Rev. 0.1, 6/2003

Outstanding Data Tenures on the MPX Bus

The MPX bus of many PowerPC™ processors (for example, the MPC7400, MPC7410 and MPC7450/MPC7451) is a split transaction bus that provides for independent arbitration and operation of the address and data buses. (The MPC7451 has the same functionality as the MPC7450, and any differences in data regarding bus timing, signal behavior, and AC, DC, and thermal characteristics are detailed in the hardware specifications. Note that because the MPC7450 and MPC7451 have the same functionality, they are used interchangeably throughout the documentation and code.)

The MPX bus is a high-performance bus that is optimized for throughput. It supports address-only and data-only transactions as well as full address and data pipelining. Because the address and data buses are independent, there can be multiple outstanding data transactions at any one time and data transactions can be independently re-ordered. The number of outstanding transactions is dependent on the capability of the processor and the system configuration. The MPC7400 and MPC7410 can support a maximum of six outstanding data transactions and the MPC7450 can support a maximum of sixteen outstanding transactions.

This application note describes a set of external signals, called Outstanding Data Tenures (ODTx), that can be added to any MPX bus system arbiter for tracking outstanding transactions and correlating data tenures with address tenures. The addition of these signals is particularly useful to an inverse assembler (IA) for correlating address tenures with their corresponding data tenures in a bus trace.

The ODTx signals are an attribute driven by the arbiter and whose only target is a logic analyzer. The width of ODTx can be four bits for systems capable of eight outstanding data tenures and five bits for systems that support sixteen outstanding data tenures.

1.1 Separate Address and Data Buses and DTIx

Systems that implement the MPX bus protocol drive the data transaction index (DTIx) signals as a data attribute input to the processor to indicate the correlation of data tenures with address tenures during pipelined transactions. DTIx is driven one cycle before $\overline{\text{DBG}}$ and it serves as a reference based on the age of outstanding transactions. For example, a DTIx of 0000 implies that the next data tenure corresponds to the oldest outstanding transaction, a DTIx of 0001 implies that the next data tenure is for the second-oldest outstanding transaction, and so on.

At the beginning of a trace, an IA can not determine how many outstanding transactions there are. Therefore, an IA cannot correlate the next data tenure with the first address tenure

captured on the trace or an address tenure that occurred before the trace. This issue becomes more complex when re-ordering of data transactions is directed by the system (via DTIx).

1.2 Correlating Outstanding Transactions

1.2.1 ODTx Definition (Same Cycle as $\overline{BG}$)

The Outstanding Data Tenures (ODTx) attribute can be driven by the arbiter on the same cycle that bus grant ($\overline{BG}$) is asserted by the arbiter to a processor. The ODTx value should be equal to the number of data tenures outstanding for the processor receiving the bus grant. The ODTx value should be zero for zero outstanding data tenures, one for one outstanding data tenure, and so on up to the maximum number of outstanding data tenures supported by the system. For a system that supports a maximum of eight outstanding data tenures, 4 bits of ODTx are required to indicate values of 0–8.

The number of outstanding data tenures at any time is unique for each processor. The following two scenarios increase the number of data tenures outstanding for a processor:

- $\overline{TS}$ assertion by a processor with a TT[0:4] attribute indicating a non address-only operation with no $\overline{ARTRY}$ driven the same cycle
- $\overline{HIT}$ assertion by a processor with no $\overline{ARTRY}$ driven the same cycle

The following two scenarios decrease the number of data tenures outstanding for a processor:

- $\overline{ARTRY}$ assertion corresponding to a prior valid assertion of $\overline{TS}$
- $\overline{DBG}$ assertion corresponding to a prior valid assertion of $\overline{TS}$

By driving ODTx coincident with $\overline{BG}$, the arbiter can drive unique values to each processor. Figure 1 shows a simple example of ODTx.

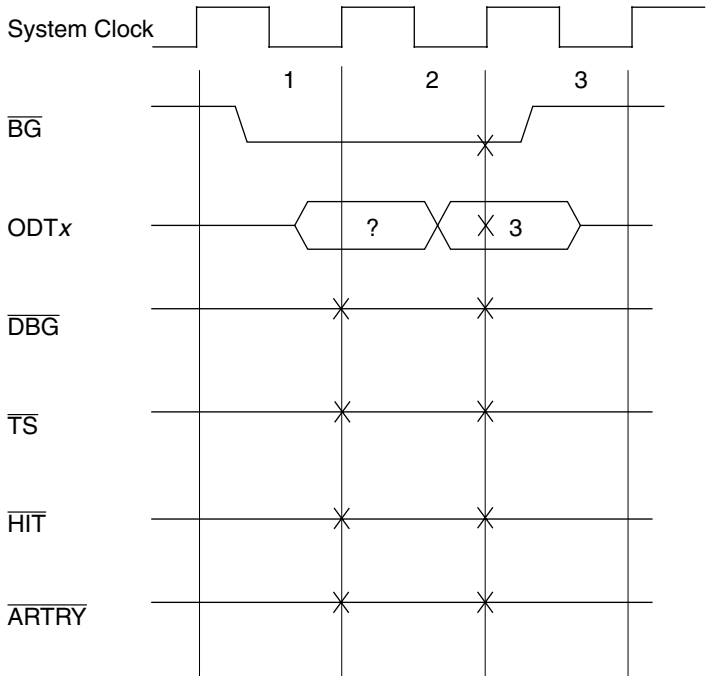

Figure 1. Simple ODTx Trace

Assume cycle 1 in Figure 1 is the first cycle of a bus trace. Correct sampling of ODTx requires that the first trace cycle be a don't care. Therefore, the ODTx value sampled at the end of cycle 1 is unknown and must be ignored. In cycle 2, $\overline{BG}$ is asserted and a value of three is driven for ODTx. This indicates that the processor receiving the bus grant has three outstanding data tenures awaiting a $\overline{DBG}$.

1.2.2 Example of ODTx and Transaction Re-ordering

Knowing the number of outstanding data tenures, indicated by ODTx, for a given processor assists an IA in correlating address and data tenures later on in the trace. Table shows a high level view of a bus trace to show how ODTx can be used. Each state corresponds to an event that either increments or decrements the number of outstanding data tenures for a processor. States 0 through 3 correspond to address tenures completed before the bus trace data begins in state 4, ODTx is determined to be equal to three. Using this information, the IA can begin to construct a table of address and outstanding data tenure pairs for the processor as shown in Table . Table shows the actual outstanding tenures for reference.

Table 1. State Table of ODT Relationship with Address and Data Transactions

$\overline{TS0}$	$\overline{TS1}$	$\overline{TS2}$ (slow I/O)	$\overline{TS3}$				$\overline{TS4}$	$\overline{TS5}$				$\overline{TS6}$	$\overline{TS7}$
					$\overline{DBG}$	$\overline{DBG}$			$\overline{DBG}$	$\overline{DBG}$	$\overline{DBG}$		$\overline{DBG}$
					DTI = 0 ($\overline{TS0}$)	DTI=0 ($\overline{TS1}$)			DTI = 1 ($\overline{TS3}$)	DTI = 2 ($\overline{TS5}$)	DTI = 0 ($\overline{TS2}$)		DTI = 0 ($\overline{TS4}$)
				$\overline{BG}$ ODT=3									
State 0	**State 1**	**State 2**	**State 3**	**State 4**	**State5**	**State 6**	**State 7**	**State 8**	**State 9**	**State 10**	**State 11**	**State 12**	**State 13**

Table 2. Queue of Outstanding Data Tenures in a Processor as seen by the IA

Unknown3				Addr5					
Unknown2	Unknown3		Addr4	Addr4	Addr5				
Unknown1	Unknown2	Unknown3	Unknown3	Unknown3	Addr4	Addr4		Addr6	Addr7
Unknown0	Unknown1	Unknown2	Unknown2	Unknown2	Unknown2	Unknown2	Addr4	Addr4	Addr6
State 4	**State 5**	**State 6**	**State 7**	**State 8**	**State 9**	**State 10**	**State 11**	**State 12**	**State 13**

Table 3. Queue of Outstanding Data Tenures in a Processor as seen by the Processor

			Addr3	Addr3				Addr5					
		Addr2	Addr2	Addr2	Addr3		Addr4	Addr4	Addr5				
	Addr1	Addr1	Addr1	Addr1	Addr2	Addr3	Addr3	Addr3	Addr4	Addr4		Addr6	Addr7
Addr0	Addr0	Addr0	Addr0	Addr0	Addr1	Addr2	Addr2	Addr2	Addr2	Addr2	Addr4	Addr4	Addr6
State 0	**State 1**	**State 2**	**State 3**	**State 4**	**State 5**	**State 6**	**State 7**	**State 8**	**State 9**	**State 10**	**State 11**	**State 12**	**State 13**

The example shown in the tables shows non-zero values of DTIx in states 9 and 10. For these states, the system is causing the data transactions to be re-ordered, such that the corresponding assertions of $\overline{DBG}$ are implied for transactions other than the oldest outstanding transaction. In this example, the system is requesting that data transactions service the address transactions in the order of Addr0, Addr1, Addr3, Addr5, Addr2, followed by Addr4. The interpretation of ODTx by the IA takes this re-ordering into account

and can still accurately correlate the address transactions with the correct data transactions. Note that by state 11, the IA has complete knowledge of the outstanding data tenures for the processor.

1.2.3 ODTx Adjustments

Three classes of ODTx values exist:

- Correct
- Incorrect and correctable
- Incorrect and not correctable

An ODTx value is guaranteed to be correct if none of the following qualifier signals are asserted on the bus the cycle before or the cycle of ODTx: $\overline{TS}$, $\overline{HIT}$, $\overline{ARTRY}$, or $\overline{DBG}$. Figure 1 is an example of this case. For Figure 1, the value of three for ODTx is guaranteed to be correct.

If any of the four qualifier signals is asserted the cycle before or the cycle of ODTx, then the arbiter does not and cannot factor them into the ODTx value. Therefore, in order to calculate the correct value, the IA must adjust the value of ODTx.

For the IA to adjust the ODTx value driven by the arbiter, the IA needs to know the source of each of the asserted signals. $\overline{DBG}$ and $\overline{HIT}$ are point-to-point signals, so the arbiter knows the source. Figure 2 shows $\overline{DBG}$ asserted for a processor in cycle 1. Although ODTx is driven as a three, the IA must interpret the value as a two because the arbiter did not include the occurrence of $\overline{DBG}$. If only $\overline{DBG}$ and/or $\overline{HIT}$ are driven in the window around ODTx, the driven ODTx is always correctable by the IA.

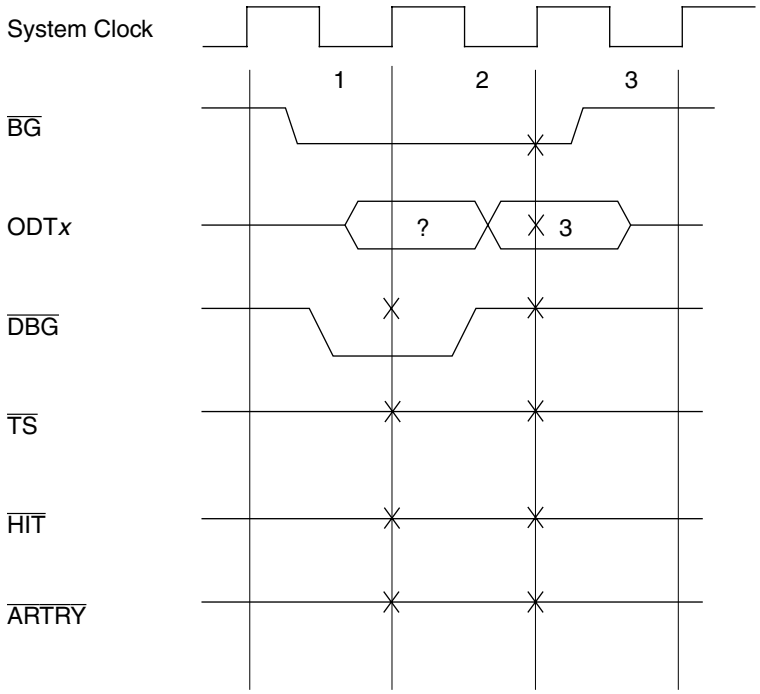

Figure 2. Simple ODTx trace with $\overline{DBG}$ Asserted the Cycle before ODTx

If $\overline{TS}$ and/or $\overline{ARTRY}$ are driven the cycle before ODTx, or $\overline{ARTRY}$ is driven during the cycle of ODTx, the ODTx value may not be correctable by the IA. Figure 3 shows $\overline{ARTRY}$ asserted in cycle 5. If between the beginning of the bus trace, cycle 1, and cycle 5 it has not been established which original $\overline{TS}$ corresponds to the $\overline{ARTRY}$, there is no way to correct the ODTx value. The IA must then proceed to the next cycle. In cycle

6, the ODTx is also not correctable due to the $\overline{\text{ARTRY}}$ assertion the previous cycle. In cycle 7, the arbiter has had time to process the $\overline{\text{ARTRY}}$, and ODTx is correct. Unlike $\overline{\text{ARTRY}}$, if $\overline{\text{TS}}$ is driven in the cycle of ODTx, a corresponding $\overline{\text{BG}}$ must have been driven in the previous cycle. Therefore, the owner of the $\overline{\text{TS}}$ is known.

To summarize, a qualified ODTx, where qualified is defined as correct or correctable and where the associated processor for the assertion of $\overline{\text{TS}}$ or $\overline{\text{ARTRY}}$ is not known, is as follows:

qualified ODTx = BG & ~ARTRY & ~latched TS & ~latched ARTRY

After a correct ODTx value has been determined for a processor, the IA can track the outstanding data tenures itself without the aid of ODTx.

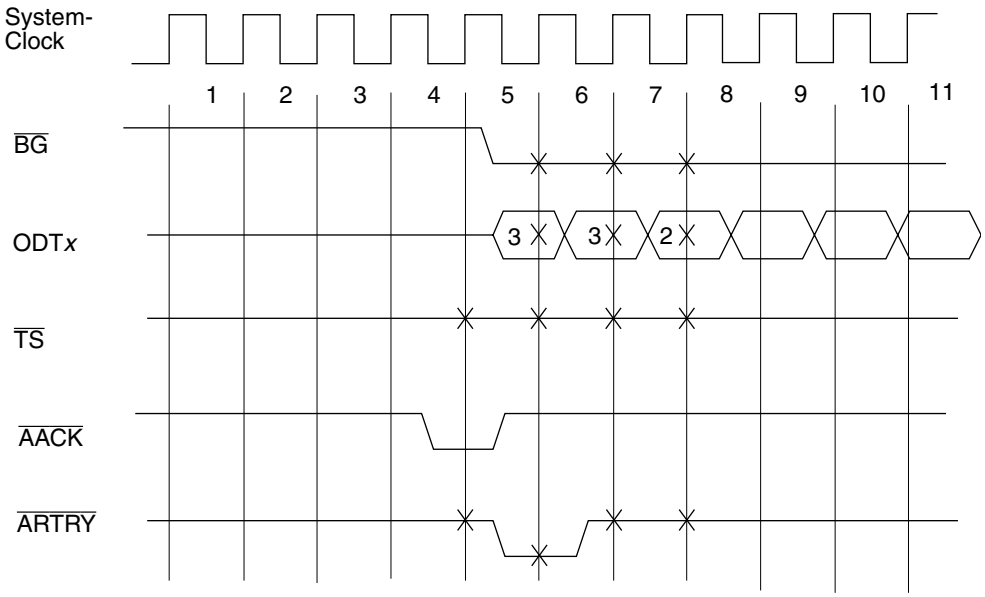

Figure 3. Trace with $\overline{\text{ARTRY}}$ Asserted the Cycle before ODTx—Not Correctable

1.2.4 Alternate Timing for ODTx

The ODTx attribute described in Section 1.2.1, “ODTx Definition (Same Cycle as $\overline{\text{BG}}$),” can also be designed with the same qualifiers described in that section, but driven one cycle later. This facilitates the use of ODTx for devices external to the system arbiter. Using this alternate timing, ODTx is driven starting the cycle after the corresponding $\overline{\text{BG}}$ is first asserted. The external device should stop driving ODTx based on the first rising clock edge in which the corresponding $\overline{\text{BG}}$ is detected as negated. Effectively, the timing is the same as that shown in Figure 1 except that ODTx is shifted to the right by one clock cycle.

In this case, there are cycles in which $\overline{\text{BG}}$ is driven with no valid ODTx and cycles in which a valid value of ODTx is driven with no driven $\overline{\text{BG}}$. In order to be able to interpret the ODTx values correctly when this alternate timing is used, the IA can set up a user preference directing its hardware to sample ODTx with this delayed timing.

Note that the ODTx signals defined in Section 1.2.1, “ODTx Definition (Same Cycle as $\overline{\text{BG}}$),” do not include qualification with $\overline{\text{TS}}$ or $\overline{\text{DBG}}$ on the cycle previous to the $\overline{\text{BG}}$. If ODTx is delayed a cycle as described in this section, then its value should not include any changes from the prior state (the $\overline{\text{BG}}$ state) or the second prior state (the state before $\overline{\text{BG}}$ is asserted) in order to have the same value as if it were driven with the $\overline{\text{BG}}$.

1.3 Multiprocessor Options

In a multiprocessor (MP) system, the arbiter only needs to provide one ODTx for an entire system. Because ODTx can be driven coincident with the assertion of a $\overline{BG}$ signal, each processor can have a unique value. This method saves pins by not requiring ODTx to be driven for each processor simultaneously.

However, the IA has to track valid $\overline{BG}$s for each processor in order to correlate all of the address and data tenures of all processors.

In the case of a bus trace for an MP system where one processor is performing address and data tenures and the second only asserts $\overline{HIT}$ (as shown in Figure 4), the IA only needs to know the address used by the one processor in order to match the address tenure with an intervention data tenure.

The IA could alternately determine the ODTx for the intervening processor in this scenario. Figure 4 shows this scenario. In cycle 6, $\overline{DBG}$ is asserted for both processors. Although the ODTx for processor one is unknown, the address is the same as that for processor zero. Using the DTIx value for processor one as a reference, the ODTx can be calculated as being six for processor one because the second-to-last $\overline{HIT}$ assertion was the sixth oldest data tenure [DTI = 5] and one more $\overline{HIT}$ assertion followed.

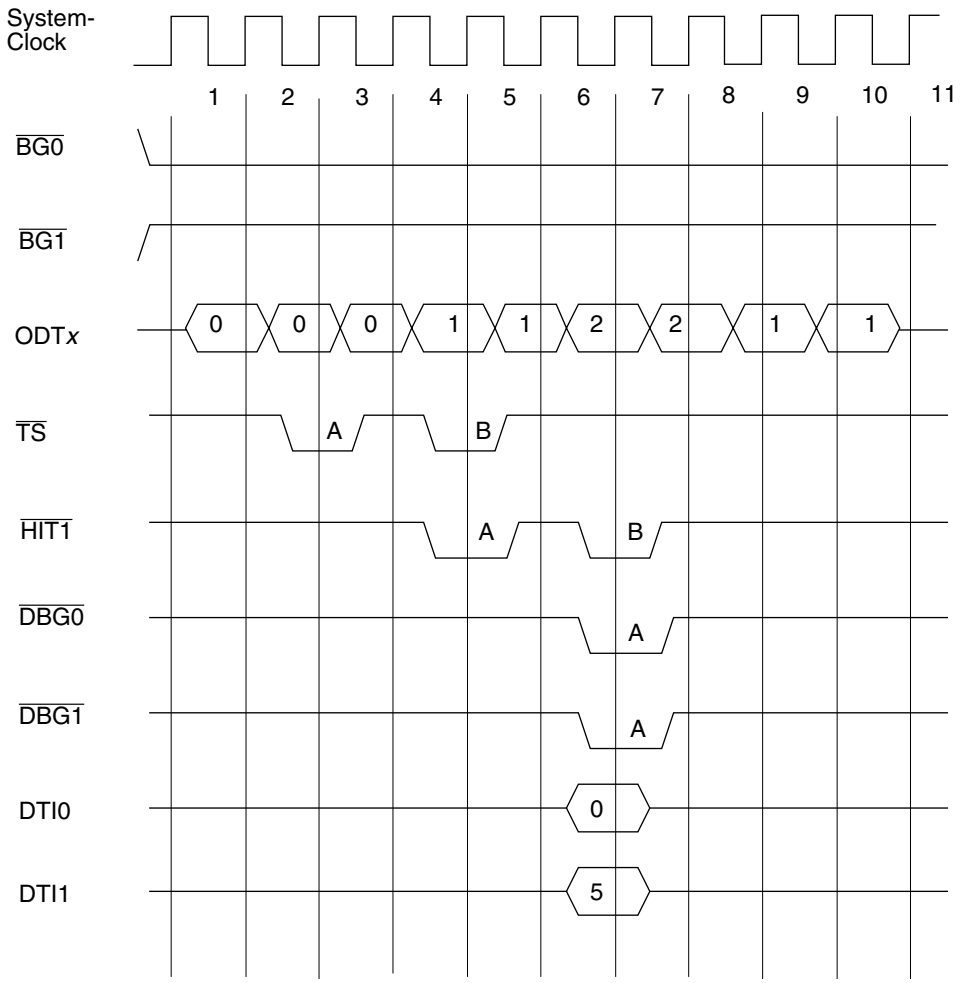

Figure 4. Trace with Intervention and Alternate Method to Calculate ODTx

1.4 Revision History

Table 4 lists this document's significant changes and revisions.

Table 4. Document History

Revision	Changes
0	Initial release
0.1	Nontechnical reformatting

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

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

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

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

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

freescale™
semiconductor

AN2161/D