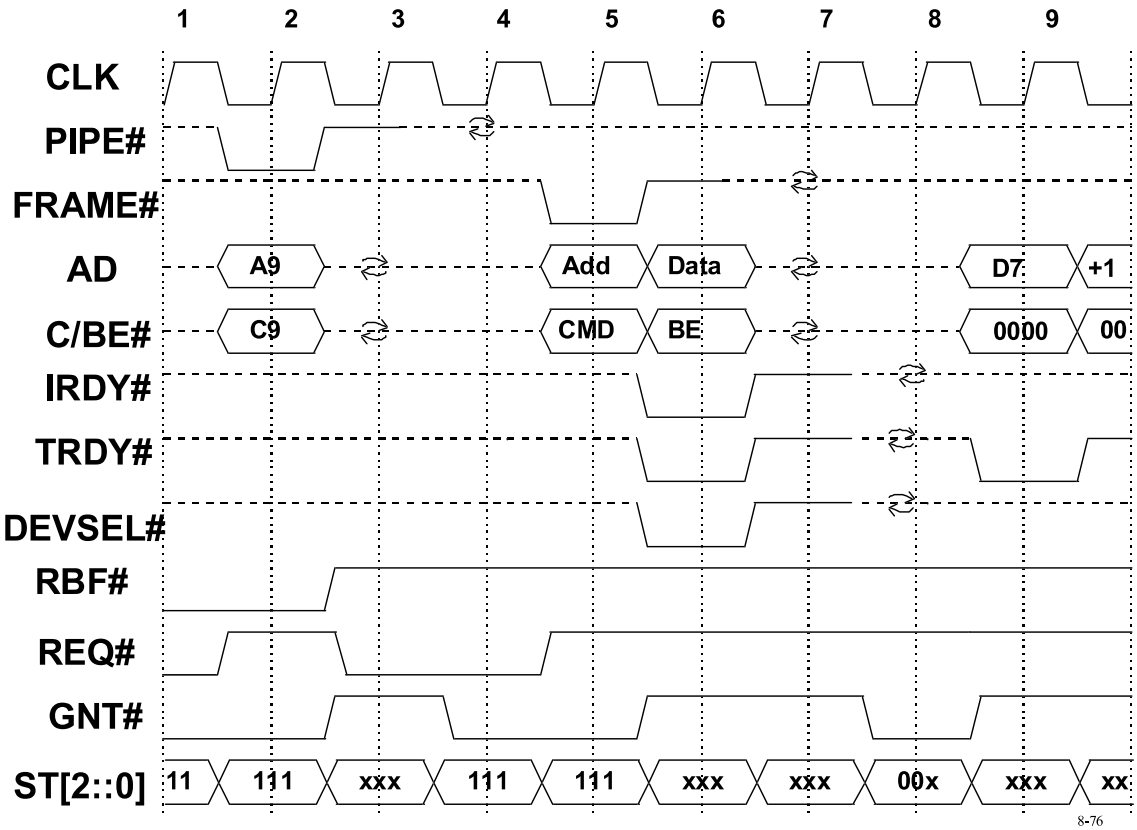


Figure 3-31 PCI Transaction Between A.G.P. Request and Data

Figure 3-30 shows the quickest that the A.G.P. compliant master can enqueue a request, perform a PCI transaction and then start returning read data. Two clocks are required between the completion of the A.G.P. request and the start of a PCI transaction, this is required because the A.G.P. **REQ#** line must be deasserted when **PIPE#** is asserted to indicate that the current request is the last to be enqueued. The earliest the A.G.P. compliant master can request a PCI transaction is on clock 3, and the earliest the arbiter can grant permission is on clock 4, which allows the PCI compliant master to initiate its request on clock 6. The two clocks between the PCI transaction and the read data is caused because of potential contention on **TRDY#**. This can occur when the PCI compliant master is the core logic and the target is the A.G.P. compliant master.

3.6 Arbitration Signaling Rules

3.6.1 Introduction

This section describes the rules that the A.G.P. compliant master's **REQ#** signal and the A.G.P. compliant arbiter's **GNT#** signal need to follow for correct A.G.P. operation. These rules are a necessary part of the A.G.P. protocol.

The rules associated with the master's **REQ#** output signal provide an early indication to the A.G.P. compliant arbiter as to when an access request transaction will complete. The arbiter may take advantage of this to eliminate idle bus clocks between transactions.

The rules associated with the **GNT#** signal minimize the amount of read data buffering required in the master while allowing back-to-back 8 byte transactions without idle bus clocks. In order to achieve back-to-back

data transactions the arbiter may pipeline grants . The master must be able to accept them. Some of the rules in this section are necessary to limit the number of pipelined transactions that can be outstanding. This section will not attempt to describe the arbitration priority algorithms. This is a chipset specific implementation issue.

3.6.2 A.G.P. Compliant Master's **REQ#**

The A.G.P. compliant master asserts its **REQ#** signal when it wants to issue either a PCI cycle or enqueue requests using **PIPE#**. The master will de-assert **REQ#** depending on the type of access request. When issuing an A.G.P. access request over the **AD** bus using **PIPE#** the master must keep its corresponding **REQ#** asserted until one clock prior to de-asserting **PIPE#** asserted and **REQ#** deasserted on the same clock edge is an early indication that the current access request transaction is the last and **PIPE#** will deasserted one clock later. The arbiter may utilize this to avoid idle bus clocks when asserting **GNT#** for a subsequent transaction. This rule implies that **REQ#** will be deasserted for at least one clock between back-to-back **PIPE#** access request transactions. The master should concatenate as many address requests as possible into a single **PIPE#** access request transaction.

When an A.G.P. compliant master or a pure PCI compliant master issues a PCI transaction using **FRAME#** (and no other access requests are pending) it will de-assert **REQ#** when it asserts **FRAME#**. If another access request is pending the master will keep its **REQ#** asserted.

These rules are summarized in Table 3-11:

Next Access Request	Current Access Request: PCI using **FRAME#**	Current Access Request: A.G.P. using **PIPE#**
PCI	Keep **REQ#** asserted.	De-assert **REQ#** one clock prior to de-asserting **PIPE#**.
A.G.P.	Keep **REQ#** asserted	De-assert **REQ#** one clock prior to de-asserting **PIPE#**. Concatenate if possible.
None	De-assert **REQ#** when asserting **FRAME#**.	De-assert **REQ#** one clock prior to de-asserting **PIPE#**.

Table 3-11 A.G.P. Arbitration Rules

Figure 3-8 **Multiple Addresses Enqueued, Maximum Delay by Master** shows an access request using **PIPE#**. The master deasserts **REQ#** one clock prior to de-asserting **PIPE#**. Simultaneous SideBand and AD Access Request Generation is Not Allowed. An A.G.P. compliant master that is configured to issue commands over the sideband signals is not allowed to generate commands with **PIPE#** over the **AD** bus.

3.6.3 **GNT#** and **ST[2::0]**

The A.G.P. compliant arbiter will assert **GNT#** to initiate PCI or A.G.P. (non-sideband) activity. The **ST[2::0]** signals are only meaningful while **GNT#** is asserted and are used to communicate the type of PCI or A.G.P. activity being initiated. The **ST[2::0]** encodings are shown in Table 3-8 **A.G.P. Status Signals**.

3.6.4 **GNT#** for Single Transactions

For PCI and A.G.P. access requests **GNT#** will stay asserted until the arbiter samples either **FRAME#** or **PIPE#** asserted. The A.G.P. compliant master must drive **PIPE#** or **FRAME#** so that it is asserted either one or two clocks after the clock it samples **GNT#** asserted. Therefore **GNT#** will be asserted to an A.G.P. compliant master for a minimum of two clocks and a maximum of three clocks (for a single access request when the bus is idle). If the A.G.P. compliant master does not assert **PIPE#** or **FRAME#** from either the same clock that **GNT#** is first sampled asserted or the following clock, the arbiter may de-assert **GNT#** and consider the master inoperative. A pure PCI compliant master[15] on the A.G.P. bus may take longer to assert **FRAME#** after sampling its **GNT#** asserted. A pure PCI compliant master may be considered inoperative if it doesn't drive **FRAME#** within 16 idle bus clocks after **GNT#** is asserted.

For read and write data transfers, **GNT#** will be asserted along with the corresponding **ST[2::0]** signals for one clock cycle per transaction. This is summarized in Table 3-12.

A.G.P. Transaction Type	**GNT#** Duration
PCI Cycle	Until **FRAME#** sampled asserted
AD Access Request	Until **PIPE#** sampled asserted
Read Data	One 1x clock period per transaction
Write Data	One 1x clock period per transaction

Table 3-12 GNT# Duration

Figure 3-31 shows an A.G.P. compliant master asserting **REQ#** to run a PCI cycle. The master samples **GNT#** asserted on clock edge #4 with **ST[2::0]** encoding of '111' indicating permission to generate either a PCI cycle or an A.G.P. request. The master is allowed to take one or two clocks to assert **FRAME#**. In this example the master asserts **FRAME#** one clock after sampling **GNT#** asserted. Since no subsequent access request is pending the master de-asserts **REQ#** at the same time it asserts **FRAME#**. The arbiter samples **FRAME#** asserted on clock edge #3 and de-asserts **GNT#** clock 6. In this case **GNT#** is asserted for two clocks. If the master would have taken an additional clock to assert **FRAME#** the arbiter would have asserted **GNT#** for three clocks. Once the arbiter asserts **GNT#** (**ST**=111), the arbiter will continue driving it until either **PIPE#** or **FRAME#** are sampled asserted.

Figure 3-11 shows a read data transaction. The arbiter asserts **GNT#** for a single clock with an **ST[2::0]** encoding of '00x' indicating permission for the target to drive either high or low priority read data. Both the master and the target sample **GNT#** asserted on clock edge #2. The master must be ready to accept data on the next clock edge. The target is allowed to take one or two clocks to assert **TRDY#** and begin driving read

[15] A PCI 2.1 compliant master that does not generate A.G.P. transactions.

data. In this example the target asserts **TRDY#** and begins driving read data one clock after sampling **GNT#** asserted. **GNT#** is only asserted for one clock since this is a single read transaction consisting of four data phases.

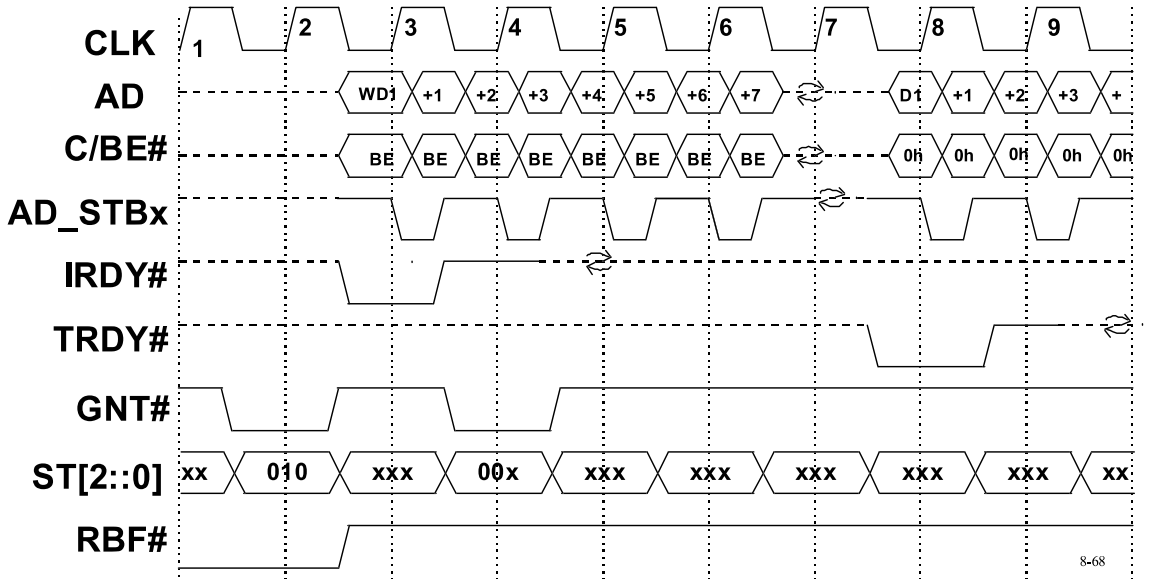


Figure 3-32 Write Data Followed by Read

Figure 3-32 shows a 32 byte write followed by a read. The arbiter asserts **GNT#** on clock edge #2 for a single clock with an **ST[2::0]** encoding of '010' indicating permission for the master to drive low priority write data. Both the master and the target sample **GNT#** asserted on clock edge #2. The target must be able to accept write data on the next clock. The master is allowed to take one or two clocks to assert **IRDY#** and begin driving write data. In this example the target asserts **IRDY#** and begins driving write data one clock after sampling **GNT#** asserted. **GNT#** is only asserted for one clock since this is a single write transaction consisting of eight data phases.

3.6.5 **GNT#** Pipelining

In order run back-to-back 8 byte data transactions (in 2x data transfer mode) without idle bus clocks between transactions the arbiter must pipeline **GNT#**s. The arbiter limits the number of outstanding **GNT#**s resulting from pipelining, to minimize the master's **GNT#** tracking logic. The master must be able to support the same number of outstanding pipelined **GNT#**s. The rules associated with attaining these goals are documented in this section.

When **GNT#** is pipelined the new bus driver is responsible for correctly sequencing from the current transaction to the next. If an idle bus clock is required between transactions to allow for bus turnaround, the new bus driver is responsible for guaranteeing the turn around bus clock.

- If **GNT#** is pipelined for an access request or for write data the master is responsible for correctly sequencing from the previous transaction to the next.

- When **GNT#** is pipelined for read data the target is responsible for correctly sequencing from the previous transaction to the next.

The rules governing the earliest point that **GNT#** may be pipelined for the next transaction is solely dependent on the current transaction type. If the current transaction is read data, the arbiter must wait to drive

GNT# for the next transaction such that **GNT#** is first sampled asserted on the last data phase of the current read. The last data phase is defined as the last rising 1x clock edge of the data transaction. This rule (along with proper use of the **RBF#** signal) minimizes the amount of low priority read data buffering required in the master. For a sequence of back-to-back 8 byte data transactions (in 2x data transfer mode) **GNT#** will be asserted on every 1x clock edge since, by definition, every 1x clock edge is the last data phase of a transaction.

If the current transaction is write data, **GNT#** for the next transaction can be asserted on the clock immediately following the **GNT#** for the current write data while there are less than **four** outstanding write data **GNT#s.** The arbiter tracks the number of outstanding write data **GNT#**s and will only assert a **GNT#** for a subsequent transaction if there are less than 4 outstanding write data **GNT#**s. The arbiter increments its write data **GNT#** counter when it asserts **GNT#** for write data and decrements the counter when it samples **IRDY#**[16] asserted by the master for a write data transaction. The *master must be able to handle five pipelined* **GNT#s** (this assumes that a master doesn't consider a **GNT#** "canceled" until the data transaction has finished, one request currently being handled and four more enqueued). This rule allows back-to-back 8 byte write data transactions to proceed when the master takes two clocks to assert the initial **IRDY#** after sampling **GNT#** asserted.

If the current transaction is a **PIPE#** request, **GNT#** for a data transaction can be asserted immediately following the **GNT#** for the current access request. Since **REQ#** will stay asserted (but doesn't indicate another request) until one clock prior to **PIPE#** de-assertion, it is impossible to pipeline a **GNT#** for another PCI or **PIPE#** access request if the current transaction is a **PIPE#** access request. Note that a **GNT#** for a **PIPE#** access request could immediately be followed by up to four **GNT#**s for write data transfers (or three writes and one additional transaction). The master's **GNT#** pipeline logic must be able to handle this case.

If the current transaction is a PCI cycle, **GNT#** for the next transaction can be asserted immediately following the **GNT#** for the current PCI cycle. Note that a **GNT#** for a PCI cycle could immediately be followed by up to four **GNT#**s for write data transfers (or three writes and one additional transaction). The master's **GNT#** pipeline logic must be able to handle this case. An A.G.P. pipelined transaction is not allowed to start (after a PCI transaction) until the bus is IDLE (**FRAME#** and **IRDY#** deasserted) for one clock. Table 3-13 entries refer to the earliest clock edge off which the arbiter can drive **GNT#** asserted for the next cycle.

[16] The count is 4 when a latched version of **IRDY#** is used to decrement the number of outstanding grants. Since the target could use either a latched or unlatched version the target is required to handle 4 outstanding pipelined transactions.

Next AD Activity \ Current AD Activity	PCI	A.G.P. Command	Read Data	Write Data
PCI or A.G.P. Command	**FRAME#** of current transaction sampled asserted.	**REQ#** sampled asserted after being deasserted.	2nd to last data phase of current transaction.	Immediately following **GNT#** for current write while <4 outstanding **GNT#**s.
Read Data	**FRAME#** of current transaction sampled asserted. (Depends on **RBF#**)	**PIPE#** of current transaction sampled asserted. (Depends on **RBF#**)	2nd to last data phase of current transaction to allow max. of 40 bytes of buffering in master. See section describing **RBF#** signal. (Depends on **RBF#**)	Immediately following **GNT#** for current write (Depends on **RBF#**) while <4 outstanding **GNT#**s.
Write Data	**FRAME#** of current transaction sampled asserted.	**PIPE#** of current transaction sampled asserted.	2nd to last data phase of current transaction.	Immediately following **GNT#** for current write while <4 outstanding **GNT#**s.

Table 3-13 Current/Next AD Activity

Figure 3-16 shows a sequence of back-to-back 8 byte read data transactions in 2x data transfer mode. The target samples **GNT#** asserted on clock edge #2 and responds by asserting **TRDY#** and driving read data (L6) on the following clock. The arbiter can assert the **GNT#** for the second read data transaction (H4) on clock edge #3 since that is the last data phase of the L6 read data transaction. **GNT#** is asserted on every clock edge so that an 8 byte read data transaction can occur on every clock edge.

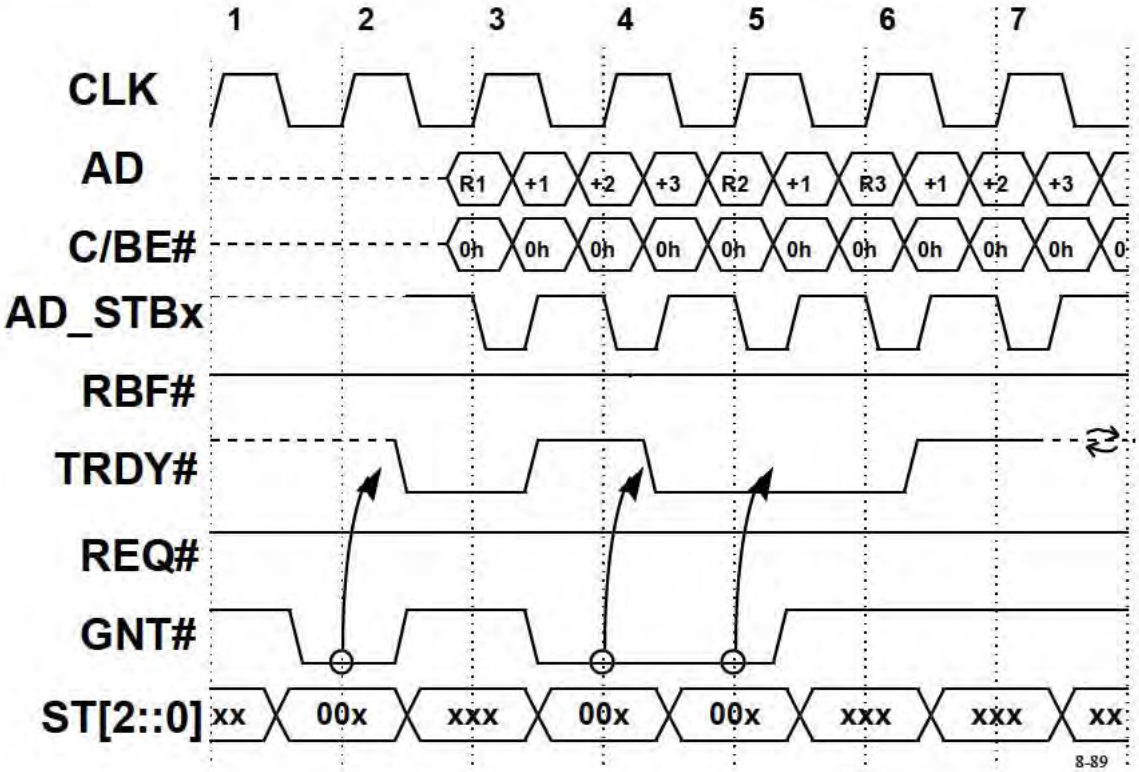


Figure 3-33 GNT# assertion for 16, 8, and then 16 byte Read Transfers

Figure 3-33 shows a sequence of 2x read data transactions. **GNT#** for the second read transaction (R2) is asserted on the clock edge #4 which is the last data phase of the R1 read transaction. **GNT#** for the third read transaction (R3) is asserted on clock edge #5 which is the last data phase of the R2 read transaction.

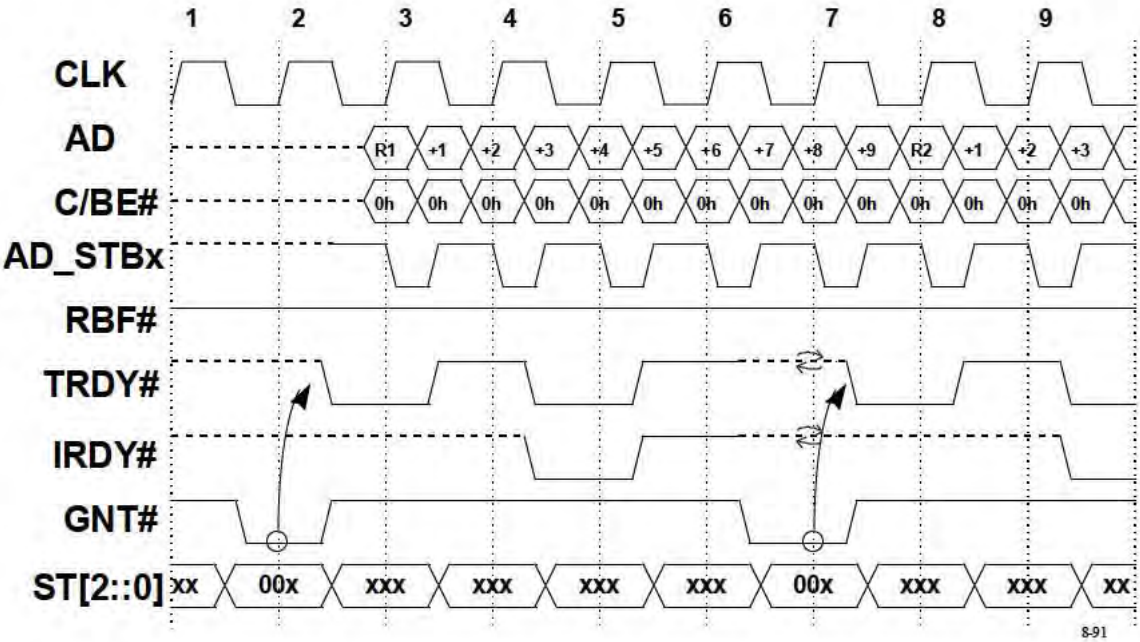


Figure 3-34 GNT# Assertion for next Read Data after long Data Transfer

Figure 3-34 shows a 40 byte read transaction followed by another read transaction in 2x data transfer mode. **GNT#** for the second read data transaction (R2) is asserted on clock edge #7 which is the last data phase of the R1 read transaction.

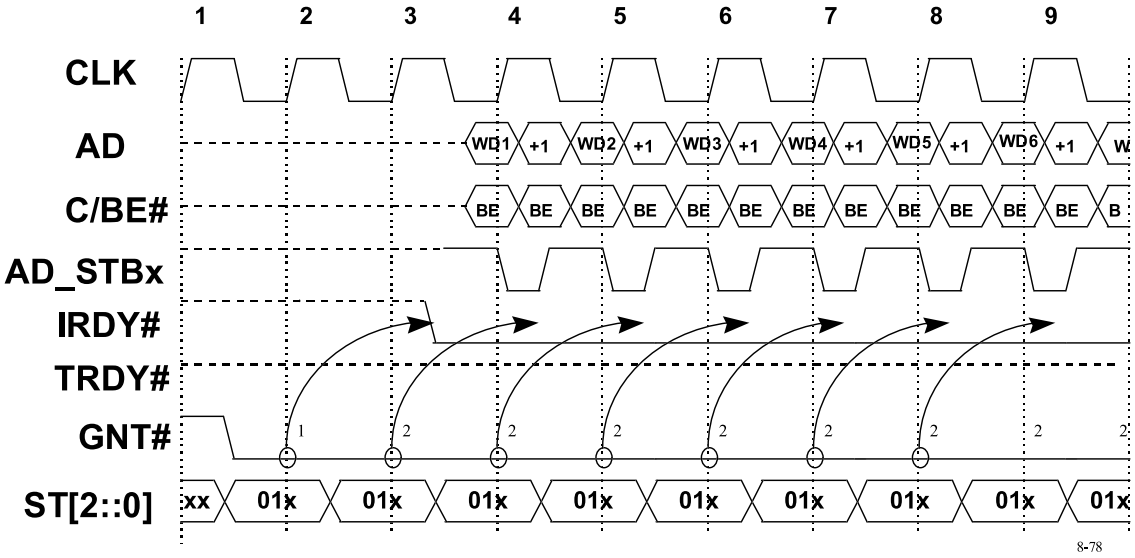


Figure 3-35 GNT# assertion for Back to Back Write Data Transfers

Figure 3-35 shows back-to-back 8 byte write data transactions in 2x data transfer mode. The following figures show that a maximum of 3 transaction are outstanding and will transfer data. The reason that it is only 3 and not 4 is that these diagrams assume that the arbiter is not using the latched version of **IRDY#**. When the latched version is used then all the number of grants outstanding are increased by one, since the arbiter delays the decrement. However, the arbiter can have 4 actually outstanding otherwise dead clocks can occur on the bus.

The master samples **GNT#** asserted on clock edge #2 and asserts **IRDY#** and drives write data W1 two clocks after sampling (clock edge #4). On clock edge #2 the arbiter increments its write **GNT#** counter to 1. Since the **GNT#** counter is less than three the arbiter asserts **GNT#** for write data W2 on clock edge #3 and the arbiter increments the write **GNT#** counter to 2. Since the **GNT#** counter is still less than three the arbiter asserts **GNT#** for write data W3 on clock edge #4. Even though **GNT#** is asserted on clock edge #4 the write **GNT#** counter does not increment since **IRDY#** for W1 is sampled asserted on clock edge #4. The arbiter continues asserting **GNT#** on every clock edge sustaining the back-to-back 8 byte transfers since the write **GNT#** counter is always less than three. In fact it is this waveform that established the need to allow up to three outstanding write **GNT#**s.

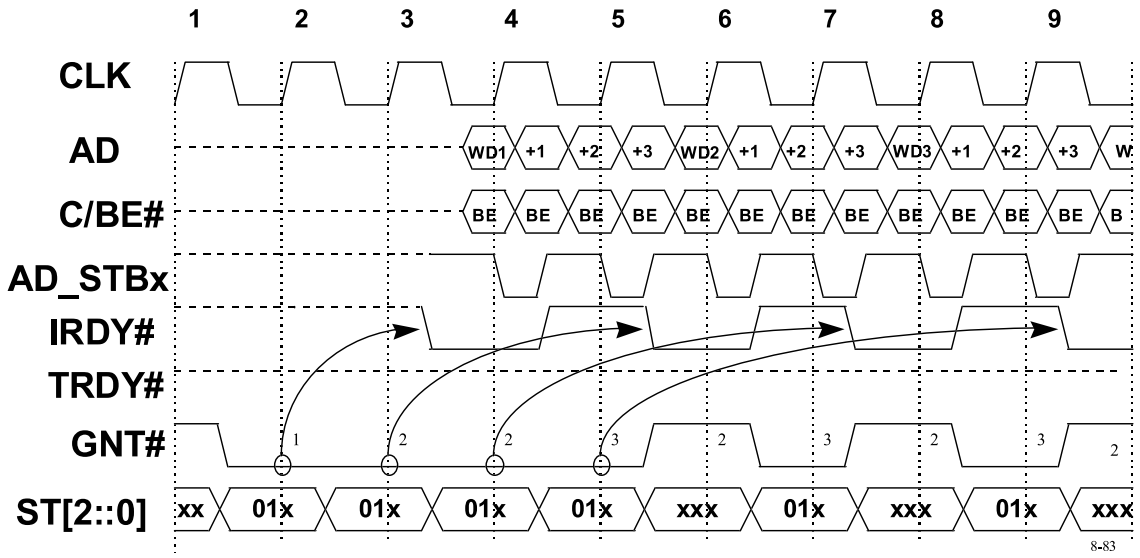


Figure 3-36 Back to Back GNT# with Delay on Initial Transfer

Figure 3-36 shows a sequence of 16 byte write data transaction in 2x data transfer mode. The master asserts **IRDY#** and drives write data W1 two clocks after sampling **GNT#** asserted on clock edge #2. On clock edge #2 the arbiter increments its write **GNT#** counter to 1. Since the **GNT#** counter is less than three the arbiter asserts **GNT#** for write data W2 on clock edge #3 and the arbiter increments the write **GNT#** counter to 2. Since the **GNT#** counter is still less than three the arbiter asserts **GNT#** for write data W3 on clock edge #4. Even though **GNT#** is asserted on clock edge #4 the write **GNT#** counter does not increment since **IRDY#** for W1 is sampled asserted on clock edge #4. Since the write **GNT#** counter is still less than three the arbiter asserts **GNT#** for write data W4 on clock edge #5. Since there is no **IRDY#** asserted on clock edge #5 the write **GNT#** counter increments to three and the arbiter is prohibited from asserting **GNT#** for W5 on clock edge #6. **IRDY#** for W2 is asserted on clock edge #6 decrementing the write **GNT#** counter to two. This allows the arbiter to assert **GNT#** for W5 on clock edge #7. This again increments the write **GNT#** counter to three and prohibits **GNT#** assertion for W6 on clock edge #8. Note that on clock edge #5 four **GNT#** have been pipelined to the master and the first transaction is still underway. This is the worst case scenario that the master's **GNT#** pipeline logic needs to account for.

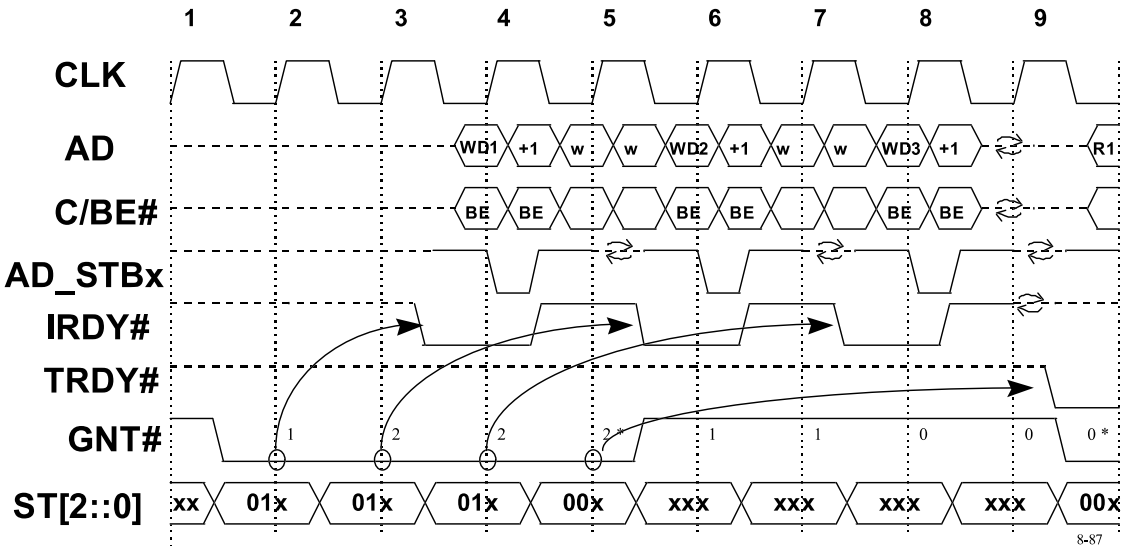


Figure 3-37 Pipelined GNT#s - Read and Writes (part 1)

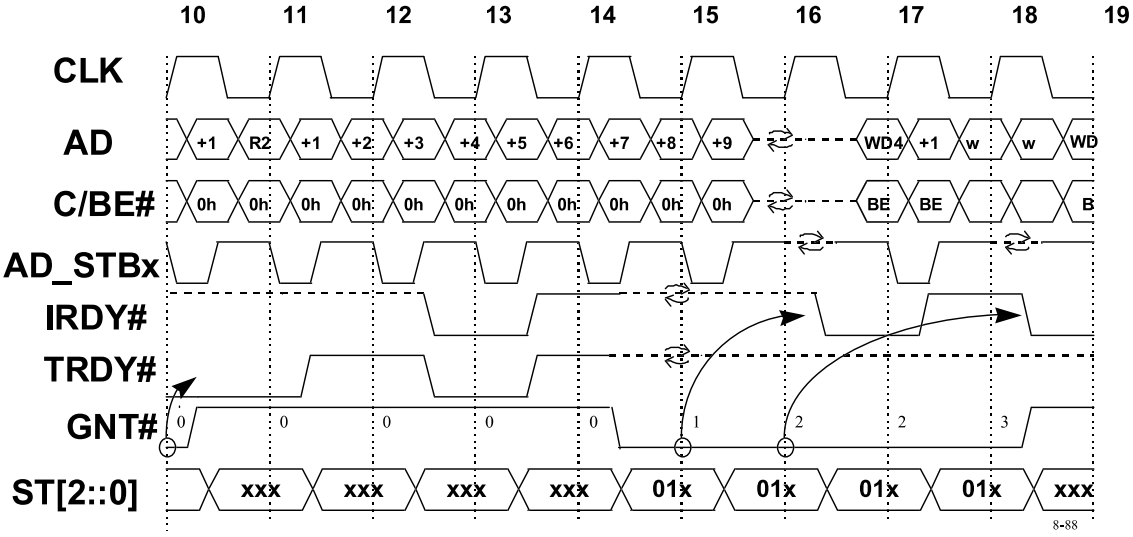


Figure 3-38 Pipelined GNT#s - Read and Writes (part 2)

Figure 3-37 shows the first half of a long sequence of write data transactions mixed with read data transactions. While Figure 3-38 shows the conclusion of the transaction. These need to be viewed as a single figure for the following discussion. The first three **GNT#**s are for write data transactions. The master inserts a wait state between the write data transactions. The **GNT#** asserted on clock edge #5 is for read data transaction R1. Note that the **GNT#** for R1 on clock edge #5 did not cause the write **GNT#** counter to increment from two to three. The write **GNT#** counter only increments for **GNT#**s associated with write data transactions. The arbiter de-asserts **GNT#** on clock edge #6 and waits to assert **GNT#** for read data R2 on clock edge #10 which is the last data phase of read data transaction R1. Note that by this time the write **GNT#** counter decremented to zero by sampling **IRDY#** asserted on clock edges #6 and #8. Note also that the write **GNT#** counter does not increment on clock edge #10 since the **GNT#** is for a read data transaction. The target is responsible for inserting the idle clock for bus turnaround between transactions W3 and R1. Read data transaction R2 is a 40 byte transaction so the next **GNT#** assertion is delayed by the arbiter until clock edge #15 which is the last data phase of R2. The **GNT#** on clock edges #15 is for write

data transaction W4. This causes the write **GNT#** counter to increment. The master is responsible for inserting the idle clock for bus turnaround between transactions R2 and W4. The arbiter asserts **GNT#** for W5, W6 and W7 on clock edges #16,17 and 18# respectively. The arbiter is prohibited from asserting **GNT#** on clock edge #19 for another transaction since the write **GNT#** counter is at three.

3.6.6 **GNT#** Interaction with **RBF#**

The A.G.P. compliant arbiter will not assert **GNT#** for a low priority read data transaction if the **RBF#** signal is asserted. In the case where **RBF#** is asserted on the same clock edge as **GNT#** is asserted, the master is required to accept that transaction. The arbiter must deassert **GNT#** immediately upon sampling **RBF#** asserted so that no further low priority read data transactions are signaled. **RBF#** only prohibits **GNT#** from being asserted for low priority read data transactions. **GNT#** assertion for high priority read data, write data, and access requests can still be generated even though **RBF#** is asserted.

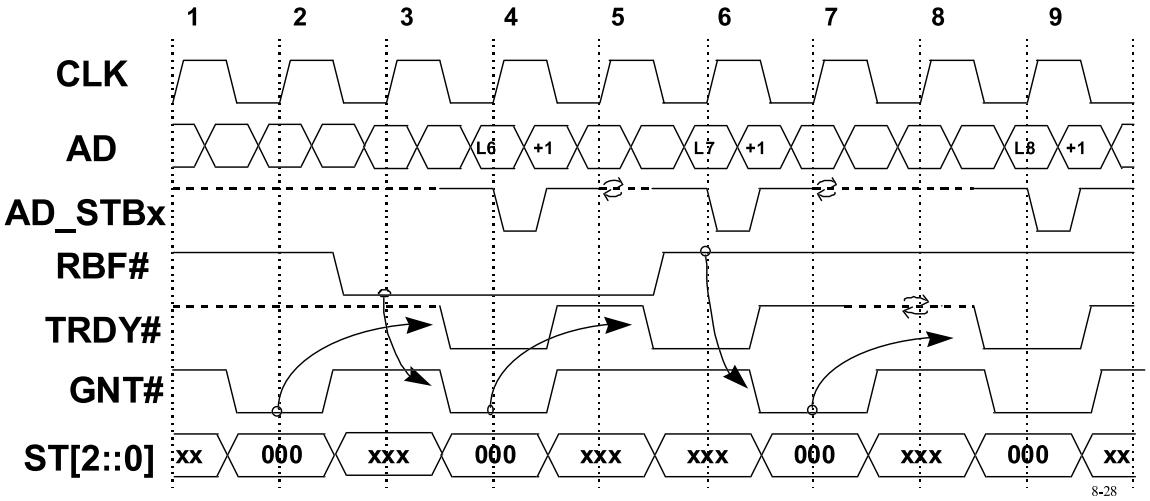


Figure 3-39 LP GNT# Pipelining Stopped While RBF# is Asserted

Figure 3-39 shows the master asserting the **RBF#** signal indicating that it can't accept further low priority read data. The master samples **GNT#** asserted on clock edge #2 with **ST[2::0]** indicating a low priority read data transaction. The master asserts **RBF#** on clock edge #3 because it doesn't have sufficient buffer space to take the next two low priority read transactions. The arbiter has already asserted **GNT#** on clock edge #3 which is the last data phase of L6. The master must accept the **GNT#** on clock edge #3 for read data transaction L7. The arbiter samples **RBF#** asserted on clock edge #3 and deasserts **GNT#** until it samples **RBF#** deasserted on clock edge # 5. Note that if the arbiter didn't deassert **GNT#** immediately upon sampling **RBF#** asserted on clock edge #3, then **GNT#** would be asserted on clock edge #4. This would increase the minimum amount of low priority read data buffering required in the master.

3.7 A.G.P. Sequencer State Machine Equations

This section will provide a concise statement of protocol rules to allow a designer to verify the correctness of his A.G.P. compliant bus sequencer. TBD.

3.7.1 Error Reporting

The error reporting philosophy of PCI applies to A.G.P. as well. This philosophy requires any device involved in the transfer or storage of permanent or retained system state (e.g., disk) to detect and report parity errors, and to compute parity. Devices involved with transient data only (e.g., graphics) are not so required. Since the sole purpose of the A.G.P. is the connection of video display devices, no provision has been made in the current spec for detecting or reporting of any bus errors. The current thinking is that if this becomes an issue in the future, bus parity detection and reporting will be added in essentially the same way as it is currently done on the PCI bus.

Admittedly this approach begs the question of address errors on writes to system memory (a problem that did not occur on PCI when graphics devices were targets only), as well as the possibility of data corruption on ancillary channels such as VBI. Nonetheless, this position seems to be congruent with current graphics accelerator practice, and therefore remains the current plan.

4. Electrical Specification

4.1 Overview

4.1.1 Introduction

As with the protocol enhancements, the A.G.P. electrical specification extends PCI to provide a much higher performance graphics interface. Most of the electrical interface requirements are based on the PCI spec, and this document will reference that spec wherever possible.

The A.G.P. physical interface is optimized for a point to point topology using a 3.3V signaling environment. The baseline performance level utilizes a 66 MHz clock, to provide a peak bandwidth of 266MB/sec.

A.G.P. includes an option for higher performance levels, providing peak bandwidths of 533MB/sec[17]. This optional mode uses a double-clocked data technique to transfer twice the data per each A.G.P. clock. This A.G.P. mode, referred to as *2X transfer mode (*also *2X mode* or *A.G.P.-133*[18]*)*, requires additional interface timing strobes and different signal timings from the 1X mode. Components supporting the 2X transfer mode must also support the 1X mode. 2X mode requirements are a superset of the 1X mode timings.

The next section establishes a conceptual framework by which to understand the two modes.

4.1.2 1X Transfer Mode Operation

The 1X mode, 66 MHz A.G.P. interface can be designed using common I/O buffer technology. Since A.G.P. is a point to point interface, adjustments have been made in the system timing budgets that relax component input requirements relative to PCI.

Conceptually, the 1X mode A.G.P. operation is similar to PCI. All timings are referenced to a single clock, the A.G.P. clock.

4.1.3 2X Transfer Mode Operation

The A.G.P. protocol provides Qword access granularity. Qword transfers normally take two clock cycles in the 1X mode. Likewise, sideband address commands normally take two clocks per each 16-bit command. The 2X transfer mode provides a mechanism for doubling the data transfer rate of the **AD**, **C/BE#** and **SBA** signals[19]. With 2X transfer, Qword transfers only require one clock cycle, and sideband commands only require one clock per 16-bit command.

---

[17] 533MB/sec results from the actual minimum clock period of 15ns.

[18] "133" connotes a 133MHz peak transfer rate.

[19] The clock mode of these signals is controlled as a unified group; independent control of the clocking mode for sub-groups is not provided.

The 2X transfer mode is implemented as a timing layer *below* the 1X protocol's flow control mechanisms. This timing layer, referred to as the *inner loop*, specifies timing relationships for the reliable transfer of data from the output latches at the transmitting device to the input latches at the receiving device. The logical protocol mechanisms operate above this layer, via an *outer loop*, to control the actual transfer of data between the data queues. A model showing these various time domains is shown in Figure 4-1.

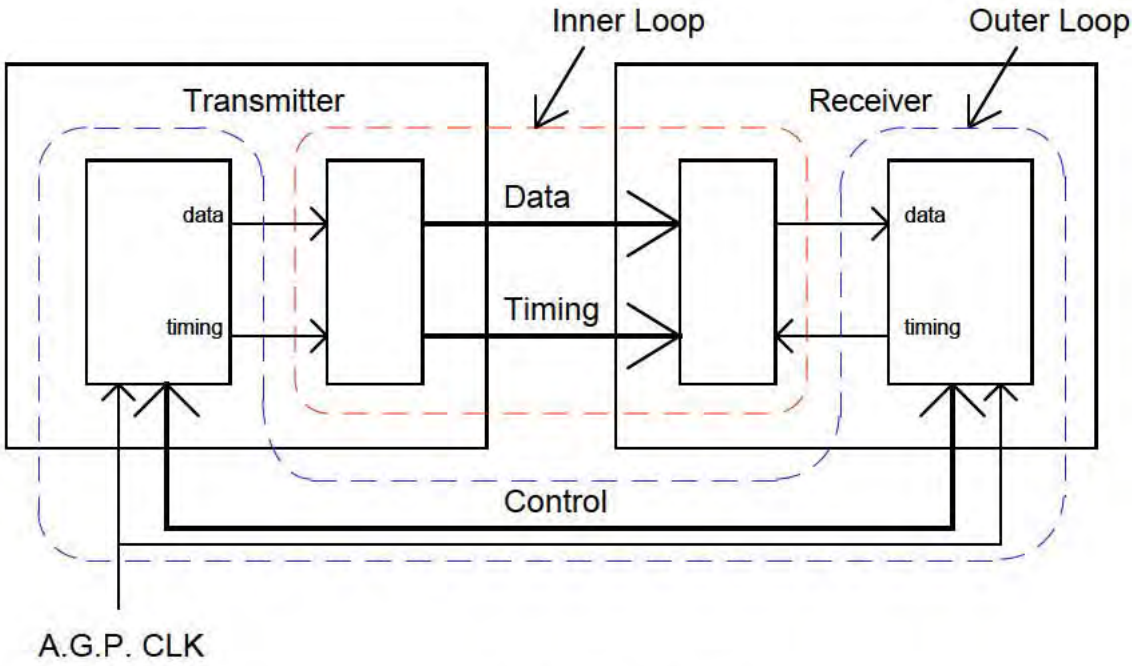


Figure 4-1 2X Mode Time Domains

The outer loop of both devices operates from a common A.G.P. clock, and outer loop controls are specified relative to this clock (as in the 1X operation mode). The inner loop timings use additional source synchronous timing signals to realize the data transfer. Note that the only clock defined in the system is the A.G.P. clock.

Source timed strobes, where the device sourcing the data also sources a timing signal for use by the receiver, are used since data transport delays are nulled out at the receiver. These source synchronous strobes are derived from the common A.G.P. clock at the transmitter, placed at the center of the output data valid window, and used by the receiver to directly capture data at the interface.

The inner loop and outer loop timing dependencies are defined by a set of relationships between the strobes and the A.G.P. clock. The relationship allows for a *deterministic* transfer of data between the inner and outer loops. These timing dependencies are specified in such a way as to allow implementation flexibility at the receiver. A tradeoff can be made between the latency through the inner loop and the implementation technology and/or design complexity. Refer to the A.G.P. Design Guide for more information.

The timing model presented above contains four different time domains, to be detailed in the following sections:

- Transmit/Receive Outer Loop
- Transmit to Receive Inner loop

- Transmit Outer to Inner Loop
- Receive Inner to Outer Loop

4.1.3.1 Transmit/Receive Outer Loop

The outer loop between the devices uses the 1X mode A.G.P. timings for bi-directional control information transfer between the transmitter and receiver.

4.1.3.2 Transmit to Receive Inner loop

Transfer of data[20] between the transmit and receive inner loop circuits is accomplished using a timing strobe sent from the transmitter to the receiver, with a set of simple timing relationships between the data and strobe. Both edges of the strobe are used to transfer data, with the first half of data corresponding to the falling edge of the strobe, and the second half corresponding to the rising edge.

The transmit strobe edges are positioned near the center of the minimum data valid window, to give the receiver a good input data sampling window for all the various system timing skew cases. A minimum data valid *before* strobe edge (tDvb), and a minimum data valid *after* strobe edge (tDva) are specified. The transmit strobe/data timings are shown in Figure 4-2.

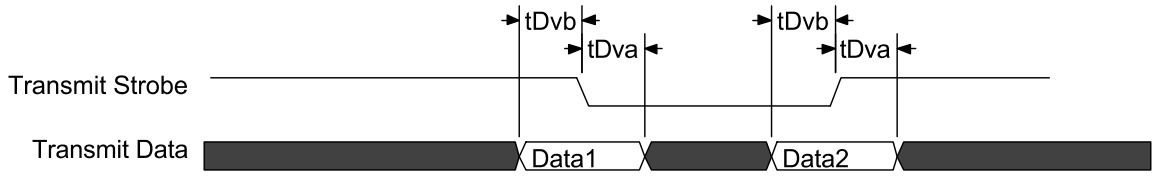


Figure 4-2 Transmit Strobe/Data Timings

The receive strobe input is directly used to capture data into the device. Again, *both* edges of the strobe are used to capture data, since new data is valid on each edge. A minimum setup (tDsu) and hold time (tDh) relative to the strobe edges is therefore required, as shown in Figure 4-3.

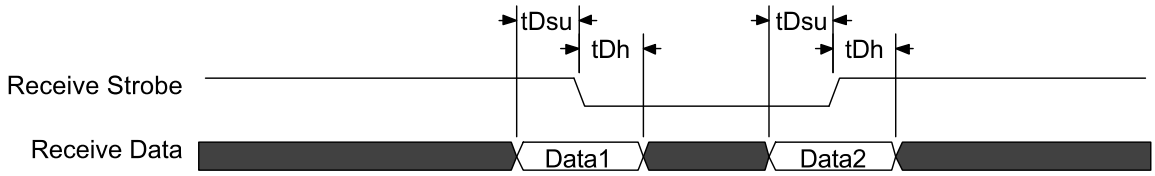


Figure 4-3 Receive Strobe/Data Timings

[20] *Data* refers to any of the 2X capable signal groups: AD[31:0], C/BE[3:0]# or SBA[7:0].

4.1.3.3 Transmit Outer to Inner Loop

The next timing relationship to understand is the relationship between the outer loop and inner loop at the transmitter. These timing specs are needed to create a deterministic data relationship between the inner loop transfer and the related outer loop flow control events (e.g. **IRDY#**). The relationship is specified by relating the output strobe to the A.G.P. clock, as shown in Figure 4-4. Note that two clock periods, T1 and T2, are shown since the strobe pulse is permitted to cross the A.G.P. clock boundary.

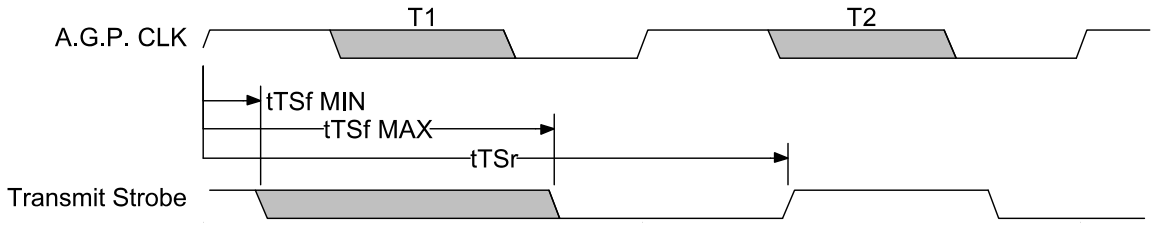


Figure 4-4 Transmit Strobe/Clock Timings

To guarantee a deterministic relationship between inner loop data transfer and the corresponding outer-loop flow control, the strobe's falling edge is required to occur within the T1 clock period, as seen at the receiver. This requirement dictates a min *and* max spec for the clock to strobe falling edge (tTSf). The clock to strobe rising edge only requires a max spec (tTSr). Actually, all the transmit strobe specs are driven by receiver requirements and system skews. The receiver requirements will be discussed in more detail in the next section.

Figure 4-5 shows the composite inner and outer loop timing relationships for the transmitter.

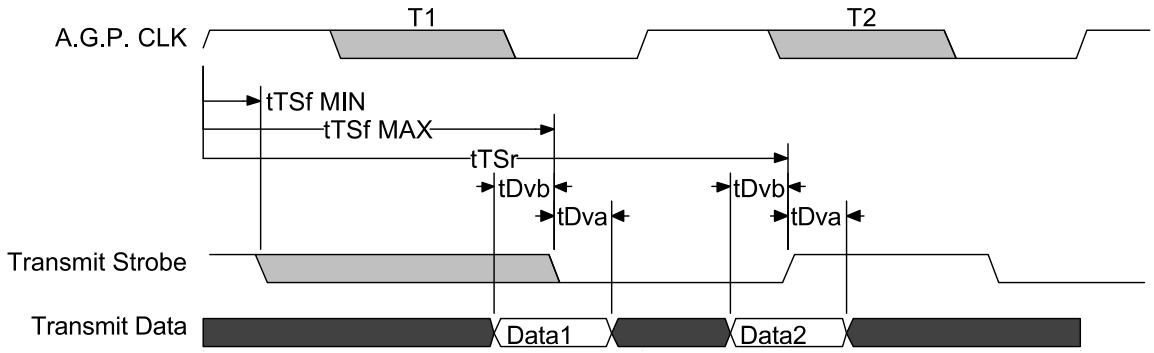


Figure 4-5 Composite Transmit Timings

4.1.3.4 Receive Inner to Outer Loop

The last, and most complex, set of timings to understand is the receiver inner to outer loop relationships. To better understand these timings, a model of the inner to outer loop transfer interface is needed. Refer to the receive transfer timing in Figure 4-6.

In the case of the **AD** bus interface, after the rising edge of the receive strobe, a Qword of valid data will be available. Data will be transferred from the inner loop to the outer loop based on a A.G.P. clock event. The requirement is to define a circuit to reliably affect this transfer for all system conditions.

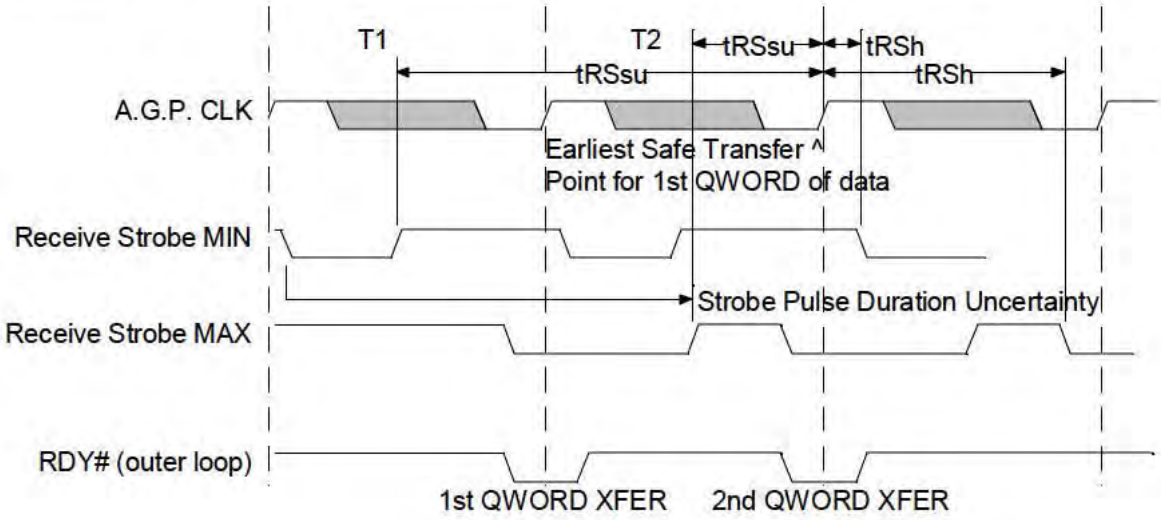


Figure 4-6 Receiver Inner to Outer Loop Transfer Timing

Figure 4-6 depicts the possible minimum and maximum strobe relationships at the receiver. Two consecutive data transfers occur, with the first one starting in T1. In the minimum strobe delay case, both strobe edges occur in T1, and in the maximum strobe delay case the second edge does not occur until T2. Thus, there is an uncertainty window for the strobe duration which can cross the clock boundary.

Due to this uncertainty window, the earliest *safe* transfer point from the inner loop to the outer loop occurs at the end of T2. Notice that in the min case a second set of strobes occurs in T2, since a second QWORD of data is being transferred. Therefore, to prevent data from being overwritten before the safe transfer point, at least *two* stages of edge-triggered latches must exist in the inner loop input circuitry.

The latched data will be transferred to the outer loop block based on the A.G.P. clock. The inner loop latched data must be guaranteed to remain stable at the point of transfer. The minimum setup spec on the receive strobe to clock (tRSsu) exists to ensure that data from the output of an inner loop latch has a defined setup time to the input of the outer loop's latch. Likewise, the minimum hold spec on the strobe (tRSh) exists to ensure that data from the output of an inner loop latch has a defined hold time relative to the input of the outer loop's latch.

Note that tRSsu and tRSh are associated with the strobe edge for a particular T1 cycle data transfer. Due to the minimum to maximum strobe variation, it is possible for tRSsu to extend *across* a clock edge into the T1 cycle. Also, the tRSh hold time is provided for data transfer from the *second* previous rising strobe event, not the immediately prior one.

The receive strobe spec values were chosen to allow most implementations to transfer the data at the *earliest* safe point, at the end of T2. Note that the *actual* transfer point is not specified, only the earliest viable point. An implementation may elect to increase the effective setup time by additional pipeline delay stages. Refer to the A.G.P. Design Guide for more details.

The composite receive timings are shown in Figure 4-7.

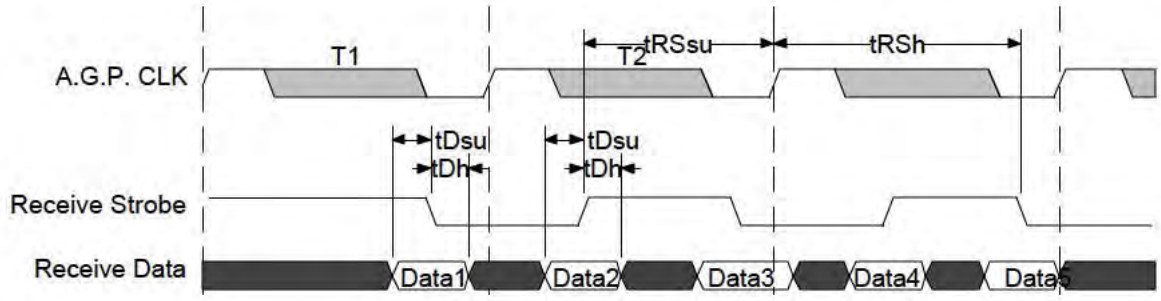


Figure 4-7 Composite Receive Timings

4.1.3.5 SB_STB Synchronization

The 2X inner loop receiver timing specifications ensure reliable transfer between the time domains. However, to affect such transfer implicit knowledge of the current clock cycle (T1 or T2) is needed, to determine which data to transfer to the outer loop. The **AD** bus outer loop protocol directly provides the required information, as each **xRDY#** event signals a new T1. However, the sideband address port, **SBA**, has no such inherent synchronization information.[21] To allow for the receiver circuitry to unambiguously synchronize with the correct clock event, an additional synchronization protocol is defined for the sideband port.

Note that this synchronization protocol has no relationship to the normal command protocol defined for the **SBA** signals. The A.G.P. compliant target protocol sequencer should be designed to be unaffected by this synchronization protocol. In other words, commands over the **SBA** port are undefined until synchronization has occurred.

Whenever the **SB_STB** signal has been idle, including after a reset, prior to sending any sideband commands the master must transmit a special synchronization cycle. This sync cycle is used by the receiver to determine proper phasing information (T1 or T2) at the interface.

The sync cycle is defined as **SBA[7::0]** being driven to FEh for a single 1X clock cycle, with timings adhering to the 1X mode requirements. Following the sync cycle, the **SBA** port state is undefined relative to the 1X clock, and is only valid at the first valid 2X command strobe point. This first valid 2X command strobe point occurs *exactly* two clocks after the sync cycle event is sampled by the target (cycle T1 in Figure 4-8), at which point 2X timing operation has commenced. The first valid command[22] occurs on this first T1 cycle.

The 2X timing operation must continue, unless **SB_STB** is stopped. Prior to stopping the **SB_STB** signal, a minimum of *four* clock cycles of NOPs must be transmitted. (Note this is equivalent to four 16-bit NOP commands in 2X mode). If stopped, **SB_STB** must be driven high and either held high or tristated. Once stopped, **SB_STB** must remain stopped for a *minimum* of *eight* clock cycles prior to any new sync event. The **SB_STB** synchronization protocol is shown below.

---

[21] The strobe to clock AC timings specs do not permit a synchronization based on sampling the strobe by the 1X clock, they only allow for deterministic data transfer from strobe-based data latches to 1X-clock based latches.

[22] The first command sent may be any of the defined sideband address commands, including a NOP.

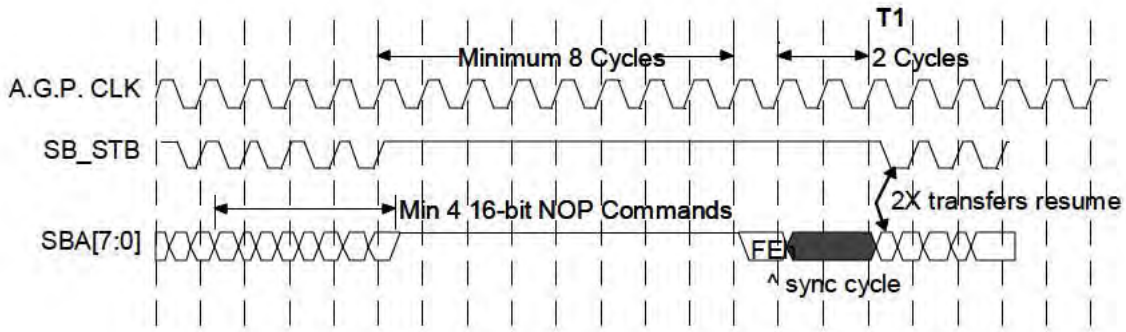


Figure 4-8 SB_STB Synchronization Protocol

4.2 Component Specification

This chapter defines the electrical characteristics of A.G.P. interface. Most of the specifications are focused on the A.G.P. modes, and will not repeat all requirements defined in the PCI Rev. 2.1 Specification.

I/O buffer design technology to meet these requirements is beyond the scope of this specification. Some general guidelines may be found in the A.G.P. Design Guide.

4.2.1 DC SPECIFICATIONS

The A.G.P. interface is optimized for a 3.3V operating environment and is based on PCI 3.3V signaling, with changes to allow higher data transfer rates. DC parameters differing from PCI 66 specifications are highlighted in bold in the table below.

4.2.1.1 A.G.P. 1X Mode DC Specification

Table 4-1: DC Specifications for A.G.P. 1X Signaling

Symbol	Parameter	Condition	Min	Max	Units	Notes
Vddq	**I/O Supply Voltage**		**3.0**	**3.6**	**V**	**1**
V_{ih}	Input High Voltage		0.5Vddq	Vddq+0.5	V	
V_{il}	Input Low Voltage		-0.5	0.3Vddq	V	
Vipu	Input Pull-up Voltage		.7Vddq			2
I_{ih}	Input High Leakage Current	V_{in} = 2.7		70	µA	3
I_{il}	Input Leakage Current	0 < V_{in} < Vddq		±10	µA	3
V_{oh}	Output High Voltage	I_{out} = -500 mA	.9Vddq		V	
V_{ol}	Output Low Voltage	I_{out} = 1500uA		.1Vddq	V	
C_{in}	**Input Pin Capacitance**			**8**	**pF**	**4**
C_{clk}	CLK Pin Capacitance		5	12	pF	

- NOTES:
- 1. Vddq only specifies the voltage at the A.G.P. interface; component supply voltages are independent of Vddq. Component I/O buffers must be powered only from Vddq, and isolated from any other power supplies on the component. Vddq for both A.G.P. compliant master and A.G.P. compliant target must be driven from the same power rail.
- 2. This specification should be guaranteed by design. It is the minimum voltage to which pull-up resistors are calculated to pull a floated network. Applications sensitive to static power utilization should ensure that the input buffer is conducting minimum current at this input voltage.
- 3. Input leakage currents include hi-Z output leakage for all bi-directional buffers with tri-state outputs.
- 4. Absolute maximum pin capacitance for an A.G.P. input is 8 pF (except for **CLK**) with an exception granted to motherboard-only devices, which could be up to 16 pF, in order to accommodate PGA packaging. This would mean, in general, that components for expansion boards would need to use alternatives to ceramic PGA packaging (i.e., PQFP, BGA, etc.).

4.2.1.2 A.G.P. 2X Mode DC Specification

The parameters below are the incremental requirements for the A.G.P. 2X mode interface. Note that the primary change is the addition of a voltage reference, which allows for a differential input buffer with common reference voltage. Implementation of a differential input buffer is not a requirement if alternate design approaches can be used to meet all other requirements.

Table 4-2: DC Specifications for A.G.P. 2X Signaling

Symbol	Parameter	Condition	Min	Max	Units	Notes
Vref	Input reference voltage		0.39Vddq	0.41Vddq	V	1,2
Iref	Vref pin input current	$0 < V_{in} < Vddq$		+10	uA	2
VIH	Input High Voltage		Vref+0.2		V	
VIL	Input Low Voltage			Vref-0.2	V	
Cin	Input Pin Capacitance			8	pf	
Δ Cin	Strobe to data Pin Capacitance delta		-1	2	pf	3

Notes:

1. A.G.P. allows differential input receivers to achieve the tighter timing tolerances needed for 133MT/s. Nominal value of Vref is 0.4Vddq which can be designed with 2% resistor to achieve the specified min and max values. The value of Vref is intended to specify near the center point of the VIL/VIH range. For example, at nominal Vddq (3.3V), Vref is 1.32V +/- 2.5%. A single input interface buffer can be designed to meet the VIL/VIH levels of both the A.G.P. and PCI specifications. As in other A.G.P. specifications, note that the Vddq references the I/O ring supply voltage, and not the component supply.

2. A differential input buffer is not a required implementation, as long as all other specifications are met. However component designs requiring a reference are required to adhere to the Vref and Iref specifications, to facilitate a common reference circuit for motherboard-only A.G.P. designs. (A common reference circuit is not applicable to add-in card designs, since Vref is not supplied via the connector.)

3. Delta Cin is required to restrict timing variations resulting from differences in input pin capacitance between the strobe and associated data pins. This delta only applies between signal groups and their associated strobes: **AD_STB1=>AD[31::16]** & **C/BE[3::2]#**; **AD_STB0=>AD[15::0]** & **C/BE[1::0]#**; **SB_STB=>SBA[7::0].**

4.2.2 AC Timings

The A.G.P. timings are specified by two sets of parameters, one corresponding to the A.G.P. 1X operation, and the second for the optional A.G.P. 2X transfer mode operation. The A.G.P. 2X specs are in *addition* to the A.G.P. 1X specs. The A.G.P. 1X specs still apply to all outer-loop control signals during A.G.P. 2X operation.

4.2.2.1 A.G.P. 1X Timing Parameters

Table 4-3: A.G.P. 1X AC Timing Parameters

Symbol	Parameter	Min	Max	Units	Notes
Clock:					
t_{CYC}	CLK cycle time	15	30	ns	1
t_{HIGH}	CLK high time	6		ns	
t_{LOW}	CLK low time	6		ns	
-	CLK slew rate	1.5	4	V/ns	2
t_{LOCK}	PLL Lock Time		1000	us	3
Transmitter Output Signals:					
t_{VAL}	**CLK to control signal and Data valid delay**	**1.5**	**6**	**ns**	**4**
t_{ON}	**Float to Active Delay**	**1.5**	**6**	**ns**	
t_{OFF}	**Active to Float Delay**	**1**	**14**	**ns**	
	Output slew rate	1.5	4	V/ns	2
Receiver Input Signals:					
t_{SU}	**Control signals setup time to CLK**	**5**		**ns**	**4**
t_{H}	**Control signals hold time to CLK**	**0.5**		**ns**	**4**
Reset Signal:					
t_{RST}	Reset active time after power stable	1		ms	
$t_{RST-CLK}$	Reset active time after **CLK** stable	100		us	
$t_{RST-OFF}$	Reset active to output float delay		40	ns	
---	**RST#** Slew Rate	50	n/a	mV/ns	5

NOTES:

1. In general, A.G.P. compliant devices must work with any stable clock frequency from 33MHz to 66 MHz. Changes in the clock frequency are allowed for mobile applications, however only a change between the normal operating state and a clock stopped state are supported. The clock may only be stopped in a low state. Devices allowing for clock stop operation must be static designs, and maintain all software visible configuration information during the clock stop state. Also, during clock stop operation, devices designed for mobile applications must control signal state as specified in the forthcoming *PCI Power Management Specification* to minimize system power dissipation. The state of all new A.G.P. signals should be handled identically to existing PCI signals of the same output type (e.g. S/T/S are floated by component, pulled-up to Vddq by central resource).

2. Rise and fall times are specified in terms of the edge rate measured in V/ns. This slew rate must be met across the minimum peak-to-peak portion of the clock waveform as shown in figure 4-7 of the PCI spec.

3. Mobile A.G.P. systems can stop the A.G.P. clock during normal operation. PLLs on mobile components must relock within the specified time from a stable A.G.P. clock. Mobile A.G.P. compliant devices must allow for the clock to stop without any loss of data or configuration information. When the clock is restarted, a bus master can not issue a new transaction or bus request until the tLOCK time period has been met.

4. In A.G.P. 2X mode, tVAL, tDsu and tDh values for the **AD**, **C/BE#**, and **SBA** signals are *superseded* by the A.G.P.-2X parameters tDvb, tDva, tDsu, tDh.

5. The minimum **RST#** slew rate applies only to the rising (deassertion) edge of the reset signal, and ensures that system noise cannot render an otherwise monotonic signal to appear to bounce in the switching range.

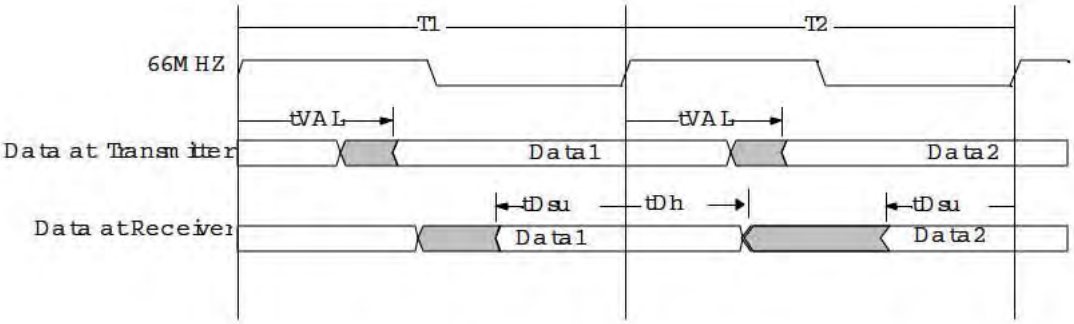


Figure 4-9 A.G.P. 1X Timing Diagram

4.2.2.2 A.G.P. 2X AC Timing Parameters

The parameters below apply only to the inner loop 2X transfer mode signals (**AD**, **C/BE#**, **SBA**) during 2X operation. The data specs below *replace* the corresponding data specs from the A.G.P. 1X table for these signals. The A.G.P. 1X parameters apply to all other signal operation.

Table 4-4 A.G.P. 2X AC Timing Parameters

Symbol	Parameter	Min	Max	Units	Notes
Transmitter Output Signals:					
t_{TSf}	**CLK** to transmit strobe falling	2	12	ns	
t_{TSr}	**CLK** to transmit strobe rising		20	ns	
t_{Dvb}	Data valid before strobe	1.7		ns	
t_{Dva}	Data valid after strobe	1.7		ns	
t_{ONd}	**Float to Active Delay**	**-1**	**9**	**ns**	
t_{OFFd}	**Active to Float Delay**	**1**	**12**	**ns**	
t_{ONS}	Strobe active to strobe falling edge setup	6	10	ns	
t_{OFFS}	Strobe rising edge to strobe float delay	6	10	ns	
Receiver Input Signals:					
t_{RSsu}	Receive strobe setup time to **CLK**	6		ns	
t_{RSh}	Receive strobe hold time hold time from **CLK**	1			
t_{Dsu}	Data to strobe setup time	1		ns	
t_{Dh}	Strobe to data hold time	1		ns	

NOTES:

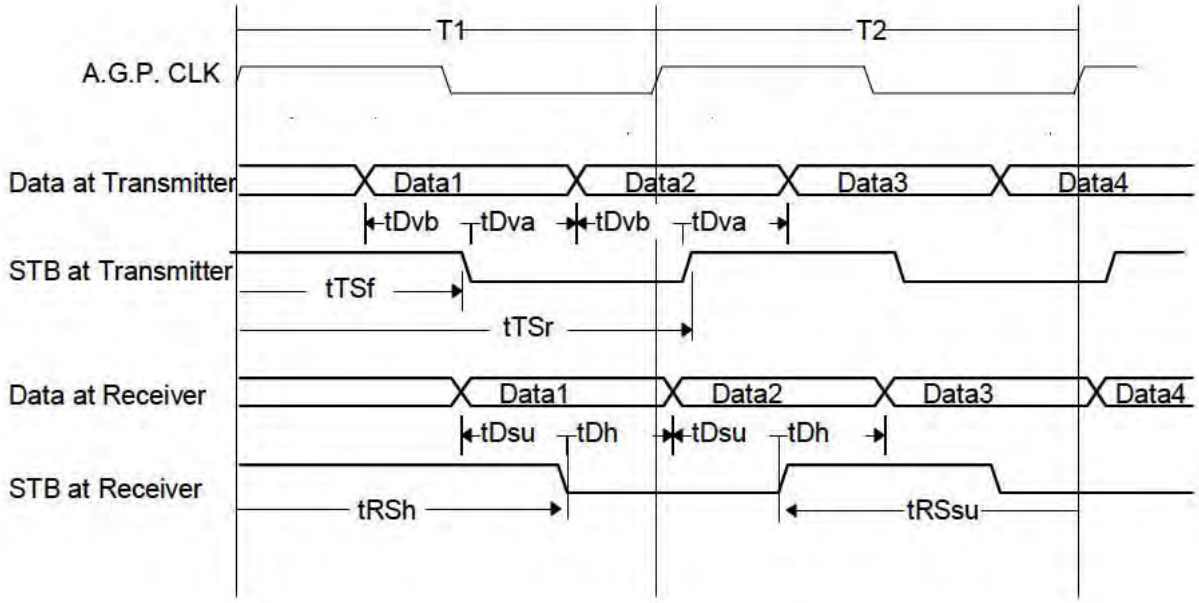


Figure 4-10 A.G.P. 133 Timing Diagram

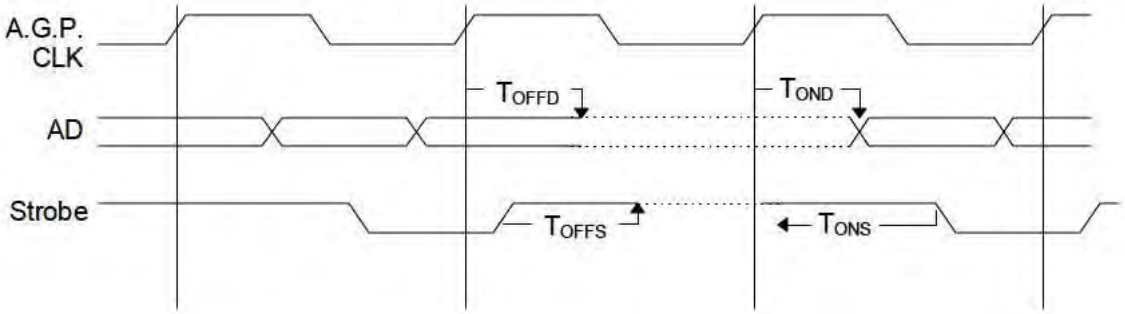


Figure 4-11 Strobe/Data Turnaround Timings

4.2.2.3 Measurement and Test Conditions

Unless otherwise specified, the reference point for all AC timings measurements is 0.4VddqThe output loading is 10pf

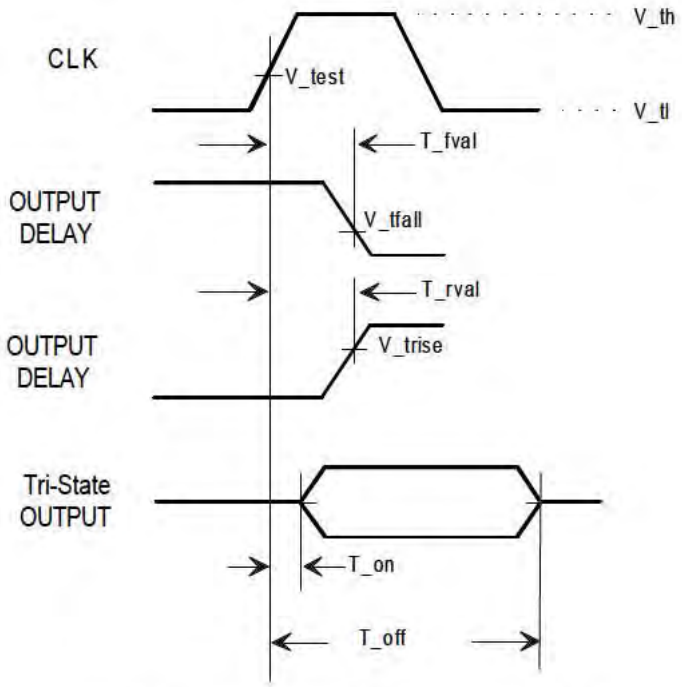


Figure 4-12 Output Timing Measurement Conditions

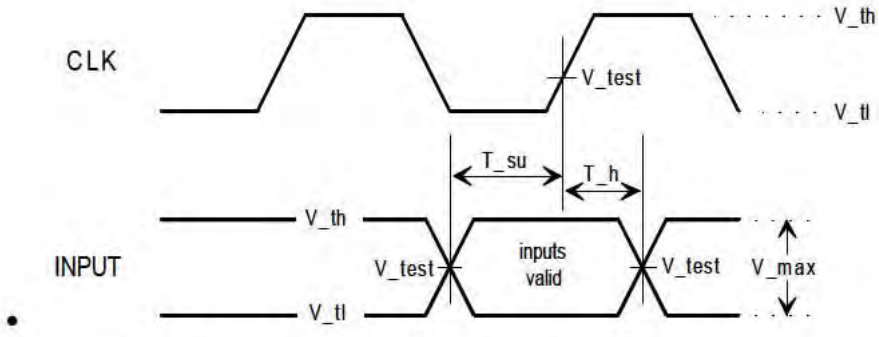


Figure 4-13 Input Timing Measurement Conditions

Table 4-5 Measurement and Test Condition Parameters

Symbol	3.3V Signaling	Units	Notes
V_{th}	0.6Vddq	V	1
V_{tl}	0.2Vddq	V	1
V_{test}	0.4Vddq	V	
V_{trise}	0.285Vddq	V	2
V_{tfall}	0.615Vddq	V	2
V_{max}	0.4Vddq	V	1
Input Signal Slew Rate	1.5-4.0	V/ns	3

NOTES:

- 1. The test for the 3.3V environment is done with 0.1*Vddq of overdrive. V_{max} specifies the maximum peak-to-peak waveform allowed for testing input timing.
- 2. V_{trise} and V_{tfall} are reference voltages for timing measurements only. Developers need to design buffers that launch enough energy into a 50Ω transmission line so that correct input levels are guaranteed after the first reflection.
- 3. Outputs will be characterized and measured at the package pin with the load shown in Figure 4-16. Input signal slew rate will be measured between 0.3Vddq and 0.6Vddq.

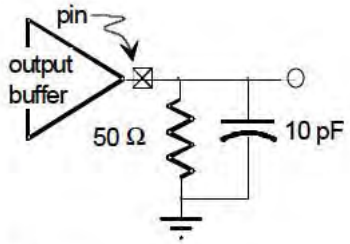


Figure 4-14 T_{val}(max) Rising Edge

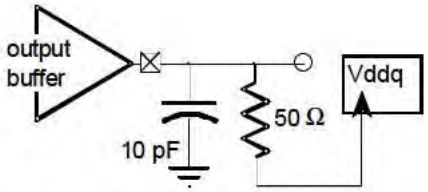


Figure 4-15 T_{val}(max) Falling Edge

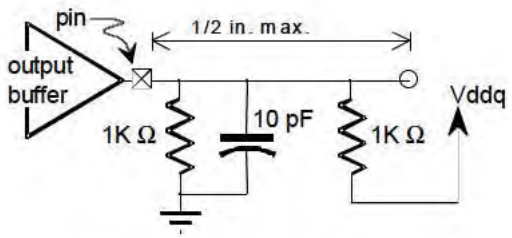


Figure 4-16 T_{val} (min) and Slew Rate

4.2.3 Signal Integrity Requirement

Table 4-6 Signal Integrity Requirements[1]

		A.G.P. 1X		A.G.P. 2X			
Symbol	**Parameter**	**Min**	**Max**	**Min**	**Max**	**Units**	**Notes**
-	Output Slew Rate	1.5	4	1.5	4	V/ns	2,3
Tset	Output settling time to +/- 10% of rail				7	nS	

NOTES:

- 1. Output buffer (OB) loading conditions under which measurements are made: 1) the OB driving a 6 inch transmission line with a characteristic impedance range from 65Ω ±15Ω, 2) one CMOS type input loading attached on the other side of an ideal transmission line
- 2. As measured at the receiver input
- 3. For mobile or other system designs without a metal enclosure, to minimize EMI problems it is recommended that the output slew rate not exceed 2.5V/ns.

4.2.4 Driver Characteristics

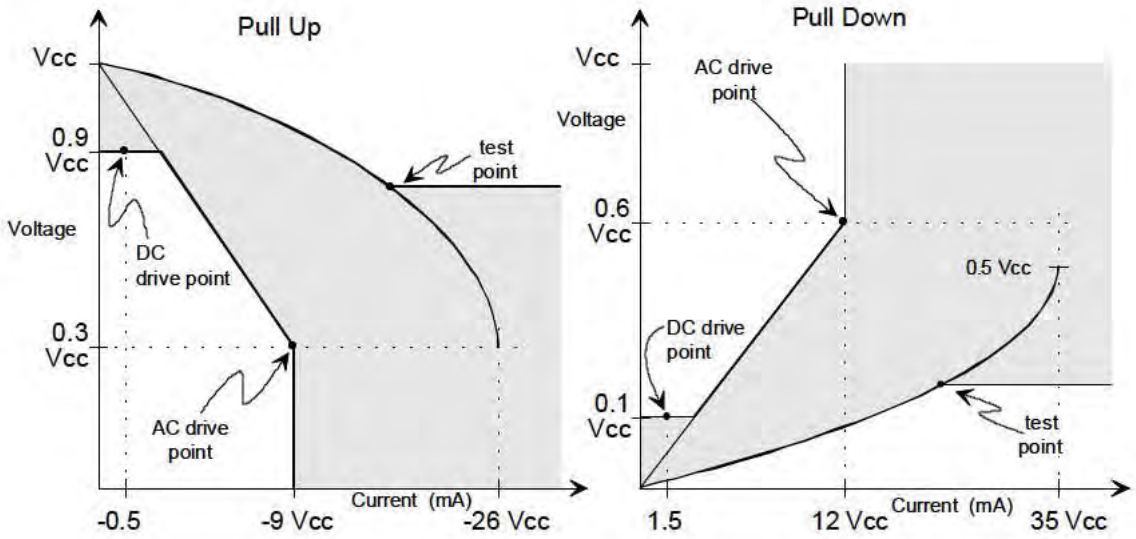
To provide optimal performance in a point-to-point environment, A.G.P. requires a driver that is roughly *half* the strength of the PCI buffer. The output driver must be able to deliver an initial voltage swing of at least the VIL/VIH value to the receiver through the bus with a known characteristic impedance. Since no external transmission line termination mechanism is specified on the A.G.P. interface, under this environment, the signal at the device pins can transition beyond Vddq and VSS voltages by a considerable amount due to signal reflection on the line. The I/O buffer must be designed to maintain acceptable signal quality levels. Output slew rate and settling time specifications are included for this purpose.

In 2X mode, the data and strobe output buffers should be designed with rise and fall delay matching to within 1ns on all process, temperature, and voltage conditions. This is required for delay matching on the data and strobe paths for source synchronous data transfer mechanism.

The minimum and maximum drive characteristics of A.G.P. output buffers are defined by V/I curves. These curves should be interpreted as traditional "DC" transistor curves with the following exceptions: the "DC Drive Point" is the only position on the curves at which steady state operation is intended, while the higher

current parts of the curves are only reached momentarily during bus switching transients. The "AC Drive Point" (the real definition of buffer strength) defines the minimum instantaneous current curve required to switch the bus with a single reflection. From a quiescent or steady state, the current associated with the AC drive point must be reached within the output delay time, T_{val}. Note however, that this delay time also includes necessary logic time. The partitioning of T_{val} between clock distribution, logic, and output buffer is not specified; but the faster the buffer (as long as it does not exceed the max rise/fall slew rate specification), the more time is allowed for logic delay inside the part. The "Test Point" defines the maximum allowable instantaneous current curve in order to limit switching noise and is selected roughly on a 50Ω load line.

Adherence to these curves should be evaluated at worst case conditions. The minimum pull up curve should be evaluated at minimum Vddq and high temperature. The minimum pull down curve should be evaluated at maximum Vddq and high temperature. The maximum curve test points should be evaluated at maximum Vddq and low temperature.



Equation C:

$$I_{oh} = (49.0/Vddq)*(V_{out}-Vddq)*(V_{out}+0.4Vddq)$$

for $Vddq > V_{out} > 0.7\ Vddq$

Equation D:

$$I_{ol} = (128/Vddq)*V_{out}*(Vddq-V_{out})$$

for $0v < V_{out} < 0.18\ Vddq$

Figure 4-17 V/I Curves for 3.3V Signaling

Inputs are required to be clamped to BOTH ground and Vddq (3.3V) rails. When dual power rails are used, parasitic diode paths could exist from one supply to another. These diode paths can become significantly forward biased (conducting) if one of the power rails goes out of spec momentarily. Diode clamps to a power rail, as well as output pull-up devices, must be able to withstand short circuit current until drivers can be tri-stated.

4.2.5 Receiver Characteristics

A differential input receiver is recommended for the 1X and 2X operation. The voltage reference is specified at 0.4Vddq. This reference voltage is compatible with the PCI 66 MHz VIL/VIH specification. A

differential buffer is not a strict requirement, as long as an implementation can meet all other AC/DC input specs.

To reduce the current consumption of the Vref supply, the differential input buffer must be designed with low input leakage current such that the combined load on Vref of all inputs is less than 10ua. The differential input buffer must be designed to have sufficient gain to convert an input differential voltage (~200mv) to a full internal CMOS voltage swing, without introducing additional skews.

4.2.6 Maximum AC Ratings and Device Protection

- All A.G.P. input, bi-directional, and tri-state output buffers should be capable of withstanding continuous exposure to the waveform shown in Figure 4-18. It is recommended that these waveforms be used as qualification criteria against which the long term reliability of each device is evaluated. This level of robustness should be guaranteed by design; it is not intended that this waveform should be used as a production test.
- These waveforms are applied with the equivalent of a zero impedance voltage source, driving through a series resistor directly into each A.G.P. input or tri-stated output pin. The open-circuit voltage of the voltage source is shown in Figure 4-18, which is based on the expected worst case overshoot and undershoot expected in actual A.G.P. busses. The resistor values are calculated to produce the worst case current into an effective (internal) clamp diode. Note that:
- The voltage waveform is supplied at the resistor shown in the evaluation setup, NOT the package pin.
- Any internal clamping in the device being tested will greatly reduce the voltage levels seen at the package pin.

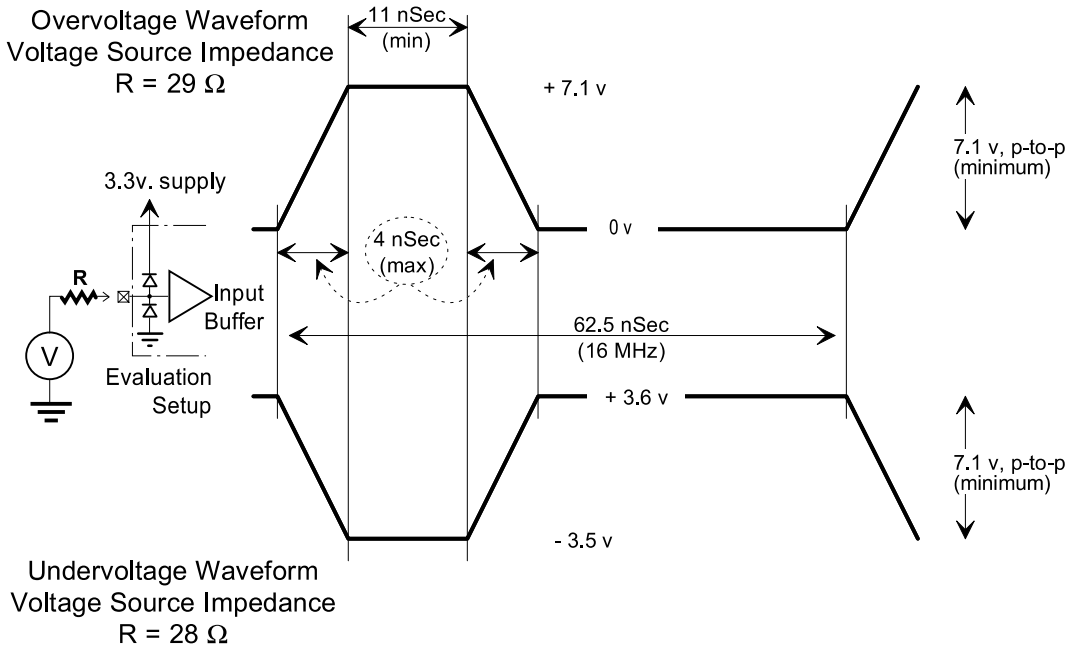


Figure 4-18 Maximum AC Waveforms for 3.3V Signaling

4.2.7 Component Pinout Recommendations

All A.G.P. signals on the graphics chip should be located to facilitate meeting the add-in card requirements as specified in section 4.3.. The component pinout should be ordered to match the connector pinout and component side of the add-in card as defined in Chapter 5 of this document. This alignment will minimize signal crossing, minimize overall trace lengths and aid in matching the trace lengths within the groups.

The strobe signals must be grouped with their associated data group:

- **AD_STB0** with **AD[15::0]** and **C/BE[1::0]#**
- **AD_STB1** with **AD[31::16]** and **C/BE[3::2]#**
- **SB_STB** with **SBA[7::0]**

Motherboard component pinouts should also be defined based on the above recommendations.

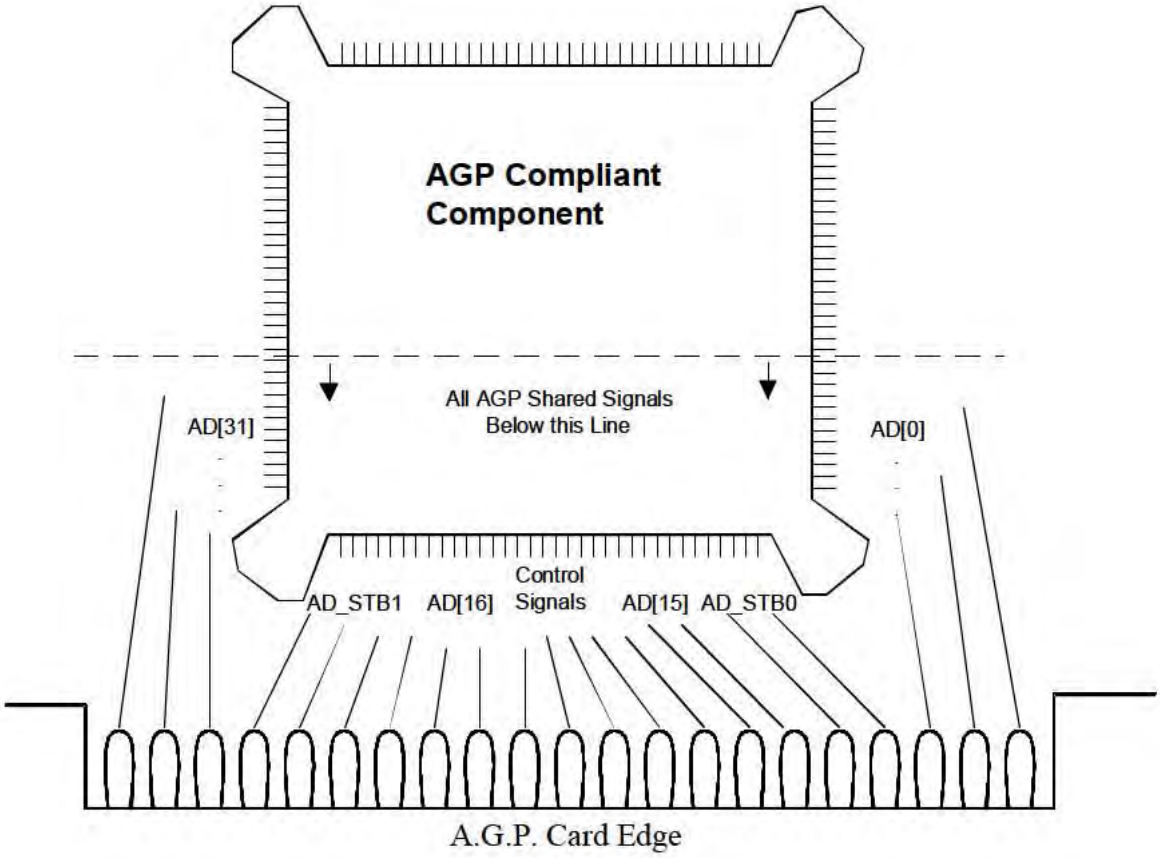


Figure 4-19 Recommended Component Pinout

4.3 Motherboard Specification

4.3.1 System Timing Budget

The table below summarizes the system timing parameters for 1X mode signals.

Table 4-7: System Timing Summary

Timing Element	Parameter	Max	Units	Notes
Tcyc	Cycle Time	15	ns	
Tval	Valid Delay	6	ns	
Tprop	Prop Delay	3	ns	
Tsu	Input Setup	5	ns	
Tskew-total	Total Skew	1	ns	1
Tskew-mb	Motherboard Skew	.9	ns	
Tskew-add-in	Add-in Card Skew	.1	ns	

NOTES:

1. Tskew is the sum of all skews (motherboard and add-in).

The table below summarizes all system interconnect delays. Note that the individual motherboard and add-in card components are repeated later in their respective sections.

Table 4-8: Interconnect Delay Summary

Symbol	Parameter	Max[1]	Units	Notes
t_{PROP}	Signal propagation	**3**	ns	2
$t_{PROP-MB}$	Signal propagation, motherboard	**2.15**	ns	
$t_{PROP-CONN}$	Signal propagation, connector	**.15**	ns	
$t_{PROP--CARD}$	Signal propagation, add-in card	**.7**	ns	
$t_{TRMATCH}$	Total Trace mismatch between Data and Strobe	**.7**	ns	3,4
$t_{TRMATCH-MB}$	Trace mismatch, motherboard	**.5**	ns	4
$t_{TRMATCH-CARD}$	Trace mismatch, card	**.2**	ns	4

NOTES:

1. Signal propagation delays are measured as the difference between the driver driving a 10pf lumped load vs. the driver driving an 80ohm transmission line terminated by a 10pf lumped load.

2. Tprop is the sum of all other propagation delays

3. Ttrmatch is the sum of all trace mismatches.

4. Trace mismatch applies between signal groups and their associated strobes: **AD_STB1=>AD[31::16]** & **C/BE[3::2]#**; **AD_STB0=>AD[15::0]** & **C/BE[1::0]#**; **SB_STB=>SBA[7::0]**. The trace mismatch spec only applies between the strobe and data signals within a group, not between data signal within a group or between groups.

4.3.2 Clock Skew

The maximum total system clock skew is 1 nsec for both 1X and 2X clock modes. This 1 nsec includes skew and jitter which originates on the motherboard, add-in card and clock synthesizer. Clock skew must be evaluated not only at a single threshold voltage, but at all points on the clock edge that fall in the switching range defined in Table 4-9 and Figure 4-20 Clock Skew Diagram.[23] This is measured between the pins of the two A.G.P. complaint components (not at the connector).

[23] The system designer may need to address an additional source of clock skew. This clock skew occurs between two components that have clock input trip points at opposite ends of the V_{il} - V_{ih} range. In certain circumstances, this can add to the clock skew measurement as described here. In all cases, total clock skew must be limited to the specified number.

- The total skew & jitter is allocated so that 0.1 nsec originates from the add-in card routing, and 0.9 nsec originates from the motherboard routing and clk synthesizer (the motherboard designer shall determine how the 0.9 nsec is allocated between the board and the synthesizer). To correctly evaluate clock skew, the system designer must take into account clock distribution on the add-in board as specified in section 4.3.2

Table 4-9 Clock Skew Parameters

Symbol	A.G.P. 1X (3.3V) Signaling	A.G.P. 2X (3.3V) Signaling	Units
V_{test}	0.4Vddq	0.4Vddq	V
T_{skew}	1 (max)	1 (max)	ns

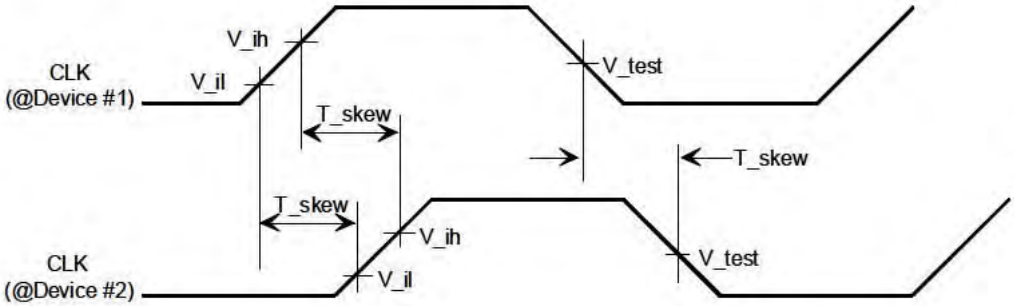


Figure 4-20 Clock Skew Diagram

4.3.3 Reset

- A.G.P. devices are reset using the PCI reset signal (**RST#**). The assertion and deassertion of **RST#** is asynchronous with respect to **CLK**. The rising (deassertion) edge of the **RST#** signal must be monotonic (bounce free) through the input switching range and must meet the minimum slew rate specified in Table 4-3. The specification does not preclude the implementation of a synchronous **RST#**, if desired. The timing parameters for reset are contained in Table 4-3, with the exception of the T_{fail} parameter. This parameter provides for system reaction to one or both of the power rails going out of spec. If this occurs, parasitic diode paths could short circuit active output buffers. Therefore, **RST#** is asserted upon power failure in order to float the output buffers.
- The value of T_{fail} is the minimum of:
- - 500 ns (maximum) from either power rail going out of specification (exceeding specified tolerances by more than 500 mV)
- - 100 ns (maximum) from the 5V rail falling below the 3.3V rail by more than 300 mV.
- The system must assert **RST#** during power up or in the event of a power failure. **RST#** should be asserted as soon as possible during the power up sequence, or as soon as the "power good" signal indicates a power failure. After **RST#** is asserted, A.G.P. complaint components must asynchronously disable (float) their outputs, but are not considered reset until both T_{rst} and $T_{rst\text{-}clk}$ parameters have been met. **RST#** therefore should not be deasserted until both T_{rst} and $T_{rst\text{-}clk}$ parameters have been met.

4.3.4 Interconnect Delay

Table 4-10: Motherboard Interconnect Delays

Symbol	Parameter	Max[1]	Units	Notes
$t_{PROP-MB}$	Signal propagation, motherboard	**2.15**	ns	
$t_{PROP-CONN}$	Signal propagation, connector	**.15**	ns	
$t_{TRMATCH-MB}$	Trace mismatch, motherboard	**.5**	ns	2

Notes:

1. Signal propagation delays are measured as the difference between the driver driving a 10pf lumped load vs. the driver driving an 80ohm transmission line terminated by a 10pf lumped load.

2. Trace mismatch applies between signal groups and their associated strobes: **AD_STB1=>AD[31::16]** & **C/BE[3::2]#**; **AD_STB0=>AD[15::0]** & **C/BE[1::0]#**; **SB_STB=>SBA[7::0].** The trace mismatch spec only applies between the strobe and data signals within a group, not between data signal within a group or between groups.

4.3.5 Physical Requirements

The AC timings and electrical loading on the A.G.P. interface are optimized for one host component on the motherboard and one A.G.P. compliant agent either on the motherboard or through a connector. The interface is a point to point network, with a maximum electrical length of 3ns. The board routing should use layout design rules consistent with high speed digital design . The followings summarize the board layout restrictions on the A.G.P. interface.

4.3.5.1 Interface Signaling

All A.G.P. signals are +3.3V compatible signals. No +5V signals are specified in the A.G.P. bus environment. The master and target device must be capable of supporting the 3.3V signaling environment. The interrupt signals from the A.G.P. bus must interface to the PCI bus interrupt controller. This controller and the PCI compliant devices may be a +5V devices. It is the requirement of the motherboard designer to properly interface the A.G.P. interrupts to the PCI bus. This can be done is several ways. One way is to pullup the PCI interrupts to 3.3V only, allowing the A.G.P. interrupts to connect directly to the PCI interrupts. Alternatively, the A.G.P. interrupts can be buffered to the PCI bus, thus isolating the 5V environment from the A.G.P. bus.

4.3.5.2 Pullups

A.G.P. control signals require pullups to Vddq on the motherboard (or, optionally integrated on motherboard chipset) to ensure they contain stable values when no agent is actively driving the bus. These signals include

FRAME#, **TRDY#**, **IRDY#**, **DEVSEL#**, **STOP#**, **SERR#**, **PERR#**, **LOCK#**, **INTA#, INTB#**[24], **PIPE#**, **AD_STB[1::0]** and **SB_STB.**

The pull-up value requirements are shown in the table below.

Table 4-11: Pull-up Resistor Values

Rmin	Rtypical	Rmax	Notes
4K Ω	8.2K Ω @ 10%	16K Ω	

NOTES:

4.3.5.3 Signal Routing and Layout

A.G.P. signals must be carefully routed on the motherboard to meet the timing and signal quality requirements of this interface specification. The following are some general guidelines that should be followed. Trace lengths included in this section are guidelines only. It is recommended that the board designer simulate the routes to verify that the specification is met.

The total flight time allowed for the A.G.P. bus is 3ns. The timing budget for the components of the flight path is identified in table 4-3. The motherboard prop delay budget of 2.15ns restricts the total trace length on the motherboard to less than approximately 10 inches.

The trace lengths for signals within a group must be matched to meet the total mismatch requirement given in Table 4-10, of 0.5ns. This means that the traces within a group on the motherboard must be matched to within approximately 2 inches, and are recommended to be matched as closely as possible to provide timing margin.

4.3.5.4 Impedances

The motherboard impedances should be controlled to minimize the impact of any mismatch between the motherboard and the add-in card. A impedance of 65Ω ±15Ω is strongly recommended, otherwise signal integrity requirements may be violated.

4.3.5.5 Vref Generation

The motherboard must generate Vref locally for any motherboard component which requires it. Vref should be generated from the A.G.P. interface Vddq rail, not the component power supply. Vref should be properly decoupled to ground to manage switching currents. Such decoupling is platform dependent, and therefore not specified.

4.3.5.6 Crosstalk Consideration

For 66 and 133MT/s transfer, noise due to crosstalk must be carefully controlled to a minimum. Refer to the A.G.P. Design Guide for typical values.

---

[24] **INTA#** and **INTB#** are special cases. The motherboard should ensure these signals can not float to other motherboard inputs, while meeting the interface signaling requirements on section 4.3.5.1. If **INTA#** or **INTB#** are connected to inputs on the add-in card, the add-in card design must ensure those inputs can not float.

4.3.5.7 Line Termination

No external line termination mechanisms are specified on the A.G.P. interface, but may be used to meet signal integrity requirements as long as these elements do not inhibit agents from meeting their performance specifications. Circuit design techniques may be used to handle signal reflection and over/undershoot, such as clamping devices and slew rate controlled output buffers, to achieve acceptable signal integrity.

4.4 Add-in Card Specification

4.4.1 Clock Skew

The clock trace on the add-in card shall be routed to achieve an interconnect delay of 0.6 ± 0.1 ns. System designers will assume the delay of 0.6 nsec while designing the motherboard for minimum clock skew. The tolerance of ± 0.1 nsec is the clock skew contribution allocated for the add-in card, as specified in section 4.3.2..

4.4.2 Interconnect Delay

Table 4-12: Add-in Card Interconnect Delays

Symbol	Parameter	Max[1]	Units	Notes
$t_{PROP--CARD}$	Signal propagation, add-in card	.7	ns	
$t_{TRMATCH-CARD}$	Trace mismatch, card	.2	ns	2

NOTES:

1. Signal propagation delays are measured as the difference between the driver driving a 10pf lumped load vs. the driver driving an 80ohm transmission line terminated by a 10pf lumped load.

2. Trace mismatch applies between signal groups and their associated strobes: **AD_STB1=>AD[31::16]** & **C/BE[3::2]#**; **AD_STB0=>AD[15::0]** & **C/BE[1::0]#**; **SB_STB=>SBA[7::0].** The trace mismatch spec only applies between the strobe and data signals within a group, not between data signal within a group or between groups.

4.4.3 Physical Requirements

4.4.3.1 Pin Assignment

Pins labeled *Vddq* are special power pins for defining and driving the A.G.P. signal rail on the board. On the board, the A.G.P. compliant component's I/O buffers must be powered from these special pins only — not from the other +5V or 3.3V power pins.[25]

4.4.3.2 Signal Routing and Layout

A.G.P. signals must be carefully routed on the graphics card to meet the timing and signal quality requirements of this interface specification. The following are some general guidelines that should be followed. Trace lengths included in this section are guidelines only. The card designer must simulate the routes to verify that the specification is met.

The total flight time allowed for the A.G.P. bus is 3ns. The timing budget for the components of the flight path is identified in Table 4-7. The add-in card prop delay budget of .7ns restricts the total trace length on the add-in card to be approximately 3 inches.

The trace lengths for signals within a group must be matched to meet the total mismatch requirement given in Table 4-12, of .2ns. This means that the traces within a group on the add-in card must be matched to within approximately 0.9 inches, and are recommended to be matched as closely as possible to provide timing margin. Refer to Figure 4-19 for add-in card component placement recommendations.

4.4.3.3 Impedances

The add-in card impedances should be controlled to minimize the impact of any mismatch between the motherboard and the add-in card. A impedance of 65Ω $\pm$15Ω is strongly recommended, - otherwise signal integrity requirements may be violated.

4.4.3.4 Vref Generation

The add-in card must generate Vref locally for any add-in card component which requires it. Vref should be generated from the A.G.P. interface Vddq rail, not the component power supply.

4.4.3.5 Power supply delivery.

The power supply to the add-in card for core supply (VCC) and I/O supply voltage(Vddq) must be separated on the die, package and add-in card. Also there is a power level sequencing of core supply (VCC) and I/O supply voltage (Vddq) that needs to be considered which will be detailed in the next version of the specification.

[25] Any clamp diodes on A.G.P. signal pins must only connect to the Vddq rail and not to the 3.3V or 5V Vcc rails.

5. Mechanical Specification

5.1 Introduction

A.G.P. provides a high performance graphics interface. The A.G.P. connector was defined to meet this requirement and provide a robust implementation for an A.G.P. expansion card. This section defines an A.G.P. connector intended for high volume, high performance desktop systems. The connector must be low cost, reliable, electrically robust, and manufacturable in high volume from multiple sources. The A.G.P. connector effectively replaces one of the planar PCI connectors. The PCI connector that was previously utilized for graphics is now replaced by the A.G.P. connector which provides a higher performance.

The A.G.P. expansion card is based on the PCI expansion card design with the same maximum dimensions and configuration. It is easily implemented in existing chassis designs from multiple manufacturers. The A.G.P. expansion card requires a mounting bracket for card location and retention which is the same as the PCI ISA Retainer. The bracket is the interface between the card and the system that provides for cable escapement just as in PCI implementations. (See PCI Rev. 2.1, page 169 for ISA bracket and page 178 for ISA retainer.) The bracket shall be supplied with the card so that the card can be easily installed in the system.

5.2 Expansion Card Physical Dimensions and Tolerances

The A.G.P. card, like the PCI card, is designed to fit most existing chassis. The maximum component height on the primary side of the A.G.P. expansion card is not to exceed 0.570 inches (14.48 mm). The maximum component height on the backside of the card is not to exceed 0.105 inches (2.67 mm). Datum A on the illustrations is used to locate the A.G.P. card to the planar and to the frame interfaces; the back of the frame and the card guide. Datum A is carried through the locating key on the card edge and the locating key on the connector.

See Figure 5-1 and Figure 5-2 for A.G.P. expansion card physical dimensions.

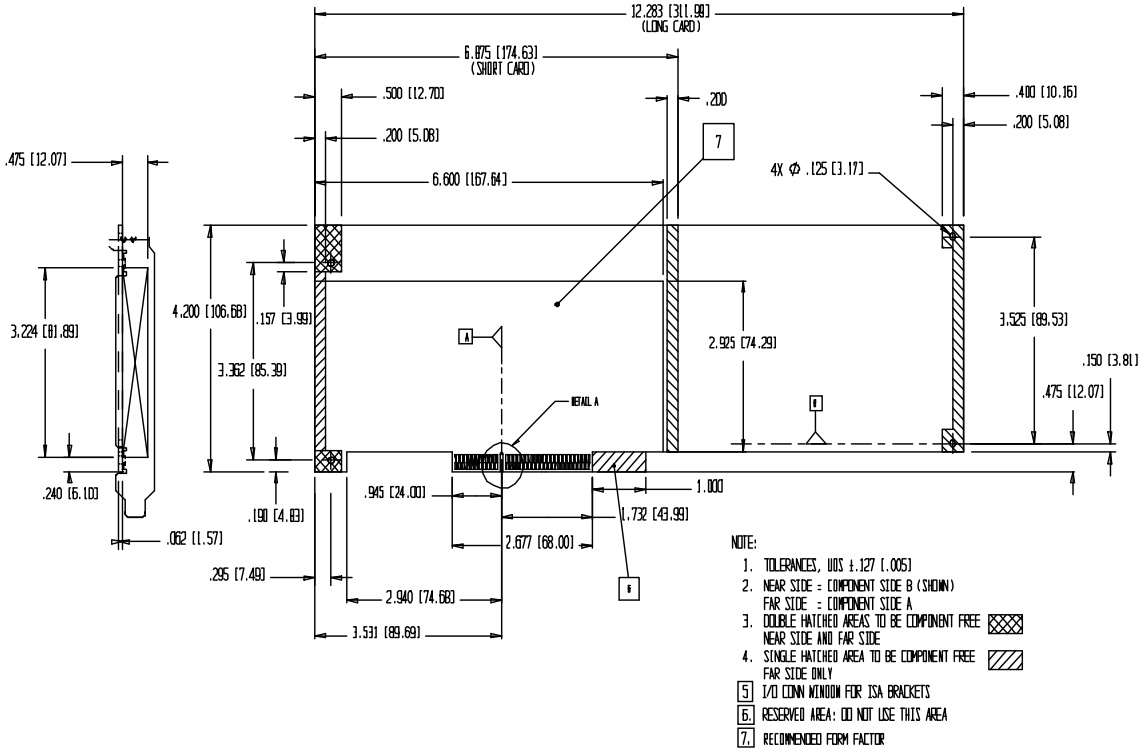


Figure 5-1 A.G.P. Card Edge Connector

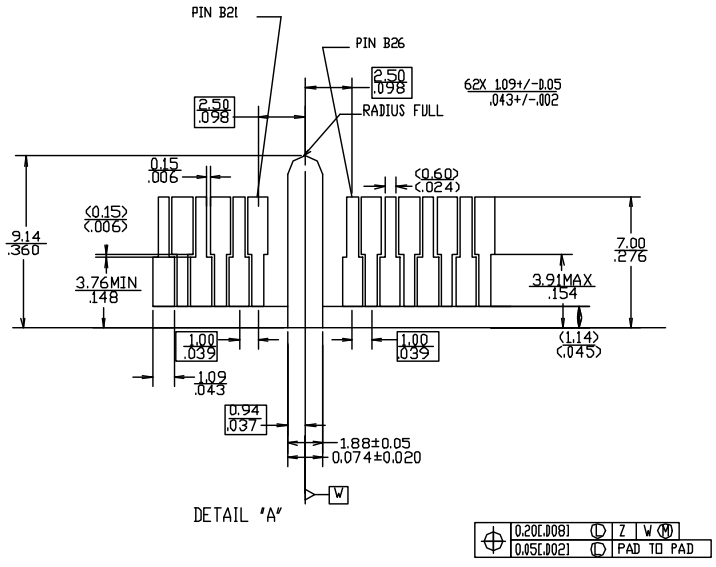


Figure 5-2 **Detail A, A.G.P. Card Edge Finger Layout**

5.3 Connector:

The connector shall hold the card at right angles to the system board. All dimensions are metric, inch dimensions are shown for reference only. The connector must accommodate a 1.57 mm (0.062 in) thick card (see Figure 5-3). Key width dimension of 1.78 mm (.070 inch) is measured prior to draft.

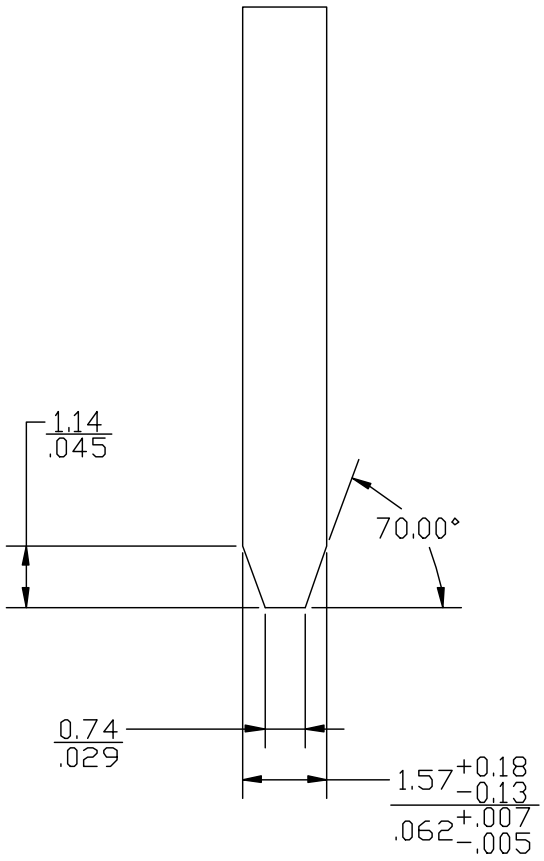


Figure 5-3 A.G.P. Card Edge Connector Bevel

5.4 Connector Physical Description

This section provides a physical description of the A.G.P. connector. The connector is intended for high volume, high performance desktop systems. In the connector drawings, the recommended board layout details are given as nominal dimensions. Layout detail tolerances should be consistent with the connector supplier's recommendations and good engineering practice. See

Figure 5-5 for connector dimensions and layout recommendations.

- The connector specification detail and connector supplier information is available from the A.G.P. homepage (http://www.teleport.com/~agfxport/).
- Caution: The connector is NOT hot unpluggable. Be sure system and Motherboard power is off.Unplugging an A.G.P. card with power and/or signals enabled at the connector may cause irreparable damage to the card and/or system boards. Disabling power and signals at the connector is a highly recommended standard practice for existing systems using ISA, EISA and PCI expansion cards.
- Caution: It is highly recommended that A.G.P. plug in cards are plugged and unplugged "straight" into the connector, without rocking. Using a "rocking" motion to seat and/or unseat the A.G.P. card may cause damage to the system board A.G.P. connector and/or damage pads on the add-in card. This is consistent with good practice and recommendations for the presently used PCI connector.

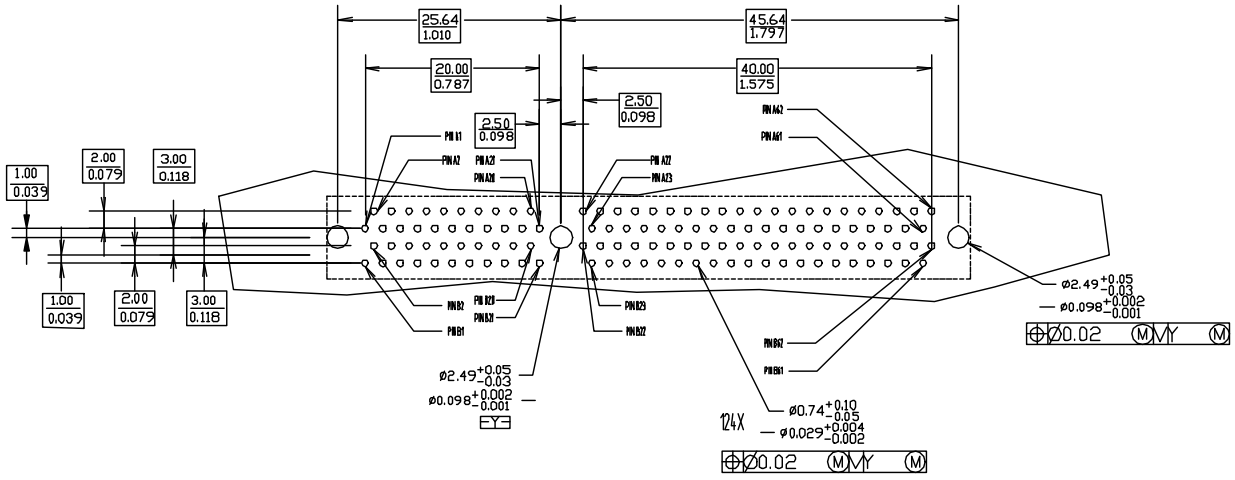


Figure 5-4 Motherboard Connector footprint and layout deminsions.

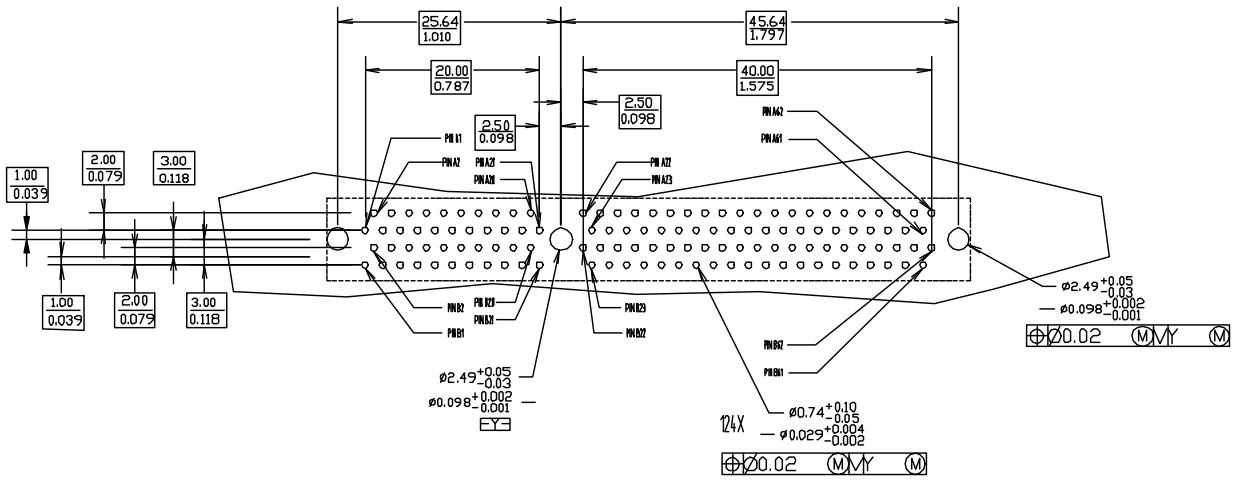


Figure 5-5 **Motherboard Connector Layout Recommendation**

5.5 Planar Implementation

5.5.1 ATX Planar Implementation

Planar implementations are supported by the A.G.P. expansion card design. For illustration purposes, the planar mounted expansion connector is detailed in Figure 5-6. This example shows an ATX form factor planar. The A.G.P. connector effectively replaces one of the planar PCI connectors. The PCI connector that was utilized for graphics is now replaced by the A.G.P. connector which provides a higher performance A.G.P. interface. The principles outlined can be applied to locate the A.G.P. connector in any motherboard.

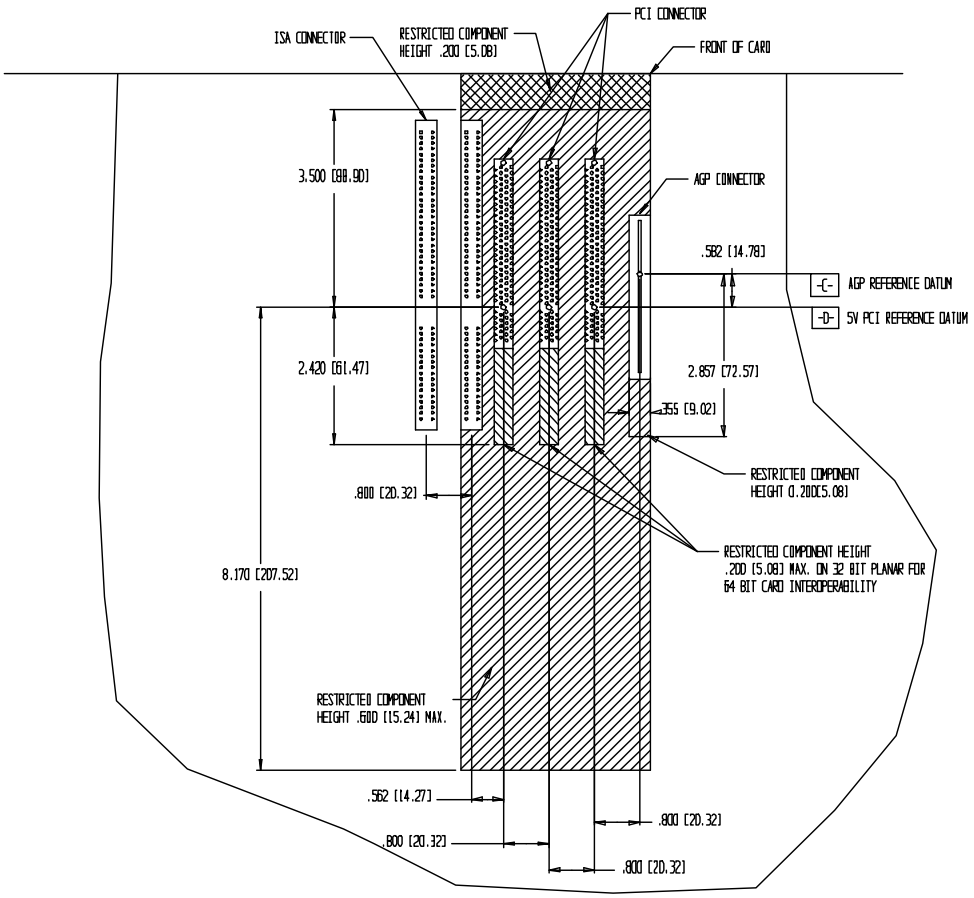


Figure 5-6 Typical ATX Implementation

5.5.2 Low Profile Planar Implementation

A.G.P.'s electrical and mechanical requirements prevent it from being placed on a low profile chassis's riser card, and requires that it be supported directly from the motherboard. To do this, the A.G.P. video solution can be implemented either integrated as video down on the motherboard, or as an A.G.P. expansion slot, but both cannot be supported simultaneously.

Integrating A.G.P. video down on the motherboard offers the fewest restrictions on the placement and routing of the circuit and connectors. This solution requires no special treatment in the A.G.P. interface specification.

5.5.3 Pin List

	A.G.P.1	
Pin#	B	A
1	Spare	**12V**
2	**5.0V**	Spare
3	**5.0V**	***Reserved****
4	***USB+***	***USB-***
5	GND	GND
6	INTB#	INTA#
7	CLK	RST#
8	REQ#	GNT#
9	VCC3.3	VCC3.3
10	ST0	ST1
11	ST2	Reserved
12	**RBF#**	PIPE#
13	GND	GND
14	**Spare**	Spare
15	SBA0	SBA1
16	VCC3.3	VCC3.3
17	SBA2	SBA3
18	SB_STB	Reserved
19	GND	GND
20	SBA4	SBA5
21	SBA6	SBA7
22	KEY	KEY
23	KEY	KEY
24	KEY	KEY
25	KEY	KEY
26	AD31	AD30
27	AD29	AD28
28	VCC3.3	VCC3.3
29	AD27	AD26
30	AD25	AD24
31	GND	GND
32	**AD_STB1**	Reserved
33	AD23	C/BE3#
34	Vddq3.3	Vddq3.3
35	AD21	AD22
36	AD19	AD20
37	GND	GND
38	AD17	AD18
39	C/BE2#	AD16
40	Vddq3.3	Vddq3.3
41	**IRDY#**	**FRAME#**
42		
43	GND	GND
44		
45	VCC3.3	VCC3.3
46	**DEVSEL#**	**TRDY#**
47	**Vddq3.3**	STOP#
48	PERR#	**Spare**
49	GND	GND
50	SERR#	PAR
51	C/BE1#	AD15
52	Vddq3.3	Vddq3.3
53	AD14	AD13
54	AD12	AD11
55	GND	GND
56	AD10	AD9
57	AD8	C/BE0#
58	Vddq3.3	Vddq3.3
59	**AD_STB0**	Reserved
60	AD7	AD6
61	GND	GND
62	AD5	AD4
63	AD3	AD2
64	Vddq3.3	Vddq3.3
65	AD1	AD0
66	SMB0	SMB1

* - This reserved pin should be connected to GND

Note: **IDSEL** is not a pin on the A.G.P. connector. A.G.P. add-in card manufacturer's should connect the **AD16** signal to the **IDSEL** pin on an A.G.P. compliant Master.

6. System Configuration and A.G.P. Initialization

A.G.P. configuration and initialization operations are of three general types.

1. Power On Startup Test (POST) code allocates resources to all devices in the system. (BIOS)
2. The operating system activates A.G.P. features. (Not BIOS)
3. The final runtime memory management activity is carried out by Microsoft's DirectDraw.

The first two of these steps are described here. Refer to Microsoft documentation for details on the third.

6.1 POST-time initialization

Conventional bus enumeration software in the PowerOnSartupTest (POST) code identifies all system devices *(includes A.G.P. compliant devices),* creates a consistent system address map and allocates system resources to each device. An A.G.P. compliant device (master or target) must provide all required fields in the device's PCI configuration header, including Device ID, Vendor ID, Status, Command, Class code, Revision ID and Header type. (See Section 6.1 of the PCI 2.1 specification for more detail.) By supporting the PCI header, this allows conventional bus enumeration software to function correctly while being completely unaware of A.G.P. features.

6.1.1 A.G.P. Compliant Master Devices

A.G.P. compliant master devices have a certain amount of memory resources that must be placed somewhere in the system memory address map using a PCI base address register. These memory resources fall into two categories, Prefetchable and Non-prefetchable address regions. Prefetchable memory space is where the Linear Framebuffer is mapped to provide performance improvements. Non-prefetchable memory space is where control registers and FIFO-like communication interfaces are mapped. Each of these address regions should have their own base address register. Refer to page 196 of the PCI 2.1 specification for a description of PCI base address registers.

6.1.2 A.G.P. Compliant Target Devices

A.G.P. compliant target devices require a certain amount of address space for A.G.P. memory that must be placed somewhere in the system address map using a PCI base address register. Non-prefetchable control registers when supported by the target are provided by a second base address register.

To enable the use of existing enumeration code (unmodified) to handle A.G.P. compliant devices, Figure 6-1 is a logical view of how an A.G.P. compliant target appears. The area inside the dotted line represents the A.G.P. compliant target and core logic chipset. The corelogic (in this example) includes ports to the System memory, Processor, PCI and A.G.P.. The two main functions in the figure are the Host Bus Bridge and the PCI to PCI Bridge. The Host Bus Bridge is the interface that exists in all corelogic that spawn a PCI bus segment. The PCI to PCI Bridge function, facilitate the configuration of the second I/O port (A.G.P.) of the corelogic without requiring new enumeration code. With the corelogic presenting the interface (required to follow the PCI to PCI Bridge Architecture Specification (1.0)), this provides a way to determine what device resides on the A.G.P. port, what system resources it requires, and the mechanism to allocate those resources.

A benefit of this is that the host bridge obtains the mapping information (without special software) to route requests to the correct destination (i.e., PCI, A.G.P. or main memory).

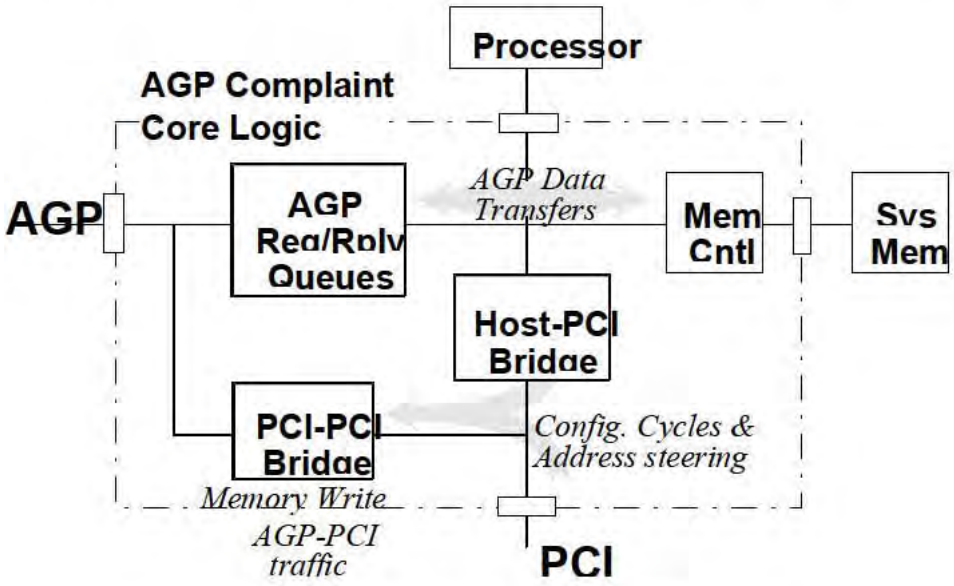


Figure 6-1 Configuration View of an A.G.P. Target

Note: the PCI to PCI Bridge function in the figure is NOT a complete and fully functional PCI to PCI Bridge. This implementation DOES not allow all access types to transverse between the A.G.P. and PCI segments. Only Memory write transactions (PCI commands Memory Write and Memory Write and Invalidate) that initiate on A.G.P. are forwarded to the Primary PCI bus segment and on Primary PCI bus are forwarded to A.G.P. segment. All other PCI commands (I/O (read and write), configuration (read and write) and Memory reads (Memory Read, Memory Read Line and Memory Read Multiple, Special Cycles, and Interrupt Acknowledge) are not allowed to cross the interface. When one of these commands is issued and has destination on the other interface the corelogic can treat this as a programming error. How it completes the access is chipset specific, one option would be to have the corelogic (acting as a PCI compliant target) to simply ignore the request and allow it to terminate with Master-abort.

The A.G.P. compliant target's base address registers should reside in the Host-to-PCI bridge because the Windows operating system can load a Host-to-PCI bridge driver, whereas there is no provision for loading a driver for a PCI-to-PCI bridge.

6.1.3 Boot-time VGA Display Device(s)

Most A.G.P. graphics accelerators will have a VGA display device. This means some systems may have more than one VGA device. Conventional BIOS codes select one VGA device by first searching the ISA bus, then PCI add-in card slots *(includes A.G.P. connector)*, then motherboard devices *(includes motherboard A.G.P. compliant devices)*.

> **Reminder:** Boot-time bus enumeration software is unaware of A.G.P. features.

6.1.4 Operating System Initialization

The operating system initializes A.G.P. features by performing the following operations:

1. Allocate memory for the A.G.P. remapping table

2. Initialize the A.G.P. compliant target's address remapping hardware

3. Set the A.G.P. compliant target and master data transfer parameters

4. Set host memory type for A.G.P. memory

5. Activate policy limiting the amount of A.G.P. memory

An A.G.P. chipset driver API will be used for the second item. Refer to the appropriate Microsoft device driver interface kit for details.

The third item requires access to configuration registers defined later in this interface specification. Setting bit 4 (Status Register) at offset 6 indicates the device implements New Capabilities mechanism as described by PCI[26]. The New Capabilities structure is implemented as a linked list of registers containing information for each function supported by the device. A.G.P. status and command registers are included in the linked list. The structure for the A.G.P. specific ID and structure is illustrated in Figure 6-2.

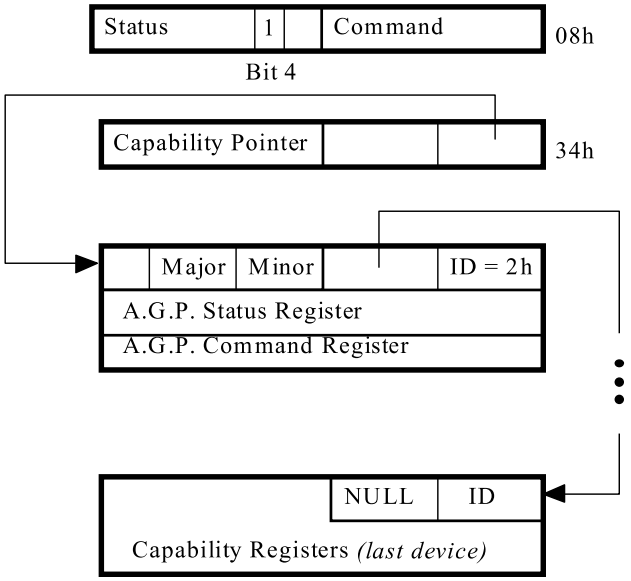


Figure 6-2 Location of A.G.P. Capabilities

Configuration registers are used by the OS to initialize A.G.P. features. These features must be supported by both A.G.P. compliant master and target devices in the following registers. The explanatory text describes the specific behavior of the target and master with respect to each function.

[26] An ECR defining the "New Capabilities" to the PCI 2.1 Specification is in process.

6.1.5 PCI Status Register

Bit	Field	Description
31:5		See PCI 2.1 specification
4	**CAP_LIST**	If the **CAP_LIST** bit is set, the device's configuration space implements a list of capabilities. This bit is *Read Only* register.
3:0		See PCI 2.1 specification

6.1.6 Capabilities Pointer - (offset 34h)

Bits	Field	Description
31:8	Reserved	Always returns 0 on read, write operations have no effect
7:0	**CAP_PTR**	This field contains a byte offset into the device's configuration space containing the first item in the capabilities list and is a *Read Only* register.

CAP_PTR gives the location of the first item in the list, which, in this example, is for the A.G.P. compliant device. Device capabilities may appear in any order in the list. The CAP_PTR register and the Capability Identifier register are *Read Only* with reserved fields returning zero when read.

6.1.7 Capability Identifier Register (Offset = CAP_PTR)

Bits	Field	Description
31:24	Reserved	Always returns 0 on read; Write operations have no effect.
23:20	**MAJOR**	Major revision number of A.G.P. interface specification this device conforms to.
19:16	**MINOR**	Minor revision number of A.G.P. interface specification this device conforms to.
15:8	**NEXT_PTR**	Pointer to next item in capabilities list. Must be *NULL* for final item in list.
7:0	**CAP_ID**	The value 02h in this field identifies the list item as pertaining to A.G.P. registers.

The first byte of each list entry is the capability ID. The PCI Special Interest Group assigned A.G.P. an ID of 02h. The **NEXT_PTR** field contains a pointer to the next item in the list. The **NEXT_PTR** field in final list item must contain a NULL pointer.

6.1.8 A.G.P. status register (offset CAP_PTR + 4)

Bits	Field	Description
31:24	**RQ**	The **RQ** field contains the maximum number of A.G.P. command requests this device can manage.
23:10	Reserved	Always returns 0 when read, write operations have no effect
9	**SBA**	If set, this device supports side band addressing.
2:8	Reserved	Always returns 0 when read, write operations have no effect
1:0	**RATE**	The **RATE** field indicates the data transfer rates supported by this device. A.G.P. compliant devices must report all that apply. *<Bit 0: 1X, Bit 1: 2X > Note: The* **RATE** *field applies to* ***AD*** *and* ***SBA*** *buses.*

The A.G.P. status register is a *Read Only* register. Writes have no affect, and reserved or unimplemented fields return zero when read.

6.1.9 A.G.P. command register - (offset CAP_PTR + 8)

Bits	Field	Description
31:24	**RQ_DEPTH**	Master: The **RQ_DEPTH** field must be programmed with the maximum number of pipelined operations the master is allowed to enqueue in the target. Value set in this field must be equal to or less than the value reported in the **RQ** field of target's status register. Target: The **RQ_DEPTH** field is reserved.
23:10	Reserved	Always returns 0 when read, write operations have no effect
9	**SBA_ENABLE**	When set, the side address mechanism is enabled in this device.
8	**AGP_ENABLE**	Master: Setting the **AGP_ENABLE** bit allows the master to initiate A.G.P. operations. When cleared, the master cannot initiate A.G.P. operations. Target: Setting the **AGP_ENABLE** bit allows the target to accept A.G.P. operations. When cleared, the target ignores incoming A.G.P. operations. *Notes: 1. The target must be enabled before the master .* *2. The* **AGP_ENABLE** *bit is cleared by* **AGP_RESET**.
7:3	Reserved	Always returns 0 when read, write operations have no effect
2:0	**DATA_RATE**	One *(and only one)* bit in the **DATA_RATE** field must be set to indicate the desired data transfer rate. <*Bit 0: 1X, Bit 1: 2X*>. The same bit must be set on both master and target. *Note: The* **DATA_RATE** *field applies to* **AD** *and* **SBA** *buses*

The A.G.P. command register is a read/write register, with reserved fields returning zero when read and writes having no affect. All bits in the A.G.P. command register are initialized to zero at reset.

7. Appendix A

[Preliminary – subject to change]

1. API	Application Programming Interface.
2. Bus 0	Compatibility PCI bus *(where ISA bridge resides).*
3. Bus *n*	PCI/A.G.P. bus. *n* = 1 when no PCI to PCI Bridges present on Bus 0.
4. Chipset	Motherboard chipset that provides connections to: Host Bus, Compatibility PCI bus, and A.G.P. interface.
5. DirectX	Microsoft API's for accessing graphics and audio hardware.
6. DirectDraw	Microsoft graphics API.
7. Display surface	Memory area containing graphics data object.
8. Fence	Means of synchronizing A.G.P. write operations with subsequent A.G.P. read operations.
9. Flush	Operation that makes an A.G.P. compliant target's accesses to system graphics memory visible to other parts of the system.
10. A.G.P. bus	Abbreviation for PCI/A.G.P. bus.
11. A.G.P. compliant master	An A.G.P. compliant master interface that is capable of generating A.G.P. pipelined read/write operations per this interface specification.
12. A.G.P. port	Connection point on a chipset where an A.G.P. compliant master may be attached.
13. A.G.P. operation	Memory read/write operation subject to A.G.P. ordering rules and protocol. A.G.P. operations that are initiated by **PIPE#** or **SBA** bus.
14. A.G.P. compliant target	An A.G.P. compliant target interface that is capable of interpreting A.G.P. addresses (e.g. the chipset) per this interface specification.
15. Local graphics memory	Memory local to the graphics controller.
16. Non-prefetchable memory	PCI registers that have side effects.
17. Prefetchable Memory	PCI memory and memory mapped registers that are free from side-effects.
18. Uncacheable Memory	Host caching protocol used on I/O operations, non-prefetchable regions or prefetchable regions not supported by hardware coherency.
19. Ordering rules	Rules specifying when the effect of a read or write operation can be observed by another operation.
20. Paging	Movement of data between disk and other memory levels of the virtual memory system.
21. Pipelining	A.G.P. read/write operations use a *split transaction* like paradigm

	where one or more addresses may be transferred during one bus operation and data is transferred during another.
22. POST or POST Code	Initialization software executed out of the startup ROM.
23. SPI	System Programming Interface.
24. Synchronization	A.G.P. ordering rules may allow certain memory operations to occur in to provide improved performance and may complete in a different order than initiated by the master. Synchronization operations provide additional control of the completion order.
25. System graphics memory	System memory accessible via the A.G.P. port by the graphics controller.
26. WriteBack	A type of Host cache coherency used for application memory.

8. Appendix B

This appendix contains other timing diagrams that are interesting but not used in the body of this interface specification. These figures are believed to be correct, however errors may exist and the rules and wording in the main interface specification have precedence.

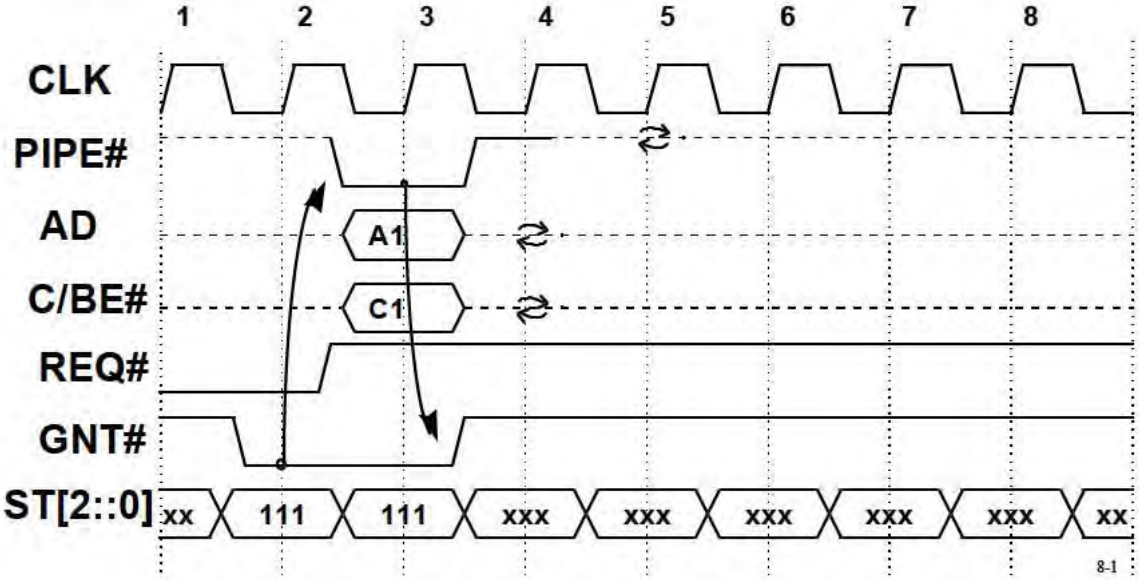


Figure 8-1 Single Address - No Delay by Master

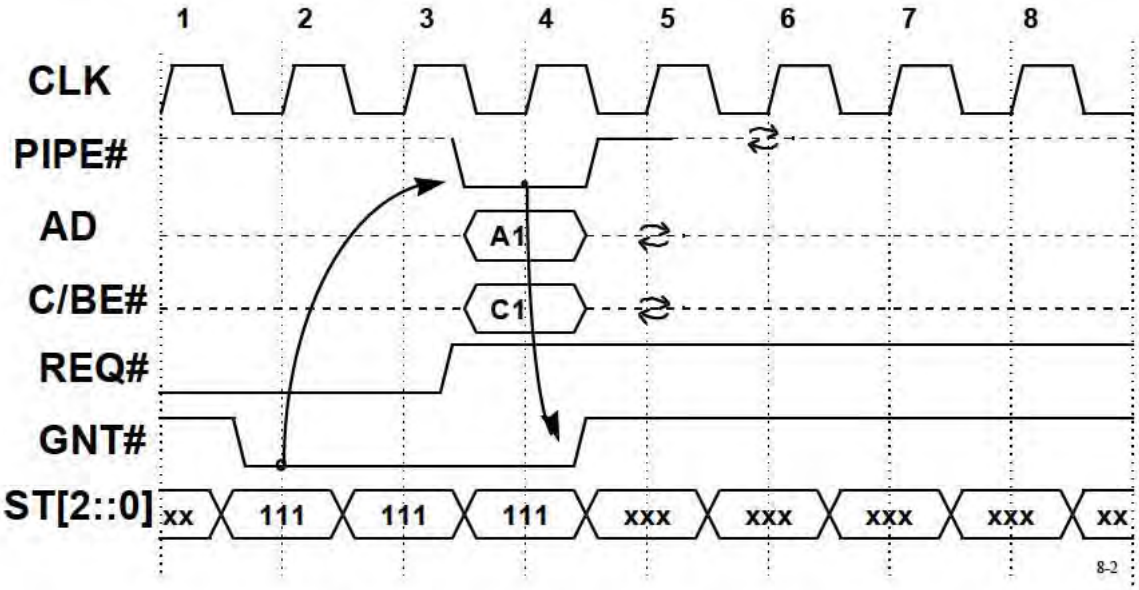


Figure 8-2 Single Address - Maximum Delay by Master

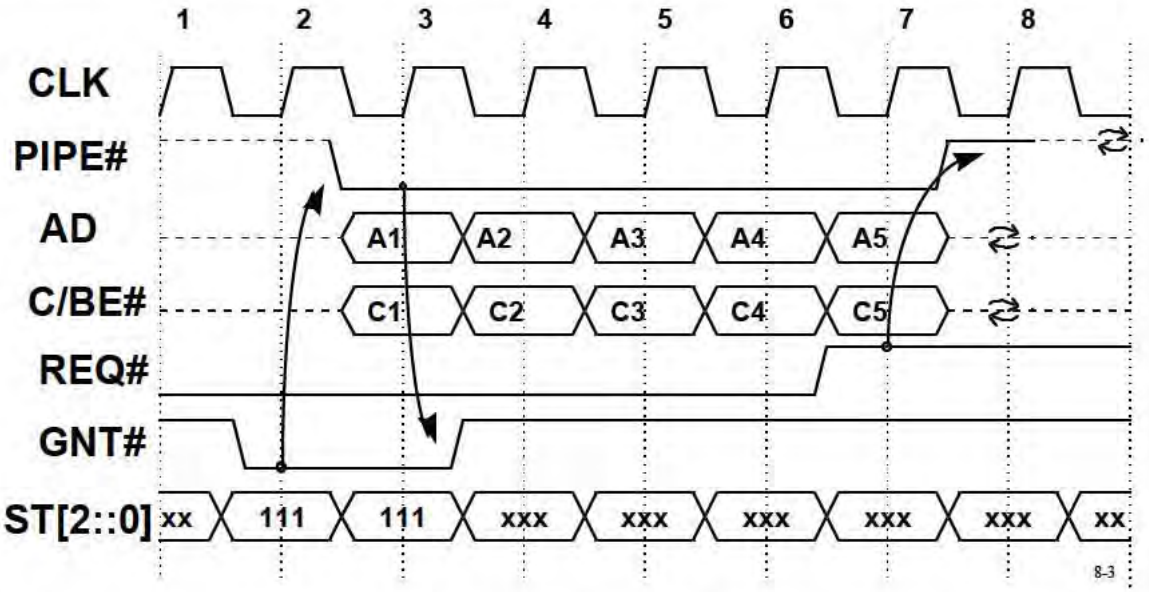


Figure 8-3 Multiple Address - No Delay by Master

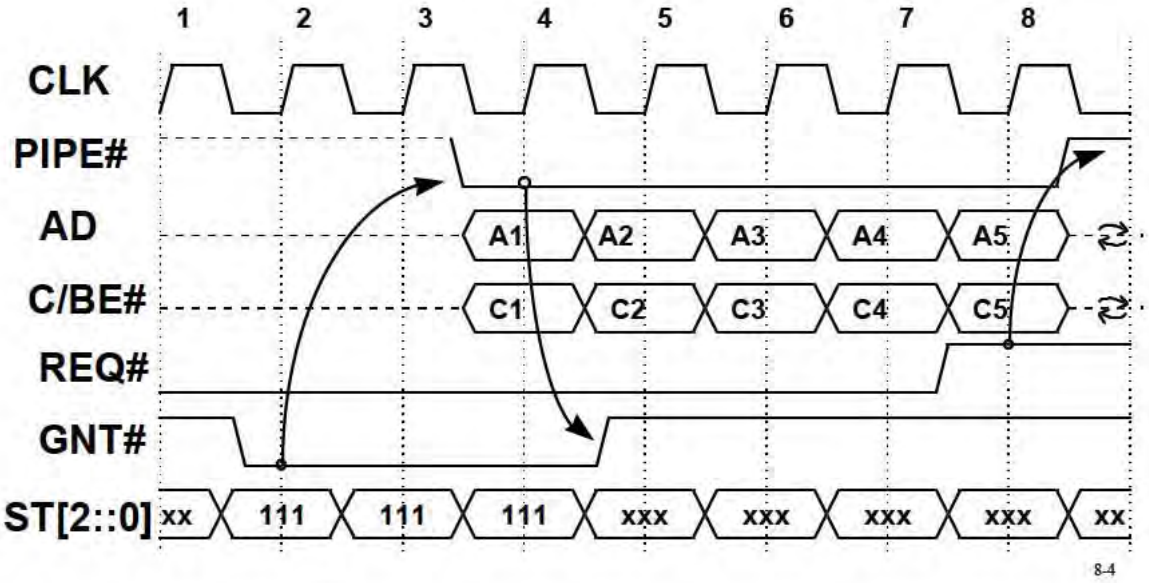


Figure 8-4 Multiple Address - Maximum Delay by Master

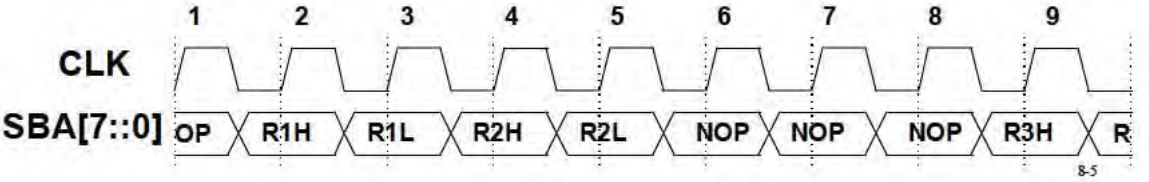


Figure 8-5 1x SideBand Addressing

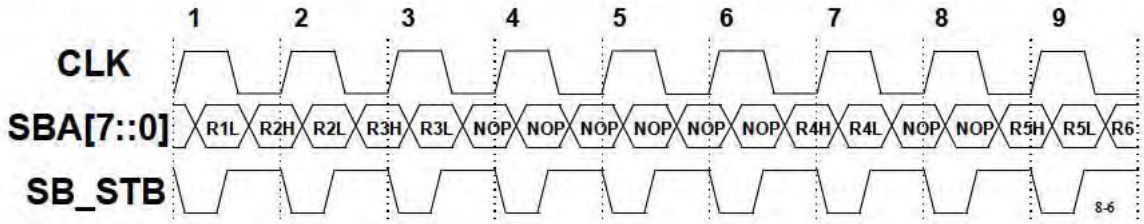


Figure 8-6 2x SideBnd Addressing

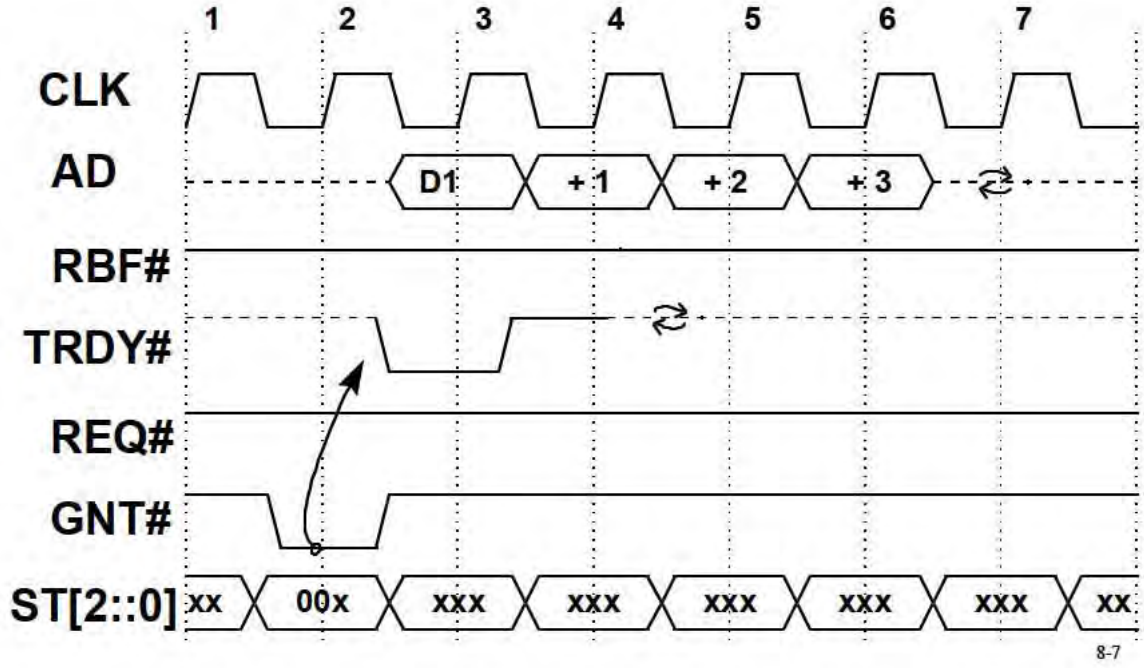


Figure 8-7 Minimum Delay by Target of Read Transaction

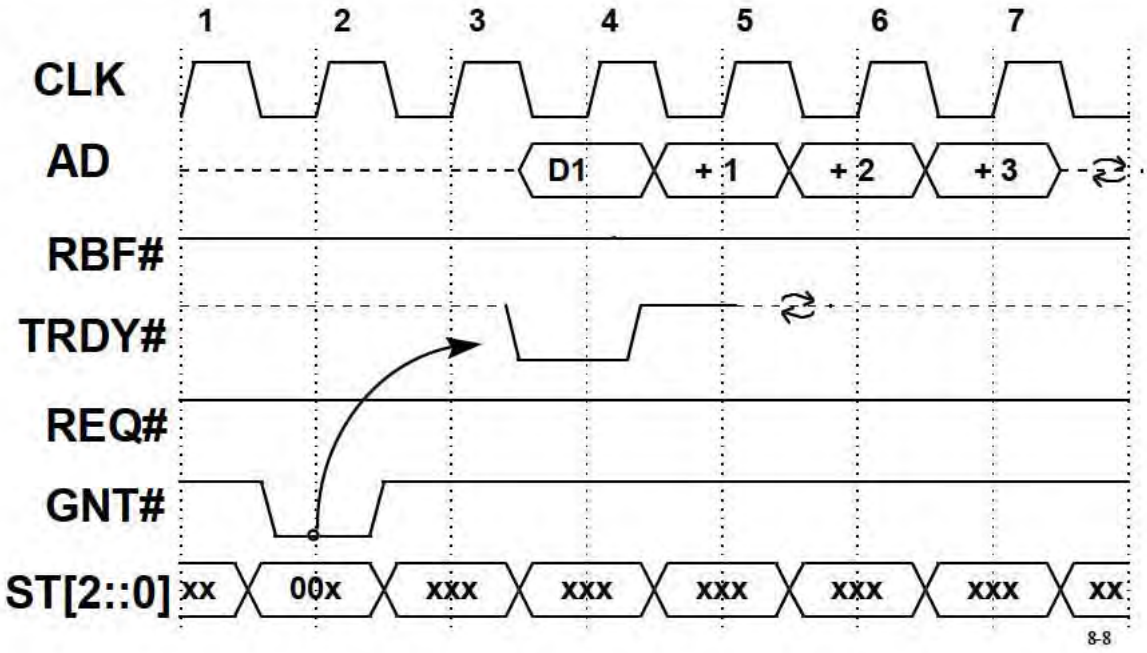


Figure 8-8 Maximum Delay by Target of Read Transaction

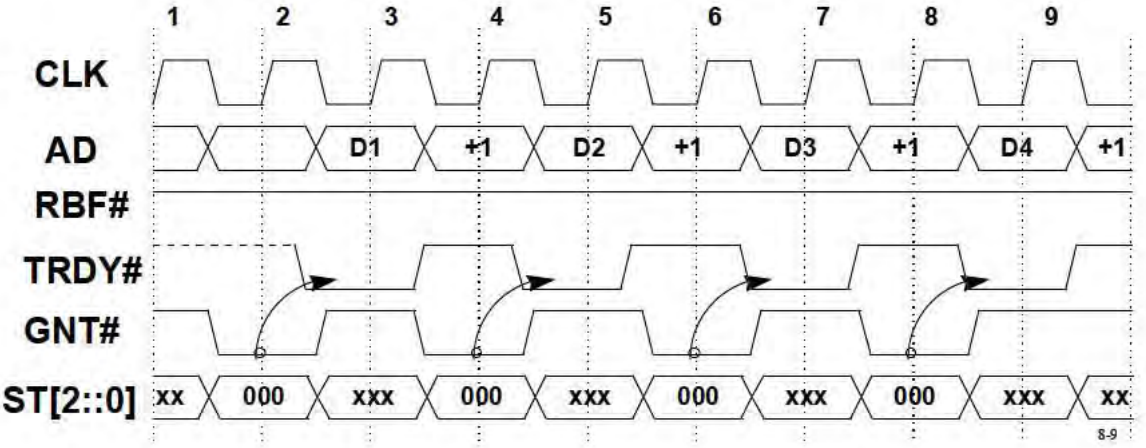


Figure 8-9 Minimum Delay of Back to Back Read Data

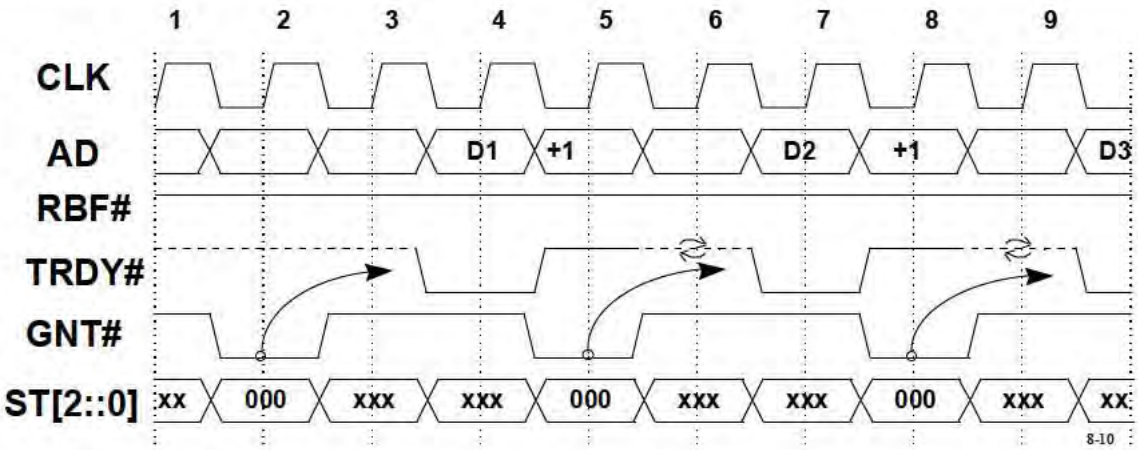


Figure 8-10 Maximum Delay of Back to Back Read Data

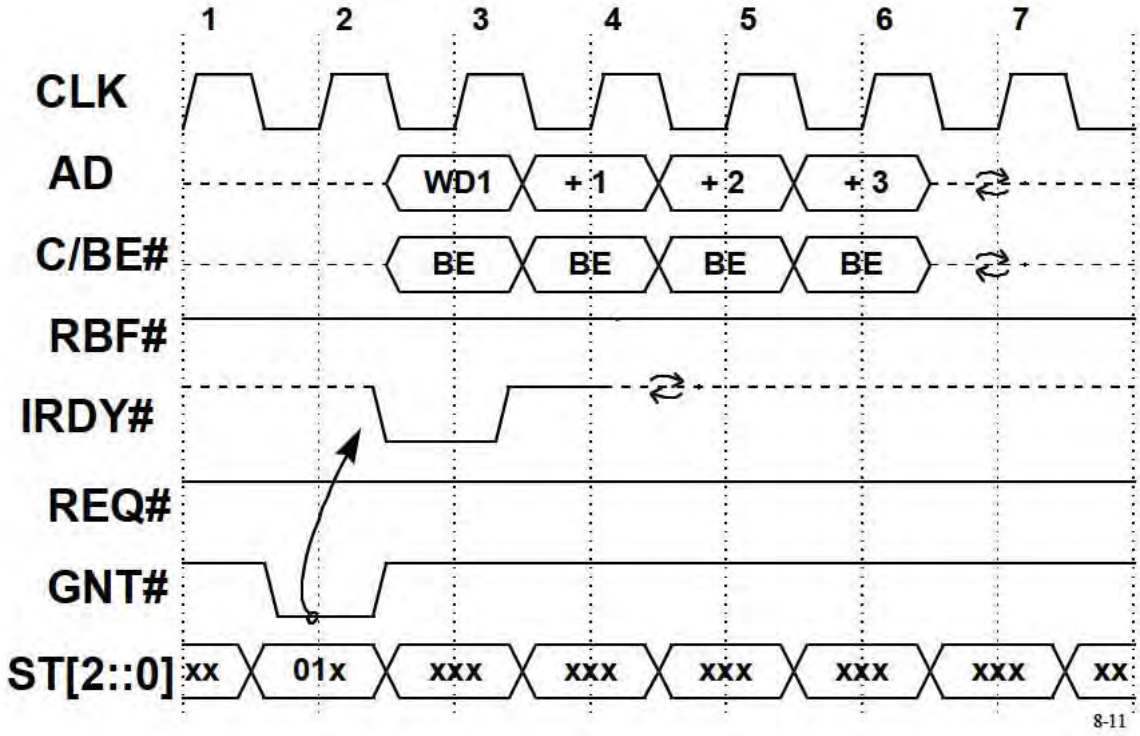


Figure 8-11 Minimum Delay by Master of Write Data

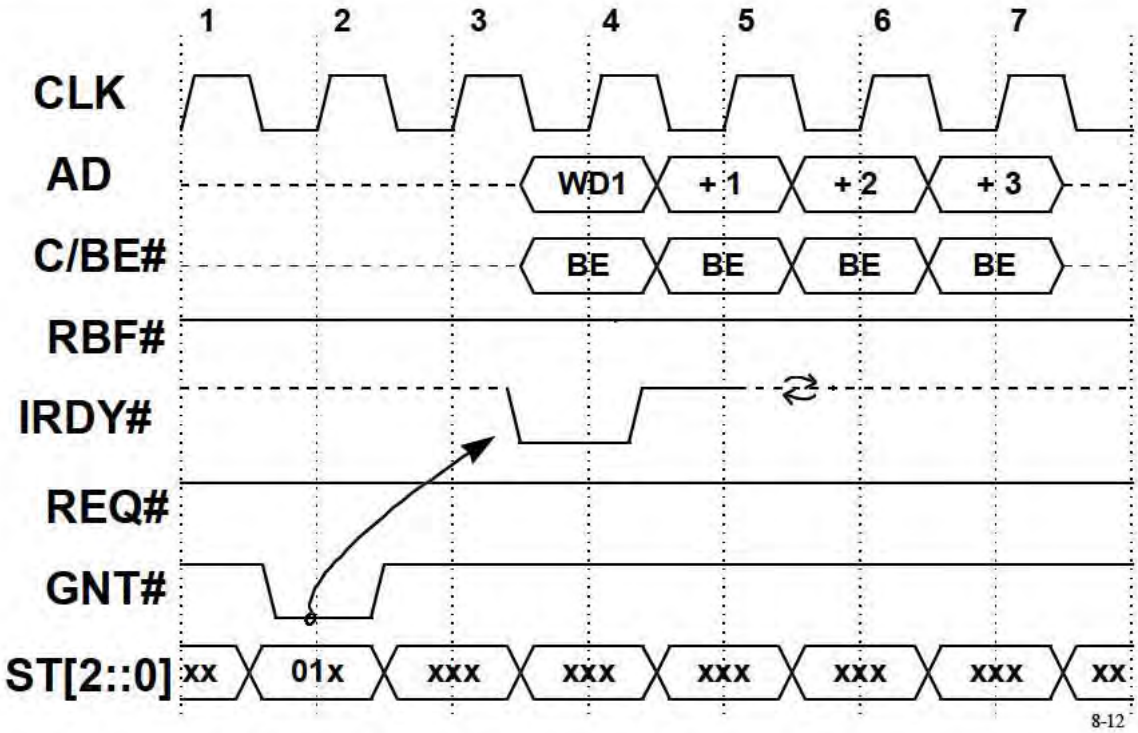


Figure 8-12 Maximum Delay by Master of Write Data

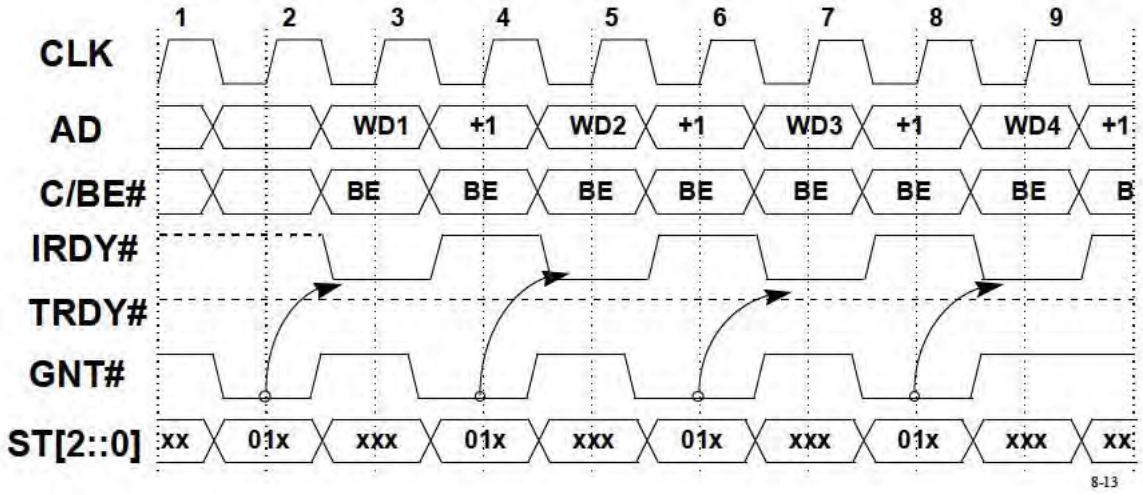


Figure 8-13 Minimum Delay by Master of Back to Back Write Data

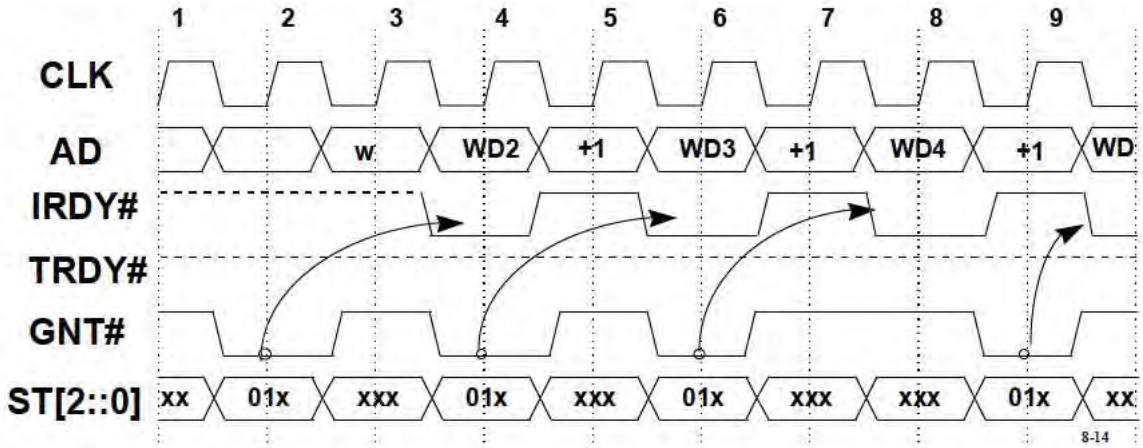


Figure 8-14 Master Delays Initial Write No Delay

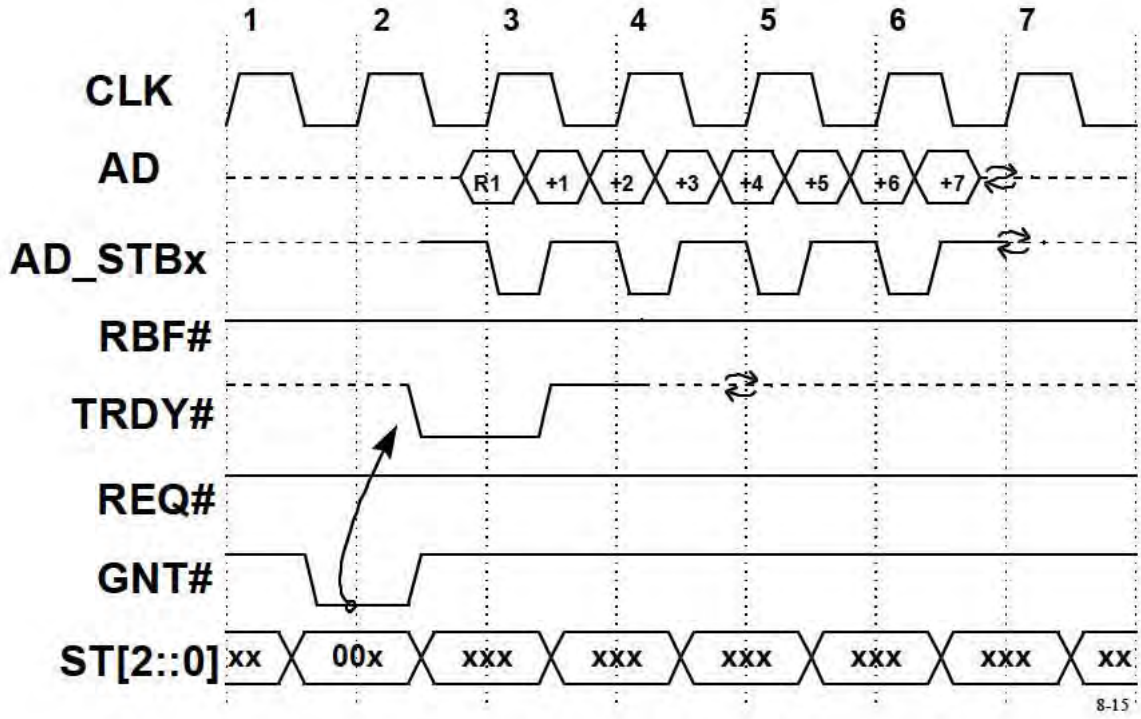


Figure 8-15 2x Read Data - No Delay

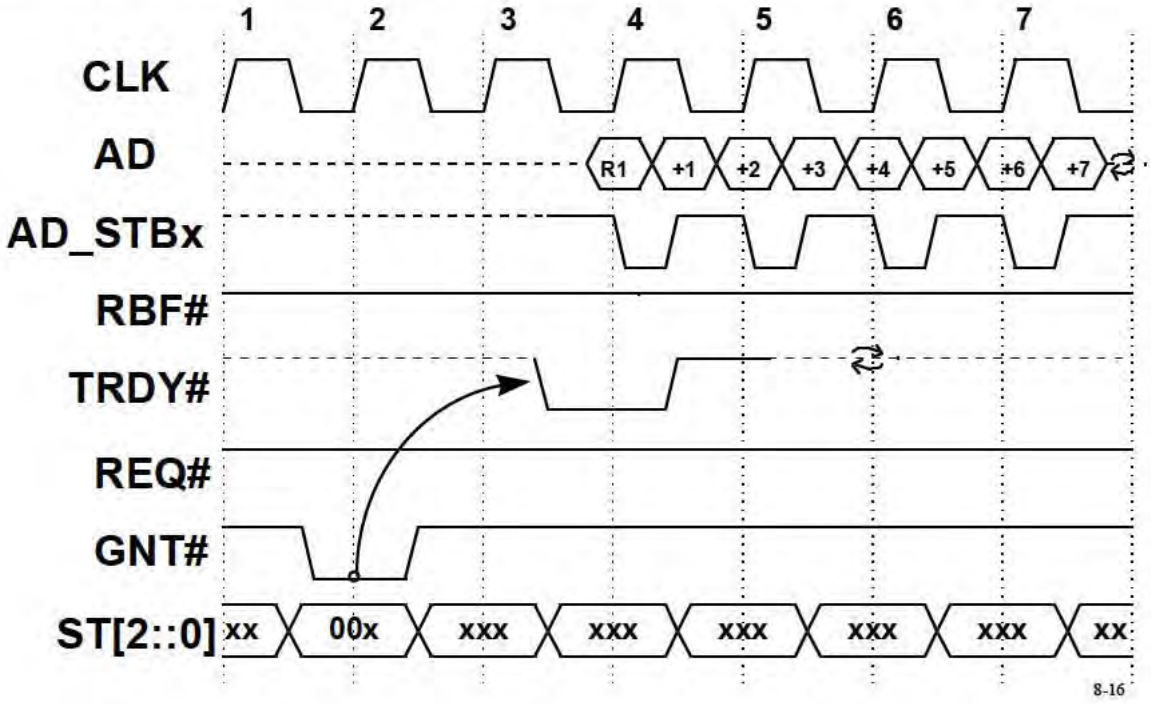


Figure 8-16 2x Read Data - With Delay

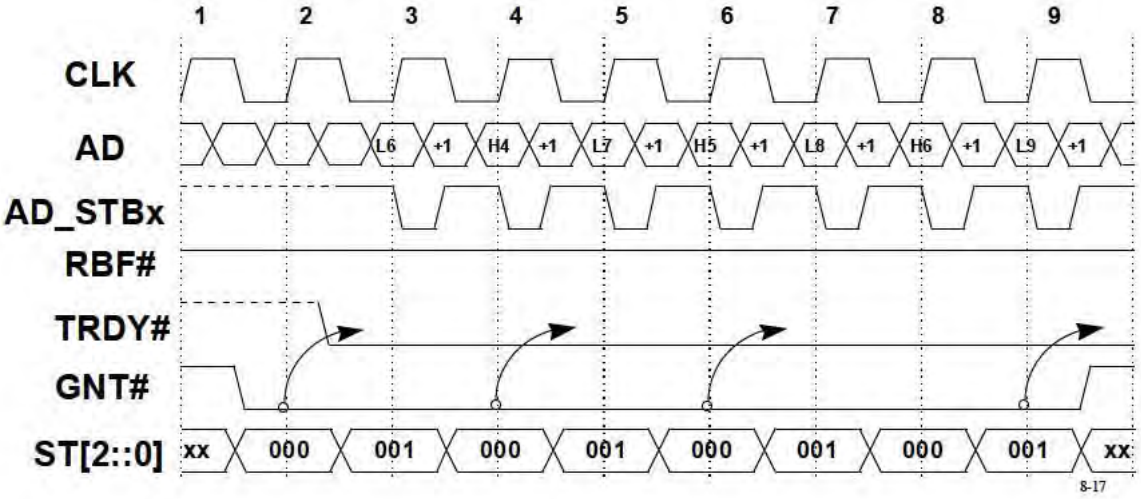


Figure 8-17 2x Back to Back Read Data - No Delay

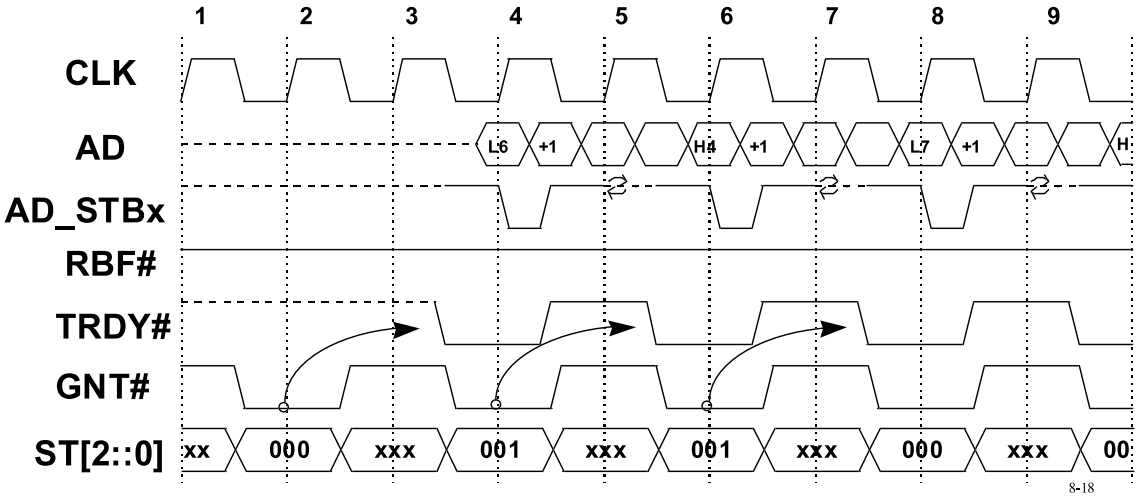


Figure 8-18 2x Back to Back Read Data - Maximum Delay

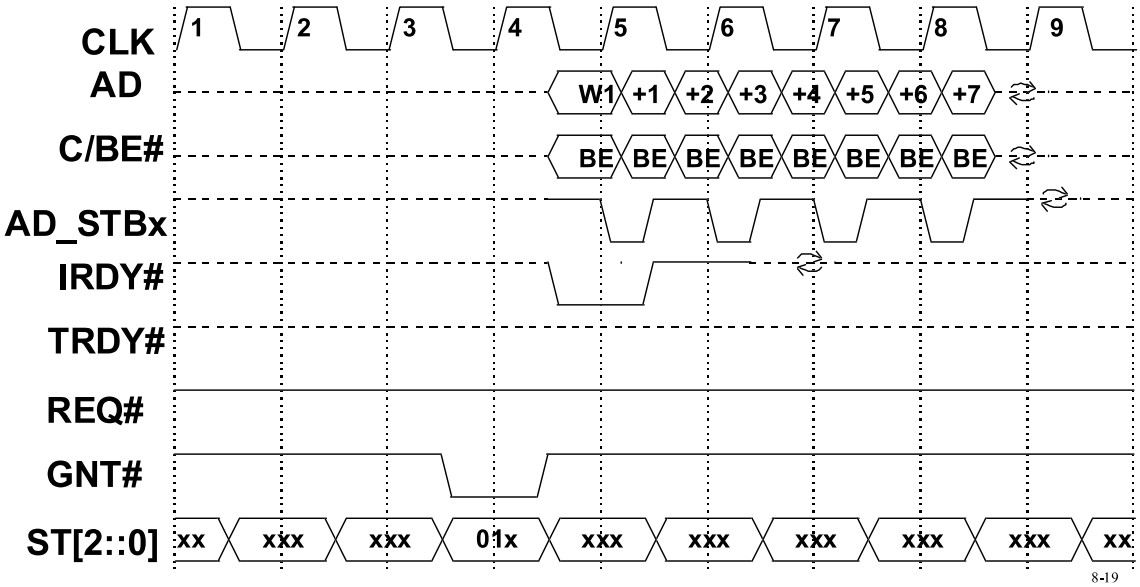


Figure 8-19 2x Write Data - No Delay

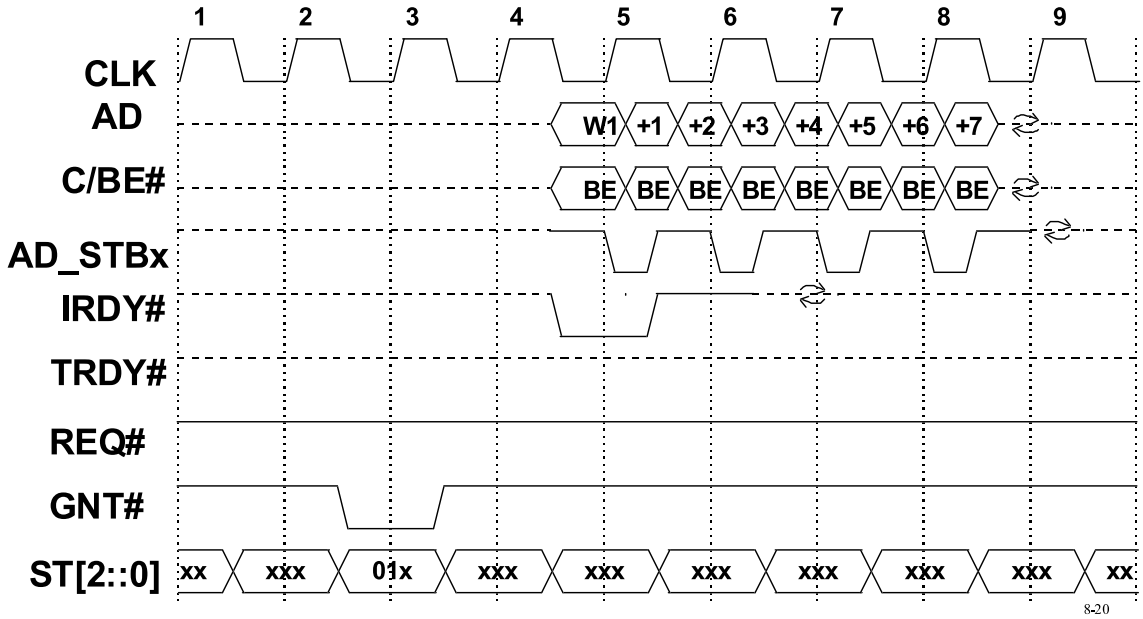


Figure 8-20 2x Write Data - Maximum Delay

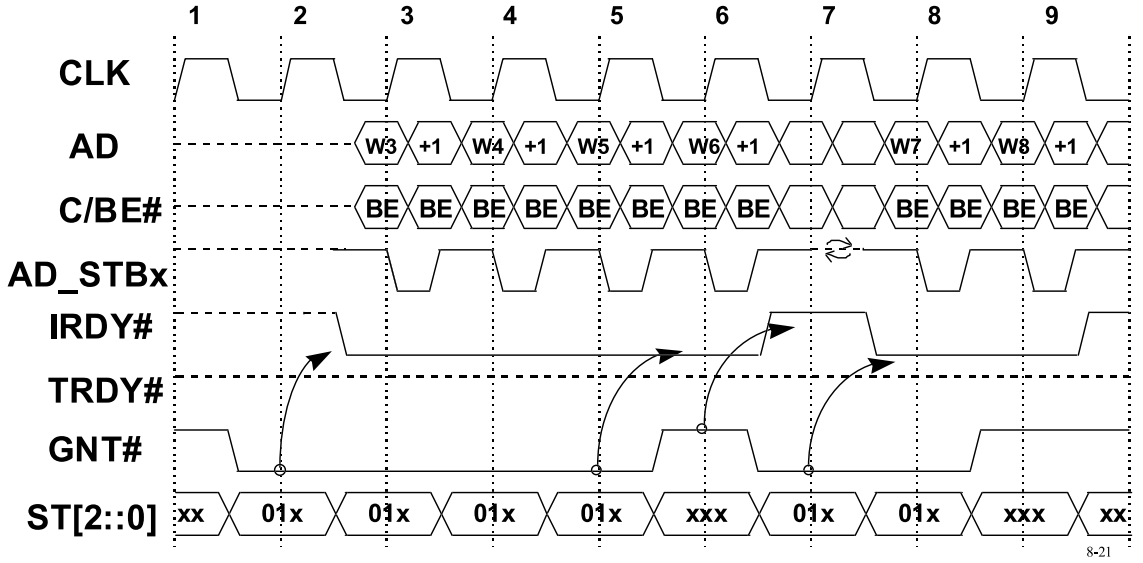


Figure 8-21 2x Back to Back Writes - No Delay

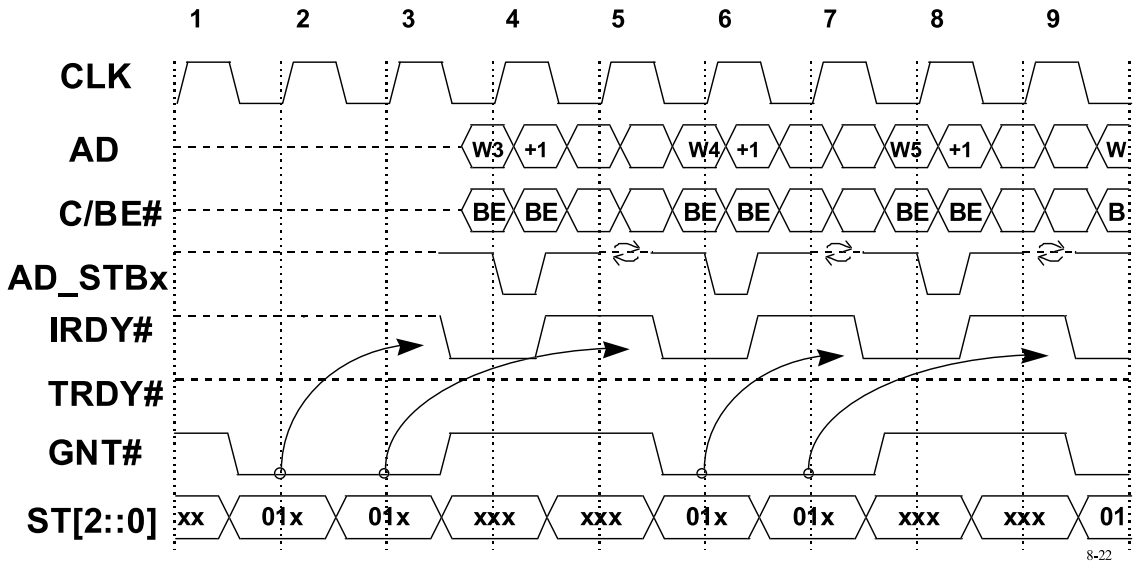


Figure 8-22 2x Back to Back Writes - Maximum Delay

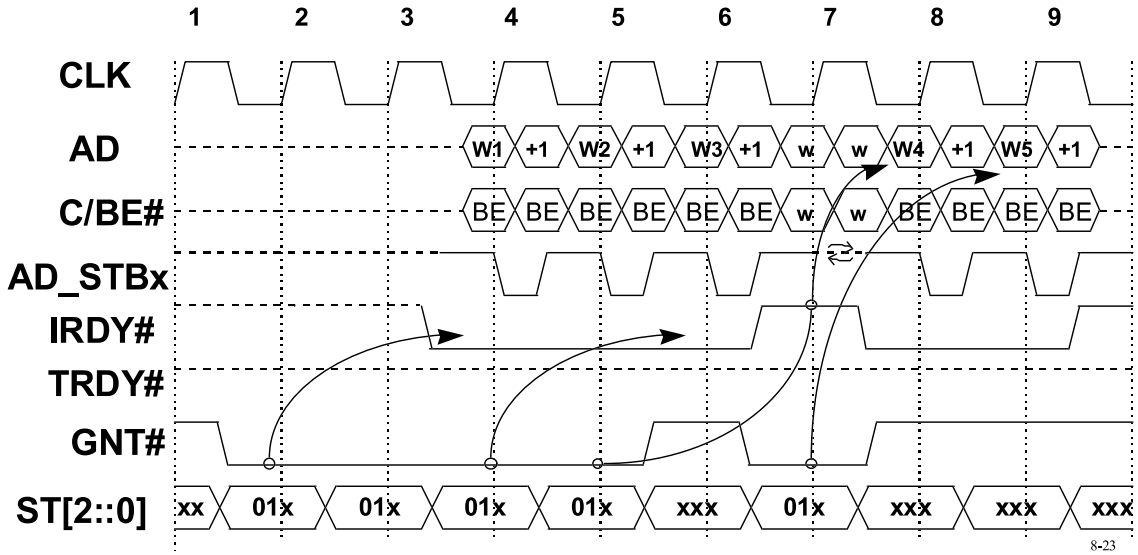


Figure 8-23 2x Writes, Initial Transaction with Delay, Subsequent Transactions No Delay

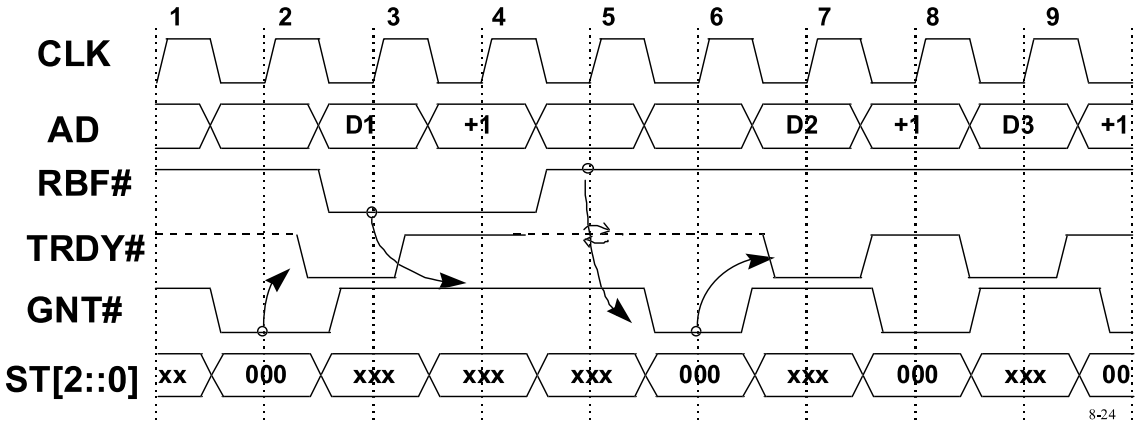


Figure 8-24 Master Data Buffer Full - Minimum Delay for Read Data

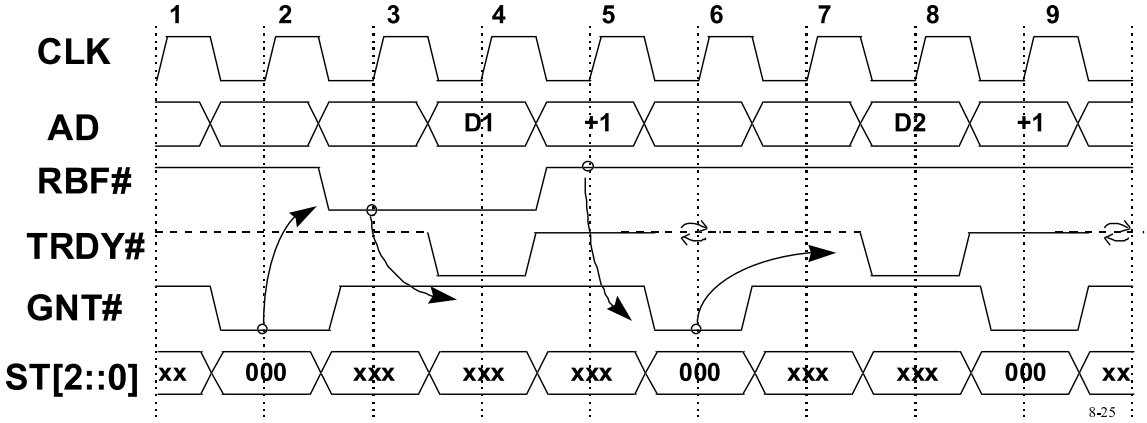


Figure 8-25 Master Data Buffer Full - Maximum Delay for Read Data

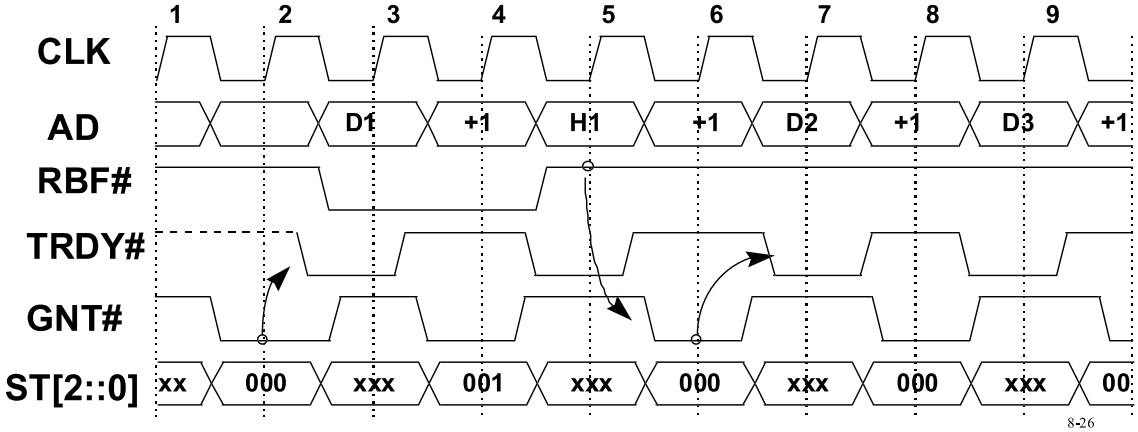


Figure 8-26 RBF# Asserted, HP Read Data Returned

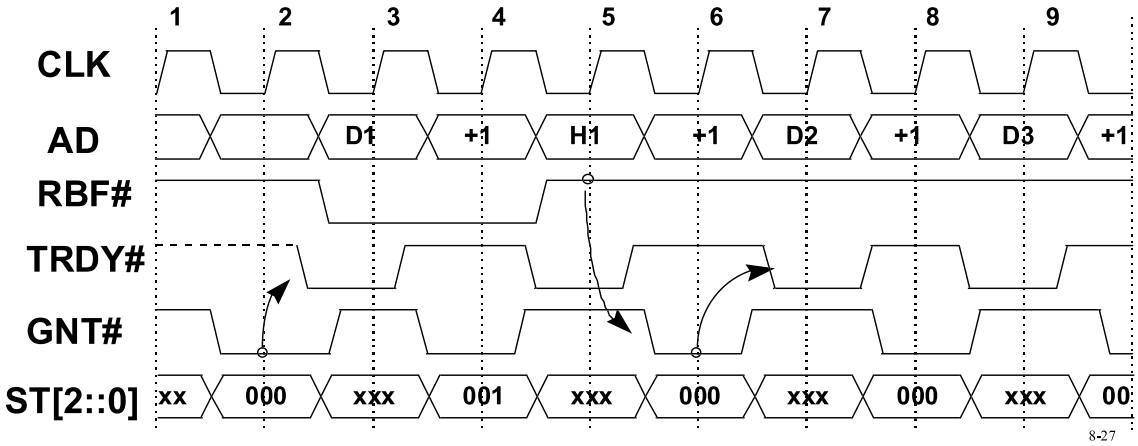


Figure 8-27 RBF# Asserted, Maximum Delay by Target, HP Read Data Returned

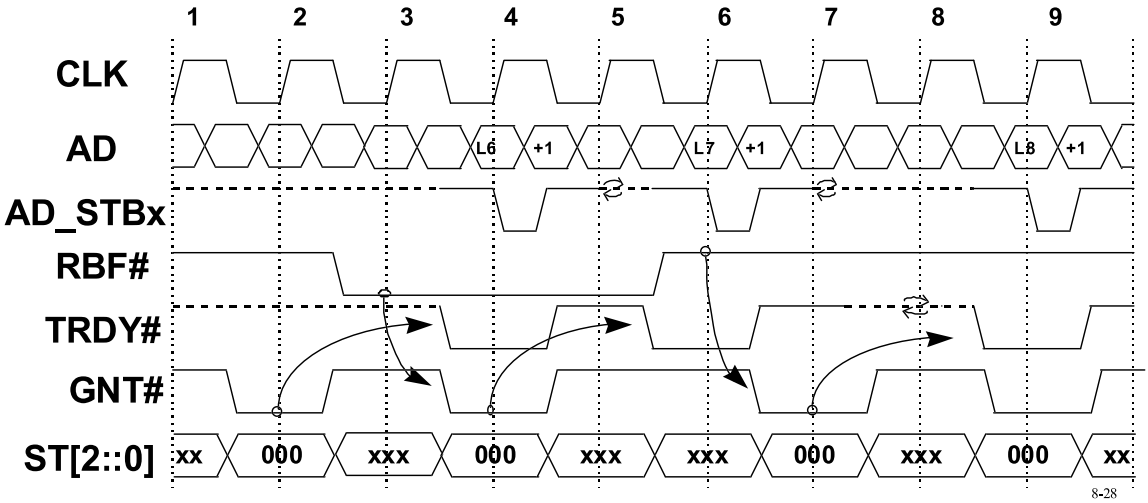


Figure 8-28 2x Read Data, RBF# Asserted, Maximum Delay By Target

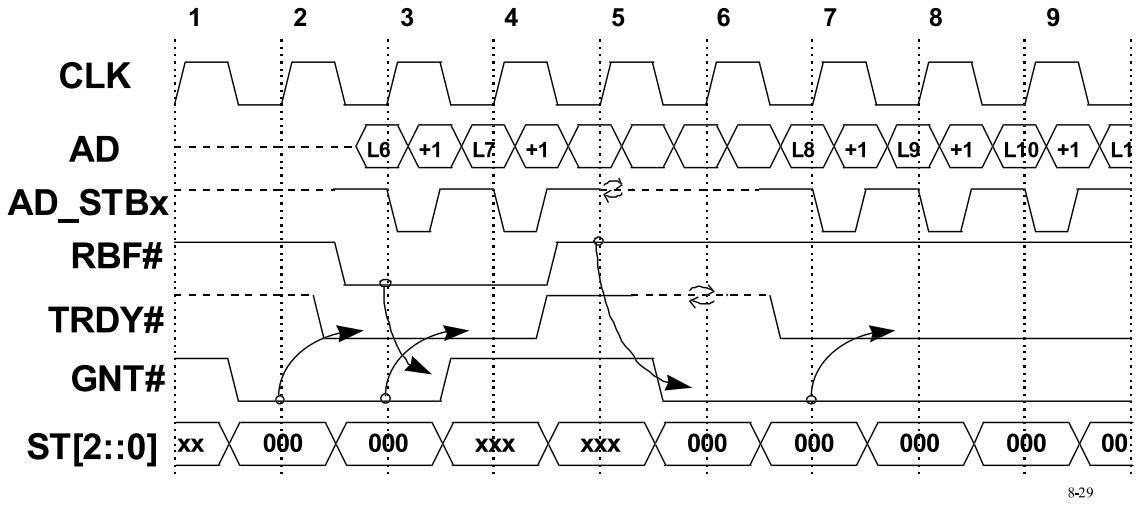


Figure 8-29 2x Read, RBF# Asserted, No HP Read Data

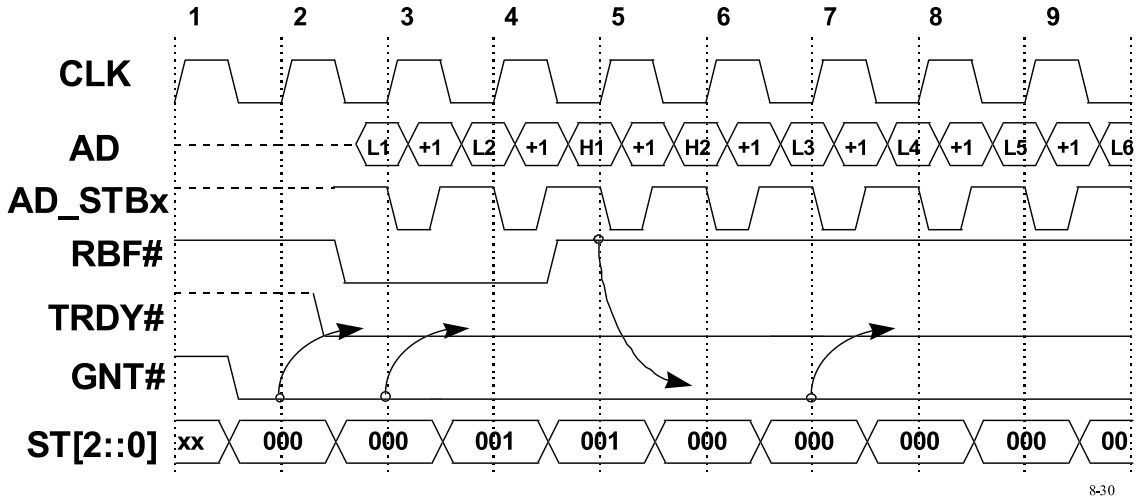


Figure 8-30 2x Read Data, with Delay, RBF# Asserted, HP Read Data Returned

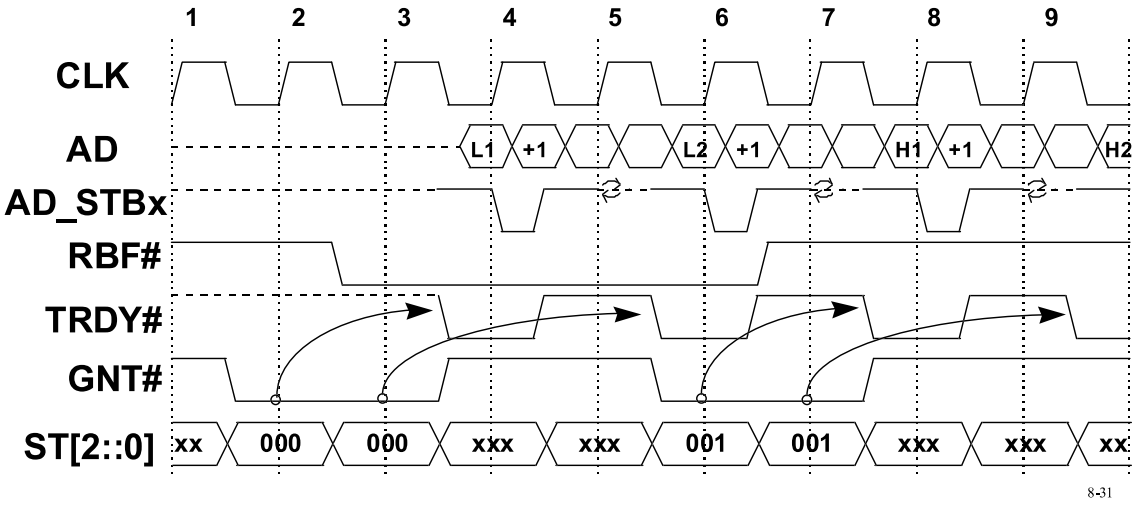


Figure 8-31 2x Read Data, RBF# Asserted, HP Read Data Delayed

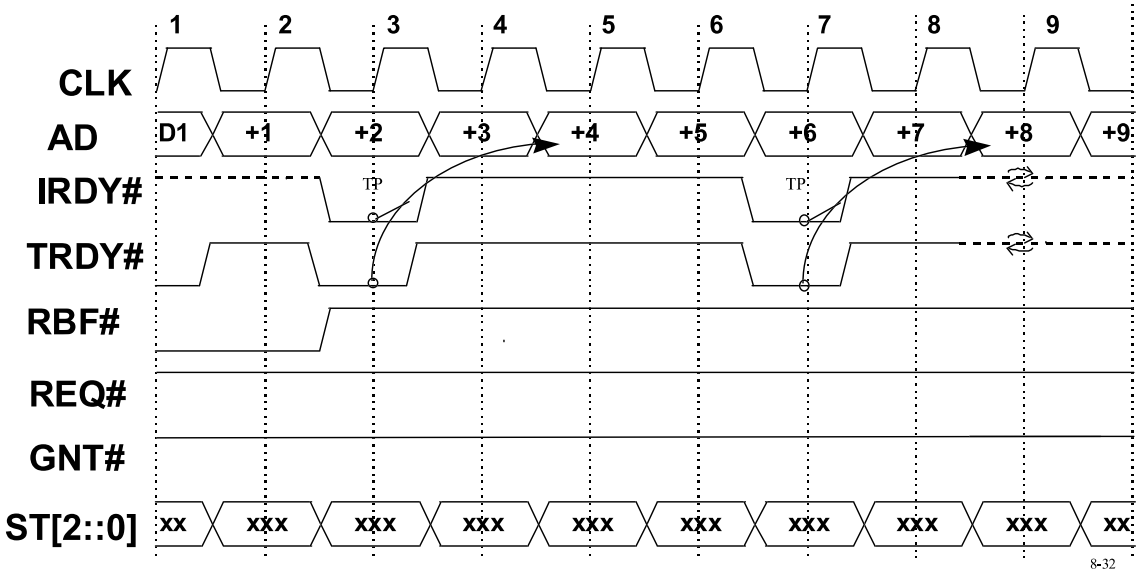


Figure 8-32 Read Data, No Delay Subsequent Block

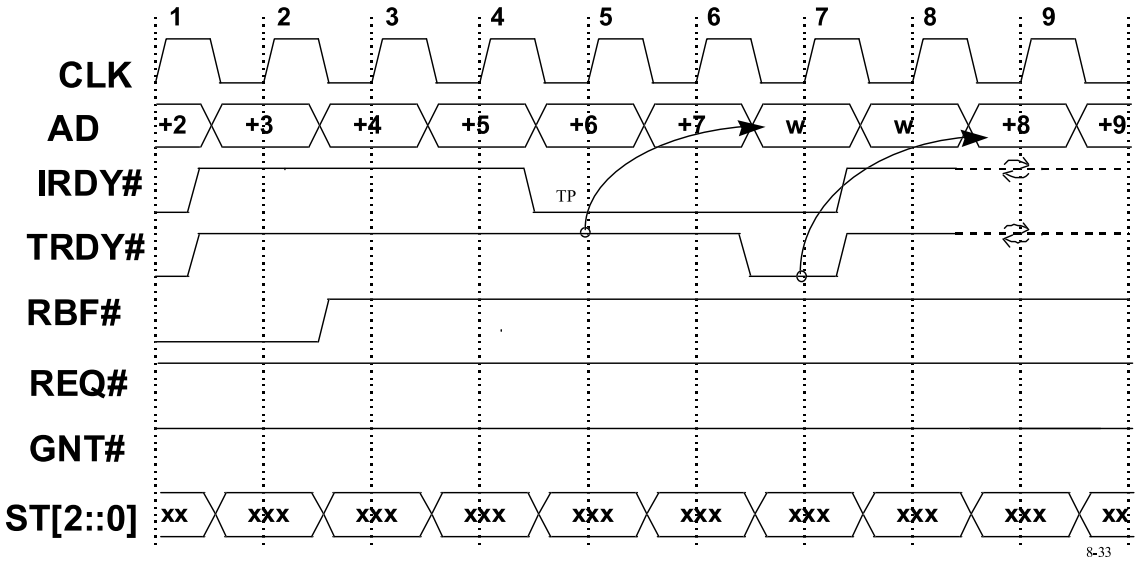


Figure 8-33 Read Data, Master Ready - Target Delays Subsequent Block

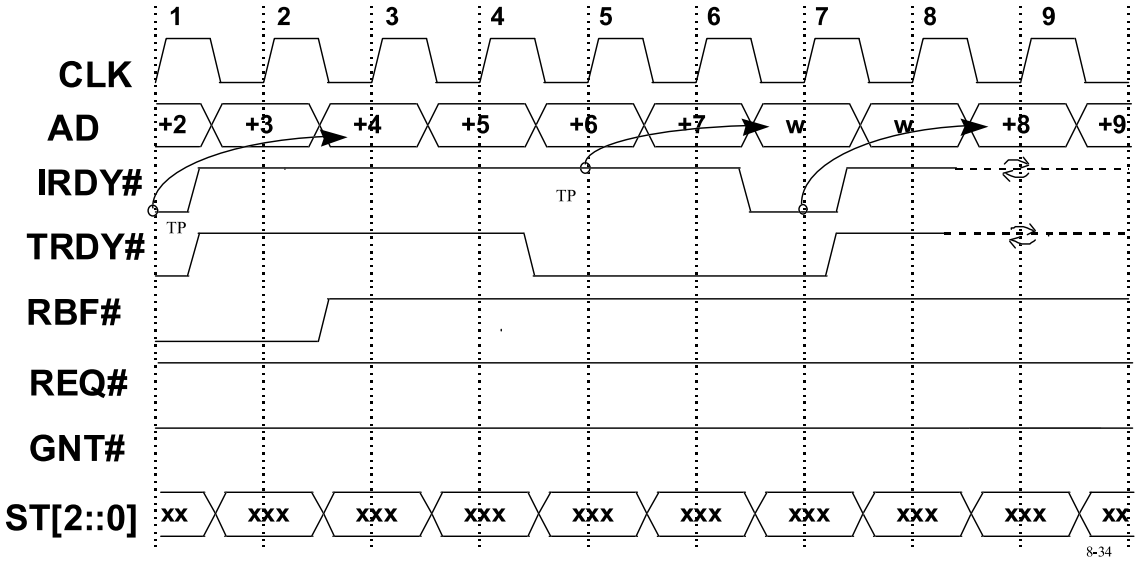


Figure 8-34 Read Data, Target Ready - Master Delays Subsequent Block

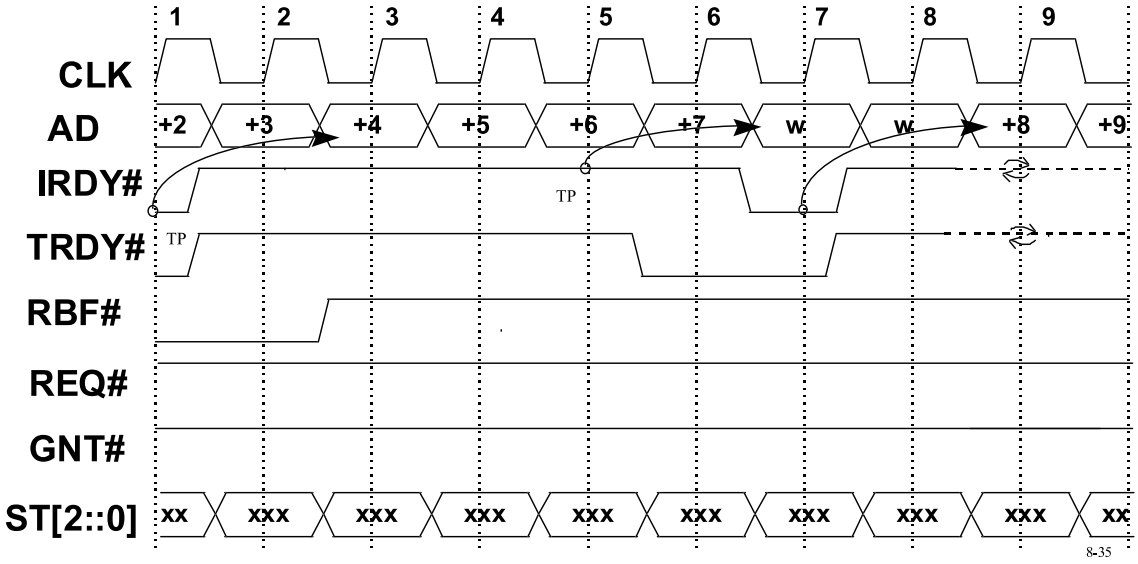


Figure 8-35 Read Data, Target Delays 1 clock - Master Delays 2 clocks Subsequent Block

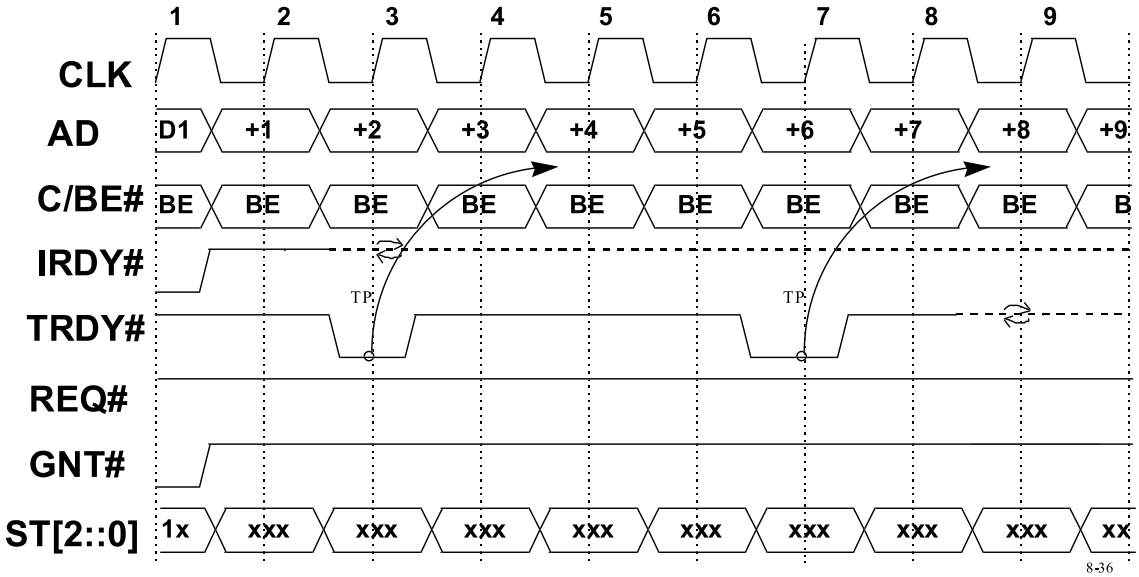


Figure 8-36 Write Data, No Delay Subsequent Block

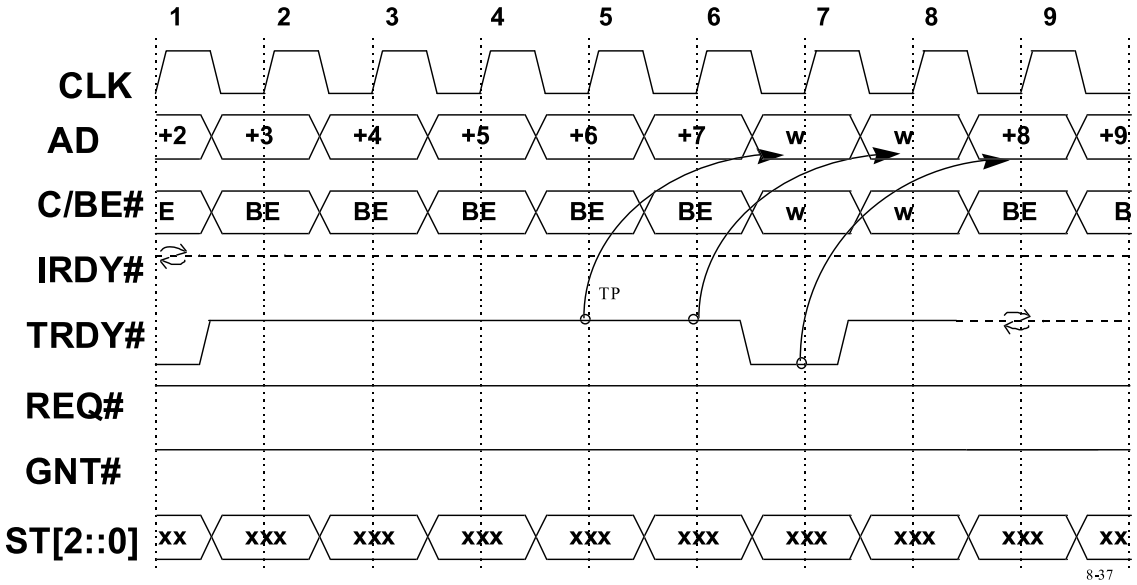


Figure 8-37 Write Data, Target Delays Subsequent Block 2 Clocks

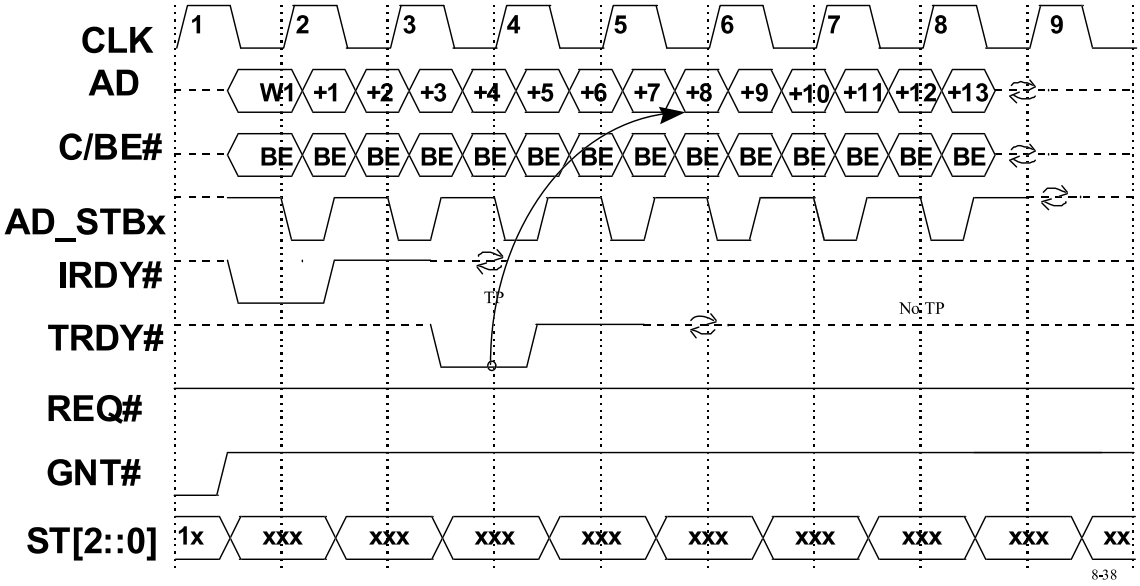


Figure 8-38 2x Write No Delay Initial or Subsequent Block

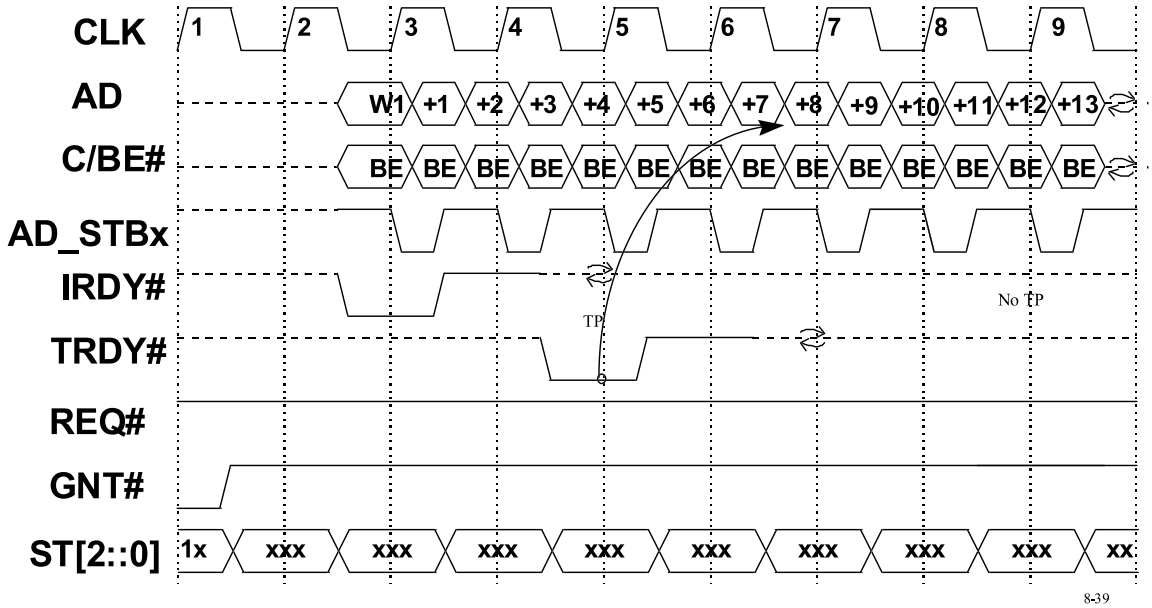


Figure 8-39 2x Write, Initial Delay, No Delay Subsequent Block

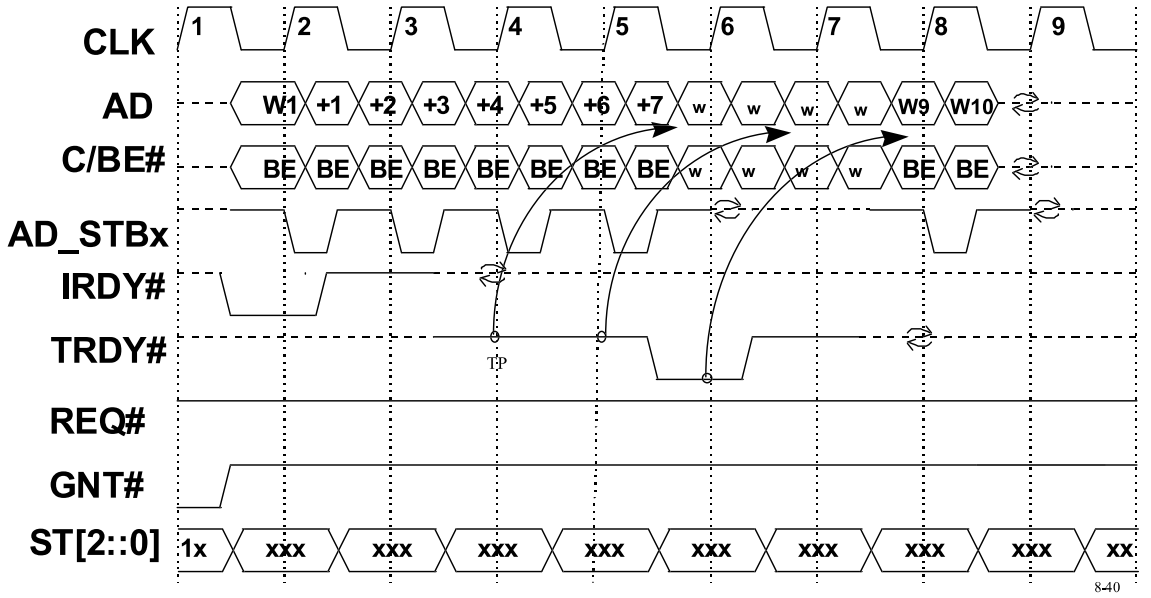


Figure 8-40 2x Write, No Initial Delay, 2 clocks Delay Subsequent Block

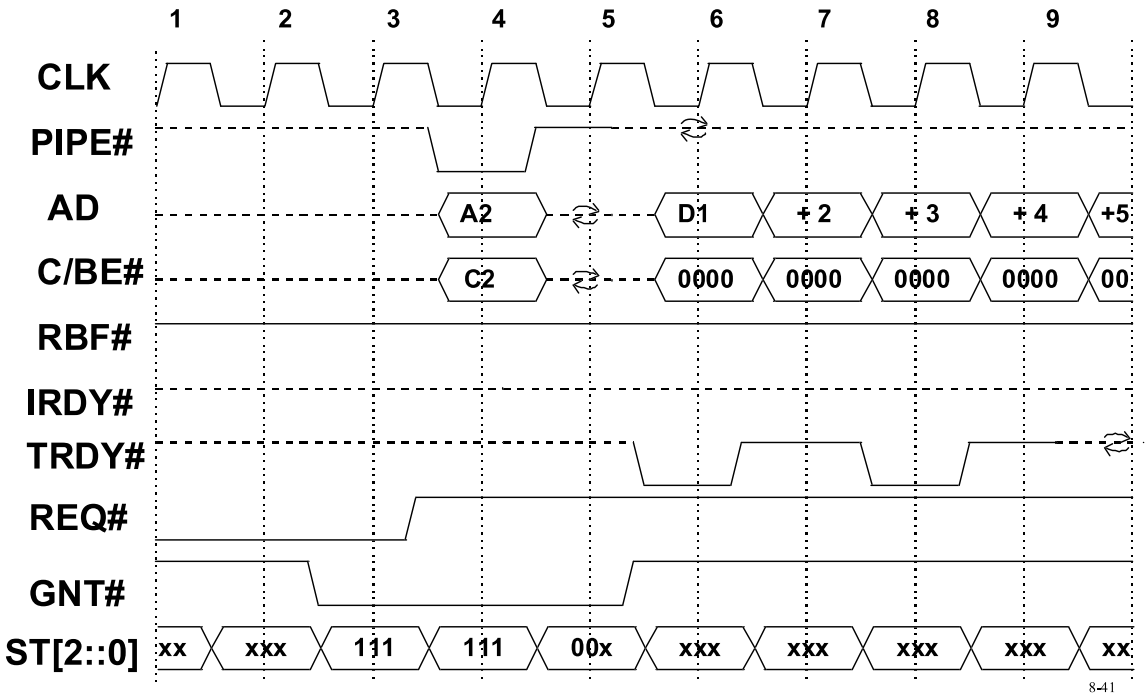


Figure 8-41 Earliest Read Data after Request

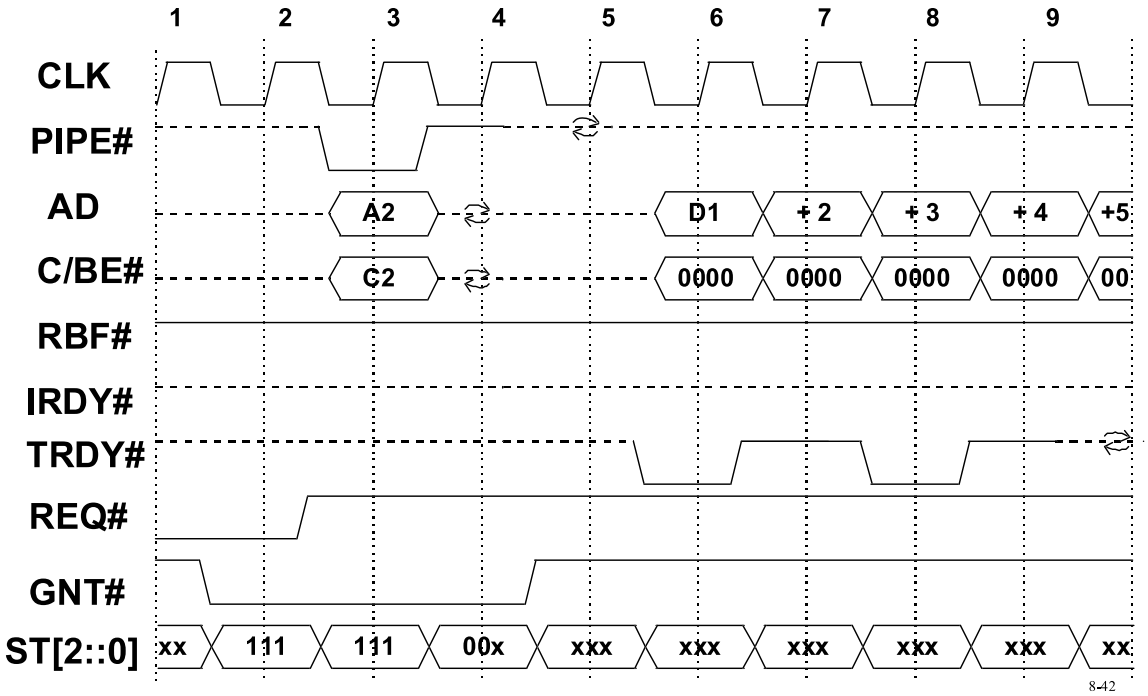


Figure 8-42 Maximum Delay Read Data after Request

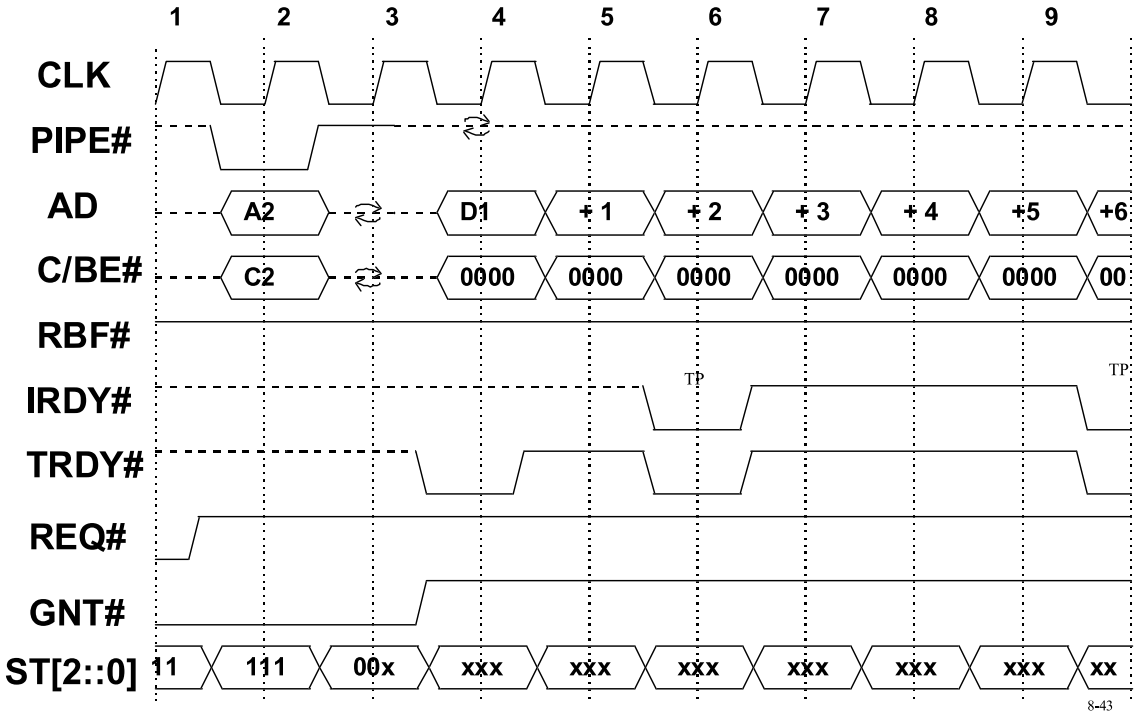


Figure 8-43 Request followed by Long Data Read

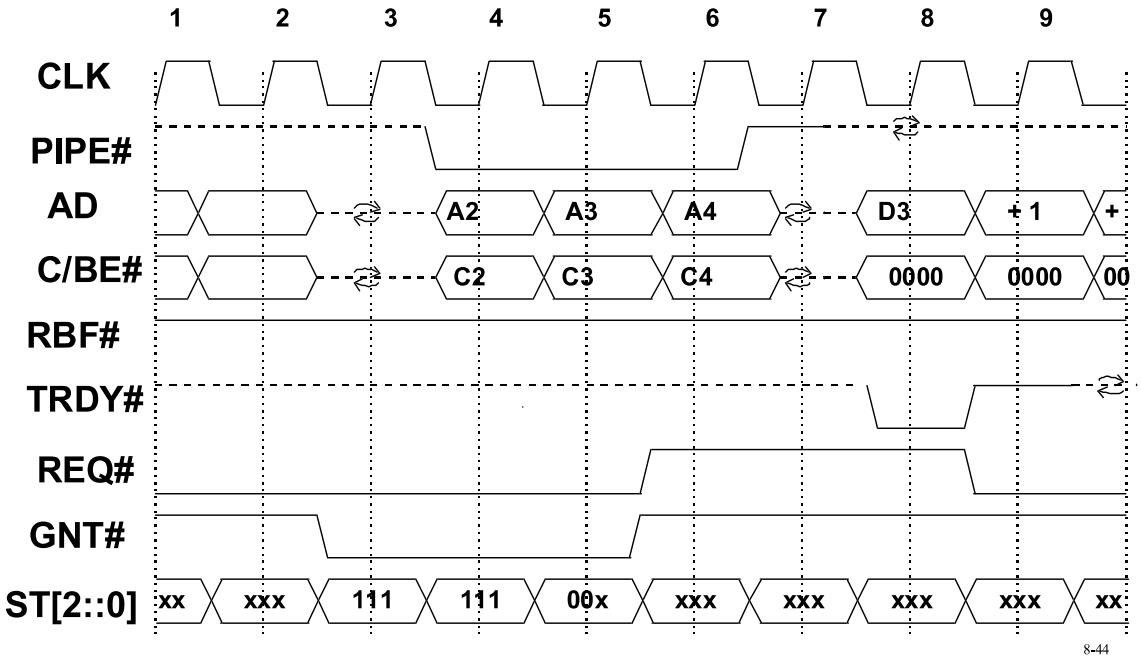


Figure 8-44 Early GNT# for Read Data

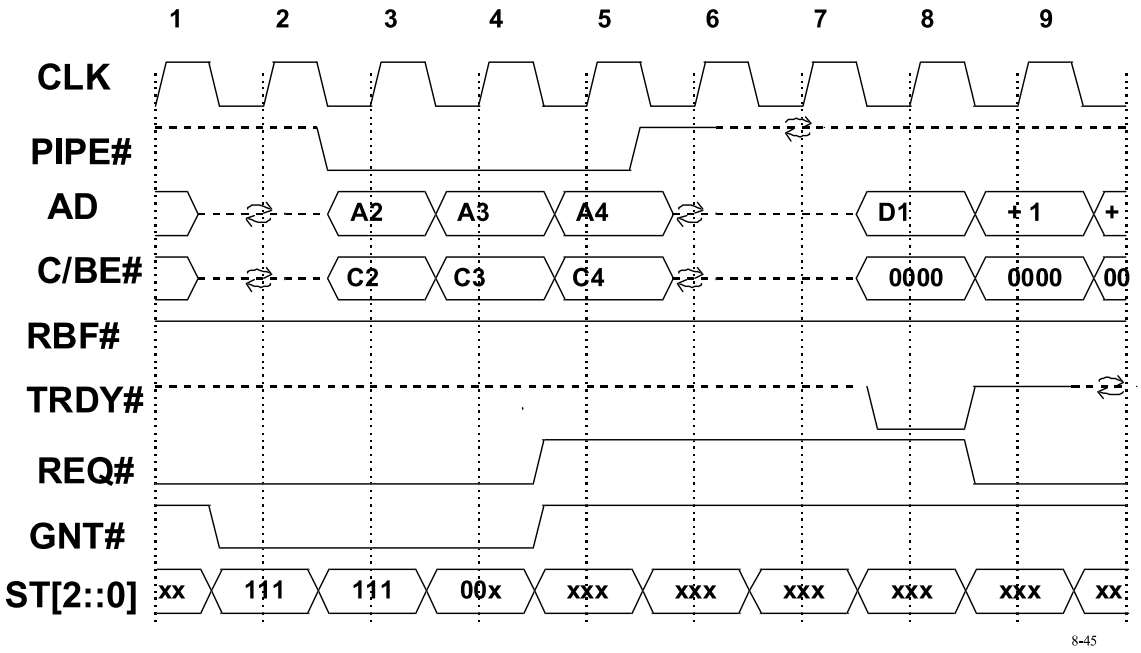


Figure 8-45 Early GNT# for Read Data, Read Data Delayed

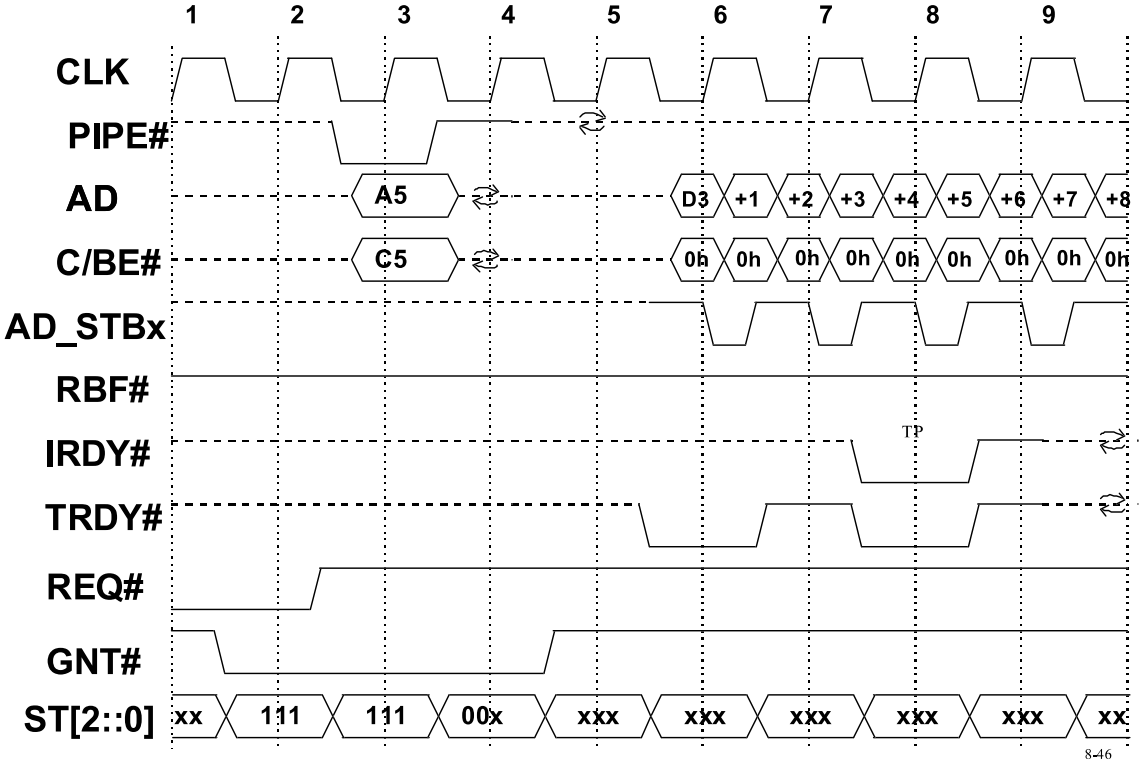


Figure 8-46 2x Read, Early GNT#, Delayed Read Data

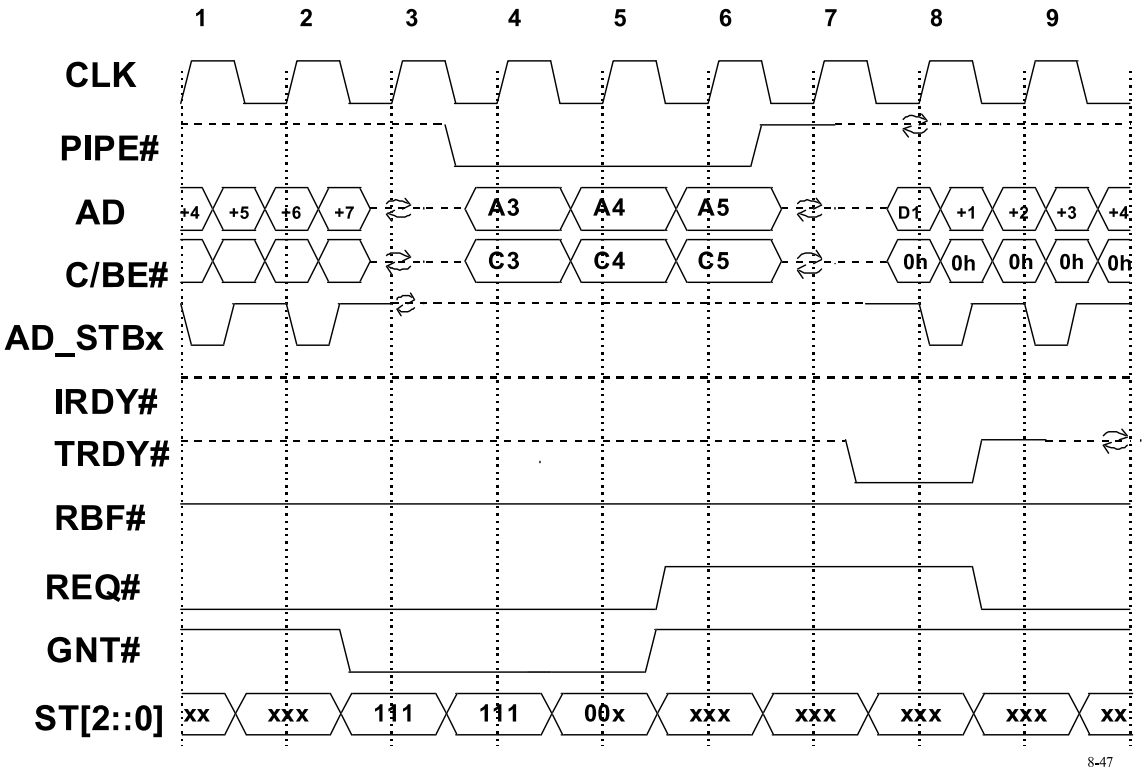


Figure 8-47 Data Transfers Intervened with Requests

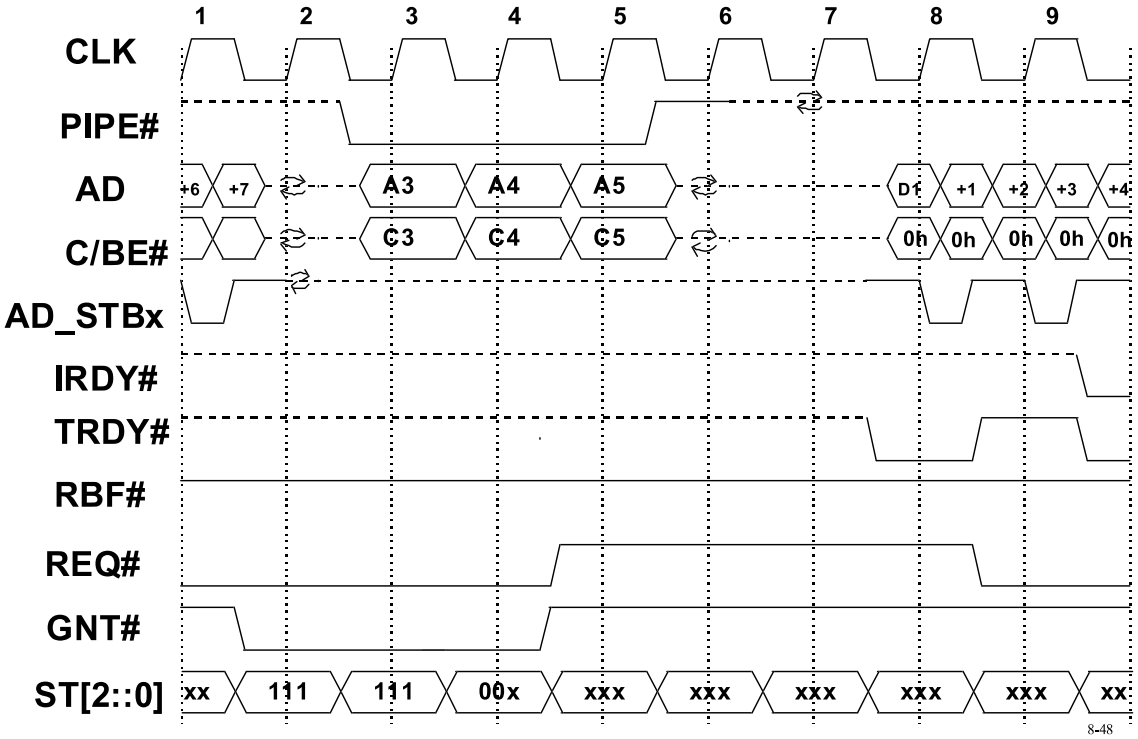


Figure 8-48 Intervened Request, Subsequent 2x Read Data Delayed

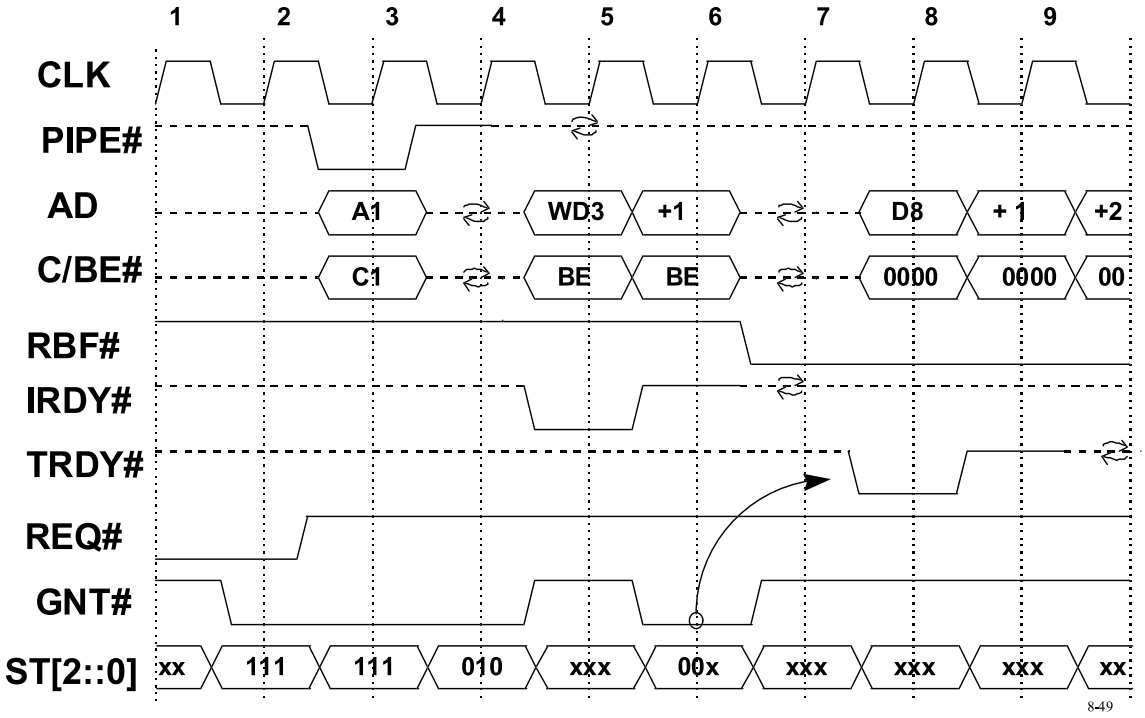


Figure 8-49 Request followed by Write Data, followed by Read Data

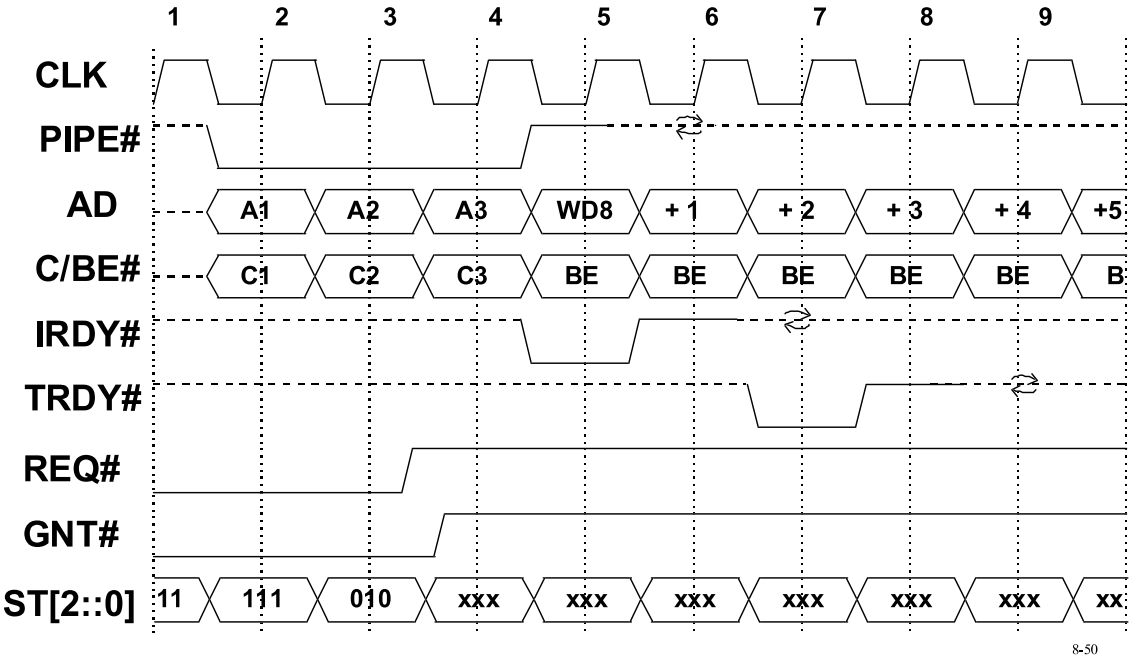


Figure 8-50 Multiple Requests followed by Write Data - No Turn-Around Cycle

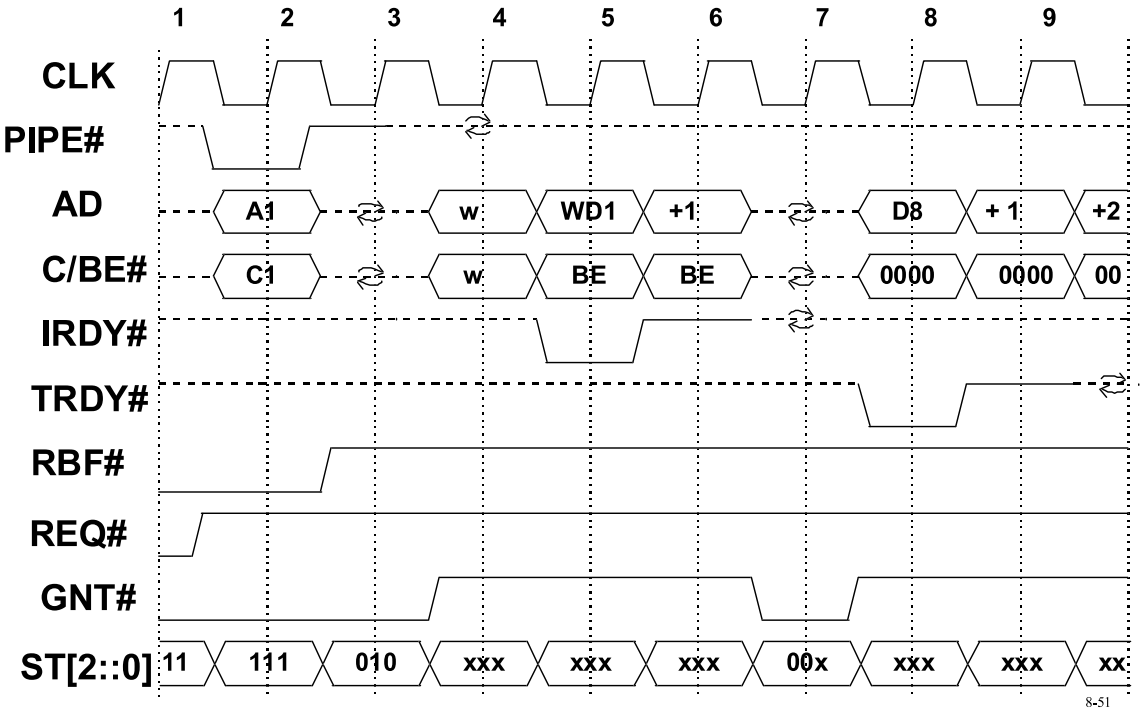


Figure 8-51 Request followed by Write Data Delayed, followed by Read Data - No Delay

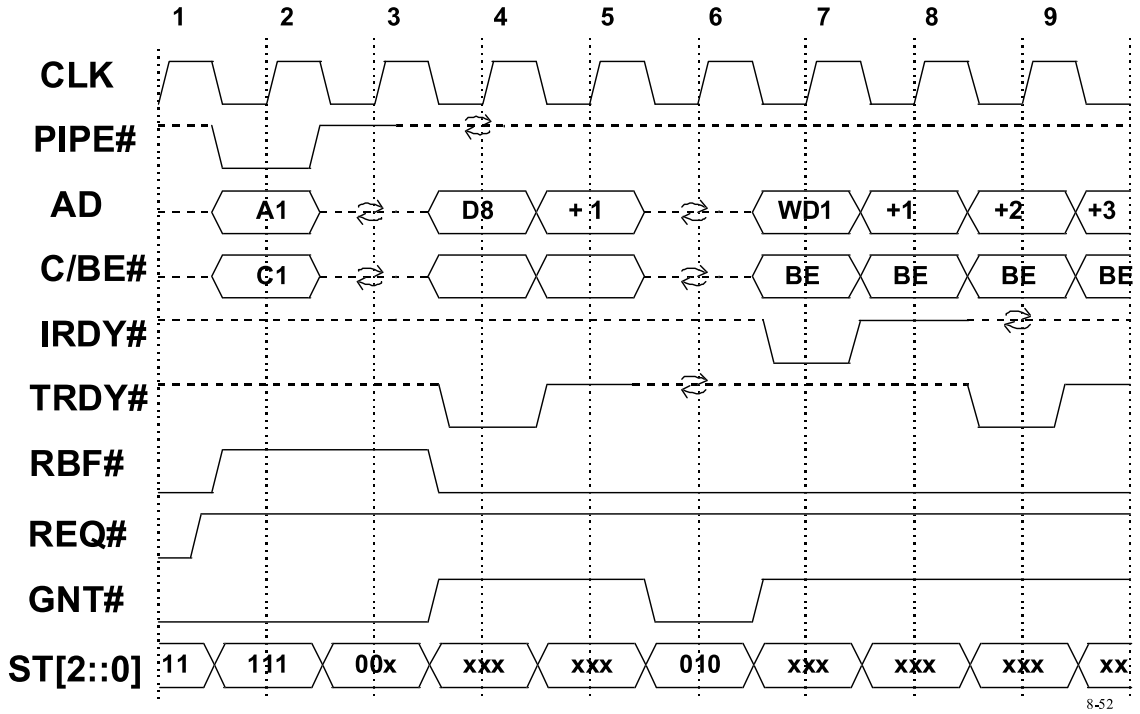


Figure 8-52 Request followed by Read Data, followed by Write Data

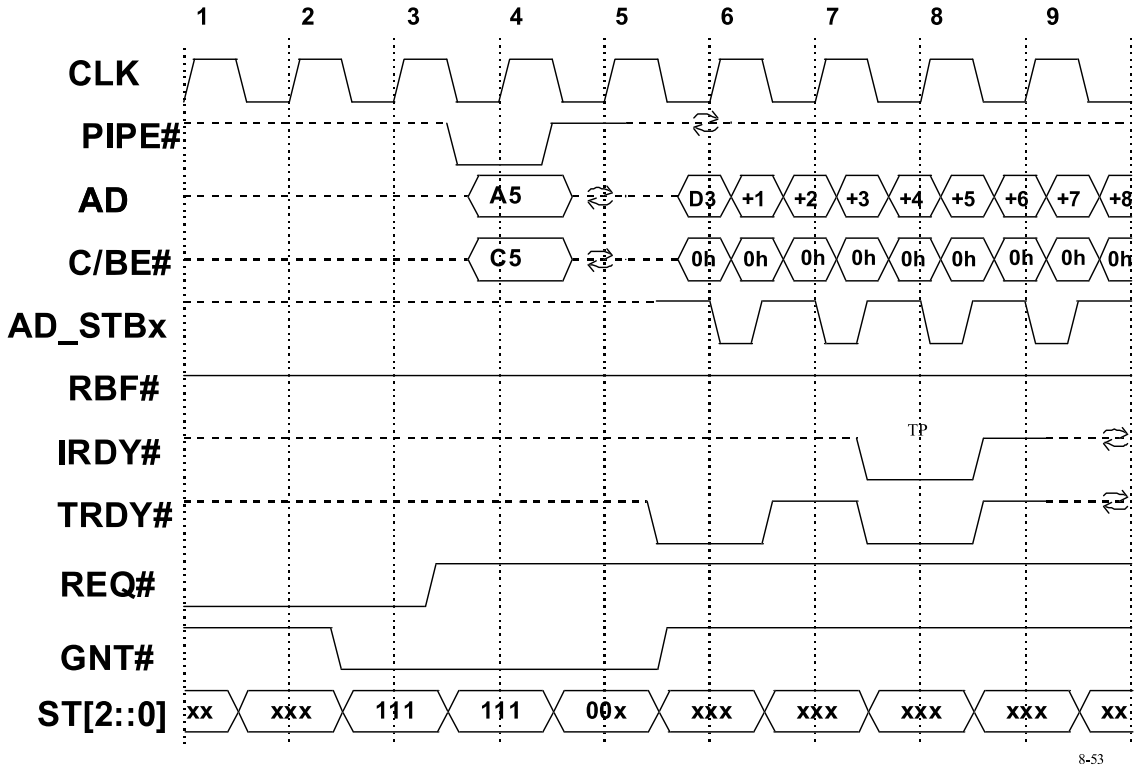


Figure 8-53 Request followed by 2x Read Data No Delay

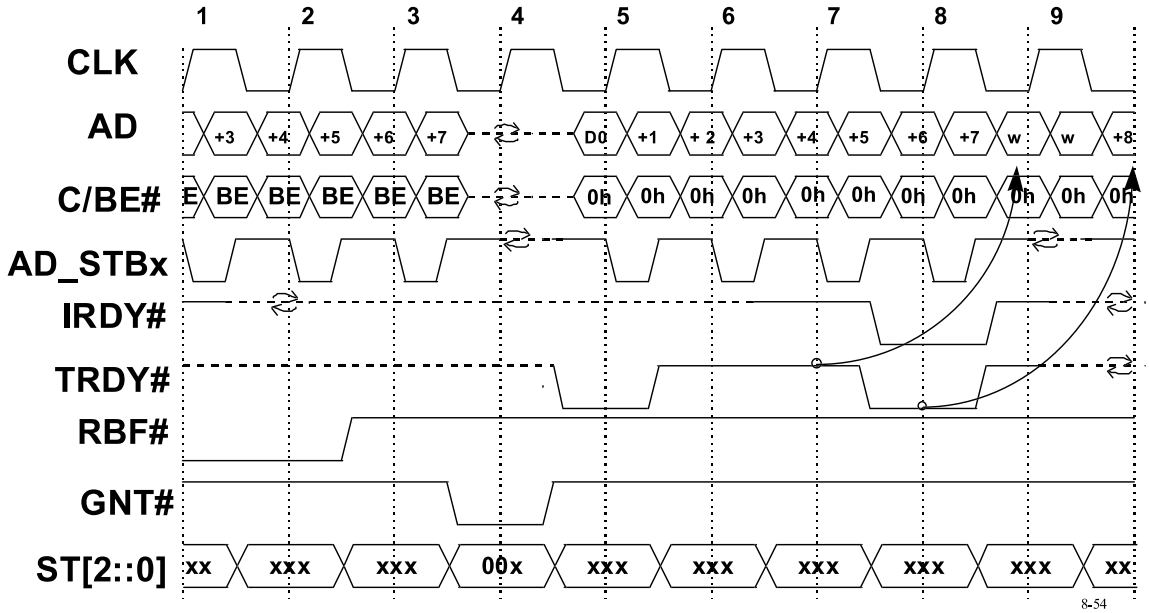


Figure 8-54 2x Write followed by 2x Read with Subsequent Block Delayed

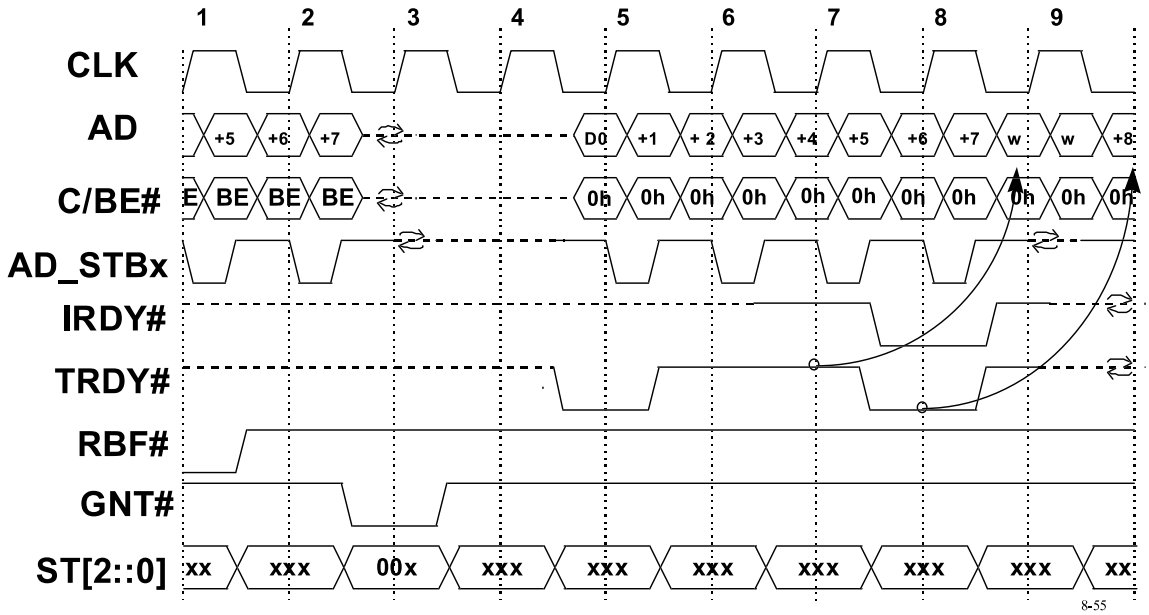


Figure 8-55 2x Write followed by 2x Read with Initial and Subsequent Blocks Delayed

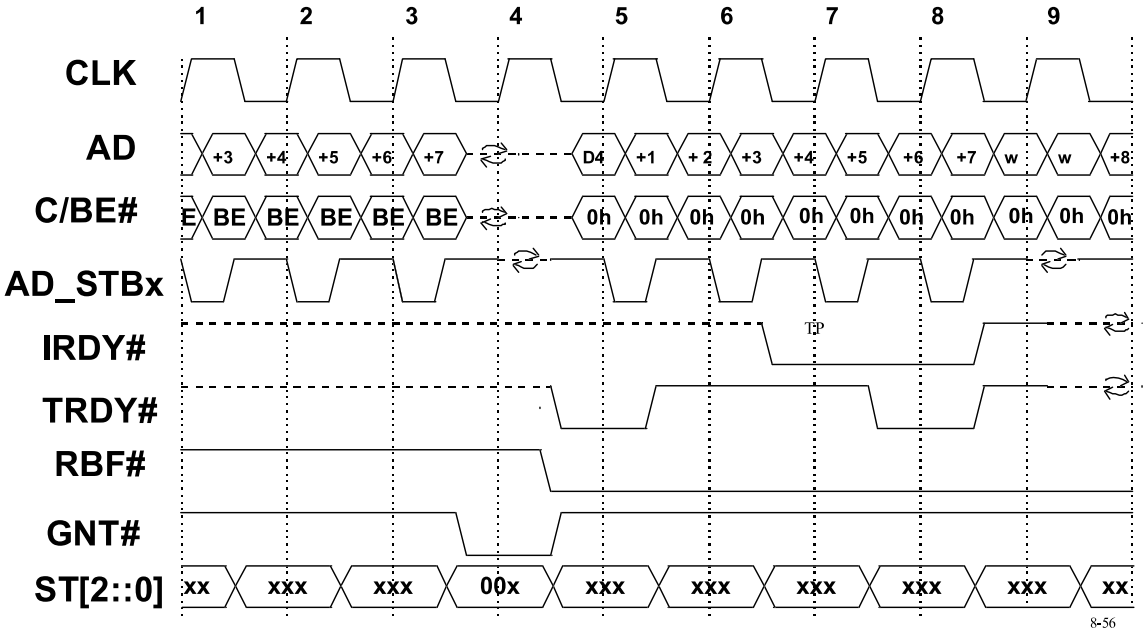


Figure 8-56 2x Write, 2x Read No Initial Delay, Subsequent Block Delayed by Target

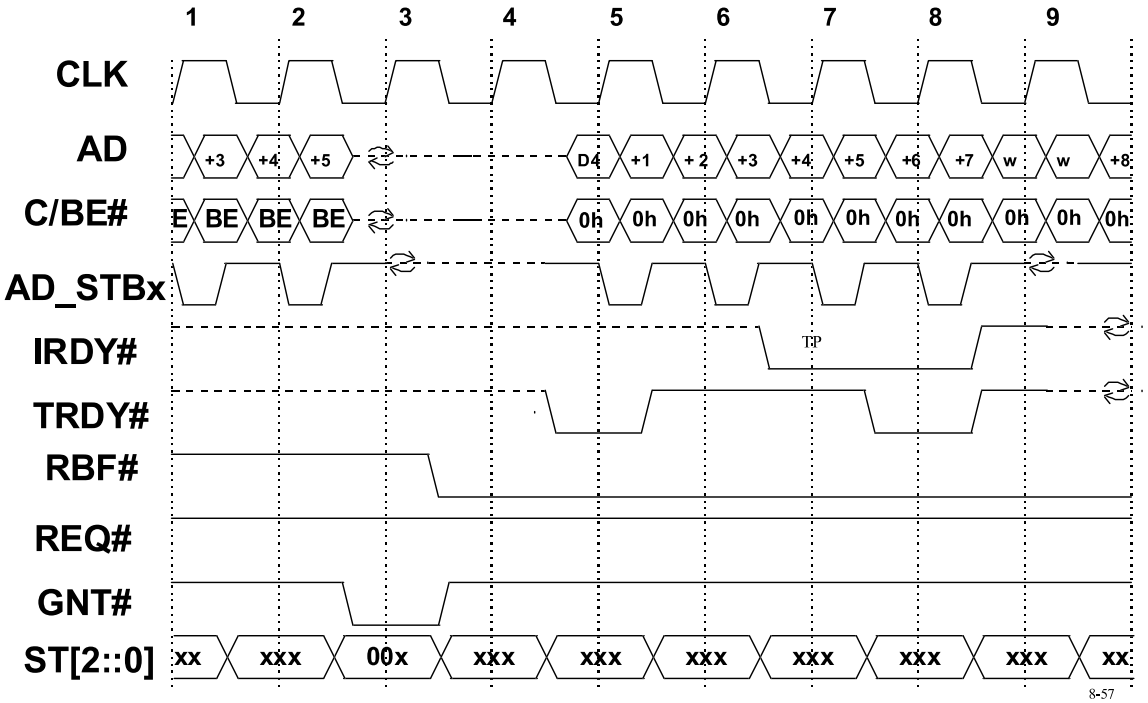


Figure 8-57 2x Write followed by 2x Read, Initial and Subsequent Block Delayed by Target

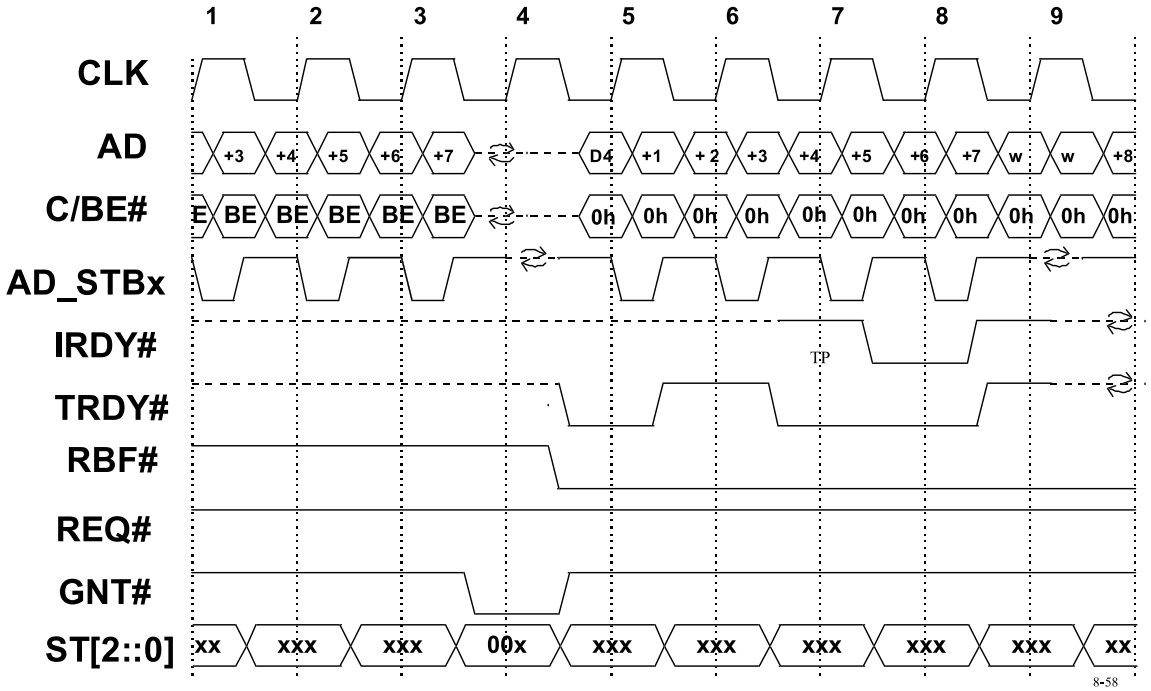


Figure 8-58 2x Write followed by 2x Read, Master Delays Subsequent Block

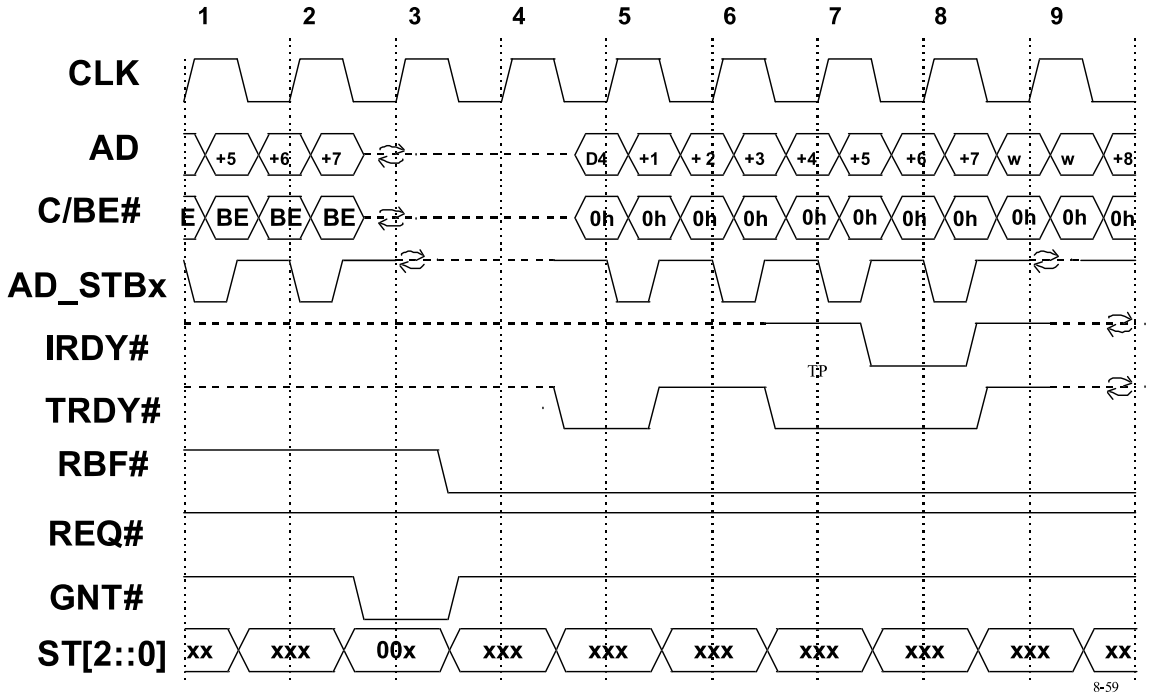


Figure 8-59 2x Write followed by 2x Read Initial and Subsequent Blocks delayed

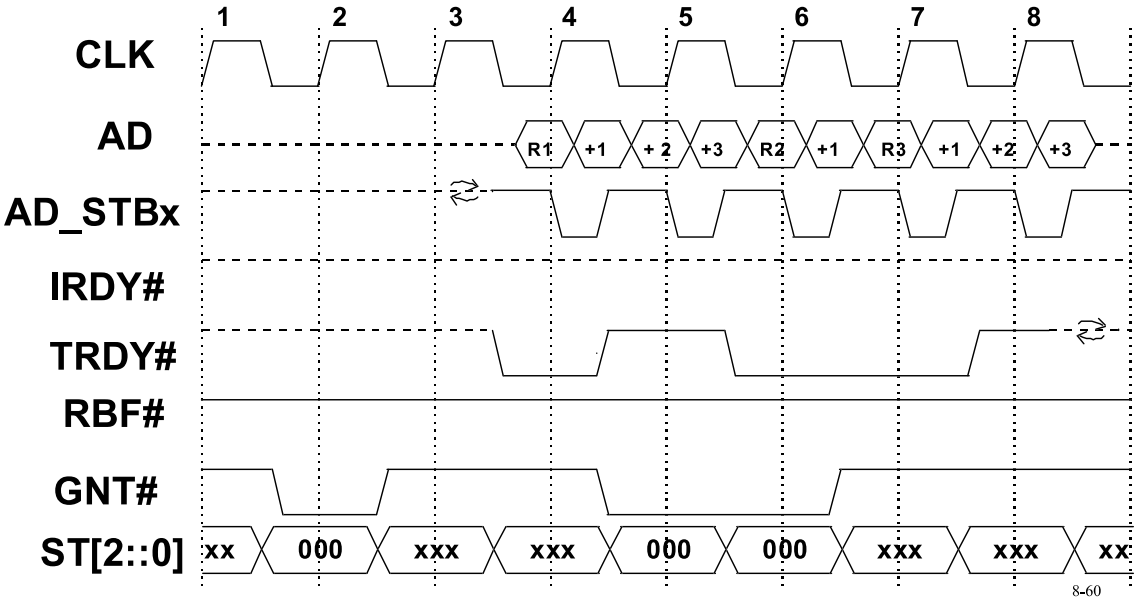


Figure 8-60 2x Reads Back to Back Different Lengths

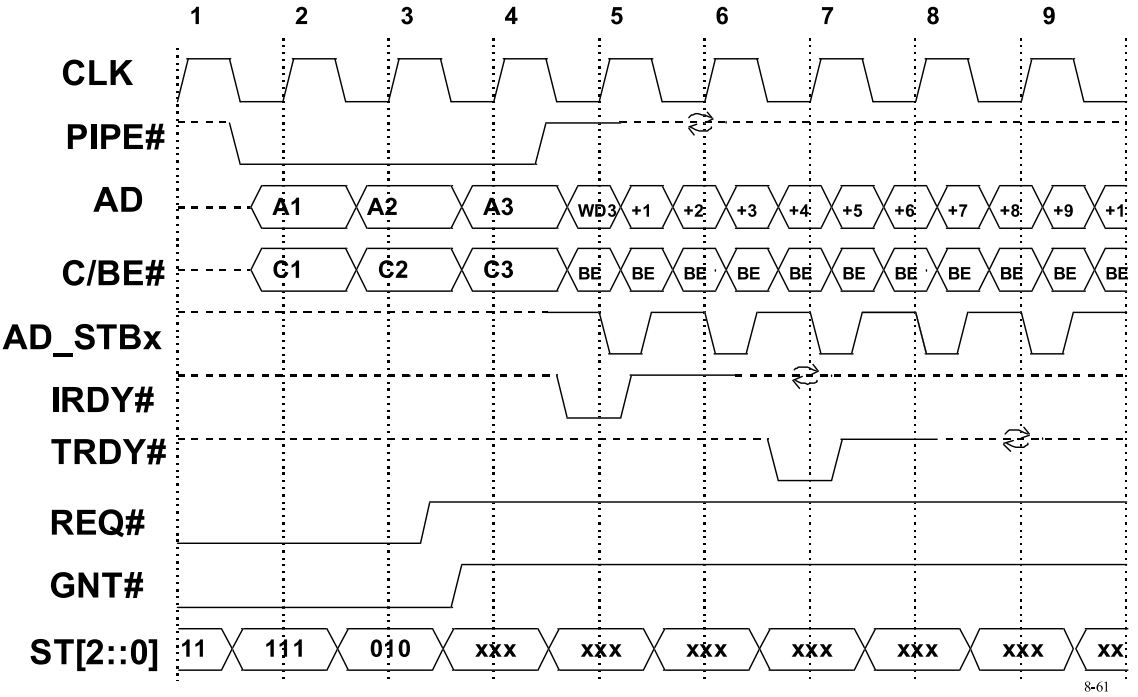


Figure 8-61 Multiple Requests followed by 2x Write No Delay Initial or Subsequent Blocks

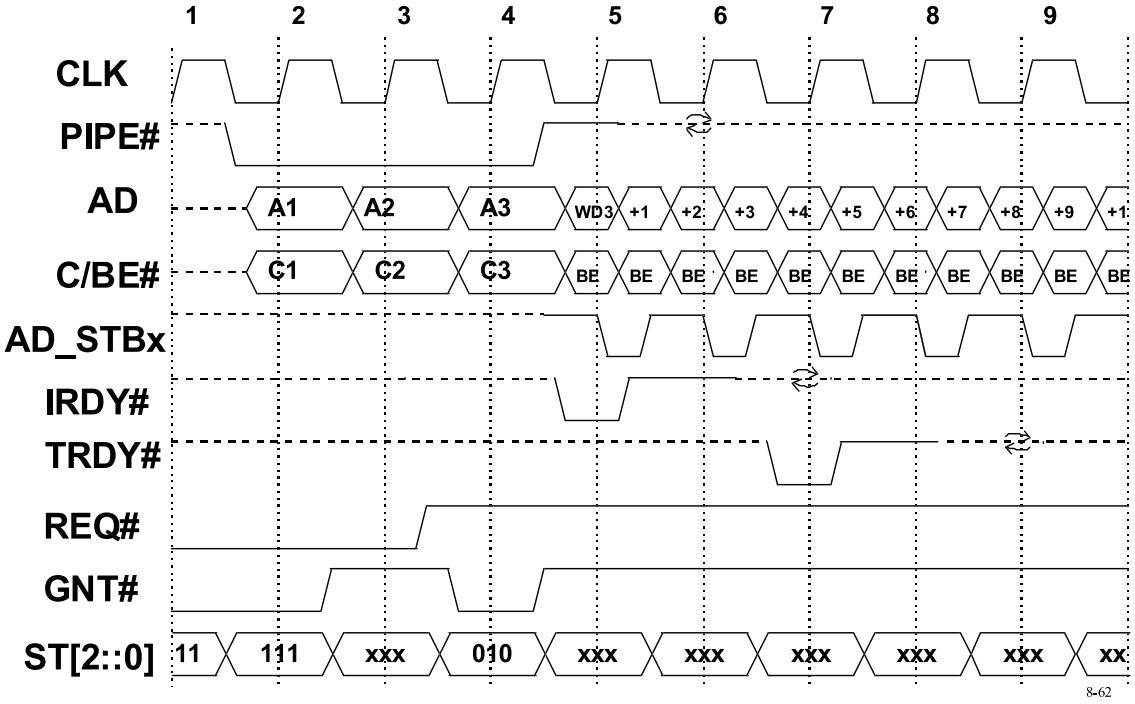


Figure 8-62 Requests followed by 2x Write Early GNT# no Delay

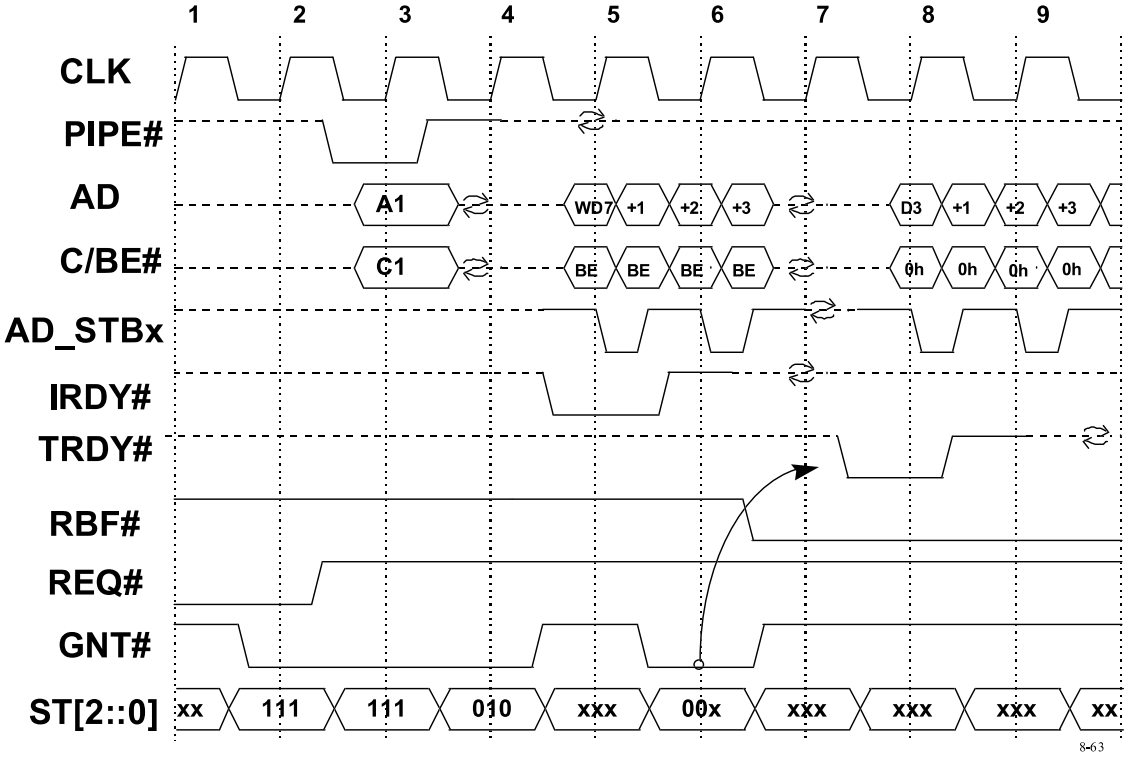


Figure 8-63 Single Request, 2x Write, 2x Read Early GNT#

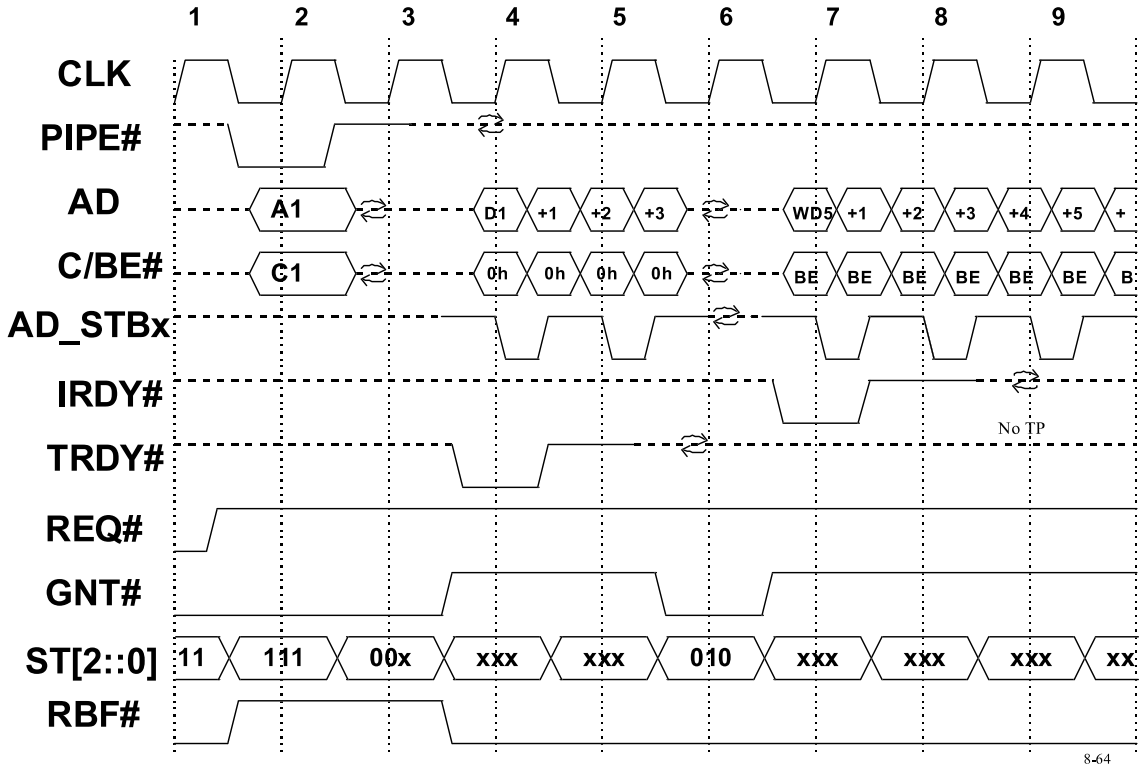


Figure 8-64 Request followed by 2x read, followed 2x Write Late GNT#

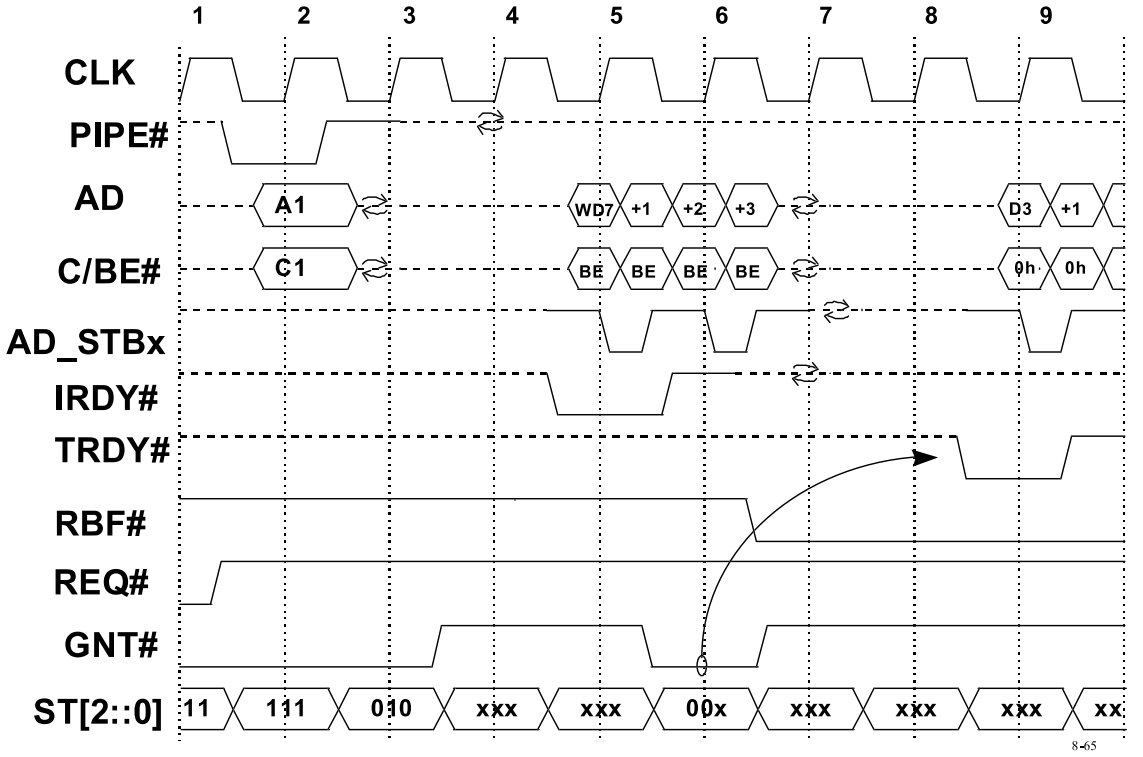


Figure 8-65 Single Request followed by 2x Write and 2x Read with Delays Early GNT#

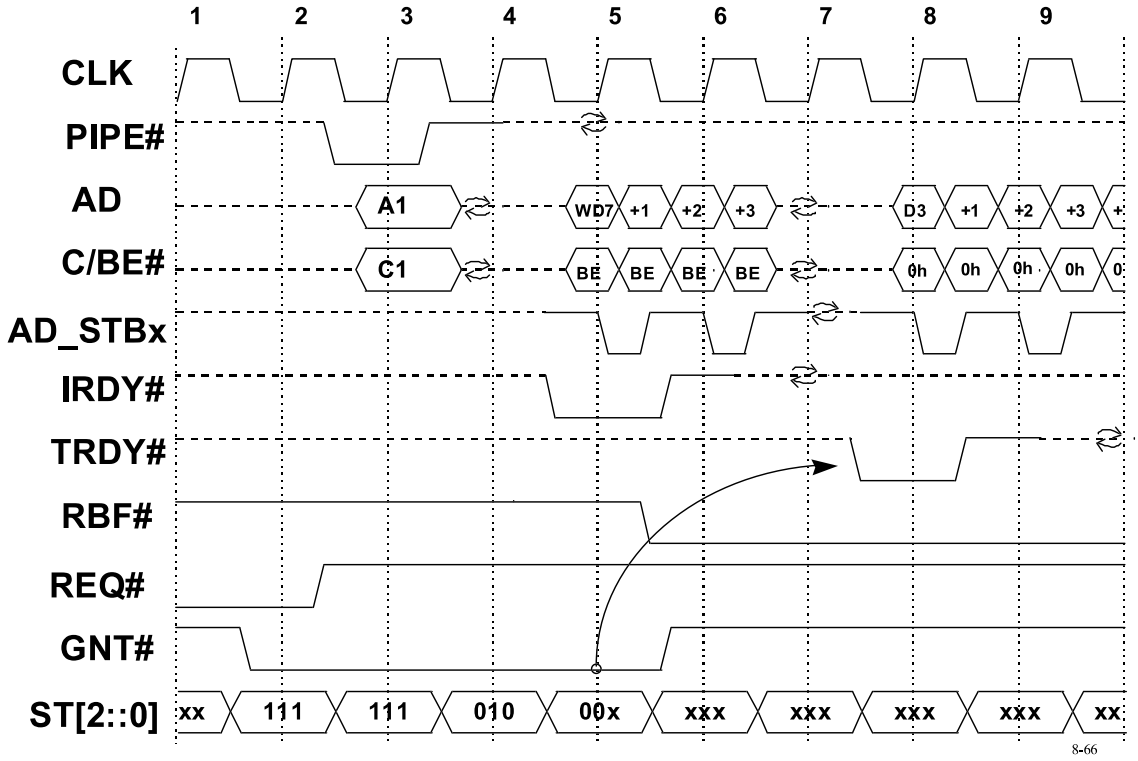


Figure 8-66 Single Request followed by 2x Write and 2x Read, Early GNT#

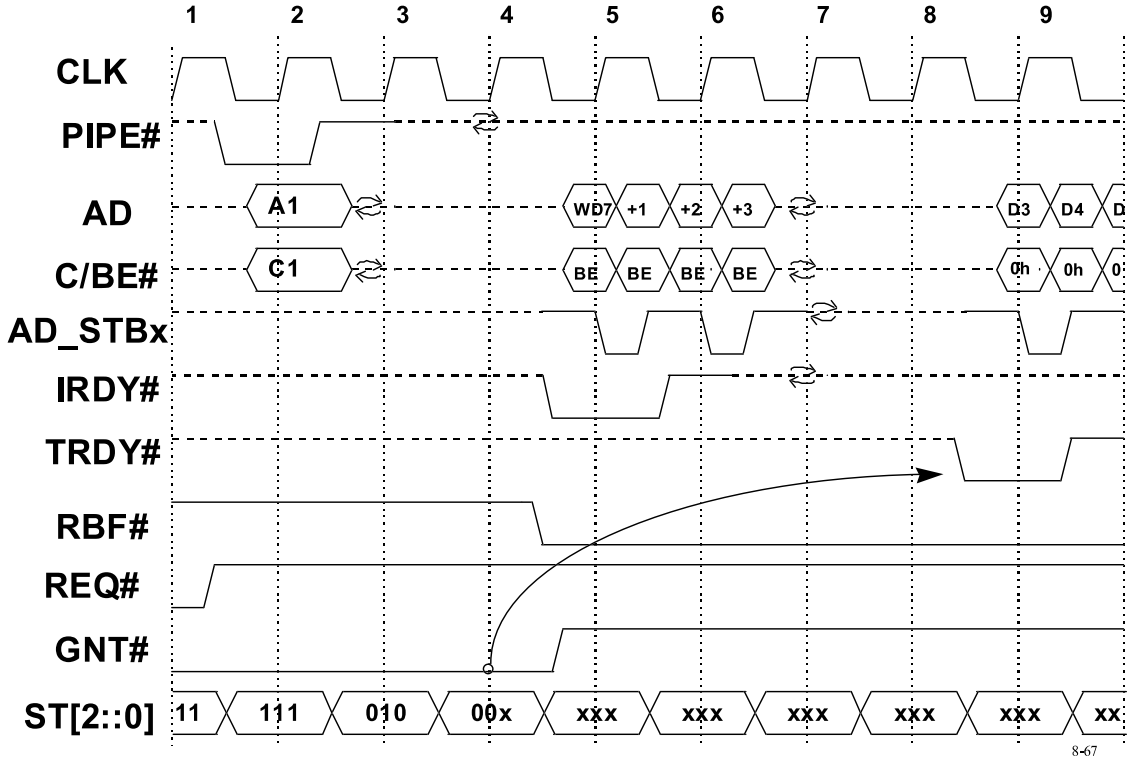


Figure 8-67 Single Request, Early GNT# for 2x Write (delay) followed by 2x Read (delay)

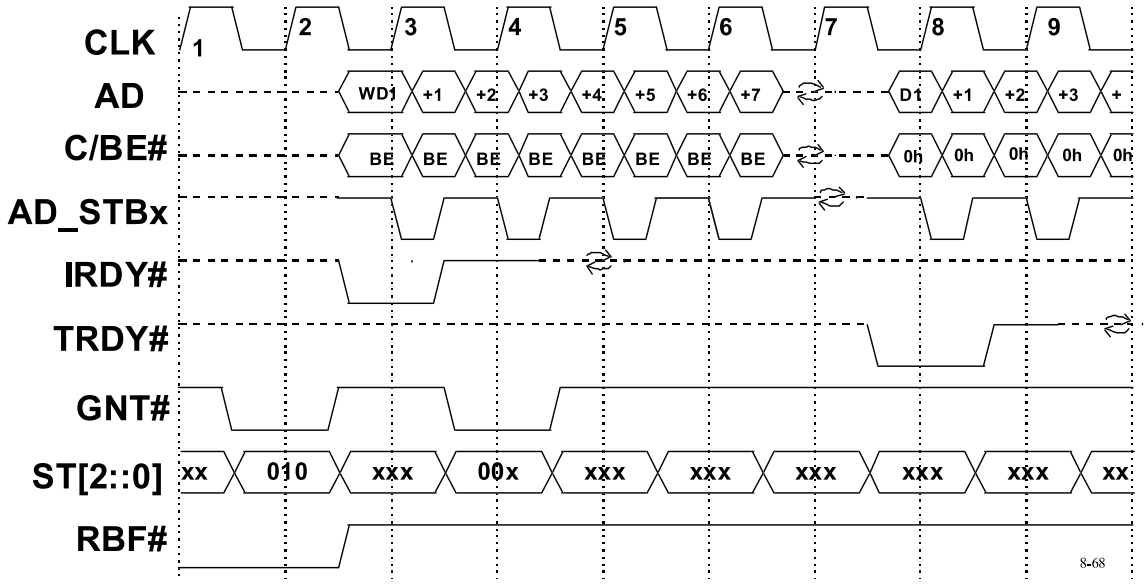


Figure 8-68 2x Write followed by 2x Read, GNT# Delayed 1 Clock

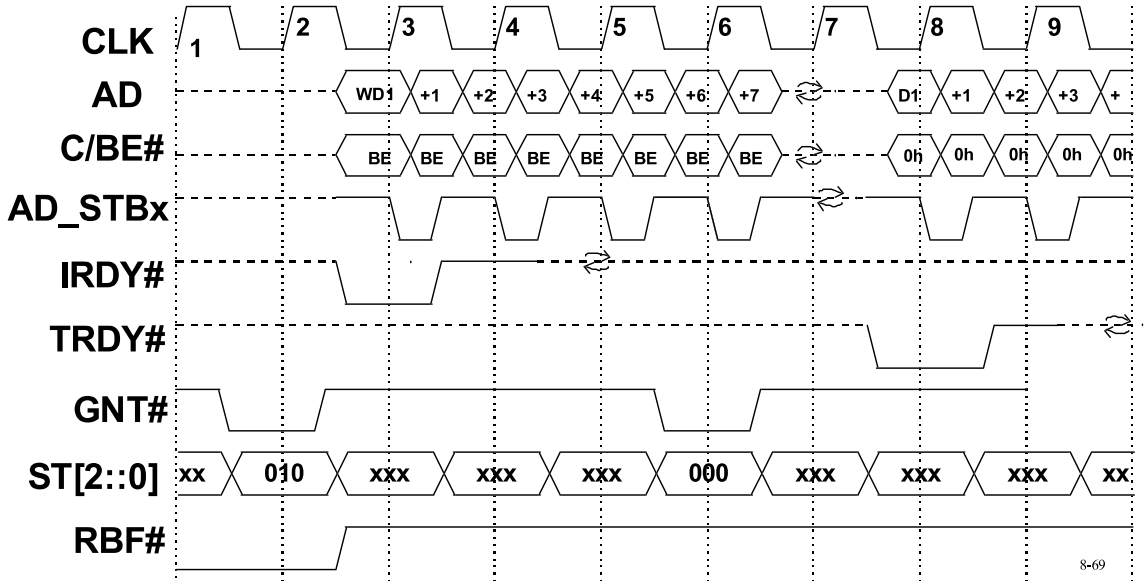


Figure 8-69 2x Write followed by 2x Read, GNT# No Delay

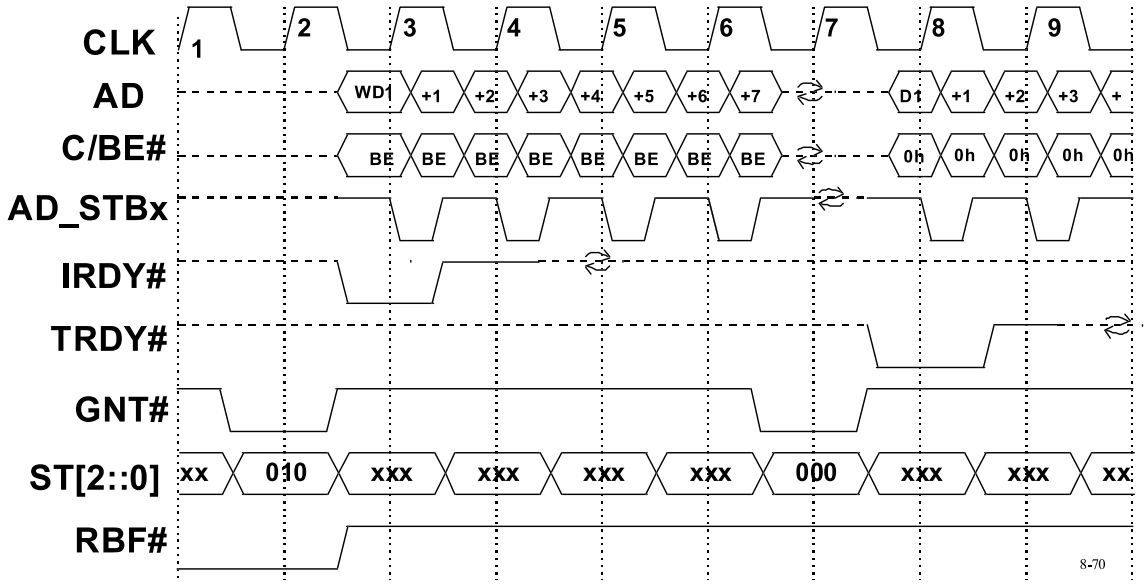


Figure 8-70 2x Write followed by 2x Read, GNT# Delayed Maximum with no Dead Clock

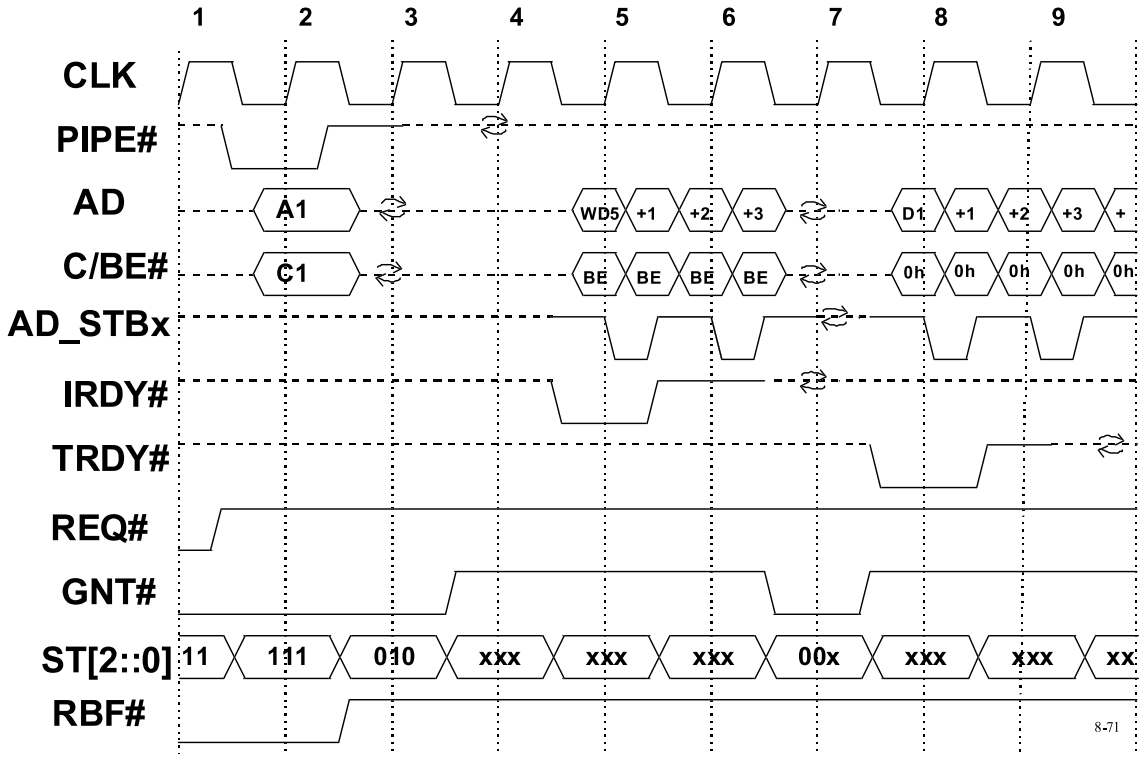


Figure 8-71 Single Request, Maximum 2x Write Delay, 2x Read Late GNT#

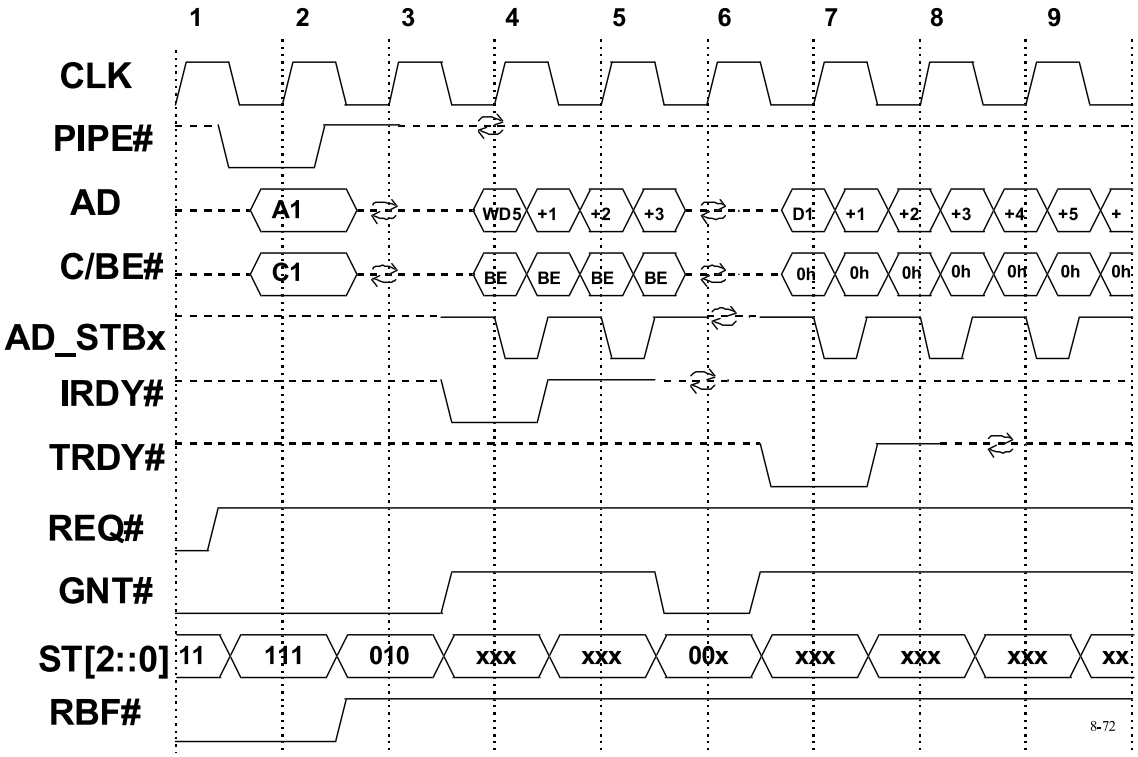


Figure 8-72 Single Request, 2x Write, 2x Read Late GNT#

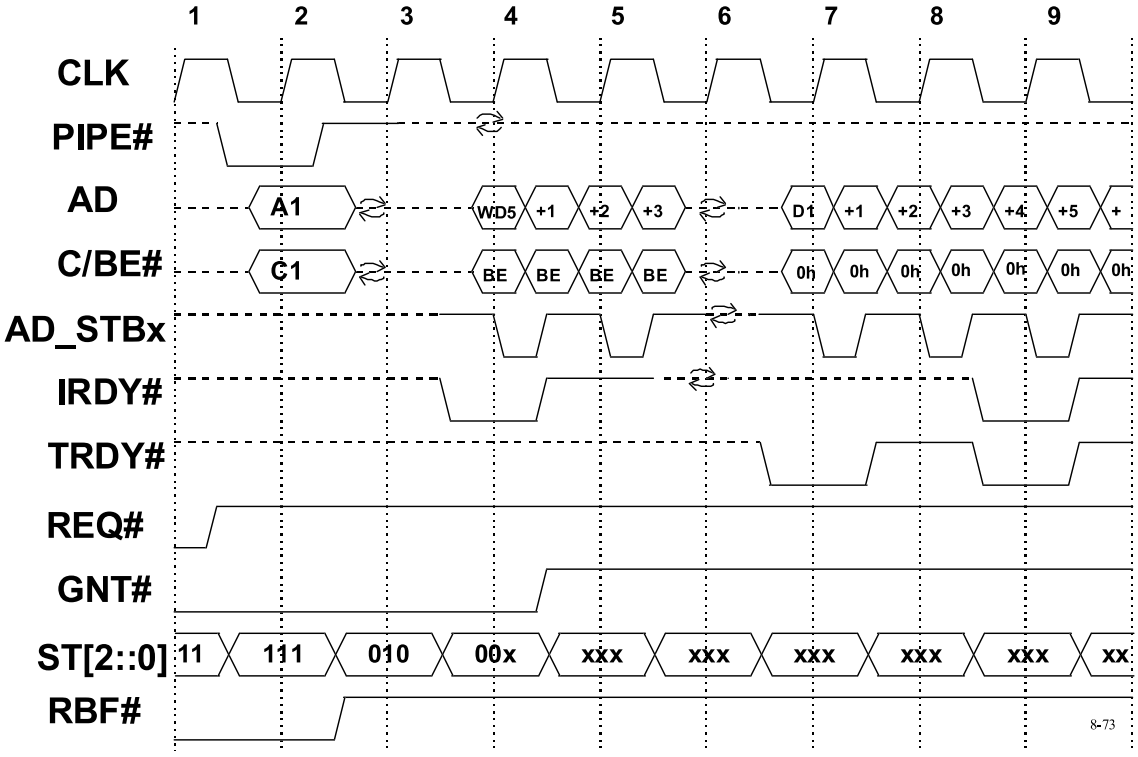


Figure 8-73 Single Request, 2x Write, 2x Read Earliest GNT#s

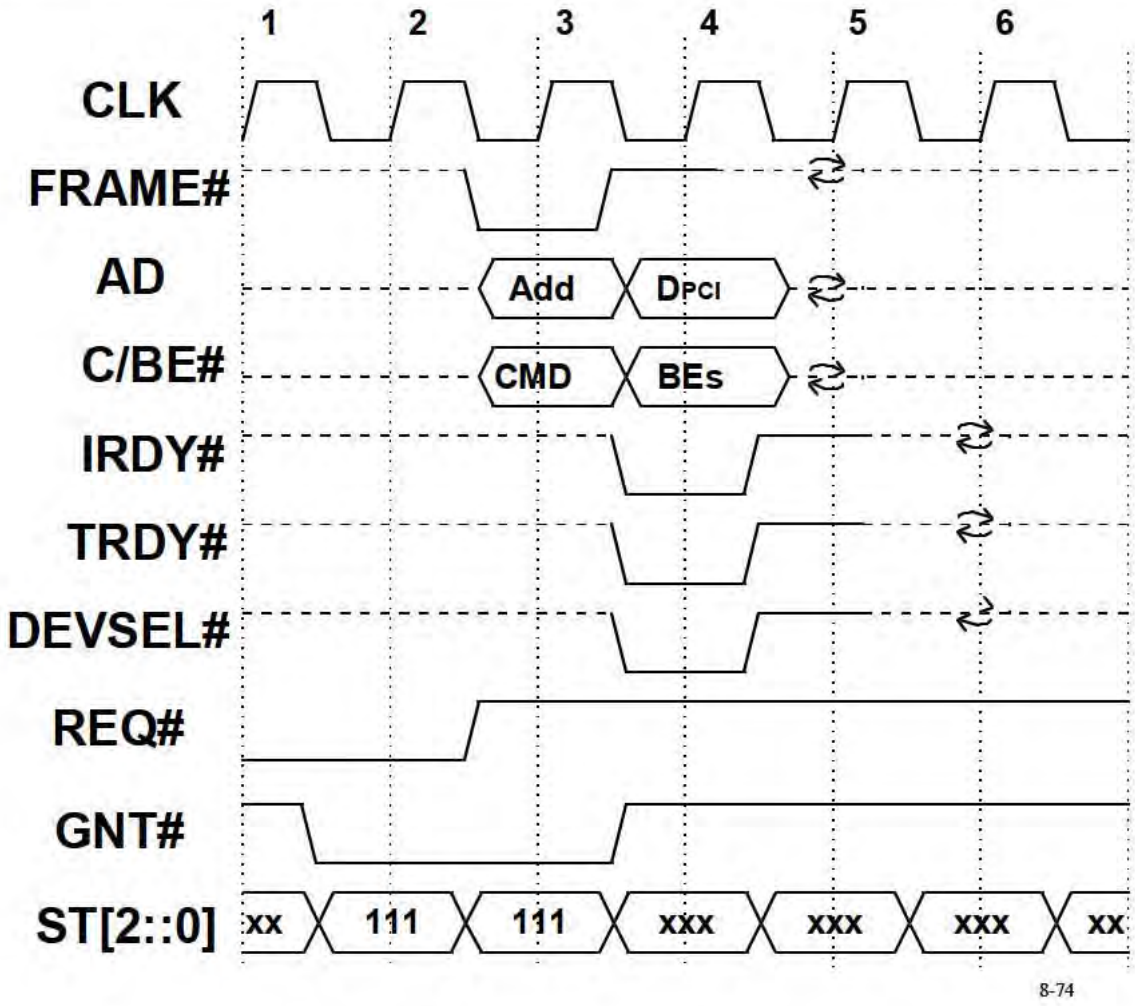


Figure 8-74 PCI Write Transaction on A.G.P.

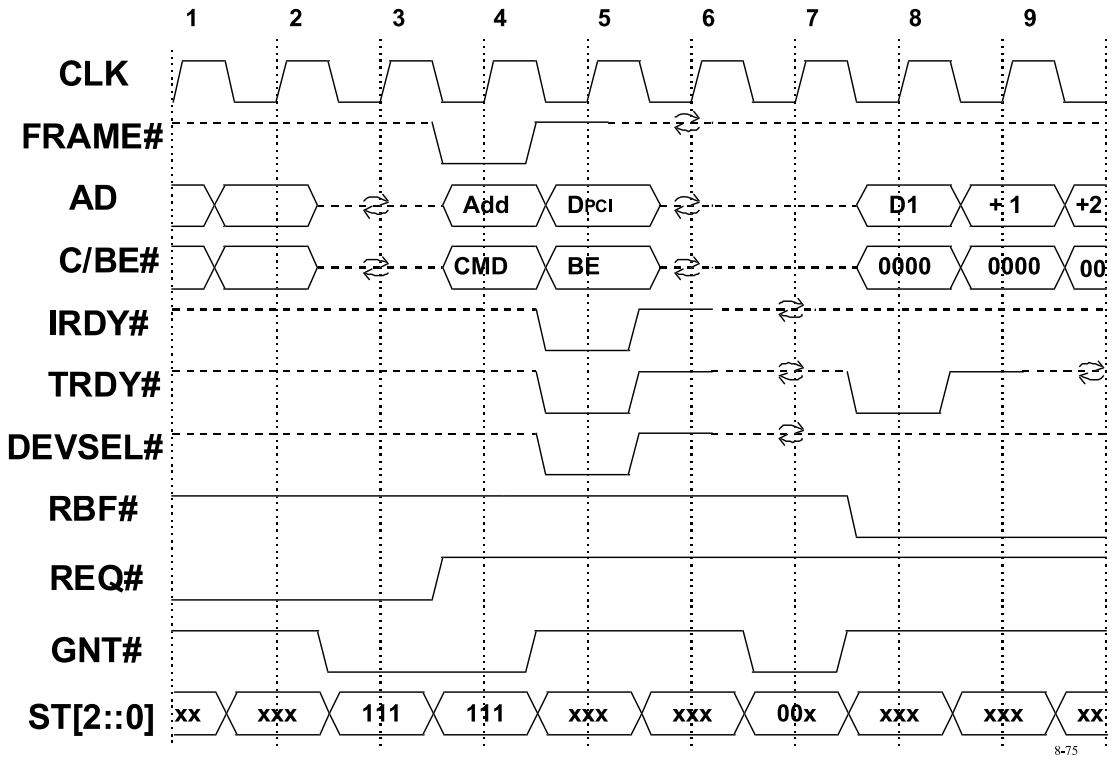


Figure 8-75 PCI Transaction Between 2 A.G.P. Data Transfers

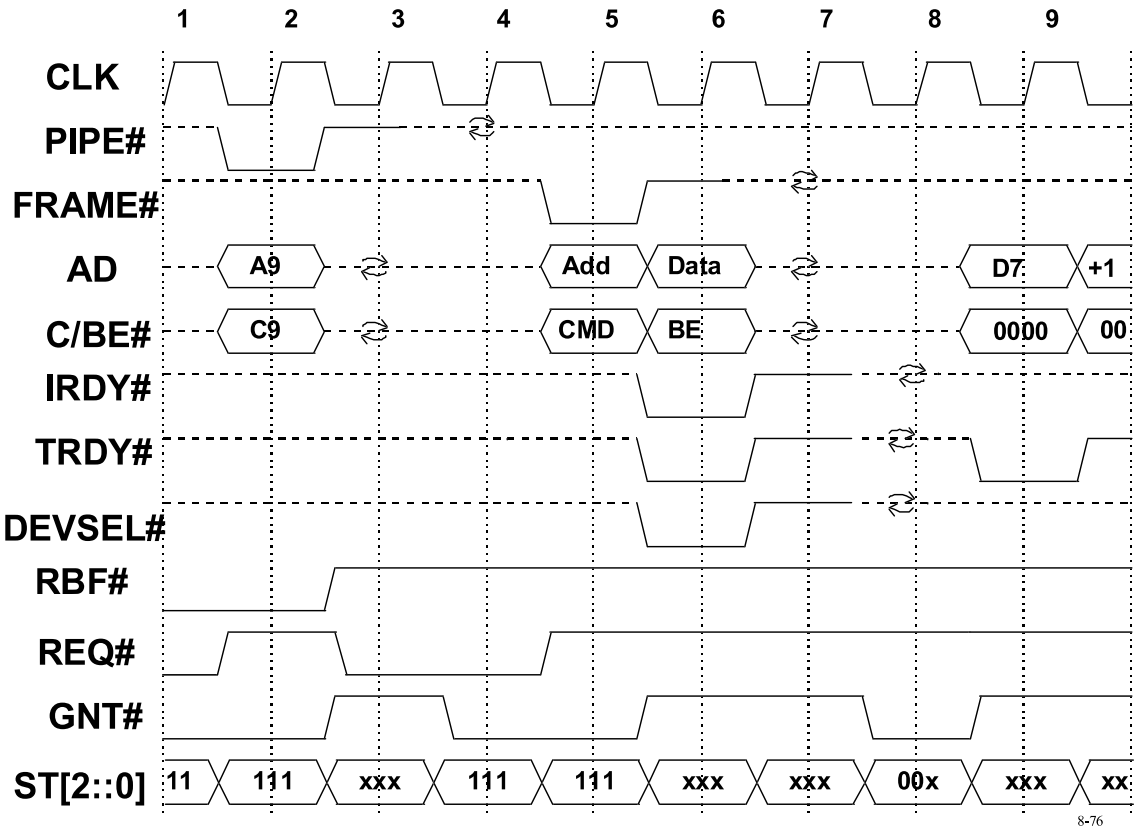


Figure 8-76 Single Request, PCI Transaction followed by Read Data (No Delays)

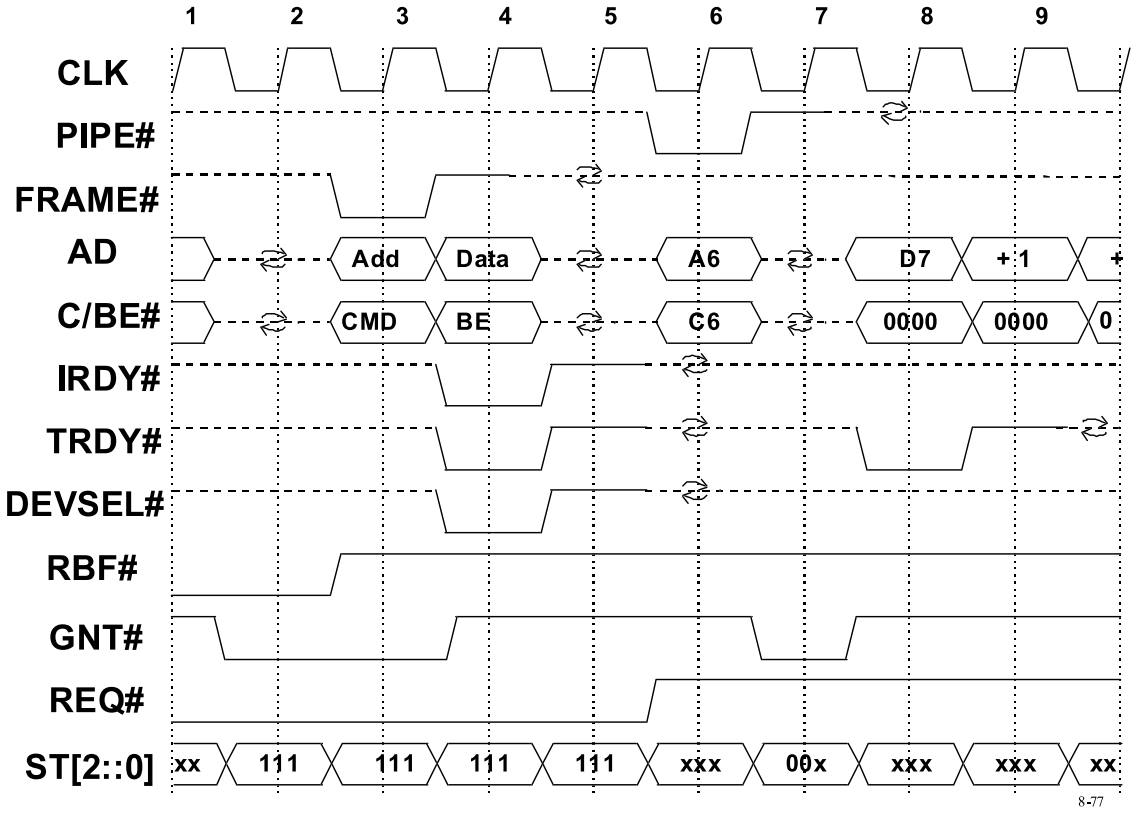


Figure 8-77 PCI Transaction followed by A.G.P. Request, Read Data (No Delays)

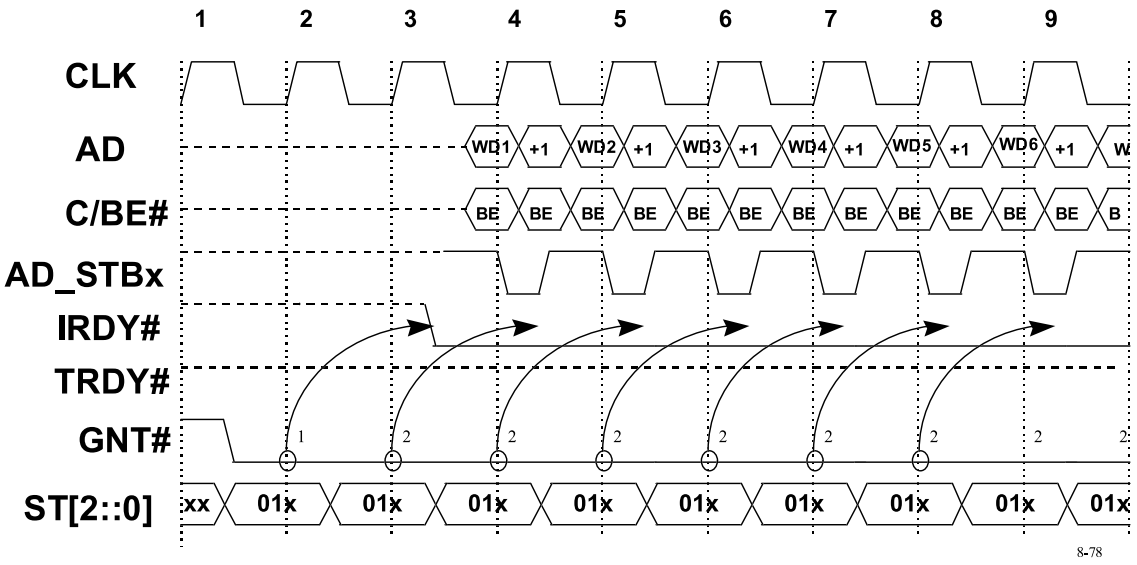


Figure 8-78 BtoB 8 Byte Writes, Initial Write with Max Delay, Subsequent Writes No Delays

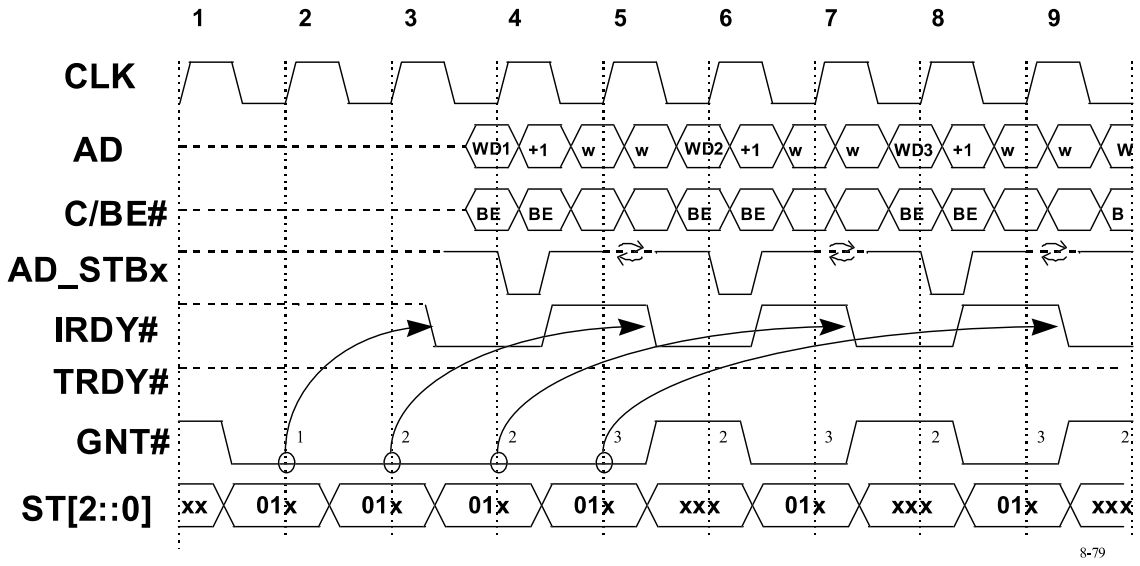


Figure 8-79 8 Byte Write, Initial Max Delay, Subsequent Max Delay (Part 1)

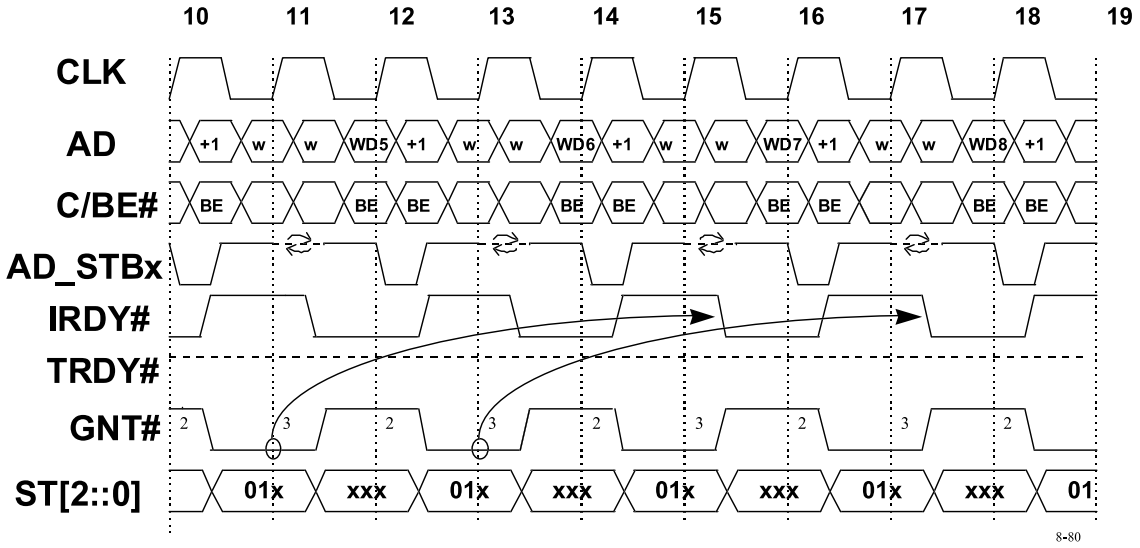


Figure 8-80 8 Byte Write, Initial Max Delay, Subsequent Max Delay (Part 2)

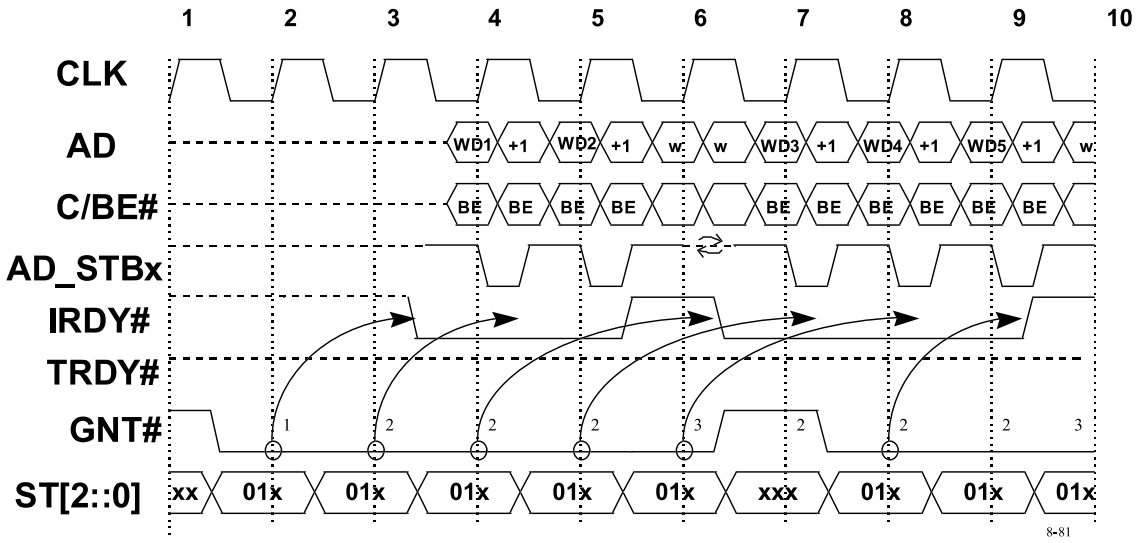


Figure 8-81 BtoB 8 Byte Writes, Initial Write with Delay, Subsequent Writes Delays (Part 1)

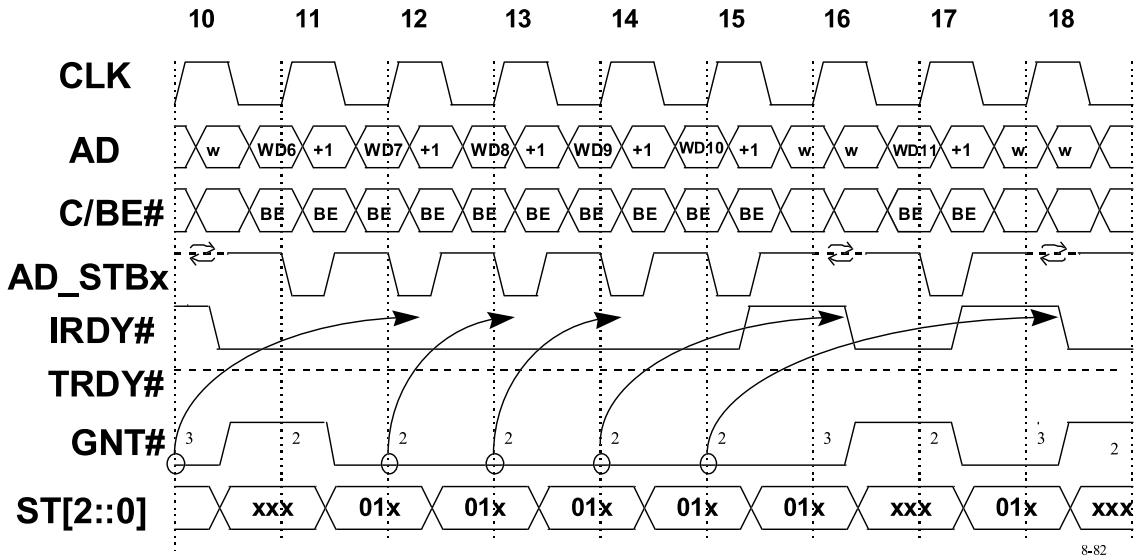


Figure 8-82 BtoB 8 Byte Writes, Initial Write with Delay, Subsequent Writes Delays(Part 2)

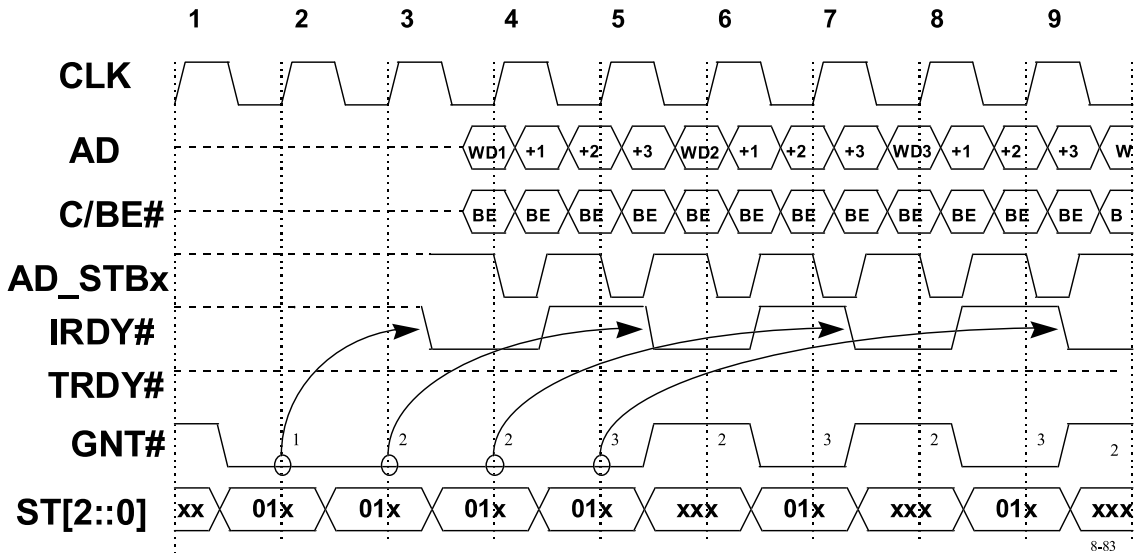


Figure 8-83 16 Byte Writes, Initial Max Delay, No Subsequent Delay

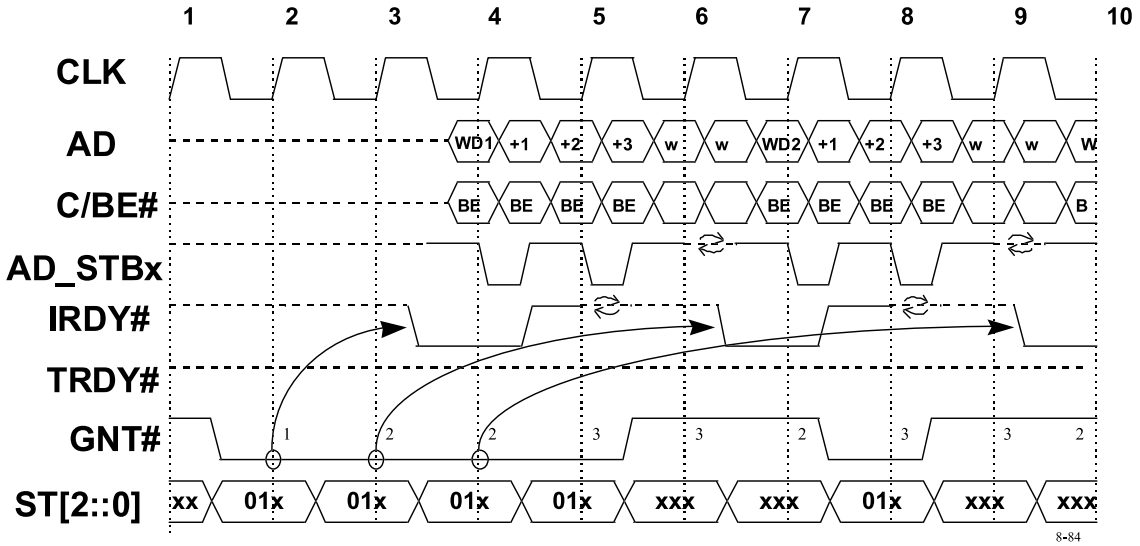


Figure 8-84 16 Byte Writes, Initial Max Delay, Subsequent Max Delays

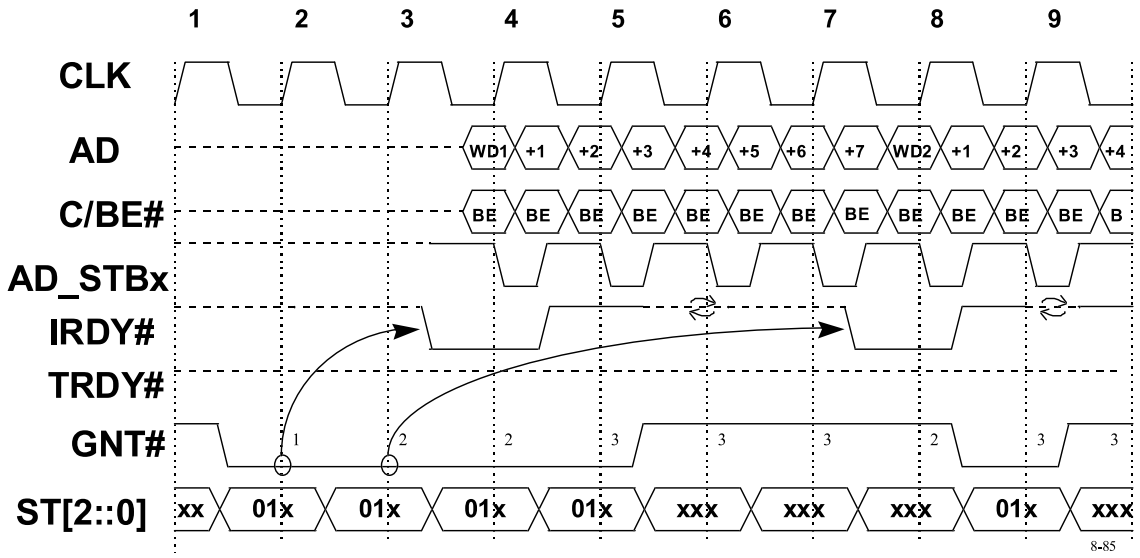


Figure 8-85 32 Byte Writes, Initial Max Delay, No Subsequent Delay

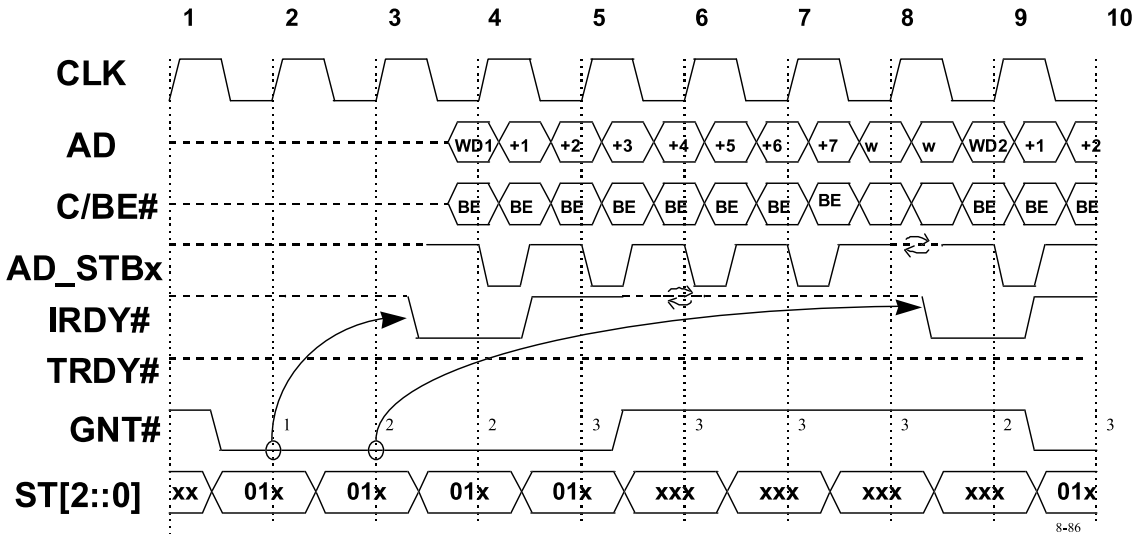


Figure 8-86 32 Byte Writes, Initial Max Delay, Subsequent Max Delays

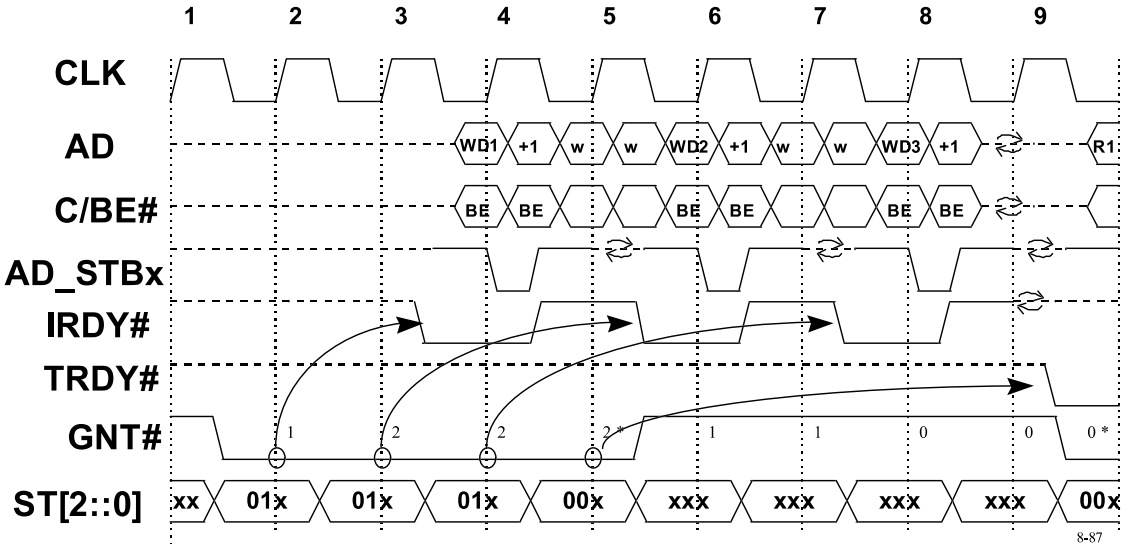


Figure 8-87 Three 8 Byte Writes, 8 Byte Read, 40 Byte Read, Three 8 Byte Writes (Part 1)

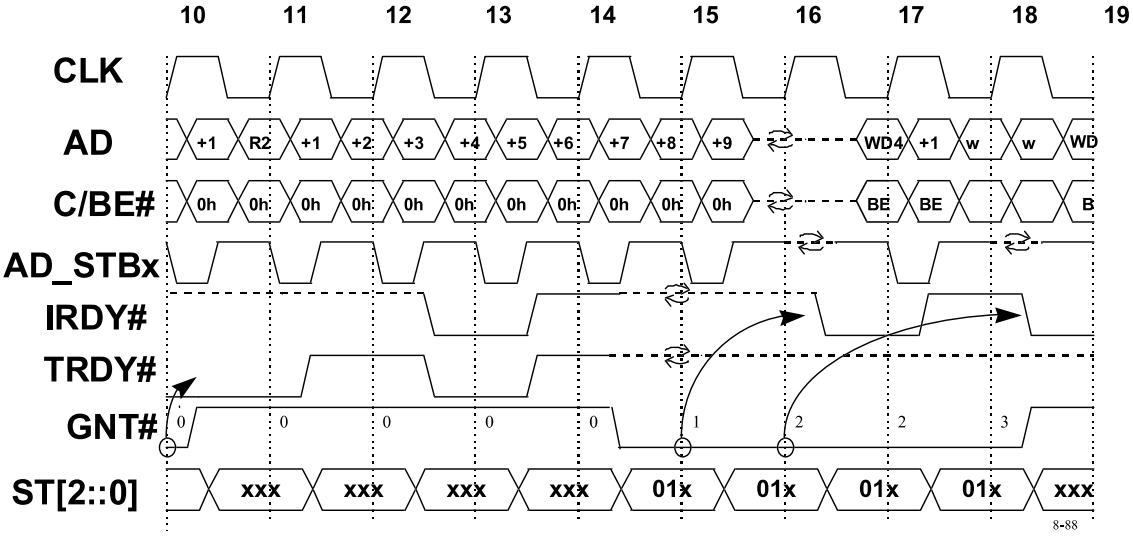


Figure 8-88 Three 8 Byte Writes, 8 Byte Read, 40 Byte Read, Three 8 Byte Writes (Part 2)

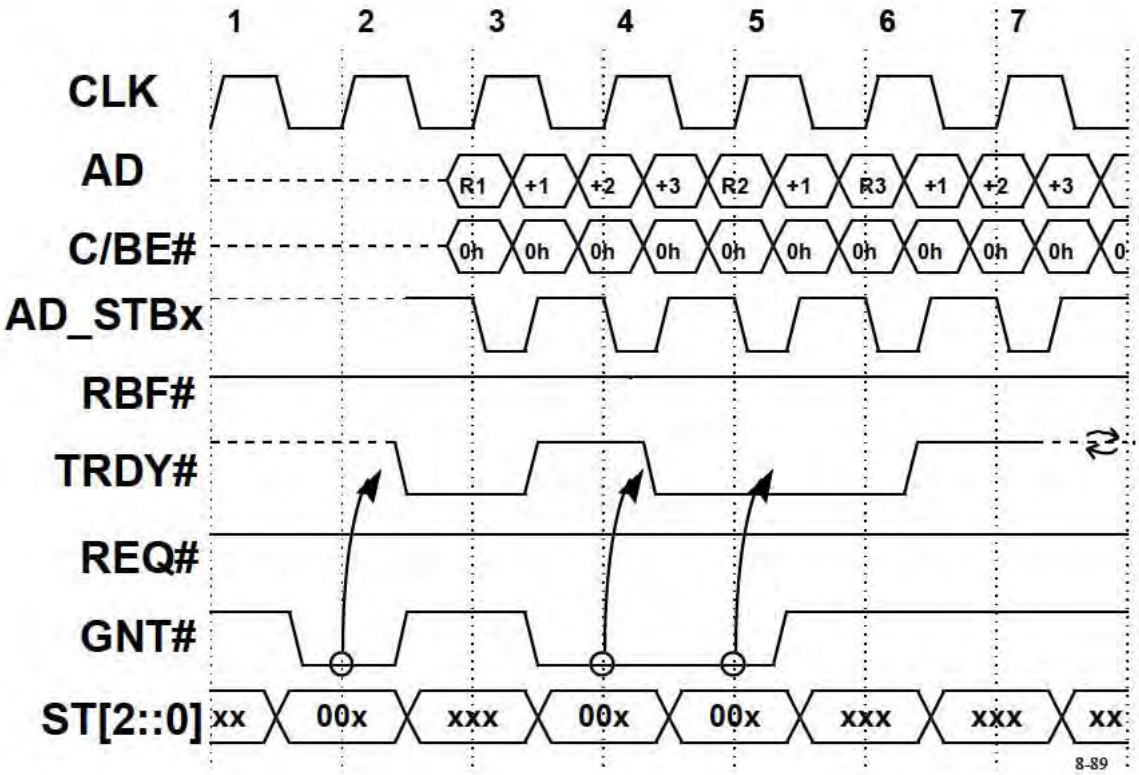


Figure 8-89 16 Byte Read, 8 Byte Read, 16 Byte Read

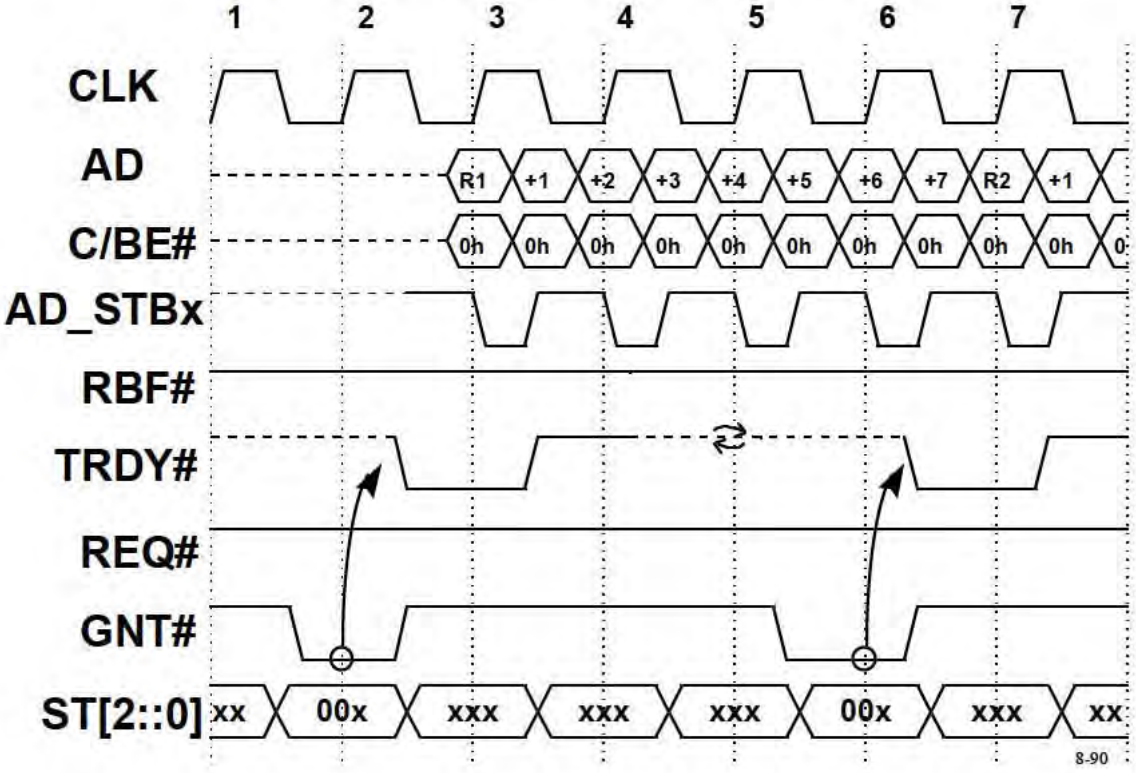


Figure 8-90 Back to Back 32 Byte Reads

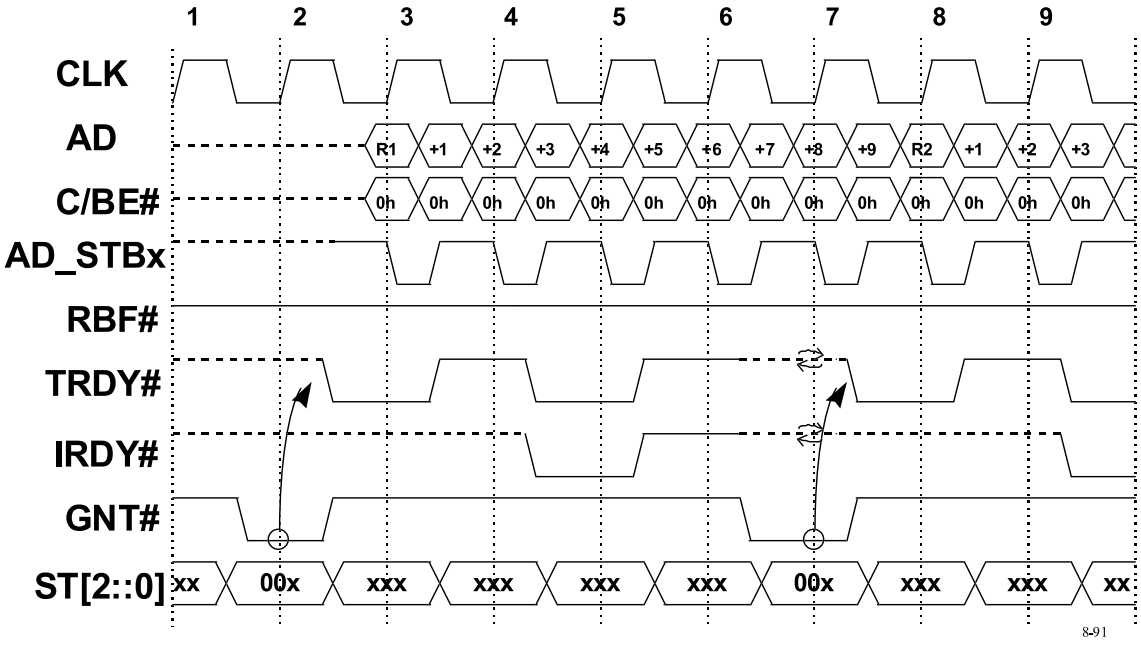


Figure 8-91 Back to Back 40 Byte Reads