**OTP Review**

# OTP 简介

## OTP框架图



## 数据空间分配

为了保证OTP的正确性，我们对于一些数据program 2次，两次里面写进去一次就算写成功，冗余设计的具体空间以及目前分配的空间如表格所示。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bank | Start | End | Size | Purpose |
| Bank0 | 0x0\_0000 | 0x0\_03FF | 1Kb | access right table for 35 regions, each region has 4 access right bits, 4x35=4.375x32= 5x32, read 10 times （1-10） |
| 0x0\_0000-0x0\_0003: Bank0 access right |
| 0x0\_0004-0x0\_0007: Bank1 access right |
| 0x0\_0008-0x0\_000b: Bank2 access right |
| 0x0\_000c-0x0\_000f: Bank3 access right |
| 0x0\_0010-0x0\_0013: Bank4 access right |
| 0x0\_0014-0x0\_0017: Bank5 access right |
| 0x0\_0018-0x0\_001b: Bank6 access right |
| 0x0\_001c-0x0\_001f:: Bank7 access right |
| 0x0\_0020-0x0\_0023: Bank8 access right |
| 0x0\_0024-0x0\_0027: Bank9 access right |
| 0x0\_0028-0x0\_002b: Bank10 access right |
| 0x0\_002c-0x0\_002f : Bank11 access right |
| 0x0\_0030-0x0\_0033: Bank12 access right |
| 0x0\_0034-0x0\_0037: Bank13 access right |
| 0x0\_0038-0x0\_003b: Bank14 access right |
| 0x0\_003c-0x0\_003f: Bank15 access right |
| 0x0\_0040-0x0\_0043: Bank16 access right |
| 0x0\_0044-0x0\_0047: Bank17 access right |
| 0x0\_0048-0x0\_004b: Bank18 access right |
| 0x0\_004c-0x0\_004f:: Bank19 access right |
| 0x0\_0050-0x0\_0053: Bank20 access right |
| 0x0\_0054-0x0\_0057: Bank21 access right |
| 0x0\_0058-0x0\_005b: Bank22 access right |
| 0x0\_005c-0x0\_005f: Bank23 access right |
| 0x0\_0060-0x0\_0063: Bank24 access right |
| 0x0\_0064-0x0\_0067: Bank25 access right |
| 0x0\_0068-0x0\_006b: Bank26 access right |
| 0x0\_006c-0x0\_006f: Bank27 access right |
| 0x0\_0070-0x0\_0073: Bank28 access right |
| 0x0\_0074-0x0\_0077: Bank29 access right |
| 0x0\_0078-0x0\_007b: Bank30 access right |
| 0x0\_007c-0x0\_007f: Bank31 access right |
| 0x0\_0080-0x0\_0083: Bank32 access right |
| 0x0\_0084-0x0\_0087: Bank33 access right |
| 0x0\_0088-0x0\_008b: Bank34 access right |
| 0x0\_00a0-0x0\_012b: redundancy of the above data from Bank0 to Bank34 access right |
|  |
| 0x0\_02e0-0x0\_02FF: each checksum tag is 1 bit, 32bits checksum tag |
| 0x0\_0300-0x0\_03FF: each checksum data is 8 bits , 256 bits checksum data |
| Bank1 | 0x0\_0400 | 0x0\_07FF | 1Kb | chip HW config (128)x2, read 8 times （33-40）, abc |
| 0x0\_0400-0x0\_047F: chip HW config |
| 0x0\_0400-0x0\_04FF: redundancy of the above chip HW config |
| 0x0\_0600-0x0\_07FF: root key |
| Bank2 | 0x0\_0800 | 0x0\_0bFF | 1Kb | tRoot HW config(128+128+32)x2, read 18 times （65-82） |
| 0x0\_0800-0x0\_091F: tRoot HW config |
| 0x0\_0920-0x0\_0a3F: redundancy of the above tRoot HW config |
| Bank3 | 0x0\_0c00 | 0x0\_0FFF | 1Kb | HDMI1.4 KEY 285B+2BCRC |
| Bank4 | 0x0\_1000 | 0x0\_13FF | 1Kb |
| Bank5 | 0x0\_1400 | 0x0\_17FF | 1Kb |
| Bank6 | 0x0\_1800 | 0x0\_1bFF | 1Kb | DP1.4 KEY 285B+2BCRC |
| Bank7 | 0x0\_1c00 | 0x0\_1FFF | 1Kb |
| Bank8 | 0x0\_2000 | 0x0\_23FF | 1Kb |
| Bank9 | 0x0\_2400 | 0x0\_27FF | 1Kb | DP2.2 KEY 387B+2BCRC |
| Bank10 | 0x0\_2800 | 0x0\_2bFF | 1Kb |
| Bank11 | 0x0\_2c00 | 0x0\_2FFF | 1Kb |
| Bank12 | 0x0\_3000 | 0x0\_33FF | 1Kb |
| Bank13 | 0x0\_3400 | 0x0\_37FF | 1Kb | 0x0\_3400-0x0\_3418: TS module 5x5bits 0x0\_3419-0x0\_341d: ABB 5bits |
| Bank14 | 0x0\_3800 | 0x0\_3bFF | 1Kb | reserved |
| Bank15 | 0x0\_3c00 | 0x0\_3FFF | 1Kb | reserved |
| Bank16 | 0x0\_4000 | 0x0\_43FF | 1Kb | reserved |
| Bank17 | 0x0\_4400 | 0x0\_47FF | 1Kb | reserved |
| Bank18 | 0x0\_4800 | 0x0\_4bFF | 1Kb | reserved |
| Bank19 | 0x0\_4c00 | 0x0\_4FFF | 1Kb | reserved |
| Bank20 | 0x0\_5000 | 0x0\_53FF | 1Kb | reserved |
| Bank21 | 0x0\_5400 | 0x0\_57FF | 1Kb | reserved |
| Bank22 | 0x0\_5800 | 0x0\_5bFF | 1Kb | reserved |
| Bank23 | 0x0\_5c00 | 0x0\_5FFF | 1Kb | reserved |
| Bank24 | 0x0\_6000 | 0x0\_63FF | 1Kb | reserved |
| Bank25 | 0x0\_6400 | 0x0\_67FF | 1Kb | reserved |
| Bank26 | 0x0\_6800 | 0x0\_6bFF | 1Kb | reserved |
| Bank27 | 0x0\_6c00 | 0x0\_6FFF | 1Kb | reserved |
| Bank28 | 0x0\_7000 | 0x0\_73FF | 1Kb | reserved |
| Bank29 | 0x0\_7400 | 0x0\_77FF | 1Kb | reserved |
| Bank30 | 0x0\_7800 | 0x0\_7bFF | 1Kb | reserved |
| Bank31 | 0x0\_7c00 | 0x0\_7FFF | 1Kb | reserved |
| Bank32 | 0x0\_8000 | 0x0\_FFFF | 32Kb | reserved |
| Bank33 | 0x1\_0000 | 0x1\_7FFF | 32Kb | reserved |
| Bank34 | 0x1\_8000 | 0x1\_FFFF | 32Kb | reserved |

# 二、OTP仿真环境

在徐卓的scompile环境下可以进行OTP的相关仿真。需要配置的相关内容是：

1. myplusargs : 添加

setenv MY\_PLUSARGS\_RUN “+KP\_OTP\_LOADMAIN”

# 三、OTP相关的SOC Flow环境

OTP是包在secure\_subsys里面的，所以环境都是基于secure\_subsys的。

1. DC



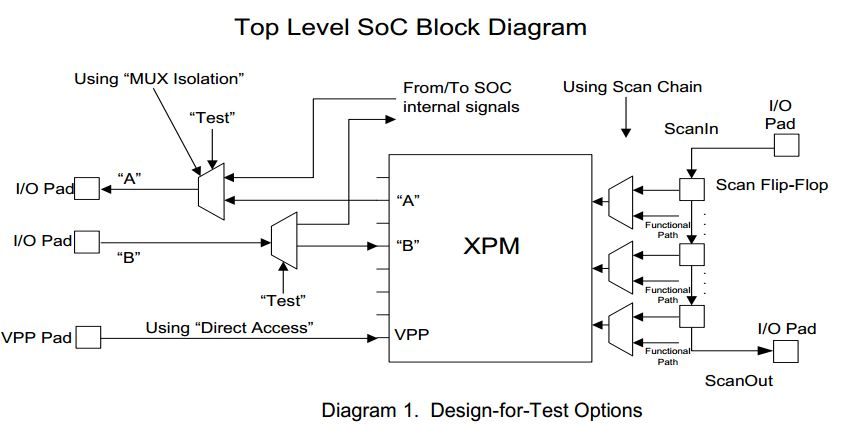
1. Formal Check



# 四、OTP ATE overview

## 1. XPM Design for Test guidelines

This document describes the design for test (DFT) guidelines, XPM test patterns and XPM test plan for ICs that incorporate Kilopass XPM Memory IP modules. This document applies to all the XPM memories  
in 28nm process node. This document is used in conjunction with datasheet.  
 The end user needs to provide top-level controllability and observability to implement DFT for embedded XPM module. To do this, a combination of the following methods is recommended: 1) Direct IP Access, 2) MUX’ed Isolation, 3) Scan Chain. With DFT properly implemented, XPM can be readily tested from the SoC top level. Diagram 1 shows the case where all of the above three methods are incorporated.



|  |  |  |  |
| --- | --- | --- | --- |
| Pad input | XPM input | Pad input | XPM input |
| QE0\_0\_PAD | A[0] | I2S\_SDO3\_PAD | A[14] |
| QE0\_1\_PAD | A[1] | I2S\_CLK3\_PAD | A[15] |
| QE0\_2\_PAD | A[2] | GP0\_PAD | A[16] |
| QE0\_3\_PAD | A[3] | GP1\_PAD | CEB |
| QE0\_4\_PAD | A[4] | GP2\_PAD | CLE |
| QE0\_5\_PAD | A[5] | GP3\_PAD | DLE |
| QE0\_6\_PAD | A[6] | GP4\_PAD | PGMEN |
| QE0\_7\_PAD | A[7] | GP5\_PAD | READEN |
| I2S\_WS2\_PAD | A[8] | GP6\_PAD | RSTB |
| I2S\_SDI2\_PAD | A[9] | GP7\_PAD | WEB |
| I2S\_SDO2\_PAD | A[10] | UART\_RX2\_PAD | CPUMPEN |
| I2S\_CLK2\_PAD | A[11] | UART\_TX2\_PAD | SELTM |
| I2S\_WS3\_PAD | A[12] | UART\_RX3\_PAD | DIN |
| I2S\_SDI3\_PAD | A[13] |  |  |
| Pad output | XPM output  (need SPI setting) | Pad output | XPM output  (need SPI setting) |
| PCLK1\_PAD | LOCK | QE1\_1\_PAD | D[4] |
| DE1\_PAD | D[0] | QE1\_2\_PAD | D[5] |
| VSYNC1\_PAD | D[1] | QE1\_3\_PAD | D[6] |
| HSYNC1\_PAD | D[2] | QE1\_4\_PAD | D[7] |
| QE1\_0\_PAD | D[3] |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Pad output | XPM output  (default) | Pad output | XPM output  (default) |
| QE1\_7\_PAD | D[0] | HSYNC0\_PAD | D[4] |
| PCLK0\_PAD | D[1] | HSYNC1\_PAD | D[5] |
| DE0\_PAD | D[2] | QE1\_0\_PAD | D[6] |
| VSYNC0\_PAD | D[3] | QE1\_1\_PAD | D[7] |

# 五. ATE test

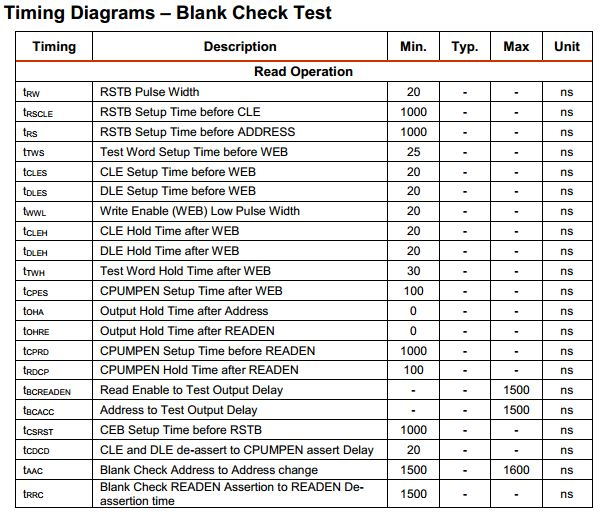
## 1 Blank Check Test (Defective Bit Screen Test)

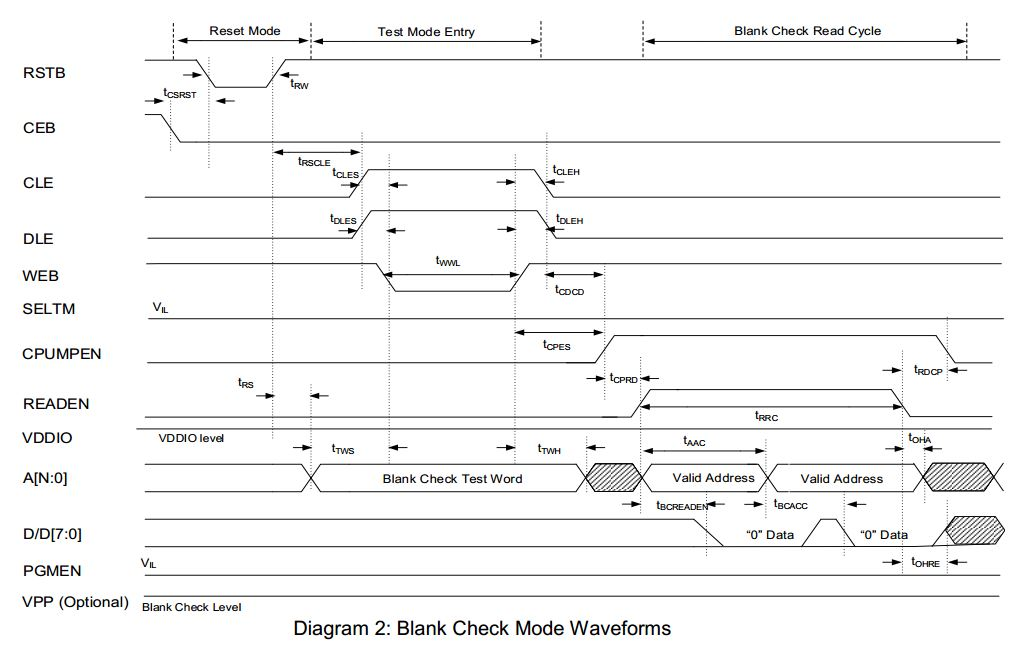
XPM memory technology depends on hard breakdown of gate oxide in standard logic CMOS process. Because of this dependency, it is important that integrity of gate oxide be checked before any tests or any programming is performed on XPM memory. This is done by reading every bit location with Blank Check Mode which ensures that defective bits caused due to defective gate oxides are screened out. If any of the bits don’t read a “0”, the chip is screened out.

This pattern is performed by enabling CLE=1 and DLE=1, latching test mode word “0x24” into the test command register and then making subsequent read operations to successive addresses until the entire XPM memory is read out. At the end of the Blank Check, perform a reset by asserting RSTB to get out of Blank Check test mode.

Blank check should only be performed once during the lifetime of the device. Running Blank check multiple times can cause permanent damage to the memory.

For IP macros without integrated programming charge pump, a voltage of 1.7V must be supplied to the VPP pin (VPP=1.7V) during Blankcheck test mode operation.





## 2 TESTDEC test (Wordline and Bitline Integrity Test)

TESTDEC mode is a built-in test mode. It enables users to verify the integrity of word-lines and bit-lines as well as screen out any gross defects in the peripheral logic. TESTDEC is to be performed on an unprogrammed unit. If TESTDEC is performed on a programmed unit, the result is undefined.

In 28HPM, 100% redundancy is built in. That means there are two arrays of memory inside: Main Array and Redundant Array. The TESTDEC command is used to test the two arrays. The test mode code for TESTDEC is “0x21”. TESTDEC test mode is initiated by asserting CLE and DLE high and latching the TESTDEC test mode code into the test command registers. After entering into the TESTDEC mode, make subsequent read operations to successive addresses while comparing the data with patterns provided in the table below until the entire memory array is read out. In TESTDEC mode, the SELTM input is used to select the Main and Redundant Arrays. When SELTM=0, the Main Array is addressed. When SELTM=1, the Redundant Array is addressed.

First, set SELTM=0 and verify the Main Array by reading out all the memory contents in Main Array. Second, verity the Redundant Array by setting SELTM=1 and read out all the memory contents in Redundant Array. After verifying Redundant Array, perform a reset by asserting RSTB to terminate TESTDEC test mode.

Notes:

(1). TESTDEC is valid only for un-programmed units.

(2). Diagram 3 shows the timing waveforms of TESTDEC test mode.

The expected test pattern is a variation of checkerboard pattern. If the expected test pattern is not observed, the chip is screened out. The expected test patterns are shown in the table 1 below for different modules.

Understanding Table 1:

Consider 64Kb XPM memory module (program by 1, read by 8). When reading out the 64Kb module during TESTDEC mode:

1) If address A<3> = 0, address A<10> = 0 and the rest are don’t cares, then the expected output

data is 8'h55

2) If address A<3> = 0, address A<10> = 1 and the rest are don’t cares, then the expected output

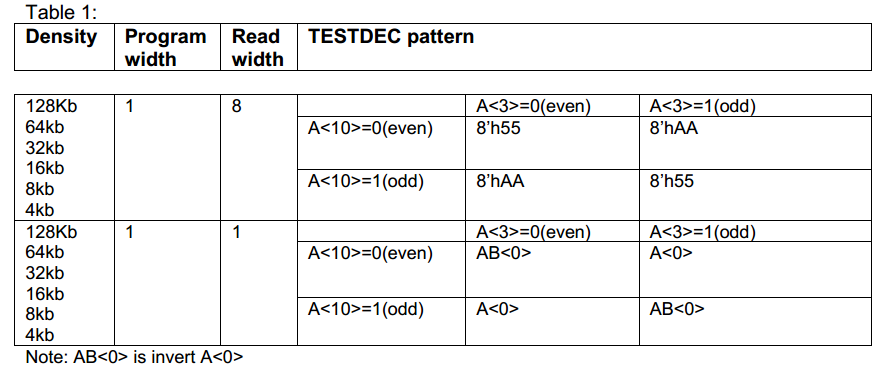
data is 8’hAA

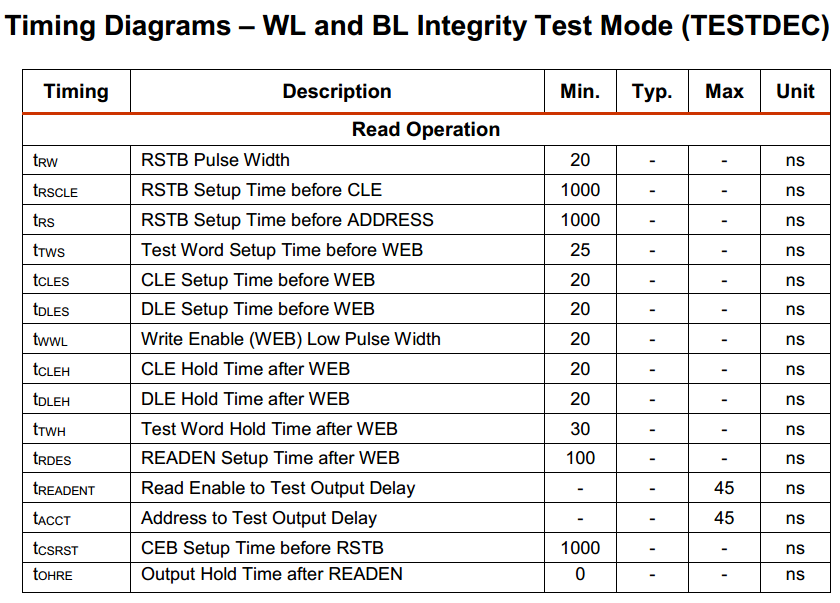
3) If address A<3> = 1, address A<10> = 0 and the rest are don’t cares, then the expected output

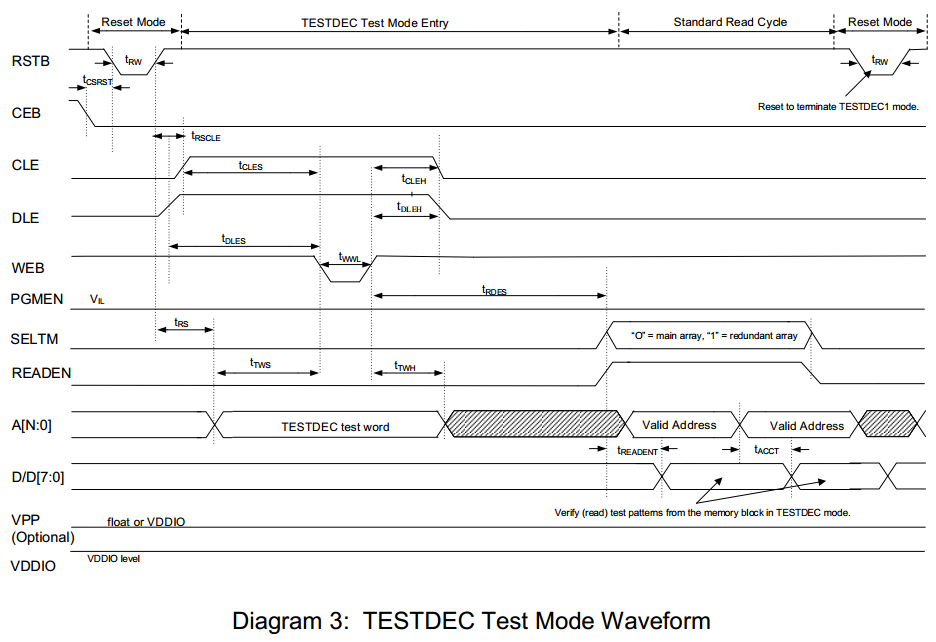
data is 8’hAA

4) If address A<3> = 1, address A<10> = 1 and the rest are don’t cares, then the expected output

data is 8’h55







## ~~3 WRTEST test (Pre-program Test)~~

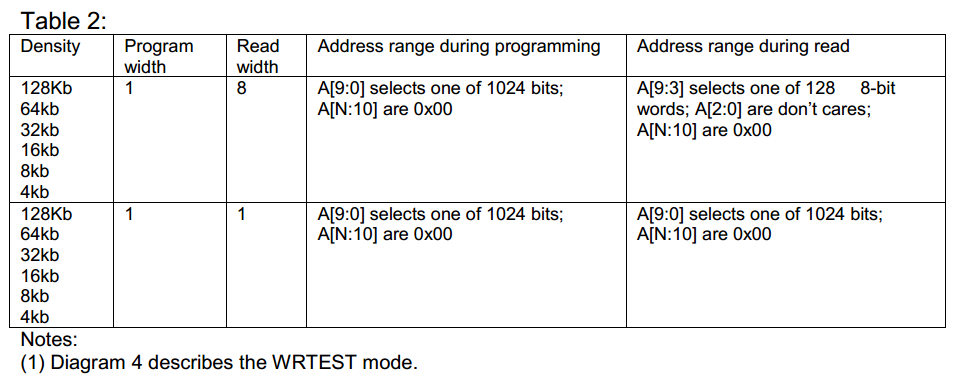
~~WRTEST mode enables an end user to screen out gross defects in programming circuitry before programming of the actual XPM memory array is done. This is enabled by availability of one spare row for programming.~~

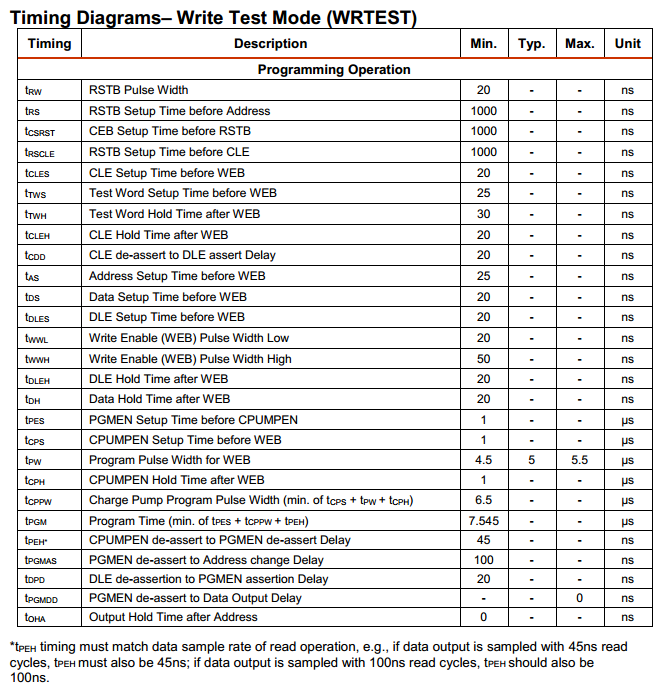
~~It is performed by asserting CLE and DLE high and latching test mode word “0x01” into the test command register, programming the spare row and verifying that spare row can be programmed correctly. When programming the spare row, follow the Smart Programming Algorithm described in the datasheet. If the spare bit/byte fails to program within the maximum number of programming pulses specified in the Smart Programming Algorithm, the chip is screened out.~~

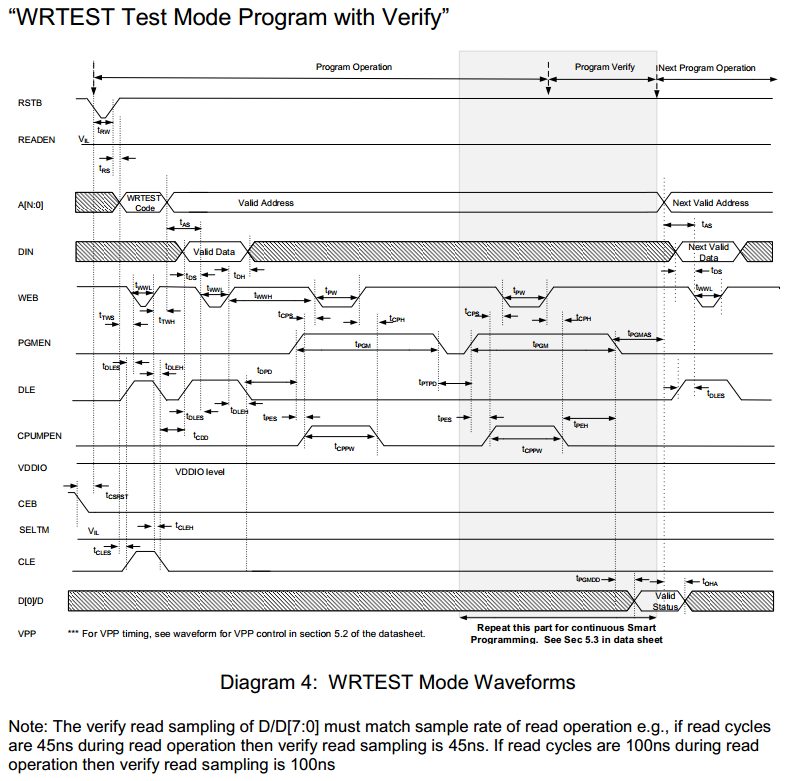
~~This mode is especially useful if the end user is planning to program the XPM memory in the field or insystem since it gives an indication of program circuitry functionality before field-programming or in-system programming is done. If the user is planning to program the XPM memory array in the factory during wafer sort, then this test mode can be skipped.~~

~~At the end of this test mode, a reset must be performed by asserting RSTB to exit this test mode and to go back to normal read or programming operations.~~

~~Table 2 specifies the valid address ranges for the WRTEST mode during programming and during read for the various XPM memories in the 28nm process node.~~

~~~~

~~~~

~~~~

## ~~4 Optional PGMLOCK test (Security Lock test)~~

~~PGMLOCK test applies ONLY to XPM memories which have LOCK output. The test enables an end user to permanently disable the memory programming circuitry and convert the memory to a “read only” memory after programming is complete.~~

~~PGMLOCK mode is performed by asserting CLE and DLE high and latching the test mode word “0x02”into the test command register. A standard programming sequence must follow (write data inputs during this operation are “don’t care”). Once "Security Lock" is programmed, the LOCK output will be "1". Once "Security Lock" is programmed, no other memory bits can be programmed. If a program operation is attempted, it will be internally inhibited and will have no effect on memory contents.~~

~~At the end of this test mode, a reset must be performed by asserting RSTB to exit this test mode and to go back to normal read operations.~~

~~Notes:~~

~~(1) Diagram 6 describes the PGMLOCK mode.~~

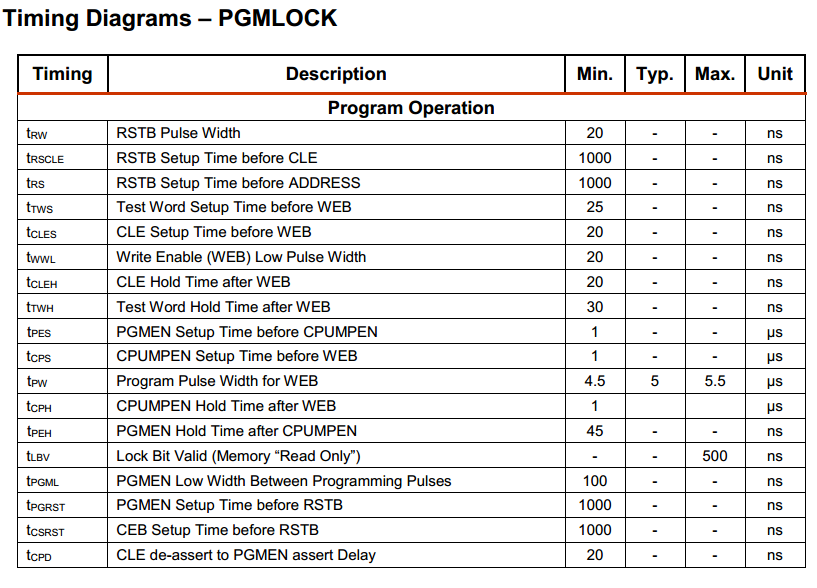
~~(2) In Diagrams 6, LOCK output initially comes up as “1” indicating the XPM memory comes up as locked i.e. read-only. At the positive edge of RSTB signal, evaluation takes place as to whether~~

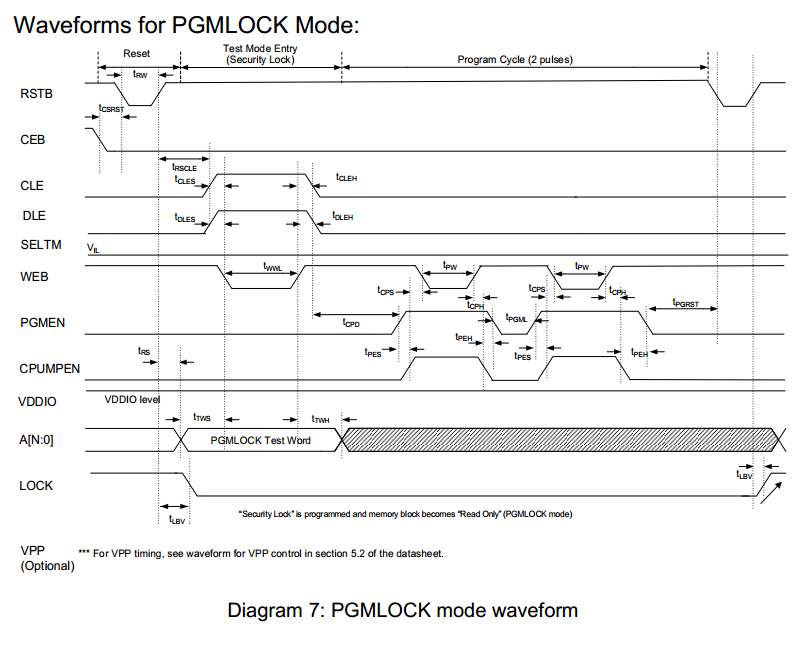
~~the “security lock” has been programmed or not programmed.~~

~~a. If it has been programmed, then LOCK output stays high indicating that the XPM memory cannot be programmed.~~

~~b. If it has not been programmed, then LOCK output goes low indicating that the XPM memory can be programmed. Once the end user programs the “security lock” by entering the PGMLOCK mode, RSTB has to be pulsed in order to make LOCK output go high as indicated in Diagram 6.~~

~~c. Two write pulses, no more, no less, are needed to program the Lock bit. Please see Diagram 6 for the waveforms.~~

~~~~

~~~~

## 5 Program Operation

