UVM - Day26

Constructing Register Model ⇒

Register Field :-

Register field are declared in register class with uvm_reg_field.

- The access policy of a field is specified using the uvm_reg_field :: configure() method.
- Configure method has to be called from the build() method of the register that instantiates it.

map() Method :-

A virtual map() function, with uvm_reg_map and address offset arguments map() method shall call uvm_reg_map :: add_reg() for all register class properties.

- The map() method may call the add_hdl_path() method for all register / register file class properties.

set_offset Method :-

A virtual set_offset() function, with a uvm_reg_map and address offset arguments, may also implemented.

- The set_offset() method shall call the set_offset() method for all register and register file class properties.

Memory Types :-

A memory type is constructed using a class extended from the uvm_mem class.

- The name of the memory type class must be unique within the scope of its declaration.

Packaging and Integration Register Model :—

Packaging a Register Model ⇒

The following practices are recommended but not required.

- Block types should be located in separate packages.
- ~~Block types~~ A header file, with all the required import statements to use the register model, should be generated.
- A lengthy build() method may be split into several, shorter sub-methods. The sub-methods shall be declared local and called by the build() method.

Integrating a Register Model ⇒

A register model must be integrated with the bus agent.

- The integration with the bus agent must only be done on root blocks.
- Root blocks model the entire DUT and they are only ones who have access to and knowledge of the externally-visible address maps. i.e in the environment of the testbench.

UVM Register Model Predictor :—

UVM RAL Predictor predicts the register access done through the register model and updates the RAL Model registers.

(i) **Implicit prediction ⇒**

Implicit prediction only requires the integration of the register model with one or more bus sequencers.

(ii) **Explicit prediction ⇒**

Explicit prediction requires the register model to be integrated with both the bus sequencer and corresponding bus monitors.

UVM RAL Adapter :—

The RAL adapter acts as a converter b/w the RAL model and interface. It converts transactions of RAL methods to interface/bus transactions.

- The adapter converts b/w register model read, write methods and the interface - specific transactions.
- The transaction adapter is implemented by extending the uvm_reg_adapter class and implementing the reg2reg() and ~~bus2reg~~ bus2reg() methods.

UVM RAL reg2bus :—

reg2bus method converts the RAL transactions to interface (bus) transactions.

UVM RAL bus2reg :—

bus2reg method converts the interface (bus) transactions to RAL transactions.