# Enhancements to SCE-MI 1.0, First Principles

## Matt Kopser, Cadence Design Systems

This document outlines the framework in which Cadence has created the proposed modifications to SCE-MI 1.0.

## **1.0 Basic Functional Agreement**

The two major functional changes embodied in the proposal have been discussed and agreed, in principle, within the Accellera ITC committee. These are:

#### 1. Native support of variable-length messages (VLMs)

In SCE-MI 1.0, segmentation and reconstruction of VLMs must be done by the writers of BMFs and tests. Native support of VLMs provides convenience to modelwriters, and is understood to be applicable across a broad range of transactor protocols.

In addition to the convenience it provides to model-writers, native support for VLMs promotes the notion of a VLM to the infrastructure itself. Knowledge of VLMs within the infrastructure affords vendor implementations the opportunity to tune such message transfers for performance.

## 2. Elimination of the uclock/cclock mechanism for as broad a class of BFMs as possible

The elimination of the uncontrolled clock mechanism is considered for enhancement to the specification, in order to support a verification-platform-independent methodology for writing BFMs. Removal of the uncontrolled clock addresses, specifically, the issue of timing congruence across cycle-based and event-based verification platforms.

The proposal will define new port macros and clock semantics to support a use model whereby congruence can be maintained across cycle-based and event-based verification engines.

#### 2.0 Backward Compatibility Requirements

The proposed changes to the specification will be compatible with the current (SCE-MI 1.0) specification. Thus, models which adhere to the SCE-MI 1.0 standard will remain compliant within the newly-proposed standard.

Note that the use of SCE-MI 1.0 port macros and clock control mechanism will result in systems which are not time-congruent across multiple verification engines.

#### 3.0 Performance

One of the primary goals of the SCE-MI standard is to provide high-performance coemulation. The proposed changes to the SCE-MI standard must not compromise the ability for vendors to provide high-performance implementations of the existing (1.0) SCE-MI standard. Similarly, decisions regarding the definition and semantics of proposed changes to the standard must be made with high-performance co-emulation as a top priority.

## 4.0 Separability of current and new port macros

This requirement is derived directly from item 2 in section 1.0 above.

The newly-proposed port macros and the associated (implicit) clock definition and control mechanism constitute a complete, usable subset of the SCE-MI specification. As such, user environments can be constructed entirely based on the newly-proposed macros, without the need for the clock control mechanism as currently defined in the SCE-MI 1.0 specification.

This use model constitutes a platform-independent congruent methodology.

## 5.0 Support broadest range of models without uclock/ cclock mechanism

This requirement relates directly to the zero-time operations (ZTOs) that the uclock/ cclock mechanism allows. It is through manipulation of the controlled clocks that SCE-MI 1.0-based BFMs can support zero-time operations. The message port macros proposed as an enhancement to the SCE-MI 1.0 standard do not provide the BFM with explicit control over the user (controlled) clocks in the design.

The acronym ZTO, as used in this document, refers to operations that do not advance the user-visible time on the hardware side of a SCE-MI-based system. That is to say, (in SCE-MI 1.0), these operations occur when the controlled clocks in the system are inactive. In actuality, in SCE-MI 1.0, the uncontrolled clock, which runs continuously, is utilized to clock state machines on the hardware side of the system in order to perform ZTOs. Thus, real clock cycles do elapse during the execution of ZTOs. (This is the root of the incongruence issue discussed in section 1.0, item 2.) The proposed enhancements to the SCE-MI specification include a set of new message port macros which eliminate the explicit reliance on the uclock/cclock mechanism. Sections 5.1 through 5.3 address the various types of ZTOs, and how the newly-proposed message port macros relate to them.

ZTOs can be categorized into 3 types, outlined in the following sections.

#### 5.1 ZTOs of Unbounded Duration

It is possible to write a BFM that implements a relatively complex state machine, which requires an indeterminate number of clock cycles to perform a 'zero-time' operation. These types of models are supported via the SCE-MI 1.0 uclock/cclock mechanism. Note, however, that the use of the uclock/cclock mechanism breaks the congruence principle that is fundamental to the proposed enhancements to the SCE-MI standard. (See section 1.0, item 2.)

Use of the uclock/cclock mechanism to support operations of indeterminate duration is limited to a small subset of model types (e.g. embedded stimulus generators and response checkers), and is not required for the vast majority of BFM models.

There is no attempt made in this proposal, to address models which require indeterminate cycle-count operations in zero time, within the context of the newly-proposed message ports.

#### 5.2 ZTOs of Fixed-Length

A BFM may have a requirement to perform a relatively simple data manipulation on message data coming into, or going out of the BFM from/to the infrastructure. This operation may require 1 to several cycles to complete. The clock control mechanism of SCE-MI 1.0 supports this explicitly.

The proposed change to the SCE-MI specification supports these types of operations. There are two techniques which can be used to support such operations in the BFM:

#### 1. Adjust the message port data width

If the requirement for data manipulation is rooted in a need to manipulate successive data words, the number of message ports and/or message port widths can be sized to accommodate the need for such data.

2. Clock the BFM-Message Port interface at a higher rate than the BFM-DUT interface

This 'over-clocking' mechanism can, under certain circumstances, result in an overall system performance degradation.

Note, however, that this mechanism may not adversely affect system performance at all, depending on the existence and relative frequency of other clocks in the system. Also, software-side activity and message transport may dominate overall system performance, so as to minimize the impact of over-clocking.

#### 5.3 Simple ZTOs (GET and PUT)

Tests may require static control over BFM configuration and/or access to BFM internal state. In contrast to the types of operations described in section 5.1, these 'in-between' accesses do not require data manipulation within a hardware state-machine. These operations are limited to simple GET and PUT operations on configuration or status-type data. Such operations are explicitly supportable by the uncontrolled/controlled clock mechanism.<sup>1</sup> In this proposal, a means of providing zero-time access 'between' messages within the context of the newly-proposed message port macros will be described.

Without explicit support of such a mechanism, configuration/status functionality could be achieved (with some drawbacks) at the application level, in one of two ways:

- The user's test application could rely on explicit messages (either through ports dedicated for this purpose, or multiplexed onto message ports used for 'normal' messages) to transfer configuration data to, or extract status information from, a BFM. The drawback to this approach is that real user-visible cycles must be consumed in order to transport such data. This can present visibility and debug issues. Or, worse, this can make it difficult or impossible to create desirable timing relationships between messages on multiple message ports.
- The application can utilize the 'over-clocking' mechanism identified in section 5.2. This approach is problematic for several reasons. Most notably, each BFM requiring such data transfer would be required to multiplex incoming and outgoing data on one or more message ports. Also, the noted potential performance problem with this mechanism is likely to be an issue for these types of data transfer requirements. For example, it would be difficult to justify an overall degradation of system performance in order to transfer a few configuration words at the beginning of a long test.

<sup>1.</sup> Again, note that the uncontrolled/controlled clock mechanism breaks the congruent use model.