

MPC8260 IDMA Timing Diagrams

By *DSD Applications, NCSG*
Freescale Semiconductor, Inc.

The MPC8260 PowerQUICC™ II integrated communications processor provides four general-purpose independent DMA (IDMA) channels that support memory-to-memory or peripheral-to/from-memory transfers. This application note presents a series of timing diagrams for several scenarios in single-address fly-by mode. A brief discussion of dual-address mode is included. All the timing diagrams are based on 60x bus accesses when the MPC8260 is in 60x-compatible mode. The timings are very similar for 60x bus accesses in single-MPC8260 mode and for local bus accesses. All the timing diagrams are based on simulations. For further information on IDMA programming, refer to the Freescale application note entitled *MPC8260 SDRAM Timing Diagrams* (AN2178).

Contents

| | | |
|-----|--|----|
| 1 | SDRAM-to-Peripheral, Peripheral-to-SDRAM Fly-by | 2 |
| 2 | GPCM-to-Peripheral, Peripheral-to-GPCM Fly-by | 8 |
| 3 | UPM-to-Peripheral, Peripheral-to-UPM Fly-by | 10 |
| 4 | Slave-to-Peripheral, Peripheral-to-Slave Fly-by | 12 |
| 5 | Dual-Address Mode | 13 |
| 6 | DREQ Timing | 15 |
| 7 | DONE Timing | 18 |
| 7.1 | Fly-by Mode Transfer Termination by External DONE | 19 |
| 7.2 | Dual-address Memory-to-Peripheral Termination by External DONE | 20 |
| 7.3 | Dual-Address Peripheral-to-Memory Termination by External DONE | 21 |
| 8 | Revision History | 22 |

1 SDRAM-to-Peripheral, Peripheral-to-SDRAM Fly-by

Figure 1 shows a DMA fly-by data transfer from SDRAM-to-peripheral. The transaction is single beat with a page miss/hit.

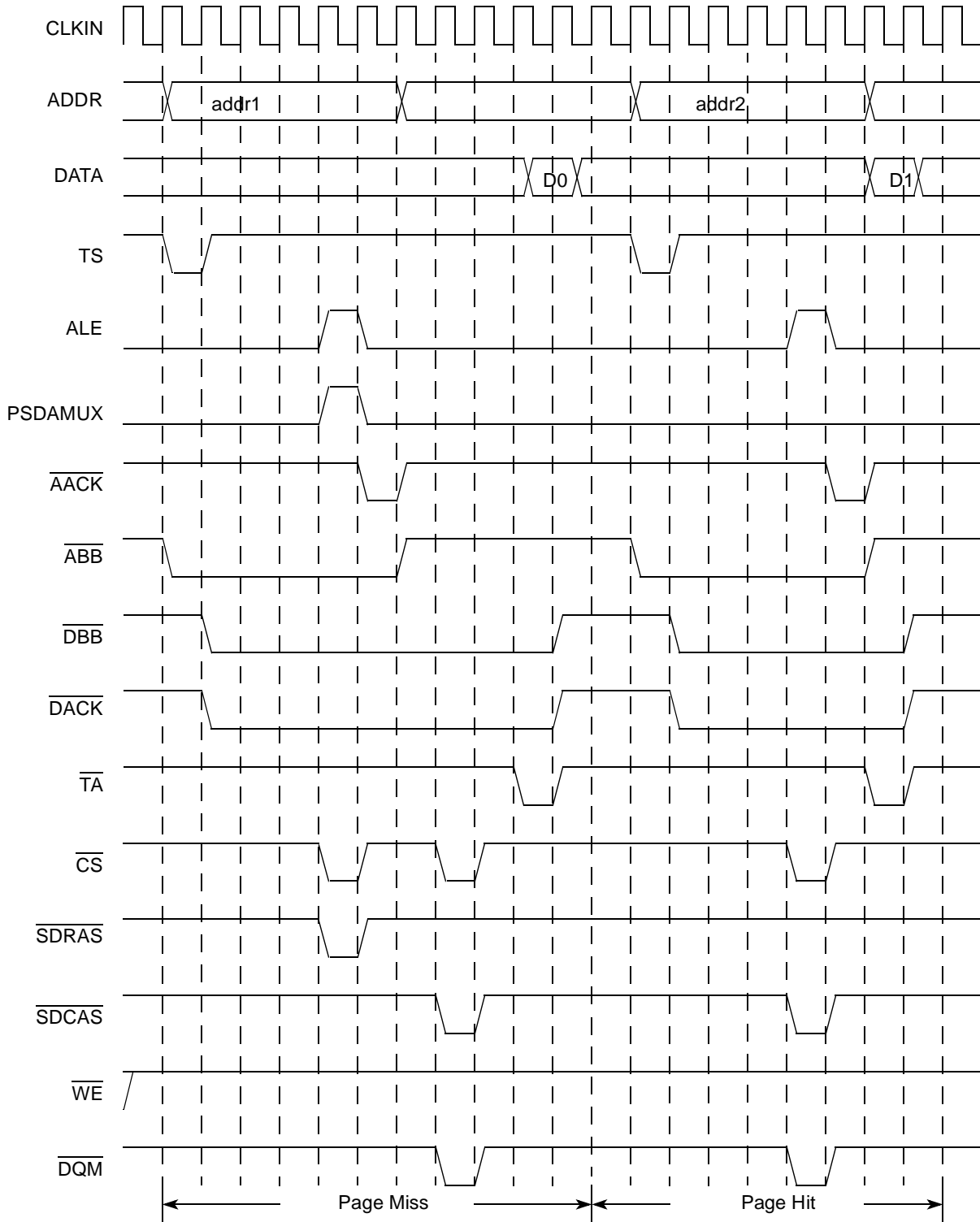


Figure 1. SDRAM-Peripheral Fly-By

Figure 2 shows an IDMA fly-by data transfer from SDRAM-to-peripheral. The transaction is a burst with a page miss.

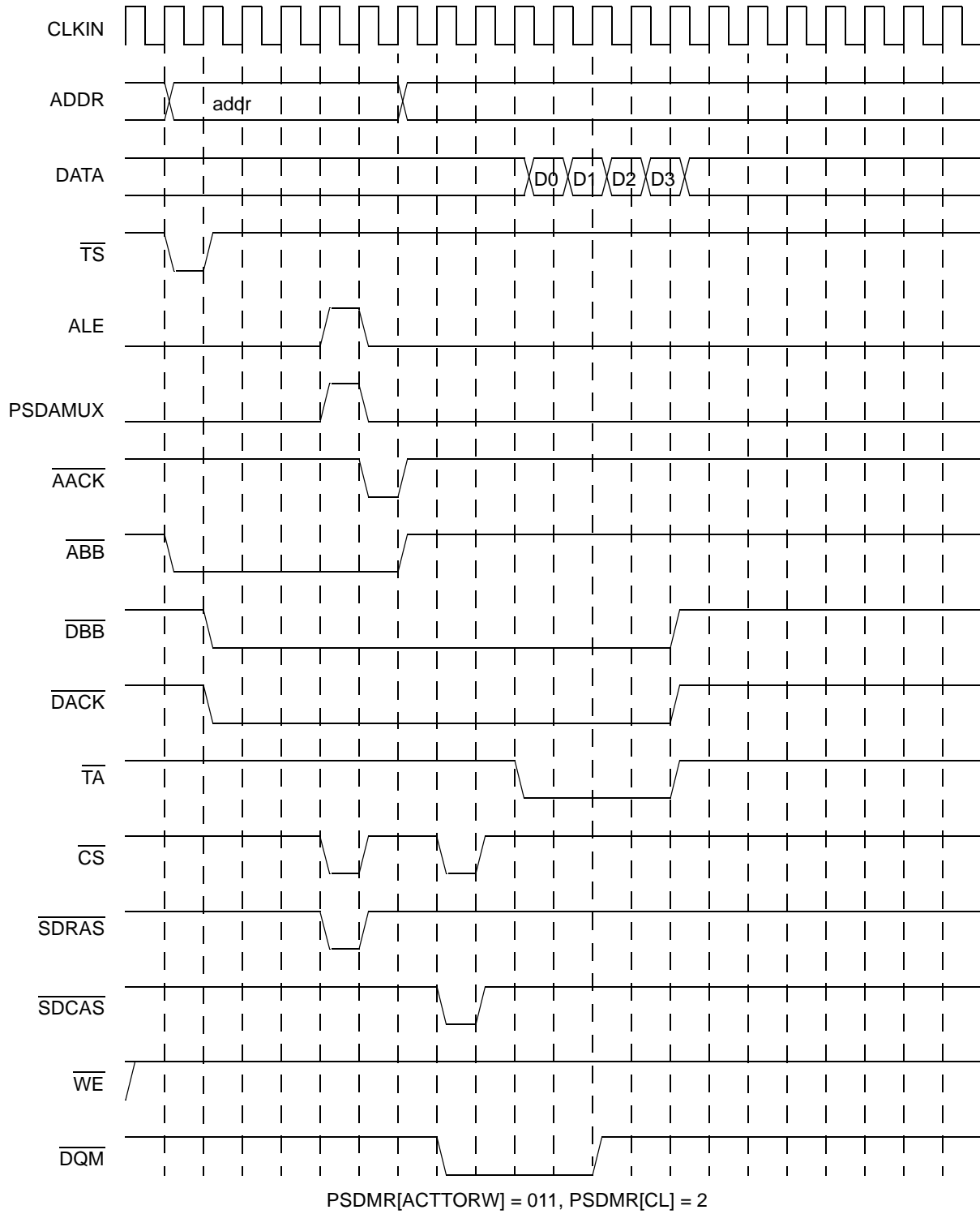
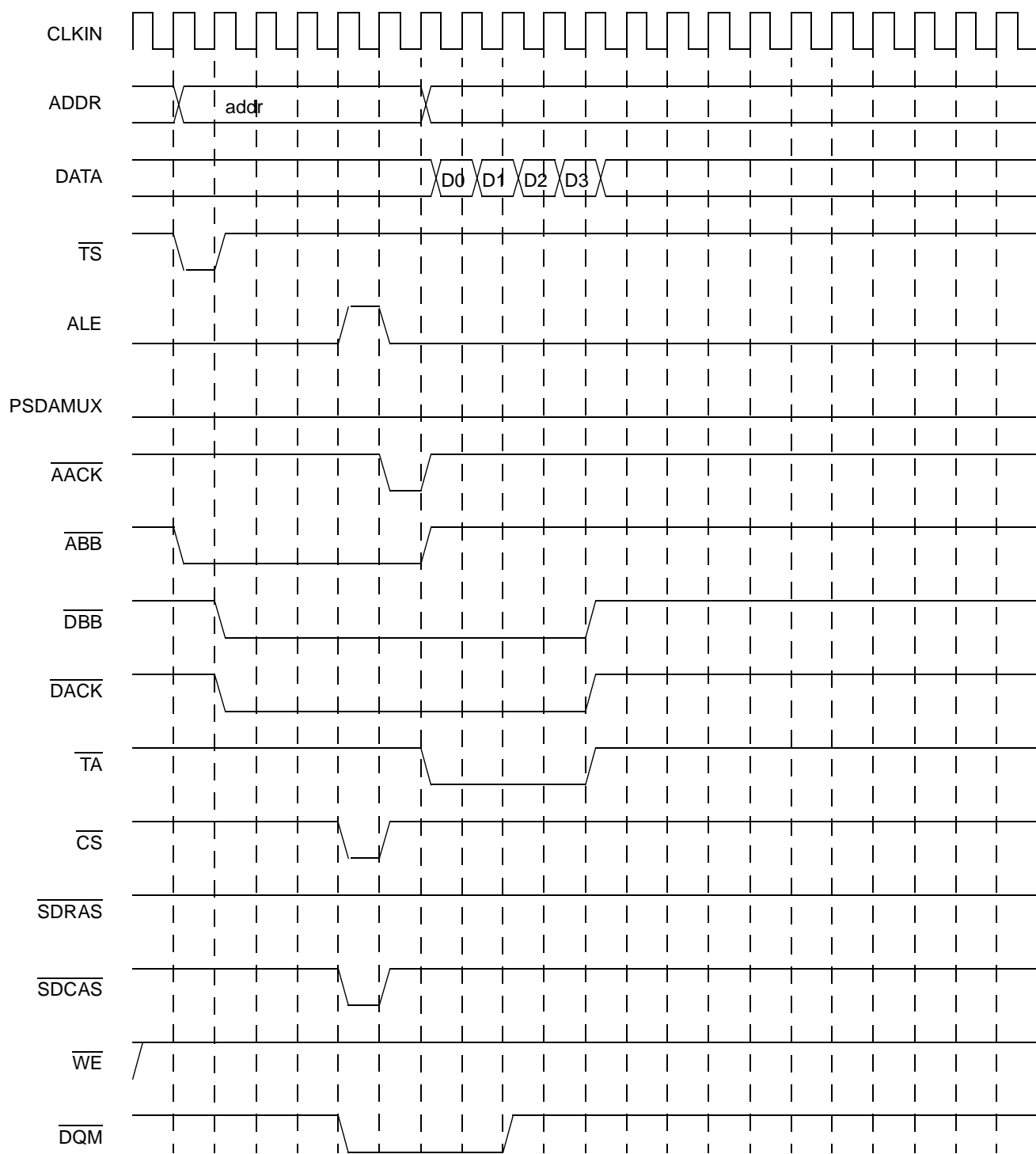


Figure 2. SDRAM-to-Peripheral Burst, Page Missed

Figure 3 shows an IDMA fly-by data transfer from SDRAM-to-peripheral. The transaction is a burst with a page hit.



PSDMR[ACTTORW] = 011, PSDMR[CL] = 2

Figure 3. SDRAM-to-Peripheral Fly-by, Page Hit

Figure 4 shows an IDMA fly-by data transfer from peripheral-to-SDRAM. The transaction is single-beat with a page miss/hit.

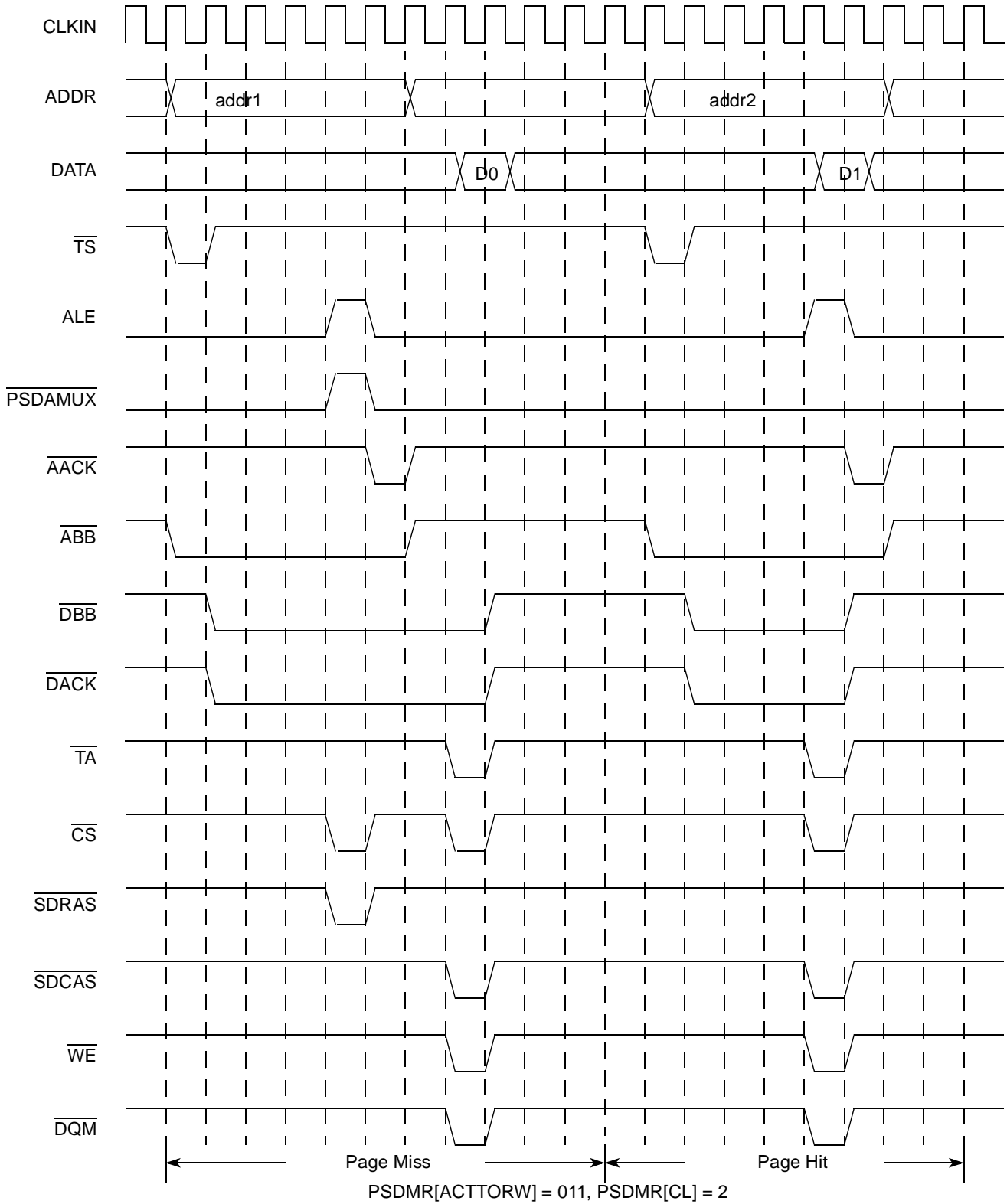
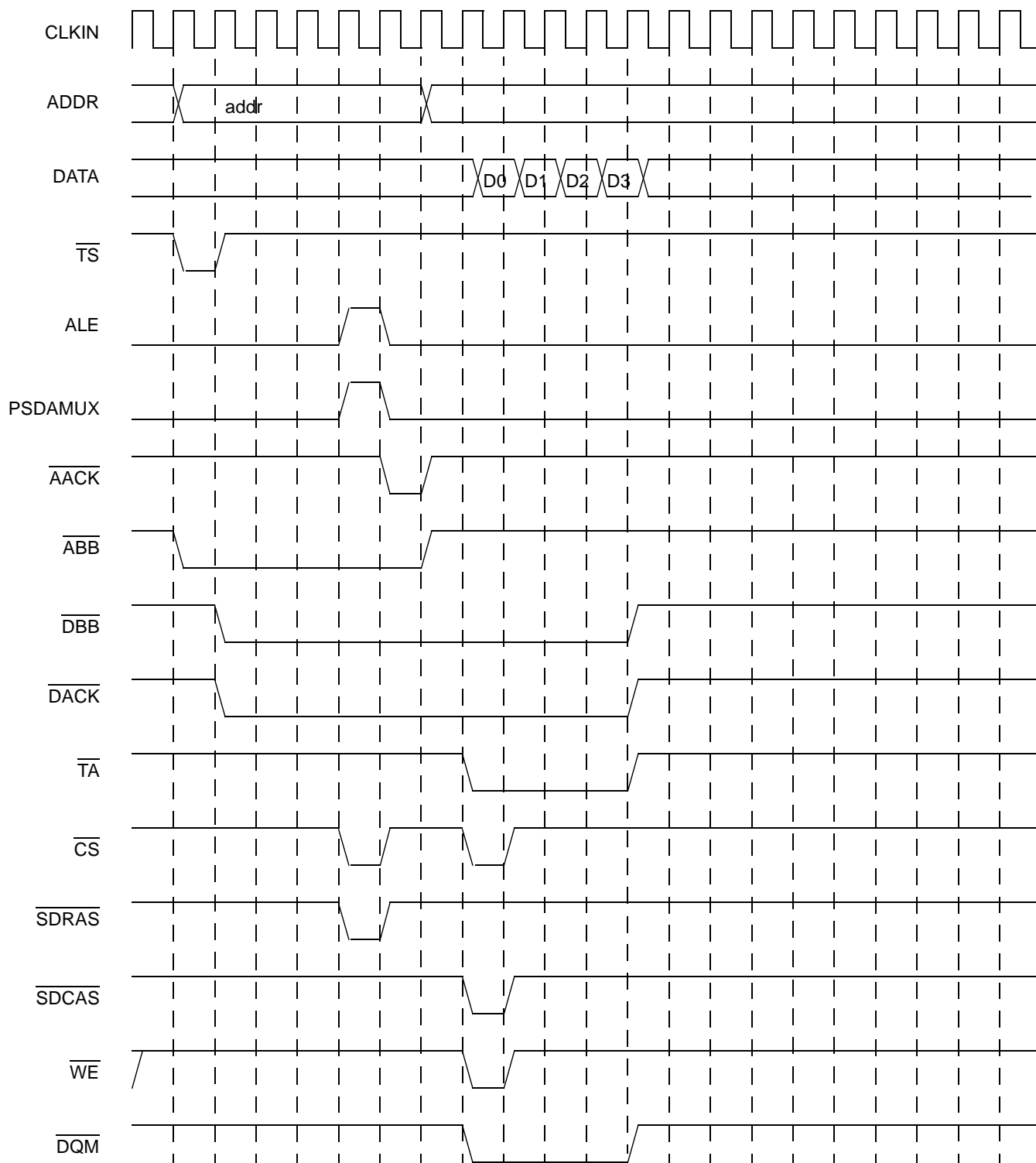


Figure 4. Peripheral-to-SDRAM, Single Beat

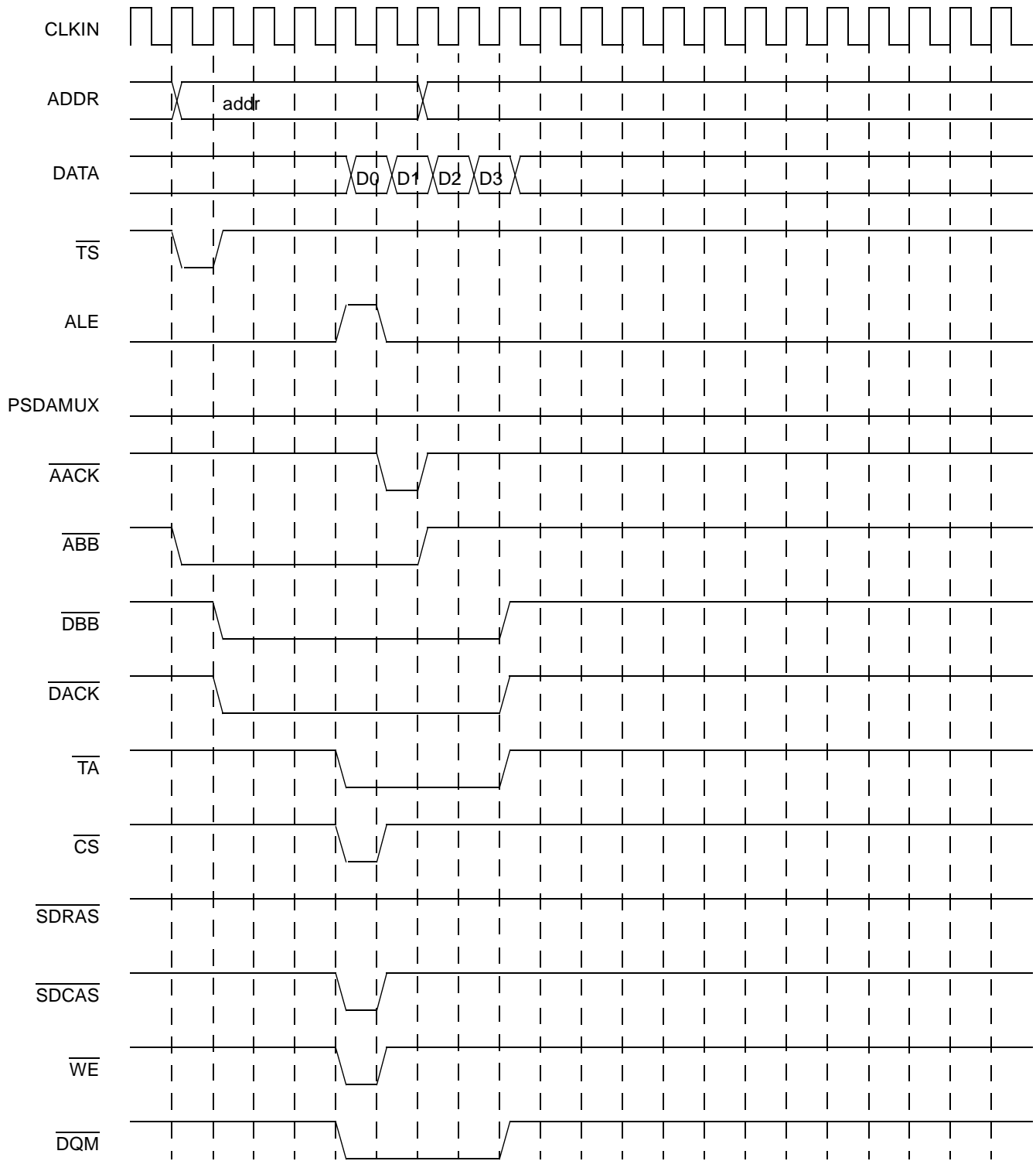
Figure 5 shows a IDMA fly-by data transfer from peripheral-to-SDRAM. The transaction is a burst with a page miss.



PSDMR[ACTTORW] = 011, PSDMR[CL] = 2

Figure 5. Peripheral-to-SDRAM Burst, Page Missed

Figure 6 shows a IDMA fly-by data transfer from peripheral-to-SDRAM. The transaction is a burst with a page hit.



PSDMR[ACTTORW] = 011, PSDMR[CL] = 2

Figure 6. Peripheral-to-SDRAM Burst, Page Hit

2 GPCM-to-Peripheral, Peripheral-to-GPCM Fly-by

Note that GPCM does not support burst mode. [Figure 7](#) shows a IDMA fly-by data transfer from GPCM-to-peripheral. The transaction has a single beat.

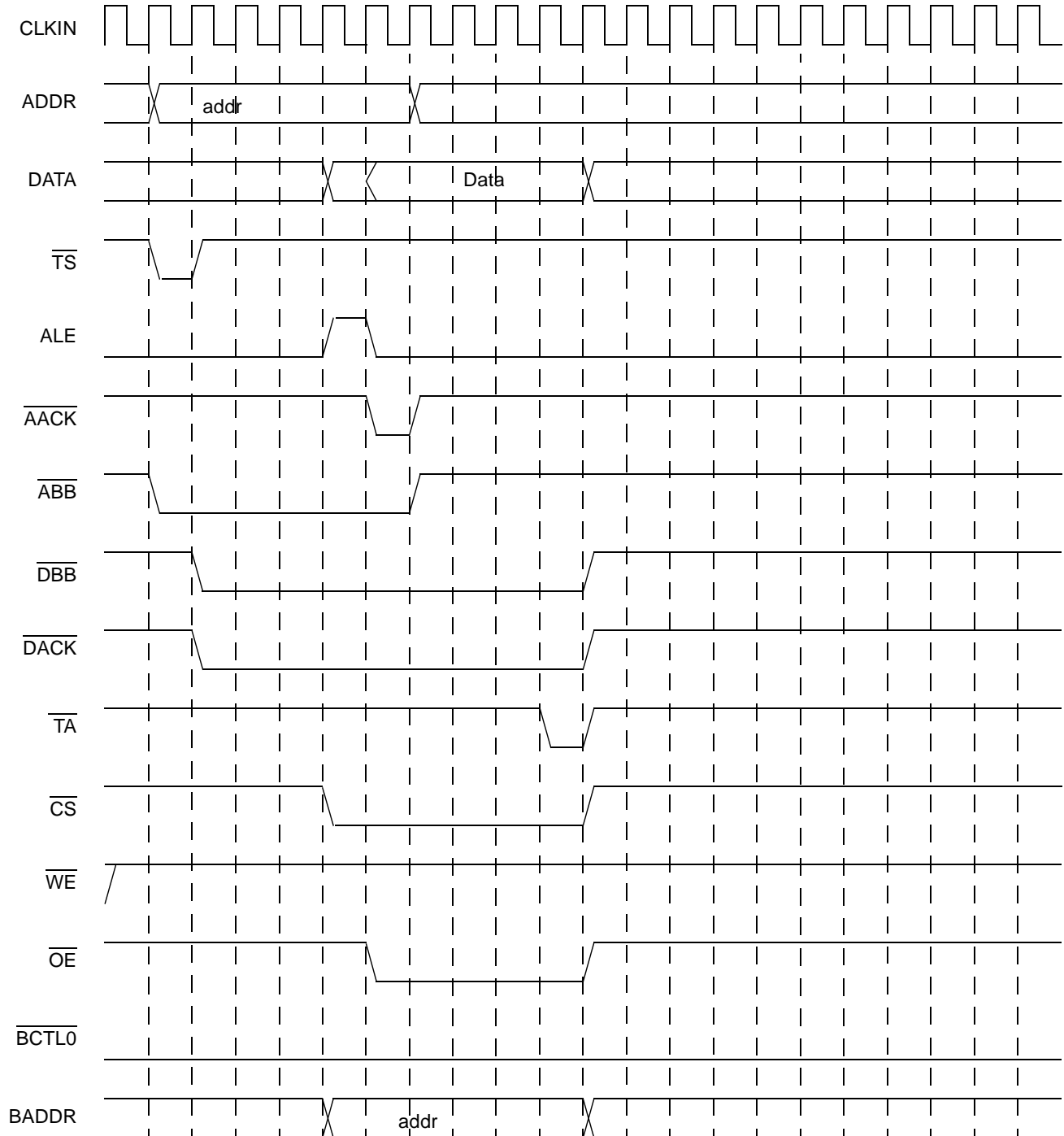


Figure 7. GPCM-to-Peripheral Single-Beat Fly-by

Figure 8 shows an IDMA fly-by data transfer from peripheral-to-GPCM. The transaction has a single beat.

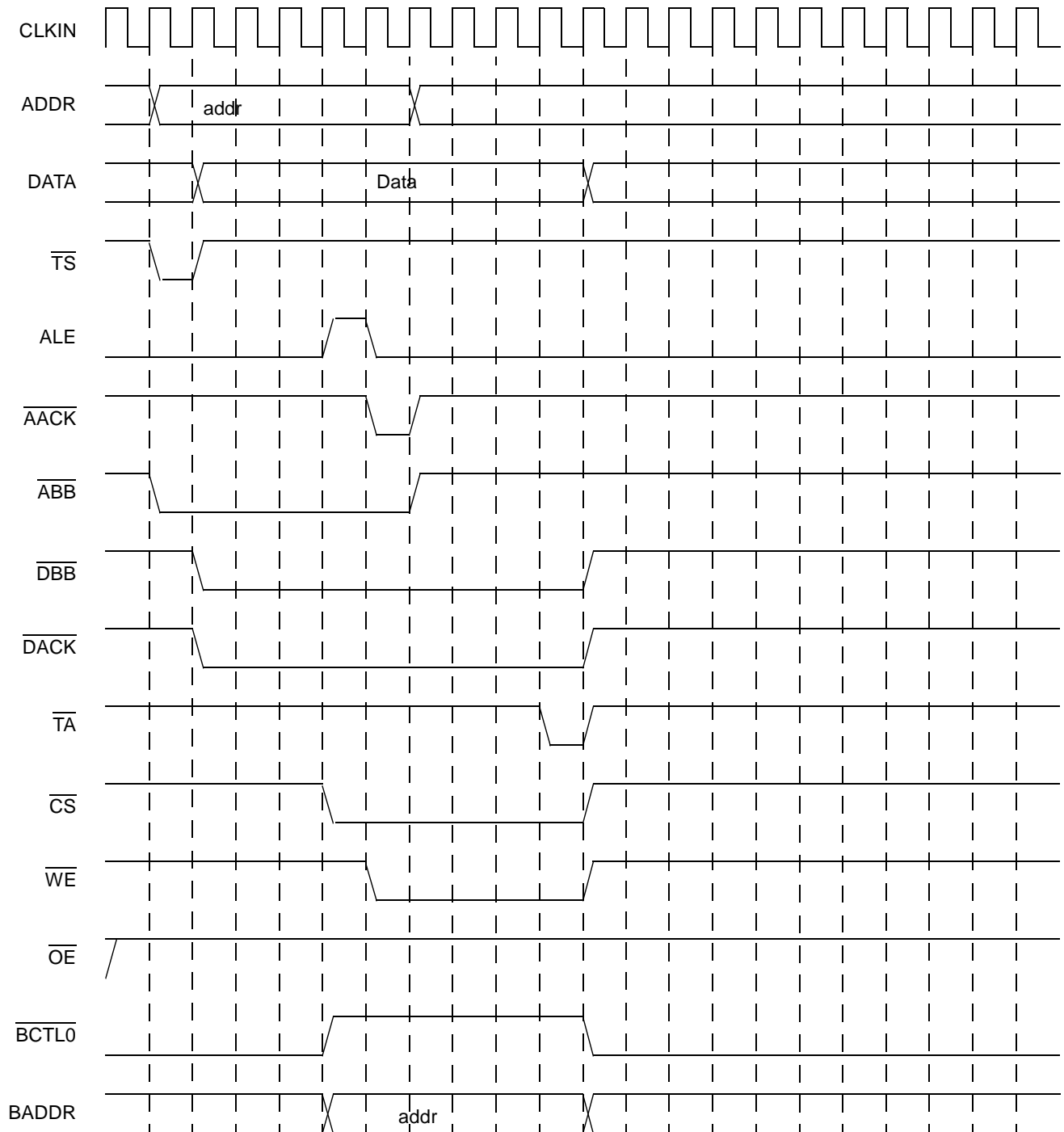
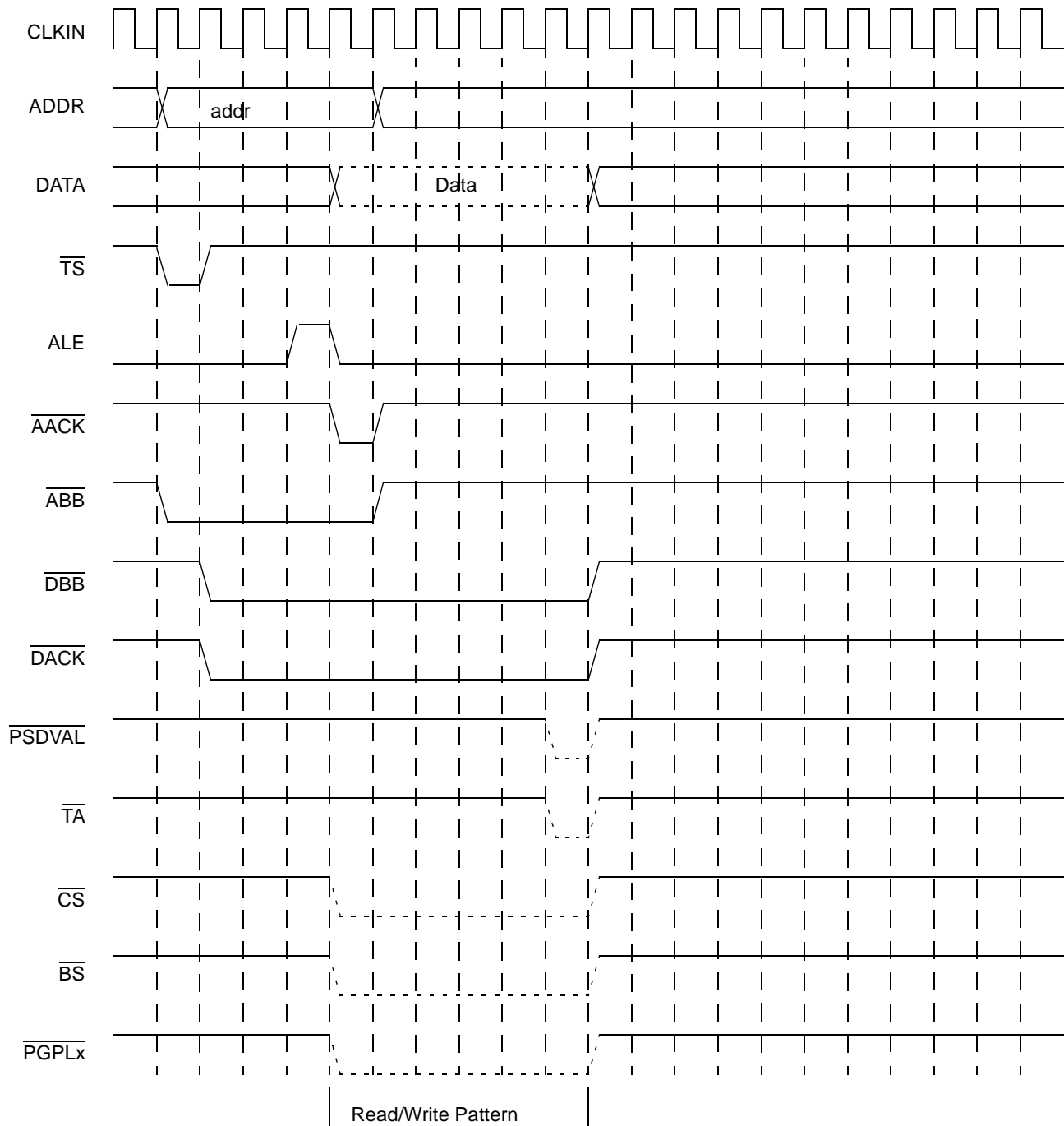


Figure 8. Peripheral-to-GPCM Single-Beat Fly-by

3 UPM-to-Peripheral, Peripheral-to-UPM Fly-by

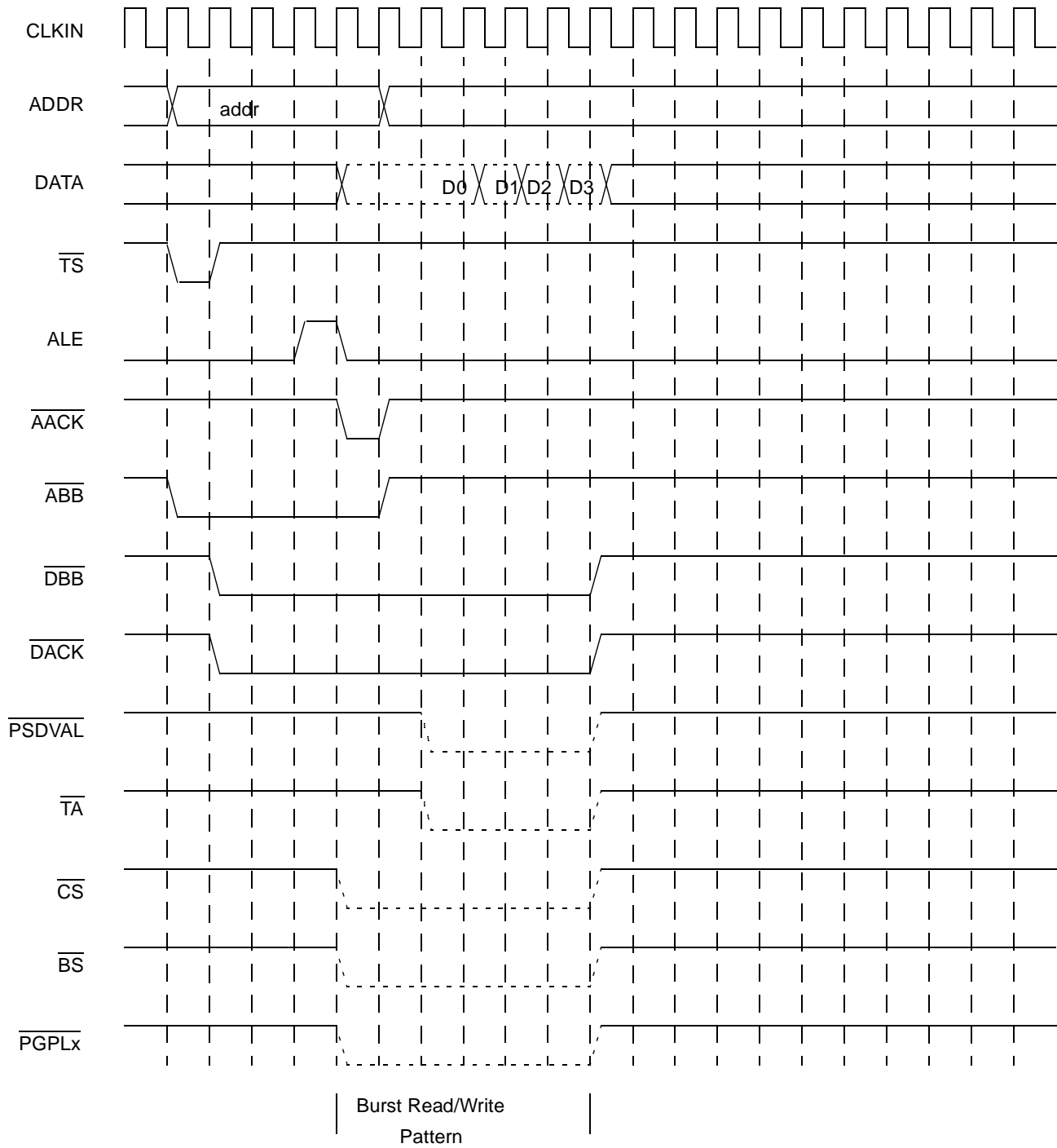
Figure 9 shows an IDMA fly-by transaction from UPM-to-peripheral/peripheral-to-UPM. The transaction has a single beat.



Note: UPM-to-Peripheral corresponds to READ pattern.
Peripheral-to-UPM corresponds to WRITE pattern.

Figure 9. UPM-to-Peripheral/Peripheral-to-UPM Single Beat Fly-by

Figure 10 shows an IDMA fly-by data transfer from UPM-to-peripheral/peripheral-to-UPM. The transaction is a burst.



Note: UPM-to-Peripheral corresponds to BURST READ pattern.
Peripheral-to-UPM corresponds to BURST WRITE pattern.

Figure 10. UPM-to-Peripheral/Peripheral-to-UPM Burst Fly-by

4 Slave-to-Peripheral, Peripheral-to-Slave Fly-by

Figure 11 shows an IDMA fly-by data transfer from slave-to-peripheral.

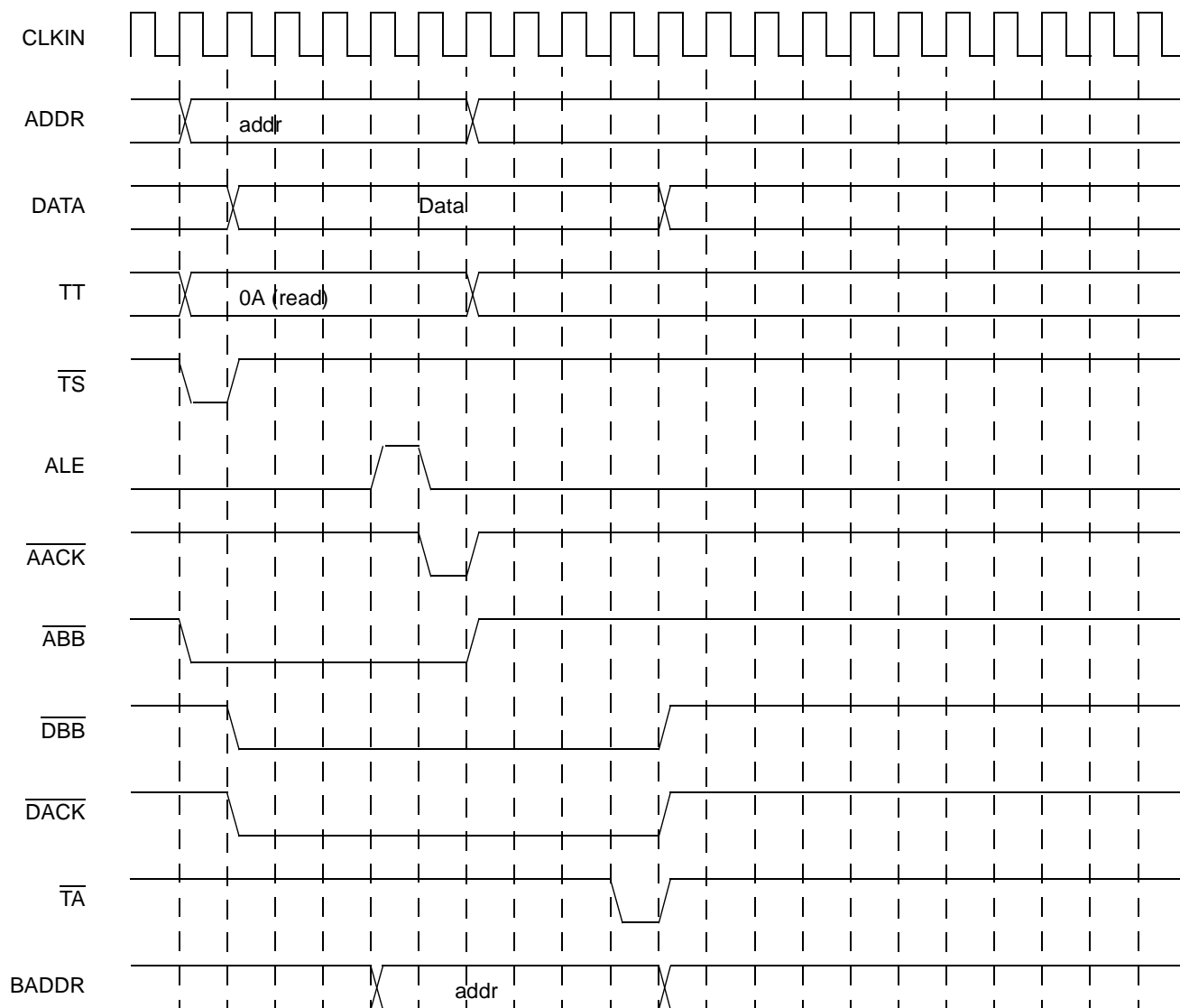


Figure 11. Slave-to-Peripheral Fly-by

Figure 12 shows an IDMA fly-by data transfer from peripheral-to-slave.

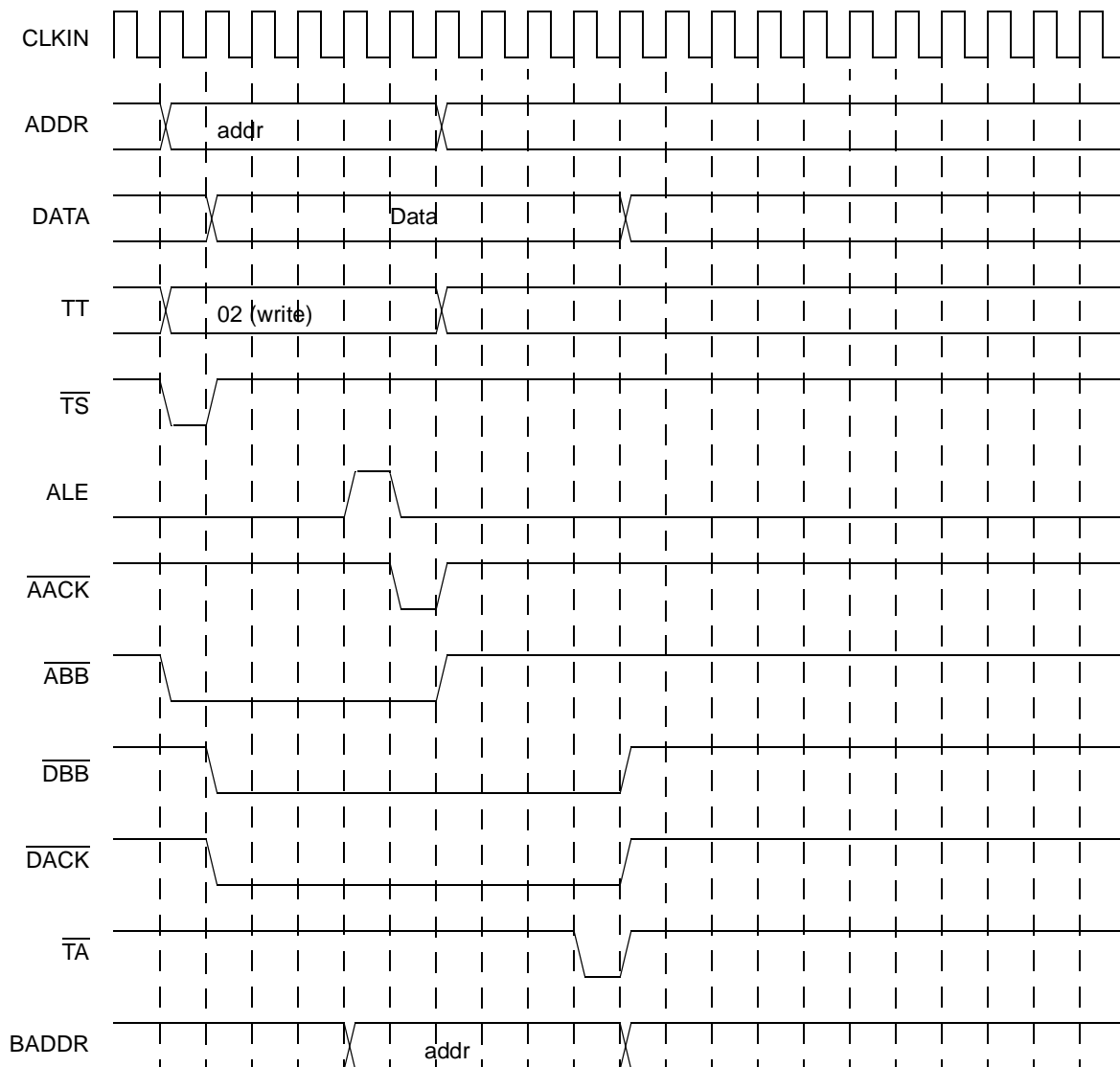


Figure 12. Peripheral-to-Slave Fly-by

5 Dual-Address Mode

IDMA dual-address mode consists of a read phase and a write phase. For timing information for dual-address SDRAM transactions, consult the burst mode section of *MPC8260 SDRAM Timing Diagrams* (AN2178). GPCM does not support burst mode. The timing of dual-address GPCM is the regular GPCM read/write, as described in *MSC8260 GPCM Timing Diagrams* (AN2176). For timing information for dual-address UPM transactions, consult the burst mode section of *MPC8260 UPM Timing Diagrams* (AN2179) if UPM burst mode is enabled (ORx[BI] is cleared). If burst mode is disabled, refer to the single-beat section of this document. The timing diagrams for dual-address peripheral-to/from-memory are shown in [Figure 13](#) and [Figure 14](#).

Figure 13 shows a dual-address peripheral-to-memory data transfer. Only the read-from-peripheral phase is shown. The write-to-memory phase is the regular memory write. The peripheral device can distinguish an IDMA read from a regular read by the assertion of $\overline{\text{DACK}}$ and $\overline{\text{DBB}}$. After the peripheral determines that the data transfer is an IDMA transaction, it can use the information on the address bus and attribute signals as the basis for action. It must assert $\overline{\text{AACK}}$ to terminate the address tenure and output data, and it asserts $\overline{\text{TA}}$ to terminate the data tenure.

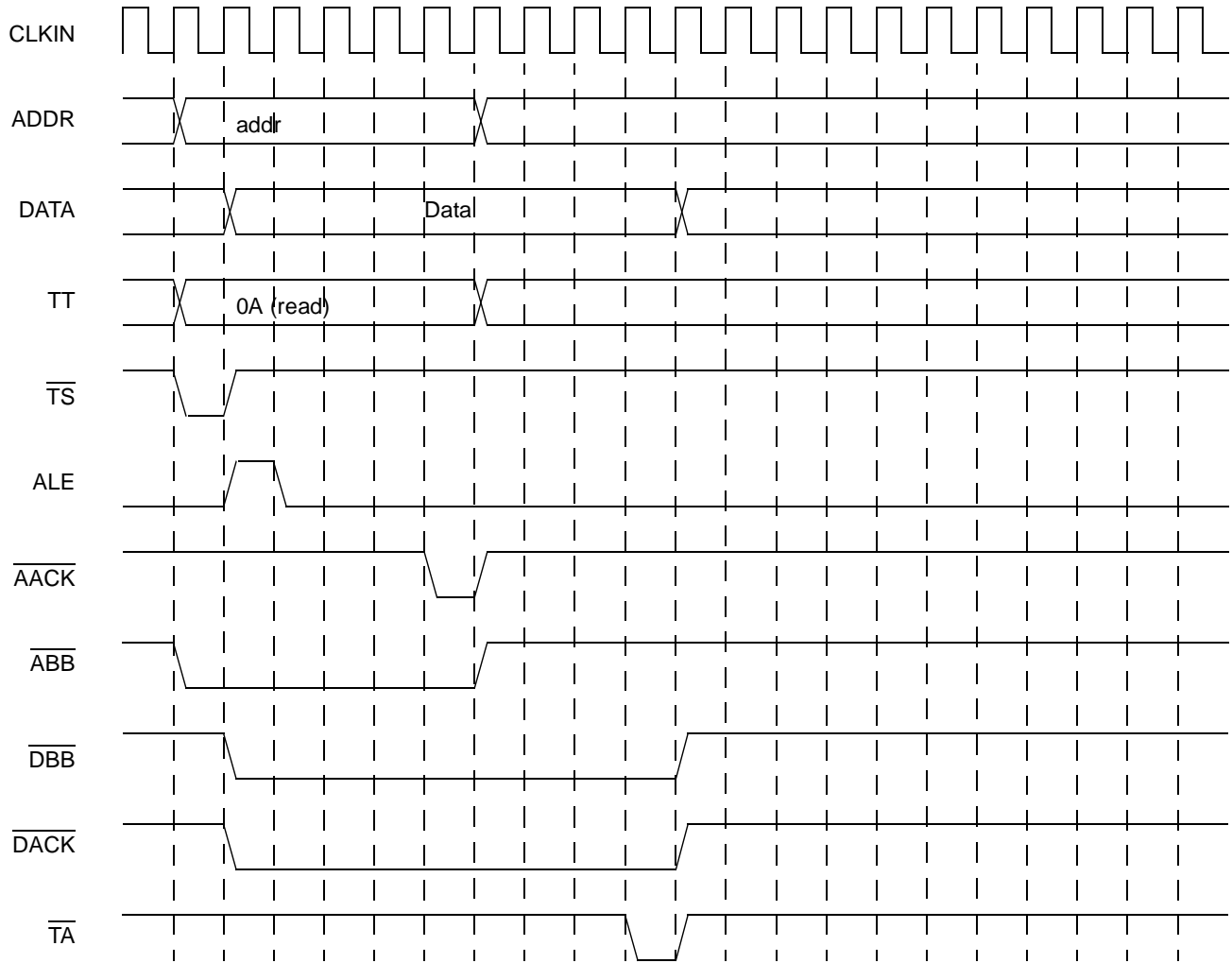


Figure 13. Peripheral-to-Memory Dual-Address

Figure 14 shows a dual-address memory-to-peripheral data transfer. Only the write-to-peripheral phase is shown. The read-from-memory phase is the regular memory read. The peripheral device can distinguish an IDMA write from a regular write by the assertion of $\overline{\text{DACK}}$ and $\overline{\text{DBB}}$. After the peripheral determines that the data transfer is an IDMA transaction, it can use the information on the address bus and attribute signals as the basis for action. It asserts $\overline{\text{AACK}}$ to terminate the address tenure and latch the data, and it asserts $\overline{\text{TA}}$ to terminate the data tenure.

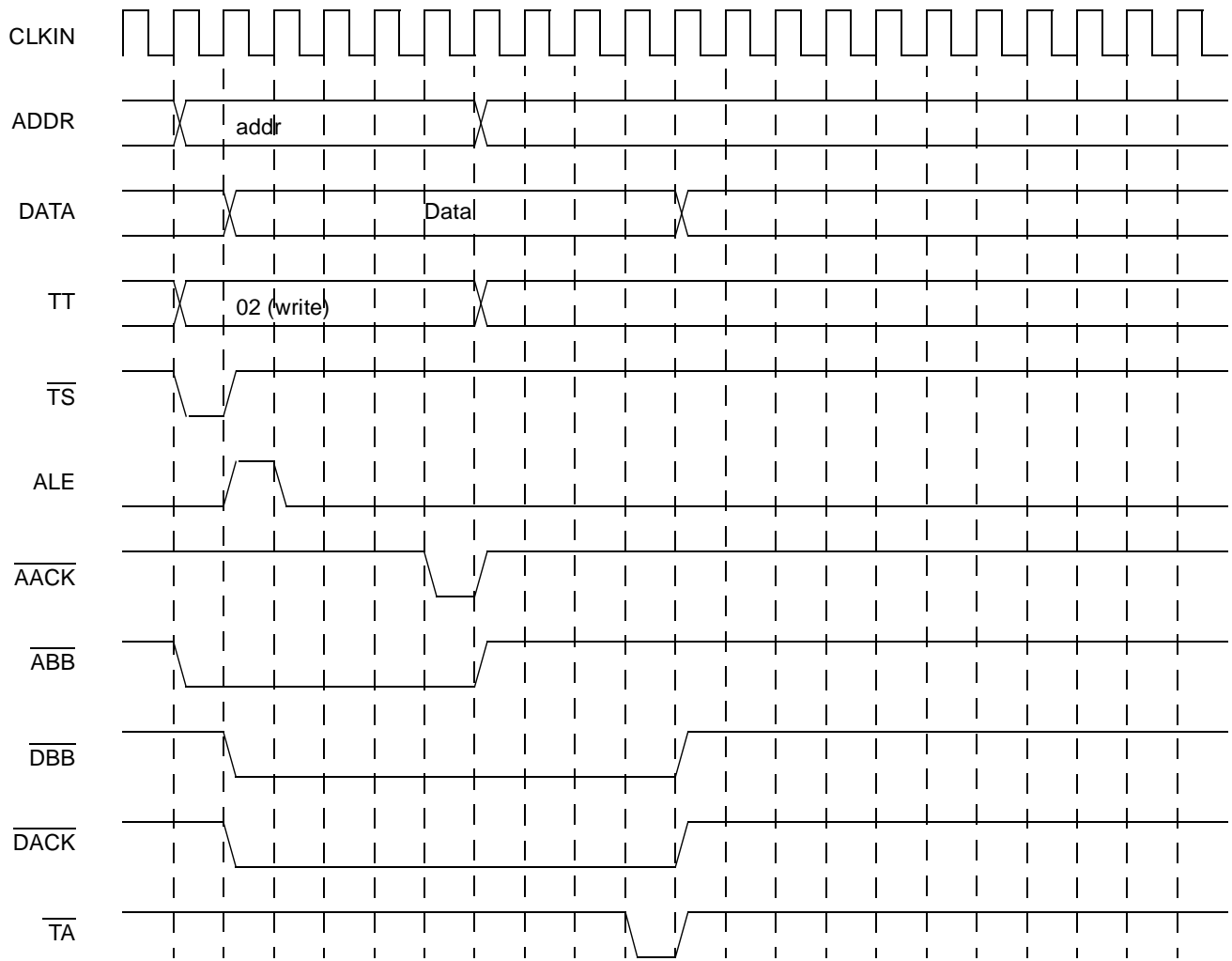


Figure 14. Memory-to-Peripheral Dual-Address

6 \overline{DREQ} Timing

Figure 15 shows \overline{DREQ} timing for a dual-address data transfer from peripheral-to-memory. The first \overline{DREQ} peripheral assertion triggers a read of STS bytes from the peripheral. Subsequent \overline{DREQ} assertions trigger the same read from the peripheral. When the internal buffer reaches the steady-state level, it is automatically written to the memory destination in one transfer.

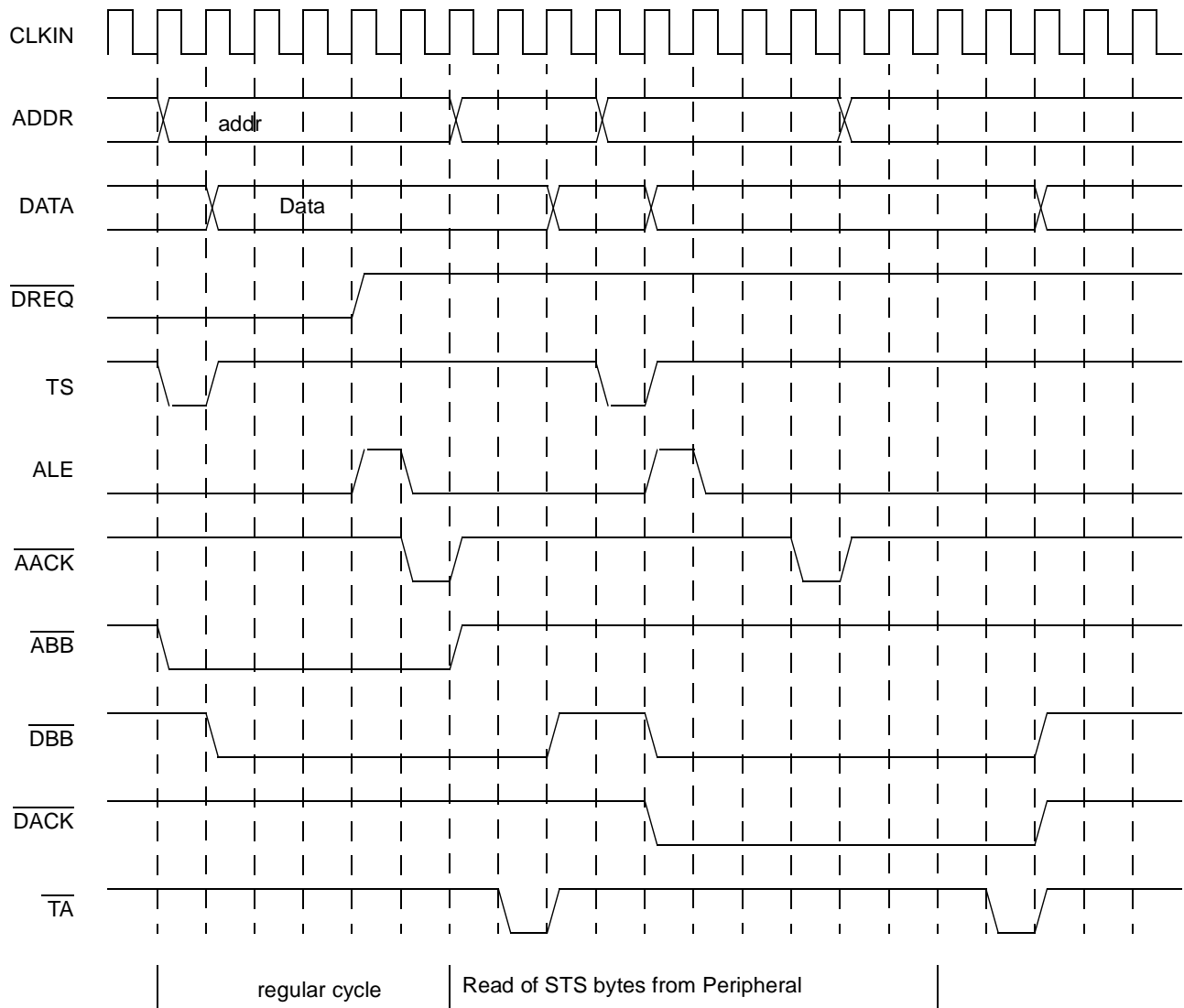


Figure 15. Peripheral-to-Memory Dual Address

Figure 16 shows the $\overline{\text{DREQ}}$ timing for a dual-address data transfer from memory-to-peripheral. The first $\overline{\text{DREQ}}$ peripheral assertion triggers a read of SS_MAX bytes from the memory into the internal transfer buffer, automatically followed by a write of DTS bytes to the peripheral. Subsequent $\overline{\text{DREQ}}$ assertions trigger writes to the peripheral. When the transfer buffer has fewer than DTS bytes left, the next $\overline{\text{DREQ}}$ assertion triggers a read of SS_MAX bytes from the memory, automatically followed by a write to the peripheral, and the sequence begins again.

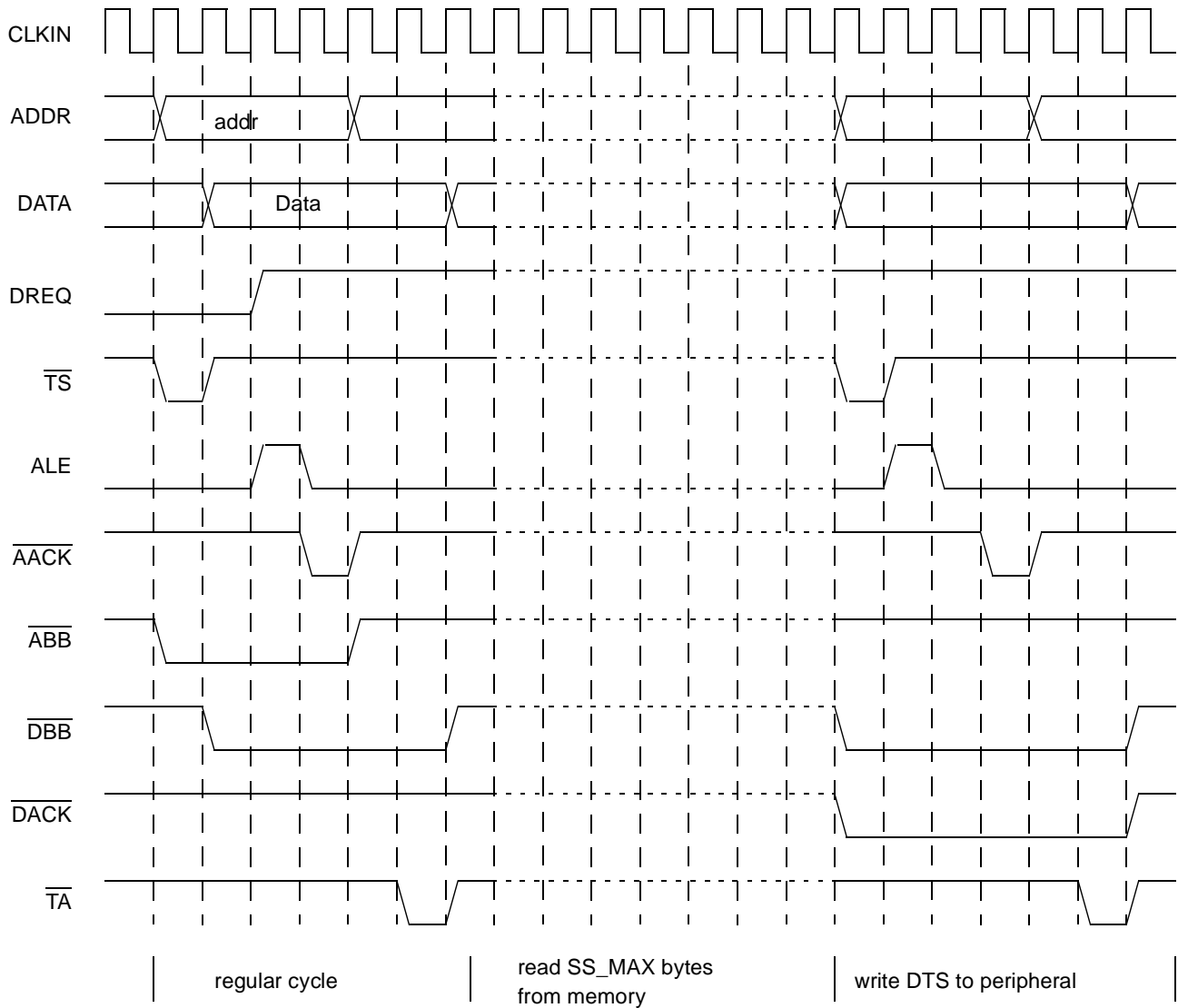


Figure 16. Memory-to-Peripheral Dual Address

Figure 17 shows $\overline{\text{DREQ}}$ timing for fly-by mode. Each $\overline{\text{DREQ}}$ peripheral assertion triggers a transfer of the port size between peripheral and memory directly. When the programmed transfer length is reached or the peripheral asserts $\overline{\text{DONE}}$, the BD is closed.

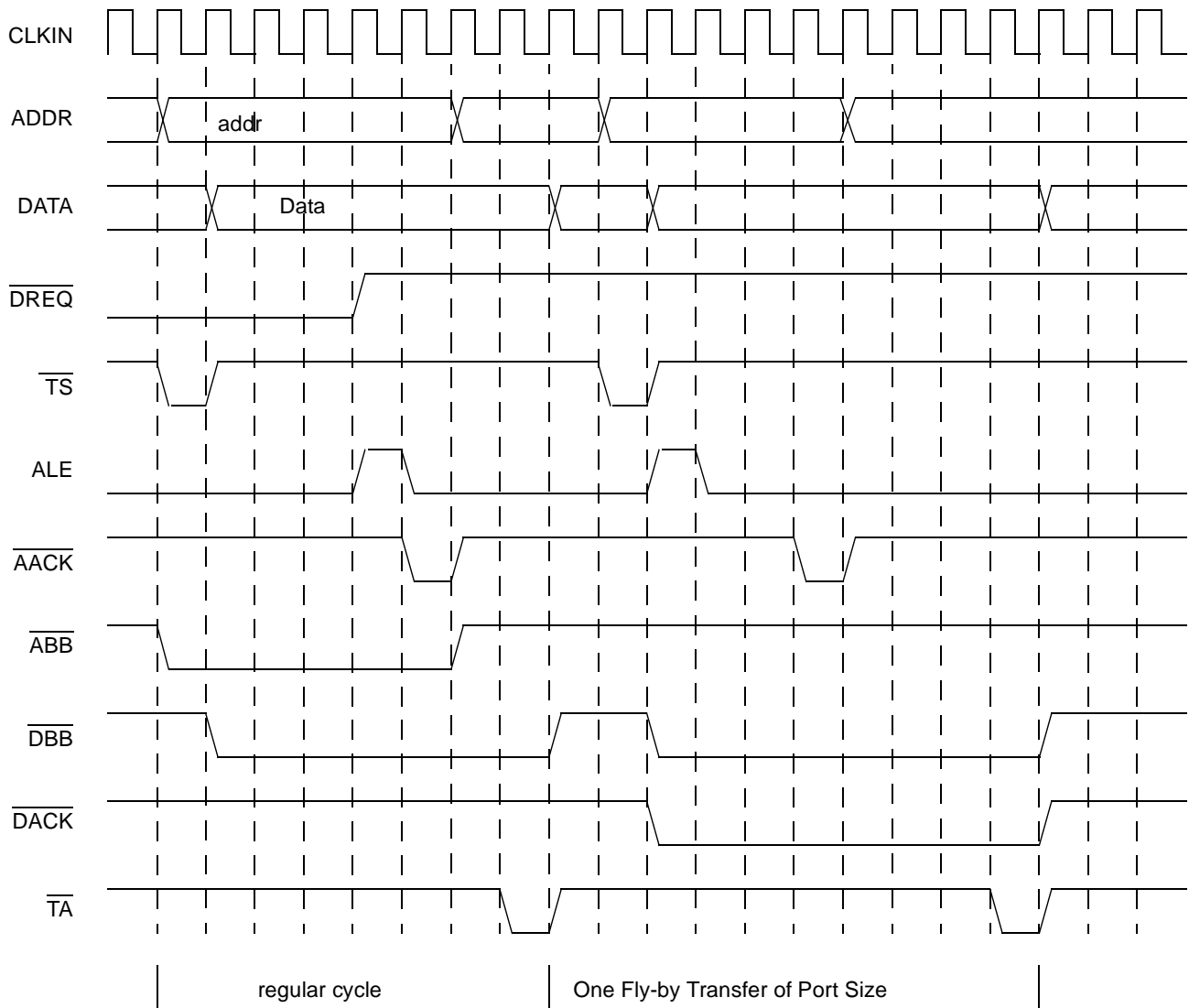


Figure 17. Fly-by Mode

7 $\overline{\text{DONE}}$ Timing

Considerations for internally asserting $\overline{\text{DONE}}$ are as follows:

- When IDMA finishes transferring the programmed number of data at its last phase of read/write, the MPC8260 asserts $\overline{\text{DONE}}$, which is enabled/disabled by the BD[SDN] and BD[DDN] bits.
- For dual-address memory-to-memory mode, $\overline{\text{DONE}}$ assertion is not supported and should be disabled.
- For fly-by mode, SDN should be the same as DDN. If SDN = DDN = 0, $\overline{\text{DONE}}$ is disabled. Otherwise, it is enabled.
- For dual-address peripheral-to/from-memory mode, only the bit associated with the peripheral can be enabled. For example, for peripheral-to-memory dual address mode, the peripheral is the source. SDN can have a value of either 0 or 1, but DDN should be cleared. If DDN is set, then during the

last write to memory phase, \overline{DONE} as well as \overline{DACK} is asserted. The assertion of \overline{DACK} during memory access cycles may cause problem for the peripheral.

- When \overline{DONE} is asserted, its waveform is the same as that of \overline{DACK} .

Considerations for externally asserting \overline{DONE} are as follows:

- If the peripheral determines that there is no more data to transfer, it can assert \overline{DONE} externally to terminate the IDMA operation.
- For dual-address memory-to-memory mode, \overline{DONE} assertion is not supported.
- [Figure 18](#) and [Figure 19](#) illustrate two scenarios for fly-by mode. When \overline{DONE} is asserted and the IDMA has not yet asserted \overline{DREQ} , the IDMA executes the pending \overline{DREQ} . That is, it does one more transfer and stops. If there is no pending \overline{DREQ} , the IDMA stops right after \overline{DONE} assertion.
- For dual-address memory-to-peripheral and peripheral-to-memory modes, timing diagrams of three scenarios are given for each mode.

7.1 Fly-by Mode Transfer Termination by External \overline{DONE}

[Figure 18](#) shows a fly-by mode transfer termination by an external \overline{DONE} without a pending \overline{DREQ} . When \overline{DONE} is asserted and all previous \overline{DREQ} signals are serviced so that there is no pending \overline{DREQ} , the BD is closed and IDMA stops right after \overline{DONE} . All the signals are not scaled. \overline{DONE} requires one cycle.

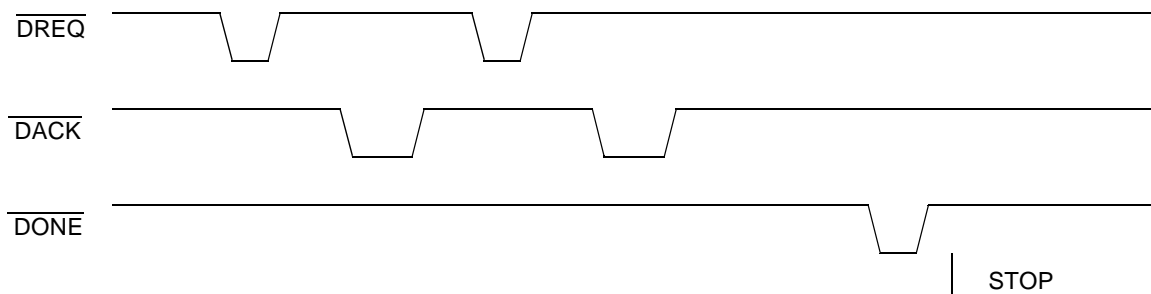


Figure 18. Fly-by Mode Extermination by \overline{DONE} Without Pending \overline{DREQ}

[Figure 19](#) shows a fly-by mode transfer termination by an external \overline{DONE} with a pending \overline{DREQ} . When \overline{DONE} is asserted, there is one unserviced \overline{DREQ} . The BD is closed and the IDMA stops after the pending \overline{DREQ} is serviced. All the signals are not scaled. \overline{DONE} requires one cycle.

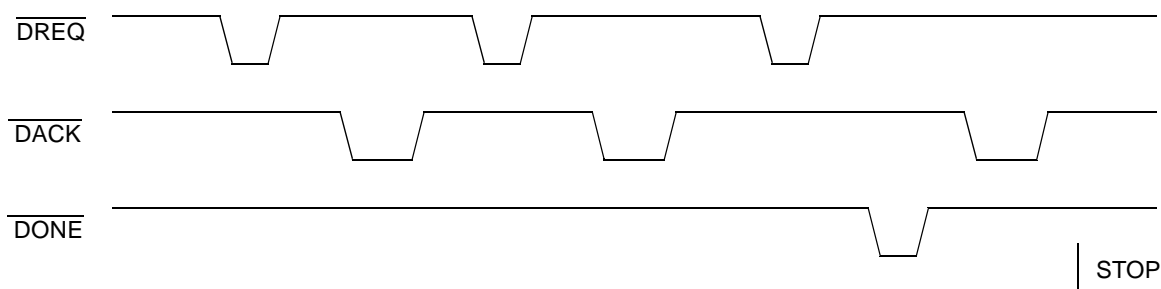


Figure 19. Fly-by Mode Extermination by \overline{DONE} with Pending \overline{DREQ}

7.2 Dual-address Memory-to-Peripheral Termination by External DONE

Figure 20 shows a dual-address memory-to-peripheral mode scenario in which the previous \overline{DREQ} triggers a memory read. After the memory read, one write to the peripheral follows automatically without the need to assert \overline{DREQ} . This memory read plus one write is integral. Even if \overline{DONE} is asserted after \overline{DREQ} but before the peripheral write, the IDMA proceeds with this memory read + write to peripheral combination and then stops. All the signals are not scaled. \overline{DONE} requires one cycle.

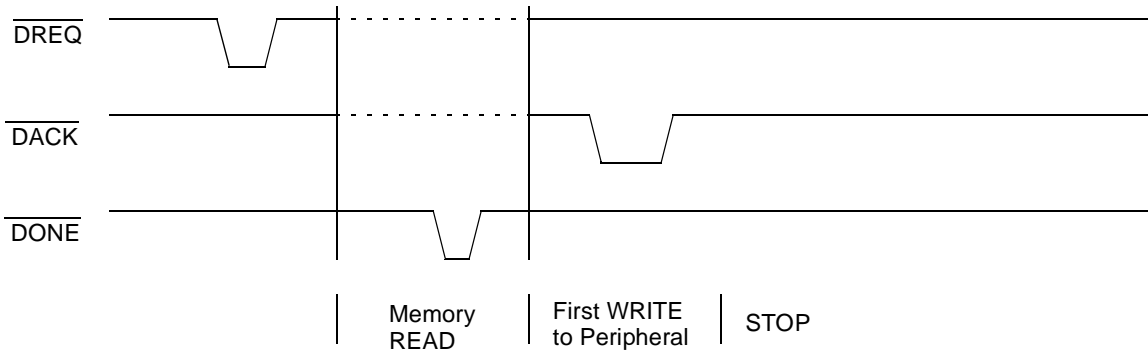


Figure 20. Memory-to-Peripheral Mode Extermination by \overline{DONE} Scenario 1

Figure 21 shows a dual-address memory-to-peripheral mode scenario in which the previous \overline{DREQ} triggers a write to the peripheral. Then \overline{DONE} assertion follows without a pending \overline{DREQ} . The IDMA stops right after \overline{DONE} assertion. All the signals are not scaled. \overline{DONE} requires one cycle.

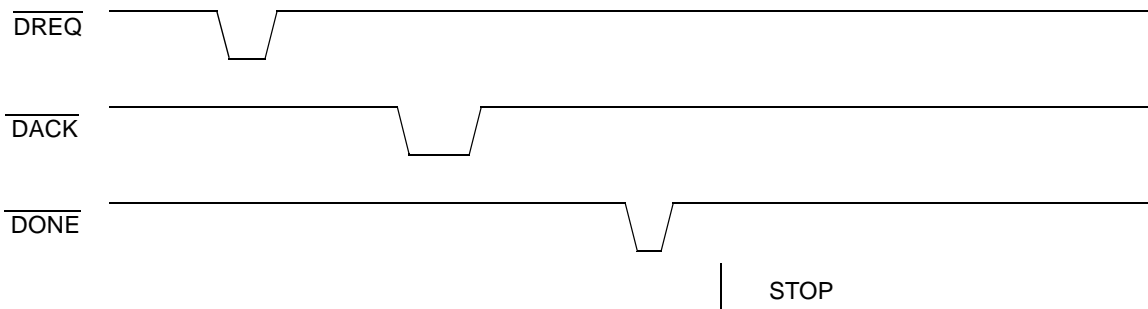


Figure 21. Memory-to-Peripheral Mode Extermination by \overline{DONE} Scenario 2

Figure 22 shows a dual-address memory-to-peripheral mode scenario in which there is one unserved \overline{DREQ} when \overline{DONE} is asserted. The IDMA stops after this pending \overline{DREQ} is serviced. All the signals are not scaled. \overline{DONE} requires one cycle.

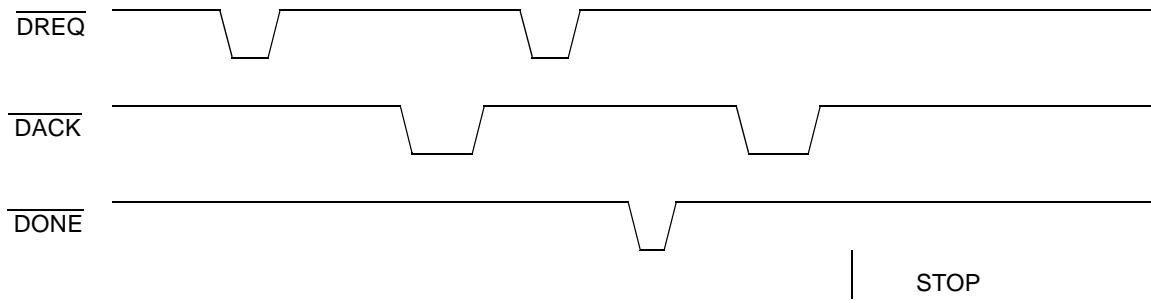


Figure 22. Memory-to-Peripheral Mode Extermination by $\overline{\text{DONE}}$ Scenario 3

7.3 Dual-Address Peripheral-to-Memory Termination by External $\overline{\text{DONE}}$

Figure 23 shows a dual-address peripheral-to-memory mode scenario in which the internal buffer reaches SS_MAX bytes and automatically triggers a write to memory without the need to assert $\overline{\text{DREQ}}$. Even when $\overline{\text{DONE}}$ is asserted prior to the memory write, the IDMA finishes the memory write and stops. All the signals are not scaled. $\overline{\text{DONE}}$ requires one cycle.

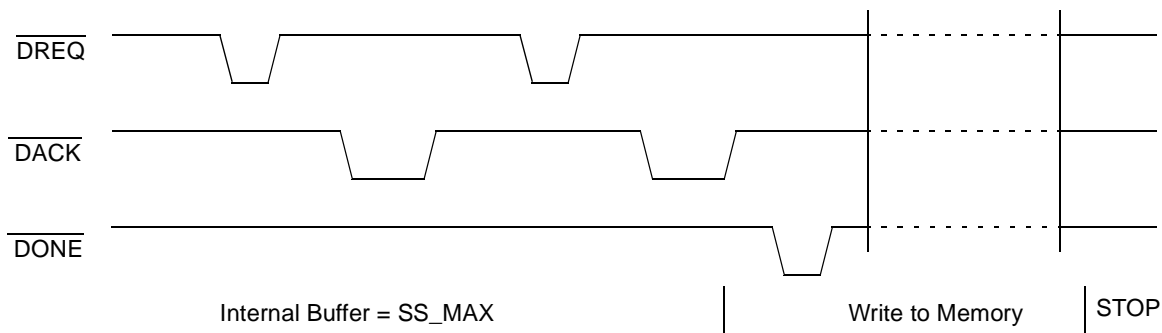


Figure 23. Peripheral-to-Memory Mode Extermination by $\overline{\text{DONE}}$ Scenario 1

Figure 24 shows a dual-address peripheral-to-memory mode scenario 2 in which the previous $\overline{\text{DREQ}}$ does not fill the internal buffer. When $\overline{\text{DONE}}$ is asserted, there is no unserved $\overline{\text{DREQ}}$. The IDMA transfers all the data in the buffer to memory and then stops. All the signals are not scaled. $\overline{\text{DONE}}$ requires one cycle.

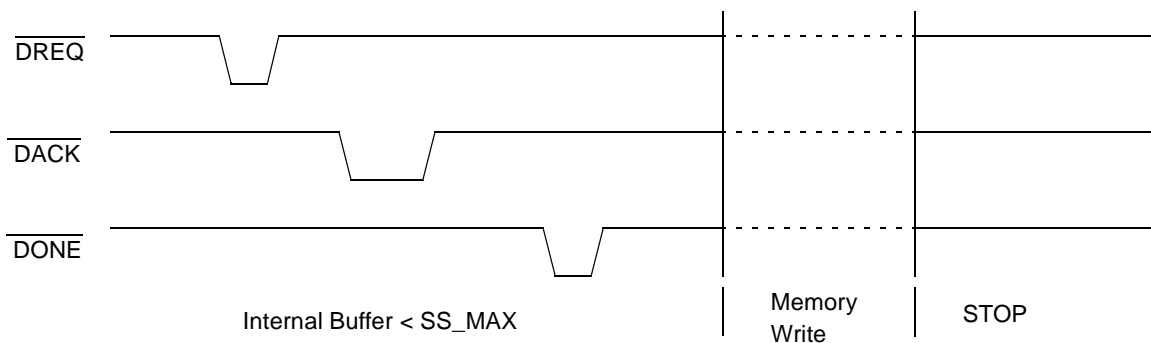


Figure 24. Peripheral-to-Memory Mode Extermination by $\overline{\text{DONE}}$ Scenario 2

Figure 25 shows a dual-address peripheral-to-memory mode scenario in which the previous $\overline{\text{DREQ}}$ does not fill the internal buffer. When $\overline{\text{DONE}}$ is asserted, there is one unserved $\overline{\text{DREQ}}$. The IDMA services this pending $\overline{\text{DREQ}}$ first and then transfers all the data in the buffer to memory and stops. All the signals are not scaled. $\overline{\text{DONE}}$ requires one cycle.

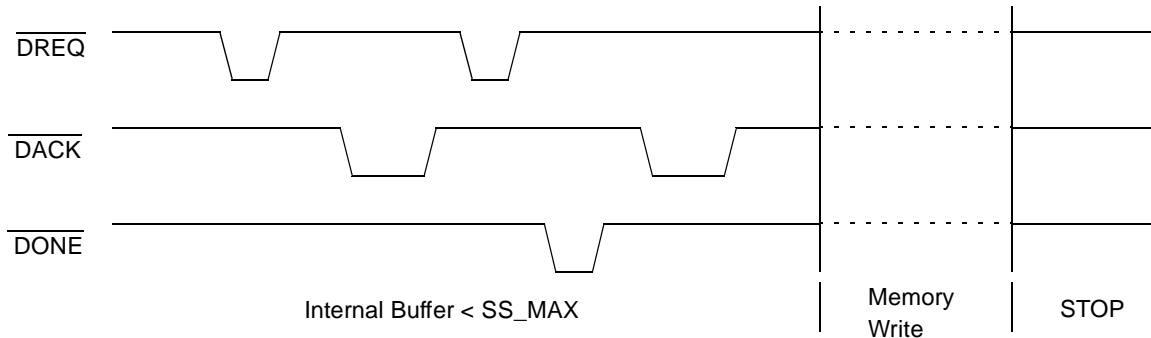


Figure 25. Memory-to-Peripheral Mode Extermination by $\overline{\text{DONE}}$ Scenario 3

8 Revision History

Table 1 provides a revision history for this application note.

Table 1. Document Revision History

| Rev. Number | Date | Change(s) |
|-------------|--------|--|
| 0 | 1998 | Initial release. |
| 1 | 1999 | No record. |
| 2 | 2000 | No record. |
| 3 | 6/2006 | Fixed the incomplete signal timings in the figures on pages 3, 5, 9. Updated the document in the Freescale template. |
| 4 | 7/2006 | Added the revision history to the document. |

THIS PAGE INTENTIONALLY LEFT BLANK

How to Reach Us:

Home Page:

www.freescale.com

email:

support@freescale.com

USA/Europe or Locations Not Listed:

Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
1-800-521-6274
480-768-2130
support@freescale.com

Europe, Middle East, and Africa:

Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:

Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064, Japan
0120 191014
+81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:

Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:

Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447
303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© Freescale Semiconductor, Inc., 2000, 2006.

Document Number: AN2177
Rev. 4
07/2006