SECTION 24
IEEE 1149.1-COMPLIANT INTERFACE (JTAG)

24.1 IEEE 1149.1 Test Access Port (TAP) and Joint Test Action Group (JTAG)

The chip design includes user-accessible test logic that is compatible with *the IEEE 1149.1-1994 Standard Test Access Port and Boundary Scan Architecture*.The implementation supports circuit-board test strategies based on this standard. An overview of the pins requirement on JTAG is shown:

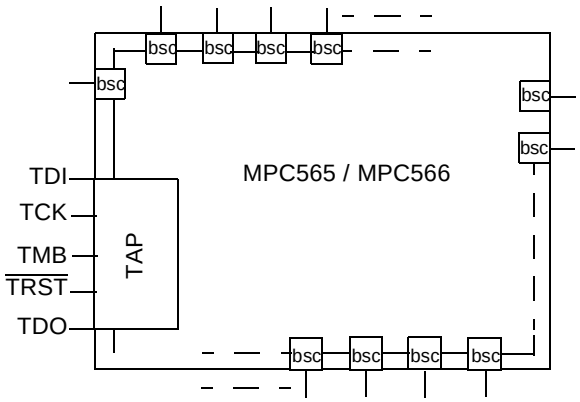

Figure 24-1 Pin Requirement on JTAG

24.2 IEEE 1149.1 Test Access Port

The MPC565 / MPC566 provides a dedicated user-accessible test access port (TAP) that is compatible with the *IEEE 1149.1 Standard Test Access Port and Boundary Scan Architecture* in all but 2 areas listed below. Problems associated with testing high density circuit boards have led to development of this proposed standard under the sponsorship of the Test Technology Committee of IEEE and the Joint Test Action Group (JTAG). The MPC565 / MPC566 implementation supports circuit-board test strategies based on this standard.

IEEE1149.1 Compatibility Exceptions

- The MPC565 enters JTAG mode by going through a standard device reset sequence with a specific configuration word applied to the databus. Once JTAG has been entered, the MPC565 remains in JTAG mode until another reset sequence is applied to exit JTAG mode, or the device is powered down.
- The JTAG output port, "tdo_dsdo" is configured with a weak pullup when its output drivers are disabled, rather than being tri-stated.

The TAP consists of five dedicated signal pins, a 16-state TAP controller, and two test data registers. A boundary scan register links all device signal pins into a single shift register. The test logic implemented utilizes static logic design. The MPC565 / MPC566 implementation provides the capability to:

1. Perform boundary scan operations to test circuit-board electrical continuity.
2. Bypass the MPC565 / MPC566 for a given circuit-board test by effectively reducing the boundary scan register to a single cell.
3. Sample the MPC565 / MPC566 system pins during operation and transparently shift out the result in the boundary scan register.
4. Disable the output drive to pins during circuit-board testing.

NOTE

Certain precautions must be observed to ensure that the IEEE 1149.-like test logic does not interfere with nontest operation. See **24.3.1 Non-Scan Chain Operation** for details.

24.2.1 Overview

An overview of the MPC565 / MPC566 scan chain implementation is shown in . The MPC565 / MPC566 implementation includes a TAP controller, a 4-bit instruction register, and two test registers (a one-bit bypass register and a 520-bit boundary scan register). This implementation includes a dedicated TAP consisting of the following signals:

- TCK — a test clock input to synchronize the test logic. (with an internal pull-down resistor)
- TMS — a test mode select input (with an internal pullup resistor) that is sampled on the rising edge of TCK to sequence the TAP controller's state machine.
- TDI — a test data input (with an internal pullup resistor) that is sampled on the rising edge of TCK.
- TDO — a three-state test data output that is actively driven in the shift-IR and shift-DR controller states. TDO changes on the falling edge of TCK. (This pin also has a weak pull-up that is active when output drivers are disabled, except during a HI-Z instruction).
- $\overline{\text{TRST}}$ — an asynchronous reset with an internal pull-up resistor that provides initialization of the TAP controller and other logic required by the standard.

NOTE

JTAG mode does not provide access to the internal MPC565 / MPC566 circuitry. It allows access only to the input or output pad (periphery) circuitry

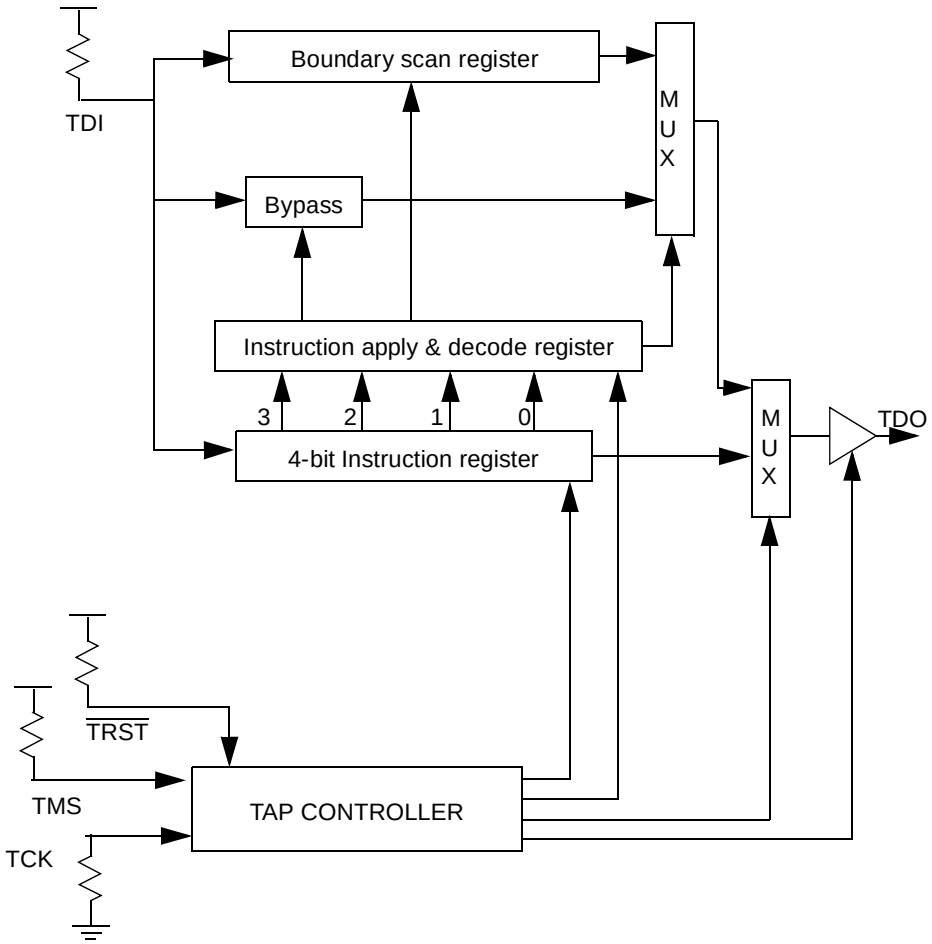

Figure 24-2 Test Logic Block Diagram

24.2.1.1 TAP CONTROLLER

The TAP controller is responsible for interpreting the sequence of logical values on the TMS signal. It is a synchronous state machine that controls the operation of the JTAG logic. The state machine is shown in . The value shown adjacent to each arc represents the value of the TMS signal sampled on the rising edge of the TCK signal.

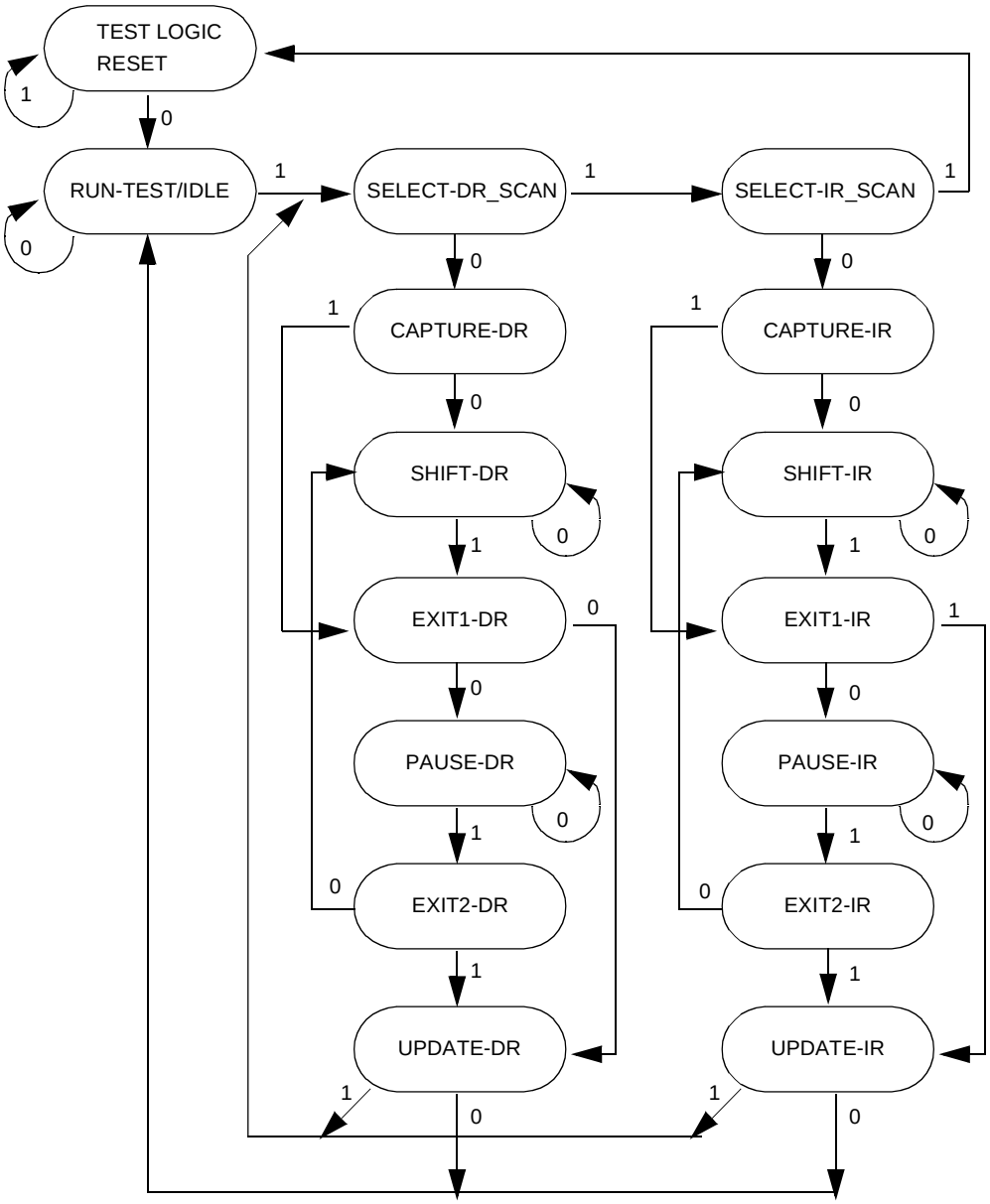

Figure 24-3 TAP Controller State Machine

24.2.1.2 Boundary Scan Register

The MPC565 / MPC566 scan chain implementation has a 520-bit boundary scan register. This register contains bits for most device signals, clock pins and associated control signals. The XTAL, EXTAL and XFC pins are associated with analog signals and are not included in the boundary scan register. The $\overline{\text{PORESET}}$, $\overline{\text{HRESET}}$, and $\overline{\text{SRESET}}$ pins are also excluded from the boundary scan register.

The 520-bit boundary scan register can be connected between TDI and TDO by selecting the EXTEST or SAMPLE/PRELOAD instructions. This register is used to capturing signal pin data on the input pins, forcing fixed values on the output signal

pins, and selecting the direction and drive characteristics (a logic value or high impedance) of the bidirectional and three-state signal pins.

The key to using the boundary scan register is knowing the boundary scan bit order and the pins that are associated with them. **Table 24-1** shows the bit order starting from the TDO output and going to the TDI input.

Table 24-1 Boundary Scan Bit Definition

BSDL Bit	Cell Type	Port Name	BSDL Function	saf. val.	cnt. cell	dis. val.	rslt	Pin Function	Pad Type
0	BC_2	*	internal	1					
1	BC_2	mdo_2	output2	1				O	26v
2	BC_2	*	internal	1					
3	BC_2	mdo_3	output2	1				O	26v
4	BC_2	*	internal	1					
5	BC_2	mseo_b	output2	1				O	26v
6	BC_2	*	internal	1					
7	BC_2	iwp0_vfls0	output2	1				O	26v
8	BC_2	*	internal	1					
9	BC_2	iwp1_vfls1	output2	1				O	26v
10	BC_2	*	controlr	0					
11	BC_7	addr_sgpioa(16)	bidir	0	10	0	Z	I/O	26v5vs
12	BC_2	*	controlr	0					
13	BC_7	addr_sgpioa(17)	bidir	0	12	0	Z	I/O	26v5vs
14	BC_2	*	controlr	0					
15	BC_7	sgpioc6_frz_ptr_b	bidir	0	14	0	Z	I/O	26v5vs
16	BC_2	*	controlr	0					
17	BC_7	addr_sgpioa(8)	bidir	0	16	0	Z	I/O	26v5vs
18	BC_2	*	controlr	0					
19	BC_7	addr_sgpioa(18)	bidir	0	18	0	Z	I/O	26v5vs
20	BC_2	*	controlr	0					
21	BC_7	addr_sgpioa(19)	bidir	0	20	0	Z	I/O	26v5vs
22	BC_2	*	controlr	0					
23	BC_7	addr_sgpioa(9)	bidir	0	22	0	Z	I/O	26v5vs
24	BC_2	*	controlr	0					
25	BC_7	addr_sgpioa(10)	bidir	0	24	0	Z	I/O	26v5vs
26	BC_2	*	controlr	0					

Table 24-1 Boundary Scan Bit Definition (Continued)

BSDL Bit	Cell Type	Port Name	BSDL Function	saf. val.	cnt. cell	dis. val.	rslt	Pin Function	Pad Type
27	BC_7	addr_sgpioa(20)	bidir	0	26	0	Z	I/O	26v6vs
28	BC_2	*	controlr	0					
29	BC_7	addr_sgpioa(21)	bidir	0	28	0	Z	I/O	26v5vs
30	BC_2	*	controlr	0					
31	BC_7	addr_sgpioa(11)	bidir	0	30	0	Z	I/O	26v5vs
32	BC_2	*	controlr	0					
33	BC_7	addr_sgpioa(12)	bidir	0	32	0	Z	I/O	26v5vs
34	BC_2	*	controlr	0					
35	BC_7	addr_sgpioa(22)	bidir	0	34	0	Z	I/O	26v5vs
36	BC_2	*	controlr	0					
37	BC_7	addr_sgpioa(23)	bidir	0	36	0	Z	I/O	26v5vs
38	BC_2	*	controlr	0					
39	BC_7	addr_sgpioa(13)	bidir	0	38	0	Z	I/O	26v5vs
40	BC_2	*	controlr	0					
41	BC_7	addr_sgpioa(24)	bidir	0	40	0	Z	I/O	26v5vs
42	BC_2	*	controlr	0					
43	BC_7	addr_sgpioa(25)	bidir	0	42	0	Z	I/O	26v5vs
44	BC_2	*	controlr	0					
45	BC_7	addr_sgpioa(14)	bidir	0	44	0	Z	I/O	26v5vs
46	BC_2	*	controlr	0					
47	BC_7	addr_sgpioa(15)	bidir	0	46	0	Z	I/O	26v5vs
48	BC_2	*	controlr	0					
49	BC_7	addr_sgpioa(30)	bidir	0	48	0	Z	I/O	26v5vs
50	BC_2	*	controlr	0					
51	BC_7	addr_sgpioa(26)	bidir	0	50	0	Z	I/O	26v5vs
52	BC_2	*	controlr	0					
53	BC_7	addr_sgpioa(27)	bidir	0	52	0	Z	I/O	26v5vs
54	BC_2	*	controlr	0					
55	BC_7	addr_sgpioa(31)	bidir	0	54	0	Z	I/O	26v5vs
56	BC_2	*	controlr	0					
57	BC_7	addr_sgpioa(28)	bidir	0	56	0	Z	I/O	26v5vs
58	BC_2	*	controlr	0					

Table 24-1 Boundary Scan Bit Definition (Continued)

BSDL Bit	Cell Type	Port Name	BSDL Function	saf. val.	cnt. cell	dis. val.	rslt	Pin Function	Pad Type
59	BC_7	addr_sgpioa(29)	bidir	0	58	0	Z	I/O	26v5vs
60	BC_2	*	controlr	0					
61	BC_7	data_sgpiod(0)	bidir	0	60	0	Z	I/O	26v5vs
62	BC_2	*	controlr	0					
63	BC_7	data_sgpiod(29)	bidir	0	62	0	Z	I/O	26v5vs
64	BC_2	*	controlr	0					
65	BC_7	data_sgpiod(1)	bidir	0	64	0	Z	I/O	26v5vs
66	BC_2	*	controlr	0					
67	BC_7	data_sgpiod(2)	bidir	0	66	0	Z	I/O	26v5vs
68	BC_2	*	controlr	0					
69	BC_7	data_sgpiod(3)	bidir	0	68	0	Z	I/O	26v5vs
70	BC_2	*	controlr	0					
71	BC_7	data_sgpiod(27)	bidir	0	70	0	Z	I/O	26v5vs
72	BC_2	*	controlr	0					
73	BC_7	data_sgpiod(4)	bidir	0	72	0	Z	I/O	26v5vs
74	BC_2	*	controlr	0					
75	BC_7	data_sgpiod(28)	bidir	0	74	0	Z	I/O	26v5vs
76	BC_2	*	controlr	0					
77	BC_7	data_sgpiod(31)	bidir	0	76	0	Z	I/O	26v5vs
78	BC_2	*	controlr	0					
79	BC_7	data_sgpiod(5)	bidir	0	78	0	Z	I/O	26v5vs
80	BC_2	*	controlr	0					
81	BC_7	data_sgpiod(6)	bidir	0	80	0	Z	I/O	26v5vs
82	BC_2	*	controlr	0					
83	BC_7	data_sgpiod(30)	bidir	0	82	0	Z	I/O	26v5vs
84	BC_2	*	controlr	0					
85	BC_7	data_sgpiod(7)	bidir	0	84	0	Z	I/O	26v5vs
86	BC_2	*	controlr	0					
87	BC_7	data_sgpiod(25)	bidir	0	86	0	Z	I/O	26v5vs
88	BC_2	*	controlr	0					
89	BC_7	data_sgpiod(8)	bidir	0	88	0	Z	I/O	26v5vs
90	BC_2	*	controlr	0					

Table 24-1 Boundary Scan Bit Definition (Continued)

BSDL Bit	Cell Type	Port Name	BSDL Function	saf. val.	cnt. cell	dis. val.	rslt	Pin Function	Pad Type
91	BC_7	data_sgpiod(24)	bidir	0	90	0	Z	I/O	26v5vs
92	BC_2	*	controlr	0					
93	BC_7	data_sgpiod(9)	bidir	0	92	0	Z	I/O	26v5vs
94	BC_2	*	controlr	0					
95	BC_7	data_sgpiod(10)	bidir	0	94	0	Z	I/O	26v5vs
96	BC_2	*	controlr	0					
97	BC_7	data_sgpiod(26)	bidir	0	96	0	Z	I/O	26v5vs
98	BC_2	*	controlr	0					
99	BC_7	data_sgpiod(22)	bidir	0	98	0	Z	I/O	26v5vs
100	BC_2	*	controlr	0					
101	BC_7	data_sgpiod(11)	bidir	0	100	0	Z	I/O	26v5vs
102	BC_2	*	controlr	0					
103	BC_7	data_sgpiod(12)	bidir	0	102	0	Z	I/O	26v5vs
104	BC_2	*	controlr	0					
105	BC_7	data_sgpiod(13)	bidir	0	104	0	Z	I/O	26v5vs
106	BC_2	*	controlr	0					
107	BC_7	data_sgpiod(20)	bidir	0	106	0	Z	I/O	26v5vs
108	BC_2	*	controlr	0					
109	BC_7	data_sgpiod(14)	bidir	0	108	0	Z	I/O	26v5vs
110	BC_2	*	controlr	0					
111	BC_7	data_sgpiod(23)	bidir	0	110	0	Z	I/O	26v5vs
112	BC_2	*	controlr	0					
113	BC_7	data_sgpiod(15)	bidir	0	112	0	Z	I/O	26v5vs
114	BC_2	*	controlr	0					
115	BC_7	data_sgpiod(16)	bidir	0	114	0	Z	I/O	26v5vs
116	BC_2	*	controlr	0					
117	BC_7	data_sgpiod(21)	bidir	0	116	0	Z	I/O	26v5vs
118	BC_2	*	controlr	0					
119	BC_7	data_sgpiod(17)	bidir	0	118	0	Z	I/O	26v5vs
120	BC_2	*	controlr	0					
121	BC_7	data_sgpiod(18)	bidir	0	120	0	Z	I/O	26v5vs
122	BC_2	*	controlr	0					

Table 24-1 Boundary Scan Bit Definition (Continued)

BSDL Bit	Cell Type	Port Name	BSDL Function	saf. val.	cnt. cell	dis. val.	rslt	Pin Function	Pad Type
123	BC_7	data_sgpiod(19)	bidir	0	122	0	Z	I/O	26v5vs
124	BC_2	*	controlr	0					
125	BC_7	irq3_b_kr_b_retry_b_sgpioc3	bidir	0	124	0	Z	I/O	26v5vs
126	BC_2	*	controlr	0					
127	BC_7	irq4_b_at2_sgpioc4	bidir	0	126	0	Z	I/O	26v5vs
128	BC_2	*	controlr	0					
129	BC_7	irq1_b_rsv_b_sgpioc1	bidir	0	128	0	Z	I/O	26v5vs
130	BC_2	*	controlr	0					
131	BC_7	sgpioc7_irqout_b_lwp0	bidir	0	130	0	Z	I/O	26v5vs
132	BC_2	*	controlr	0					
133	BC_7	bb_b_vf2_iwp3	bidir	0	132	0	Z	I/O	26v
134	BC_2	*	controlr	0					
135	BC_7	bg_b_vf0_lwp1	bidir	0	134	0	Z	I/O	26v
136	BC_2	*	controlr	0					
137	BC_7	br_b_vf1_iwp2	bidir	0	136	0	Z	I/O	26v
138	BC_2	*	controlr	0					
139	BC_7	rd_wr_b	bidir	0	138	0	Z	I/O	26v
140	BC_2	*	internal	1					
141	BC_2	oe_b	output2	1				O	26v
142	BC_2	*	controlr	0					
143	BC_7	tea_b	bidir	0	142	0	Z	O	26v
144	BC_2	*	controlr	0					
145	BC_7	irq2_b_cr_b_sgpioc2	bidir	0	144	0	Z	I/O	26v5vs
146	BC_2	*	controlr	0					
147	BC_7	irq0_b_sgpioc0	bidir	0	146	0	Z	I/O	26v5vs
148	BC_2	*	internal	1					
149	BC_2	we_b_at(0)	output2	1				O	26v
150	BC_2	*	internal	1					
151	BC_2	we_b_at(1)	output2	1				O	26v

Table 24-1 Boundary Scan Bit Definition (Continued)

BSDL Bit	Cell Type	Port Name	BSDL Function	saf. val.	cnt. cell	dis. val.	rslt	Pin Function	Pad Type
152	BC_2	*	internal	1					
153	BC_2	we_b_at(2)	output2	1				O	26v
154	BC_2	*	internal	1					
155	BC_2	we_b_at(3)	output2	1				O	26v
156	BC_2	*	internal	1					
157	BC_2	cs0_b	output2	1				O	26v
158	BC_2	*	internal	1					
159	BC_2	cs1_b	output2	1				O	26v
160	BC_2	*	internal	1					
161	BC_2	cs2_b	output2	1				O	26v
162	BC_2	*	internal	1					
163	BC_2	cs3_b	output2	1				O	26v
164	BC_2	*	controlr	0					
165	BC_7	burst_b	bidir	0	164	0	Z	I/O	26v
166	BC_2	*	controlr	0					
167	BC_7	bi_b_sts_b	bidir	0	166	0	Z	I/O	26v
168	BC_2	*	controlr	0					
169	BC_7	tsiz0	bidir	0	168	0	Z	I/O	26v
170	BC_2	*	controlr	0					
171	BC_7	tsiz1	bidir	0	170	0	Z	I/O	26v
172	BC_2	*	controlr	0					
173	BC_7	ts_b	bidir	0	172	0	Z	I/O	26v
174	BC_2	*	controlr	0					
175	BC_7	ta_b	bidir	0	174	0	Z	I/O	26v
176	BC_2	*	controlr	0					
177	BC_7	bdip_b	bidir	0	176	0	Z	I/O	26v
178	BC_2	*	internal	0					
179	BC_4	b0epee	input	X				I	26v
180	BC_2	*	internal	0					
181	BC_4	epee	input	X				I	26v
182	BC_2	*	internal	1					
183	BC_2	clkout	output2	1				I/O	26vf

Table 24-1 Boundary Scan Bit Definition (Continued)

BSDL Bit	Cell Type	Port Name	BSDL Function	saf. val.	cnt. cell	dis. val.	rslt	Pin Function	Pad Type
184	BC_2	*	internal	1					
185	BC_2	engclk_buclk	output2	1				O	26vs5vr
186	BC_2	*	controlr	0					
187	BC_7	irq5_b_sgpioc5_modck1	bidir	0	186	0	Z	I/O	26v
188	BC_2	*	controlr	0					
189	BC_7	irq6_b_modck2	bidir	0	188	0	Z	I	26v
190	BC_2	*	controlr	0					
191	BC_7	irq7_b_modck3	bidir	0	190	0	Z	I	26v
192	BC_2	*	controlr	0					
193	BC_7	rstconf_b_texp	bidir	0	192	0	Z	I/O	26v
194	BC_4	extclk	input	X				I	extclk
195	BC_2	*	controlr	0					
196	BC_7	a_cnrx0	bidir	0	195	0	Z	I	5vsa
197	BC_2	*	internal	1					
198	BC_2	a_cntx0	output2	1				O	5vfa
199	BC_2	*	controlr	0					
200	BC_7	a_pcs0_ss_b_qgpio0	bidir	0	199	0	Z	I/O	5vfa
201	BC_2	*	internal	0					
202	BC_4	a_eck	input	X				I	5vfa
203	BC_2	*	controlr	0					
204	BC_7	a_pcs1_qgpio1	bidir	0	203	0	Z	I/O	5vfa
205	BC_2	*	internal	1					
206	BC_2	a_txd2_qgpo2	output2	1				O	5vfa
207	BC_2	*	controlr	0					
208	BC_7	a_pcs2_qgpio2	bidir	0	207	0	Z	I/O	5vfa
209	BC_4	a_rxd2_qgpi2	input	X				I	5vido
210	BC_4	b_rxd1_qgpi1	input	X				I	5vido
211	BC_4	a_rxd1_qgpi1	input	X				I	5vido
212	BC_2	*	controlr	0					
213	BC_7	a_mosi_qgpio5	bidir	0	212	0	Z	I/O	5vh
214	BC_2	*	controlr	0					

Table 24-1 Boundary Scan Bit Definition (Continued)

BSDL Bit	Cell Type	Port Name	BSDL Function	saf. val.	cnt. cell	dis. val.	rslt	Pin Function	Pad Type
215	BC_7	a_pcs3_qgpio3	bidir	0	214	0	Z	I/O	5vfa
216	BC_2	*	controlr	0					
217	BC_7	a_miso_qgpio4	bidir	0	216	0	Z	I/O	5vh
218	BC_2	*	controlr	0					
219	BC_7	a_sck_qgpio6	bidir	0	218	0	Z	I/O	5vh
220	BC_2	*	controlr	0					
221	BC_7	b_pcs2_qgpio2	bidir	0	220	0	Z	I/O	5vfa
222	BC_2	*	internal	0					
223	BC_4	b_rxd2_j1850_rx	input	X				I	5vsa
224	BC_2	*	internal	1					
225	BC_2	a_txd1_qgpo1	output2	1				O	5vsa
226	BC_2	*	internal	1					
227	BC_2	b_txd2_qgpo2	output2	1				O	5vsa
228	BC_2	*	internal	1					
229	BC_2	b_txd1_qgpo1	output2	1				O	5vsa
230	BC_2	*	internal	0					
231	BC_4	b_eck	input	X				I	5vsa
232	BC_2	*	controlr	0					
233	BC_7	b_sck_qgpio6	bidir	0	232	0	Z	I/O	5vh
234	BC_2	*	controlr	0					
235	BC_7	b_mosi_qgpio5	bidir	0	234	0	Z	I/O	5vh
236	BC_2	*	controlr	0					
237	BC_7	b_miso_qgpio4	bidir	0	236	0	Z	I/O	5vh
238	BC_2	*	controlr	0					
239	BC_7	b_pcs3_j1850_tx	bidir	0	238	0	Z	O	5vfa
240	BC_2	*	controlr	0					
241	BC_7	b_pcs1_qgpio1	bidir	0	240	0	Z	I/O	5vfa
242	BC_2	*	controlr	0					
243	BC_7	b_pcs0_ss_b_qgpio0	bidir	0	242	0	Z	I/O	5vfa
244	BC_2	*	controlr	0					
245	BC_7	vfls1_mpio32b4	bidir	0	244	0	Z	I/O	26v5vs

Table 24-1 Boundary Scan Bit Definition (Continued)

BSDL Bit	Cell Type	Port Name	BSDL Function	saf. val.	cnt. cell	dis. val.	rslt	Pin Function	Pad Type
246	BC_2	*	controlr	0					
247	BC_7	vfls0_mpio32b3	bidir	0	246	0	Z	I/O	26v5vs
248	BC_2	*	controlr	0					
249	BC_7	vf2_mpio32b2	bidir	0	248	0	Z	I/O	26v5vs
250	BC_2	*	controlr	0					
251	BC_7	vf1_mpio32b1	bidir	0	250	0	Z	I/O	26v5vs
252	BC_2	*	controlr	0					
253	BC_7	vf0_mpio32b0	bidir	0	252	0	Z	I/O	26v5vs
254	BC_2	*	controlr	0					
255	BC_7	mpwm4_mpio32b5	bidir	0	254	0	Z	I/O	5vsa
256	BC_2	*	controlr	0					
257	BC_7	mpwm19	bidir	0	256	0	Z	I/O	5vsa
258	BC_2	*	controlr	0					
259	BC_7	mpio32b15	bidir	0	258	0	Z	I/O	5vsa
260	BC_2	*	controlr	0					
261	BC_7	c_cnrx0_mpio32b14	bidir	0	260	0	Z	I/O	5vsa
262	BC_2	*	controlr	0					
263	BC_7	c_cntx0_mpio32b13	bidir	0	262	0	Z	I/O	5vfa
264	BC_2	*	controlr	0					
265	BC_7	mpwm21_mpio32b12	bidir	0	264	0	Z	I/O	5vsa
266	BC_2	*	controlr	0					
267	BC_7	mpwm20_mpio32b11	bidir	0	266	0	Z	I/O	5vsa
268	BC_2	*	controlr	0					
269	BC_7	mda15	bidir	0	268	0	Z	I/O	5vsa
270	BC_2	*	controlr	0					
271	BC_7	mda14	bidir	0	270	0	Z	I/O	5vsa
272	BC_2	*	controlr	0					
273	BC_7	mpwm16	bidir	0	272	0	Z	I/O	5vsa
274	BC_2	*	controlr	0					
275	BC_7	mpwm3	bidir	0	274	0	Z	I/O	5vsa

Table 24-1 Boundary Scan Bit Definition (Continued)

BSDL Bit	Cell Type	Port Name	BSDL Function	saf. val.	cnt. cell	dis. val.	rslt	Pin Function	Pad Type
276	BC_2	*	controlr	0					
277	BC_7	mpwm2	bidir	0	276	0	Z	I/O	5vsa
278	BC_2	*	controlr	0					
279	BC_7	mpwm1	bidir	0	278	0	Z	I/O	5vsa
280	BC_2	*	controlr	0					
281	BC_7	mpwm0	bidir	0	280	0	Z	I/O	5vsa
282	BC_2	*	controlr	0					
283	BC_7	mda31	bidir	0	282	0	Z	I/O	5vsa
284	BC_2	*	controlr	0					
285	BC_7	mda30	bidir	0	284	0	Z	I/O	5vsa
286	BC_2	*	controlr	0					
287	BC_7	mda29	bidir	0	286	0	Z	I/O	5vsa
288	BC_2	*	controlr	0					
289	BC_7	mda28	bidir	0	288	0	Z	I/O	5vsa
290	BC_2	*	controlr	0					
291	BC_7	mda27	bidir	0	290	0	Z	I/O	5vsa
292	BC_2	*	controlr	0					
293	BC_7	mda13	bidir	0	292	0	Z	I/O	5vsa
294	BC_2	*	controlr	0					
295	BC_7	mda12	bidir	0	294	0	Z	I/O	5vsa
296	BC_2	*	controlr	0					
297	BC_7	mda11	bidir	0	296	0	Z	I/O	5vsa
298	BC_2	*	controlr	0					
299	BC_7	mpwm18	bidir	0	298	0	Z	I/O	5vsa
300	BC_2	*	controlr	0					
301	BC_7	mpwm17	bidir	0	300	0	Z	I/O	5vsa
302	BC_2	*	controlr	0					
303	BC_7	b_t2clk	bidir	0	302	0	Z	I/O	5vsa
304	BC_2	*	controlr	0					
305	BC_7	b_tpuch1	bidir	0	304	0	Z	I/O	5vsa
306	BC_2	*	controlr	0					
307	BC_7	b_tpuch0	bidir	0	306	0	Z	I/O	5vsa

Table 24-1 Boundary Scan Bit Definition (Continued)

BSDL Bit	Cell Type	Port Name	BSDL Function	saf. val.	cnt. cell	dis. val.	rslt	Pin Function	Pad Type
308	BC_2	*	controlr	0					
309	BC_7	mpwm5_mpio32b6	bidir	0	308	0	Z	I/O	5vsa
310	BC_2	*	controlr	0					
311	BC_7	b_tpuch2	bidir	0	310	0	Z	I/O	5vsa
312	BC_2	*	controlr	0					
313	BC_7	b_tpuch15	bidir	0	312	0	Z	I/O	5vsa
314	BC_2	*	controlr	0					
315	BC_7	b_tpuch14	bidir	0	314	0	Z	I/O	5vsa
316	BC_2	*	controlr	0					
317	BC_7	b_tpuch13	bidir	0	316	0	Z	I/O	5vsa
318	BC_2	*	controlr	0					
319	BC_7	b_tpuch12	bidir	0	318	0	Z	I/O	5vsa
320	BC_2	*	controlr	0					
321	BC_7	b_tpuch11	bidir	0	320	0	Z	I/O	5vsa
322	BC_2	*	controlr	0					
323	BC_7	b_tpuch10	bidir	0	322	0	Z	I/O	5vsa
324	BC_2	*	controlr	0					
325	BC_7	b_tpuch9	bidir	0	324	0	Z	I/O	5vsa
326	BC_2	*	controlr	0					
327	BC_7	b_tpuch8	bidir	0	326	0	Z	I/O	5vsa
328	BC_2	*	controlr	0					
329	BC_7	b_tpuch7	bidir	0	328	0	Z	I/O	5vsa
330	BC_2	*	controlr	0					
331	BC_7	b_tpuch6	bidir	0	330	0	Z	I/O	5vsa
332	BC_2	*	controlr	0					
333	BC_7	b_tpuch5	bidir	0	332	0	Z	I/O	5vsa
334	BC_2	*	controlr	0					
335	BC_7	b_tpuch4	bidir	0	334	0	Z	I/O	5vsa
336	BC_2	*	controlr	0					
337	BC_7	b_tpuch3	bidir	0	336	0	Z	I/O	5vsa
338	BC_2	*	controlr	0					
339	BC_7	a_tpuch1	bidir	0	338	0	Z	I/O	5vsa

Table 24-1 Boundary Scan Bit Definition (Continued)

BSDL Bit	Cell Type	Port Name	BSDL Function	saf. val.	cnt. cell	dis. val.	rslt	Pin Function	Pad Type
340	BC_2	*	controlr	0					
341	BC_7	a_tpuch0	bidir	0	340	0	Z	I/O	5vsa
342	BC_2	*	controlr	0					
343	BC_7	a_t2clk	bidir	0	342	0	Z	I/O	5vsa
344	BC_2	*	controlr	0					
345	BC_7	a_tpuch15	bidir	0	344	0	Z	I/O	5vsa
346	BC_2	*	controlr	0					
347	BC_7	a_tpuch14	bidir	0	346	0	Z	I/O	5vsa
348	BC_2	*	controlr	0					
349	BC_7	a_tpuch13	bidir	0	348	0	Z	I/O	5vsa
350	BC_2	*	controlr	0					
351	BC_7	a_tpuch12	bidir	0	350	0	Z	I/O	5vsa
352	BC_2	*	controlr	0					
353	BC_7	a_tpuch11	bidir	0	352	0	Z	I/O	5vsa
354	BC_2	*	controlr						
355	BC_7	a_tpuch10	bidir	0	354	0	Z	I/O	5vsa
356	BC_2	*	controlr	0					
357	BC_7	a_tpuch9	bidir	0	356	0	Z	I/O	5vsa
358	BC_2	*	controlr	0					
359	BC_7	a_tpuch8	bidir	0	358	0	Z	I/O	5vsa
360	BC_2	*	controlr	0					
361	BC_7	a_tpuch7	bidir	0	360	0	Z	I/O	5vsa
362	BC_2	*	controlr	0					
363	BC_7	a_tpuch6	bidir	0	362	0	Z	I/O	5vsa
364	BC_2	*	controlr	0					
365	BC_7	a_tpuch5	bidir	0	364	0	Z	I/O	5vsa
366	BC_2	*	controlr	0					
367	BC_7	a_tpuch4	bidir	0	366	0	Z	I/O	5vsa
368	BC_2	*	controlr	0					
369	BC_7	a_tpuch3	bidir	0	368	0	Z	I/O	5vsa
370	BC_2	*	controlr	0					
371	BC_7	a_tpuch2	bidir	0	370	0	Z	I/O	5vsa

Table 24-1 Boundary Scan Bit Definition (Continued)

BSDL Bit	Cell Type	Port Name	BSDL Function	saf. val.	cnt. cell	dis. val.	rslt	Pin Function	Pad Type
372	BC_2	*	internal	0					
373	BC_4	etrig1	input	X				I	5vsa
374	BC_2	*	internal	0					
375	BC_4	etrig2	input	X				I	5vsa
376	BC_2	*	controlr	0					
377	BC_7	an64_b_pqb0	bidir	0	376	0	Z	I/O	5vsa
378	BC_2	*	controlr	0					
379	BC_7	an65_b_pqb1	bidir	0	378	0	Z	I/O	5vsa
380	BC_2	*	controlr	0					
381	BC_7	an66_b_pqb2	bidir	0	380	0	Z	I/O	5vsa
382	BC_2	*	controlr	0					
383	BC_7	an67_b_pqb3	bidir	0	382	0	Z	I/O	5vsa
384	BC_2	*	controlr	0					
385	BC_7	an68_b_pqb4	bidir	0	384	0	Z	I/O	5vsa
386	BC_2	*	controlr	0					
387	BC_7	an69_b_pqb5	bidir	0	386	0	Z	I/O	5vsa
388	BC_2	*	controlr	0					
389	BC_7	an70_b_pqb6	bidir	0	388	0	Z	I/O	5vsa
390	BC_2	*	controlr	0					
391	BC_7	an71_b_pqb7	bidir	0	390	0	Z	I/O	5vsa
392	BC_2	*	controlr	0					
393	BC_7	an72_b_ma0_pqa0	bidir	0	392	0	Z	I/O	5vsa
394	BC_2	*	controlr	0					
395	BC_7	an73_b_ma1_pqa1	bidir	0	394	0	Z	I/O	5vsa
396	BC_2	*	controlr	0					
397	BC_7	an74_b_ma2_pqa2	bidir	0	396	0	Z	I/O	5vsa
398	BC_2	*	controlr	0					
399	BC_7	an75_b_pqa3	bidir	0	398	0	Z	I/O	5vsa
400	BC_2	*	controlr	0					
401	BC_7	an76_b_pqa4	bidir	0	400	0	Z	I/O	5vsa
402	BC_2	*	controlr	0					
403	BC_7	an77_b_pqa5	bidir	0	402	0	Z	I/O	5vsa

Table 24-1 Boundary Scan Bit Definition (Continued)

BSDL Bit	Cell Type	Port Name	BSDL Function	saf. val.	cnt. cell	dis. val.	rslt	Pin Function	Pad Type
404	BC_2	*	controlr	0					
405	BC_7	an78_b_pqa6	bidir	0	404	0	Z	I/O	5vsa
406	BC_2	*	controlr	0					
407	BC_7	an79_b_pqa7	bidir	0	406	0	Z	I/O	5vsa
408	BC_2	*	controlr	0					
409	BC_7	an59_a_pqa7	bidir	0	408	0	Z	I/O	5vsa
410	BC_2	*	controlr	0					
411	BC_7	an58_a_pqa6	bidir	0	410	0	Z	I/O	5vsa
412	BC_2	*	controlr	0					
413	BC_7	an57_a_pqa5	bidir	0	412	0	Z	I/O	5vsa
414	BC_2	*	controlr	0					
415	BC_7	an56_a_pqa4	bidir	0	414	0	Z	I/O	5vsa
416	BC_2	*	controlr	0					
417	BC_7	an55_a_pqa3	bidir	0	416	0	Z	I/O	5vsa
418	BC_2	*	controlr	0					
419	BC_7	an54_a_ma2_pqa2	bidir	0	418	0	Z	I/O	5vsa
420	BC_2	*	controlr	0					
421	BC_7	an53_a_ma1_pqa1	bidir	0	420	0	Z	I/O	5vsa
422	BC_2	*	controlr	0					
423	BC_7	an52_a_ma0_pqa0	bidir	0	422	0	Z	I/O	5vsa
424	BC_2	*	controlr	0					
425	BC_7	an51_a_pqb7	bidir	0	424	0	Z	I/O	5vsa
426	BC_2	*	controlr	0					
427	BC_7	an50_a_pqb6	bidir	0	426	0	Z	I/O	5vsa
428	BC_2	*	controlr	0					
429	BC_7	an49_a_pqb5	bidir	0	428	0	Z	I/O	5vsa
430	BC_2	*	controlr	0					
431	BC_7	an48_a_pqb4	bidir	0	430	0	Z	I/O	5vsa
432	BC_2	*	controlr	0					
433	BC_7	an47_anz_a_pqb3	bidir	0	432	0	Z	I/O	5vsa
434	BC_2	*	controlr	0					
435	BC_7	an46_any_a_pqb2	bidir	0	434	0	Z	I/O	5vsa

Table 24-1 Boundary Scan Bit Definition (Continued)

BSDL Bit	Cell Type	Port Name	BSDL Function	saf. val.	cnt. cell	dis. val.	rslt	Pin Function	Pad Type
436	BC_2	*	internal	0					
437	BC_4	an80	input	X				I	5vsa
438	BC_2	*	internal	0					
439	BC_4	an81	input	X				I	5vsa
440	BC_2	*	internal	0					
441	BC_4	an82	input	X				I	5vsa
442	BC_2	*	internal	0					
443	BC_4	an83	input	X				I	5vsa
444	BC_2	*	internal	0					
445	BC_4	an84	input	X				I	5vsa
446	BC_2	*	internal	0					
447	BC_4	an85	input	X				I	5vsa
448	BC_2	*	internal	0					
449	BC_4	an86	input	X				I	5vsa
450	BC_2	*	internal	0					
451	BC_4	an87	input	X				I	5vsa
452	BC_2	*	controlr	0					
453	BC_7	an45_anx_a_pqb1	bidir	0	452	0	Z	I/O	5vsa
454	BC_2	*	controlr	0					
455	BC_7	an44_anw_a_pqb0	bidir	0	454	0	Z	I/O	5vsa
456	BC_2	*	controlr	0					
457	BC_7	b_cnrx0	bidir	0	456	0	Z	I	5vsa
458	BC_2	*	internal	1					
459	BC_2	b_cntx0	output2	1				O	5vfa
460	BC_2	*	controlr	0					
461	BC_7	c_t2clk	bidir	0	460	0	Z	I/O	5vsa
462	BC_2	*	controlr	0					
463	BC_7	c_tpuch15	bidir	0	462	0	Z	I/O	5vsa
464	BC_2	*	controlr	0					
465	BC_7	c_tpuch14	bidir	0	464	0	Z	I/O	5vsa
466	BC_2	*	controlr	0					
467	BC_7	c_tpuch13	bidir	0	466	0	Z	I/O	5vsa

Table 24-1 Boundary Scan Bit Definition (Continued)

BSDL Bit	Cell Type	Port Name	BSDL Function	saf. val.	cnt. cell	dis. val.	rslt	Pin Function	Pad Type
468	BC_2	*	controlr	0					
469	BC_7	c_tpuch12	bidir	0	468	0	Z	I/O	5vsa
470	BC_2	*	controlr	0					
471	BC_7	c_tpuch11	bidir	0	470	0	Z	I/O	5vsa
472	BC_2	*	controlr	0					
473	BC_7	c_tpuch10	bidir	0	472	0	Z	I/O	5vsa
474	BC_2	*	controlr	0					
475	BC_7	c_tpuch9	bidir	0	474	0	Z	I/O	5vsa
476	BC_2	*	controlr	0					
477	BC_7	c_tpuch8	bidir	0	476	0	Z	I/O	5vsa
478	BC_2	*	controlr	0					
479	BC_7	c_tpuch7	bidir	0	478	0	Z	I/O	5vsa
480	BC_2	*	controlr	0					
481	BC_7	c_tpuch6	bidir	0	480	0	Z	I/O	5vsa
482	BC_2	*	controlr	0					
483	BC_7	c_tpuch5	bidir	0	482	0	Z	I/O	5vsa
484	BC_2	*	controlr	0					
485	BC_7	c_tpuch4	bidir	0	484	0	Z	I/O	5vsa
486	BC_2	*	controlr	0					
487	BC_7	c_tpuch3	bidir	0	486	0	Z	I/O	5vsa
488	BC_2	*	controlr	0					
489	BC_7	c_tpuch2	bidir	0	488	0	Z	I/O	5vsa
490	BC_2	*	controlr	0					
491	BC_7	c_tpuch1	bidir	0	490	0	Z	I/O	5vsa
492	BC_2	*	controlr	0					
493	BC_7	c_tpuch0	bidir	0	492	0	Z	I/O	5vsa
494	BC_2	*	controlr						
495	BC_7	mcki	bidir	0	494	0	Z	I	26v
496	BC_2	*	controlr	0					
497	BC_7	mdi_0	bidir	0	496	0	Z	I	26v
498	BC_2	*	controlr	0					
499	BC_7	mdi_1	bidir	0	498	0	Z	I	26v

Table 24-1 Boundary Scan Bit Definition (Continued)

BSDL Bit	Cell Type	Port Name	BSDL Function	saf. val.	cnt. cell	dis. val.	rslt	Pin Function	Pad Type
500	BC_2	*	controlr	0					
501	BC_7	msei_b	bidir	0	500	0	Z	I	26v
502	BC_2	*	internal	1					
503	BC_2	mdo_1	output2	1				O	26v
504	BC_2	*	internal	1					
505	BC_2	mdo_0	output2	1				O	26v
506	BC_2	*	internal	1					
507	BC_2	mcko	output2	1				I/O	26v
508	BC_2	*	controlr	0					
509	BC_7	mdo_7_mpio32b7	bidir	0	508	0	Z	I/O	26v5vs
510	BC_2	*	controlr	0					
511	BC_7	mdo_6_mpio32b8	bidir	0	510	0	Z	I/O	26v5vs
512	BC_2	*	controlr	0					
513	BC_7	mdo_5_mpio32b9	bidir	0	512	0	Z	I/O	26v5vs
514	BC_2	*	controlr	0					
515	BC_7	mdo_4_mpio32b10	bidir	0	514	0	Z	I/O	26v5vs
516	BC_2	*	internal	0					
517	BC_4	*[rsti_b force to 0]	internal	0				I	26v
518	BC_2	*	controlr	0					
519	BC_7	evti_b	bidir	0	518	0	Z	I	26v

Notes:

1. Bi-state outputs (Pin Function = O) such as mdo_2, and mdo_3, are incorporated with general I/O pads hard-wired to keep output enable always on in system mode. The JTAG Control cell, indicated by the next lower bsdl bit in the chain, is configured as an “internal” only cell to be held at a “1” value (always driving out) during JTAG testing.
2. Some input-only cells made with generic I/O pads are configured with “internal” control cells to keep them always in input mode, such as epee, b0epee, and input pins that may be attached to analog references. Other input-only cells are configured as bidirectional for JTAG testing, to give the board-level ATPG tools the flexability to use the pad as an input or output, depending on the network of other devices that the pin is connected too. If it is desired to restrict these pins to only act as receivers during JTAG mode, then these JTAG bsdl entries can be converted as shown in the example below:

3. This description allows ATPG tools to use a pin as a driver or receiver:

188	BC_2	*	controlr	0					
189	BC_7	irq6_b_modck2	bidir	0	188	0	Z	I	26v

4. A modification to restrict ATPG tools to use a functional input-only pin as an input receiver only:

188	BC_2	*	internal	0					
189	BC_4	irq6_b_modck2	input	X				I	26v

5. The $\overline{\text{PORESET}}$, $\overline{\text{HRESET}}$, and $\overline{\text{SRESET}}$ pins are not part of the JTAG boundary scan chain. These pins are used in the reset configuration to enter JTAG. Board-level connections to them will not be testable with the EXTEST and CLAMP instructions. They do respond to the HI-Z JTAG instruction for parametric testing purposes.
6. The XTAL, EXTAL, and XFC pins are associated with analog signals and are excluded from the boundary scan chain.
7. The READI module reset pin, rsti_b, (bsdl pin 517) is in the JTAG boundary scan chain, but must be kept at a “0” level during JTAG testing, (except for Hi-Z testing), due to system interactions. It is classified as a “linkage” pin, and its data and control cells are configured to advise ATPG tools to drive a “0” value in during JTAG testing.
8. Pad type naming conventions:
   - 26 V – 2.6 V
   - 5 V – 5 V
   - s – slow
   - f – fast
   - h – high drive
   - a – analog input
   - i – input only
   - d – has direct connection to the pad (may be used for module test)
   - r – resized cell instance
9. Column Descriptions:
   - Columns 1 through 8 are entries from the boundary-scan description from the BSDL file. The columns and formats for each of these entries are defined in the *IEEE Std. 1149.1b-1994 Supplement to the IEEE Std. 1149.1-1990, IEEE Standard Test Access Port and Boundary-Scan Architecture* document. Descriptions of these columns are described below:
   - Column 1: Defines the bit’s ordinal position in the boundary scan register. The shift register cell nearest TDO (i.e., first to be shifted in) is defined as bit 0; the last bit to be shifted in is 519.
   - Column 2: References one of the three standard JTAG Cell Types (BC_4, BC_2, and BC_7) that are used for this JTAG cell in the MPC565 / MPC566. See the *IEEE Std. 1149.1-1990, IEEE Standard Test Access Port and Boundary-Scan Architecture* document for further description of these standard cell types.

- Column 3: Lists the pin name (also called the PortID) for all pin-related cells. For JTAG control cells or data cells that have been designated as “internal”, an asterisk, is shown in this column.
- Column 4: Lists the BSDL pin function.
- Column 5: The “safe bit” column specifies the value that should be loaded into the capture (and update) flip-flop of a given cell when board-level test generation software might otherwise choose a value randomly.
- Column 6: The “control cell” column identifies the cell number of the control cell that is associated with this data cell, and can disable its output.
- Column 7: The “disable value” column gives the value that must be scanned into the control cell identified by the previous “control cell” (column 6) to disable the port named by the relevant portID.
- Column 8: The “disable result” column identifies a given signal value of the PortID if that signal can be disabled. The values shown specifies the condition of the driver of that signal when it is disabled.
- Column 9: The “pin function” column indicates the normal system pin directionality. (– Input Only Pin, O – Output Only Pin, I/O – Bidirectional I/O pin)
- Column 10: The pad type column describes relevant characteristics about each pad type. See the Pad Type Keys in Note 5 above.

24.2.2 Instruction Register

The MPC565 / MPC566 JTAG implementation includes the public instructions (EXTEST, SAMPLE/PRELOAD, and BYPASS), and also supports the CLAMP instruction. One additional public instruction (HI-Z) provides the capability for disabling all device output drivers. The MPC565 / MPC566 includes a 4-bit instruction register without parity consisting of a shift register with four parallel outputs. Data is transferred from the shift register to the parallel outputs during the update-IR controller state. The four bits are used to decode the five unique instructions listed in.

Table 24-2 Instruction Decoding

Code				
B3	B2	B1	B0	Instruction
0	0	0	0	EXTEST
0	0	0	1	SAMPLE/PRELOAD
0	X	1	X	BYPASS
0	1	0	0	HI-Z
0	1	0	1	CLAMP and BYPASS

NOTE

B0 (LSB) is shifted first.

The parallel output of the instruction register is reset to all ones in the test-logic-reset controller state.

NOTE

This preset state is equivalent to the BYPASS instruction.

During the capture-IR controller state, the parallel inputs to the instruction shift register are loaded with the CLAMP command code.

24.2.2.1 EXTEST

The external test (EXTEST) instruction selects the 520-bit boundary scan register. EXTEST also asserts internal reset for the MPC565 / MPC566 system logic to force a predictable beginning internal state while performing external boundary scan operations.

By using the TAP, the register is capable of:

a. scanning user-defined values into the output buffers

b. capturing values presented to input pins

c. controlling the output drive of three-state output or bidirectional pins

24.2.2.2 SAMPLE/PRELOAD

The SAMPLE/PRELOAD instruction initializes the boundary scan register output cells prior to selection of EXTEST. This initialization ensures that known data will appear on the outputs when entering the EXTEST instruction. The SAMPLE/PRELOAD instruction also provides a means to obtain a snapshot of system data and control signals.

NOTE

Since there is no internal synchronization between the scan chain clock (TCK) and the system clock (CLKOUT), there must be provision of some form of external synchronization to achieve meaningful results.

24.2.2.3 BYPASS

The BYPASS instruction selects the single-bit bypass register as shown in **Figure 24-4**. This creates a shift register path from TDI to the bypass register and, finally, to TDO, circumventing the 520-bit boundary scan register. This instruction is used to enhance test efficiency when a component other than the MPC565 / MPC566 becomes the device under test.

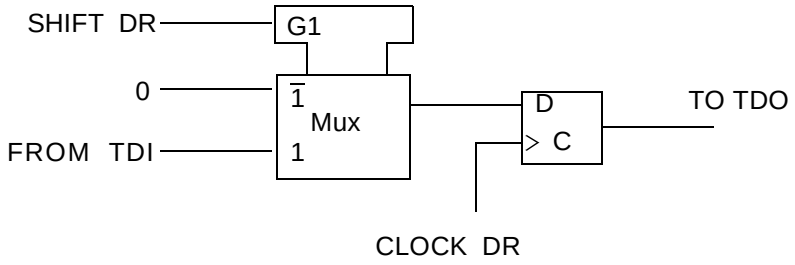

Figure 24-4 Bypass Register

When the bypass register is selected by the current instruction, the shift register stage is set to a logic zero on the rising edge of TCK in the capture-DR controller state. Therefore, the first bit to be shifted out after selecting the bypass register will always be a logic zero.

24.2.2.4 CLAMP

The CLAMP instruction selects the single-bit bypass register as shown in **Figure 24-4**, and the state of all signals driven from system output pins is completely defined by the data previously shifted into the boundary scan register (for example, using the SAMPLE/PRELOAD instruction).

24.2.3 HI-Z

The HI-Z instruction is provided as a manufacturer's optional public instruction to prevent having to backdrive the output pins during circuit-board testing. When HI-Z is invoked, all output drivers, including the two-state drivers, are turned off (i.e., high impedance). The instruction selects the bypass register.

24.3 MPC565 / MPC566 Restrictions

The control afforded by the output enable signals using the boundary scan register and the EXTEST instruction requires a compatible circuit-board test environment to avoid device-destructive configurations. The user must avoid situations in which the MPC565 / MPC566 output drivers are enabled into actively driven networks.

The MPC565 / MPC566 features a low-power stop mode. The interaction of the scan chain interface with low-power stop mode is as follows:

1. The TAP controller must be in the test-logic-reset state to either enter or remain in the low-power stop mode. Leaving the TAP controller in the test-logic-reset state negates the ability to achieve low-power, but does not otherwise affect device functionality.
2. The TCK input is not blocked in low-power stop mode. To consume minimal power, the TCK input should be externally connected to V_{DD} or ground.

3. The TMS, TDI, $\overline{\text{TRST}}$ pins include on-chip pullup resistors. In low-power stop mode, these three pins should remain either unconnected or connected to V_{DD} to achieve minimal power consumption. Note that for proper reset of the scan chain test logic, the best approach is to pull active $\overline{\text{TRST}}$ at power on reset (PORESET).

24.3.1 Non-Scan Chain Operation

In non-scan chain operation, there are two constraints. First, the TCK input does not include an internal pullup resistor and should not be left unconnected to preclude mid-level inputs. The second constraint is to ensure that the scan chain test logic is kept transparent to the system logic by forcing TAP into the test-logic-reset controller state, using either of two methods. Connecting pin $\overline{\text{TRST}}$ to logic 0 (or one of the reset pins), or TMS must be sampled as a logic one for five consecutive TCK rising edges. If then TMS either remains unconnected or is connected to V_{DD}, then the TAP controller cannot leave the test-logic-reset state, regardless of the state of TCK.

24.3.2 Motorola MPC565 / MPC566 BSDL Description

```
entity spanishoak is
 generic(PHYSICAL_PIN_MAP: string := “DIE”);
port(
a_cnrx0: inout bit;
 a_cntx0: buffer bit;
 a_eck: in bit;
 a_miso_qgpio4: inout bit;
 a_mosi_qgpio5: inout bit;
 a_pcs0_ss_b_qgpio0: inout bit;
 a_pcs1_qgpio1: inout bit;
 a_pcs2_qgpio2: inout bit;
 a_pcs3_qgpio3: inout bit;
 a_rxd1_qgpi1: in bit;
 a_rxd2_qgpi2: in bit;
 a_sck_qgpio6: inout bit;
 a_t2clk: inout bit;
 a_tpuch0: inout bit;
 a_tpuch1: inout bit;
 a_tpuch10: inout bit;
 a_tpuch11: inout bit;
 a_tpuch12: inout bit;
 a_tpuch13: inout bit;
 a_tpuch14: inout bit;
 a_tpuch15: inout bit;
 a_tpuch2: inout bit;
 a_tpuch3: inout bit;
 a_tpuch4: inout bit;
 a_tpuch5: inout bit;
 a_tpuch6: inout bit;
```

```
a_tpuch7: inout bit;
a_tpuch8: inout bit;
a_tpuch9: inout bit;
a_txd1_qgpo1: buffer bit;
a_txd2_qgpo2: buffer bit;
addr_sgpioa: inout bit_vector(8 to 31);
an44_anw_a_pqb0: inout bit;
an45_anx_a_pqb1: inout bit;
an46_any_a_pqb2: inout bit;
an47_anz_a_pqb3: inout bit;
an48_a_pqb4: inout bit;
an49_a_pqb5: inout bit;
an50_a_pqb6: inout bit;
an51_a_pqb7: inout bit;
an52_a_ma0_pqa0: inout bit;
an53_a_ma1_pqa1: inout bit;
an54_a_ma2_pqa2: inout bit;
an55_a_pqa3: inout bit;
an56_a_pqa4: inout bit;
an57_a_pqa5: inout bit;
an58_a_pqa6: inout bit;
an59_a_pqa7: inout bit;
an64_b_pqb0: inout bit;
an65_b_pqb1: inout bit;
an66_b_pqb2: inout bit;
an67_b_pqb3: inout bit;
an68_b_pqb4: inout bit;
an69_b_pqb5: inout bit;
an70_b_pqb6: inout bit;
an71_b_pqb7: inout bit;
```

```
an72_b_ma0_pqa0: inout bit;
an73_b_ma1_pqa1: inout bit;
an74_b_ma2_pqa2: inout bit;
an75_b_pqa3: inout bit;
an76_b_pqa4: inout bit;
an77_b_pqa5: inout bit;
an78_b_pqa6: inout bit;
an79_b_pqa7: inout bit;
an80: in bit;
an81: in bit;
an82: in bit;
an83: in bit;
an84: in bit;
an85: in bit;
an86: in bit;
an87: in bit;
b0epee: in bit;
b_cnrx0: inout bit;
b_cntx0: buffer bit;
b_eck: in bit;
b_miso_qgpio4: inout bit;
b_mosi_qgpio5: inout bit;
b_pcs0_ss_b_qgpio0: inout bit;
b_pcs1_qgpio1: inout bit;
b_pcs2_qgpio2: inout bit;
b_pcs3_j1850_tx: inout bit;
b_rxd1_qgpi1: in bit;
b_rxd2_j1850_rx: in bit;
b_sck_qgpio6: inout bit;
b_t2clk: inout bit;
```

```
b_tpuch0: inout bit;
b_tpuch1: inout bit;
b_tpuch10: inout bit;
b_tpuch11: inout bit;
b_tpuch12: inout bit;
b_tpuch13: inout bit;
b_tpuch14: inout bit;
b_tpuch15: inout bit;
b_tpuch2: inout bit;
b_tpuch3: inout bit;
b_tpuch4: inout bit;
b_tpuch5: inout bit;
b_tpuch6: inout bit;
b_tpuch7: inout bit;
b_tpuch8: inout bit;
b_tpuch9: inout bit;
b_txd1_qgpo1: buffer bit;
b_txd2_qgpo2: buffer bit;
bb_b_vf2_iwp3: inout bit;
bdip_b: inout bit;
bg_b_vf0_lwp1: inout bit;
bi_b_sts_b: inout bit;
br_b_vf1_iwp2: inout bit;
burst_b: inout bit;
c_cnrx0_mpio32b14: inout bit;
c_cntx0_mpio32b13: inout bit;
c_t2clk: inout bit;
c_tpuch0: inout bit;
c_tpuch1: inout bit;
c_tpuch10: inout bit;
```

```
c_tpuch11: inout bit;
c_tpuch12: inout bit;
c_tpuch13: inout bit;
c_tpuch14: inout bit;
c_tpuch15: inout bit;
c_tpuch2: inout bit;
c_tpuch3: inout bit;
c_tpuch4: inout bit;
c_tpuch5: inout bit;
c_tpuch6: inout bit;
c_tpuch7: inout bit;
c_tpuch8: inout bit;
c_tpuch9: inout bit;
clkout: buffer bit;
clockout1: linkage bit;
cs0_b: buffer bit;
cs1_b: buffer bit;
cs2_b: buffer bit;
cs3_b: buffer bit;
d_corner: linkage bit;
data_sgpiod: inout bit_vector(0 to 31);
engclk_buclk: buffer bit;
epee: in bit;
etrig1: in bit;
etrig2: in bit;
evti_b: inout bit;
extal: linkage bit;
extclk: in bit;
hreset_b: linkage bit;
irq0_b_sgpioc0: inout bit;
```

```
irq1_b_rsv_b_sgpioc1: inout bit;
irq2_b_cr_b_sgpioc2: inout bit;
irq3_b_kr_b_retry_b_sgpioc3: inout bit;
irq4_b_at2_sgpioc4: inout bit;
irq5_b_sgpioc5_modck1: inout bit;
irq6_b_modck2: inout bit;
irq7_b_modck3: inout bit;
iwp0_vfls0: buffer bit;
iwp1_vfls1: buffer bit;
kapwr: linkage bit;
l_corner: linkage bit;
mcki: inout bit;
mcko: buffer bit;
mda11: inout bit;
mda12: inout bit;
mda13: inout bit;
mda14: inout bit;
mda15: inout bit;
mda27: inout bit;
mda28: inout bit;
mda29: inout bit;
mda30: inout bit;
mda31: inout bit;
mdi_0: inout bit;
mdi_1: inout bit;
mdo_0: buffer bit;
mdo_1: buffer bit;
mdo_2: buffer bit;
mdo_3: buffer bit;
mdo_4_mpio32b10: inout bit;
```

```
mdo_5_mpio32b9: inout bit;
mdo_6_mpio32b8: inout bit;
mdo_7_mpio32b7: inout bit;
mpio32b15: inout bit;
mpwm0: inout bit;
mpwm1: inout bit;
mpwm16: inout bit;
mpwm17: inout bit;
mpwm18: inout bit;
mpwm19: inout bit;
mpwm2: inout bit;
mpwm20_mpio32b11: inout bit;
mpwm21_mpio32b12: inout bit;
mpwm3: inout bit;
mpwm4_mpio32b5: inout bit;
mpwm5_mpio32b6: inout bit;
mrtc_extal32: linkage bit;
mrtc_xtal32: linkage bit;
msei_b: inout bit;
mseo_b: buffer bit;
nvddl: linkage bit_vector(1 to 19);
oe_b: buffer bit;
poreset_b: linkage bit;
qvddl: linkage bit_vector(1 to 12);
r_corner: linkage bit;
rd_wr_b: inout bit;
rstconf_b_texp: inout bit;
rsti_b: linkage bit;
sgpioc6_frz_ptr_b: inout bit;
sgpioc7_irqout_b_lwp0: inout bit;
```

```
sreset_b: linkage bit;

ta_b: inout bit;

tck_dsck: in bit;

tdi_dsdi: in bit;

tdo_dsdo: out bit;

tea_b: inout bit;

tms: in bit;

trst_b: in bit;

ts_b: inout bit;

tsiz0: inout bit;

tsiz1: inout bit;

u_corner: linkage bit;

vdd: linkage bit_vector(1 to 6);

vdda: linkage bit;

vddf: linkage bit;

vddh: linkage bit_vector(1 to 6);

vddsram1: linkage bit;

vddsram2: linkage bit;

vddsram3: linkage bit;

vddsyn: linkage bit;

vddsyn32: linkage bit;

vf0_mpio32b0: inout bit;

vf1_mpio32b1: inout bit;

vf2_mpio32b2: inout bit;

vflash: linkage bit;

vfls0_mpio32b3: inout bit;

vfls1_mpio32b4: inout bit;

vrh: linkage bit;

vrhaltref: linkage bit;

vrl: linkage bit;
```

```
vss: linkage bit_vector(1 to 28);
vssa: linkage bit;
vssf: linkage bit;
vsssyn: linkage bit;
vsssyn32: linkage bit;
we_b_at: buffer bit_vector(0 to 3);
xfc: linkage bit;
xtal: linkage bit);
  use STD_1149_1_1994.all;
   attribute COMPONENT_CONFORMANCE of SpanishOak: entity is
      “STD_1149_1_1993”;   -- complies with Std. 1149.1a-1993
 attribute PIN_MAP of SpanishOak: entity is PHYSICAL_PIN_MAP;
constant DIE: PIN_MAP_STRING :=

“a_cnrx0:1,” &
“a_cntx0:2,” &
“a_eck:3,” &
“a_miso_qgpio4:4,” &
“a_mosi_qgpio5:5,” &
“a_pcs0_ss_b_qgpio0:6,” &
“a_pcs1_qgpio1:7,” &
“a_pcs2_qgpio2:8,” &
“a_pcs3_qgpio3:9,” &
“a_rxd1_qgpi1:10,” &
“a_rxd2_qgpi2:11,” &
“a_sck_qgpio6:12,” &
“a_t2clk:13,” &
“a_tpuch0:14,” &
“a_tpuch1:15,” &
“a_tpuch10:16,” &
```

```
 “a_tpuch11:17,” &
 “a_tpuch12:18,” &
 “a_tpuch13:19,” &
 “a_tpuch14:20,” &
 “a_tpuch15:21,” &
 “a_tpuch2:22,” &
 “a_tpuch3:23,” &
 “a_tpuch4:24,” &
 “a_tpuch5:25,” &
 “a_tpuch6:26,” &
 “a_tpuch7:27,” &
 “a_tpuch8:28,” &
 “a_tpuch9:29,” &
 “a_txd1_qgpo1:30,” &
 “a_txd2_qgpo2:31,” &

“addr_sgpioa:(32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,5
0,51,52,53,54,55),” &
 “an44_anw_a_pqb0:56,” &
 “an45_anx_a_pqb1:57,” &
 “an46_any_a_pqb2:58,” &
 “an47_anz_a_pqb3:59,” &
 “an48_a_pqb4:60,” &
 “an49_a_pqb5:61,” &
 “an50_a_pqb6:62,” &
 “an51_a_pqb7:63,” &
 “an52_a_ma0_pqa0:64,” &
 “an53_a_ma1_pqa1:65,” &
 “an54_a_ma2_pqa2:66,” &
 “an55_a_pqa3:67,” &
 “an56_a_pqa4:68,” &
```

```
“an57_a_pqa5:69,” &
“an58_a_pqa6:70,” &
“an59_a_pqa7:71,” &
“an64_b_pqb0:72,” &
“an65_b_pqb1:73,” &
“an66_b_pqb2:74,” &
“an67_b_pqb3:75,” &
“an68_b_pqb4:76,” &
“an69_b_pqb5:77,” &
“an70_b_pqb6:78,” &
“an71_b_pqb7:79,” &
“an72_b_ma0_pqa0:80,” &
“an73_b_ma1_pqa1:81,” &
“an74_b_ma2_pqa2:82,” &
“an75_b_pqa3:83,” &
“an76_b_pqa4:84,” &
“an77_b_pqa5:85,” &
“an78_b_pqa6:86,” &
“an79_b_pqa7:87,” &
“an80:88,” &
“an81:89,” &
“an82:90,” &
“an83:91,” &
“an84:92,” &
“an85:93,” &
“an86:94,” &
“an87:95,” &
“b0epee:96,” &
“b_cnrx0:97,” &
“b_cntx0:98,” &
```

```
“b_eck:99,” &
“b_miso_qgpio4:100,” &
“b_mosi_qgpio5:101,” &
“b_pcs0_ss_b_qgpio0:102,” &
“b_pcs1_qgpio1:103,” &
“b_pcs2_qgpio2:104,” &
“b_pcs3_j1850_tx:105,” &
“b_rxd1_qgpi1:106,” &
“b_rxd2_j1850_rx:107,” &
“b_sck_qgpio6:108,” &
“b_t2clk:109,” &
“b_tpuch0:110,” &
“b_tpuch1:111,” &
“b_tpuch10:112,” &
“b_tpuch11:113,” &
“b_tpuch12:114,” &
“b_tpuch13:115,” &
“b_tpuch14:116,” &
“b_tpuch15:117,” &
“b_tpuch2:118,” &
“b_tpuch3:119,” &
“b_tpuch4:120,” &
“b_tpuch5:121,” &
“b_tpuch6:122,” &
“b_tpuch7:123,” &
“b_tpuch8:124,” &
“b_tpuch9:125,” &
“b_txd1_qgpo1:126,” &
“b_txd2_qgpo2:127,” &
“bb_b_vf2_iwp3:128,” &
```

```
“bdip_b:129,” &
“bg_b_vf0_lwp1:130,” &
“bi_b_sts_b:131,” &
“br_b_vf1_iwp2:132,” &
“burst_b:133,” &
“c_cnrx0_mpio32b14:134,” &
“c_cntx0_mpio32b13:135,” &
“c_t2clk:136,” &
“c_tpuch0:137,” &
“c_tpuch1:138,” &
“c_tpuch10:139,” &
“c_tpuch11:140,” &
“c_tpuch12:141,” &
“c_tpuch13:142,” &
“c_tpuch14:143,” &
“c_tpuch15:144,” &
“c_tpuch2:145,” &
“c_tpuch3:146,” &
“c_tpuch4:147,” &
“c_tpuch5:148,” &
“c_tpuch6:149,” &
“c_tpuch7:150,” &
“c_tpuch8:151,” &
“c_tpuch9:152,” &
“clkout:153,” &
“clockout1:154,” &
“cs0_b:155,” &
“cs1_b:156,” &
“cs2_b:157,” &
“cs3_b:158,” &
```

```
 “d_corner:159,” &

“data_sgpiod:(160,161,162,163,164,165,166,167,168,169,170,171,17,173,
174,175,176,177,178,179,180,181,182,183,184,185,186,187,188,189,190,1
91),” &

 “engclk_buclk:192,” &

 “epee:193,” &

 “etrig1:194,” &

 “etrig2:195,” &

 “evti_b:196,” &

 “extal:197,” &

 “extclk:198,” &

 “hreset_b:199,” &

 “irq0_b_sgpioc0:200,” &

 “irq1_b_rsv_b_sgpioc1:201,” &

 “irq2_b_cr_b_sgpioc2:202,” &

 “irq3_b_kr_b_retry_b_sgpioc3:203,” &

 “irq4_b_at2_sgpioc4:204,” &

 “irq5_b_sgpioc5_modck1:205,” &

 “irq6_b_modck2:206,” &

 “irq7_b_modck3:207,” &

 “iwp0_vfls0:208,” &

 “iwp1_vfls1:209,” &

 “kapwr:210,” &

 “l_corner:211,” &

 “mcki:212,” &

 “mcko:213,” &

 “mda11:214,” &

 “mda12:215,” &

 “mda13:216,” &

 “mda14:217,” &

 “mda15:218,” &
```

```
“mda27:219,” &
“mda28:220,” &
“mda29:221,” &
“mda30:222,” &
“mda31:223,” &
“mdi_0:224,” &
“mdi_1:225,” &
“mdo_0:226,” &
“mdo_1:227,” &
“mdo_2:228,” &
“mdo_3:229,” &
“mdo_4_mpio32b10:230,” &
“mdo_5_mpio32b9:231,” &
“mdo_6_mpio32b8:232,” &
“mdo_7_mpio32b7:233,” &
“mpio32b15:234,” &
“mpwm0:235,” &
“mpwm1:236,” &
“mpwm16:237,” &
“mpwm17:238,” &
“mpwm18:239,” &
“mpwm19:240,” &
“mpwm2:241,” &
“mpwm20_mpio32b11:242,” &
“mpwm21_mpio32b12:243,” &
“mpwm3:244,” &
“mpwm4_mpio32b5:245,” &
“mpwm5_mpio32b6:246,” &
“mrtc_extal32:247,” &
“mrtc_xtal32:248,” &
```

```
 “msei_b:249,” &

 “mseo_b:250,” &

“nvddl:(251,252,253,254,255,256,257,258,259,260,261,262,263,264,265,2
66,267,268,269),” &

 “oe_b:270,” &

 “poreset_b:271,” &

 “qvddl:(272,273,274,275,276,277,278,279,280,281,282,283),” &

 “r_corner:284,” &

 “rd_wr_b:285,” &

 “rstconf_b_texp:286,” &

 “rsti_b:287,” &

 “sgpioc6_frz_ptr_b:288,” &

 “sgpioc7_irqout_b_lwp0:289,” &

 “sreset_b:290,” &

 “ta_b:291,” &

 “tck_dsck:292,” &

 “tdi_dsdi:293,” &

 “tdo_dsdo:294,” &

 “tea_b:295,” &

 “tms:296,” &

 “trst_b:297,” &

 “ts_b:298,” &

 “tsiz0:299,” &

 “tsiz1:300,” &

 “u_corner:301,” &

 “vdd:(302,303,304,305,306,307),” &

 “vdda:308,” &

 “vddf:309,” &

 “vddh:(310,311,312,313,314,315),” &

 “vddsram1:316,” &
```

```
"vddsram2:317," &
"vddsram3:318," &
"vddsyn:319," &
"vddsyn32:320," &
"vf0_mpio32b0:321," &
"vf1_mpio32b1:322," &
"vf2_mpio32b2:323," &
"vflash:324," &
"vfls0_mpio32b3:325," &
"vfls1_mpio32b4:326," &
"vrh:327," &
"vrhaltref:328," &
"vrl:329," &

"vss:(330,331,332,333,334,335,336,337,338,339,340,341,342,343,344,345
,346,347,348,349,350,351,352,353,354,355,356,357)," &
"vssa:358," &
"vssf:359," &
"vsssyn:360," &
"vsssyn32:361," &
"we_b_at:(362,363,364,365)," &
"xfc:366," &
"xtal:367";

attribute TAP_SCAN_IN    of tdi_dsdi:  signal is true;
attribute TAP_SCAN_MODE  of tms:  signal is true;
attribute TAP_SCAN_OUT   of tdo_dsdo:  signal is true;
attribute TAP_SCAN_RESET of trst_b:  signal is true;
attribute TAP_SCAN_CLOCK of tck_dsck:  signal is (20.0e6, BOTH);

attribute INSTRUCTION_LENGTH of SpanishOak: entity is 4;
```

```
attribute INSTRUCTION_OPCODE of SpanishOak: entity is
   “EXTEST   (0000),”  &
   “SAMPLE   (0001),”  &
   “CLAMP    (0101),”  &
   “HIGHZ    (0100),”  &
   “BYPASS   (0111, 0010, 0110, 0011)”;

attribute INSTRUCTION_CAPTURE of SpanishOak: entity is “0101”;

attribute BOUNDARY_LENGTH of SpanishOak: entity is 520;

attribute BOUNDARY_REGISTER of SpanishOak: entity is
--  numcellport functionsafe[ccell disval rslt]
“0( BC_2,* ,internal, 1), “ &
  “1( BC_2,mdo_2,output2 , 1), “ &

“2( BC_2,* ,internal, 1),“ &
“3 ( BC_2,mdo_3,output2 , 1),“ &

“4 ( BC_2,* ,internal, 1), “ &
“5 ( BC_2,mseo_b,output2 , 1), “ &

“6 ( BC_2,* ,internal, 1),“ &
“7 ( BC_2,iwp0_vfls0,output2 , 1),“ &

“8 ( BC_2,* ,internal, 1),“ &
“9( BC_2,iwp1_vfls1,output2 , 1),“ &

“10 ( BC_2,*,controlr, 0),“ &
“11( BC_7,addr_sgpioa(16),bidir, 0, 10, 0, Z),“ &
```

```
“12 ( BC_2,* ,controlr, 0),“ &
“13 ( BC_7, addr_sgpioa(17),bidir, 0, 12, 0, Z),“ &
“14 ( BC_2,* ,controlr, 0),   “ &
“15 ( BC_7,sgpioc6_frz_ptr_b,bidir, 0, 14, 0, Z),“ &
“16 ( BC_2,  * ,controlr, 0),   “ &
“17 ( BC_7, addr_sgpioa(8),bidir, 0, 16, 0, Z),“ &

“18 ( BC_2,* ,controlr, 0),“ &
“19 ( BC_7,addr_sgpioa(18),bidir, 0, 18, 0, Z),“ &

“20 ( BC_2,* ,controlr, 0),“ &
“21 ( BC_7,addr_sgpioa(19),bidir, 0, 20, 0, Z),“ &

“22 ( BC_2,* ,controlr, 0),   “ &
“23 ( BC_7,addr_sgpioa(9),bidir, 0, 22, 0, Z),“ &

“24 ( BC_2,* ,controlr, 0),   “ &
“25 ( BC_7,addr_sgpioa(10),bidir, 0, 24, 0, Z),“ &

“26 ( BC_2,* ,controlr, 0),   “ &
“27 ( BC_7,addr_sgpioa(20),bidir, 0, 26, 0, Z),“ &

“28 ( BC_2,* ,controlr, 0),   “ &
“29 ( BC_7,addr_sgpioa(21),bidir, 0, 28, 0, Z),“ &

“30 ( BC_2,* ,controlr, 0),   “ &
“31 ( BC_7,addr_sgpioa(11),bidir, 0, 30, 0, Z),“ &

“32 ( BC_2,* ,controlr, 0),   “ &
```

```
“33 ( BC_7,  addr_sgpioa(12),bidir, 0, 32, 0, Z),“ &

“34 ( BC_2,* ,controlr, 0),   “ &
“35 ( BC_7,  addr_sgpioa(22),bidir, 0, 34, 0, Z),“ &

“36 ( BC_2,* ,controlr, 0),   “ &
“37 ( BC_7,  addr_sgpioa(23),bidir, 0, 36, 0, Z),“ &

“38 ( BC_2,* ,controlr, 0),   “ &
“39 ( BC_7,addr_sgpioa(13),bidir, 0, 38, 0, Z),“ &

“40 ( BC_2,* ,controlr, 0),   “ &
“41 ( BC_7,  addr_sgpioa(24),bidir, 0, 40, 0, Z),“ &

“42 ( BC_2,  * ,controlr, 0),   “ &
“43 ( BC_7,  addr_sgpioa(25),bidir, 0, 42, 0, Z),“ &

“44( BC_2,  * ,controlr, 0),   “ &
“45( BC_7,  addr_sgpioa(14),bidir, 0, 44, 0, Z),“ &

“46 ( BC_2,  *,controlr, 0),   “ &
“47 ( BC_7,  addr_sgpioa(15),bidir, 0, 46, 0, Z),“ &

“48 ( BC_2,  *,controlr,0),   “ &
“49 ( BC_7,  addr_sgpioa(30),bidir, 0, 48, 0, Z),“ &

“50 ( BC_2,  * ,controlr, 0),   “ &
“51 ( BC_7,  addr_sgpioa(26),bidir, 0, 50, 0, Z),“ &

“52 ( BC_2,  * ,controlr, 0),   “ &
```

```
“53 ( BC_7,  addr_sgpioa(27),bidir, 0, 52, 0, Z)“ &

“54 ( BC_2,  * ,controlr, 0),   “ &
“55 ( BC_7,  addr_sgpioa(31),bidir, 0, 54, 0, Z),“ &

“56 ( BC_2,  * ,controlr, 0),   “ &
“57 ( BC_7,  addr_sgpioa(28),bidir, 0, 56, 0, Z),“ &

“58 ( BC_2,  * ,controlr, 0),   “ &
“59 ( BC_7,  addr_sgpioa(29),bidir, 0 ,58, 0, Z),“ &

“60 ( BC_2,  * ,controlr, 0),   “ &
“61 ( BC_7,  data_sgpiod(0),bidir, 0, 60, 0, Z),“ &

“62 ( BC_2,  * ,controlr, 0),   “ &
“63 ( BC_7,  data_sgpiod(29),bidir, 0, 62, 0, Z),“ &

“64 ( BC_2,  * ,controlr, 0),   “ &
“65 ( BC_7,  data_sgpiod(1),bidir, 0, 64, 0, Z),“ &

“66 ( BC_2,  * ,controlr, 0),   “ &
“67 ( BC_7,  data_sgpiod(2),bidir, 0, 66, 0, Z),“ &

“68 ( BC_2,  *,controlr, 0),   “ &
“69 ( BC_7,  data_sgpiod(3),bidir, 0, 68, 0, Z),“ &

“70 ( BC_2,  * ,controlr, 0),   “ &
“71 ( BC_7,  data_sgpiod(27),bidir, 0, 70, 0, Z),“ &

“72 ( BC_2,  * ,controlr, 0),   “ &
```

```
“73 ( BC_7,  data_sgpiod(4),bidir, 0, 72, 0, Z),“ &

“74 ( BC_2,  * ,controlr, 0),   “ &
“75 ( BC_7,  data_sgpiod(28),bidir, 0, 74, 0, Z),“ &

“76 ( BC_2,  * ,controlr, 0),   “ &
“77 ( BC_7,  data_sgpiod(31),bidir, 0, 76, 0, Z),“ &

“78( BC_2,*,controlr, 0),   “ &
“79( BC_7, data_sgpiod(5),bidir, 0 78, 0, Z),“ &

“80 ( BC_2,  * ,controlr, 0),   “ &
“81 ( BC_7,  data_sgpiod(6),bidir, 0, 80, 0, Z),“ &

“82 ( BC_2,  * ,controlr, 0),   “ &
“83 ( BC_7,  data_sgpiod(30),bidir, 0, 82, 0, Z),“ &

“84 ( BC_2,  * ,controlr, 0),   “ &
“85 ( BC_7,  data_sgpiod(7),bidir, 0, 84 0 Z),“ &

“86 ( BC_2,  * ,controlr,0),   “ &
“87 ( BC_7,  data_sgpiod(25),bidir, 0, 86, 0, Z),“ &

“88 ( BC_2,  * ,controlr, 0),   “ &
“89 ( BC_7,  data_sgpiod(8),bidir, 0, 88, 0, Z),“ &

“90 ( BC_2,  * ,controlr, 0),   “ &
“91 ( BC_7,  data_sgpiod(24),bidir, 0, 90, 0, Z),“ &

“92 ( BC_2,  * ,controlr, 0),   “ &
```

```
“93 ( BC_7,  data_sgpiod(9),bidir, 0, 92, 0, Z) “ &

“94 ( BC_2,  * ,controlr, 0),   “ &
“95 ( BC_7,  data_sgpiod(10),bidir, 0, 94, 0, Z),“ &

“96 ( BC_2,  * ,controlr, 0),   “ &
“97 ( BC_7,  data_sgpiod(26),bidir, 0, 96, 0, Z),“ &

“98 ( BC_2,  * ,controlr, 0),   “ &
“99 ( BC_7,  data_sgpiod(22),bidir, 0, 98, 0, Z),“ &

“100 ( BC_2,  * ,controlr, 0),   “ &
“101 ( BC_7, data_sgpiod(11),bidir, 0, 100, 0, Z),“ &

“102 ( BC_2,  * ,controlr, 0),   “ &
“103 ( BC_7, data_sgpiod(12),bidir, 0, 102, 0, Z),“ &

“104 ( BC_2,  * ,controlr, 0),   “ &
“105 ( BC_7, data_sgpiod(13),bidir, 0, 104, 0, Z),“ &

“106 ( BC_2,  * ,controlr, 0),   “ &
“107 ( BC_7, data_sgpiod(20),bidir, 0, 106, 0, Z),“ &

“108 ( BC_2,  * ,controlr, 0),   “ &
“109 ( BC_7, data_sgpiod(14),bidir, 0, 108, 0, Z),“ &

“110 ( BC_2,  * ,controlr, 0),   “ &
“111 ( BC_7, data_sgpiod(23),bidir, 0, 110, 0, Z),“ &

“112 ( BC_2,  * ,controlr, 0),   “ &
```

```
“113 ( BC_7, data_sgpiod(15),bidir, 0, 112, 0, Z),“ &

“114 ( BC_2,  * ,controlr, 0),   “ &
“115 ( BC_7, data_sgpiod(16),bidir, 0, 114, 0, Z),“ &

“116 ( BC_2,  * ,controlr, 0),   “ &
“117 ( BC_7,  data_sgpiod(21),bidir, 0, 116, 0, Z),“ &

“118 ( BC_2,  * controlr, 0),   “ &
“119 ( BC_7,  data_sgpiod(17),bidir, 0, 118, 0, Z),“ &

“120 ( BC_2,  * ,controlr, 0),   “ &
“121 ( BC_7,  data_sgpiod(18),bidir, 0, 120, 0, Z),“ &

“122 ( BC_2,  * ,controlr, 0),   “ &
“123 ( BC_7,  data_sgpiod(19),bidir, 0, 122, 0, Z),“ &

“124 ( BC_2, * ,controlr, 0),“ &
“125 (BC_7,irq3_b_kr_b_retry_b_sgpioc3,bidir,0,124,0,Z),“&

“126 ( BC_2,  * ,controlr, 0),“ &
“127 ( BC_7,irq4_b_at2_sgpioc4,bidir, 0, 126, 0, Z),“ &

“128 ( BC_2,  * ,controlr, 0),   “ &
“129 ( BC_7,irq1_b_rsv_b_sgpioc1,bidir, 0, 128, 0, Z),“ &

“130 ( BC_2,  * ,controlr, 0),   “ &
“131 ( BC_7,sgpioc7_irqout_b_lwp0,bidir, 0, 130, 0, Z),“ &

“132 ( BC_2,  * ,controlr, 0),   “ &
```

```
“133 ( BC_7,  bb_b_vf2_iwp3,bidir, 0, 132, 0, Z),“ &

“134 ( BC_2,  * ,controlr, 0),   “ &
“135 ( BC_7,  bg_b_vf0_lwp,bidir, 0, 134, 0, Z),“ &

“136 ( BC_2,  * ,controlr, 0),   “ &
“137 ( BC_7,  br_b_vf1_iwp2,bidir, 0, 136, 0, Z),“ &

“138 ( BC_2,  * ,controlr, 0),   “ &
“139 ( BC_7,  rd_wr_b,bidir, 0, 138, 0, Z),“ &

“140 ( BC_2,  * ,internal, 1),“ &
“141 ( BC_2,  oe_b,output2, 1),“ &

“142 ( BC_2,  * ,controlr, 0),“ &
“143 ( BC_7,  tea_b,bidir, 0, 142, 0, Z),“ &

“144 ( BC_2,  * ,controlr, 0),“ &
“145 ( BC_7, irq2_b_cr_b_sgpioc2,bidir, 0, 144, 0, Z),“ &

“146 ( BC_2,  * ,controlr, 0),“ &
“147 ( BC_7,  irq0_b_sgpioc0,bidir, 0, 146, 0, Z),“ &

“148 ( BC_2,  * ,internal, 1),“ &
“149 ( BC_2,  we_b_at(0),output2, 1),“ &

“150 ( BC_2,  * ,internal, 1),   “ &
“151 ( BC_2,  we_b_at(1),output2, 1),“ &

“152 ( BC_2,  * ,internal, 1),   “ &
```

```
“153 ( BC_2,  we_b_at(2),output2, 1),“ &

“154 ( BC_2,  * ,internal, 1),   “ &
“155 ( BC_2,  we_b_at(3),output2, 1),“ &

“156 ( BC_2,  * ,internal, 1),   “ &
“157 ( BC_2,  cs0_b,output2, 1),“ &
“158 ( BC_2,  *,internal, 1),“ &
“159 ( BC_2,  cs1_b,output2 , 1),“ &

“160 ( BC_2,  *,internal, 1),“ &
“161 ( BC_2,  cs2_b,output2 , 1),“ &

“162 ( BC_2,  *,internal, 1),“ &
“163 ( BC_2,  cs3_b,output2 , 1),“ &

“164 ( BC_2,  *,controlr, 0),“ &
“165 ( BC_7,  burst_b,bidir, 0, 164, 0, Z),“ &

“166 ( BC_2,  *,controlr, 0),“ &
“167 ( BC_7,  bi_b_sts_b,bidir, 0, 166, 0, Z),“ &

“168 ( BC_2,  *,controlr, 0),“ &
“169 ( BC_7,  tsiz0,bidir, 0, 168, 0, Z),“ &

“170( BC_2,  *,controlr, 0),“ &
“171( BC_7,  tsiz1,bidir, 0, 170, 0, Z),“ &

“172 ( BC_2,  *,controlr, 0),“ &
“173 ( BC_7,  ts_b,bidir, 0, 172, 0, Z),“ &
```

```
“174 ( BC_2,  *,controlr, 0),“ &
“175 ( BC_7,  ta_b,bidir, 0, 174, 0, Z),“ &

“176 ( BC_2,  *,controlr, 0),“ &
“177 ( BC_7,  bdip_b,bidir, 0, 176, 0, Z),“ &

“178 ( BC_2,  *,internal, 0), “ &
“179 ( BC_4,  b0epee,input, X), “ &

“180 ( BC_2,  *,internal, 0), “ &
“181 ( BC_4,  epee,input, X), “ &

“182 ( BC_2,  *,internal, 1),“ &
“183 ( BC_2,  clkout,output2, 1), “ &

“184 ( BC_2,  *,internal, 1), “ &
“185 ( BC_2,  engclk_buclk,output2 , 1), “ &

“186 ( BC_2,  *,controlr, 0), “ &
“187 ( BC_7,irq5_b_sgpioc5_modck1,bidir, 0, 186, 0, Z),“ &

“188 ( BC_2,  *,controlr, 0),   “ &
“189 ( BC_7,  irq6_b_modck2,bidir, 0, 188, 0, Z),“ &

“190 ( BC_2,  *,controlr, 0), “ &
“191 ( BC_7,  irq7_b_modck3,bidir, 0, 190, 0, Z),“ &

“192 ( BC_2,  * ,controlr, 0) “ &
“193 ( BC_7,  rstconf_b_texp,bidir, 0, 192, 0, Z),“ &
```

```
“194 ( BC_4,  extclk,input, X),“ &

“195 ( BC_2,  *,controlr, 0),“ &
“196 ( BC_7,  a_cnrx0,bidir, 0, 195, 0, Z),“ &

“197 ( BC_2,  * ,internal, 1),“ &
“198 ( BC_2,  a_cntx0,output2 , 1),“ &

“199 ( BC_2,  * ,controlr, 0),“ &
“200 ( BC_7, a_pcs0_ss_b_qgpio0,bidir, 0, 199, 0, Z),“ &

“201 ( BC_2,  * ,internal, 0),“ &
“202 ( BC_4,  a_eck ,input, X),   “ &

“203 ( BC_2,  * ,controlr, 0),“ &
“204 ( BC_7,  a_pcs1_qgpio1,bidir, 0, 203, 0, Z),“ &

“205 ( BC_2,  * ,internal, 1),“ &
“206 ( BC_2,  a_txd2_qgpo2,output2 , 1),“ &

“207 ( BC_2,  * ,controlr, 0),“ &
“208 ( BC_7,  a_pcs2_qgpio2,bidir, 0, 207, 0, Z),“ &

“209 ( BC_4,  a_rxd2_qgpi2,input, X),“ &

“210 ( BC_4,  b_rxd1_qgpi1,input, X),“ &
“211 ( BC_4,  a_rxd1_qgpi1,input, X),“ &

“212 ( BC_2,  * ,controlr, 0),“
```

```
“213 ( BC_7,  a_mosi_qgpio5,bidir, 0, 212, 0, Z),“ &

“214 ( BC_2,  * ,controlr, 0),“ &
“215 ( BC_7,  a_pcs3_qgpio3,bidir, 0, 214, 0, Z),“ &

“216 ( BC_2,  * ,controlr, 0),“ &
“217 ( BC_7,  a_miso_qgpio4,bidir, 0, 216, 0, Z),“ &

“218 ( BC_2,  * ,controlr, 0),“ &
“219 ( BC_7,  a_sck_qgpio6,bidir, 0, 218, 0, Z),“ &

“220 ( BC_2,  * ,controlr, 0),“ &
“221 ( BC_7,  b_pcs2_qgpio2,bidir, 0, 220, 0, Z),“ &

“222 ( BC_2,  * ,internal, 0),“ &
“223 ( BC_4,  b_rxd2_j1850_rx,input, X),“ &

“224 ( BC_2,  * ,internal, 1),“ &
“225 ( BC_2,  a_txd1_qgpo1,output2 , 1),“ &

“226 ( BC_2,  * ,internal, 1),“ &
“227 ( BC_2,  b_txd2_qgpo2,output2, 1),“ &

“228 ( BC_2,  * ,internal, 1),“ &
“229 ( BC_2,  b_txd1_qgpo1,output2 , 1),“ &

“230 ( BC_2,  * ,internal, 0),“ &
“231 ( BC_4,  b_eck,input, X),   “ &

“232 ( BC_2,  * ,controlr, 0),“ &
```

```
“233 ( BC_7,  b_sck_qgpio6,bidir, 0, 232, 0, Z),“ &

“234 ( BC_2,  * ,controlr, 0),   “ &
“235 ( BC_7,  b_mosi_qgpio5,bidir, 0, 234, 0, Z),“ &

“236 ( BC_2,  * ,controlr, 0),   “ &
“237 ( BC_7,  b_miso_qgpio4,bidir, 0, 236, 0, Z),“ &

“238 ( BC_2,  * ,controlr, 0),   “ &
“239 ( BC_7,  b_pcs3_j1850_tx,bidir, 0, 238, 0, Z),“ &

“240 ( BC_2,  * ,controlr, 0),   “ &
“241 ( BC_7, b_pcs1_qgpio1,bidir, 0, 240, 0, Z),“ &

“242 ( BC_2,  * ,controlr, 0),   “ &
“243 ( BC_7,  b_pcs0_ss_b_qgpio0,bidir, 0, 242, 0, Z),“ &

“244 ( BC_2,  * ,controlr, 0),   “ &
“245 ( BC_7,  vfls1_mpio32b4,bidir, 0, 244, 0, Z),“ &

“246 ( BC_2,  * ,controlr, 0),   “ &
“247 ( BC_7,  vfls0_mpio32b3,bidir, 0, 246, 0, Z),“ &

“248 ( BC_2,  * ,controlr, 0),   “ &
“249 ( BC_7,  vf2_mpio32b2,bidir, 0, 248, 0, Z),“ &

“250 ( BC_2,  * ,controlr, 0),   “ &
“251 ( BC_7,  vf1_mpio32b1,bidir, 0, 250, 0, Z),“ &

“252 ( BC_2,  * ,controlr, 0),   “ &
```

```
“253 ( BC_7,  vf0_mpio32b0,bidir, 0, 252, 0, Z),“ &

“254 ( BC_2,  * ,controlr, 0),“ &
“255 ( BC_7,  mpwm4_mpio32b5,bidir, 0, 254, 0, Z),“ &

“256 ( BC_2,  * ,controlr, 0),   “ &
“257 ( BC_7,  mpwm19,bidir, 0, 256, 0, Z),“ &

“258 ( BC_2,  * ,controlr, 0),“ &
“259 ( BC_7,  mpio32b15,bidir, 0, 258, 0, Z),“ &

“260 ( BC_2,  * ,controlr, 0),   “ &
“261 ( BC_7, c_cnrx0_mpio32b14,bidir, 0, 260, 0, Z),“ &

“262 ( BC_2,* ,controlr, 0),   “ &
“263 ( BC_7, c_cntx0_mpio32b1,bidir, 0, 262, 0, Z),“ &

“264 ( BC_2,* ,controlr, 0),   “ &
“265 ( BC_7, mpwm21_mpio32b12,bidir, 0, 264, 0, Z),“ &

“266 ( BC_2,  * ,controlr, 0),   “ &
“267 ( BC_7, mpwm20_mpio32b11,bidir, 0, 266, 0, Z),“ &

“268 ( BC_2,  * ,controlr, 0),   “ &
“269 ( BC_7,  mda15 ,bidir, 0, 268, 0, Z),“ &

“270 ( BC_2,  * ,controlr, 0),   “ &
“271 ( BC_7,  mda14,bidir, 0, 270, 0, Z),“ &

“272 ( BC_2,  * ,controlr, 0),   “ &
```

```
“273 ( BC_7,  mpwm16,bidir, 0, 272, 0, Z), “ &

“274 ( BC_2,  * ,controlr, 0),   “ &
“275 ( BC_7,  mpwm3,bidir, 0, 274, 0, Z),“ &

“276 ( BC_2,  * ,controlr, 0),   “ &
“277 ( BC_7,  mpwm2,bidir, 0, 276, 0, Z),“ &

“278 ( BC_2,  * ,controlr, 0),   “ &
“279 ( BC_7,  mpwm1,bidir, 0, 278, 0, Z),“ &

“280 ( BC_2,  * ,controlr, 0),   “ &
“281 ( BC_7,  mpwm0,bidir, 0, 280, 0, Z),“ &

“282 ( BC_2,  * ,controlr, 0),   “ &
“283 ( BC_7,  mda31,bidir, 0, 282, 0, Z),“ &

“284 ( BC_2,  * ,controlr, 0),   “ &
“285 ( BC_7,  mda30,bidir, 0, 284, 0, Z),“ &

“286 ( BC_2,  * ,controlr, 0),   “ &
“287 ( BC_7,  mda29,bidir, 0, 286, 0, Z),“ &

“288 ( BC_2,  * ,controlr, 0),   “ &
“289 ( BC_7,  mda28,bidir, 0, 288, 0, Z),“ &

“290 ( BC_2,  * ,controlr, 0),   “ &
“291 ( BC_7,  mda27,bidir, 0 290, 0, Z),“ &

“292 ( BC_2,  * ,controlr, 0),   “ &
```

```
“293 ( BC_7,  mda13,bidir, 0, 292, 0, Z),“ &

“294 ( BC_2,  * ,controlr, 0),   “ &
“295 ( BC_7,  mda12,bidir, 0, 294, 0, Z),“ &

“296 ( BC_2,  * ,controlr, 0),   “ &
“297 ( BC_7,  mda11,bidir, 0, 296, 0, Z),“ &

“298 ( BC_2,  * ,controlr, 0),   “ &
“299 ( BC_7,  mpwm18,bidir, 0, 298, 0, Z),“ &

“300 ( BC_2,  * ,controlr, 0),   “ &
“301 ( BC_7,  mpwm17,bidir, 0, 300, 0, Z)“ &

“302 ( BC_2,  * ,controlr, 0),   “ &
“303 ( BC_7,  b_t2clk,bidir, 0, 302, 0, Z),“ &

“304 ( BC_2,  * ,controlr, 0),   “ &
“305 ( BC_7,  b_tpuch1 ,bidir, 0, 304, 0, Z),“ &

“306 ( BC_2,  * ,controlr, 0),   “ &
“307 ( BC_7,  b_tpuch0,bidir, 0, 306, 0, Z),“ &

“308 ( BC_2,  * ,controlr, 0),   “ &
“309 ( BC_7,  mpwm5_mpio32b6,bidir, 0, 308, 0, Z),“ &

“310 ( BC_2,  * ,controlr, 0),   “ &
“311 ( BC_7,  b_tpuch2,bidir, 0, 310, 0, Z),“ &

“312 ( BC_2,  * ,controlr, 0),   “ &
```

```
“313 ( BC_7,  b_tpuch15,bidir, 0, 312, 0, Z),“ &

“314 ( BC_2,  * ,controlr, 0),   “ &
“315 ( BC_7,  b_tpuch14,bidir, 0, 314, 0, Z),“ &

“316 ( BC_2,  * ,controlr, 0),   “ &
“317 ( BC_7,  b_tpuch13,bidir, 0, 316, 0, Z),“ &

“318 ( BC_2,  * ,controlr, 0),   “ &
“319 ( BC_7,  b_tpuch12,bidir, 0, 318, 0, Z),“ &

“320 ( BC_2,  * ,controlr, 0),   “ &
“321 ( BC_7,  b_tpuch11,bidir, 0, 320, 0, Z),“ &

“322 ( BC_2,  * ,controlr, 0),   “ &
“323 ( BC_7,  b_tpuch10,bidir, 0, 322, 0, Z),“ &

“324 ( BC_2,  * ,controlr, 0),   “ &
“325 ( BC_7,  b_tpuch9,bidir, 0, 324, 0, Z),“ &

“326 ( BC_2,  * ,controlr, 0),   “ &
“327 ( BC_7,  b_tpuch8,bidir, 0, 326, 0, Z),“ &

“328 ( BC_2,  * ,controlr, 0),   “ &
“329 ( BC_7,  b_tpuch7,bidir, 0, 328, 0, Z),“ &

“330 ( BC_2,  * ,controlr, 0),   “ &
“331 ( BC_7,  b_tpuch6,bidir, 0, 330, 0, Z),“ &

“332 ( BC_2,  * ,controlr, 0),   “ &
```

```
“333 ( BC_7,  b_tpuch5,bidir, 0, 332, 0, Z),“ &

“334 ( BC_2,  * ,controlr, 0),   “ &
“335 ( BC_7,  b_tpuch4,bidir, 0, 334, 0, Z),“ &

“336 ( BC_2,  * ,controlr, 0),   “ &
“337 ( BC_7,  b_tpuch3,bidir, 0, 336, 0, Z),“ &

“338 ( BC_2,  * ,controlr, 0),   “ &
“339 ( BC_7,  a_tpuch1,bidir, 0, 338, 0, Z),“ &

“340 ( BC_2,  * ,controlr, 0),   “ &
“341 ( BC_7,  a_tpuch0,bidir, 0, 340, 0, Z),“ &

“342 ( BC_2,  * ,controlr, 0),   “ &
“343 ( BC_7,  a_t2cl,bidir, 0, 342, 0, Z),“ &

“344 ( BC_2,  * ,controlr, 0),   “ &
“345 ( BC_7, a_tpuch15,bidir, 0, 344, 0, Z),“ &

“346 ( BC_2,  * ,controlr, 0),   “ &
“347( BC_7,  a_tpuch14,bidir, 0, 346, 0, Z),“ &

“348 ( BC_2,  * ,controlr, 0),   “ &
“349 ( BC_7,  a_tpuch13,bidir, 0, 348, 0, Z),“ &

“351 ( BC_7,  a_tpuch12,bidir, 0, 350, 0, Z),“ &

“352 ( BC_2,  * ,controlr, 0),   “ &
“353 ( BC_7,  a_tpuch11,bidir, 0, 352, 0, Z),“ &
```

```
“354 ( BC_2,  * ,controlr, 0),   “ &
“355 ( BC_7,  a_tpuch10,bidir, 0, 354, 0, Z),“ &

“356 ( BC_2,  * ,controlr, 0),   “ &
“357 ( BC_7,  a_tpuch9 ,bidir, 0, 356, 0, Z),“ &

“358 ( BC_2,  * ,controlr, 0),   “ &
“359 ( BC_7,  a_tpuch8,bidir, 0, 358, 0, Z),“ &

“360 ( BC_2,  * ,controlr, 0),   “ &
“361 ( BC_7,  a_tpuch7,bidir, 0, 360, 0, Z),“ &

“362 ( BC_2,  * ,controlr, 0),   “ &
“363 ( BC_7,  a_tpuch6,bidir, 0, 362, 0, Z),“ &

“364 ( BC_2,  * ,controlr, 0),   “ &
“365 ( BC_7,  a_tpuch5,bidir, 0, 364, 0, Z),“ &

“366 ( BC_2,  * ,controlr, 0),   “ &
“367 ( BC_7,  a_tpuch4,bidir, 0, 366, 0, Z),“ &

“368 ( BC_2,  * ,controlr, 0),   “ &
“369 ( BC_7,  a_tpuch3,bidir, 0, 368, 0, Z),“ &

“370 ( BC_2,  * ,controlr, 0),   “ &
“371 ( BC_7,  a_tpuch2,bidir, 0, 370, 0, Z),“ &

“372 ( BC_2,  * ,internal, 0),   “ &
“373 ( BC_4,  etrig1,input, X), “ &
```

```
“374 ( BC_2,  * ,internal, 0),   “ &
“375 ( BC_4,  etrig2,input, X),   “ &

“376 ( BC_2,  * ,controlr, 0),   “ &
“377 ( BC_7,  an64_b_pqb0,bidir, 0, 376, 0, Z),“ &

“378 ( BC_2,  * ,controlr, 0),   “ &
“379 ( BC_7,  an65_b_pqb1,bidir, 0, 378, 0, Z)“ &

“380 ( BC_2,  * ,controlr, 0),   “ &
“381 ( BC_7,  an66_b_pqb2,bidir, 0, 380, 0, Z),“ &

“382 ( BC_2,  * ,controlr, 0),   “ &
“383 ( BC_7,  an67_b_pqb3,bidir, 0, 382, 0, Z),“ &

“384 ( BC_2,  * ,controlr, 0),   “ &
“385 ( BC_7,  an68_b_pqb4,bidir, 0, 384, 0, Z),“ &

“386 ( BC_2,  * ,controlr, 0),   “ &
“387 ( BC_7,  an69_b_pqb5,bidir, 0, 386, 0, Z),“ &

“388 ( BC_2,  * ,controlr, 0),   “ &
“389 ( BC_7,  an70_b_pqb6,bidir, 0, 388, 0, Z),“ &

“390 ( BC_2,  * ,controlr, 0),   “ &
“391 ( BC_7,  an71_b_pqb7,bidir, 0, 390, 0, Z),“ &

“392 ( BC_2,  * ,controlr, 0),   “ &
“393 ( BC_7,  an72_b_ma0_pqa0,bidir, 0, 392, 0, Z),“ &
```

```
“394 ( BC_2,  * ,controlr, 0),   “ &
“395 ( BC_7,  an73_b_ma1_pqa1,bidir, 0, 394, 0, Z),“ &

“396 ( BC_2,  * ,controlr, 0),“ &
“397 ( BC_7,  an74_b_ma2_pqa2,bidir, 0, 396, 0, Z),“ &

“398 ( BC_2, * ,controlr, 0),   “ &
“399 ( BC_7,  an75_b_pqa3,bidir, 0, 398, 0, Z),“ &

“400 ( BC_2,  * ,controlr, 0),   “ &
“401 ( BC_7,  an76_b_pqa4,bidir, 0, 400, 0, Z),“ &

“402 ( BC_2,  * ,controlr, 0),   “ &
“403 ( BC_7,  an77_b_pqa5,bidir, 0, 402, 0, Z),“ &

“404 ( BC_2,  * ,controlr, 0),   “ &
“405 ( BC_7,  an78_b_pqa6,bidir, 0, 404, 0, Z),“ &

“406 ( BC_2,  * ,controlr, 0), “ &
“407 ( BC_7,  an79_b_pqa7,bidir, 0, 406, 0, Z),“ &

“408 ( BC_2,  * ,controlr, 0), “ &
“409 ( BC_7, an59_a_pqa7,bidir, 0, 408, 0, Z),“ &

“410 ( BC_2,  * ,controlr, 0),   “ &
“411 ( BC_7,  an58_a_pqa6,bidir, 0, 410, 0, Z),“ &

“412 ( BC_2,  * ,controlr, 0),   “ &
“413 ( BC_7,  an57_a_pqa5,bidir, 0, 412, 0, Z),“ &
```

```
“414 ( BC_2,  * ,controlr, 0),   “ &
“415 ( BC_7,  an56_a_pqa4,bidir, 0, 414, 0, Z),“ &

“416 ( BC_2,  * ,controlr, 0),“ &
“417 ( BC_7,  an55_a_pqa3,bidir, 0, 416, 0, Z),“ &

“418 ( BC_2,  * ,controlr, 0), “ &
“419 ( BC_7,  an54_a_ma2_pqa2,bidir, 0, 418, 0, Z),“ &

“420 ( BC_2,  * ,controlr, 0),   “ &
“421 ( BC_7,  an53_a_ma1_pqa1,bidir, 0, 420, 0, Z),“ &

“422 ( BC_2,  * ,controlr, 0),   “ &
“423 ( BC_7,  an52_a_ma0_pqa0,bidir, 0, 422, 0, Z),“ &

“424 ( BC_2,  * ,controlr, 0),   “ &
“425 ( BC_7,  an51_a_pqb7,bidir, 0, 424, 0, Z),“ &

“426 ( BC_2,  * ,controlr, 0),   “ &
“427 ( BC_7,  an50_a_pqb6,bidir, 0, 426, 0, Z),“ &

“428 ( BC_2,  * ,controlr, 0),   “ &
“429 ( BC_7,  an49_a_pqb5,bidir, 0, 428, 0, Z),“ &

“430 ( BC_2,  * ,controlr, 0),   “ &
“431 ( BC_7,  an48_a_pqb4,bidir, 0, 430, 0, Z),“ &

“432 ( BC_2,  * ,controlr, 0),   “ &
“433 ( BC_7,  an47_anz_a_pqb3,bidir, 0, 432, 0, Z),“ &
```

```
“434 ( BC_2,  * ,controlr, 0),   “ &
“435 ( BC_7,  an46_any_a_pqb2,bidir, 0, 434, 0, Z),“ &

“436 ( BC_2,  * ,internal, 0),   “ &
“437 ( BC_4,  an80,input, X),    “ &

“438 ( BC_2,  * ,internal, 0),   “ &
“439 ( BC_4,  an81,input, X),    “ &

“440 ( BC_2,  * ,internal, 0),   “ &
“441 ( BC_4,  an82,input, X),    “ &

“442 ( BC_2,  * ,internal, 0),   “ &
“443 ( BC_4,  an83,input, X),    “ &

“444 ( BC_2,  * ,internal, 0),   “ &
“445 ( BC_4,  an84,input, X),    “ &

“446 ( BC_2,  * ,internal, 0),   “ &
“447 ( BC_4,  an85,input, X),    “ &

“448 ( BC_2,  * ,internal, 0),   “ &
“449 ( BC_4,  an86,input, X),    “ &

“450 ( BC_2,  * ,internal, 0),   “ &
“451 ( BC_4,  an87,input, X),    “ &

“452 ( BC_2,  * ,controlr, 0),   “ &
“453 ( BC_7,  an45_anx_a_pqb1,bidir, 0, 452, 0, Z),“ &
```

```
“454 ( BC_2,  * ,controlr, 0),   “ &
“455 ( BC_7,  an44_anw_a_pqb0,bidir, 0, 454, 0, Z),“ &

“456 ( BC_2,  * ,controlr, 0),   “ &
“457 ( BC_7,  b_cnrx0,bidir, 0, 456, 0, Z),“ &

“458 ( BC_2,  * ,internal, 1),   “ &
“459 ( BC_2,  b_cntx0,output2, 1),   “ &

“460 ( BC_2,  * ,controlr, 0),   “ &
“461 ( BC_7,  c_t2clk,bidir, 0, 460, 0, Z),“ &

“462 ( BC_2,  * ,controlr, 0),   “ &
“463 ( BC_7,  c_tpuch15,bidir, 0, 462, 0, Z),“ &

“464 ( BC_2,  * ,controlr, 0),   “ &
“465 ( BC_7,  c_tpuch14,bidir, 0, 464, 0, Z),“ &

“466 ( BC_2,  * ,controlr, 0),   “ &
“467 ( BC_7,  c_tpuch13,bidir, 0, 466, 0, Z),“ &

“468 ( BC_2,  * ,controlr, 0),   “ &
“469 ( BC_7,  c_tpuch12,bidir, 0, 468, 0, Z),“ &

“470 ( BC_2,  * ,controlr, 0),   “ &
“471 ( BC_7,  c_tpuch11,bidir, 0, 470, 0, Z),“ &

“472 ( BC_2,  * ,controlr, 0),   “ &
“473 ( BC_7,  c_tpuch10,bidir, 0, 472, 0, Z),“ &
```

```
“474 ( BC_2,  * ,controlr, 0),   “ &
“475 ( BC_7,  c_tpuch9,bidir, 0, 474, 0, Z),“ &

“476 ( BC_2,  * ,controlr, 0),   “ &
“477 ( BC_7,  c_tpuch8,bidir, 0, 476, 0, Z),“ &

“478 ( BC_2,  * ,controlr, 0),   “ &
“479 ( BC_7,  c_tpuch7,bidir, 0, 478, 0, Z),“ &

“480 ( BC_2,  * ,controlr, 0),   “ &
“481 ( BC_7,  c_tpuch6,bidir, 0, 480, 0, Z),“ &

“482 ( BC_2,  * ,controlr, 0),   “ &
“483 ( BC_7,  c_tpuch5,bidir, 0, 482, 0, Z),“ &

“484 ( BC_2,  * ,controlr, 0),   “ &
“485 ( BC_7,  c_tpuch4,bidir, 0, 484, 0, Z),“

“486 ( BC_2,  * , controlr, 0),   “ &
“487 ( BC_7,  c_tpuch3,bidir, 0, 486, 0, Z),“ &

“488 ( BC_2,  * ,controlr, 0),   “ &
“489 ( BC_7,  c_tpuch2,bidir, 0, 488, 0, Z),“ &

“490 ( BC_2,  * ,controlr, 0),   “ &
“491 ( BC_7,  c_tpuch1,bidir, 0, 490, 0, Z),“ &

“492 ( BC_2,  * ,controlr, 0),   “ &
“493 ( BC_7,  c_tpuch0,bidir, 0, 492, 0, Z),“ &
```

```
“494 ( BC_2,  * ,controlr, 0),   “ &
“495 ( BC_7,  mcki,bidir, 0, 494, 0, Z),“ &

“496 ( BC_2,  * ,controlr, 0),   “ &
“497 ( BC_7,  mdi_0,bidir, 0, 496, 0, Z),“ &

“498 ( BC_2,  * ,controlr, 0),   “ &
“499 ( BC_7,  mdi_1,bidir, 0, 498, 0, Z),“ &

“500 ( BC_2,  * ,controlr, 0),   “ &
“501 ( BC_7,  msei_b,bidir, 0, 500, 0, Z),“ &

“502 ( BC_2,  * ,internal, 1),   “ &
“503 ( BC_2,  mdo_1,output2 , 1),   “ &

“504 ( BC_2,  * ,internal, 1),   “ &
“505 ( BC_2,  mdo_0,output2 , 1),   “ &

“506 ( BC_2,  * ,internal, 1),   “ &
“507 ( BC_2,  mcko,output2 , 1),   “ &

“508 ( BC_2,  * ,controlr, 0),   “ &
“509 ( BC_7,  mdo_7_mpio32b7,bidir, 0, 508 0 Z),“ &

“510 ( BC_2,  * ,controlr, 0),   “ &
“511 ( BC_7,  mdo_6_mpio32b8,bidir, 0, 510, 0, Z),“ &

“512 ( BC_2,  * ,controlr, 0),   “ &
“513 ( BC_7,  mdo_5_mpio32b9,bidir, 0, 512, 0, Z),“ &
```

```
“514 ( BC_2,  * ,controlr, 0),   “ &
“515 ( BC_7,  mdo_4_mpio32b10,bidir, 0, 514, 0, Z),“ &

“516 ( BC_2,* ,controlr, 0),“ &
“517 ( BC_4, * ,internal, 0),“ &

“518 ( BC_2,  * ,controlr, 0),   “ &
“519 ( BC_7, evti_b,bidir, 0, 518, 0, Z)“;

end spanishoak;
```