Modeling Cycle Information and Cycle Aware Simulation

Introduction

SANKHYA Teraptor supports modeling Plant/Domain Components (using C++), Processor Models (using SMDL) and System and System of System Models (using SSDL). The cycle level behavior of processors and devices can be specified using attributes provided by SMDL and SSDL. The cycle aware edition of Teraptor Player (which plays the models) enforces the cycle-level behavior during simulation. The performance of a system modeled with cycle information can be studied with different applications as well as for different settings of the cycle attributes, thereby covering a spectrum of hardware timings and system loads.

Specifying Cycle Information in Processor Models

The SANKHYA Machine Description Language (SMDL) provides attributes for modeling cycle-level behavior of processors [1]. These attributes specify the relative clock-cycle when each component of a CBDL expression will be executed.

The *cycle* attribute is attached to an operator and specifies when the corresponding operation is performed relative to the start of the expression. The *read-cycle* attribute is attached to an operand and specifies when the operand is fetched. The *write-cycle* attribute is attached to an operand and specifies when the operand is written back.

Example

Add instruction that takes 3 cycles. Registers r2 and r3 are read during cycle 1, the add operation is performed during cycle 2 and the result is written back to register r1 during cycle 3. The CBDL expression for this instruction is given below.

```
= {write-cycle=3}r1{cycle=2}+{read-cycle=1}r2{read-cycle=1}r3
```

Specifying Cycle Information in Domain and System Models

Teraptor supports modeling of domain/peripheral devices through a C++ API [2]. This API supports a Component class which provides a cycle() method for modeling cycle-level behavior Actual domain/device components can derive from the Component class and define the cycle() method to specify cycle-level behavior of the component.

For example, a bus controller model can specify the behavior of the controller during each cycle by defining the cycle() method.

```
SSDLErrorCode BusController::cycle(int cycle_count)
{
      //Cycle level behavior
}
```

The domain and system models can support attributes to specify the number of cycles required for completing a specific operation. For example, a Bus model can support attributes to specify the number of cycles taken to transmit a frame, the number of idle cycles between two frames etc.

```
component can_bus can_bus fs=100 ifs=50
```

Here, the attribute 'fs' indicates the number of cycles in the frame state and ifs indicates the number of cycles in the inter-frame state for the CAN bus component.

A Cache memory model [3] can support attributes for the number of cycles taken to read data when it is present in the cache (hit_cycles) and the number of cycles taken to read data when there is a cache miss (miss_cycles). These attributes can be configured when instantiating a component of the specified model in the SSDL description of the system. For example, the following SSDL statements define a cache memory component with hit cycles of 2 and miss cycles of 4.

```
component Cache cache size=0x500000 hit_cycles=2
miss_cycles=4 <other attributes>
```

Playing Cycle Aware Models

The cycle aware edition of Teraptor Player can play systems with cycle-level information. The Player supports *cycle* and *scycle* commands [4] to perform a cycle operation on active components – processor, peripheral devices and controller of the system. The *cycle* command invokes a cycle operation on the currently active processor component and the *scycle* command invokes a cycle operation on all active components of the system – viz. Processors, peripheral devices, domain components etc.

```
; Execute one cycle on all active components
scycle
; Execute 10 system cycles
scycle 10
```

Reference

[1] Sankhya Teraptor User Guide and Reference Manual – Section 17.5.1.9 *Cycle Attribute*
[2] Sankhya Teraptor User Guide and Reference Manual – Chapter 18 *Teraptor Device API Reference*

[3] Sankhya Teraptor User Guide and Reference Manual – Chapter 22 *Cache Memory Unit*
[4] Sankhya Teraptor User Guide and Reference Manual – Sections 11.4.2.9 and 11.4.2.10 *Execution Control Commands – cycle and scycle*

Technical Support

Contact support@sankhya.com for more information.

SANKHYA™

Sankhya Technologies Private Limited
Contact:
+1 408-556-9757 (USA)
+91 94449 72818 (India and South Asia)
http://sankhya.com/contact.html