DBI Background

Spec update was done to fix the DBI location bits

4/8/2021	2.0.1	26	Updated CONF_DONE description
		27	Updated ns_mac_rdy description
		28	Clarified reset and standby conditions
		29	Updated dual_mode_select description
		31	Corrected Word Mark bit recommended location
		32	Added detail about the DBI bit location in the full rate word

According to this the location should be as shown below (half rate example):

	159	158	157	156	120	119	118	117	80	79	78	77	76	40	39	38	37	2	1	0
Leader to Follower 297b with	DBI	DBI	Mark=1	D[296:260]		DBI	DBI	D[259:122]		DBI	DBI	Mark=0	D[221:185]		DBI	DBI	D[184:149]		Strobe	D[148]
TVALID on AIB2.0Gen2 Half	AIB Channel N+1																			
Rate	159	158	157	156	120	119	118	117	80	79	78	77	76	40	39	38		2	1	0
	DBI	DBI	Mark=1	D[147:111]		DBI	DBI	D[110:73]		DBI	DBI	Mark=0	D[72:36]		DBI	DBI	D[35:0]		Strobe	Push
	AIB Channel N																			

As described in spec the data comes on AIB/IO as even, odd, even, odd format.

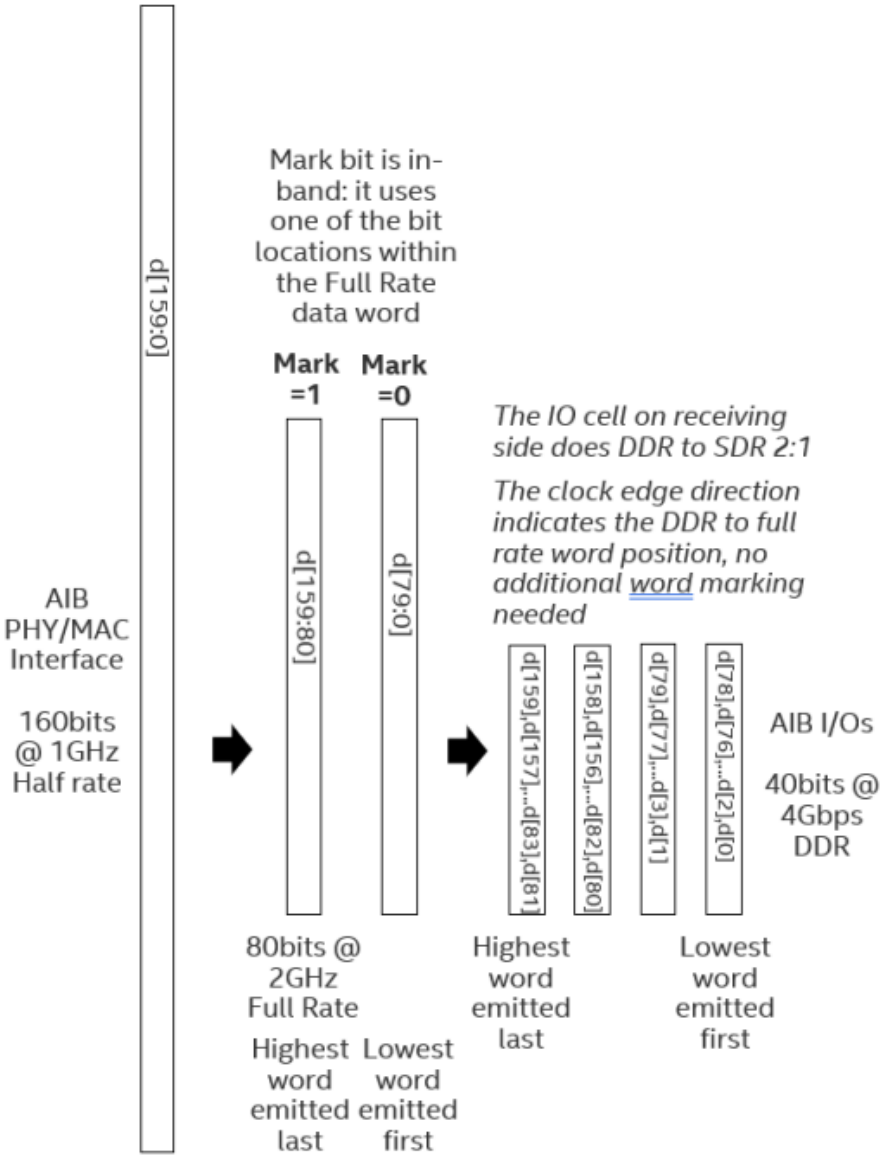

Figure 31. Word Marking Example, Half Rate PC Mode, 40 Tx Bits per Channel

DBI bits are present on the DDR'ed bits. AIB IO 19 and 39 are physical DBI IOs but the logical bit location corresponding to this D[38], D[39], D[78]. D[79], as shown below (**correct mapping**)

AIB Channel Data					IO
159	158	79	78	↔	IO 39
157	156	77	76	↔	IO 38
155	154	75	74	↔	IO 37
153	152	73	72	↔	IO 36
151	150	71	70	↔	IO 35
149	148	69	68	↔	IO 34
147	146	67	66	↔	IO 33
145	144	65	64	↔	IO 32
143	142	63	62	↔	IO 31
141	140	61	60	↔	IO 30
139	138	59	58	↔	IO 29
137	136	57	56	↔	IO 28
135	134	55	54	↔	IO 27
133	132	53	52	↔	IO 26
131	130	51	50	↔	IO 25
129	128	49	48	↔	IO 24
127	126	47	46	↔	IO 23
125	124	45	44	↔	IO 22
123	122	43	42	↔	IO 21
121	120	41	40	↔	IO 20
119	118	39	38	↔	IO 19
117	116	37	36	↔	IO 18
115	114	35	34	↔	IO 17
113	112	33	32	↔	IO 16
111	110	31	30	↔	IO 15
109	108	29	28	↔	IO 14
107	106	27	26	↔	IO 13
105	104	25	24	↔	IO 12
103	102	23	22	↔	IO 11
101	100	21	20	↔	IO 10
99	98	19	18	↔	IO 9
97	96	17	16	↔	IO 8
95	94	15	14	↔	IO 7
93	92	13	12	↔	IO 6
91	90	11	10	↔	IO 5
89	88	9	8	↔	IO 4
87	86	7	6	↔	IO 3
85	84	5	4	↔	IO 2
83	82	3	2	↔	IO 1
81	80	1	0	↔	IO 0

Section 2.2.4 in the spec is with respect to the IOs so it is consistent with the logical location shown above

2.2.4 Data Bus Inversion

Data Bus Inversion (DBI) is intended to reduce the power delivery network load of an AIB PHY by limiting the number of AIB data bits that can switch between immediate data transfer cycles. In Gen2 mode (DDR data transfers), AIB shall support DBI-AC in groups of 20 RX and 20 TX data wires. Channels may have multiple DBI data groups, for example a channel with 40 RX and 40 TX has a total of 4 DBI data groups. DBI shall be configurable on or off. When DBI is configured on, the AIB adapter shall process both RX and TX data for DBI. With DBI off, the data inside RX and TX are unaffected.

With DBI on, DBI bits replace certain data bits. The example below shows a 40 bit TX channel. RX wires have DBI at the same bit locations. Channels with more IOs continue with same 19 data bits + 1 DBI bit increment.

DBI Off: TX[39:0] = data[39:0]

DBI On: TX[39:0] = {DBI[1], data[38:20], DBI[0], data[18:0]}

The MAC data_in and data_out (section 1.3.5) contains even and odd bits that are multiplexed at the DDR transmitter and demultiplexed at DDR receiver (sections 2.1.1, 2.1.2 and 2.1.3). The “data” in the above example is selected from the MAC data_in, anticipating DDR transmission. With DBI On, the data bits at the DBI locations are not used.

2.2.4.1 TX DBI with DBI Enabled

Within a group of 19 data signals, the TX DBI logic calculates the DBI bit based on the number of data signals changing from their previous state on the AIB data bus.

The DBI logic below uses “+” to indicate arithmetic addition, “^” to indicate exclusive OR, and “?” as a ternary IF. “Current” refers to the new data word being prepared for sending on TX. “Prev” refers to the data immediately preceding the current data, that is the data issued onto the AIB bus before the current data.

$$DBI_{current} = ((data[18]_{current} \wedge data[18]_{prev}) + (data[17]_{current} \wedge data[17]_{prev}) \ldots + (data[1]_{current} \wedge data[1]_{prev}) + (data[0]_{current} \wedge data[0]_{prev})) > 9 \; ? \; 1 : 0);$$

Within a group of 19 data signals, if the DBI bit=1 then the TX DBI logic inverts the data signals. Each calculated DBI bit replaces one data bit in TX as described previously.

Issue

aib-phy-hardware/v2.0/rev1/rtl/aib_adapttxdbi_txdp.v uses non-DDR version which should be fixed (**incorrect mapping** shown below)

AIB Channel Data					IO
159	139	79	59	↔	IO 39
158	138	78	58	↔	IO 38
157	137	77	57	↔	IO 37
156	136	76	56	↔	IO 36
155	135	75	55	↔	IO 35
154	134	74	54	↔	IO 34
153	133	73	53	↔	IO 33
152	132	72	52	↔	IO 32
151	131	71	51	↔	IO 31
150	130	70	50	↔	IO 30
149	129	69	49	↔	IO 29
148	128	68	48	↔	IO 28
147	127	67	47	↔	IO 27
146	126	66	46	↔	IO 26
145	125	65	45	↔	IO 25
144	124	64	44	↔	IO 24
143	123	63	43	↔	IO 23
142	122	62	42	↔	IO 22
141	121	61	41	↔	IO 21
140	120	60	40	↔	IO 20
119	99	39	19	↔	IO 19
118	98	38	18	↔	IO 18
117	97	37	17	↔	IO 17
116	96	36	16	↔	IO 16
115	95	35	15	↔	IO 15
114	94	34	14	↔	IO 14
113	93	33	13	↔	IO 13
112	92	32	12	↔	IO 12
111	91	31	11	↔	IO 11
110	90	30	10	↔	IO 10
109	89	29	9	↔	IO 9
108	88	28	8	↔	IO 8
107	87	27	7	↔	IO 7
106	86	26	6	↔	IO 6
105	85	25	5	↔	IO 5
104	84	24	4	↔	IO 4
103	83	23	3	↔	IO 3
102	82	22	2	↔	IO 2
101	81	21	1	↔	IO 1
100	80	20	0	↔	IO 0

```
module aib_adapttxdbi_txdp (

   input  wire                rst_n,
   input  wire                clk,
   input  wire [79:0]         data_in,
   input  wire                dbi_en,

   output wire [79:0]         data_out
   );

   wire [3:0] dbi_calc;
   reg  [79:0] last_din, dbi_data_out;

   assign data_out = dbi_en? dbi_data_out : data_in;
   assign dbi_calc = {dbi_value(data_in[78:60], last_din[78:60]), dbi_value(data_in[58:40], last_din[58:40]),
                      dbi_value(data_in[38:20], last_din[38:20]), dbi_value(data_in[18:0],  last_din[18:0])};
```