PROMISE®
TECHNOLOGY, INC.

PDC20621

PCI Bus Mastering ATA RAID Accelerator

Programming Guide for

Local DIMM

Revision 1.4

General Description

PDC20621 can be used for FastTRAK100, SuperTRAK100 and others applicable application. At the controller initial stage, a PLL base clock synthesizer that refer to PCI clock to generate internal clock, must be programmed to generate appropriate clock for ASIC internal reference. If any PC66 DIMM module is detected, then the internal clock must be programmed to 66MHz and ATA only support up to Ultra66. If all DIMM modules support PC100, the internal clock of PDC20621 must be programmed to 100MHz in all applications. The maximum memory size is a total of 2GB local memory space. The PC-100 or upper SDRAM must be provided for this control. All timing parameter can be setting up using SPD.

1. How to Program DIMM Size and Timing

2.1 Read SPD0 for DIMM#0 by serial interface Engine.

a. Program serial interface Address/Data Register(BA#3 C004Ch). The Dev-Address of SPD0 is “0x50” and the Sub-Addresses of internal registers in SPD0 are between “0x00” and ”0x7F”.
b. Program Sequence Counter Control Register.
c. Program the serial interface Control register Bit[7:0] (BA#3 C0048h) to start serial interface Engine.

d. Wait the Interrupt(or polling Sequence Interrupt Status register (BA#3 C0480h)).

e. Check serial interface Error(no acknowledge) (Sequence Counter Control Register Bit7). If the error generates, SPD0 don't exist.

f. If SPD0 exists, program D0REG.

D0REG(BA#3 C0080h)	**Mapping Table**	**Reasonable Value**
PC133/PC100/PC66	if **SPD0[126] is 66** => PC66 else **SPD0[9] <=75h** => PC133 else => PC100	SPD0[126] {100/66}
D0REG[2:0]	SPD0[4]-8	13,12,11,10,9,8(SPD[4])
D0REG[3]	SPD0[21] != 0x00	0, 0x1F, 0x16(SPD[21])
D0REG[5:4]	SPD0[3]-11	14,13,12,11(SPD[3])
D0REG[6]	[SPD0[17]/4]	4,2(SPD[17])
D0REG[7]	[SPD0[5]/2]	2,1(SPD[5])
D0REG[9:8]	[(SPD0[27]+9)/10] –1	40~10(SPD[27])
D0REG[11:10]	[({SPD0[29],SPD[28]}+9)/10] –1	40~10(SPD[29],SPD[28])
D0REG[13:12]	(SPD0[30]-SPD0[29]+9)/10-2	50~20(SPD[30]
D0REG[15:14]	If SPD0[18][3]=> 11 Elseif SPD0[18][2] =>10 Elseif SPD0[18][1]=> 01 Else SPD0[18][0]=> 00	15~0(SPD[18])
DIMM SIZE0(DISZE0)	2^(SPD0[4]+(SPD0[5]/2)+SPD0[3]+(SPD0[17]/2)+3)	2^31~2^24(DIMM SIZE) (2G~16M)
D0REG[23]	If DIMM#0 exit => 0 else => 1	
D0REG[22:16]	If DSIZE0>=DSIZE1 => DS0/16M-1 else =>(DS0+DS1)/16M-1	
D0REG[31]	If DIMM#0 exit => 0 Else => 1	
D0REG[30:24]	If DSIZE0>=DSIZE1 => 0 Else	

=>DS1/16M

*SPD[sub][bit], “sub” is the sub-address in SPD, and “bit” is the bit in the sub-address register.
*”[a]” is to obtain the integer of “a”.
*”{a,b}” is the maximum value between “a” and “b”.
*If one of all parameters in all SPDs is not reasonable, show the warning or error massages.
*The DIMM0 default space is from 0x00000000 to 0x3FFFFFFF(0x00 in D0REG[30:24] and 0x3F inD0REG[22:16]). If you want to disable the space decoder, set D0REG[31] or D0REG[23] to one.

2.2 Read SPD1 for DIMM#1 by Serial Interface Engine.

a. Program serial interface Address/Data Register(BA#3 C004Ch). The Dev-Address of SPD1 is “0x51” and the Sub-Addresses of internal registers in SPD1 are between “0x00” and ”0x7F”.
b. Program Sequence Counter Control Register.
c. Program the serial interface Control register Bit[7:0] (BA#3 C0048h) to start serial interface Engine.
d. Wait the Interrupt(or polling Sequence Interrupt Status register (BA#3 C0480h)).
e. Check serial interface Error(no acknowledge) (Sequence Counter Control Register Bit7). If the error generates, SPD1 don’t exist.
f. If SPD1 exists, program D1REG.

D1REG(BA#3 C0084h)	**Mapping Table**	**Reasonable Value**
PC133/PC100/PC66	if **SPD1[126] is 66** => PC66 else **SPD1[9] <=75h** => PC133 else =>PC100	SPD1[126] {100/66}
D1REG[2:0]	SPD1[4]-8	13,12,11,10,9,8(SPD[4])
D1REG[3]	if SPD1[21] !=0x00	0x00, 0x1F, 0x16(SPD[21])
D1REG[5:4]	SPD1[3]-11	14,13,12,11(SPD[3])
D1REG[6]	[SPD1[17]/4]	4,2(SPD[17])
D1REG[7]	[SPD1[5]/2]	2,1(SPD[5])
D1REG[9:8]	[(SPD1[27]+9)/10] –1	40~10(SPD[27])
D1REG[11:10]	[({SPD0[29],SPD[28]}+9)/10] –1	40~10(SPD[29],SPD[28])

D1REG[13:12]	(SPD1[30]-SPD1[29]+9)/10-2	50~20(SPD[30]
D1REG[15:14]	If SPD1[18][3]=> 11 Elseif SPD1[18][2] =>10 Elseif SPD1[18][1]=> 01 Else SPD1[18][0]=> 00	15~0(SPD[18])
DIMM SIZE1(DISZE1)	2^(SPD1[4]+(SPD1[5]/2)+SPD1[3]+(SPD1[17]/2)+3)	2^31-2^24 (DIMM SIZE) (2G~16M)
D1REG[23]	If DIMM#1 exit => 0 else => 1	
D1REG[22:16]	If DSIZE1>DSIZE0 => DS1/16M-1 else =>(DS1+DS0)/16M-1	
D1REG[31]	If DIMM#1 exit => 0 else => 1	
D1REG[30:24]	If DSIZE1>DSIZE0 => 0 Else =>DS0/16M	

*The DIMM1 default space is from 0x40000000 to 0x7FFFFFF (0x40 in D1REG[30:24] and 0x7F inD1REG[22:16]). If you want to disable the space decoder, set D1REG[31] or D1REG[23] to one.

*If one of two DIMM modules is 2Gbytes and the other still exits, an error will be generated. Because the maximum memory space is 2Gbytes, they exceed the maximum space. Please disable a smaller DIMM module by program D?REG[23] or D?REG[31].

2.3 Check Multi-Chip Application.

a. Program the BA#2:88h register by the value 0x0e
b. Program the BA#2:88h register by the value 0x0d
c. Program the BA#2:88h register by the value 0x0b
d. Program the BA#2:88h register by the value 0x07
e. Read the BA#2:8ch register and check whether these slave chips exist or not.
f. If slave chips exist, then program D0REG and D1REG in all slave chips.

2.4 Program DGREG.

a. If slave chips exist, then program DGREG[20] by the value 0b in all chips.

b. Check DGREG[25]. If DGREG[25] is zero, BBM exists.
c. Enable refresh in all PDC20621 chips by Programming DGREG[11:0] and DGREG[17].
d. Program DGREG[24] by the value 1b and DGREG[21] by the value 0b.
e. Wait until DGREG[25:24] is 10b.
e. Program DGREG[19:18] by the value 10b if DIMM#0 exists.
f. Wait until DGREG[19] is zero if DIMM#0 exists.
g. Program DGREG[19:18] by the value 11b if DIMM#1 exists.
h. Wait until DGREG[19] is zero if DIMM#1 exists.

DGREG(BA#3 C0088h)	Mapping Table	Reasonable Value
DGREG[3:0]	(Muti-Chip*3)-1	1~4(Multi-Chip)
DGREG[7:4]	If {SPD0[12], SPD1[12]}==80 => 3 If {SPD0[12], SPD1[12]}==82 => 7	80,82 (SPD[12])
DREG[11:8]	({SPD0[27]+SPD0[30],SPD1[27]+SPD1[30]}+9)/10 – 1	10~160
DREG[15:12]	0x5	
DGREG[16]	SPD0[11]==2 && SPD1[11]==2 && QWORD Write Operation	0,2 (SPD[11])
DGREG[17]	All PDC20621 Chip Enable this bit	
DGREG[19:18]	Initializing only in the master Chip	

2.5 SDRAM Access Restriction in PIO Transfer

1. ECC-SDRAM:
   - Byte-access or Word-Access is not allowed here!!
   - Double-Word is only way to access SDRAM, every access length should be guaranteed to Quadruple-Word.

     Note:

     (1) Before forcing dump the FIFO data, all amount of data you write to FIFO should be quadruple.

     (2) The initial address of first access should be quadruple.

2. Non-ECC SDRAM: No-Restriction!!

   However, please avoid to use Byte access:

   PDC621 will activate an SDRAM access whenever receive byte access command, and that will result in non-necessary overhead.

When PDC621 activate SDRAM access for Host-side request:

- Force-dump bit set to one.
- The data in FIFO is over threshold.
- The coming command and the previous commnad can not form a continuous address-space.
- Direction of data flow change (Read => Write, Write=> Read).
- Any write command of non-Double Word Access.

EngineerToEngineerNote :

- ***Using Force-Dump bit after the last data is written to FIFO.***
- ***Once you want to access SDRAM, the larger data block is preferred. (For better performance)***
- ***Adjacent command to SDRAM should be formed a continuous address space.***
- ***Double-word access with Quadruple-Word boundary.***