Microsemi BSDL Files Format

Table of Contents

Description 1
Why BSDL? 2
Microsemi BSDL Files 2
Entity Description 3
Generic Parameter 3
Logical Port Description 3
Use Statement 4
Pin Mapping(s) 4
Scan Port Identification 4
Instruction Register Description 4
Register Access Description 5
Boundary Register Description 6
First Line (Input Scan Cell) 6
Second Line (Output Scan Cell) 6
Third Line (Enable Scan Cell) 6
Appendix A: Sample Microsemi BSDL File 8
Appendix B: Standard VHDL Package STD_1149_1_2001 11
List of Changes 14

Description

BSDL is a standard data format (a subset of VHDL) that describes the implementation of JTAG (IEEE 1149.1) in a device. BSDL was approved as IEEE Standard 1149.1b. Understanding JTAG architecture becomes especially important with the possibility of reprogramming a ProASIC® or ProASIC$^{\text{PLUS}}$® device through the JTAG port. This application note describes the elements of BSDL implemented in the BSDL files that are provided as support for Microsemi® devices. These files are available on the Microsemi website using the following link:

http://www.microsemi.com/products/fpga-soc/design-resources/bsdl-models

Note: BSDL files can also be generated from designer software. The procedure is described under the "Microsemi BSDL Files" section on page 2.

Why BSDL?

The purpose of BSDL is to provide a common language to describe boundary scan implementations in different devices from multiple vendors. Vendors of components supporting the 1149.1 standard are expected to supply BSDL descriptions to customers. Customers can use one or more BSDL files to do the following:

- Describe the test logic
- Synthesize the test logic
- Enable the generation of a test for a loaded board

The third item is by far the most important. The description in the BSDL file enables the user to determine the bit sequences required when performing chip tests using JTAG. These tests include reading out the serial JTAG chain as well as stimulating the device outputs. The latter is done by inserting and shifting serial test vectors.

A quick summary of the features included in the file is as follows:

- Length and structure of the boundary scan register
- Binary instruction codes
- Device identification code
- Availability of the TRST pin
- Physical locations of the JTAG pins

BSDL describes features of the test logic that vary from device to device—optional features. Mandatory features are not included in the file. The next section describes the file content and format in more detail.

Microsemi BSDL Files

Microsemi BSDL files are grouped into two categories: generic and device-specific. Generic files are available for all product families on the web at *http://www.microsemi.com/products/fpga-soc/design-resources/bsdl-models*. The generic files assign all user I/Os as inout. Generic BSDL files can also be generated for most product families in designer software using **File** > **Export** > **Auxiliary Files**. Device-specific files assign user I/Os as input, output, or inout.

Note: Device-specific files still use generic names for user I/Os (example, IO_19).

The choice of a generic or device-specific file is controlled by the GENERIC BSDL variable in the designer software. This is set or changed through **Options** > **Set Variable**. A value of 1 indicates a generic file, whereas a 0 value will generate a device-specific file. Although the generic version can be used in many cases, there are some situations where a device-specific version is required. One example is a ProASIC$^{\underline{\text{PLUS}}}$ design that uses Schmitt-trigger inputs. An I/O configured as a Schmitt-trigger input functions as input-only during testing, so the I/O definition in the file must be changed accordingly. If a generic file is used and the JTAG tester tries to configure the I/O as an output, the boundary scan test fails.

Microsemi BSDL description for a device consists of the following elements:

- Entity descriptions
- Generic parameter
- Logical port description
- Use statements
- Pin mapping(s)
- Scan port identification
- Instruction register description
- Register access description
- Boundary register description

These descriptions are discussed in more detail below and are keyed to the sample BSDL file in "Appendix A: Sample Microsemi BSDL File" on page 8.

Entity Description

The entity statement names the Microsemi device and package combination. In this case, it is the Microsemi RT54SX-S family and the RT54SX32S device in the CQFP-208 package. The entity description begins with an entity statement and terminates with an end statement.

```
entity a54200rtscqfp208s is
  [statements to describe the entity go here]
end a54200rtscqfp208s
```

Generic Parameter

The generic parameter in Microsemi files is the package type (example, "cqfp208s").

```
generic (PHYSICAL_PIN_MAP : string := "cqfp208s");
```

Logical Port Description

The port description gives logical names to the I/O pins (system and TAP pins). For JTAG inputs and outputs, the nature of the pin is what the user expects—either input or output. User outputs and bi-directionals are denoted in the BSDL file as outputs and inouts, respectively. However, for user inputs, the situation is more complicated. A user input is denoted as an inout in the BSDL file. Both a user input and a user bi-directional employ the same boundary-scan cell structure: three cells associated with the input buffer, the output buffer, and the enable. If the I/O structure is used as an input, the output buffer itself is always disabled by disabling the enable signal. However, the scan cell can still load the output buffer's input data from the chip core, just like the behavior of the inout structure. In summary, a user input will be marked as inout in an Microsemi BSDL file.

In addition to in, out, and inout, a pin can be identified as type linkage. This is used for no-connects (NC), power (V_{CC}), or ground (GND).

```
    port(
   TMS: in bit;
   TCK: in bit;
   TDI: in bit;
   TRST: in bit;
   TDO: out bit;
   CLKA: in bit;
   CLKB: in bit;
   FCLK: in bit;
   IO_51: out bit;
   IO_37: out bit;
   IO_36: out bit;
   IO_31: inout bit;
   IO_29: inout bit;
   IO_24: inout bit;
NC : linkage bit_vector(1 to 5);
   VCC: linkage bit_vector(1 to 15);
   GND: linkage bit_vector(1 to 13)
);
```

Use Statement

The use statement refers to the IEEE standard package external definitions found in packages and package bodies. There are three versions of the package corresponding to the code released with the 1990 (draft), 1994, and 2001 versions of IEEE Standard 1149.1. The Microsemi files currently reference the 1990 draft version of the package. For more details see "Appendix B: Standard VHDL Package STD_1149_1_2001" on page 11.

```
use STD_1149_1_1990.all;
```

Pin Mapping(s)

The pin mapping provides a mapping of logical signals to the physical pins in a particular device package.

```
 attribute PIN_MAP of a54200rtscqfp208s : entity is PHYSICAL_PIN_MAP;

 constant cqfp208s : PIN_MAP_STRING:=
"TRST: 30,"&
"TMS: 11,"&
"TDI: 2,"&
"TCK: 208,"&
"CLKB: 181,"&
"CLKA: 180,"&
"TDO: 103,"&
"FCLK: 82,"&
"VCC: ( 41, 40, 27, 12, 201, 184, 164, 148, 145, 130,"&
   "  115, 114, 98, 78, 60),"&
"GND: ( 52, 28, 26, 1, 185, 183, 157, 146, 131, 129,"&
   "  105, 79, 77),"&
"NC: ( 25, 182, 132, 80, 65),"&
"IO_51: 51,"&
"IO_37: 37,"&
"IO_36: 36,"&
"IO_31: 31,"&
"IO_29: 29,"&
"IO_24: 24,"&
```

Scan Port Identification

The scan port identification statements identify the signals present in the device's TAP. In addition, the maximum value of TCK is designated as 10 MHz.

```
attribute TAP_SCAN_IN of TDI : signal is TRUE;

attribute TAP_SCAN_OUT of TDO : signal is TRUE;

attribute TAP_SCAN_MODE of TMS : signal is TRUE;

attribute TAP_SCAN_CLOCK of TCK : signal is (10.0e6, BOTH);
```

Instruction Register Description

In the following example, the RT54SX-S has a 5-bit instruction code. The example also provides legal instruction mnemonics and opcodes. The remainder of this section lists additional device-dependent characteristics of the Instruction Register. The INSTRUCTION_CAPTURE attribute specifies the bit pattern loaded into the Instruction Register when the TAP Controller passes through the Capture-IR state.

The INSTRUCTION_DISABLE attribute lists all opcodes for which device outputs are disabled (this applies to the HIGHZ instruction). The statement in the file associates the name with the correct opcode.

The INSTRUCTION_GUARD attribute identifies an opcode that places the Bypass Register between TDI and TDO while driving the device outputs with the previous contents of the Boundary Scan Register. This is equivalent to the JTAG CLAMP instruction; again, the statement in the file associates this with the correct opcode.

Note: The INSTRUCTION_DISABLE and INSTRUCTION_GUARD attributes have been dropped from STD_1149_1_1994 and STD_1149_1_2001. The INSTRUCTION_PRIVATE attribute identifies the Microsemi commands—potentially unsafe for customers to use.

```
attribute INSTRUCTION_LENGTH of a54200rtscqfp208s: entity is 5;
attribute INSTRUCTION_OPCODE of a54200rtscqfp208s: entity is
      "EXTEST(00000),"&
      "SAMPLE(00001),"&
      "INTEST(00010),"&
      --"USERCODE(00011),"&
      "IDCODE(00100),"&
      --"USRSC(00101),"&
      "HIGHZ(01110),"&
      "CLAMP(01111),"&
      "PROBE(10000),"&
      "BYPASS(11111)";

attribute INSTRUCTION_CAPTURE of a54200rtscqfp208s: entity is "XXX01";
attribute INSTRUCTION_DISABLE of a54200rtscqfp208s: entity is "HIGHZ";
attribute INSTRUCTION_GUARD of a54200rtscqfp208s: entity is "CLAMP";
attribute INSTRUCTION_PRIVATE of a54200rtscqfp208s: entity is "PROBE";
```

Register Access Description

These commands indicate the structure of registers for user instructions such as IDCODE. REGISTER_ACCESS defines which register is placed between TDI and TDO for each instruction. The example shows that the HIGHZ and CLAMP instructions select the Bypass Register.

```
attribute REGISTER_ACCESS of a54200rtscqfp208s: entity is
      "BYPASS (HIGHZ, CLAMP)";
attribute IDCODE_REGISTER of a54200rtscqfp208s: entity is
      "XXXX"&--Version
      "XXXXXXXXXXXXXXXX"&--Device
      "00000101111"&--Manufacturer
      "1";--Required
```

Boundary Register Description

The Boundary Register description contains a list of boundary scan cells, along with information regarding the cell type and associated control. BOUNDARY_CELLS is included in the example, which gives a list of the cell types (BC_0 through BC-10) used. These are standard types provided in the IEEE package file; in this example, BC-1 is used. As with INSTRUCTION_GUARD and INSTRUCTION_DISABLE, BOUNDARY_CELLS has been dropped from STD_1149_1_1994 and STD_1149_1_2001.

BOUNDARY_LENGTH is the length of the boundary scan chain—729 in this case. BOUNDARY_REGISTER is the list of all boundary scan cells. The list below is a shortened version to give an idea of the different types of cells. This explains the gaps in the cell numbering (the first entry in each field).

The entry for PIN51 in the "Third Line (Enable Scan Cell)" section on page 6 has the following parts (note that there are three lines, one for each boundary scan cell associated with this pin):

First Line (Input Scan Cell)

- 0 – cell number
- BC_1 – boundary cell type (from IEEE package)
- * - Port ID - * – (the null element) indicates internal cell or output control (see next element)
- internal – internal cell
- X – value that should be loaded into scan-cell flip-flops when board-level software might choose a random value. In this case, the X indicates "don’t care."

Second Line (Output Scan Cell)

- 1 – cell number
- BC_1 – boundary cell type (from IEEE package)
- IO_51 – Port ID – system output connected to cell
- output3 – cell that drives data to a tristate output
- X – value that should be loaded into scan-cell flip-flops when board-level software might choose a random value. In this case the X indicates "don’t care."
- 2 – control cell that can disable the output. In this case, it is the Enable scan cell from Line 3.
- 0 – the value that must be scanned into the control cell (cell 2) to disable the output IO_51
- Z – the condition of the output driver when disabled

Third Line (Enable Scan Cell)

- 2 – cell number
- BC_1 – boundary cell type (from IEEE package)
- * - Port ID - * – (the null element) indicates internal cell or output control (see next element)
- control – output control
- 0 – value that should be loaded into scan-cell flip-flops when board-level software might choose a random value

```
   attribute BOUNDARY_CELLS of a54200rtscqfp208s: entity is "BC_1";
attribute BOUNDARY_LENGTH of a54200rtscqfp208s: entity is 729;
attribute BOUNDARY_REGISTER of a54200rtscqfp208s: entity is
--BSR75, PIN51, spec=BIN_OUT
" 0(BC_1, *,internal,   X), "&
" 1(BC_1, IO_51,output3, X, 2, 0, Z), "&
" 2(BC_1, *,control, 0), "&

--BSR73, NC
" 6(BC_1, *, internal, X), "&
" 7(BC_1, *, internal, X), "&
" 8(BC_1, *, internal, X), "&

--BSR56, PIN37
" 48(BC_1, *,internal,   X), "&
" 49(BC_1, IO_37,output3, X, 50, 0, Z), "&
" 50(BC_1, *,control, 0), "&

--BSR55, PIN36
" 51(BC_1, *, internal, X), "&
" 52(BC_1, IO_36,output3, X, 53, 0, Z), "&
" 53(BC_1, *,control, 0), "&

--BSR44, PIN31
" 78(BC_1, IO_31,input,   X), "&
" 79(BC_1, IO_31,output3, X, 80, 0, Z), "&
" 80(BC_1, *,control, 0), "&

--BSR43, PIN30, spec=TRSTB

--BSR42, PIN29
" 81(BC_1, IO_29,input,   X), "&
" 82(BC_1, IO_29,output3, X, 83, 0, Z), "&
" 83(BC_1, *,control, 0), "&

--BSR37, PIN24
" 84(BC_1, IO_24,input,   X), "&
" 85(BC_1, IO_24,output3, X, 86, 0, Z), "&
" 86(BC_1, *,control, 0), "&
```

Appendix A: Sample Microsemi BSDL File

This appendix contains the code for sample Microsemi BSDL File

```
-- C:\share\pcitest\tdma32\sxstdma32.bsd
-- DESIGN:sxstdma32
-- FAMILY:54SXA
-- DEVICE:54200rts
-- PACKAGE:cqfp208s
-- RESTRICT JTAG:1
-- RESTRICT TRST:1

entity a54200rtscqfp208s is
   generic (PHYSICAL_PIN_MAP : string := "cqfp208s");
port(
   TMS: in bit;
   TCK: in bit;
   TDI: in bit;
   TRST: in bit;
   TDO: out bit;
   CLKA: in bit;
   CLKB: in bit;
   FCLK: in bit;
   IO_51: out bit;
   IO_37: out bit;
   IO_36: out bit;
   IO_31: inout bit;
   IO_29: inout bit;
   IO_24: inout bit;
   NC : linkage bit_vector(1 to 5);
   VCC: linkage bit_vector(1 to 15);
   GND: linkage bit_vector(1 to 13)
);

use STD_1149_1_1990.all;

attribute PIN_MAP of a54200rtscqfp208s : entity is PHYSICAL_PIN_MAP;

constant cqfp208s : PIN_MAP_STRING:=
   "TRST: 30,"&
   "TMS: 11,"&
   "TDI: 2,"&
   "TCK: 208,"&
   "CLKB: 181,"&
   "CLKA: 180,"&
   "TDO: 103,"&
   "FCLK: 82,"&
   "VCC: ( 41, 40, 27, 12, 201, 184, 164, 148, 145, 130,"&
      "  115, 114, 98, 78, 60),"&
   "GND: ( 52, 28, 26, 1, 185, 183, 157, 146, 131, 129,"&
      "  105, 79, 77),"&
   "NC: ( 25, 182, 132, 80, 65),"&
   "IO_51: 51,"&
   "IO_37: 37,"&
   "IO_36: 36,"&
   "IO_31: 31,"&
   "IO_29: 29,"&
   "IO_24: 24,"&
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_RESET of TRST: signal is true;
attribute TAP_SCAN_CLOCK of TCK: signal is (10.00E6, BOTH);

attribute INSTRUCTION_LENGTH of a54200rtscqfp208s: entity is 5;
attribute INSTRUCTION_OPCODE of a54200rtscqfp208s: entity is
   "EXTEST(00000),"&
   "SAMPLE(00001),"&
   "INTEST(00010),"&
   --"USERCODE(00011),"&
   "IDCODE(00100),"&
   --"USRSC(00101),"&
   "HIGHZ(01110),"&
   "CLAMP(01111),"&
   "PROBE(10000),"&
   "BYPASS(11111)";

attribute INSTRUCTION_CAPTURE of a54200rtscqfp208s: entity is "XXX01";
attribute INSTRUCTION_DISABLE of a54200rtscqfp208s: entity is "HIGHZ";
attribute INSTRUCTION_GUARD of a54200rtscqfp208s: entity is "CLAMP";
attribute INSTRUCTION_PRIVATE of a54200rtscqfp208s: entity is "PROBE";
attribute REGISTER_ACCESS of a54200rtscqfp208s: entity is
   "BYPASS (HIGHZ, CLAMP)";
attribute IDCODE_REGISTER of a54200rtscqfp208s: entity is
   "XXXX"&--Version
   "XXXXXXXXXXXXXXXX"&--Device
   "00000101111"&--Manufacturer
   "1";--Required

attribute BOUNDARY_CELLS of a54200rtscqfp208s: entity is "BC_1";
attribute BOUNDARY_LENGTH of a54200rtscqfp208s: entity is 729;
attribute BOUNDARY_REGISTER of a54200rtscqfp208s: entity is

--BSR75, PIN51, spec=BIN_OUT
" 0(BC_1, *,internal,   X), "&
" 1(BC_1, IO_51,output3, X, 2, 0, Z), "&
" 2(BC_1, *,control, 0), "&

--BSR73, NC
" 6(BC_1, *, internal, X), "&
" 7(BC_1, *, internal, X), "&
" 8(BC_1, *, internal, X), "&

--BSR56, PIN37
" 48(BC_1, *,internal,   X), "&
" 49(BC_1, IO_37,output3, X, 50, 0, Z), "&
" 50(BC_1, *,control, 0), "&

--BSR55, PIN36
" 51(BC_1, *, internal, X), "&
" 52(BC_1, IO_36,output3, X, 53, 0, Z), "&
" 53(BC_1, *,control, 0), "&

--BSR44, PIN31
" 78(BC_1, IO_31,input,   X), "&
" 79(BC_1, IO_31,output3, X, 80, 0, Z), "&
" 80(BC_1, *,control, 0), "&

--BSR43, PIN30, spec=TRSTB

--BSR42, PIN29
" 81(BC_1, IO_29,input,   X), "&
" 82(BC_1, IO_29,output3, X, 83, 0, Z), "&
" 83(BC_1, *,control, 0), "&

--BSR37, PIN24
" 84(BC_1, IO_24,input,   X), "&
" 85(BC_1, IO_24,output3, X, 86, 0, Z), "&
" 86(BC_1, *,control, 0), "&
```

```
end a54200rtscqfp208s;
```

Appendix B: Standard VHDL Package STD_1149_1_2001

This appendix contains the code for the 1990 version of the IEEE package that is referenced in Microsemi files. Note that there are minor differences in the 1994 and 2001 packages. Here is a summary of those attributes that are used in the Microsemi files:

- Five 1990 attributes—INSTRUCTION GUARD, INSTRUCTION_DISABLE, BOUNDARY_CELLS, INSTRUCTION_SEQUENCE, and INSTRUCTION_USAGE—were deleted.
- In the 1990 version, the device ID register was named IDCODE in the REGISTER_ACCESS attribute. The name has since been changed to DEVICE_ID.

```
--  STD_1149_1_1990    VHDL Package and Package Body in support of
--  BSDL Version 0.0
--
--  source        :  %F%
--
--  revision      :  %I%
--
--  date          :  %G%-%U%

 package STD_1149_1_1990 is             -- Created 900525

 -- 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 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, RUNBIST);
 type CELL_TYPE is (INPUT, INTERNAL, CLOCK,
                     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 defered 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 1149.1-1990 Package


package body STD_1149_1_1990 is   -- Standard Boundary Cells
                                  -- Written by Ken Parker  900525

-- Description for f10-12, f10-16, f10-18c, f10-18d, f10-21c

constant BC_1 : CELL_INFO :=
 ((INPUT,   EXTEST,  PI),  (OUTPUT2,  EXTEST,  PI),
  (INPUT,   SAMPLE,  PI),  (OUTPUT2,  SAMPLE,  PI),
  (INPUT,   INTEST,  PI),  (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, PI) );

-- 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, EXTEST,  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, PI) );

-- 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 1149.1-1990 Package Body
```

List of Changes

The following table shows important changes made in this document for each revision.

Revision	Changes	Pages
Revision 2 (November 2015)	Non-technical updates.	NA
Revision 1 (November 2002)	The "Microsemi BSDL Files" section was updated.	2

**The part number is located on the last page of the document.*

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for communications, defense & security, aerospace and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; security technologies and scalable anti-tamper products; Ethernet Solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif., and has approximately 3,600 employees globally. Learn more at **www.microsemi.com**.

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA

Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996

E-mail: sales.support@microsemi.com

51900003-2/11.15