、次の5個のインストラクションはBypassレジスタを選択します。最後の2個のインストラクション(SCANCN, SCANCT)はBCR[2]という2ビットのレジスタを選択します。これは設計者定義のテスト用のレジスタです。この例ではSCANCTに Capture－DRの状態遷移を通過するときにBCR[2]レジスタにロードされる0Xという値を示しています。この例ではSCANCNインストラクションに対する値は定義されていません。

☆注意点

(1) BYPASS, CLAMP, EXTEST, HIGHZ, IDCODE, INTEST, SAMPLE／PRELOAD, USERCODE インストラクションはIEEE Std. 1149. 1 で定義されています。
(2) <instruction register>で定義される上記(1)のインストラクション以外のパブリックなインストラクションは関連するテストデータレジスタが定義されなければなりません。
(3) パブリックにアクセスできる設計者定義のテストデータレジスタの長さは0より大きくなくてはなりません。
(4) <instruction capture>の<instruction name>は“4. 11 Instruction register description”の<instruction opcode stmt>の<opcode description> で定義されていなければなりません。
(5) <instruction capture>の<pattern>は同じ<register assocoation>内の<register>のレジスタ長と一致しなければなりません。

4. 14 Boundary register description 文

boundary－scan register description には0から LENGTH－1 (LENGTHは“4. 11 Instruction register description”の <instruction length stmt> で定義されます)のバウンダリスキャンセルのリストが記述されます。セルの並びは順不同です。但し全てのセルが定義されていなければなりません。Cell 0がTDoに最も近いセルになります。

デバイスに使われているセル構成の特性は<boundary－scan register description>で参照される前に定義されていなければなりません。例えば、IEEE Std. 1149. 1に含まれるセル構成はSTD_1149_1_1994のスタンダードVHDLパッケージに含まれています。このVHDLパッケージのセル構成は以下の表に示される単純な名前を使って参照されます。

```
<boundary−scan register description> ::= <boundary length stmt>
                                         <boundary register stmt>

<boundary length stmt>    ::= attribute BOUNDARY_LENGTH of
                              <component name> : entity is <integer> ;
<boundary register stmt>  ::= attribute BOUNDARY_REGISTER of
                              <component name> : entity is
                              <cell table string> ;
<cell table string>  ::= " <cell table> "
<cell table>         ::= <cell entry> { , <cell entry> }
<cell entry>         ::= <cell number> ( <cell info> )
<cell number>        ::= <integer>
<cell info>          ::= <cell spec> [ , <disable spec> ]
<cell spec>          ::= <cell name> , <port ID or null> , <function>
                         , <safe bit>
<cell name>          ::= <VHDL identifier>
<port ID or null>    ::= <port ID> | *
<function>           ::= INPUT | OUTPUT2 | OUTPUT3 | CONTROL | CONTROLR
                         | INTERNAL | CLOCK | BIDIR | OBSERVE_ONLY
<safe bit>           ::= 0 | 1 | X
<disable spec>       ::= <ccell> , <disable value> , <disable result>
<ccell>              ::= <integer>
<disable value>      ::= 0 | 1
<disable result>     ::= Z | WEAK0 | WEAK1 | PULL0 | PULL1
```

☆例:

```
        attribute BOUNDARY_LENGTH    of sn74bct8244a : entity is 18;
        attribute BOUNDARY_REGISTER of sn74bct8244a : entity is

    −− num   cell    port function safe [ccell disval rslt]
        " 17 (BC_1,  OE_NEG1, input,    X), " &
        " 17 (BC_1,  *,       control,  1) " &
        " 16 (BC_1,  OE_NEG2, input,    X), " &
        " 16 (BC_1,  *,       control,  1), " &
        " 15 (BC_1,  A1(1),   input,    X), " &
        " 14 (BC_1,  A1(2),   input,    X), " &
        " 13 (BC_1,  A1(3),   input,    X), " &
        " 12 (BC_1,  A1(4),   input,    X), " &
```

```
"3  (BC_1, Y2(1), output3, X, 16,    1,   Z), " &
"2  (BC_1, Y2(2), output3, X, 16,    1,   Z), " &
"1  (BC_1, Y2(3), output3, X, 16,    1,   Z), " &
"0  (BC_1, Y2(4), output3, X, 16,    1,   Z) ;"
```

＜boundary register stmt＞は＜cell entry＞の列挙で構成される＜cell table string＞を含みます。＜cell entry＞は2つの要素（＜cell number＞と＜cell info＞）から構成されます。又その並びは順不同です。

－ ＜cell number＞は0から LENGTH－1 （LENGTHは“4．11 Instruction register description”の ＜instruction length stmt＞ で定義されます）の範囲でなければなりません。
－ ＜cell info＞は4つ又は7つの要素から構成されます。

全ての＜cell entry＞の要素は最初の4つの要素（＜cell name＞，＜port ID or null＞，＜function＞，＜safe bit＞）の値をもたなければなりません。そして出力を制御できるセル（オープンコレクタや3ステート出力）を定義した＜cell entry＞はさらに3つの要素（＜ccell＞，＜disable value＞，＜disable result＞）を持たなければなりません。＜function＞の値が OUTPUT あるいは BIDIR のときは、3つの要素（＜ccell＞，＜disable value＞，＜disable result＞）を定義しなければばりません。＜function＞の値がBIDIRで出力ドライバがインアクティブの場合、そのセルは受信動作を行います。＜function＞の値がOUTPUT2の場合、3つの要素（＜ccell＞，＜disable value＞，＜disable result＞）は定義される場合とされない場合があります。そしてそれは定義されるドライバびVHDL＜port type＞がOUT であるかどうか、又はBUFFERであるかどうかに依存します。

＜cell name＞の要素は使われているセル構成を定義します。そしてその値は、Standard VHDL Package 或いはユーザー指定のVHDL package で定義されている値でなければなりません。

＜port ID or null＞の要素はセルに対して接続される外部信号名を示します。この値は“4．3 Logical port description” で定義された名前でなければなりません。出力コントロールあるいはインターナルセルとして使用されるセルはこの要素に “＊” を用います。

＜function＞の要素は関連するセルの主要な機能を定義します。表4．14－1に＜function＞で持つことができる値を示します。

表4．14－1 function値

値	説明	pin type
INPUT	モニタ機能を持ち外部ピンに接続されます。	in または inout
CLOCK	外部クロック入力ピン似接続されます。	in
OUTPUT2	2値出力をもつ外部ピンに接続されます。	out, buffer, inout
OUTPUT3	3ステート出力をもつ外部ピンに接続されます。	out , inout
CONTROL	1つ以上の出力信号或いは双方向信号のイネーブル制御又は方向制御を行います。	－
CONTROLR	1つ以上の出力信号或いは双方向信号のイネーブル制御又は方向制御を行います。またTAPがTest－Logic－Reset 遷移状態を通るとディスエーブル状態にします。	－
INTERNAL	デバイスの内部で使用します。外部ピンには接続されません。	－
BIDIR	双方向ピンに接続されます。	inout
OBSERVE__ONLY	INPUTと同じ機能ですが、INTESTをサポートしません。セルのモニタ信号として使用されます。	in または inout

＜safe bit＞の要素は、バウンダリスキャン試験パターン生成ソフトウェアがランダムに値を選択する時、セルのCAP FFにロードされるべき値を定義します。例えば以下のような場合が考えられます。

・関連付けされたドライバを無効にするようなContorl Cell の値
・INTEST実行時、ドライバに流れる電流値を最小にするような出力信号の値
・EXTEST実行時、内部論理回路に影響を与えないようにする入力信号の値

＜ccell＞の要素は，対応する＜port ID＞の出力をディスエーブルにするコントロールセルの＜cell number＞を示します。

＜disable value＞の要素は、対応する＜port ID＞の出力をディスエーブルにするための＜ccell＞の値を示します。

＜disable result＞の要素は、対応する＜port ID＞の出力がディスエーブルになった場合の出力ドライバの状態を示します。設定できる値には以下のものがあります。

ハイ・インピーダンス（Hi－Z）状態 :Z
外部プルダウンによるWeak“0”状態 :WEAK0
外部プルダアップよるWeak“1”状態 :WEAK1
内部プルダウンによるWeak“0”状態 :PULL0
内部プルダアップよるWeak“1”状態 :PULL1

☆注意点

（1） ＜boundary length stmt＞の＜integer＞で示されるバウンダリスキャンレジスタ長（LENGTH）は1以上でなければなりません。
（2） ＜cell entry＞ の ＜cell number＞ は 0 から LENGTH－1 の範囲でないといけません。
（3） ＜cell table＞ の中に 同じ値の ＜cell number＞ を持つ ＜cell entry＞（merged cellと呼ばれます） が存在する場合があります。この場合以下の条件を満たさないといけません。
　－ ＜function＞が INPUT の要素の cell と＜function＞がOUTPUT2, OUTPUT3, CONTROL, CONTROLR の要素のcell のみがマージ可能です。
　－ 両方の ＜cell entry＞ の ＜cell name＞ は同じ名前を持ちます。
　－ 2つの cell の ＜safe bit＞ の値は少なくともどちらかが X でないといけません。
（4） ＜cell table＞ の中の ＜cell name＞ はStandard VHDL Package 或いはユーザー指定のVHDL package で定義されている値でなければなりません。
（5） ＜subscripted port name＞ を持たない ＜cell spec＞ の ＜port name＞ の値は＜logical port description＞ の BIT で定義された ＜pin spec＞ で定義されていなければなりません。
（6） ＜cell spec＞ の ＜function＞ 要素が CONTROL , CONTROLR , INTERNAL の時は、＜port ID or null＞ は “＊” でなければなりません。
（7） INPUT , CONTROL , CONTROLR , INTERNAL , OBSERVE__ONLY , CLOCK と等しい＜function＞ の要素を持つ ＜cell info＞ は ＜disable spec＞ の要素を持つ事ができません。
（8） OUTPUT3 , BIDIR と等しい ＜function＞ の要素を持つ ＜cell info＞ は ＜disable spec＞ の定義を持たなければなりません。
（9） OUTPUT2 と等しい ＜function＞ の要素を持ち、＜disable spec＞ の要素をもつ ＜cell info＞ は ＜cell number＞ の値が ＜ccell＞ の値と等しくなければなりません。
（10） CONTROL又はCONTROLR と等しい ＜function＞ の要素を持つ、＜cell entry＞ の ＜cell number＞ は ＜disable spec＞ の ＜ccell＞ に使用されなければなりません。
（11） ＜VHDL identifiier＞ が ＜logical port description＞ の ＜pin spec＞ の ＜identifiier list＞ に使用され、かつ ＜pin spec＞ 内の ＜pin type＞ が linkagae の値をとるとき、＜VHDL identifiier＞は＜boundary register stmt＞ の ＜cell entry＞ の ＜port ID＞ としては使用されません。
（12） ＜boundary register stmt＞ の ＜port ID＞ に対応する ＜pin spec＞ の値によってとるべき ＜function＞ の値は以下の通りとなります。

pin type	function
in	INPUT , CLOCK , OBSERVE__ONLY
out	OUTPUT2 , OUTPUT3
buffer	OUTPUT2
inout	BIDIR , OUTPUT2 , OUTPUT3 , INPUT , OBSERVE__ONLY

４．１５ ＲＵＮＢＩＳＴ ｄｅｓｃｒｉｐｔｉｏｎ 文

RUNBIST descriptionはIEEE Std. 1149. 1 で定義されるRUNBIST インストラクションに関する定義を行います。IEEE Std. 1149. 1で定義されるRUNBIST インストラクションの概要に関することだけです。BIST(Build－In Self－Test)に関する詳細な内容に関しては他の規格等を参照してください。

```
<runbist description> ::= attribute RUNBIST_EXECUTION of <component name> :
                                                entity is " <runbist spec> ";
<runbist spec>        ::= <wait spec> , <pin spec> , <signature spec>
<wait spec>           ::= WAIT_DURATION( <duration spec> )
<duration spec>       ::= <clock cycles list> | <time> [ , <clock cycles list> ]
<clock cycles list>   ::= <clock cycles> { , <clock cycles> }
<time>                ::= <real number>
<clock cycles>        ::= <port ID> <integer>
<pin spec>            ::= OBSERVING <condition> AT_PINS
<condition>           ::= HIGHZ | BOUNDARY
<signature spec>      ::= EXPECT_DATA <det pattern>
<det pattern>         ::= <bit> { <bit> }
<bit>                 ::= 0 | 1
```

<det pattern> は1つ以上の連続した0または1の並びでなければなりません。その間にはスペースや区切り記号等を含んではいけません。<time>要素の値で定義されるTimeは秒を示しています。TimeとClockサイクルの両方が定義されると、最大時間またはクロックサイクル数に応じた必要な時間が定義されていると解釈されます。

☆例:

```
attribute RUNBIST_EXECUTION of xxCPU : entity is
        " Wait_Duration (1. 0e－3), " &
        " Observing HIGHZ At_Pins, " &
        " Expect_Data 0 ";
```

☆注意点

(1) 1990年バージョンに対応したBSDLでは、RUNBIST description 文をサポートしていません。
(2) <signature spec> の <det pattern> の値のビットの数は <register access description> の <register> 中の <integer> の値として定義されるビットの数と等しくなければなりません。そしてその時、RUNBIST の値が <register access description> の <instruction name> に定義されます。

(3) ＜wait spec＞ の ＜port ID＞ の値は “4. 9 Scan Port identification” の ＜TCK stmt＞ の中の ＜port ID＞ で定義されるか、或いは、“4. 14 Boundary Scan register description” の ＜function＞ がCLOCK で定義される ＜cell spec＞ の＜port ID＞で定義されなければなりません。
(4) ＜runbist description＞ が BSDL の中で定義される時、＜instruction opcode stmt＞ の ＜opcode table＞ のいくつかの ＜instruction name＞ に RUNBIST の値が定義されなければなりません。
(5) ＜time＞ と ＜clock cycles＞ の値は、0より大きくなければなりません。

４．１６ ＩＮＴＥＳＴ ｄｅｓｃｒｉｐｔｉｏｎ 文

INTEST description 文は デバイスの INTEST を実行する際、設定するテストパターンの条件と、INTEST を実行している間の、デバイスの外部状態を定義します。

INTESTのためのテストパターンは、IEEE Std. 1149. 1b では定義されません。そのパターンは別の方法で提供されます。

```
＜intest description＞          ::＝ attribute INTEST_EXECUTION of
                                        ＜component name＞ entity is
                                        " ＜intest execution sequence＞ ";
＜intest execution sequence＞ ::＝ ＜wait spec＞ , ＜pin spec＞
＜wait spec＞                  ::＝ WAIT_DURATION( ＜duration spec＞ )
＜duration spec＞              ::＝ ＜clock cycles list＞ | ＜time＞
                                    [ , ＜clock cycles list＞ ]
＜clock cycles list＞          ::＝ ＜clock cycles＞ { , ＜clock cycles＞ }
＜time＞                       ::＝ ＜real number＞
＜clock cycles＞               ::＝ ＜port ID＞ ＜integer＞
＜pin spec＞                   ::＝ OBSERVING ＜condition＞ AT_PINS
```

☆例:

```
attribute INTEST_EXECUTION of xxCPU : entity is
        "Wait_Duration (1. 0e−3), " &
        "Observing HIGHZ At_Pins ";
```

☆注意点

(1) 1990年バージョンに対応したBSDLでは、INTEST description 文をサポートしていません。
(2) ＜wait spec＞ の ＜port ID＞ の値は “4. 9 Scan Port identification” の ＜TCK stmt＞ の中の ＜port ID＞ で定義されるか、或いは、“4. 14 Boundary Scan register description” の ＜function＞ がCLOCK で定義される ＜cell spec＞ の＜port ID＞で定義されなければなりません。

（3） ＜intest description＞ がBSDLの中で定義される時、 ＜instruction opcode stmt＞ の ＜opcode table＞ のいくつかの ＜instruction name＞ に INTEST の値が定義されなければなりません。
（4） ＜time＞ と ＜clock cycles＞ の値は、0より大きくなければなりません。

４．１７ Ｕｓｅｒ ｅｘｔｅｎｓｉｏｎｓ ｔｏ ＢＳＤＬ

オプションのBSDL extensions は一般的なBSDL定義文に互換性を保ちながら必要となるBSDLの拡張を行うことができます。VHDLの標準Package STD__1149__1__1994 はVHDL のサブタイプであるBSDL__EXTENSION を定義することができます。これはユーザーが “BSDL extensions” として他の属性を定義することを許しています。しかし、これらはBSDLパーサー（解析プログラム）によって無視されます。BSDL extensions は“4．1 BSDLの構造について” で示されるように DESIGN__WARNING 文の前に記述される必要があります。

＜BSDL extensioins＞ ::= ＜BSDL extension＞ { ＜BSDL extension＞ }
＜BSDL extensioin＞ ::= ＜extension declaration＞ | ＜extension definition＞
＜extensioin declaration＞ ::= attribute ＜extension name＞ : BSDL__EXTENSION;
＜extensioin definition＞ ::= attribute ＜extension name＞ of ＜component name＞ :
entity is ＜extension parameter string＞;
＜extensioin name＞ ::= ＜entity defined name＞
| ＜VHDL package defined name＞
＜entity defined name＞ ::= ＜VHDL identifier＞
＜VHDL package defined name＞ ::= ＜VHDL identifier＞
＜extention parameter string＞ ::= ＜string＞

＜extensioin definition＞ は、対応する ＜extensioin declaration＞ の後に記述されなければなりません。＜extensioin declaration＞ は同じBSDLに記述されるか、ユーザーが定義したVHDL Package に記述します。

☆例:

```
attribute Extension__Sample : BSDL__EXTENSION;
attribute Extension__Sample of example : entity is “My__define”;
```

☆注意点

（1） ＜extensioin definition＞ の ＜extensioin name＞ で定義される ＜VHDL identifier＞ は、それより前の行で定義されている ＜extensioin declaration＞ の ＜extensioin name＞ 又は ＜extensioin declaration＞ より前で定義されているVHDLパッケージで定義されなければなりません。
（2） ＜extensioin declaration＞ の ＜extensioin name＞ はユニークな名前でなければなりません。

４．１８ Ｄｅｓｉｇｎ Ｗａｒｎｉｎｇ 文

Design Warning 文は バウンダリスキャンの特性によってシステムの回路に問題を発生させる可能性がある場合、ユーザーに知らせることを目的としています。例えば、デバイスがその状態を保持するためのクロックが必要なダイナミックな回路を持っている場合、デバイスがシステムモードから抜けてINTESTのためのテストモードになった場合でもそのクロックは保持されなければなりません。Design Warning は問題となる可能性を警告としてユーザーに知らせることができます。

＜design warning＞ ::= attribute DESIGN_WARNING of ＜component name＞ :
entity is ＜string＞;

☆例:

```
attribute DESIGN_WARNING of EPM7032SL44 : xxxxLSI is
        " Some pins have both controlled and uncontrolled input paths. " &
        " Test pin ignore for intest. " ;
```

5. Standard VHDL Package

1990年版 Standard VHDL Package を以下に示します。

```
-- STD_1149_1_1990

package STD_1149_1_1990 is

-- Give pin mapping declarations
attribute PIN_MAP : string;
subtype PIN_MAP_STRING is string;

-- Give TAP control declarations
type CLOCK_LEVEL is (LOW, BOTH);
type CLOCK_INFO is record
        FREQ : real;
        LEVEL: CLOCK_LEVEL;
end record;

attribute TAP_SCAN_IN          : boolean;
attribute TAP_SCAN_OUT         : boolean;
attribute TAP_SCAN_CLOCK       : CLOCK_INFO;
attribute TAP_SCAN_MODE        : boolean;
attribute TAP_SCAN_RESET       : boolean;

-- Give instruction register declarations
attribute INSTRUCTION_LENGTH     : integer;
attribute INSTRUCTION_OPCODE     : string;
attribute INSTRUCTION_CAPTURE    : string;
attribute INSTRUCTION_DISABLE    : string;
attribute INSTRUCTION_GUARD      : string;
attribute INSTRUCTION_PRIVATE    : string;
attribute INSTRUCTION_USAGE      : string;
attribute INSTRUCTION_SEQUENCE : string;

-- Give ID and USER code declarations
type ID_BITS is ('0', '1', 'x', 'X');
type ID_STRING is array (31 dounto 0) of ID_BITS;
attribute IDCODE_REGISTER     : ID_STRING;
attribute USERCODE_REGISTER: ID_STRING;
```

```
-- Give register declarations
attribute REGISTER_ACCESS : string;

-- Give boundary cell declarations
type BSCAN_INST is (EXTEST, SAMPLE, INTEST, RUNBIST);
type CELL_TYPE is (INPUT, INTERNAL, CLOCK, CONTROL, CONTROLR,
                                   OUTPUT2, OUTPUT3, BIDIR_IN, BIDIR_OUT);
type CAP_DATA is (P1, P0, UPD, CAP, X, ZERO, ONE);

type CELL_DATA is record
        CT   : CELL_TYPE;
        I    : BSCAN_INST;
        CD   : CAP_DATA;
end record;

type CELL_INFO is array (positive range <>) of CELL_DATA;

-- Boundary cell deferred constants (see package body)
constant BC_1 : CELL_INFO;
constant BC_2 : CELL_INFO;
constant BC_3 : CELL_INFO;
constant BC_4 : CELL_INFO;
constant BC_5 : CELL_INFO;
constant BC_6 : CELL_INFO;

-- Boundary register declarations
attribute BOUNDARY_CELLS    : string;
attribute BOUNDARY_LENGTH   : integer;
attribute BOUNDARY_REGISTER: string;

-- Miscellaneous
attribute DESIGN_WARNING : string;

end STD_1149_1_1990;      -- End of IEEE Std 1149.1-1990 Package


package body STD_1149_1_1990 is -- Standard boundary cells

-- Description for f10-12, f10-16, flO-18c, f10-18d, f10-21c
constant BC_I : CELL_INFO :=
        ( (INPUT, EXTEST, P1), (OUTPUT2, EXTEST, PI),
```

```
          (INPUT, SAMPLE, P1), (OUTPUT2, SAMPLE, P1),
          (INPUT, INTEST, P1), (OUTPUT2, INTEST, PI),
          (INPUT, RUNBIST, PI), (OUTPUT2, RUNBIST, PI),
          (OUTPUT3, EXTEST, PI), (INTERNAL, EXTEST, PI),
          (OUTPUT3, SAMPLE, PI), (INTERNAL, SAMPLE, PI),
          (OUTPUT3, INTEST, PI), (INTERNAL, INTEST, PI),
          (OUTPUT3, RUNBIST, PI), (INTERNAL, RUNBIST, PI),
          (CONTROL, EXTEST, PI), (CONTROLR, EXTEST, PI),
          (CONTROL, SAMPLE, PI), (CONTROLR, SAMPLE, PI),
          (CONTROL, INTEST, PI), (CONTROLR, INTEST, PI),
          (CONTROL, RUNBIST, PI), (CONTROLR, RUNBIST, P1) );

--- Description for f10-8, f10-17, f10-19c, f10-19d, f10-22c
constant BC_2 : CELL_INFO :
        ( (INPUT, EXTEST, PI)       , (OUTPUT2, EXTEST, UPD),
          (INPUT, SAMPLE, PI)       , (OUTPUT2, SAMPLE, PI),
          (INPUT, INTEST, UPD), -- Intest on output2 not supported
          (INPUT, RUNBIST, UPD)     , (OUTPUT2, RUNBIST, UPD),
          (OUTPUT3, EXTEST, UPD)    , (INTERNAL, EXTZST, PI),
          (OUTPUT3, SAMPLE, PI)     , (INTERNAL, SAMPLE, PI),
          (OUTPUT3, INTEST, PI)     , (INTERNAL, INTEST, UPD),
          (OUTPUT3, RUNBIST, PI)    , (INTERNAL, RUNBIST, UPD),
          (CONTROL, EXTEST, UPD)    , (CONTROLR, EXTEST, UPD),
          (CONTROL, SAMPLE, PI)     , (CONTROLR, SAMPLE, PI),
          (CONTROL, INTEST, PI)     , (CONTROLR, INTEST, PI),
          (CONTROL, RUNBIST, PI)    , (CONTROLR, RUNBIST, P1) );

--- Description for f10-9
constant BC_3 : CELL_INFO :=
        ( (INPUT, EXTEST, PI)       , (INTERNAL, EXTEST, PI),
          (INPUT, SAMPLE, PI)       , (INTERNAL, SAMPLE, PI),
          (INPUT, INTEST, PI)       , (INTERNAL, INTEST, PI),
          (INPUT, RUNBIST, PI)      , (INTERNAL, RUNBIST, PI) );

--- Description for f10-10, f10-11
constant BC_4 : CELL_INFO :=
        ( (INPUT, EXTEST, PI), -- Intest on input not supported
          (INPUT, SAMPLE, PI), -- Runbist on input not supported
          (CLOCK, EXTEST, PI)       , (INTERNAL, EXTEST, PI),
          (CLOCK, SAMPLE, PI)       , (INTERNAL, SAMPLE, PI),
          (CLOCK, INTEST, PI)       , (INTERNAL, INTEST, PI),
          (CLOCK, RUNBIST, PI)      , (INTERNAL, RUNBIST, PI) );
```

```
--- Description for f10-20c, a combined input/control
constant BC_5 : CELL_INFO :=
        ( (INPUT, EXTEST, PI)       , (CONTROL, EXTEST, PI),
          (INPUT, SAMPLE, PI)       , (CONTROL, SAMPLE, PI),
          (INPUT, INTEST, UPD)      , (CONTROL, INTEST, UPD),
          (INPUT, RUNBIST, PI)      , (CONTROL, RUNBIST, PI) );

--- Description for f10-22d, a reversible cell
constant BC_6 : CELL_INFO :=
        ( (BIDIR_IN, EXTEST, PI)    , (BIDIR_OUT, EXTEST, UPD),
          (BIDIR_IN, SAMPLE, PI)    , (BIDIR_OUT, SAMPLE, PI),
          (BIDIR_IN, INTEST, UPD)   , (BIDIR_OUT, INTEST, PI),
          (BIDIR_IN, RUNBIST, UPD), (BIDIR_OUT, RUNBIST, PI) );

end STD_1149_1_1990; -- End of 1990 Package Body
```

1994年版 Standard VHDL Package を以下に示します。

```
package STD_1149_1_1994 is

-- Give component conformance declaration
attribute COMPONENT_CONFORMANCE : string;

-- Give pin mapping declarations
attribute PIN_MAP : string;
subtype PIN_MAP_STRING is string;

-- Give TAP control declarations
type CLOCK_LEVEL is (LOW, BOTH);

type CLOCK_INFO is record
        FREQ  : real;
        LEVEL : CLOCK_LEVEL;
end record;

attribute TAP_SCAN_IN        : boolean;
attribute TAP_SCAN_OUT       : boolean;
attribute TAP_SCAN_CLOCK     : CLOCK_INFO;
attribute TAP_SCAN_MODE      : boolean;
attribute TAP_SCAN_RESET     : boolean;

-- Give instruction register declarations
attribute INSTRUCTION_LENGTH    : integer;
attribute INSTRUCTION_OPCODE    : string;
attribute INSTRUCTION_CAPTURE   : string;
attribute INSTRUCTION_PRIVATE   : string;

-- Give ID and USER code declarations
type ID_BITS is ('0', '1', 'x', 'X');
type ID_STRING is array (31 downto 0) of ID_BITS;
attribute IDCODE_REGISTER    : ID_STRING;
attribute USERCODE_REGISTER: ID_STRING;

-- Give register declarations
attribute REGISTER_ACCESS : String;
```

```
-- Give boundary cell declarations
type BSCAN_INST is (EXTEST, SAMPLE, INTEST);
type CELL_TYPE is (INPUT, INTERNAL, CLOCK, OBSERVE_ONLY, CONTROL,
                   CONTROLR, OUTPUT2, OUTPUT3, BIDIR_IN, BIDIR_OUT);
type CAP_DATA is (PI, PO, UPD, CAP, X, ZERO, ONE);

type CELL_DATA is record
        CT      : CELL_TYPE;
        I       : BSCAN_INST;
        CD      : CAP_DATA;
end record;

type CELL_INFO is array (positive range <>) of CELL_DATA;

-- Boundary cell deferred constants (see package body)
constant BC_0 : CELL_INFO;
constant BC_1 : CELL_INFO;
constant BC_2 : CELL_INFO;
constant BC_3 : CELL_INFO;
constant BC_4 : CELL_INFO;
constant BC_5 : CELL_INFO;
constant BC_6 : CELL_INFO;
constant BC_7 : CELL_INFO;

-- Boundary register declarations
attribute BOUNDARY_LENGTH  : integer;
attribute BOUNDARY_REGISTER: string;

-- Miscellaneous
attribute PORT_GROUPING      : string;
attribute RUNBIST_EXECUTION  : string;
attribute INTEST_EXECUTION   : string;

subtype BSDL_EXTENSION is string;
attribute COMPLIANCE_PATTERNS : string;
attribute DESIGN_WARNING : string;

end STD_1149_1_1994; -- End of 1149.1-1994 Package


package body STD_1149_1_1994 is -- Standard boundary cells
```

```
-- Generic cell capturing minimum allowed data
constant BC_0 : CELL_INFO :=
        ( (INPUT, EXTEST, PI)        , (OUTPUT2, EXTEST, X),
          (INPUT, SAMPLE, PI)        , (OUTPUT2, SAMPLE, PI),
          (INPUT, INTEST, X)         , (OUTPUT2, INTEST, PI),
          (OUTPUT3, EXTEST, X)       , (INTERNAL, EXTEST, X),
          (OUTPUT3, SAMPLE, PI)      , (INTERNAL, SAMPLE, X),
          (OUTPUT3, INTEST, PI)      , (INTERNAL, INTEST, X),
          (CONTROL, EXTEST, X)       , (CONTROLR, EXTEST, X),
          (CONTROL, SAMPLE, PI)      , (CONTROLR, SAMPLE, PI),
          (CONTROL, INTEST, PI)      , (CONTROLR, INTEST, PI),
          (BIDIR_IN, EXTEST, PI)     , (BIDIR_OUT, EXTEST, X),
          (BIDIR_IN, SAMPLE, PI)     , (BIDIR_OUT, SAMPLE, PI),
          (BIDIR_IN, INTEST, X)      , (BIDIR_OUT, INTEST, PI),
          (OBSERVE_ONLY, SAMPLE, PI), (OBSERVE_ONLY, EXTEST, PI) );

-- Description for f10-18, f10-29, f10-31c, f10-31d, f10-33c, f10-41d
constant BC_1 : CELL_INFO :=
        ( (INPUT, EXTEST, PI)        , (OUTPUT2, EXTEST, PI),
          (INPUT, SAMPLE, PI)        , (OUTPUT2, SAMPLE, PI),
          (INPUT, INTEST, PI)        , (OUTPUT2, INTEST, PI),
          (OUTPUT3, EXTEST, PI)      , (INTERNAL, EXTEST, PI),
          (OUTPUT3, SAMPLE, PI)      , (INTERNAL, SAMPLE, PI),
          (OUTPUT3, INTEST, PI)      , (INTERNAL, INTEST, PI),
          (CONTROL, EXTEST, PI)      , (CONTROLR, EXTEST, PI),
          (CONTROL, SAMPLE, PI)      , (CONTROLR, SAMPLE, PI),
          (CONTROL, INTEST, PI)      , (CONTROLR, INTEST, PI) );

-- Description for f10-14, f10-30, f10-32c, f10-32d, f10-35c
constant BC_2 : CELL_INFO :=
        ( (INPUT, EXTEST, PI)        , (OUTPUT2, EXTEST, UPD),
          (INPUT, SAMPLE, PI)        , (OUTPUT2, SAMPLE, PI),
          (INPUT, INTEST, UPD)       , -- Intest on output2 not supported
          (OUTPUT3, EXTEST, UPD)     , (INTERNAL, EXTEST, PI),
          (OUTPUT3, SAMPLE, PI)      , (INTERNAL, SAMPLE, PI),
          (OUTPUT3, INTEST, PI)      , (INTERNAL, INTEST, UPD),
          (CONTROL, EXTEST, UPD)     , (CONTROLR, EXTEST, UPD),
          (CONTROL, SAMPLE, PI)      , (CONTROLR, SAMPLE, PI),
          (CONTROL, INTEST, PI)      , (CONTROLR, INTEST, PI) );

-- Description for f10-15
constant BC_3 : CELL_INFO :=
```

```
        ( (INPUT,  EXTEST,  PI)        , (INTERNAL,  EXTEST,  PI),
          (INPUT,  SAMPLE,  PI)        , (INTERNAL,  SAMPLE,  PI),
          (INPUT,  INTEST,  PI)        , (INTERNAL,  INTEST,  PI) );

-- Description for f10-16, f10-17
constant BC_4 : CELL_INFO :=
        ( (INPUT,  EXTEST,  PI),  -- Intest on input not supported
          (INPUT,  SAMPLE,  PI),
          (OBSERVE_ONLY,  EXTEST,  PI),
          (OBSERVE_ONLY,  SAMPLE,  PI),
                          -- Intest on observe_only not supported
          (CLOCK,  EXTEST,  PI)        , (INTERNAL,  EXTEST,  PI),
          (CLOCK,  SAMPLE,  PI)        , (INTERNAL,  SAMPLE,  PI),
          (CLOCK,  INTEST,  PI)        , (INTERNAL,  INTEST,  PI) );

-- Description for f10-41c, a combined input/control
constant BC_5 : CELL_INFO :=
        ( (INPUT,  EXTEST,  PI)        , (CONTROL,  EXTEST,  PI),
          (INPUT,  SAMPLE,  PI)        , (CONTROL,  SAMPLE,  PI),
          (INPUT,  INTEST,  UPD)       , (CONTROL,  INTEST,  UPD) );

-- Description for f10-35d, a reversible cell
-- ! ! Not recommended; replaced by BC_7 below ! !
constant BC_6 : CELL_INFO :=
        ( (BIDIR_IN,  EXTEST,  PI)     , (BIDIR_OUT,  EXTEST,  UPD),
          (BIDIR_IN,  SAMPLE,  PI)     , (BIDIR_OUT,  SAMPLE,  PI),
          (BIDIR_IN,  INTEST,  UPD)    , (BIDIR_OUT,  INTEST,  PI) );

-- Description for f10-34d, self monitor reversible
-- ! ! Recommended over cell BC_6 ! !
constant BC_7 : CELL_INFO :=
        ( (BIDIR_IN,  EXTEST,  PI)     , (BIDIR_OUT,  EXTEST,  PO),
          (BIDIR_IN,  SAMPLE,  PI)     , (BIDIR_OUT,  SAMPLE,  PI),
          (BIDIR_IN,  INTEST,  UPD)    , (BIDIR_OUT,  INTEST,  PI) );

end STD_1149_1_1994; -- End of IEEE Std 1149.1-1994 Package Body
```

2001年版 Standard VHDL Package を以下に示します。

```
package STD__1149__1__2001 is
－－ Give component conformance declaration
attribute COMPONENT__CONFORMANCE : string;
－－ Give pin mapping declarations
attribute PIN__MAP : string;
subtype PIN__MAP__STRING is string;
－－ Give TAP control declarations
type CLOCK__LEVEL is (LOW, BOTH);
type CLOCK__INFO is record
FREQ : real;
LEVEL: CLOCK__LEVEL;
end record;
attribute TAP__SCAN__IN : boolean;
attribute TAP__SCAN__OUT : boolean;
attribute TAP__SCAN__CLOCK: CLOCK__INFO;
attribute TAP__SCAN__MODE : boolean;
attribute TAP__SCAN__RESET: boolean;
－－ Give instruction register declarations
attribute INSTRUCTION__LENGTH : integer;
attribute INSTRUCTION__OPCODE : string;
attribute INSTRUCTION__CAPTURE : string;
attribute INSTRUCTION__PRIVATE : string;
－－ Give ID and USER code declarations
type ID__BITS is ('0', '1', 'x', 'X');
type ID__STRING is array (31 downto 0) of ID__BITS;
attribute IDCODE__REGISTER : ID__STRING;
attribute USERCODE__REGISTER: ID__STRING;
－－ Give register declarations
attribute REGISTER__ACCESS : string;
－－ Give boundary cell declarations
type BSCAN__INST is (EXTEST, SAMPLE, INTEST);
type CELL__TYPE is (INPUT, INTERNAL, CLOCK, OBSERVE__ONLY,
CONTROL, CONTROLR, OUTPUT2,
OUTPUT3, BIDIR__IN, BIDIR__OUT);
type CAP__DATA is (PI, PO, UPD, CAP, X, ZERO, ONE);
type CELL__DATA is record
CT : CELL__TYPE;
I : BSCAN__INST;
CD : CAP__DATA;
```

```
end record;
type CELL_INFO is array (positive range <>) of CELL_DATA;
-- Boundary cell deferred constants (see package body)
constant BC_0 : CELL_INFO;
constant BC_1 : CELL_INFO;
constant BC_2 : CELL_INFO;
constant BC_3 : CELL_INFO;
constant BC_4 : CELL_INFO;
constant BC_5 : CELL_INFO;
constant BC_6 : CELL_INFO;
constant BC_7 : CELL_INFO;
constant BC_8 : CELL_INFO;
constant BC_9 : CELL_INFO;
constant BC_10 : CELL_INFO;
-- Boundary register declarations
attribute BOUNDARY_LENGTH : integer;
attribute BOUNDARY_REGISTER : string;
-- Miscellaneous
attribute PORT_GROUPING : string;
attribute RUNBIST_EXECUTION : string;
attribute INTEST_EXECUTION : string;
subtype BSDL_EXTENSION is string;
attribute COMPLIANCE_PATTERNS : string;
attribute DESIGN_WARNING : string;
end STD_1149_1_2001; -- End of 1149.1-2001 Package
package body STD_1149_1_2001 is -- Standard boundary cells
-- Generic cell capturing minimum allowed data
constant BC_0 : CELL_INFO :=
((INPUT, EXTEST, PI), (OUTPUT2, EXTEST, X),
(INPUT, SAMPLE, PI), (OUTPUT2, SAMPLE, PI),
(INPUT, INTEST, X), (OUTPUT2, INTEST, PI),
(OUTPUT3, EXTEST, X), (INTERNAL, EXTEST, X),
(OUTPUT3, SAMPLE, PI), (INTERNAL, SAMPLE, X),
(OUTPUT3, INTEST, PI), (INTERNAL, INTEST, X),
(CONTROL, EXTEST, X), (CONTROLR, EXTEST, X),
(CONTROL, SAMPLE, PI), (CONTROLR, SAMPLE, PI),
(CONTROL, INTEST, PI), (CONTROLR, INTEST, PI),
(BIDIR_IN, EXTEST, PI), (BIDIR_OUT, EXTEST, X ),
(BIDIR_IN, SAMPLE, PI), (BIDIR_OUT, SAMPLE, PI),
(BIDIR_IN, INTEST, X ), (BIDIR_OUT, INTEST, PI),
(OBSERVE_ONLY, SAMPLE, PI), (OBSERVE_ONLY, EXTEST, PI) );
-- Description for f11-18, f11-30, f11-34c, f11-34d, f11-36c, f11-46d
```

```
constant BC_1 : CELL_INFO :=
((INPUT, EXTEST, PI), (OUTPUT2, EXTEST, PI),
(INPUT, SAMPLE, PI), (OUTPUT2, SAMPLE, PI),
(INPUT, INTEST, PI), (OUTPUT2, INTEST, PI),
(OUTPUT3, EXTEST, PI), (INTERNAL, EXTEST, PI),
(OUTPUT3, SAMPLE, PI), (INTERNAL, SAMPLE, PI),
(OUTPUT3, INTEST, PI), (INTERNAL, INTEST, PI),
(CONTROL, EXTEST, PI), (CONTROLR, EXTEST, PI),
(CONTROL, SAMPLE, PI), (CONTROLR, SAMPLE, PI),
(CONTROL, INTEST, PI), (CONTROLR, INTEST, PI) );
-- Description for f11-14, f11-31, f11-35c, f11-35d, f11-37c,
-- f11-38c, f11-39(output) and f11-41c
constant BC_2 : CELL_INFO :=
((INPUT, EXTEST, PI), (OUTPUT2, EXTEST, UPD),
(INPUT, SAMPLE, PI), (OUTPUT2, SAMPLE, PI),
(INPUT, INTEST, UPD), -- Intest on output2 not supported
(OUTPUT3, EXTEST, UPD), (INTERNAL, EXTEST, PI),
(OUTPUT3, SAMPLE, PI), (INTERNAL, SAMPLE, PI),
(OUTPUT3, INTEST, PI), (INTERNAL, INTEST, UPD),
(CONTROL, EXTEST, UPD), (CONTROLR, EXTEST, UPD),
(CONTROL, SAMPLE, PI), (CONTROLR, SAMPLE, PI),
(CONTROL, INTEST, PI), (CONTROLR, INTEST, PI) );
-- Description for f11-15
constant BC_3 : CELL_INFO :=
((INPUT, EXTEST, PI), (INTERNAL, EXTEST, PI),
(INPUT, SAMPLE, PI), (INTERNAL, SAMPLE, PI),
(INPUT, INTEST, PI), (INTERNAL, INTEST, PI) );
-- Description for f11-16, f11-17, f11-39(input)
constant BC_4 : CELL_INFO :=
((INPUT, EXTEST, PI), -- Intest on input not supported
(INPUT, SAMPLE, PI),
(OBSERVE_ONLY, EXTEST, PI),
(OBSERVE_ONLY, SAMPLE, PI), -- Intest on observe_only not supported
(CLOCK, EXTEST, PI), (INTERNAL, EXTEST, PI),
(CLOCK, SAMPLE, PI), (INTERNAL, SAMPLE, PI),
(CLOCK, INTEST, PI), (INTERNAL, INTEST, PI) );
-- Description for f11-46c, a combined input/control
constant BC_5 : CELL_INFO :=
((INPUT, EXTEST, PI), (CONTROL, EXTEST, PI),
(INPUT, SAMPLE, PI), (CONTROL, SAMPLE, PI),
(INPUT, INTEST, UPD), (CONTROL, INTEST, UPD) );
-- Description for f11-38d, a reversible cell
```

```
-- ！！ Not recommended; replaced by BC_7 below ！！
constant BC_6 : CELL_INFO :=
((BIDIR_IN, EXTEST, PI), (BIDIR_OUT, EXTEST, UPD),
(BIDIR_IN, SAMPLE, PI), (BIDIR_OUT, SAMPLE, PI),
(BIDIR_IN, INTEST, UPD), (BIDIR_OUT, INTEST, PI) );
-- Description for f11－37d, self monitor reversible
-- ！！ Recommended over cell BC_6 ！！
constant BC_7 : CELL_INFO :=
((BIDIR_IN, EXTEST, PI), (BIDIR_OUT, EXTEST, PO),
(BIDIR_IN, SAMPLE, PI), (BIDIR_OUT, SAMPLE, PI),
(BIDIR_IN, INTEST, UPD), (BIDIR_OUT, INTEST, PI) );
-- Description for 11－40, f11－41d
constant BC_8 : CELL_INFO :=
-- Intest on bidir not supported
((BIDIR_IN, EXTEST, PI), (BIDIR_OUT, EXTEST, PO),
(BIDIR_IN, SAMPLE, PI), (BIDIR_OUT, SAMPLE, PO) );
-- Description for f11－32
constant BC_9 : CELL_INFO :=
-- Self－monitoring output that supports Intest
((OUTPUT2, EXTEST, PO), (OUTPUT3, EXTEST, PO),
(OUTPUT2, SAMPLE, PI), (OUTPUT3, SAMPLE, PI),
(OUTPUT2, INTEST, PI), (OUTPUT3, INTEST, PI) );
-- Description for f11－33
constant BC_10 : CELL_INFO :=
-- Self－monitoring output that does not support Intest
((OUTPUT2, EXTEST, PO), (OUTPUT3, EXTEST, PO),
(OUTPUT2, SAMPLE, PO), (OUTPUT3, SAMPLE, PO) );
end STD_1149_1_2001; -- End of IEEE Std 1149.1－2001 Package Body
```

６．１９９０年版ＢＳＤＬと１９９４年版ＢＳＤＬとの差

1994年版BSDLでは以下の属性が廃止されました。

INSTRUCTION_GUARD
INSTRUCTION_DISABLE
BOUNDARY_CELLS
INSTRUCTION_SEQUENCE
INSTRUCTION_USAGE

1990年版 Standard VHDL Package の RUNBIST の値は削除されました。

1990年版BSDLの REGISTER_ACCESS属性 の IDCODE は1994年版BSDLではDEVICE_IDに変更されました。

1990年版BSDLでは，OBSERVE_ONLYの＜function＞と＜cell context＞の値は存在しません。

デバッグソリューションズがここで提供する情報は、正確かつ信頼できるものと考えておりますが、その使用に関する責務は一切負いません。ここに記載される情報は、2003年3月におけるものです。訂正、変更、改版に追従していない場合があります。最終的な確認はヘルプデスクにお問い合わせ下さい。

Web http://www.debsol.com
E-Mail mail@debsol.com

デバッグソリューションズ
Debug Solutions