# THE DATASHEET OF

# M28W640HCB70N6F

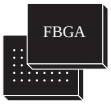
www.win-source.net 0086-755-83957316

# M28W640HCT
# M28W640HCB

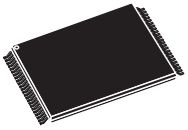
## 64 Mbit (4 Mb x 16, boot block) 3 V supply Flash memory

## Features

- Supply voltage
  - $V_{DD}$ = 2.7 V to 3.6 V
  - $V_{PP}$ = 12 V for fast program (optional)
- Access times: 70 ns
- Asynchronous Page Read mode
  - Page width: 4 words
  - Page access: 25 ns
  - Random access: 70 ns
- Programming time:
  - 10 µs typical
  - Double Word Programming option
  - Quadruple Word Programming option
- Common Flash interface
- Memory blocks
  - Parameter blocks (top or bottom location)
  - Main blocks
- Block locking
  - All blocks locked at power-up
  - Any combination of blocks can be locked
  - $\overline{WP}$ for block lock-down
- Security
  - 128 bit user programmable OTP cells
  - 64 bit unique device identifier
- Automatic standby mode
- Program and Erase Suspend
- 100,000 program/erase cycles per block
- Electronic signature
  - Manufacturer code: 20h
  - Top device code, M28W640HCT: 8848h
  - Bottom device code, M28W640HCB: 8849h




TFBGA48 (ZB)
6.39 x 10.5 mm



TSOP48 (N)
12 x 20 mm

- Packages
  - RoHS compliant

# Contents


**1 Description . . . . . 7**

**2 Signal descriptions . . . . . 12**

2.1 Address inputs (A0-A21) . . . . . 12

2.2 Data input/output (DQ0-DQ15) . . . . . 12

2.3 Chip Enable ($\overline{E}$) . . . . . 12

2.4 Output Enable ($\overline{G}$) . . . . . 12

2.5 Write Enable ($\overline{W}$) . . . . . 12

2.6 Write Protect ($\overline{WP}$) . . . . . 12

2.7 Reset ($\overline{RP}$) . . . . . 13

2.8 $V_{DD}$ supply voltage . . . . . 13

2.9 $V_{PP}$ program supply voltage . . . . . 13

2.10 $V_{SS}$ ground . . . . . 13

**3 Bus operations . . . . . 14**

3.1 Read . . . . . 14

3.2 Write . . . . . 14

3.3 Output Disable . . . . . 14

3.4 Standby . . . . . 14

3.5 Automatic Standby . . . . . 15

3.6 Reset . . . . . 15

**4 Command interface . . . . . 16**

4.1 Read Memory Array command . . . . . 16

4.2 Read Status Register command . . . . . 16

4.3 Read Electronic Signature command . . . . . 16

4.4 Read CFI Query command . . . . . 17

4.5 Block Erase command . . . . . 17

4.6 Program command . . . . . 18

4.7 Double Word Program command . . . . . 18

4.8 Quadruple Word Program command . . . . . 19

4.9 Clear Status Register command . . . . . 19

4.10 Program/Erase Suspend command . . . . . 19
4.11 Program/Erase Resume command . . . . . 20
4.12 Protection Register Program command . . . . . 20
4.13 Block Lock command . . . . . 20
4.14 Block Unlock command . . . . . 21
4.15 Block Lock-down command . . . . . 21

**5 Block locking . . . . . 25**
5.1 Reading a block's lock status . . . . . 25
5.2 Locked state . . . . . 25
5.3 Unlocked state . . . . . 25
5.4 Lock-down state . . . . . 26
5.5 Locking operations during Erase Suspend . . . . . 26

**6 Status Register . . . . . 28**
6.1 Program/Erase controller status (bit 7) . . . . . 28
6.2 Erase Suspend status (bit 6) . . . . . 28
6.3 Erase status (bit 5) . . . . . 29
6.4 Program status (bit 4) . . . . . 29
6.5 $V_{PP}$ status (bit 3) . . . . . 29
6.6 Program Suspend status (bit 2) . . . . . 29
6.7 Block Protection status (bit 1) . . . . . 30
6.8 Reserved (bit 0) . . . . . 30

**7 Maximum ratings . . . . . 31**

**8 DC and AC parameters . . . . . 32**

**9 Package mechanical . . . . . 42**

**10 Ordering information . . . . . 44**

**Appendix A Block address tables . . . . . 45**

**Appendix B Common Flash interface (CFI) . . . . . 53**

**Appendix C Flowcharts and pseudocodes. . . . . 59**

**Appendix D Command interface and Program/Erase controller state . . . . . 67**

**11 Revision history . . . . . 71**

# List of tables

Table 1. Signal names . . . 8
Table 2. Bus operations . . . 15
Table 3. Command codes. . . . 17
Table 4. Commands . . . 22
Table 5. Read electronic signature . . . 23
Table 6. Read block lock signature. . . . 23
Table 7. Read Protection Register and Lock Register . . . 23
Table 8. Program, Erase times and Program/Erase endurance cycles . . . 24
Table 9. Block Lock status . . . 26
Table 10. Protection status . . . 27
Table 11. Status Register bits. . . . 30
Table 12. Absolute maximum ratings . . . 31
Table 13. Operating and AC measurement conditions. . . . 32
Table 14. Capacitance . . . 33
Table 15. DC characteristics. . . . 34
Table 16. Read AC characteristics . . . 35
Table 17. Write AC characteristics, Write Enable controlled . . . 38
Table 18. Write AC characteristics, Chip Enable controlled . . . 40
Table 19. Power-up and Reset AC characteristics . . . 41
Table 20. TSOP48 - 48 lead plastic thin small outline, 12 x 20 mm, package mechanical data. . . . 42
Table 21. TFBGA48 6.39 x 10.5 mm - 8 x 6 ball array, 0.75 mm pitch, package mechanical data . . 43
Table 22. Ordering information scheme . . . 44
Table 23. Top boot block addresses, M28W640HCT. . . . 45
Table 24. Bottom boot block addresses, M28W640HCB . . . 49
Table 25. Query structure overview . . . 53
Table 26. CFI query identification string . . . 54
Table 27. CFI query system interface information . . . 55
Table 28. Device geometry definition . . . 56
Table 29. Primary algorithm-specific extended query table . . . 57
Table 30. Security code area . . . 58
Table 31. Write state machine current/next . . . 67
Table 32. Write state machine current/next . . . 69
Table 33. Document revision history . . . 71

# List of figures

Figure 1. Logic diagram . . . 8
Figure 2. TSOP connections . . . 9
Figure 3. TFBGA connections (top view through package) . . . 10
Figure 4. Block addresses . . . 11
Figure 5. Protection register memory map . . . 11
Figure 6. AC measurement I/O waveform . . . 32
Figure 7. AC measurement load circuit . . . 32
Figure 8. Read AC waveforms . . . 35
Figure 9. Page Read AC waveforms . . . 36
Figure 10. Write AC waveforms, Write Enable controlled . . . 37
Figure 11. Write AC waveforms, Chip Enable controlled . . . 39
Figure 12. Power-up and Reset AC waveforms . . . 41
Figure 13. TSOP48 - 48 lead plastic thin small outline, 12 x 20 mm, package outline . . . 42
Figure 14. TFBGA48 6.39 x 10.5 mm - 8 x 6 ball array, 0.75 mm pitch, bottom view package outline 43
Figure 15. Program flowchart and pseudocode . . . 59
Figure 16. Double Word Program flowchart and pseudocode . . . 60
Figure 17. Quadruple Word Program flowchart and pseudocode . . . 61
Figure 18. Program Suspend & Resume flowchart and pseudocode . . . 62
Figure 19. Erase flowchart and pseudocode . . . 63
Figure 20. Erase Suspend & Resume flowchart and pseudocode . . . 64
Figure 21. Locking operations flowchart and pseudocode . . . 65
Figure 22. Protection Register Program flowchart and pseudocode . . . 66

# 1 Description

The M28W640HCT and M28W640HCB are 64 Mbit (4 Mbit x 16) non-volatile Flash memories that can be erased electrically at block level and programmed in-system on a word-by-word basis using a 2.7 V to 3.6 V $V_{DD}$ supply. An optional 12V $V_{PP}$ power supply is provided to speed up customer programming.

The devices feature an asymmetrical blocked architecture. They have an array of 135 blocks: 8 parameter blocks of 4 Kwords and 127 main blocks of 32 Kwords. The M28W640HCT has the parameter blocks at the top of the memory address space while the M28W640HCB locates the parameter blocks starting from the bottom. The memory maps are shown in *Figure 4: Block addresses*.

The M28W640HCT and M28W640HCB feature an instant, individual block locking scheme that allows any block to be locked or unlocked with no latency, enabling instant code and data protection. All blocks have three levels of protection. They can be locked and locked-down individually preventing any accidental programming or erasure. There is an additional hardware protection against program and erase. When $V_{PP} \leq V_{PPLK}$ all blocks are protected against program or erase. All blocks are locked at power-up.

Each block can be erased separately. Erase can be suspended in order to perform either read or program in any other block and then resumed. Program can be suspended to read data in any other block and then resumed. Each block can be programmed and erased over 100,000 cycles.

The device includes a 192-bit protection register to increase the protection of a system design. The protection register is divided into a 64-bit segment and a 128-bit segment. The 64-bit segment contains a unique device number written by Numonyx, while the second one is one-time-programmable by the user. The user programmable segment can be permanently protected. *Figure 5*, shows the protection register memory map.

Program and Erase commands are written to the command interface of the memory. An on-chip Program/Erase controller takes care of the timings necessary for program and erase operations. The end of a program or erase operation can be detected and any error conditions identified. The command set required to control the memory is consistent with JEDEC standards.

The memory is offered in TSOP48 (12 × 20 mm) and TFBGA48 (6.39 × 10.5 mm, 0.75 mm pitch) packages and is supplied with all the bits erased (set to ’1’).

**Figure 1. Logic diagram**

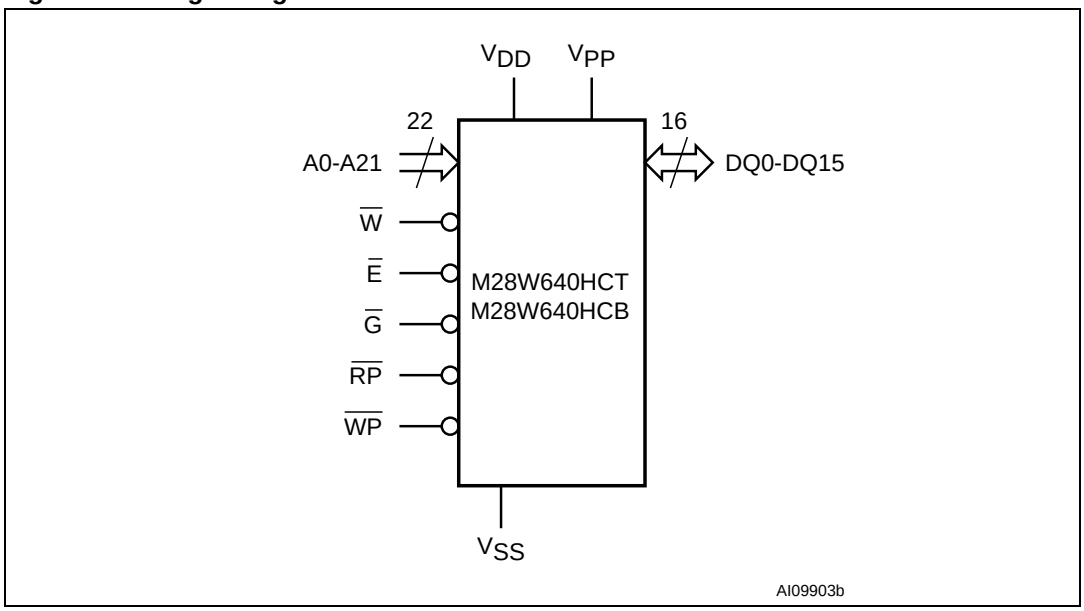
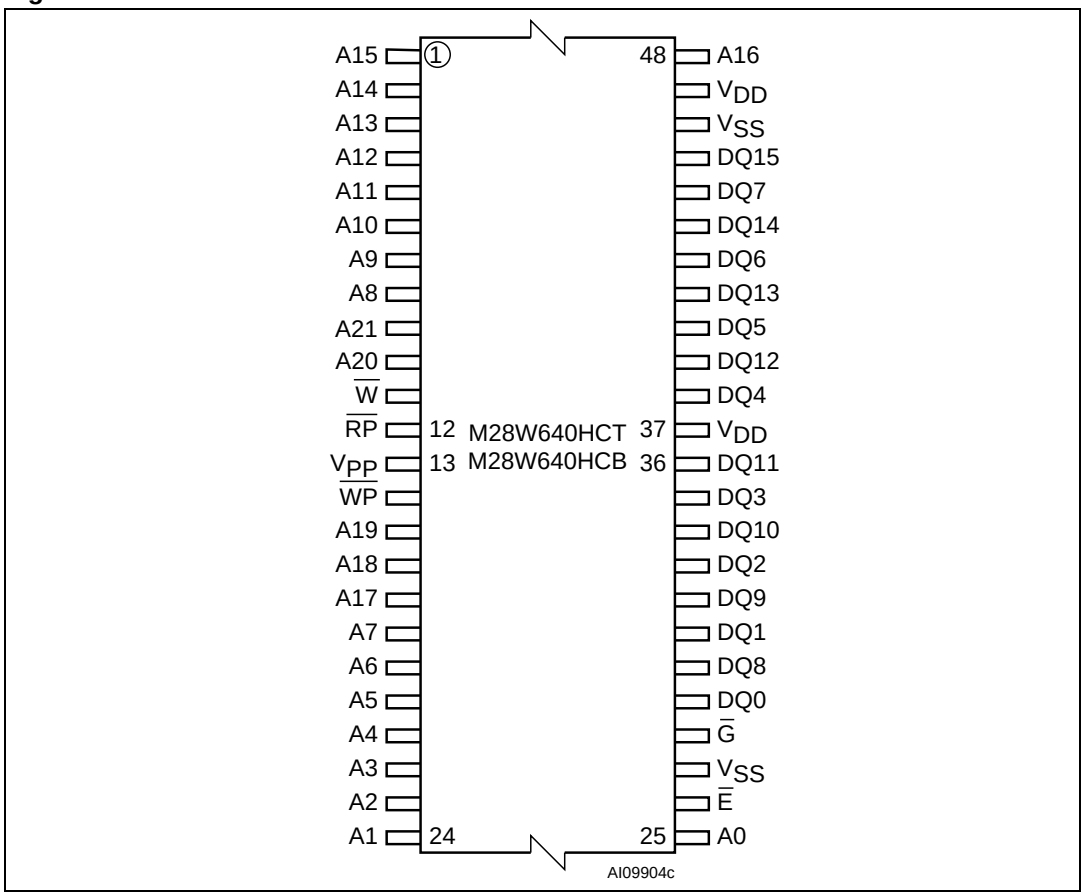



**Table 1. Signal names**

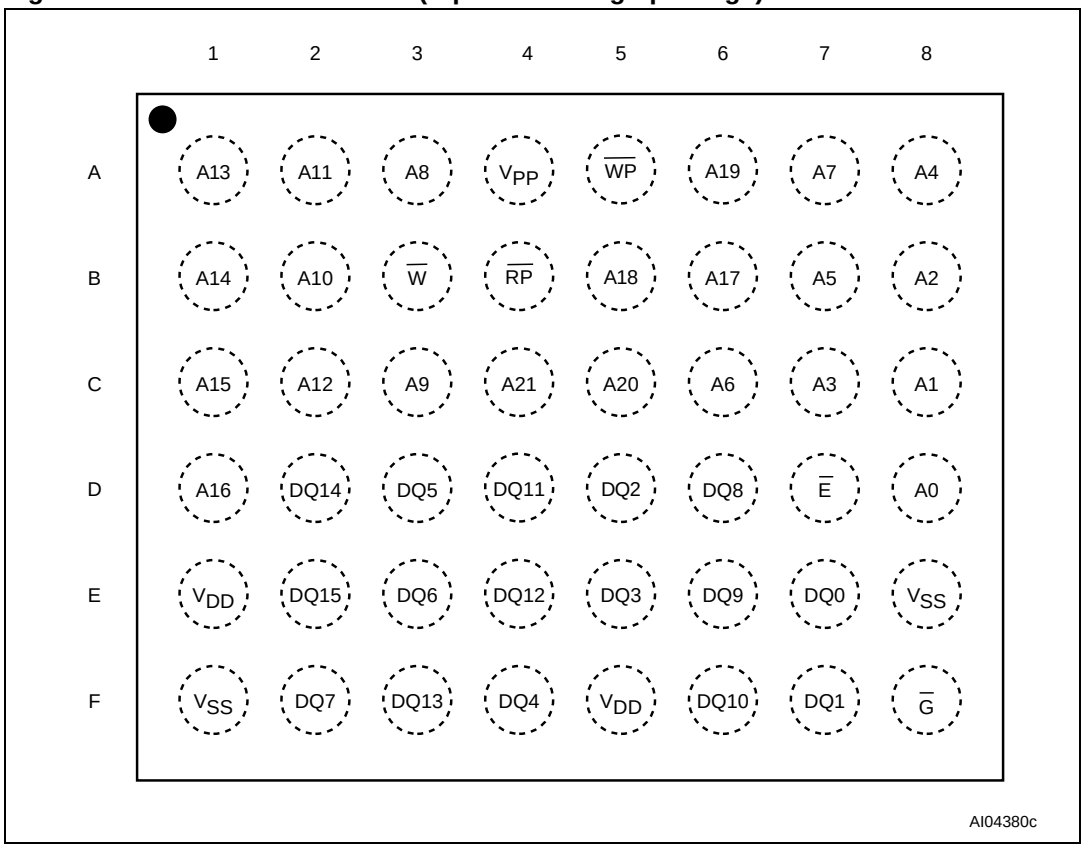
| Name | Description | Direction |
|---|---|---|
| A0-A21 | Address inputs | Inputs |
| DQ0-DQ15 | Data input/output | I/O |
| $\overline{E}$ | Chip Enable | Input |
| $\overline{G}$ | Output Enable | Input |
| $\overline{W}$ | Write Enable | Input |
| $\overline{RP}$ | Reset | Input |
| $\overline{WP}$ | Write Protect | Input |
| $V_{DD}$ | Power supply | Power supply |
| $V_{PP}$ | Optional supply voltage for fast program & erase | Power supply |
| $V_{SS}$ | Ground | Power supply |
| NC | Not connected internally | – |

**Figure 2. TSOP connections**



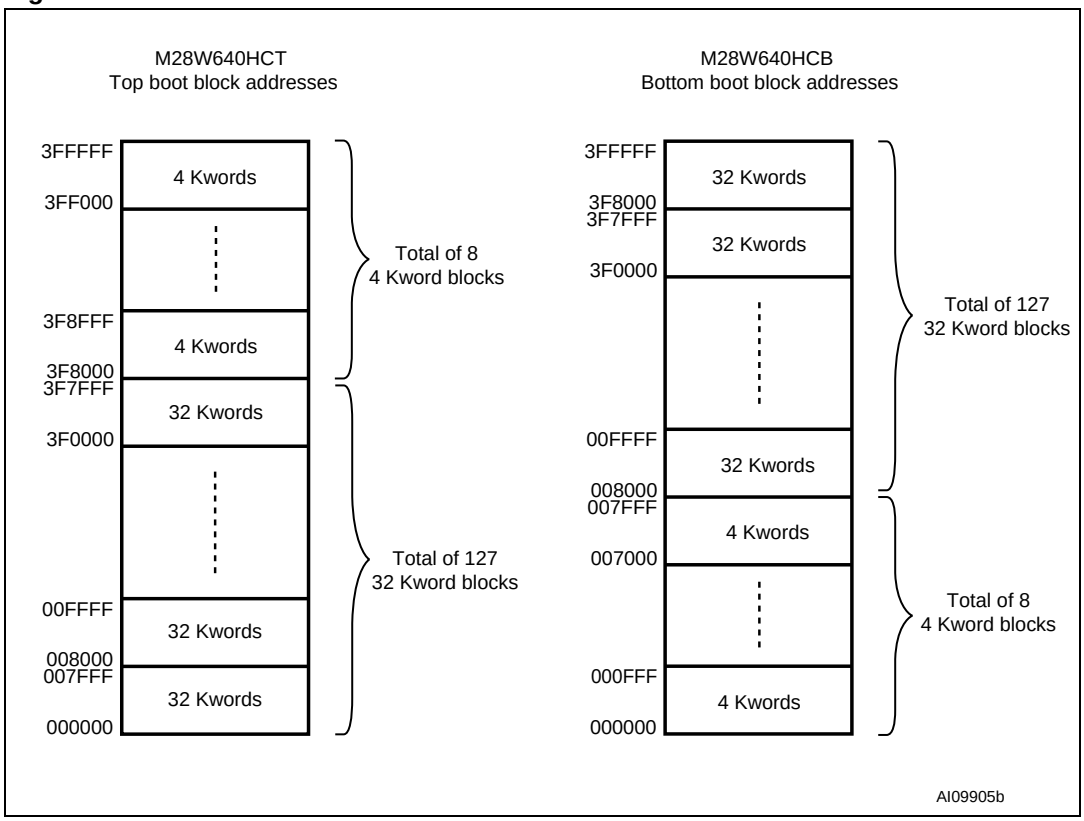

1. All $V_{DD}$ pins must be connected to the power supply.
2. All $V_{SS}$ pins must be connected to the ground.

**Figure 3. TFBGA connections (top view through package)**



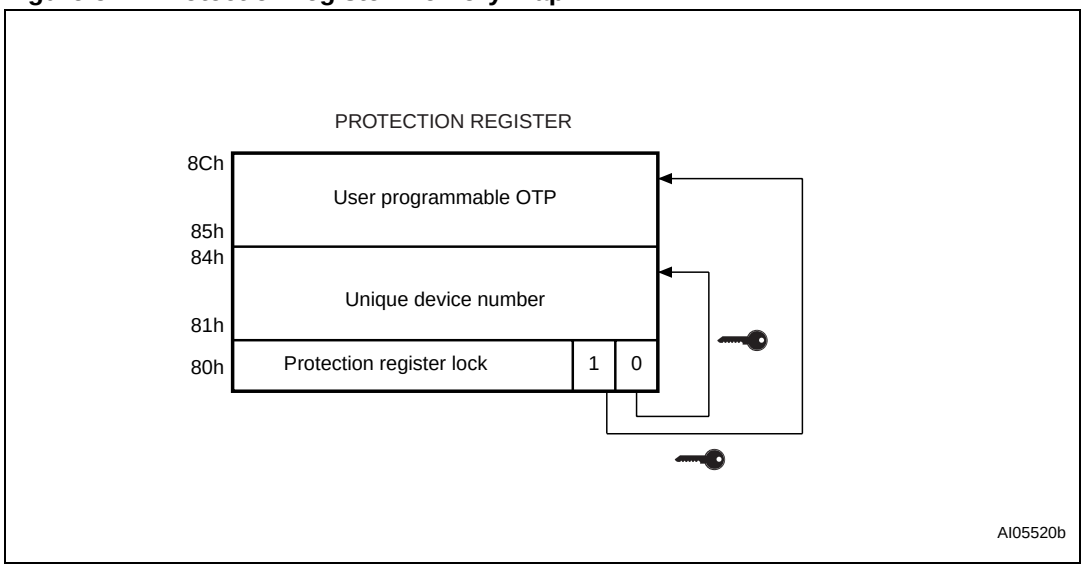

1. All $V_{DD}$ pins must be connected to the power supply.
2. All $V_{SS}$ pins must be connected to the ground.

**Figure 4. Block addresses**




1. Also see *Appendix A*, Tables *23* and *24* for a full listing of the block addresses.

**Figure 5. Protection register memory map**



# 2 Signal descriptions

See *Figure 1: Logic diagram* and *Table 1: Signal names*, for a brief overview of the signals connected to this device.

## 2.1 Address inputs (A0-A21)

The Address inputs select the cells in the memory array to access during bus read operations. During bus write operations they control the commands sent to the command interface of the internal state machine.

## 2.2 Data input/output (DQ0-DQ15)

The Data I/O outputs the data stored at the selected address during a bus read operation or inputs a command or the data to be programmed during a write bus operation.

## 2.3 Chip Enable ($\overline{E}$)

The Chip Enable input activates the memory control logic, input buffers, decoders and sense amplifiers. When Chip Enable is at $V_{IL}$and Reset is at $V_{IH}$ the device is in active mode. When Chip Enable is at $V_{IH}$ the memory is deselected, the outputs are high impedance and the power consumption is reduced to the standby level.

## 2.4 Output Enable ($\overline{G}$)

The Output Enable controls data outputs during the bus read operation of the memory.

## 2.5 Write Enable ($\overline{W}$)

The Write Enable controls the bus write operation of the memory's command interface. The data and address inputs are latched on the rising edge of Chip Enable, $\overline{E}$, or Write Enable, $\overline{W}$, whichever occurs first.

## 2.6 Write Protect ($\overline{WP}$)

Write Protect is an input that gives an additional hardware protection for each block. When Write Protect is at $V_{IL}$, the lock-down is enabled and the protection status of the block cannot be changed. When Write Protect is at $V_{IH}$, the lock-down is disabled and the block can be locked or unlocked (refer to *Table 7: Read Protection Register and Lock Register*).

## 2.7 Reset ($\overline{RP}$)

The Reset input provides a hardware reset of the memory. When Reset is at $V_{IL}$, the memory is in reset mode: the outputs are high impedance and the current consumption is minimized. After Reset all blocks are in the locked state. When Reset is at $V_{IH}$, the device is in normal operation. Exiting reset mode the device enters read array mode, but a negative transition of Chip Enable or a change of the address is required to ensure valid data outputs.

## 2.8 $V_{DD}$ supply voltage

$V_{DD}$ provides the power supply to the internal core and the I/O pins of the memory device. It is the main power supply for all operations (read, program and erase).

## 2.9 $V_{PP}$ program supply voltage

$V_{PP}$ is both a control input and a power supply pin. The two functions are selected by the voltage range applied to the pin. The supply voltage, $V_{DD}$, and the program supply voltage, $V_{PP}$, can be applied in any order.

If $V_{PP}$ is kept in a low voltage range (0 V to 3.6 V) $V_{PP}$ is seen as a control input. In this case a voltage lower than $V_{PPLK}$ gives an absolute protection against program or erase, while $V_{PP} > V_{PP1}$ enables these functions (see *Table 15: DC characteristics*, for the relevant values). $V_{PP}$ is only sampled at the beginning of a program or erase; a change in its value after the operation has started does not have any effect on program or erase, however for Double or Quadruple Word Program the results are uncertain.

If $V_{PP}$ is in the range 11.4 V to 12.6 V it acts as a power supply pin. In this condition $V_{PP}$ must be stable until the Program/Erase algorithm is completed (see *Table 17* and *Table 18*).

## 2.10 $V_{SS}$ ground

$V_{SS}$ is the reference for all voltage measurements.

*Note: Each device in a system should have $V_{DD}$ and $V_{PP}$ decoupled with a 0.1 µF capacitor close to the pin. See Figure 7: AC measurement load circuit. The PCB track widths should be sufficient to carry the required $V_{PP}$ program and erase currents.*

# 3 Bus operations

There are six standard bus operations that control the device. These are Bus Read, Bus Write, Output Disable, Standby, Automatic Standby and Reset. See *Table 2: Bus operations*, for a summary.

Typically glitches of less than 5 ns on Chip Enable or Write Enable are ignored by the memory and do not affect bus operations.

## 3.1 Read

Read bus operations are used to output the contents of the memory array, the Electronic Signature, the Status Register and the common Flash interface. Both Chip Enable and Output Enable must be at $V_{IL}$ in order to perform a read operation. The Chip Enable input should be used to enable the device. Output Enable should be used to gate data onto the output. The data read depends on the previous command written to the memory (see *Section 4: Command interface*). See *Figure 8: Read AC waveforms*, and *Table 16: Read AC characteristics*, for details of when the output becomes valid.

Read operations of the memory array can be performed in asynchronous page mode, which provides a fast access time. Data is internally read and stored in a page buffer. The page has a size of 4 words and is addressed by A0-A1 address inputs. Read operations of the electronic signature, the Status Register, the command Flash interface, the Block Protection status, the Configuration Register status and the security code are performed as asynchronous read cycles (Random Read). Both Chip Enable, $\overline{E}$, and Output Enable, $\overline{G}$, must be at $V_{IL}$ in order to read the output of the memory (see *Figure 9: Page Read AC waveforms*).

Read mode is the default state of the device when exiting reset or after power-up.

## 3.2 Write

Bus write operations write commands to the memory or latch input data to be programmed. A write operation is initiated when Chip Enable and Write Enable are at $V_{IL}$ with Output Enable at $V_{IH}$. Commands, input data and addresses are latched on the rising edge of Write Enable or Chip Enable, whichever occurs first.

See *Figure 9* and *Figure 11*, Write AC waveforms, and *Table 17* and *Table 18*, Write AC characteristics, for details of the timing requirements.

## 3.3 Output Disable

The data outputs are high impedance when the Output Enable is at $V_{IH}$.

## 3.4 Standby

Standby disables most of the internal circuitry allowing a substantial reduction of the current consumption. The memory is in standby when Chip Enable is at $V_{IH}$ and the device is in read mode. The power consumption is reduced to the standby level and the outputs are set

to high impedance, independently from the Output Enable or Write Enable inputs. If Chip Enable switches to $V_{IH}$ during a program or erase operation, the device enters Standby mode when finished.

## 3.5 Automatic Standby

Automatic standby provides a low power consumption state during Read mode. Following a read operation, the device enters automatic standby after 150 ns of bus inactivity even if Chip Enable is Low, $V_{IL}$, and the supply current is reduced to $I_{DD1}$. The data inputs/outputs will still output data if a bus read operation is in progress.

## 3.6 Reset

During Reset mode when Output Enable is Low, $V_{IL}$, the memory is deselected and the outputs are high impedance. The memory is in Reset mode when Reset is at $V_{IL}$. The power consumption is reduced to the standby level, independently from the Chip Enable, Output Enable or Write Enable inputs. If Reset is pulled to $V_{SS}$ during a program or erase, this operation is aborted and the memory content is no longer valid.

**Table 2. Bus operations(1)**

| Operation | $\overline{E}$ | $\overline{G}$ | $\overline{W}$ | $\overline{RP}$ | $\overline{WP}$ | $V_{PP}$ | DQ0-DQ15 |
|---|---|---|---|---|---|---|---|
| Bus Read | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | $V_{IH}$ | X | Don't care | Data output |
| Bus Write | $V_{IL}$ | $V_{IH}$ | $V_{IL}$ | $V_{IH}$ | X | $V_{DD}$ or $V_{PPH}$ | Data input |
| Output Disable | $V_{IL}$ | $V_{IH}$ | $V_{IH}$ | $V_{IH}$ | X | Don't care | Hi-Z |
| Standby | $V_{IH}$ | X | X | $V_{IH}$ | X | Don't care | Hi-Z |
| Reset | X | X | X | $V_{IL}$ | X | Don't care | Hi-Z |

1. X = $V_{IL}$ or $V_{IH}$, $V_{PPH}$ = 12 V ± 5%.

# 4 Command interface

All bus write operations to the memory are interpreted by the command interface. Commands consist of one or more sequential bus write operations. An internal Program/Erase controller handles all timings and verifies the correct execution of the Program and Erase commands. The Program/Erase controller provides a Status Register whose output may be read at any time to monitor the progress of the operation, or the Program/Erase states. See *Table 3: Command codes*, for a summary of the commands and see *Appendix D*, *Table 31: Write state machine current/next*, for a summary of the command interface.

The command interface is reset to Read mode when power is first applied, when exiting from reset or whenever $V_{DD}$ is lower than $V_{LKO}$. Command sequences must be followed exactly. Any invalid combination of commands will reset the device to Read mode. Refer to *Table 4: Commands*, in conjunction with the text descriptions below.

## 4.1 Read Memory Array command

The Read command returns the memory to its Read mode. One Bus Write cycle is required to issue the Read Memory Array command and return the memory to Read mode. Subsequent read operations will read the addressed location and output the data. When a device Reset occurs, the memory defaults to Read mode.

## 4.2 Read Status Register command

The Status Register indicates when a program or erase operation is complete and the success or failure of the operation itself. Issue a Read Status Register command to read the Status Register's contents. Subsequent bus read operations read the Status Register at any address, until another command is issued. See *Table 11: Status Register bits*, for details on the definitions of the bits.

The Read Status Register command may be issued at any time, even during a program/erase operation. Any read attempt during a program/erase operation will automatically output the content of the Status Register.

## 4.3 Read Electronic Signature command

The Read Electronic Signature command reads the manufacturer and device codes and the Block Locking Status, or the Protection Register.

The Read Electronic Signature command consists of one write cycle, a subsequent read will output the manufacturer code, the device code, the Block Lock and Lock-Down Status, or the Protection and Lock Register. See Tables *5*, *6* and *7* for the valid address.

**Table 3. Command codes**

| Hex code | Command |
|---|---|
| 01h | Block Lock confirm |
| 10h | Program |
| 20h | Erase |
| 2Fh | Block Lock-down confirm |
| 30h | Double Word Program |
| 40h | Program |
| 50h | Clear Status Register |
| 56h | Quadruple Word Program |
| 60h | Block Lock, Block Unlock, Block Lock-down |
| 70h | Read Status Register |
| 90h | Read Electronic Signature |
| 98h | Read CFI Query |
| B0h | Program/Erase Suspend |
| C0h | Protection Register Program |
| D0h | Program/Erase Resume, Block Unlock confirm |
| FFh | Read Memory Array |

## 4.4 Read CFI Query command

The Read Query command is used to read data from the common Flash interface (CFI) memory area, allowing programming equipment or applications to automatically match their interface to the characteristics of the device. One Bus Write cycle is required to issue the Read Query command. Once the command is issued subsequent bus read operations read from the common Flash interface memory area. See *Appendix B: Common Flash interface (CFI)*, tables *25*, *26*, *27*, *28*, *29* and *30* for details on the information contained in the common Flash interface memory area.

## 4.5 Block Erase command

The Block Erase command can be used to erase a block. It sets all the bits within the selected block to '1'. All previous data in the block is lost. If the block is protected then the erase operation will abort, the data in the block will not be changed and the Status Register will output the error.

Two Bus Write cycles are required to issue the command:

- The first bus cycle sets up the Erase command
- The second latches the block address in the internal state machine and starts the Program/Erase controller.

If the second bus cycle is not Write Erase Confirm (D0h), Status Register bits b4 and b5 are set and the command aborts.

Erase aborts if Reset turns to $V_{IL}$. As data integrity cannot be guaranteed when the erase operation is aborted, the block must be erased again.

During erase operations the memory will accept the Read Status Register command and the Program/Erase Suspend command, all other commands will be ignored. Typical Erase times are given in *Table 8: Program, Erase times and Program/Erase endurance cycles*.

See *Appendix C*, *Figure 19: Erase flowchart and pseudocode*, for a suggested flowchart for using the Erase command.

## 4.6 Program command

The memory array can be programmed word-by-word. Two bus write cycles are required to issue the Program command:

- The first bus cycle sets up the Program command.
- The second latches the address and the data to be written and starts the Program/Erase controller.

During program operations the memory will accept the Read Status Register command and the Program/Erase Suspend command. Typical Program times are given in *Table 8: Program, Erase times and Program/Erase endurance cycles*.

Programming aborts if Reset goes to $V_{IL}$. As data integrity cannot be guaranteed when the program operation is aborted, the block containing the memory location must be erased and reprogrammed.

See *Appendix C*, *Figure 15: Program flowchart and pseudocode*, for the flowchart for using the Program command.

## 4.7 Double Word Program command

This feature is offered to improve the programming throughput, writing a page of two adjacent words in parallel.The two words must differ only for the address A0. Programming should not be attempted when $V_{PP}$ is not at $V_{PPH}$.

Three bus write cycles are necessary to issue the Double Word Program command:

- The first bus cycle sets up the Double Word Program command
- The second bus cycle latches the address and the data of the first word to be written
- The third bus cycle latches the address and the data of the second word to be written and starts the Program/Erase controller.

Read operations output the Status Register content after the programming has started. Programming aborts if Reset goes to $V_{IL}$. As data integrity cannot be guaranteed when the program operation is aborted, the block containing the memory location must be erased and reprogrammed.

See *Appendix C*, *Figure 16: Double Word Program flowchart and pseudocode* for the flowchart for using the Double Word Program command.

## 4.8 Quadruple Word Program command

This feature is offered to improve the programming throughput, writing a page of four adjacent words in parallel.The four words must differ only for the addresses A0 and A1. Programming should not be attempted when $V_{PP}$ is not at $V_{PPH}$.

Five bus write cycles are necessary to issue the Quadruple Word Program command:

- The first bus cycle sets up the Quadruple Word Program command.
- The second bus cycle latches the address and the data of the first word to be written
- The third bus cycle latches the address and the data of the second word to be written
- The fourth bus cycle latches the address and the data of the third word to be written
- The fifth bus cycle latches the address and the data of the fourth word to be written and starts the Program/Erase controller.

Read operations output the Status Register content after the programming has started. Programming aborts if Reset goes to $V_{IL}$. As data integrity cannot be guaranteed when the program operation is aborted, the block containing the memory location must be erased and reprogrammed.

See *Appendix C*, *Figure 17: Quadruple Word Program flowchart and pseudocode*, for the flowchart for using the Quadruple Word Program command.

## 4.9 Clear Status Register command

The Clear Status Register command can be used to reset bits 1, 3, 4 and 5 in the Status Register to ‘0’. One bus write cycle is required to issue the Clear Status Register command.

The bits in the Status Register do not automatically return to ‘0’ when a new Program or Erase command is issued. The error bits in the Status Register should be cleared before attempting a new Program or Erase command.

## 4.10 Program/Erase Suspend command

The Program/Erase Suspend command is used to pause a program or erase operation. One bus write cycle is required to issue the Program/Erase command and pause the Program/Erase controller.

During Program/Erase Suspend the command interface will accept the Program/Erase Resume, Read Array, Read Status Register, Read Electronic Signature and Read CFI Query commands. Additionally, if the suspend operation was Erase then the Program, Double Word Program, Quadruple Word Program, Block Lock, Block Lock-down or Protection Program commands will also be accepted. The block being erased may be protected by issuing the Block Protect, Block Lock or Protection Program commands. When the Program/Erase Resume command is issued the operation will complete. Only the blocks not being erased may be read or programmed correctly.

During a Program/Erase Suspend, the device can be placed in a pseudo-standby mode by taking Chip Enable to $V_{IH}$. Program/Erase is aborted if Reset turns to $V_{IL}$.

See *Appendix C*, *Figure 18: Program Suspend & Resume flowchart and pseudocode*, and *Figure 20: Erase Suspend & Resume flowchart and pseudocode*, for flowcharts for using the Program/Erase Suspend command.

## 4.11 Program/Erase Resume command

The Program/Erase Resume command can be used to restart the Program/Erase controller after a program/erase suspend operation has paused it. One Bus Write cycle is required to issue the command. Once the command is issued subsequent bus read operations read the Status Register.

See *Appendix C*, *Figure 18: Program Suspend & Resume flowchart and pseudocode*, and *Figure 20: Erase Suspend & Resume flowchart and pseudocode*, for flowcharts for using the Program/Erase Resume command.

## 4.12 Protection Register Program command

The Protection Register Program command is used to program the 128 bit user one-time-programmable (OTP) segment of the Protection Register. The segment is programmed 16 bits at a time. When shipped all bits in the segment are set to ‘1’. The user can only program the bits to ‘0’.

Two write cycles are required to issue the Protection Register Program command:

- The first bus cycle sets up the Protection Register Program command
- The second latches the address and the data to be written to the Protection Register and starts the Program/Erase controller.

Read operations output the Status Register content after the programming has started.

The segment can be protected by programming bit 1 of the Protection Lock Register (see *Figure 5: Protection register memory map*). Attempting to program a previously protected Protection Register will result in a Status Register error. The protection of the Protection Register is not reversible.

The Protection Register Program cannot be suspended.

## 4.13 Block Lock command

The Block Lock command is used to lock a block and prevent program or erase operations from changing the data in it. All blocks are locked at power-up or reset.

Two Bus Write cycles are required to issue the Block Lock command:

- The first bus cycle sets up the Block Lock command
- The second Bus Write cycle latches the block address.

The lock status can be monitored for each block using the Read Electronic Signature command. *Table 10* shows the protection status after issuing a Block Lock command.

The Block Lock bits are volatile, once set they remain set until a hardware reset or power-down/power-up. They are cleared by a Blocks Unlock command. Refer to the section, Block Locking, for a detailed explanation.

## 4.14 Block Unlock command

The Block Unlock command is used to unlock a block, allowing the block to be programmed or erased. Two Bus Write cycles are required to issue the Block Unlock command:

- The first bus cycle sets up the Block Unlock command
- The second Bus Write cycle latches the block address.

The lock status can be monitored for each block using the Read Electronic Signature command. *Table 10* shows the protection status after issuing a Block Unlock command. Refer to the *Section 5: Block locking*, for a detailed explanation.

## 4.15 Block Lock-down command

A locked block cannot be programmed or erased, or have its protection status changed when $\overline{WP}$ is Low, $V_{IL}$. When $\overline{WP}$ is High, $V_{IH,}$ the Lock-down function is disabled and the locked blocks can be individually unlocked by the Block Unlock command.

Two Bus Write cycles are required to issue the Block Lock-down command:

- The first bus cycle sets up the Block Lock command
- The second Bus Write cycle latches the block address.

The lock status can be monitored for each block using the Read Electronic Signature command. Locked-down blocks revert to the locked (and not locked-down) state when the device is reset on power-down. *Table 10* shows the protection status after issuing a Block Lock-down command. Refer to the *Section 5: Block locking* for a detailed explanation.

**Table 4. Commands[1]**

| Commands | Cycles | Bus write operations | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 1st cycle | | | 2nd cycle | | | 3rd cycle | | | 4th cycle | | | 5th cycle | | |
| | | Op. | Add | Data | Op. | Add | Data | Op. | Add | Data | Op. | Add | Data | Op. | Add | Data |
| Read Memory Array | 1+ | Write | X | FFh | Read | RA | RD | | | | | | | | | |
| Read Status Register | 1+ | Write | X | 70h | Read | X | SRD | | | | | | | | | |
| Read Electronic Signature | 1+ | Write | X | 90h | Read | SA[2] | IDh | | | | | | | | | |
| Read CFI Query | 1+ | Write | X | 98h | Read | QA | QD | | | | | | | | | |
| Erase | 2 | Write | X | 20h | Write | BA | D0h | | | | | | | | | |
| Program | 2 | Write | X | 40h or 10h | Write | PA | PD | | | | | | | | | |
| Double Word Program[3] | 3 | Write | X | 30h | Write | PA1 | PD1 | Write | PA2 | PD2 | | | | | | |
| Quadruple Word Program[4] | 5 | Write | X | 56h | Write | PA1 | PD1 | Write | PA2 | PD2 | Write | PA3 | PD3 | Write | PA4 | PD4 |
| Clear Status Register | 1 | Write | X | 50h | | | | | | | | | | | | |
| Program/Erase Suspend | 1 | Write | X | B0h | | | | | | | | | | | | |
| Program/Erase Resume | 1 | Write | X | D0h | | | | | | | | | | | | |
| Block Lock | 2 | Write | X | 60h | Write | BA | 01h | | | | | | | | | |
| Block Unlock | 2 | Write | X | 60h | Write | BA | D0h | | | | | | | | | |
| Block Lock-down | 2 | Write | X | 60h | Write | BA | 2Fh | | | | | | | | | |
| Protection Register Program | 2 | Write | X | C0h | Write | PRA | PRD | | | | | | | | | |

1. X = Don't care, RA=Read Address, RD=Read Data, SRD=Status Register Data, ID=Identifier (manufacturer and device code), QA=Query Address, QD=Query Data, BA=Block Address, PA=Program Address, PD=Program Data, PRA=Protection Register Address, PRD=Protection Register Data.
2. The signature addresses are listed in Tables *5*, *6* and *7*.
3. Program addresses 1 and 2 must be consecutive addresses differing only for A0.
4. Program addresses 1,2,3 and 4 must be consecutive addresses differing only for A0 and A1.

**Table 5. Read electronic signature(1)**

| Code | Device | $\overline{E}$ | $\overline{G}$ | $\overline{W}$ | A0 | A1 | A2-A7 | A8-A21 | DQ0-DQ7 | DQ8-DQ15 |
|---|---|---|---|---|---|---|---|---|---|---|
| Manufacturer code | | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | $V_{IL}$ | $V_{IL}$ | 0 | Don't care | 20h | 00h |
| Device code | M28W640HCT | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | $V_{IH}$ | $V_{IL}$ | 0 | Don't care | 48h | 88h |
| | M28W640HCB | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | $V_{IH}$ | $V_{IL}$ | 0 | Don't care | 49h | 88h |

1. $\overline{RP} = V_{IH}$.

**Table 6. Read block lock signature**

| Block status | $\overline{E}$ | $\overline{G}$ | $\overline{W}$ | A0 | A1 | A2-A7 | A8-A11 | A12-A21 | DQ0 | DQ1 | DQ2-DQ15 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Locked block | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | $V_{IL}$ | $V_{IH}$ | 0 | Don't care | Block address | 1 | 0 | 00h |
| Unlocked block | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | $V_{IL}$ | $V_{IH}$ | 0 | Don't care | Block address | 0 | 0 | 00h |
| Locked-down block | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | $V_{IL}$ | $V_{IH}$ | 0 | Don't care | Block address | X(1) | 1 | 00h |

1. A locked-down block can be locked ‘DQ0 = 1’ or unlocked ‘DQ0 = 0’; see *Section 5: Block locking*.

**Table 7. Read Protection Register and Lock Register**

| Word | $\overline{E}$ | $\overline{G}$ | $\overline{W}$ | A0-A7 | A8-A21 | DQ0 | DQ1 | DQ2 | DQ3-DQ7 | DQ8-DQ15 |
|---|---|---|---|---|---|---|---|---|---|---|
| Lock | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | 80h | Don't care | 0 | OTP Prot. data | 0 | 00h | 00h |
| Unique ID 0 | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | 81h | Don't care | ID data | ID data | ID data | ID data | ID data |
| Unique ID 1 | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | 82h | Don't care | ID data | ID data | ID data | ID data | ID data |
| Unique ID 2 | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | 83h | Don't care | ID data | ID data | ID data | ID data | ID data |
| Unique ID 3 | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | 84h | Don't care | ID data | ID data | ID data | ID data | ID data |
| OTP 0 | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | 85h | Don't care | OTP data | OTP data | OTP data | OTP data | OTP data |
| OTP 1 | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | 86h | Don't care | OTP data | OTP data | OTP data | OTP data | OTP data |
| OTP 2 | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | 87h | Don't care | OTP data | OTP data | OTP data | OTP data | OTP data |
| OTP 3 | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | 88h | Don't care | OTP data | OTP data | OTP data | OTP data | OTP data |
| OTP 4 | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | 89h | Don't care | OTP data | OTP data | OTP data | OTP data | OTP data |
| OTP 5 | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | 8Ah | Don't care | OTP data | OTP data | OTP data | OTP data | OTP data |
| OTP 6 | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | 8Bh | Don't care | OTP data | OTP data | OTP data | OTP data | OTP data |
| OTP 7 | $V_{IL}$ | $V_{IL}$ | $V_{IH}$ | 8Ch | Don't care | OTP data | OTP data | OTP data | OTP data | OTP data |

**Table 8. Program, Erase times and Program/Erase endurance cycles**

| Parameter | Test conditions | M28W640HCT, M28W640HCB | | | Unit |
|---|---|---|---|---|---|
| | | Min | Typ | Max | |
| Word Program | $V_{PP} = V_{DD}$ | | 10 | 200 | µs |
| Double Word Program | $V_{PP}$ = 12 V ± 5% | | 10 | 200 | µs |
| Quadruple Word Program | $V_{PP}$ = 12 V ± 5% | | 10 | 200 | µs |
| Main Block Program | $V_{PP}$ = 12 V ± 5% | | 0.16/0.08(1) | 5 | s |
| | $V_{PP} = V_{DD}$ | | 0.32 | 5 | s |
| Parameter Block Program | $V_{PP}$ = 12 V ± 5% | | 0.02/0.01(1) | 4 | s |
| | $V_{PP} = V_{DD}$ | | 0.04 | 4 | s |
| Main Block Erase | $V_{PP}$ = 12 V ± 5% | | 1 | 10 | s |
| | $V_{PP} = V_{DD}$ | | 1 | 10 | s |
| Parameter Block Erase | $V_{PP}$ = 12 V ± 5% | | 0.4 | 10 | s |
| | $V_{PP} = V_{DD}$ | | 0.4 | 10 | s |
| Program/Erase cycles (per block) | | 100,000 | | | cycles |

1. Typical time to program a main or parameter block using the Double Word Program and the Quadruple Word Program commands respectively.

# 5 Block locking

The M28W640HCT and M28W640HCB feature an instant, individual block locking scheme that allows any block to be locked or unlocked with no latency. This locking scheme has three levels of protection:

- Lock/unlock - this first level allows software-only control of block locking
- Lock-down - this second level requires hardware interaction before locking can be changed
- $V_{PP} \leq V_{PPLK}$ - the third level offers a complete hardware protection against program and erase on all blocks.

The protection status of each block can be set to 'Locked', 'Unlocked', and 'Lock-down'. *Table 10*, defines all of the possible protection states ($\overline{WP}$, DQ1, DQ0), and *Appendix C*, *Figure 21*, shows a flowchart for the locking operations.

## 5.1 Reading a block's lock status

The lock status of every block can be read in the Read Electronic Signature mode of the device. To enter this mode write 90h to the device. Subsequent reads at the address specified in *Table 6*, will output the protection status of that block. The lock status is represented by DQ0 and DQ1. DQ0 indicates the block lock/unlock status and is set by the Lock command and cleared by the Unlock command. It is also automatically set when entering Lock-down. DQ1 indicates the Lock-down status and is set by the Lock-down command. It cannot be cleared by software, only by a hardware reset or power-down.

The following sections explain the operation of the locking system.

## 5.2 Locked state

The default status of all blocks on power-up or after a hardware reset is 'Locked' (states (0,0,1) or (1,0,1)). Locked blocks are fully protected from any program or erase. Any program or erase operations attempted on a locked block will return an error in the Status Register. The status of a locked block can be changed to 'Unlocked' or 'Lock-down' using the appropriate software commands. An unlocked block can be locked by issuing the Lock command.

## 5.3 Unlocked state

Unlocked blocks (states (0,0,0), (1,0,0) (1,1,0)), can be programmed or erased. All unlocked blocks return to the locked state after a hardware reset or when the device is powered-down. The status of an unlocked block can be changed to 'Locked' or 'Locked-down' using the appropriate software commands. A locked block can be unlocked by issuing the Unlock command.

## 5.4 Lock-down state

Blocks that are Locked-down (state (0,1,x)) are protected from program and erase operations (as for locked blocks) but their protection status cannot be changed using software commands alone. A Locked or Unlocked block can be Locked-down by issuing the Lock-down command. Locked-down blocks revert to the locked state when the device is reset or powered-down.

The Lock-down function is dependent on the $\overline{WP}$ input pin. When $\overline{WP}$=0 ($V_{IL}$), the blocks in the Lock-down state (0,1,x) are protected from program, erase and protection status changes. When $\overline{WP}$=1 ($V_{IH}$) the Lock-down function is disabled (1,1,1) and Locked-down blocks can be individually unlocked to the (1,1,0) state by issuing the software command, where they can be erased and programmed. These blocks can then be relocked (1,1,1) and unlocked (1,1,0) as desired while $\overline{WP}$ remains High. When $\overline{WP}$ is Low, blocks that were previously Locked-down return to the Lock-down state (0,1,x) regardless of any changes made while $\overline{WP}$ was High. Device reset or power-down resets all blocks, including those in Lock-down or in a Locked state.

## 5.5 Locking operations during Erase Suspend

Changes to block lock status can be performed during an Erase Suspend by using the standard locking command sequences to unlock, lock or lock-down a block. This is useful in the case when another block needs to be updated while an erase operation is in progress.

To change block locking during an erase operation, first write the Erase Suspend command, then check the Status Register until it indicates that the erase operation has been suspended. Next write the desired Lock command sequence to a block and the Lock Status will be changed. After completing any desired lock, read, or program operations, resume the erase operation with the Erase Resume command.

If a block is Locked or Locked-down during an Erase Suspend of the same block, the locking status bits will be changed immediately, but when the erase is resumed, the erase operation will complete.

Locking operations cannot be performed during a Program Suspend. Refer to *Appendix D*, command interface and Program/Erase controller state, for detailed information on which commands are valid during Erase Suspend.

**Table 9. Block Lock status**

| Item | Address | Data |
|---|---|---|
| Block Lock configuration | xx002 | LOCK |
| Block is Unlocked | | DQ0=0 |
| Block is Locked | | DQ0=1 |
| Block is Locked-down | | DQ1=1 |

**Table 10. Protection status**

| Current Protection status(1) ($\overline{WP}$, DQ1, DQ0) | | Next Protection status(1) ($\overline{WP}$, DQ1, DQ0) | | | |
|---|---|---|---|---|---|
| **Current state** | **Program/Erase allowed** | **After Block Lock command** | **After Block Unlock command** | **After Block Lock-down command** | **After $\overline{WP}$ transition** |
| 1,0,0 | yes | 1,0,1 | 1,0,0 | 1,1,1 | 0,0,0 |
| 1,0,1(2) | no | 1,0,1 | 1,0,0 | 1,1,1 | 0,0,1 |
| 1,1,0 | yes | 1,1,1 | 1,1,0 | 1,1,1 | 0,1,1 |
| 1,1,1 | no | 1,1,1 | 1,1,0 | 1,1,1 | 0,1,1 |
| 0,0,0 | yes | 0,0,1 | 0,0,0 | 0,1,1 | 1,0,0 |
| 0,0,1(2) | no | 0,0,1 | 0,0,0 | 0,1,1 | 1,0,1 |
| 0,1,1 | no | 0,1,1 | 0,1,1 | 0,1,1 | 1,1,1 or 1,1,0 (3) |

1. The lock status is defined by the write protect pin and by DQ1 ('1' for a locked-down block) and DQ0 ('1' for a locked block) as read in the Read Electronic Signature command with A1 = $V_{IH}$ and A0 = $V_{IL}$.
2. All blocks are locked at power-up, so the default configuration is 001 or 101 according to $\overline{WP}$ status.
3. A $\overline{WP}$ transition to $V_{IH}$ on a locked block will restore the previous DQ0 value, giving a 111 or 110.

# 6 Status Register

The Status Register provides information on the current or previous program or erase operation. The various bits convey information and errors on the operation. To read the Status Register the Read Status Register command can be issued, refer to *Section 4.2: Read Status Register command*. To output the contents, the Status Register is latched on the falling edge of the Chip Enable or Output Enable signals, and can be read until Chip Enable or Output Enable returns to $V_{IH}$. Either Chip Enable or Output Enable must be toggled to update the latched data.

Bus read operations from any address always read the Status Register during program and erase operations.

The bits in the Status Register are summarized in *Table 11: Status Register bits*. Refer to *Table 11* in conjunction with the following text descriptions.

## 6.1 Program/Erase controller status (bit 7)

The Program/Erase controller status bit indicates whether the Program/Erase controller is active or inactive. When the Program/Erase controller status bit is Low (set to ‘0’), the Program/Erase controller is active; when the bit is High (set to ‘1’), the Program/Erase controller is inactive, and the device is ready to process a new command.

The Program/Erase controller Status is Low immediately after a Program/Erase Suspend command is issued until the Program/Erase controller pauses. After the Program/Erase controller pauses the bit is High.

During program, erase, operations the Program/Erase controller status bit can be polled to find the end of the operation. Other bits in the Status Register should not be tested until the Program/Erase controller completes the operation and the bit is High.

After the Program/Erase controller completes its operation the Erase Status, Program Status, $V_{PP}$ Status and Block Lock Status bits should be tested for errors.

## 6.2 Erase Suspend status (bit 6)

The Erase Suspend status bit indicates that an erase operation has been suspended or is going to be suspended. When the Erase Suspend status bit is High (set to ‘1’), a Program/Erase Suspend command has been issued and the memory is waiting for a Program/Erase Resume command.

The Erase Suspend Status should only be considered valid when the Program/Erase controller status bit is High (Program/Erase controller inactive). Bit 7 is set within 30 µs of the Program/Erase Suspend command being issued therefore the memory may still complete the operation rather than entering the Suspend mode.

When a Program/Erase Resume command is issued the Erase Suspend Status bit returns Low.

## 6.3 Erase status (bit 5)

The Erase status bit can be used to identify if the memory has failed to verify that the block has erased correctly. When the Erase status bit is High (set to '1'), the Program/Erase controller has applied the maximum number of pulses to the block and still failed to verify that the block has erased correctly. The Erase Status bit should be read once the Program/Erase controller status bit is High (Program/Erase controller inactive).

Once set High, the Erase Status bit can only be reset Low by a Clear Status Register command or a hardware reset. If set High it should be reset before a new Program or Erase command is issued, otherwise the new command will appear to fail.

## 6.4 Program status (bit 4)

The Program Status bit is used to identify a Program failure. When the Program Status bit is High (set to '1'), the Program/Erase controller has applied the maximum number of pulses to the byte and still failed to verify that it has programmed correctly. The Program Status bit should be read once the Program/Erase controller status bit is High (Program/Erase controller inactive).

Once set High, the Program status bit can only be reset Low by a Clear Status Register command or a hardware reset. If set High it should be reset before a new command is issued, otherwise the new command will appear to fail.

## 6.5 $V_{PP}$ status (bit 3)

The $V_{PP}$ status bit can be used to identify an invalid voltage on the $V_{PP}$ pin during program and erase operations. The $V_{PP}$ pin is only sampled at the beginning of a program or erase operation. Indeterminate results can occur if $V_{PP}$ becomes invalid during an operation.

When the $V_{PP}$ status bit is Low (set to '0'), the voltage on the $V_{PP}$ pin was sampled at a valid voltage; when the $V_{PP}$ status bit is High (set to '1'), the $V_{PP}$ pin has a voltage that is below the $V_{PP}$ Lockout voltage, $V_{PPLK}$, the memory is protected and program and erase operations cannot be performed.

Once set High, the $V_{PP}$ status bit can only be reset Low by a Clear Status Register command or a hardware reset. If set High it should be reset before a new Program or Erase command is issued, otherwise the new command will appear to fail.

## 6.6 Program Suspend status (bit 2)

The Program Suspend status bit indicates that a program operation has been suspended. When the Program Suspend status bit is High (set to '1'), a Program/Erase Suspend command has been issued and the memory is waiting for a Program/Erase Resume command. The Program Suspend status should only be considered valid when the Program/Erase controller status bit is High (Program/Erase controller inactive). Bit 2 is set within 5 µs of the Program/Erase Suspend command being issued therefore the memory may still complete the operation rather than entering the Suspend mode.

When a Program/Erase Resume command is issued the Program Suspend Status bit returns Low.

## 6.7 Block Protection status (bit 1)

The Block Protection status bit can be used to identify if a program or erase operation has tried to modify the contents of a locked block.

When the Block Protection status bit is High (set to ‘1’), a program or erase operation has been attempted on a locked block.

Once set High, the Block Protection status bit can only be reset Low by a Clear Status Register command or a hardware reset. If set High it should be reset before a new command is issued, otherwise the new command will appear to fail.

## 6.8 Reserved (bit 0)

Bit 0 of the Status Register is reserved. Its value must be masked.

*Note: Refer to Appendix C: Flowcharts and pseudocodes, for using the Status Register.*

**Table 11. Status Register bits(1)**

| Bit | Name | Logic level | Definition |
|---|---|---|---|
| 7 | P/E.controller status | '1' | Ready |
| | | '0' | Busy |
| 6 | Erase Suspend status | '1' | Suspended |
| | | '0' | In progress or completed |
| 5 | Erase status | '1' | Erase error |
| | | '0' | Erase success |
| 4 | Program status | '1' | Program error |
| | | '0' | Program success |
| 3 | $V_{PP}$ status | '1' | $V_{PP}$ invalid, abort |
| | | '0' | $V_{PP}$ OK |
| 2 | Program Suspend status | '1' | Suspended |
| | | '0' | In progress or completed |
| 1 | Block Protection status | '1' | Program/Erase on protected Block, abort |
| | | '0' | No operation to protected blocks |
| 0 | Reserved | | |

1. Logic level '1' is High, '0' is Low.

# 7 Maximum ratings

Stressing the device above the rating listed in *Table 12: Absolute maximum ratings* may cause permanent damage to the device. These are stress ratings only and operation of the device at these or any other conditions above those indicated in the operating sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability. Refer also to the Numonyx SURE Program and other relevant quality documents.

**Table 12. Absolute maximum ratings**

| Symbol | Parameter | Value | | Unit |
|---|---|---|---|---|
| | | Min | Max | |
| $T_A$ | Ambient operating temperature | – 40 | 85 | °C |
| $T_{BIAS}$ | Temperature under bias | – 40 | 125 | °C |
| $T_{STG}$ | Storage temperature | – 55 | 155 | °C |
| $V_{IO}$ | Input or output voltage | – 0.6 | $V_{DD}$+0.6 | V |
| $V_{DD}$ | Supply voltage | – 0.6 | 4.1 | V |
| $V_{PP}$ | Program voltage | – 0.6 | 13 | V |

# 8 DC and AC parameters

This section summarizes the operating and measurement conditions, and the DC and AC characteristics of the device. The parameters in the DC and AC characteristics tables that follow, are derived from tests performed under the measurement conditions summarized in *Table 13: Operating and AC measurement conditions*. Designers should check that the operating conditions in their circuit match the measurement conditions when relying on the quoted parameters.

**Table 13. Operating and AC measurement conditions**

| Parameter | M28W640HCT, M28W640HCB | | |
|---|---|---|---|
| | 70 ns | | Units |
| | Min | Max | |
| $V_{DD}$ supply voltage | 2.7 | 3.6 | V |
| Ambient operating temperature | –40 | 85 | °C |
| Load capacitance ($C_L$) | 50 | | pF |
| Input rise and fall times | | 5 | ns |
| Input pulse voltages | 0 to $V_{DD}$ | | V |
| Input and output timing reference voltages | $V_{DD}/2$ | | V |

**Figure 6. AC measurement I/O waveform**

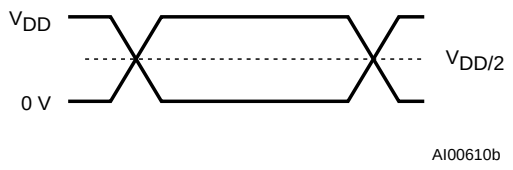



**Figure 7. AC measurement load circuit**

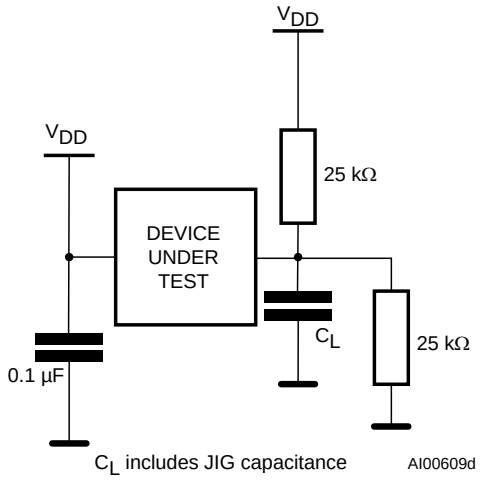


**Table 14. Capacitance(1)**

| Symbol | Parameter | Test condition | Min | Max | Unit |
|---|---|---|---|---|---|
| $C_{IN}$ | Input capacitance | $V_{IN}$ = 0 V | | 6 | pF |
| $C_{OUT}$ | Output capacitance | $V_{OUT}$ = 0 V | | 12 | pF |

1. Sampled only, not 100% tested.

**Table 15. DC characteristics**

| Symbol | Parameter | Test condition | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $I_{LI}$ | Input Leakage current | $0\ V \le V_{IN} \le V_{DD}$ | | | ±1 | µA |
| $I_{LO}$ | Output Leakage current | $0\ V \le V_{OUT} \le V_{DD}$ | | | ±10 | µA |
| $I_{DD}$ | Supply current (Read) | $\overline{E} = V_{SS}$, $\overline{G} = V_{IH}$, f = 5 MHz | | 9 | 18 | mA |
| $I_{DD1}$ | Supply current (Standby or Automatic Standby) | $\overline{E} = V_{DD} \pm 0.2$ V, $\overline{RP} = V_{DD} \pm 0.2$ V | | 15 | 50 | µA |
| $I_{DD2}$ | Supply current (Reset) | $\overline{RP} = V_{SS} \pm 0.2$ V | | 15 | 50 | µA |
| $I_{DD3}$ | Supply current (Program) | Program in progress $V_{PP} = 12\ V \pm 5\%$ | | 5 | 10 | mA |
| | | Program in progress $V_{PP} = V_{DD}$ | | 10 | 20 | mA |
| $I_{DD4}$ | Supply current (Erase) | Erase in progress $V_{PP} = 12\ V \pm 5\%$ | | 10 | 20 | mA |
| | | Erase in progress $V_{PP} = V_{DD}$ | | 10 | 20 | mA |
| $I_{DD5}$ | Supply current (Program/Erase Suspend) | $\overline{E} = V_{DD} \pm 0.2$ V, Erase suspended | | 15 | 50 | µA |
| $I_{PP}$ | Program current (Read or Standby) | $V_{PP} > V_{DD}$ | | | 400 | µA |
| $I_{PP1}$ | Program current (Read or Standby) | $V_{PP} \le V_{DD}$ | | 1 | 5 | µA |
| $I_{PP2}$ | Program current (Reset) | $\overline{RP} = V_{SS} \pm 0.2$ V | | 1 | 5 | µA |
| $I_{PP3}$ | Program current (Program) | Program in progress $V_{PP} = 12\ V \pm 5\%$ | | 1 | 10 | mA |
| | | Program in progress $V_{PP} = V_{DD}$ | | 1 | 5 | µA |
| $I_{PP4}$ | Program Current (Erase) | Erase in progress $V_{PP} = 12V \pm 5\%$ | | 3 | 10 | mA |
| | | Erase in progress $V_{PP} = V_{DD}$ | | 1 | 5 | µA |
| $V_{IL}$ | Input Low voltage | | –0.5 | | 0.4 | V |
| $V_{IH}$ | Input High voltage | | $0.7V_{DD}$ | | $V_{DD} + 0.4$ | V |
| $V_{OL}$ | Output Low voltage | $I_{OL} = 100$ µA, $V_{DD} = V_{DD}$ min | | | 0.1 | V |
| $V_{OH}$ | Output High voltage | $I_{OH} = -100$ µA, $V_{DD} = V_{DD}$ min | $V_{DD} - 0.1$ | | | V |
| $V_{PP1}$ | Program voltage (program or erase operations) | | 2.7 | | 3.6 | V |
| $V_{PPH}$ | Program voltage (program or erase operations) | | 11.4 | | 12.6 | V |
| $V_{PPLK}$ | Program voltage (program and erase lock-out) | | | | 1 | V |
| $V_{LKO}$ | $V_{DD}$ supply voltage (Program and Erase lock-out) | | | | 2 | V |

**Figure 8. Read AC waveforms**

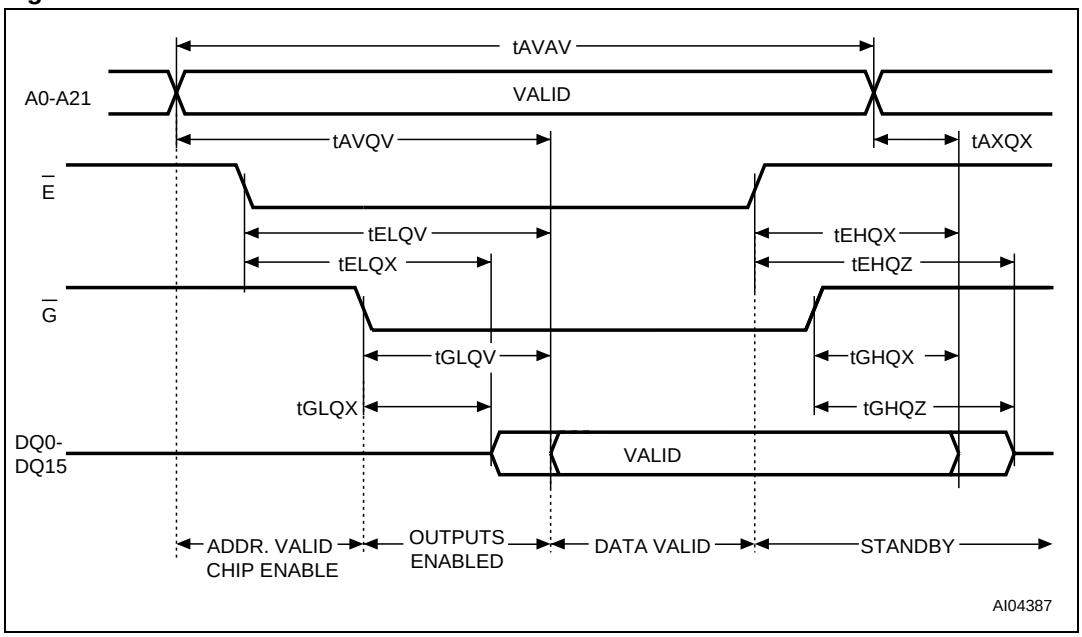



**Table 16. Read AC characteristics**

| Symbol | Alt | Parameter | | M28W640HCT, M28W640HCB | |
|---|---|---|---|---|---|
| | | | | 70 ns | Unit |
| $t_{AVAV}$ | $t_{RC}$ | Address valid to Next Address Valid | Min | 70 | ns |
| $t_{AVQV}$ | $t_{ACC}$ | Address valid to Random Output Valid | Max | 70 | ns |
| $t_{AXQX}$(1) | $t_{OH}$ | Address Transition to Output Transition | Min | 0 | ns |
| $t_{EHQX}$(1) | $t_{OH}$ | Chip Enable High to Output Transition | Min | 0 | ns |
| $t_{EHQZ}$(1) | $t_{HZ}$ | Chip Enable High to Output Hi-Z | Max | 20 | ns |
| $t_{ELQV}$(2) | $t_{CE}$ | Chip Enable Low to Output Valid | Max | 70 | ns |
| $t_{ELQX}$(1) | $t_{LZ}$ | Chip Enable Low to Output Transition | Min | 0 | ns |
| $t_{GHQX}$(1) | $t_{OH}$ | Output Enable High to Output Transition | Min | 0 | ns |
| $t_{GHQZ}$(1) | $t_{DF}$ | Output Enable High to Output Hi-Z | Max | 20 | ns |
| $t_{GLQV}$(2) | $t_{OE}$ | Output Enable Low to Output Valid | Max | 20 | ns |
| $t_{GLQX}$(1) | $t_{OLZ}$ | Output Enable Low to Output Transition | Min | 0 | ns |
| $t_{AVQV1}$ | $t_{PAGE}$ | Page address Valid to Page Output Valid | Max | 25 | ns |
| $t_{AXQX1}$ | $t_{OH}$ | Address Transition to Page Output Transition | Min | 0 | ns |

1. Sampled only, not 100% tested.
2. $\overline{G}$ may be delayed by up to $t_{ELQV}$ - $t_{GLQV}$ after the falling edge of $\overline{E}$ without increasing $t_{ELQV}$.

**Figure 9. Page Read AC waveforms**

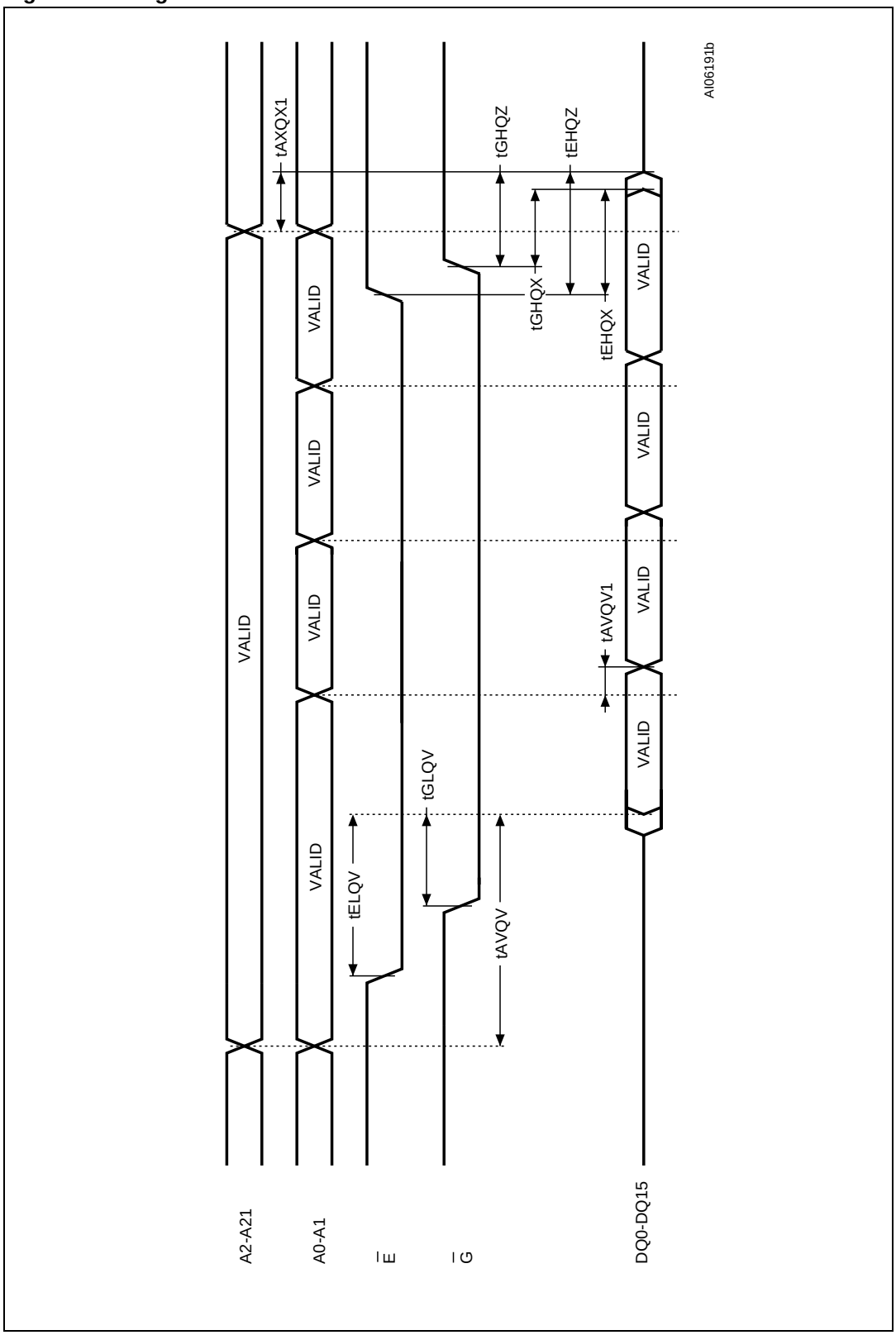


**Figure 10. Write AC waveforms, Write Enable controlled**

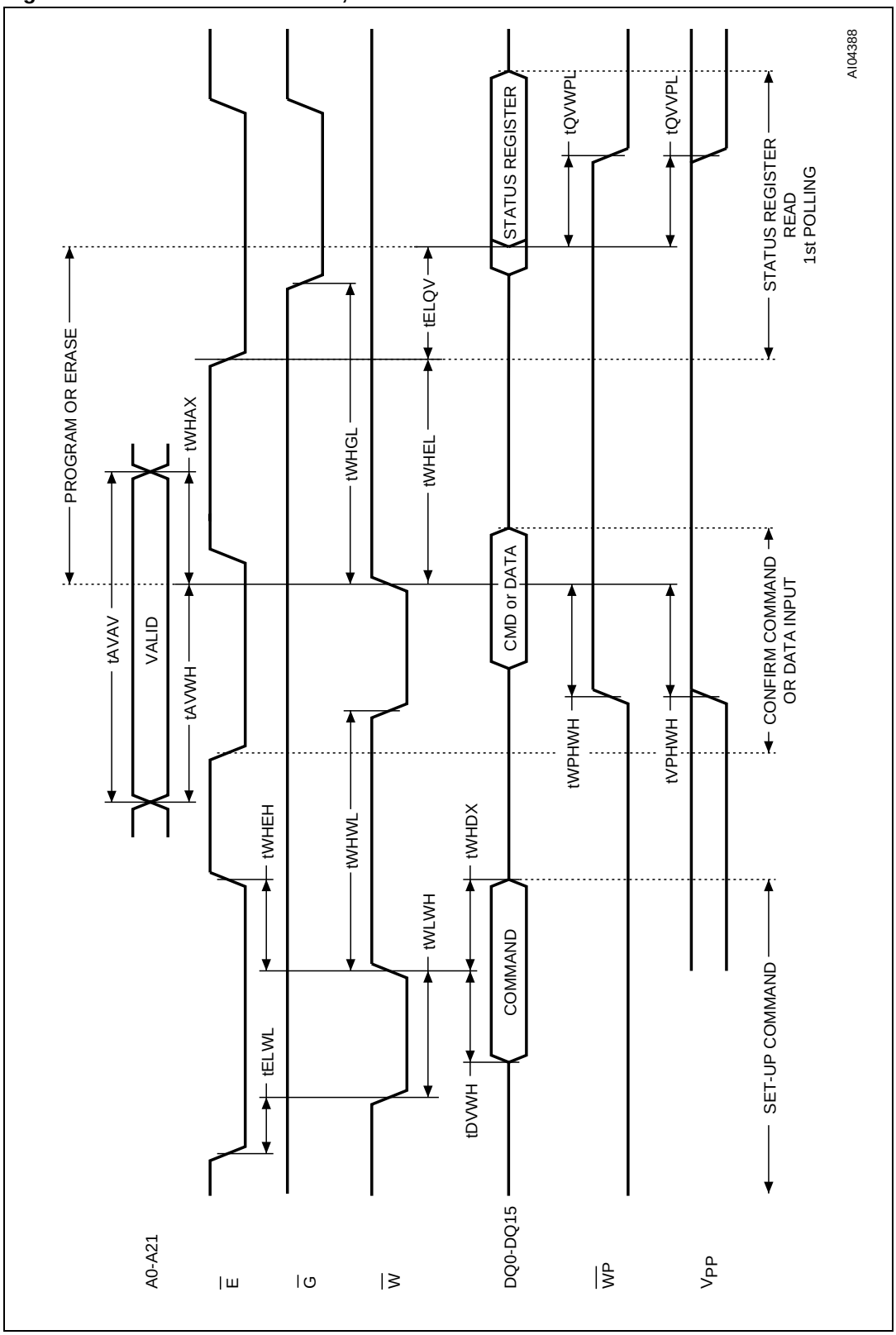


**Table 17. Write AC characteristics, Write Enable controlled**

| Symbol | Alt | Parameter | | M28W640HCT, M28W640HCB | |
|---|---|---|---|---|---|
| | | | | 70 ns | Unit |
| $t_{AVAV}$ | $t_{WC}$ | Write Cycle time | Min | 70 | ns |
| $t_{AVWH}$ | $t_{AS}$ | Address Valid to Write Enable High | Min | 45 | ns |
| $t_{DVWH}$ | $t_{DS}$ | Data Valid to Write Enable High | Min | 45 | ns |
| $t_{ELWL}$ | $t_{CS}$ | Chip Enable Low to Write Enable Low | Min | 0 | ns |
| $t_{ELQV}$ | | Chip Enable Low to Output Valid | Min | 70 | ns |
| $t_{QVVPL}$ (1)(2) | | Output Valid to $V_{PP}$ Low | Min | 0 | ns |
| $t_{QVWPL}$ | | Output Valid to Write Protect Low | Min | 0 | ns |
| $t_{VPHWH}$ (1) | $t_{VPS}$ | $V_{PP}$ High to Write Enable High | Min | 200 | ns |
| $t_{WHAX}$ | $t_{AH}$ | Write Enable High to Address Transition | Min | 0 | ns |
| $t_{WHDX}$ | $t_{DH}$ | Write Enable High to Data Transition | Min | 0 | ns |
| $t_{WHEH}$ | $t_{CH}$ | Write Enable High to Chip Enable High | Min | 0 | ns |
| $t_{WHEL}$ | | Write Enable High to Chip Enable Low | Min | 25 | ns |
| $t_{WHGL}$ | | Write Enable High to Output Enable Low | Min | 20 | ns |
| $t_{WHWL}$ | $t_{WPH}$ | Write Enable High to Write Enable Low | Min | 25 | ns |
| $t_{WLWH}$ | $t_{WP}$ | Write Enable Low to Write Enable High | Min | 45 | ns |
| $t_{WPHWH}$ | | Write Protect High to Write Enable High | Min | 45 | ns |

1. Sampled only, not 100% tested.
2. Applicable if $V_{PP}$ is seen as a logic input ($V_{PP}$ < 3.6 V).

**Figure 11. Write AC waveforms, Chip Enable controlled**

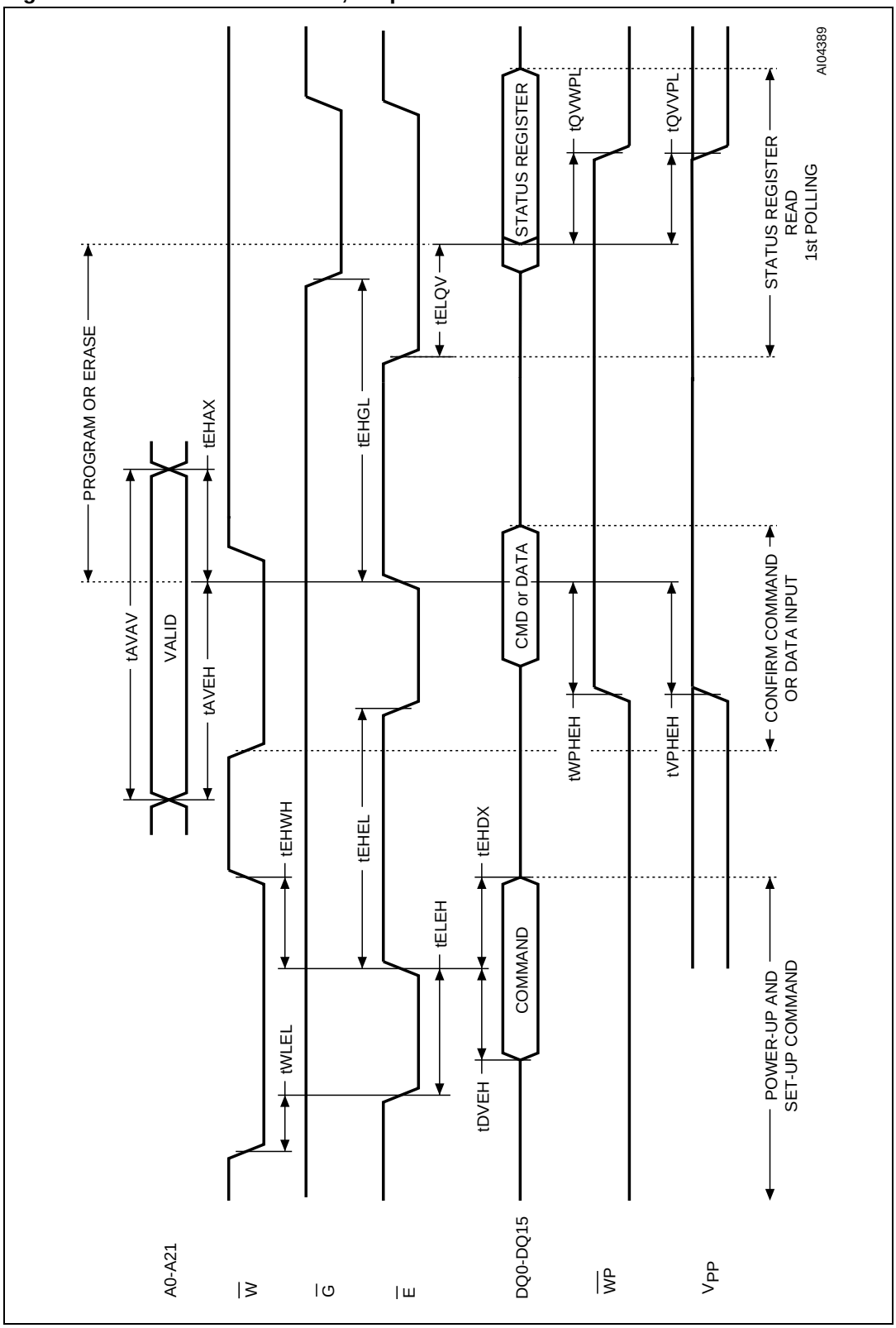


**Table 18. Write AC characteristics, Chip Enable controlled**

| Symbol | Alt | Parameter | | M28W640HCT, M28W640HCB | |
|---|---|---|---|---|---|
| | | | | 70 ns | Unit |
| $t_{AVAV}$ | $t_{WC}$ | Write Cycle time | Min | 70 | ns |
| $t_{AVEH}$ | $t_{AS}$ | Address Valid to Chip Enable High | Min | 45 | ns |
| $t_{DVEH}$ | $t_{DS}$ | Data Valid to Chip Enable High | Min | 45 | ns |
| $t_{EHAX}$ | $t_{AH}$ | Chip Enable High to Address Transition | Min | 0 | ns |
| $t_{EHDX}$ | $t_{DH}$ | Chip Enable High to Data Transition | Min | 0 | ns |
| $t_{EHEL}$ | $t_{CPH}$ | Chip Enable High to Chip Enable Low | Min | 25 | ns |
| $t_{EHGL}$ | | Chip Enable High to Output Enable Low | Min | 25 | ns |
| $t_{EHWH}$ | $t_{WH}$ | Chip Enable High to Write Enable High | Min | 0 | ns |
| $t_{ELEH}$ | $t_{CP}$ | Chip Enable Low to Chip Enable High | Min | 45 | ns |
| $t_{ELQV}$ | | Chip Enable Low to Output Valid | Min | 70 | ns |
| $t_{QVVPL}$ (1)(2) | | Output Valid to $V_{PP}$ Low | Min | 0 | ns |
| $t_{QVWPL}$ | | Data Valid to Write Protect Low | Min | 0 | ns |
| $t_{VPHEH}$ (1) | $t_{VPS}$ | $V_{PP}$ High to Chip Enable High | Min | 200 | ns |
| $t_{WLEL}$ | $t_{CS}$ | Write Enable Low to Chip Enable Low | Min | 0 | ns |
| $t_{WPHEH}$ | | Write Protect High to Chip Enable High | Min | 45 | ns |

1. Sampled only, not 100% tested.
2. Applicable if $V_{PP}$ is seen as a logic input ($V_{PP}$ < 3.6 V).

**Figure 12. Power-up and Reset AC waveforms**

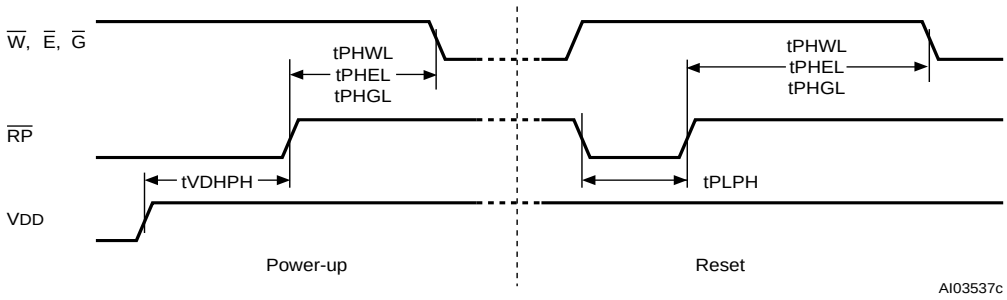



**Table 19. Power-up and Reset AC characteristics**

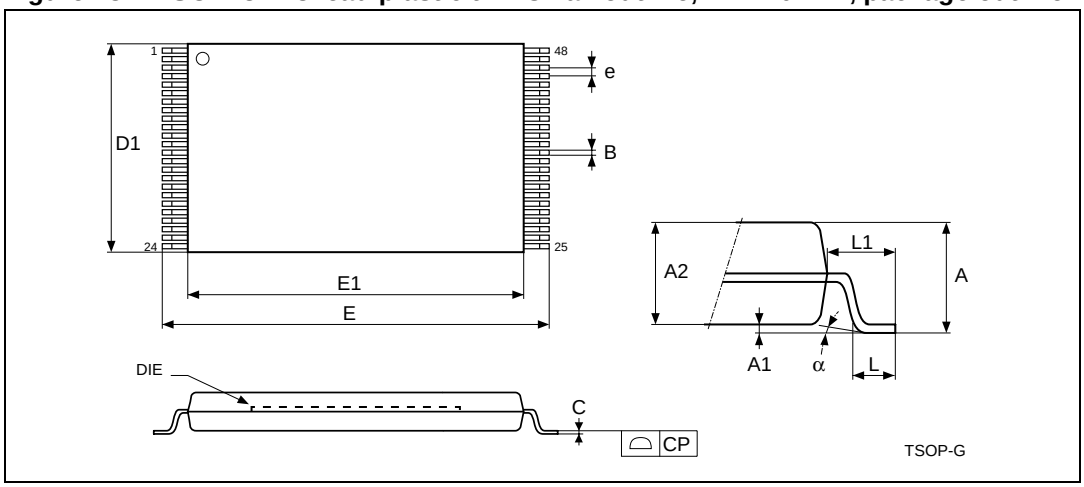
| Symbol | Parameter | Test condition | | M28W640HCT, M28W640HCB | |
|---|---|---|---|---|---|
| | | | | 70 ns | Unit |
| $t_{PHWL}$ $t_{PHEL}$ $t_{PHGL}$ | Reset High to Write Enable Low, Chip Enable Low, Output Enable Low | During program and erase | Min | 50 | µs |
| | | others | Min | 30 | ns |
| $t_{PLPH}$(1)(2) | Reset Low to Reset High | | Min | 100 | ns |
| $t_{VDHPH}$(3) | Supply voltages High to Reset High | | Min | 300 | µs |

1. The device reset is possible but not guaranteed if $t_{PLPH}$ < 100 ns.
2. Sampled only, not 100% tested.
3. It is important to assert $\overline{RP}$ in order to allow proper CPU initialization during power-up or reset.

# 9 Package mechanical

In order to meet environmental requirements, Numonyx offers these devices in RoHS compliant packages. RoHS compliant packages are lead-free. The category of second-level interconnect is marked on the package and on the inner box label, in compliance with JEDEC Standard JESD97. The maximum ratings related to soldering conditions are also marked on the inner box label.

**Figure 13. TSOP48 - 48 lead plastic thin small outline, 12 x 20 mm, package outline**

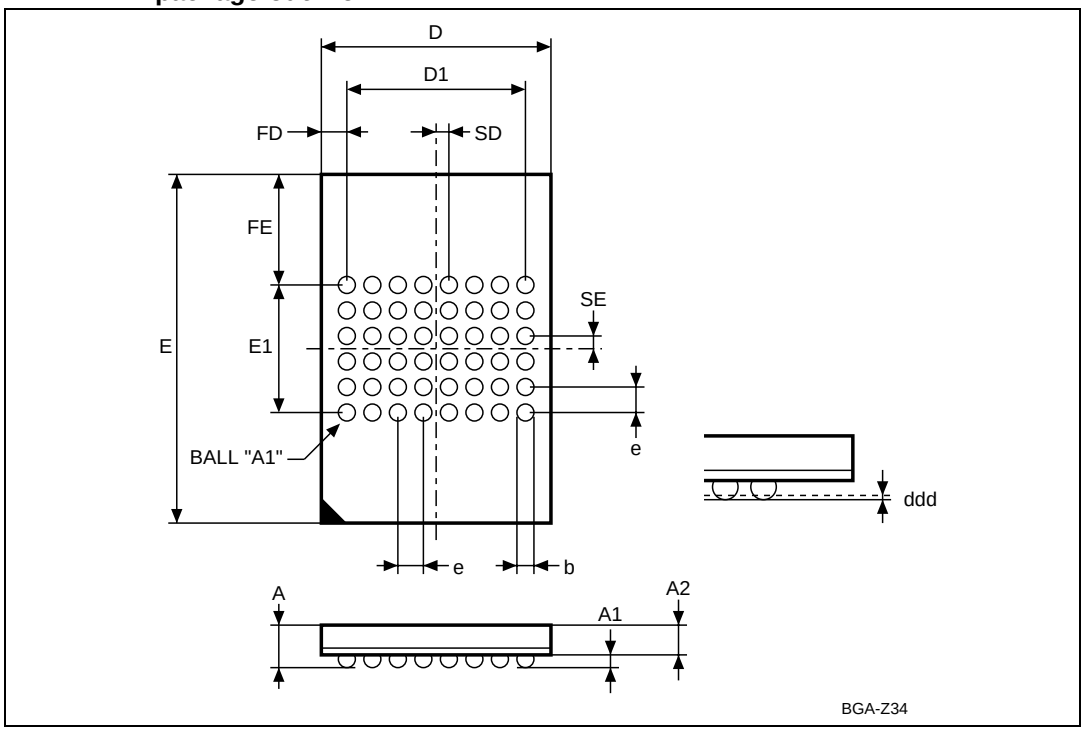



1. Drawing is not to scale.

**Table 20. TSOP48 - 48 lead plastic thin small outline, 12 x 20 mm, package mechanical data**

| Symbol | millimeters | | | inches | | |
|---|---|---|---|---|---|---|
| | Typ | Min | Max | Typ | Min | Max |
| A | | | 1.20 | | | 0.047 |
| A1 | 0.10 | 0.05 | 0.15 | 0.004 | 0.002 | 0.006 |
| A2 | 1.00 | 0.95 | 1.05 | 0.039 | 0.037 | 0.041 |
| B | 0.22 | 0.17 | 0.27 | 0.009 | 0.007 | 0.011 |
| C | | 0.10 | 0.21 | | 0.004 | 0.008 |
| CP | | | 0.10 | | | 0.004 |
| D1 | 12.00 | 11.90 | 12.10 | 0.472 | 0.468 | 0.476 |
| E | 20.00 | 19.80 | 20.20 | 0.787 | 0.779 | 0.795 |
| E1 | 18.40 | 18.30 | 18.50 | 0.724 | 0.720 | 0.728 |
| e | 0.50 | – | – | 0.020 | – | – |
| L | 0.60 | 0.50 | 0.70 | 0.024 | 0.020 | 0.028 |
| L1 | 0.80 | | | 0.031 | | |
| α | 3° | 0° | 5° | 3° | 0° | 5° |

**Figure 14. TFBGA48 6.39 x 10.5 mm - 8 x 6 ball array, 0.75 mm pitch, bottom view package outline**




1. Drawing is not to scale.

**Table 21. TFBGA48 6.39 x 10.5 mm - 8 x 6 ball array, 0.75 mm pitch, package mechanical data**

| Symbol | millimeters | | | inches | | |
|---|---|---|---|---|---|---|
| | Typ | Min | Max | Typ | Min | Max |
| A | | | 1.20 | | | 0.047 |
| A1 | | 0.26 | | | 0.010 | |
| A2 | | | 1.00 | | | 0.039 |
| b | 0.40 | 0.35 | 0.45 | 0.016 | 0.014 | 0.018 |
| D | 6.39 | 6.29 | 6.49 | 0.252 | 0.248 | 0.255 |
| D1 | 5.250 | – | – | 0.207 | – | – |
| ddd | | | 0.10 | | | 0.004 |
| E | 10.50 | 10.40 | 10.60 | 0.413 | 0.409 | 0.417 |
| E1 | 3.75 | – | – | 0.148 | – | – |
| e | 0.75 | – | – | 0.029 | – | – |
| FD | 0.57 | – | – | 0.022 | – | – |
| FE | 3.37 | – | – | 0.133 | – | – |
| SD | 0.37 | – | – | 0.015 | – | – |
| SE | 0.37 | – | – | 0.015 | – | – |

# 10 Ordering information

**Table 22. Ordering information scheme**

Example: M28W640HCT 70 N 6 E

**Device type**

M28

**Operating voltage**

W = $V_{DD}$ = 2.7 V to 3.6 V

**Device function**

640HC = 64 Mbit (4 Mb x 16), boot block

**Array matrix**

T = top boot

B = bottom boot

**Speed**

70 = 70 ns

**Package**

N = TSOP48: 12 x 20 mm

ZB = TFBGA48: 6.39 x 10.5 mm, 0.75 mm pitch

**Temperature range**

6 = –40 to 85 °C

**Option**

E = RoHS compliant package, standard packing

F = RoHS compliant package, tape & reel packing

*Note: Devices are shipped from the factory with the memory content bits erased to '1'. For a list of available options (speed, package, etc.) or for further information on any aspect of this device, please contact the Numonyx Sales Office nearest to you.*

# Appendix A Block address tables

**Table 23. Top boot block addresses, M28W640HCT**

| # | Size (Kword) | Address range |
|---|---|---|
| 0 | 4 | 3FF000-3FFFFF |
| 1 | 4 | 3FE000-3FEFFF |
| 2 | 4 | 3FD000-3FDFFF |
| 3 | 4 | 3FC000-3FCFFF |
| 4 | 4 | 3FB000-3FBFFF |
| 5 | 4 | 3FA000-3FAFFF |
| 6 | 4 | 3F9000-3F9FFF |
| 7 | 4 | 3F8000-3F8FFF |
| 8 | 32 | 3F0000-3F7FFF |
| 9 | 32 | 3E8000-3EFFFF |
| 10 | 32 | 3E0000-3E7FFF |
| 11 | 32 | 3D8000-3DFFFF |
| 12 | 32 | 3D0000-3D7FFF |
| 13 | 32 | 3C8000-3CFFFF |
| 14 | 32 | 3C0000-3C7FFF |
| 15 | 32 | 3B8000-3BFFFF |
| 16 | 32 | 3B0000-3B7FFF |
| 17 | 32 | 3A8000-3AFFFF |
| 18 | 32 | 3A0000-3A7FFF |
| 19 | 32 | 398000-39FFFF |
| 20 | 32 | 390000-397FFF |
| 21 | 32 | 388000-38FFFF |
| 22 | 32 | 380000-387FFF |
| 23 | 32 | 378000-37FFFF |
| 24 | 32 | 370000-377FFF |
| 25 | 32 | 368000-36FFFF |
| 26 | 32 | 360000-367FFF |
| 27 | 32 | 358000-35FFFF |
| 28 | 32 | 350000-357FFF |
| 29 | 32 | 348000-34FFFF |
| 30 | 32 | 340000-347FFF |
| 31 | 32 | 338000-33FFFF |

**Table 23. Top boot block addresses, M28W640HCT (continued)**

| # | Size (Kword) | Address range |
|---|---|---|
| 32 | 32 | 330000-337FFF |
| 33 | 32 | 328000-32FFFF |
| 34 | 32 | 320000-327FFF |
| 35 | 32 | 318000-31FFFF |
| 36 | 32 | 310000-317FFF |
| 37 | 32 | 308000-30FFFF |
| 38 | 32 | 300000-307FFF |
| 39 | 32 | 2F8000-2FFFFF |
| 40 | 32 | 2F0000-2F7FFF |
| 41 | 32 | 2E8000-2EFFFF |
| 42 | 32 | 2E0000-2E7FFF |
| 43 | 32 | 2D8000-2DFFFF |
| 44 | 32 | 2D0000-2D7FFF |
| 45 | 32 | 2C8000-2CFFFF |
| 46 | 32 | 2C0000-2C7FFF |
| 47 | 32 | 2B8000-2BFFFF |
| 48 | 32 | 2B0000-2B7FFF |
| 49 | 32 | 2A8000-2AFFFF |
| 50 | 32 | 2A0000-2A7FFF |
| 51 | 32 | 298000-29FFFF |
| 52 | 32 | 290000-297FFF |
| 53 | 32 | 288000-28FFFF |
| 54 | 32 | 280000-287FFF |
| 55 | 32 | 278000-27FFFF |
| 56 | 32 | 270000-277FFF |
| 57 | 32 | 268000-26FFFF |
| 58 | 32 | 260000-267FFF |
| 59 | 32 | 258000-25FFFF |
| 60 | 32 | 250000-257FFF |
| 61 | 32 | 248000-24FFFF |
| 62 | 32 | 240000-247FFF |
| 63 | 32 | 238000-23FFFF |
| 64 | 32 | 230000-237FFF |
| 65 | 32 | 228000-22FFFF |
| 66 | 32 | 220000-227FFF |

**Table 23. Top boot block addresses, M28W640HCT (continued)**

| # | Size (Kword) | Address range |
|---|---|---|
| 67 | 32 | 218000-21FFFF |
| 68 | 32 | 210000-217FFF |
| 69 | 32 | 208000-20FFFF |
| 70 | 32 | 200000-207FFF |
| 71 | 32 | 1F8000-1FFFFF |
| 72 | 32 | 1F0000-1F7FFF |
| 73 | 32 | 1E8000-1EFFFF |
| 74 | 32 | 1E0000-1E7FFF |
| 75 | 32 | 1D8000-1DFFFF |
| 76 | 32 | 1D0000-1D7FFF |
| 77 | 32 | 1C8000-1CFFFF |
| 78 | 32 | 1C0000-1C7FFF |
| 79 | 32 | 1B8000-1BFFFF |
| 80 | 32 | 1B0000-1B7FFF |
| 81 | 32 | 1A8000-1AFFFF |
| 82 | 32 | 1A0000-1A7FFF |
| 83 | 32 | 198000-19FFFF |
| 84 | 32 | 190000-197FFF |
| 85 | 32 | 188000-18FFFF |
| 86 | 32 | 180000-187FFF |
| 87 | 32 | 178000-17FFFF |
| 88 | 32 | 170000-177FFF |
| 89 | 32 | 168000-16FFFF |
| 90 | 32 | 160000-167FFF |
| 91 | 32 | 158000-15FFFF |
| 92 | 32 | 150000-157FFF |
| 93 | 32 | 148000-14FFFF |
| 94 | 32 | 140000-147FFF |
| 95 | 32 | 138000-13FFFF |
| 96 | 32 | 130000-137FFF |
| 97 | 32 | 128000-12FFFF |
| 98 | 32 | 120000-127FFF |
| 99 | 32 | 118000-11FFFF |
| 100 | 32 | 110000-117FFF |
| 101 | 32 | 108000-10FFFF |

**Table 23. Top boot block addresses, M28W640HCT (continued)**

| # | Size (Kword) | Address range |
|---|---|---|
| 102 | 32 | 100000-107FFF |
| 103 | 32 | 0F8000-0FFFFF |
| 104 | 32 | 0F0000-0F7FFF |
| 105 | 32 | 0E8000-0EFFFF |
| 106 | 32 | 0E0000-0E7FFF |
| 107 | 32 | 0D8000-0DFFFF |
| 108 | 32 | 0D0000-0D7FFF |
| 109 | 32 | 0C8000-0CFFFF |
| 110 | 32 | 0C0000-0C7FFF |
| 111 | 32 | 0B8000-0BFFFF |
| 112 | 32 | 0B0000-0B7FFF |
| 113 | 32 | 0A8000-0AFFFF |
| 114 | 32 | 0A0000-0A7FFF |
| 115 | 32 | 098000-09FFFF |
| 116 | 32 | 090000-097FFF |
| 117 | 32 | 088000-08FFFF |
| 118 | 32 | 080000-087FFF |
| 119 | 32 | 078000-07FFFF |
| 120 | 32 | 070000-077FFF |
| 121 | 32 | 068000-06FFFF |
| 122 | 32 | 060000-067FFF |
| 123 | 32 | 058000-05FFFF |
| 124 | 32 | 050000-057FFF |
| 125 | 32 | 048000-04FFFF |
| 126 | 32 | 040000-047FFF |
| 127 | 32 | 038000-03FFFF |
| 128 | 32 | 030000-037FFF |
| 129 | 32 | 028000-02FFFF |
| 130 | 32 | 020000-027FFF |
| 131 | 32 | 018000-01FFFF |
| 132 | 32 | 010000-017FFF |
| 133 | 32 | 008000-00FFFF |
| 134 | 32 | 000000-007FFF |

**Table 24. Bottom boot block addresses, M28W640HCB**

| # | Size (Kword) | Address range |
|---|---|---|
| 134 | 32 | 3F8000-3FFFFF |
| 133 | 32 | 3F0000-3F7FFF |
| 132 | 32 | 3E8000-3EFFFF |
| 131 | 32 | 3E0000-3E7FFF |
| 130 | 32 | 3D8000-3DFFFF |
| 129 | 32 | 3D0000-3D7FFF |
| 128 | 32 | 3C8000-3CFFFF |
| 127 | 32 | 3C0000-3C7FFF |
| 126 | 32 | 3B8000-3BFFFF |
| 125 | 32 | 3B0000-3B7FFF |
| 124 | 32 | 3A8000-3AFFFF |
| 123 | 32 | 3A0000-3A7FFF |
| 122 | 32 | 398000-39FFFF |
| 121 | 32 | 390000-397FFF |
| 120 | 32 | 388000-38FFFF |
| 119 | 32 | 380000-387FFF |
| 118 | 32 | 378000-37FFFF |
| 117 | 32 | 370000-377FFF |
| 116 | 32 | 368000-36FFFF |
| 115 | 32 | 360000-367FFF |
| 114 | 32 | 358000-35FFFF |
| 113 | 32 | 350000-357FFF |
| 112 | 32 | 348000-34FFFF |
| 111 | 32 | 340000-347FFF |
| 110 | 32 | 338000-33FFFF |
| 109 | 32 | 330000-337FFF |
| 108 | 32 | 328000-32FFFF |
| 107 | 32 | 320000-327FFF |
| 106 | 32 | 318000-31FFFF |
| 105 | 32 | 310000-317FFF |
| 104 | 32 | 308000-30FFFF |
| 103 | 32 | 300000-307FFF |
| 102 | 32 | 2F8000-2FFFFF |
| 101 | 32 | 2F0000-2F7FFF |
| 100 | 32 | 2E8000-2EFFFF |

**Table 24. Bottom boot block addresses, M28W640HCB (continued)**

| # | Size (Kword) | Address range |
|---|---|---|
| 99 | 32 | 2E0000-2E7FFF |
| 98 | 32 | 2D8000-2DFFFF |
| 97 | 32 | 2D0000-2D7FFF |
| 96 | 32 | 2C8000-2CFFFF |
| 95 | 32 | 2C0000-2C7FFF |
| 94 | 32 | 2B8000-2BFFFF |
| 93 | 32 | 2B0000-2B7FFF |
| 92 | 32 | 2A8000-2AFFFF |
| 91 | 32 | 2A0000-2A7FFF |
| 90 | 32 | 298000-29FFFF |
| 89 | 32 | 290000-297FFF |
| 88 | 32 | 288000-28FFFF |
| 87 | 32 | 280000-287FFF |
| 86 | 32 | 278000-27FFFF |
| 85 | 32 | 270000-277FFF |
| 84 | 32 | 268000-26FFFF |
| 83 | 32 | 260000-267FFF |
| 82 | 32 | 258000-25FFFF |
| 81 | 32 | 250000-257FFF |
| 80 | 32 | 248000-24FFFF |
| 79 | 32 | 240000-247FFF |
| 78 | 32 | 238000-23FFFF |
| 77 | 32 | 230000-237FFF |
| 76 | 32 | 228000-22FFFF |
| 75 | 32 | 220000-227FFF |
| 74 | 32 | 218000-21FFFF |
| 73 | 32 | 210000-217FFF |
| 72 | 32 | 208000-20FFFF |
| 71 | 32 | 200000-207FFF |
| 70 | 32 | 1F8000-1FFFFF |
| 69 | 32 | 1F0000-1F7FFF |
| 68 | 32 | 1E8000-1EFFFF |
| 67 | 32 | 1E0000-1E7FFF |
| 66 | 32 | 1D8000-1DFFFF |
| 65 | 32 | 1D0000-1D7FFF |

**Table 24. Bottom boot block addresses, M28W640HCB (continued)**

| # | Size (Kword) | Address range |
|---|---|---|
| 64 | 32 | 1C8000-1CFFFF |
| 63 | 32 | 1C0000-1C7FFF |
| 62 | 32 | 1B8000-1BFFFF |
| 61 | 32 | 1B0000-1B7FFF |
| 60 | 32 | 1A8000-1AFFFF |
| 59 | 32 | 1A0000-1A7FFF |
| 58 | 32 | 198000-19FFFF |
| 57 | 32 | 190000-197FFF |
| 56 | 32 | 188000-18FFFF |
| 55 | 32 | 180000-187FFF |
| 54 | 32 | 178000-17FFFF |
| 53 | 32 | 170000-177FFF |
| 52 | 32 | 168000-16FFFF |
| 51 | 32 | 160000-167FFF |
| 50 | 32 | 158000-15FFFF |
| 49 | 32 | 150000-157FFF |
| 48 | 32 | 148000-14FFFF |
| 47 | 32 | 140000-147FFF |
| 46 | 32 | 138000-13FFFF |
| 45 | 32 | 130000-137FFF |
| 44 | 32 | 128000-12FFFF |
| 43 | 32 | 120000-127FFF |
| 42 | 32 | 118000-11FFFF |
| 41 | 32 | 110000-117FFF |
| 40 | 32 | 108000-10FFFF |
| 39 | 32 | 100000-107FFF |
| 38 | 32 | 0F8000-0FFFFF |
| 37 | 32 | 0F0000-0F7FFF |
| 36 | 32 | 0E8000-0EFFFF |
| 35 | 32 | 0E0000-0E7FFF |
| 34 | 32 | 0D8000-0DFFFF |
| 33 | 32 | 0D0000-0D7FFF |
| 32 | 32 | 0C8000-0CFFFF |
| 31 | 32 | 0C0000-0C7FFF |
| 30 | 32 | 0B8000-0BFFFF |

**Table 24. Bottom boot block addresses, M28W640HCB (continued)**

| # | Size (Kword) | Address range |
|---|---|---|
| 29 | 32 | 0B0000-0B7FFF |
| 28 | 32 | 0A8000-0AFFFF |
| 27 | 32 | 0A0000-0A7FFF |
| 26 | 32 | 098000-09FFFF |
| 25 | 32 | 090000-097FFF |
| 24 | 32 | 088000-08FFFF |
| 23 | 32 | 080000-087FFF |
| 22 | 32 | 078000-07FFFF |
| 21 | 32 | 070000-077FFF |
| 20 | 32 | 068000-06FFFF |
| 19 | 32 | 060000-067FFF |
| 18 | 32 | 058000-05FFFF |
| 17 | 32 | 050000-057FFF |
| 16 | 32 | 048000-04FFFF |
| 15 | 32 | 040000-047FFF |
| 14 | 32 | 038000-03FFFF |
| 13 | 32 | 030000-037FFF |
| 12 | 32 | 028000-02FFFF |
| 11 | 32 | 020000-027FFF |
| 10 | 32 | 018000-01FFFF |
| 9 | 32 | 010000-017FFF |
| 8 | 32 | 008000-00FFFF |
| 7 | 4 | 007000-007FFF |
| 6 | 4 | 006000-006FFF |
| 5 | 4 | 005000-005FFF |
| 4 | 4 | 004000-004FFF |
| 3 | 4 | 003000-003FFF |
| 2 | 4 | 002000-002FFF |
| 1 | 4 | 001000-001FFF |
| 0 | 4 | 000000-000FFF |

# Appendix B Common Flash interface (CFI)

The Common Flash interface is a JEDEC approved, standardized data structure that can be read from the Flash memory device. It allows a system software to query the device to determine various electrical and timing parameters, density information and functions supported by the memory. The system can interface easily with the device, enabling the software to upgrade itself when necessary.

When the CFI Query command (RCFI) is issued the device enters CFI Query mode and the data structure is read from the memory. Tables *25*, *26*, *27*, *28*, *29* and *30* show the addresses used to retrieve the data.

The CFI data structure also contains a security area where a 64 bit unique security number is written (see *Table 30: Security code area*). This area can be accessed only in Read mode by the final user. It is impossible to change the security number after it has been written by Numonyx. Issue a Read command to return to Read mode.

**Table 25. Query structure overview(1)**

| Offset | Sub-section name | Description |
|---|---|---|
| 00h | Reserved | Reserved for algorithm-specific information |
| 10h | CFI Query identification string | Command set ID and algorithm data offset |
| 1Bh | System interface information | Device timing & voltage information |
| 27h | Device geometry definition | Flash device layout |
| P | Primary algorithm-specific extended query table | Additional information specific to the primary algorithm (optional) |
| A | Alternate algorithm-specific extended query table | Additional information specific to the alternate algorithm (optional) |

1. Query data are always presented on the lowest order data outputs.

**Table 26. CFI query identification string(1)**

| Offset | Data | Description | Value |
|---|---|---|---|
| 00h | 0020h | Manufacturer code | Numonyx |
| 01h | 8848h<br>8849h | Device code | Top<br>Bottom |
| 02h-0Fh | reserved | Reserved | |
| 10h<br>11h<br>12h | 0051h<br>0052h<br>0059h | Query unique ASCII string ‘QRY’ | ‘Q’<br>‘R’<br>‘Y’ |
| 13h<br>14h | 0003h<br>0000h | Primary algorithm command set and control interface ID code 16 bit ID code defining a specific algorithm | Intel compatible |
| 15h<br>16h | 0035h<br>0000h | Address for primary algorithm extended query table (see *Table 28*) | P = 35h |
| 17h<br>18h | 0000h<br>0000h | Alternate vendor command set and control interface ID code second vendor - specified algorithm supported (0000h means none exists) | NA |
| 19h<br>1Ah | 0000h<br>0000h | Address for Alternate algorithm extended query table (0000h means none exists) | NA |

1. Query data are always presented on the lowest order data outputs (DQ7-DQ0) only. DQ8-DQ15 are ‘0’.

**Table 27. CFI query system interface information**

| Offset | Data | Description | Value |
|---|---|---|---|
| 1Bh | 0027h | $V_{DD}$ logic supply minimum program/erase or write voltage<br>bit 7 to 4BCD value in volts<br>bit 3 to 0BCD value in 100 mV | 2.7 V |
| 1Ch | 0036h | $V_{DD}$ logic supply maximum Program/Erase or Write voltage<br>bit 7 to 4BCD value in volts<br>bit 3 to 0BCD value in 100 mV | 3.6 V |
| 1Dh | 00B4h | $V_{PP}$ [programming] supply minimum Program/Erase voltage<br>bit 7 to 4HEX value in volts<br>bit 3 to 0BCD value in 100 mV | 11.4 V |
| 1Eh | 00C6h | $V_{PP}$ [programming] supply maximum Program/Erase voltage<br>bit 7 to 4HEX value in volts<br>bit 3 to 0BCD value in 100 mV | 12.6 V |
| 1Fh | 0004h | Typical time-out per single word program = $2^n$ µs | 16 µs |
| 20h | 0004h | Typical time-out for Double/Quadruple Word Program = $2^n$ µs | 16 µs |
| 21h | 000Ah | Typical time-out per individual block erase = $2^n$ ms | 1 s |
| 22h | 0000h | Typical time-out for full chip erase = $2^n$ ms | NA |
| 23h | 0005h | Maximum time-out for Word program = $2^n$ times typical | 512 µs |
| 24h | 0005h | Maximum time-out for Double/Quadruple Word Program = $2^n$ times typical | 512 µs |
| 25h | 0003h | Maximum time-out per individual block erase = $2^n$ times typical | 8 s |
| 26h | 0000h | Maximum time-out for chip erase = $2^n$ times typical | NA |

**Table 28. Device geometry definition**

| | Offset Word Mode | Data | Description | Value |
|---|---|---|---|---|
| | 27h | 0017h | Device size = $2^n$ in number of bytes | 8 Mbyte |
| | 28h<br>29h | 0001h<br>0000h | Flash device interface code description | x 16<br>Async. |
| | 2Ah<br>2Bh | 0003h<br>0000h | Maximum number of bytes in multi-byte program or page = $2^n$ | 8 |
| | 2Ch | 0002h | Number of Erase block regions within the device.<br>It specifies the number of regions within the device containing contiguous Erase blocks of the same size. | 2 |
| M28W640HCT | 2Dh<br>2Eh | 007Eh<br>0000h | Region 1 information<br>Number of identical-size erase block = 007Eh+1 | 127 |
| M28W640HCT | 2Fh<br>30h | 0000h<br>0001h | Region 1 information<br>Block size in Region 1 = 0100h * 256 byte | 64 Kbyte |
| M28W640HCT | 31h<br>32h | 0007h<br>0000h | Region 2 information<br>Number of identical-size erase block = 0007h+1 | 8 |
| M28W640HCT | 33h<br>34h | 0020h<br>0000h | Region 2 information<br>Block size in region 2 = 0020h * 256 byte | 8 Kbyte |
| M28W640HCB | 2Dh<br>2Eh | 0007h<br>0000h | Region 1 information<br>Number of identical-size erase block = 0007h+1 | 8 |
| M28W640HCB | 2Fh<br>30h | 0020h<br>0000h | Region 1 information<br>Block size in region 1 = 0020h * 256 byte | 8 Kbyte |
| M28W640HCB | 31h<br>32h | 007Eh<br>0000h | Region 2 information<br>Number of identical-size erase block = 007Eh=1 | 127 |
| M28W640HCB | 33h<br>34h | 0000h<br>0001h | Region 2 information<br>Block size in region 2 = 0100h * 256 byte | 64 Kbyte |

**Table 29. Primary algorithm-specific extended query table**

| Offset P = 35h(1) | Data | Description | Value |
|---|---|---|---|
| (P+0)h = 35h<br>(P+1)h = 36h<br>(P+2)h = 37h | 0050h<br>0052h<br>0049h | Primary algorithm extended query table unique ASCII string ‘PRI’ | ‘P’<br>‘R’<br>‘I’ |
| (P+3)h = 38h | 0031h | Major version number, ASCII | ‘1’ |
| (P+4)h = 39h | 0030h | Minor version number, ASCII | ‘0’ |
| (P+5)h = 3Ah<br>(P+6)h = 3Bh<br>(P+7)h = 3Ch<br><br><br><br><br>(P+8)h = 3Dh | 0066h<br>0000h<br>0000h<br><br><br><br><br>0000h | Extended query table contents for primary algorithm. address (P+5)h contains less significant byte.<br>bit 0Chip Erase supported(1 = Yes, 0 = No)<br>bit 1Suspend Erase supported(1 = Yes, 0 = No)<br>bit 2Suspend Program supported(1 = Yes, 0 = No)<br>bit 3Legacy Lock/Unlock supported(1 = Yes, 0 = No)<br>bit 4Queued Erase supported(1 = Yes, 0 = No)<br>bit 5Instant individual block locking supported(1 = Yes, 0 = No)<br>bit 6Protection bits supported(1 = Yes, 0 = No)<br>bit 7Page mode read supported(1 = Yes, 0 = No)<br>bit 8Synchronous read supported(1 = Yes, 0 = No)<br>bit 31 to 9Reserved; undefined bits are ‘0’ | <br><br>No<br>Yes<br>Yes<br>No<br>No<br>Yes<br>Yes<br>No<br>No |
| (P+9)h = 3Eh | 0001h | Supported functions after Suspend<br>Read Array, Read Status Register and CFI query are always supported during erase or program operation<br>bit 0Program supported after Erase Suspend (1 = Yes, 0 = No)<br>bit 7 to 1Reserved; undefined bits are ‘0’ | <br><br><br>Yes |
| (P+A)h = 3Fh<br><br><br>(P+B)h = 40h | 0003h<br><br><br>0000h | Block Lock status<br>Defines which bits in the block Status Register section of the query are implemented.<br>Address (P+A)h contains less significant byte<br>bit 0Block Lock Status Register Lock/Unlock bit active(1 = Yes, 0 = No)<br>bit 1Block Lock Status Register Lock-down bit active (1 = Yes, 0 = No)<br>bit 15 to 2Reserved for future use; undefined bits are ‘0’ | <br><br><br>Yes<br>Yes |
| (P+C)h = 41h | 0030h | $V_{DD}$ logic supply optimum Program/Erase voltage (highest performance)<br>bit 7 to 4HEX value in volts<br>bit 3 to 0BCD value in 100 mV | 3 V |
| (P+D)h = 42h | 00C0h | $V_{PP}$ supply optimum Program/Erase voltage<br>bit 7 to 4HEX value in volts<br>bit 3 to 0BCD value in 100 mV | 12 V |
| (P+E)h = 43h | 0001h | Number of protection register fields in JEDEC ID space.<br>"00h," indicates that 256 protection bytes are available | 01 |

**Table 29. Primary algorithm-specific extended query table (continued)**

| Offset P = 35h(1) | Data | Description | Value |
|---|---|---|---|
| (P+F)h = 44h | 0080h | Protection field 1: protection description | 80h |
| (P+10)h = 45h | 0000h | This field describes user-available one-time-programmable (OTP) protection register bytes. Some are pre-programmed with device unique serial numbers. Others are user programmable. bits 0–15 point to the Protection Register Lock byte, the section's first byte. | 00h |
| (P+11)h = 46h | 0003h | | 8 byte |
| (P+12)h = 47h | 0004h | The following bytes are factory pre-programmed and user-programmable.<br>bit 0 to 7 Lock/bytes JEDEC-plane physical low address<br>bit 8 to 15Lock/bytes JEDEC-plane physical high address<br>bit 16 to 23 "n" such that $2^n$ = factory pre-programmed bytes<br>bit 24 to 31 "n" such that $2^n$ = user programmable bytes | 16 byte |
| (P+13)h = 48h | | Reserved | |

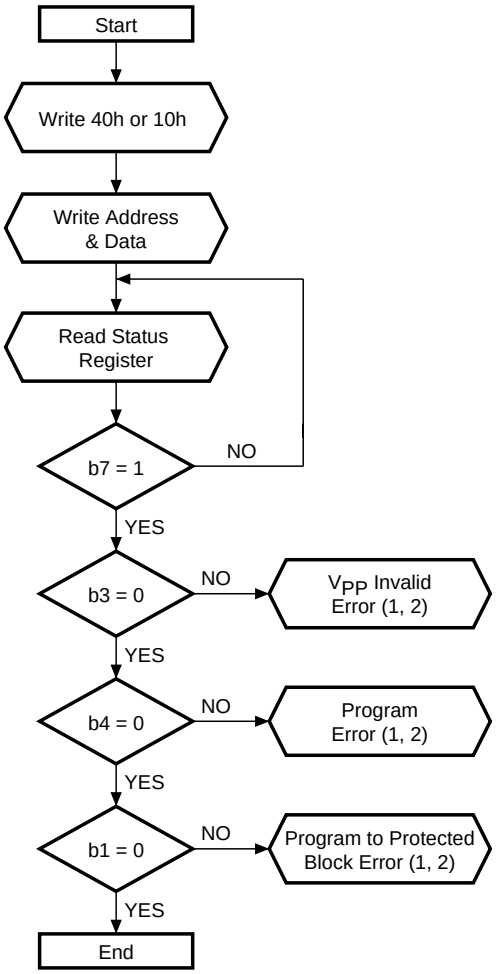
1. See *Table 26*, offset 15 for P pointer definition.

**Table 30. Security code area**

| Offset | Data | Description |
|---|---|---|
| 80h | 00XX | Protection Register Lock |
| 81h | XXXX | 64 bits: unique device number |
| 82h | XXXX | |
| 83h | XXXX | |
| 84h | XXXX | |
| 85h | XXXX | 128 bits: user programmable OTP |
| 86h | XXXX | |
| 87h | XXXX | |
| 88h | XXXX | |
| 89h | XXXX | |
| 8Ah | XXXX | |
| 8Bh | XXXX | |
| 8Ch | XXXX | |

# Appendix C Flowcharts and pseudocodes

**Figure 15. Program flowchart and pseudocode**

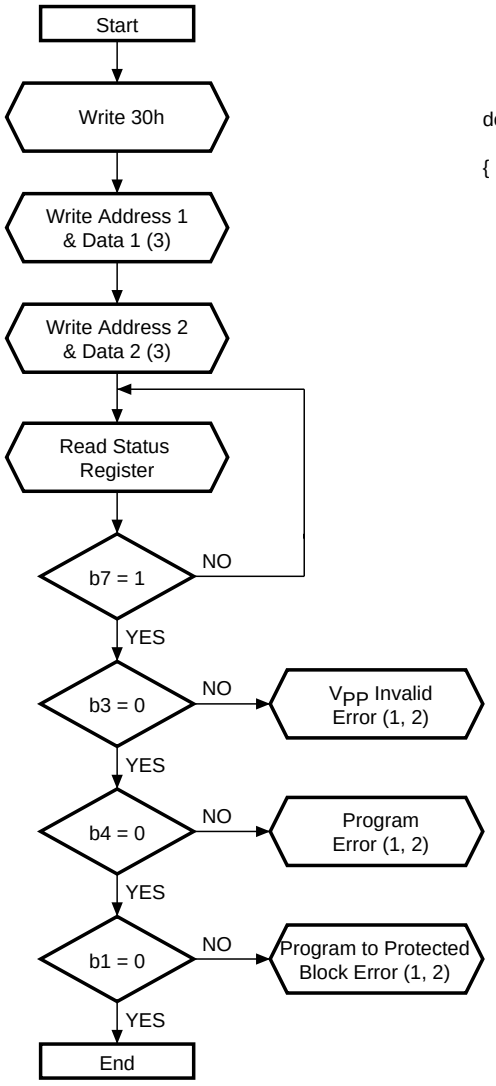



```
program_command (addressToProgram, dataToProgram) {:
        writeToFlash (any_address, 0x40) ;
        /*or writeToFlash (any_address, 0x10) ; */

        writeToFlash (addressToProgram, dataToProgram) ;
        /*Memory enters read status state after
        the Program Command*/

      do {
          status_register=readFlash (any_address) ;
          /* Ē or Ḡ must be toggled*/

        }  while (status_register.b7== 0) ;

if  (status_register.b3==1) /*VPP invalid error */
      error_handler ( ) ;

if  (status_register.b4==1) /*program error */
      error_handler ( ) ;

if  (status_register.b1==1) /*program to protect block error */
      error_handler ( ) ;

}
```

AI03538b

1. Status check of b1 (protected block), b3 ($V_{PP}$ invalid) and b4 (program error) can be made after each program operation or after a sequence.
2. If an error is found, the Status Register must be cleared before further Program/Erase controller operations.

**Figure 16. Double Word Program flowchart and pseudocode**

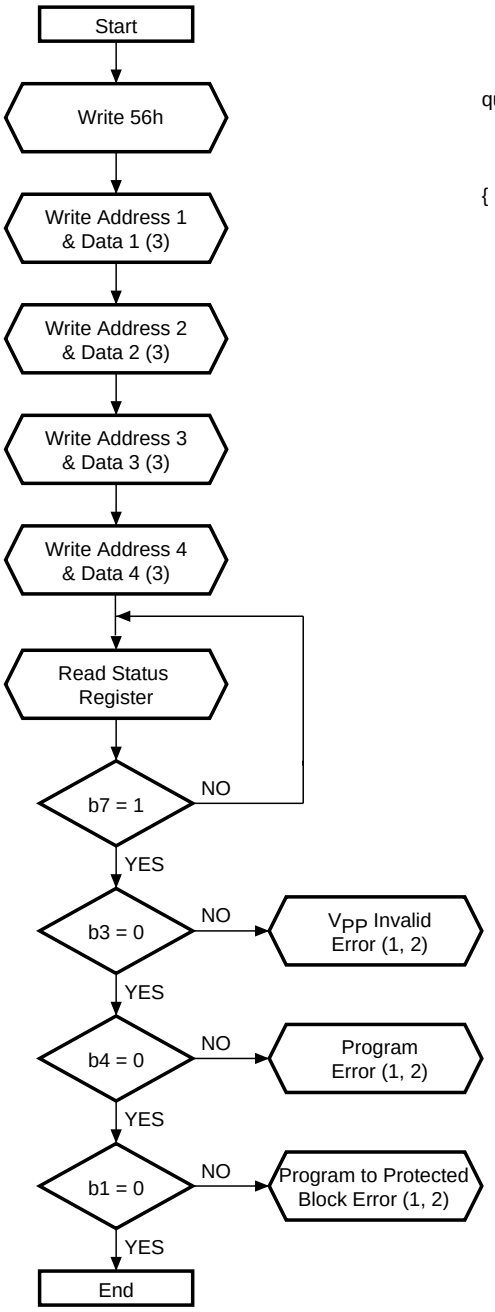



```
double_word_program_command (addressToProgram1, dataToProgram1,
                                        addressToProgram2, dataToProgram2)
{
        writeToFlash (any_address, 0x30) ;
        writeToFlash (addressToProgram1, dataToProgram1) ;
                                        /*see note (3) */
        writeToFlash (addressToProgram2, dataToProgram2) ;
                                        /*see note (3) */
         /*Memory enters read status state after
        the Program command*/

      do {
        status_register=readFlash (any_address) ;
        /* Ē or Ḡ must be toggled*/

        } while (status_register.b7== 0) ;

if  (status_register.b3==1) /*VPP invalid error */
      error_handler ( ) ;

if  (status_register.b4==1) /*program error */
      error_handler ( ) ;

if  (status_register.b1==1) /*program to protect block error */
      error_handler ( ) ;

}
```

AI03539b

1. Status check of b1 (protected block), b3 ($V_{PP}$ invalid) and b4 (program error) can be made after each program operation or after a sequence.
2. If an error is found, the Status Register must be cleared before further Program/Erase operations.
3. Address 1 and address 2 must be consecutive addresses differing only for bit A0.

**Figure 17. Quadruple Word Program flowchart and pseudocode**

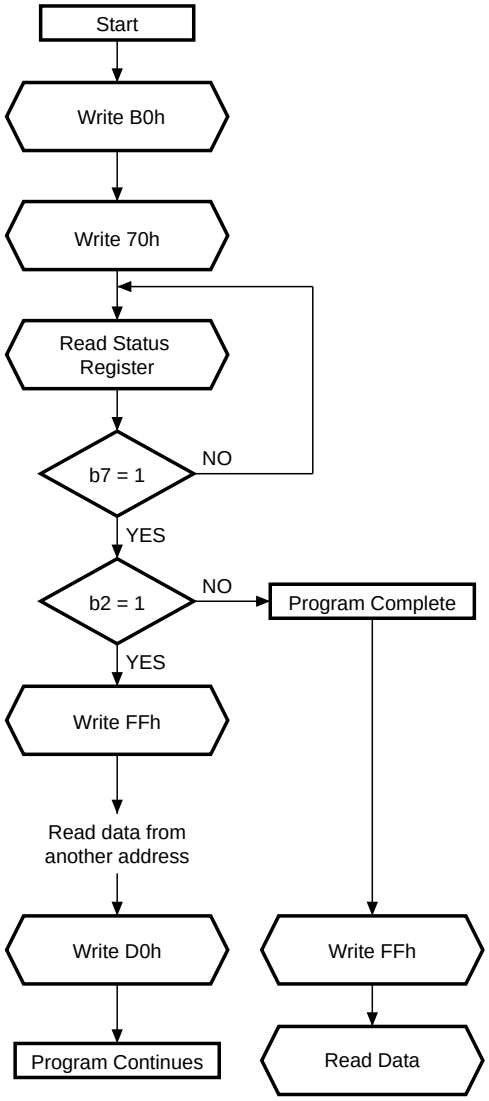



```
quadruple_word_program_command (addressToProgram1, dataToProgram1,
                                addressToProgram2, dataToProgram2,
                                addressToProgram3, dataToProgram3,
                                addressToProgram4, dataToProgram4)
{
        writeToFlash (any_address, 0x56) ;

        writeToFlash (addressToProgram1, dataToProgram1) ;
                                /*see note (3) */

        writeToFlash (addressToProgram2, dataToProgram2) ;
                                /*see note (3) */

        writeToFlash (addressToProgram3, dataToProgram3) ;
                                /*see note (3) */

        writeToFlash (addressToProgram4, dataToProgram4) ;
                                /*see note (3) */

         /*Memory enters read status state after
        the Program command*/

      do {
        status_register=readFlash (any_address) ;
        /* Ē or Ḡ must be toggled*/

        } while (status_register.b7== 0) ;

if (status_register.b3==1) /*VPP invalid error */
    error_handler ( ) ;

if (status_register.b4==1) /*program error */
    error_handler ( ) ;

if (status_register.b1==1) /*program to protect block error */
    error_handler ( ) ;

}
```

AI06233

1. Status check of b1 (protected block), b3 ($V_{PP}$ invalid) and b4 (program error) can be made after each program operation or after a sequence.
2. If an error is found, the Status Register must be cleared before further Program/Erase operations.
3. Address 1 to address 4 must be consecutive addresses differing only for bits A0 and A1.

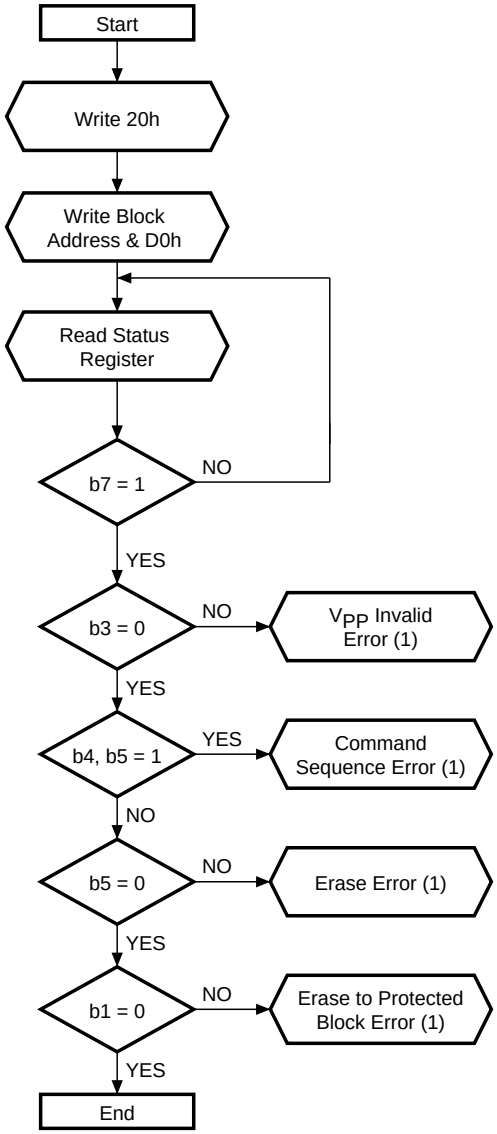
**Figure 18. Program Suspend & Resume flowchart and pseudocode**




```
program_suspend_command   ( )  {
        writeToFlash (any_address, 0xB0) ;

        writeToFlash (any_address, 0x70) ;
        /* read status register to check if
        program has already completed */


   do {
        status_register=readFlash (any_address) ;
        /* E or G must be toggled*/



        }  while (status_register.b7== 0) ;


if  (status_register.b2==0)  /*program completed */
        {  writeToFlash (any_address, 0xFF)  ;
           read_data ( ) ; /*read data from another block*/
           /*The device returns to Read Array
           (as if program/erase suspend was not issued).*/


         }
else
        {  writeToFlash (any_address, 0xFF)  ;
           read_data ( ); /*read data from another address*/
           writeToFlash (any_address, 0xD0) ;
           /*write 0xD0 to resume program*/
         }
}
```

AI03540b

**Figure 19. Erase flowchart and pseudocode**

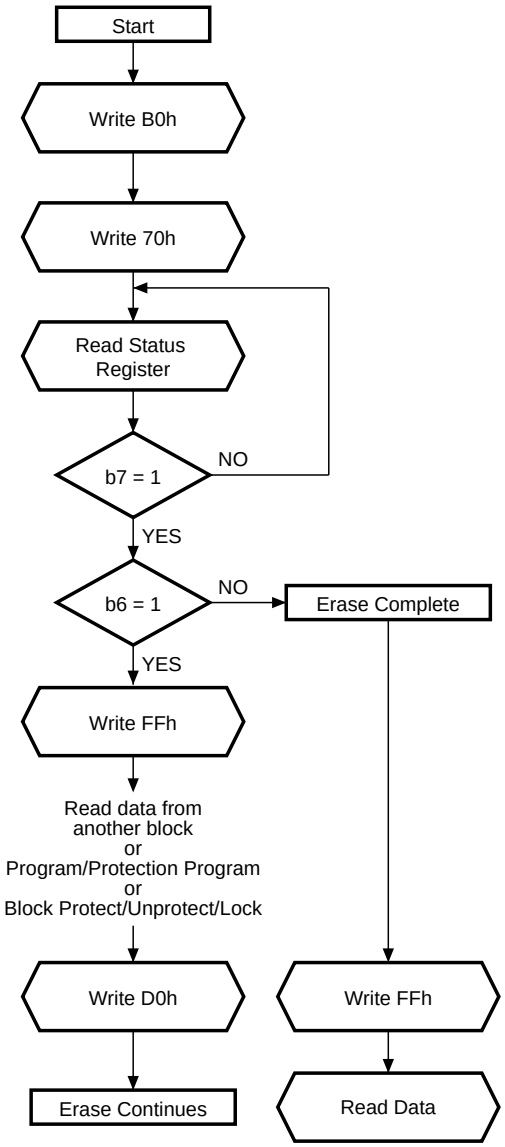



```
erase_command   ( blockToErase )  {
        writeToFlash (any_address, 0x20) ;

        writeToFlash (blockToErase, 0xD0) ;
        /* only A12-A20 are significannt */
        /* Memory enters read status state after
        the Erase Command */


    do {
        status_register=readFlash (any_address) ;
         /* E or G must be toggled*/



        }  while (status_register.b7== 0) ;


if  (status_register.b3==1) /*VPP invalid error */
        error_handler ( ) ;


if  ( (status_register.b4==1) && (status_register.b5==1) )
/* command sequence error */
        error_handler ( ) ;


if  ( (status_register.b5==1) )
/* erase error */
        error_handler ( ) ;


if  (status_register.b1==1) /*program to protect block error */
        error_handler ( ) ;


}
```

AI03541b

1. If an error is found, the Status Register must be cleared before further program/erase operations.

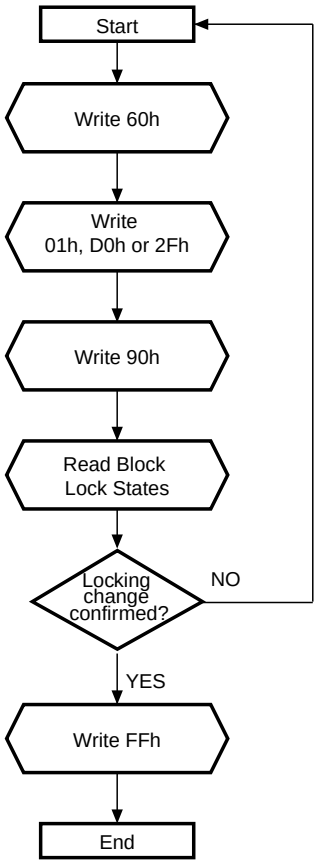
**Figure 20. Erase Suspend & Resume flowchart and pseudocode**




```
erase_suspend_command   ( ) {
        writeToFlash (any_address, 0xB0) ;

        writeToFlash (any_address, 0x70) ;
        /* read status register to check if
        erase has already completed */


    do {
         status_register=readFlash (any_address) ;
         /* E or G must be toggled*/


        }  while (status_register.b7== 0) ;


if  (status_register.b6==0)  /*erase completed */
        {  writeToFlash (any_address, 0xFF)  ;


           read_data ( ) ;
           /*read data from another block*/
           /*The device returns to Read Array
           (as if program/erase suspend was not issued).*/


        }
else
        {  writeToFlash (any_address, 0xFF)  ;
           read_program_data ( );
           /*read or program data from another address*/
           writeToFlash (any_address, 0xD0) ;
           /*write 0xD0 to resume erase*/
        }
}
```

AI03542b

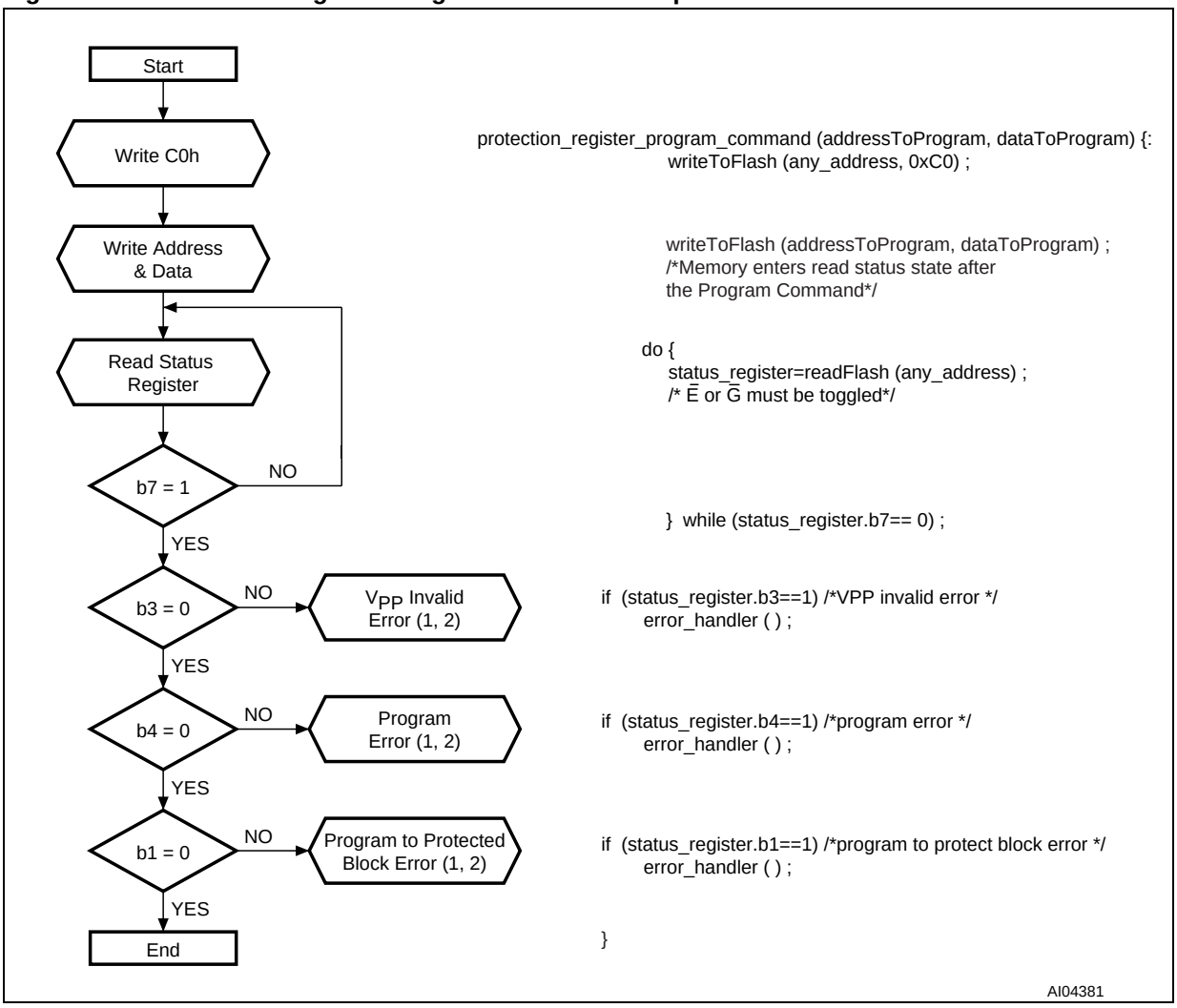
**Figure 21. Locking operations flowchart and pseudocode**




```
locking_operation_command   (address, lock_operation)  {
        writeToFlash (any_address, 0x60) ; /*configuration setup*/

        if  (lock_operation==LOCK)  /*to protect the block*/
                writeToFlash (address, 0x01)  ;
        else if  (lock_operation==UNLOCK)  /*to unprotect the block*/
                writeToFlash (address, 0xD0)  ;
        else if  (lock_operation==LOCK-DOWN)  /*to lock the block*/
                writeToFlash (address, 0x2F)  ;

                writeToFlash (any_address, 0x90)  ;

        if  (readFlash (address) ! = locking_state_expected)
                error_handler () ;
                /*Check the locking state (see Read Block Signature table )*/

        writeToFlash (any_address, 0xFF) ; /*Reset to Read Array mode*/

}
```

AI04364

**Figure 22. Protection Register Program flowchart and pseudocode**




1. Status check of b1 (protected block), b3 ($V_{PP}$ invalid) and b4 (program error) can be made after each program operation or after a sequence.
2. If an error is found, the Status Register must be cleared before further Program/Erase controller operations.

# Appendix D Command interface and Program/Erase controller state

**Table 31. Write state machine current/next(1)**

| Current state | SR bit 7 | Data when Read | Command input (and next state) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Read Array (FFh) | Program Setup (10/40h) | Erase Setup (20h) | Erase Confirm (D0h) | Prog/Ers Suspend (B0h) | Prog/Ers Resume (D0h) | Read Status (70h) | Clear Status (50h) |
| Read Array | ‘1’ | Array | Read Array | Prog.Setup | Ers. Setup | Read Array | | | Read Sts. | Read Array |
| Read Status | ‘1’ | Status | Read Array | Program Setup | Erase Setup | Read Array | | | Read Status | Read Array |
| Read Elect.Sg. | ‘1’ | Electronic signature | Read Array | Program Setup | Erase Setup | Read Array | | | Read Status | Read Array |
| Read CFI Query | ‘1’ | CFI | Read Array | Program Setup | Erase Setup | Read Array | | | Read Status | Read Array |
| Lock Setup | ‘1’ | Status | Lock Command Error | | | Lock (complete) | Lock Cmd Error | Lock (complete) | Lock Command Error | |
| Lock Cmd Error | ‘1’ | Status | Read Array | Program Setup | Erase Setup | Read Array | | | Read Status | Read Array |
| Lock (complete) | ‘1’ | Status | Read Array | Program Setup | Erase Setup | Read Array | | | Read Status | Read Array |
| Prot. Prog. Setup | ‘1’ | Status | Protection Register Program | | | | | | | |
| Prot. Prog. (continue) | ‘0’ | Status | Protection Register Program continue | | | | | | | |
| Prot. Prog. (complete) | ‘1’ | Status | Read Array | Program Setup | Erase Setup | Read Array | | | Read Status | Read Array |
| Prog. Setup | ‘1’ | Status | Program | | | | | | | |
| Program (continue) | ‘0’ | Status | Program (continue) | | | | Prog. Sus Read Sts | Program (continue) | | |
| Prog. Sus Status | ‘1’ | Status | Prog. Sus Read Array | Program Suspend to Read Array | | Program (continue) | Prog. Sus Read Array | Program (continue) | Prog. Sus Read Sts | Prog. Sus Read Array |
| Prog. Sus Read Array | ‘1’ | Array | Prog. Sus Read Array | Program Suspend to Read Array | | Program (continue) | Prog. Sus Read Array | Program (continue) | Prog. Sus Read Sts | Prog. Sus Read Array |
| Prog. Sus Read Elect.Sg. | ‘1’ | Electronic signature | Prog. Sus Read Array | Program Suspend to Read Array | | Program (continue) | Prog. Sus Read Array | Program (continue) | Prog. Sus Read Sts | Prog. Sus Read Array |

**Table 31. Write state machine current/next(1) (continued)**

| Current state | SR bit 7 | Data when Read | Command input (and next state) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Read Array (FFh) | Program Setup (10/40h) | Erase Setup (20h) | Erase Confirm (D0h) | Prog/Ers Suspend (B0h) | Prog/Ers Resume (D0h) | Read Status (70h) | Clear Status (50h) |
| Prog. Sus Read CFI | '1' | CFI | Prog. Sus Read Array | Program Suspend to Read Array | | Program (continue) | Prog. Sus Read Array | Program (continue) | Prog. Sus Read Sts | Prog. Sus Read Array |
| Program (complete) | '1' | Status | Read Array | Program Setup | Erase Setup | Read Array | | | Read Status | Read Array |
| Erase Setup | '1' | Status | Erase Command Error | | | Erase (continue) | Erase CmdError | Erase (continue) | Erase Command Error | |
| Erase Cmd.Error | '1' | Status | Read Array | Program Setup | Erase Setup | Read Array | | | Read Status | Read Array |
| Erase (continue) | '0' | Status | Erase (continue) | | | | Erase Sus Read Sts | Erase (continue) | | |
| Erase Sus Read Sts | '1' | Status | Erase Sus Read Array | Program Setup | Erase Sus Read Array | Erase (continue) | Erase Sus Read Array | Erase (continue) | Erase Sus Read Sts | Erase Sus Read Array |
| Erase Sus Read Array | '1' | Array | Erase Sus Read Array | Program Setup | Erase Sus Read Array | Erase (continue) | Erase Sus Read Array | Erase (continue) | Erase Sus Read Sts | Erase Sus Read Array |
| Erase Sus Read Elect.Sg. | '1' | Electronic Signature | Erase Sus Read Array | Program Setup | Erase Sus Read Array | Erase (continue) | Erase Sus Read Array | Erase (continue) | Erase Sus Read Sts | Erase Sus Read Array |
| Erase Sus Read CFI | '1' | CFI | Erase Sus Read Array | Program Setup | Erase Sus Read Array | Erase (continue) | Erase Sus Read Array | Erase (continue) | Erase Sus Read Sts | Erase Sus Read Array |
| Erase (complete) | '1' | Status | Read Array | Program Setup | Erase Setup | Read Array | | | Read Status | Read Array |

1. Cmd = Command, Elect.Sg. = Electronic Signature, Ers = Erase, Prog. = Program, Prot = Protection, Sus = Suspend.

**Table 32. Write state machine current/next(1)**

| Current state | Command input (and next state) | | | | | | |
|---|---|---|---|---|---|---|---|
| | Read Elect.Sg. (90h) | Read CFI Query (98h) | Lock Setup (60h) | Prot. Prog. Setup (C0h) | Lock Confirm (01h) | Lock Down Confirm (2Fh) | Unlock Confirm (D0h) |
| Read Array | Read Elect.Sg. | Read CFI Query | Lock Setup | Prot. Prog. Setup | Read Array | | |
| Read Status | Read Elect.Sg. | Read CFI Query | Lock Setup | Prot. Prog. Setup | Read Array | | |
| Read Elect.Sg. | Read Elect.Sg. | Read CFI Query | Lock Setup | Prot. Prog. Setup | Read Array | | |
| Read CFI Query | Read Elect.Sg. | Read CFI Query | Lock Setup | Prot. Prog. Setup | Read Array | | |
| Lock Setup | Lock Command Error | | | | Lock (complete) | | |
| Lock Cmd Error | Read Elect.Sg. | Read CFI Query | Lock Setup | Prot. Prog. Setup | Read Array | | |
| Lock (complete) | Read Elect.Sg. | Read CFI Query | Lock Setup | Prot. Prog. Setup | Read Array | | |
| Prot. Prog. Setup | Protection Register Program | | | | | | |
| Prot. Prog. (continue) | Protection Register Program (continue) | | | | | | |
| Prot. Prog. (complete) | Read Elect.Sg. | Read CFI Query | Lock Setup | Prot. Prog. Setup | Read Array | | |
| Prog. Setup | Program | | | | | | |
| Program (continue) | Program (continue) | | | | | | |
| Prog. Suspend Read Status | Prog. Suspend Read Elect.Sg. | Prog. Suspend Read CFI Query | Program Suspend Read Array | | | | Program (continue) |
| Prog. Suspend Read Array | Prog. Suspend Read Elect.Sg. | Prog. Suspend Read CFI Query | Program Suspend Read Array | | | | Program (continue) |
| Prog. Suspend Read Elect.Sg. | Prog. Suspend Read Elect.Sg. | Prog. Suspend Read CFI Query | Program Suspend Read Array | | | | Program (continue) |
| Prog. Suspend Read CFI | Prog. Suspend Read Elect.Sg. | Prog. Suspend Read CFI Query | Program Suspend Read Array | | | | Program (continue) |
| Program (complete) | Read Elect.Sg. | Read CFIQuery | Lock Setup | Prot. Prog. Setup | Read Array | | |

**Table 32. Write state machine current/next(1) (continued)**

| Current state | Command input (and next state) | | | | | | |
|---|---|---|---|---|---|---|---|
| | **Read Elect.Sg. (90h)** | **Read CFI Query (98h)** | **Lock Setup (60h)** | **Prot. Prog. Setup (C0h)** | **Lock Confirm (01h)** | **Lock Down Confirm (2Fh)** | **Unlock Confirm (D0h)** |
| Erase Setup | Erase Command Error | | | | | | Erase (continue) |
| Erase Cmd.Error | Read Elect.Sg. | Read CFI Query | Lock Setup | Prot. Prog. Setup | Read Array | | |
| Erase (continue) | Erase (continue) | | | | | | |
| Erase Suspend Read Ststus | Erase Suspend Read Elect.Sg. | Erase Suspend Read CFI Query | Lock Setup | Erase Suspend Read Array | | | Erase (continue) |
| Erase Suspend Read Array | Erase Suspend Read Elect.Sg. | Erase Suspend Read CFI Query | Lock Setup | Erase Suspend Read Array | | | Erase (continue) |
| Erase Suspend Read Elect.Sg. | Erase Suspend Read Elect.Sg. | Erase Suspend Read CFI Query | Lock Setup | Erase Suspend Read Array | | | Erase (continue) |
| Erase Suspend Read CFI Query | Erase Suspend Read Elect.Sg. | Erase Suspend Read CFI Query | Lock Setup | Erase Suspend Read Array | | | Erase (continue) |
| Erase (complete) | Read Elect.Sg. | Read CFI Query | Lock Setup | Prot. Prog. Setup | Read Array | | |

1. Cmd = Command, Elect.Sg. = Electronic Signature, Prog. = Program, Prot = Protection.

# 11 Revision history

**Table 33. Document revision history**

| Date | Version | Changes |
|---|---|---|
| 29-Jan-2008 | 1 | Initial release. |
| 20-Mar-2008 | 2 | Applied Numonyx branding. |
| 06-Nov-2008 | 3 | Changed title page to remove “preliminary” status.<br>Corrected minimum voltage for $V_{IH}$ in *Table 15.: DC characteristics* |

**Please Read Carefully:**

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH NUMONYX™ PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN NUMONYX'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, NUMONYX ASSUMES NO LIABILITY WHATSOEVER, AND NUMONYX DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF NUMONYX PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Numonyx products are not intended for use in medical, life saving, life sustaining, critical control or safety systems, or in nuclear facility applications.

Numonyx may make changes to specifications and product descriptions at any time, without notice.

Numonyx, B.V. may have patents or pending patent applications, trademarks, copyrights, or other intellectual property rights that relate to the presented subject matter. The furnishing of documents and other materials and information does not provide any license, express or implied, by estoppel or otherwise, to any such patents, trademarks, copyrights, or other intellectual property rights.

Designers must not rely on the absence or characteristics of any features or instructions marked “reserved” or “undefined.” Numonyx reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

Contact your local Numonyx sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Numonyx literature may be obtained by visiting Numonyx's website at http://www.numonyx.com.

Numonyx StrataFlash is a trademark or registered trademark of Numonyx or its subsidiaries in the United States and other countries.

*Other names and brands may be claimed as the property of others.

Copyright © 11/5/7, Numonyx, B.V., All Rights Reserved.

## Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

- View M28W640HCB70N6F on WIN SOURCE
- Micron Technology Inc. Information

## Optimize Your Supply Chain with WIN SOURCE Solutions

- Global Sourcing Solution
- Obsolete Management
- Cost Control Management
- Shortage Management
- Alternative Solution
- Excess Inventory Management