Execution Tracing 23

The embedded Pentium® processor family uses special bus cycles to support execution tracing. These bus cycles, which are optional, have a significant impact on overall performance. Execution tracing allows the external hardware to track the flow of instructions as they execute inside the processor.

The special bus cycles generated by the processor are Branch Trace Messages (BTM). Due to physical limitations, the maximum number of outstanding taken branches allowed is two. Once the second taken branch reaches the last stage of the pipeline, execution is stalled until the first branch message is sent on the bus.

Branch trace messages may be enabled by setting the Execution Tracing bit, TR, of TR12 (bit 1) to a 1. Once enabled, there are two forms of branch trace messages: normal and fast. Normal messages produce two cycles, one for the branch target linear address, and one for the linear address of the instruction causing the taken branch. Fast messages only produce the second of these two cycles. The second message will always contain the linear address of the instruction executed in the u pipe even if the instruction that caused the branch was executed in the v pipe. For serializing instructions and segment descriptor loads the address field of the first cycle will contain the address of the next sequential instruction after the instruction that caused the BTM. Fast execution tracing is enabled by setting bit 8 of TR12 to 1. Note that switching between the normal and fast formats by using the WRMSR instruction to change bit 8 of TR12, the WRMSR instruction causes a branch trace message when they are enabled. The format for this branch trace message will be the format that was programmed *before* the WRMSR instruction was executed.

Normal and fast branch trace messages may be delayed by 0 or more clocks after the cycle in which the branch was taken depending on the bus activity. Also, higher priority cycles may be run between the first and second cycles of a normal branch trace message. In dual-processor mode, branch trace message cycles may be interleaved with cycles from the other processor. Branch trace message cycles are buffered so they do not normally stall the processor.

Branch trace messages, normal and fast, may be identified by the following special cycle:

M/IO#	= 0
D/C#	= 0
W/R#	= 1
BE7#–BE0#	= 0DFH

The address and data bus fields for the two bus cycles associated with a branch trace message are defined below:

First Cycle (Normal)

A31–A4	Bits 31 – 4 of the branch target linear address
A3	“1” if the default operand size is 32 bits “0” if the default operand size is 16 bits
D63–D60	Bits 3 – 0 of the branch target linear address
D59	“0” - indicating the first of the two cycles
D58–D0	Reserved. Driven to a valid state, but must be ignored

Second Cycle (Normal)

A31–A4	Bits 31 – 4 of the linear address of the instruction causing the taken branch
A3	“1” if the default operand size is 32 bits
	“0” if the default operand size is 16 bits
D63–D60	Bits 3 – 0 of the linear address of the instruction causing the taken branch
D59	“1” - indicating the second of the two cycles
D58–D0	Reserved. Driven to a valid state, but must be ignored

Fast Cycle

A31–A4	Bits 31 – 4 of the linear address of the instruction causing the taken branch
A3	“1” if the default operand size is 32 bits
	“0” if the default operand size is 16 bits
D63–D60	Bits 3 – 0 of the linear address of the instruction causing the taken branch
D59	Driven to a “1”
D58–D0	Reserved. Driven to a valid state, but must be ignored

In addition to conditional branches, jumps, calls, returns, software interrupts, and interrupt returns, the processor treats the following operations as causing taken branches:

- Serializing instructions
- Some segment descriptor loads
- Hardware interrupts
- Certain floating-point exceptions (both masked and unmasked) and all other exceptions that invoke a trap or fault handler
- Exiting the HALT state

With execution tracing enabled, these operations will also cause a corresponding branch trace message cycle. The processor data bus is valid during branch trace message special cycles. Instructions which cause masked floating point exceptions may cause one or more branch trace special cycles. This is because execution of an instruction may be aborted and restarted several times due to the exception.

Also note that the WRMSR instruction to enable branch trace messages will cause a BTM to be generated (WRMSR is a serializing instruction and serializing instructions cause BTMs). A WRMSR to disable BTMs will not generate a BTM. Conditions which cause the VERR, VERW, LAR and LSL instruction to clear the ZF bit in EFLAGS will also cause these instructions to be treated as taken branches. However, if these instructions fail the protection checks, no branch trace message will be generated.

Note that if an instruction faults, it does not complete execution but instead is flushed from the pipeline and an exception handler is invoked. This faulting instruction effectively causes a branch; a branch trace message is generated accordingly.