SEtSim Reference Manual

Mohammad Ashjaei

November 6, 2013

1 Getting Started

This reference manual describes the use of SEtSim Rev 2.0, which is desinged and developed based on Simulink. This manual contains the fundamental steps to create a simulation model and generating the outputs from that. This includes how to generate a model of three different architectures in the FTT-SE protocol, i.e., single-/multi-master and cluster-based architecture.

SEtSim requires Matlab 7.9 (R2009b) with Simulink 4.1 or later versions. All the functions are developed in Matlab m-file, therefore C++ compiler is unnecessary to run the tool. The folder of SEtSim contains *model_lib.mdl* which is the library consisting of master, slave and switch blocks. Moreover, the *data_table.m* is a Matlab m-file comprises all parameters and structures which are needed to be set before running a simulation.

2 Creating a Model

The blocks in the SEtSim are connected with ordinary Simulink connection lines to/from a network model. Before running a simulation, all connections for blocks in particular for slave nodes, master nodes and switches should be done as well as configuration of the model. To create a new model, open a model inside Matlab under the SEtSim folder. Copy the blocks that are required for the example from the *model_lib.mdl* into the new model. The library model is illustrated in Figure 1.

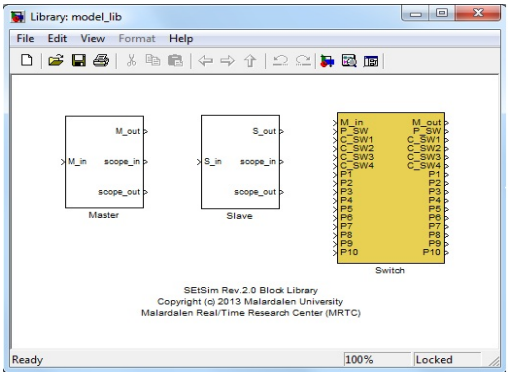

Figure 1: SEtSim Blocks Library

The network model can be created using ordinary sub-systems of Simulink to make each sub-network as one sub-system in the model. This can help to have better structure for large-scale network examples. Figure 2 shows an example with three sub-systems in Simulink Model, each of which contains a part of network including one switch, one master node and three slave nodes.

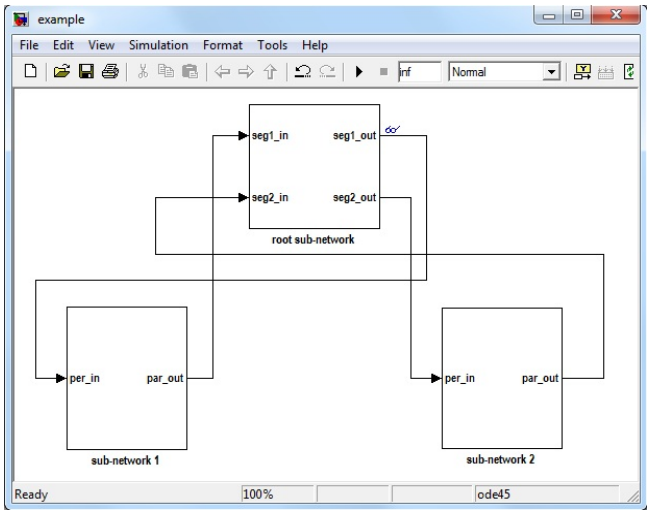

Figure 2: An Example of a Network Containing Three Sub-Networks

Note that, the network example should be created as a tree topology. In this example the root sub-network comprises a switch with three slave nodes as it is illustrated in Figure 3.

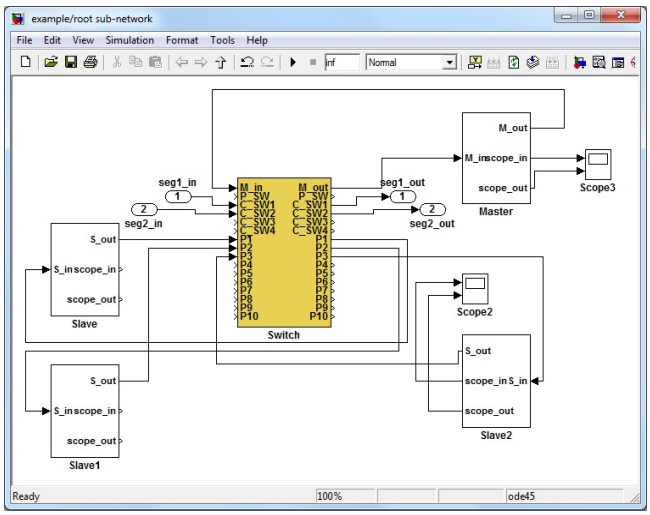

Figure 3: The Root Sub-Network for The Example

In the single-master architecture the root sub-network should have a master node and for other sub-networks it is not necessary to connect a master block. However, in the multi-master architecture each sub-network should have a master node and in the cluster-based architecture the network should have a master node per each cluster.

After completing the model connections, each block should have a unique identifier. For this purpose by double-clicking on each block a parameter window appears which the user needs to set a unique number to that. An example of parameter window is depicted in Figure 4. The order of assigning the identifier is important. The switch and the master node in each sub-network should have the same identifier. Also, the identifier of the root switch and the master node connected to it should be set to 1, essentially, due to identify the root level in the network. Continuously, the other switches and the master nodes connected to that (if any) in the second level of hierarchy should be assigned to 2, 3 and so on. This procedure should continues until the last level in the network hierarchy. This is simialr for the slave blocks which the parameters should be set for the root sub-network and continue to the last level in the hierarchy.

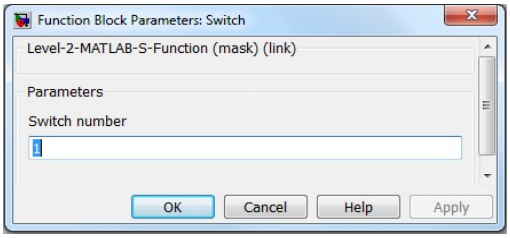

Figure 4: The Block Parameter

3 Configuration

In order to configure the setting of each example, the user should set some parameters manually inside the *data_table.m* Matlab m-file. The parameters that should be changed along with the description of them are shown in Figure 5.

In addition, the parameters for the messages should be configured manually inside *message_declare.m* Matlab file. The parameters that should be set for each message is shown in Figure 6.

After configuring the messages and the model, the example is ready for the experiment. Using start and stop in the simulink model, the simulation initiates to run.

4 Outputs and Reports

All the message transmissions are generated in scope-in and scope-out of each block. The ordinary scope block in Simulink is used to visualize the message transmission. All the messages are shown by their identifier number in the scope. Moreover, to analyse the end-to-end delay of the messages, the report funtion is used. Using *report* command in the Matlab command window, after

the simulation is stopped, the information of the model consisting of slave and message numebrs in the model along with the end-to-end delay of them will be presented.

Parameter	Description
architect	Shows the FTT-SE architecture of the model, i.e., single-/multi-master or cluster-based architectures. Value 1 for cluster-based, value 2 for multi-master and value 3 for single-master architecture.
slave_number	Number of slave nodes which is used in the model.
switch_number	Number of switches used in the model. In multi-master architecture it is equal to number of master nodes.
cluster	Number of clusters in the network.
EC	The duration of EC in microseconds.
tw_win	The duration of TM transmitting. The recommended value is 200 micoseconds.
synch_win	The duration of synchronous window within EC in microseconds.
local_per_win	The duration of local/internal window within synchronous windoe in microseconds. In case of single-master architecture it is equal to synch_win.
global_per_win	The duration of global/external window within synchronous in microseconds. In case of single-master architecture it is equal to 0.
local_aper_win	The duration of local/internal window within asynchronous window in microseconds.
cluster_win	The duration of cluster sub-window within asynchronous window in microseconds. All clusters are assumed to be equal.
Identify the cluster number for each master node	To assign each master to the cluster that it belongs to. For instance, master(1).cluster = 1, which means master number 1 belongs to cluster number 1. This should be done for all master nodes.
Identify the parent master for each slave node	To assign a parent master for each slave node. For instance, slave(1).parentNumber = 1, which means slave 1 has a parent master number which is master 1.
Identify the segment number for each slave node	To assign each slave node to its segment number. For instance, slave(1).segmentNumber = 1, which means slave 1 is inside the sub-network with switch sumber 1.
Define the port for each slave node	Each slave node is connected to a specific port of the switch, which is defined here. For instance, slave(1).swPortNumber = 7, which means slave 1 is connected to port 7 of the switch.

Figure 5: The Model Configuration

Parameter	Description
id	The identifier for each message that is unique in the model.
prio	Priority of the message.
exec	Transmission time of the message that is set in microseconds / 10000.
gType	It shows the type of message if it is local/internal, it should be 0 and if the message is global/external it should be set to 1.
pType	It shows the type of message if it is periodic or aperiodic. In case of periodic it should be set to 0, otherwise if it is aperiodic it should be set to 1.
period	Period of the message in number of Ecs.
deadline	Deadline of the message in number of Ecs.
dynPeriod	In case of periodic message it should be set to 0, otherwise if the message is aperiodic it should be set equal to period.
source	The source slave number.
destination	The destination slave number.
sourceMasterNumber	The switch number that the source slave belongs to.
destMasterNumber	The switch number that the destination slave node belongs to.
SwitchInRout	The switch numbers that the message passes until the destination, excluding the destination switch. For instance, [2 1] means the message will pass switch 2, switch 1 and the destination switch.
SwitchPort	The output port for the switch that the message crosses. It should be equal size of SwitchInRout. For instance, [2 4] means the message goes out from port 2 of switch 2 and then goes out of port 4 of switch 1.
tickCount	Should be set to period – 1.
readyTime	Should be set to 0.
receiveTime	Should be set to 0.
defNum	Should be set to 0.
part	Should be set to 0.
refMsgID	Should be set equal to id.
cluster	It shows the cluster that the source of the message belongs to. This is just for global/external messages, other messages do not need that which should be set to 0.
ClParentMaster	It shows the parent master of the cluster. It should be set just for global/external messages and for the other typrs should be set to 0.
offset	To set any offset for the message in number of Ecs.

Figure 6: The Message Parameters