Accellera ALC – C++ Initiative Working Group meeting 09/072000

Present at Mentor Graphics:

Brian Bailey – Mentor Graphics
John Sanguinetti – CynApps
David Springer – CynApps
Frederic Doucet – Intel / UCI
Wolfgang Nebel OFFIS / Oldenburg Univ
Kamal Hashmi – ICL (phone)
Joe Daniels – Intrinsix
Martin Baynes – C Level
Asa Ben Tzur – Intel
Dan Gajski - UCI

The meeting was not properly structured because of my preceding vacation time and the limited time availability of Dan Gajski and John Sanguinetti. We thus conducted this meeting in a less than ideal manner and failed to review certain procedural items such as action items from previous meetings. However, it was a good meeting during which I think we made significant progress.

The meeting started with Dan reviewing some of his comments on the rules that had been created in the previous meeting and documented by Martin. Some of these issues were ones of remembering or understanding the rules and confirmed that these need to be tightened up to assure that the meaning of all rules is precise.
Dan had two primary issues. The first was dealing with the need for a High Impedance state 'Z'. His argument was that this is an artifact of implementation and not an RTL construct and does not need to be included for an FSMD description. While this was generally agreed, there was also a strong argument that customer designs in reality require these even though the theoretical model doesn't. Much of this has to do with how the user's design is partitioned and the types of interconnect modeling that they use. David argues for the fact that even at RTL, user design contain busses and Z's are a necessary part of a bus model. This broadened into a discussion about unknown values - How do you resolve the difference between unresolved, I don't know what my value is etc. Dan is arguing that for every signal it is either defined or undefined and if defined has a value. David wants to have a signal be defined but without a value. Martin- as a minimum it must be an attribute of a driver so that it can be resolved and expressed to the user as an error.

** Table this for now. Must come up with examples that are for and against and discuss at next meeting.

Some other points that came out of Dan's comments:
a- iv naming issue on wire. Continuous evaluation of an expression that is more than the normal expectation of a wire. Undefined is for simulation. Don't care is for synthesis – a directive. Decide later if actually needed.
a- viii needs to be revisited when an example can be shown for its need.
a- x still needs further discussion. Again example needed. Lump together with viii

b - v. Still needs further simplification. It needs to talk about all wires executing before all registers, but still needs to consider multiple clocks etc.

Need to carefully consider reset, especially async reset which overrides the behavior of the FSMD. Is this a special kind of flop or an attribute. It can also change the state of the state register. Should it be limited to one asynchronous input.
Reset is an example of a signal that has the ability to change the value of a register or state register without the need of a clock. Not all registers need to be controlled by the same signal.
The only way to correct this is to introduce a third primitive. Wire, Register and Register with Asynchronous capabilities.

There appears to be a large amount of confusion, at least in my mind, as to what a wire is and what level of persistence exists for them. Many of these confusions are associated with the desire to create efficient simulation, which requires that values are sampled at discrete times rather than being constantly sensitive to the input values that drive them. If the signal is sampled, it seems to me that there must be an implied register to hold this value. Then there is the question about if this is persistent over a clock period. For Cynlib, it automatically detects in implementation if it needs to implement it as a wire or register.

Functions that execute asynchronously must always execute completely. This implies that all initial conditions must be assigned before the function is performed, else it finishes up having implicit storage defined within the function.
The final result of this is a question. Should level 4 require that all wires and registers are explicit. If rules can be written that enable in a language to define exactly what it is, then it is explicit and passes the semantic check.

Need to go back and clarify the levels and start to work towards a more formal definition of these semantics.

We started trying to formalize a notation for the types of expressions and connections so that we could explicitly define how simulation should deal with them and could only be interpreted one way for synthesis. Please see the first draft of these on the following page. Green lines are not allowed to cross state boundaries since they have become out of scope.
Frederic felt uncomfortable since he wanted to see values persistent over many cycles. However, the behavior of this is uncertain since the inputs are not controlled during this period and thus becomes timing dependent.

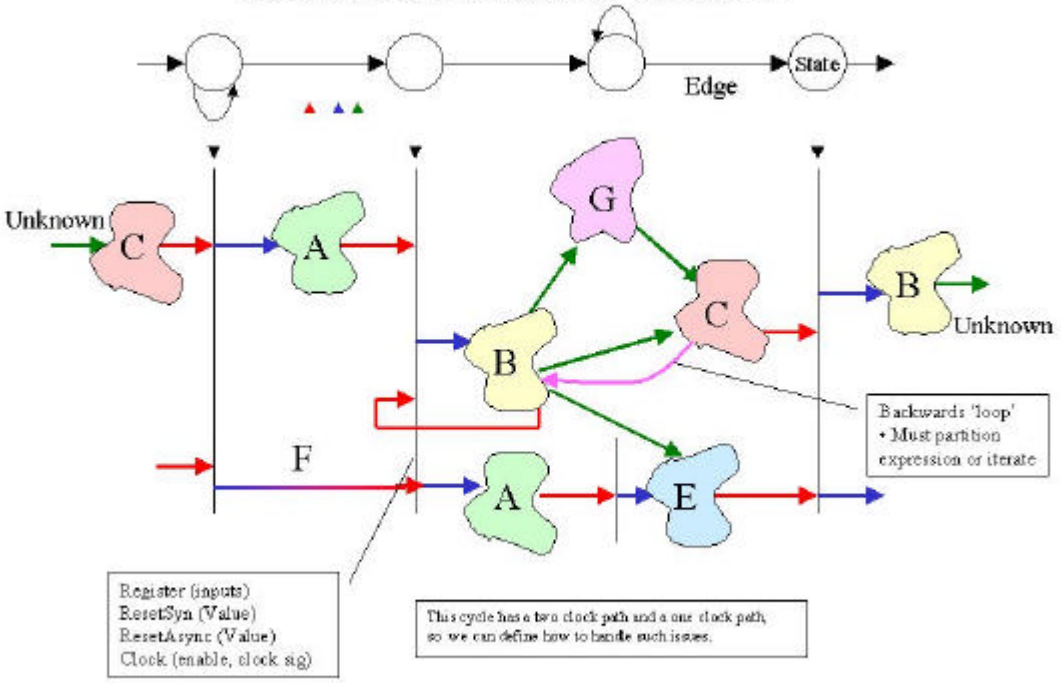
Action Items

** Get Dave Gjalt email address so that he can review the VSIA Datatypes spec. (Brian Bailey)
** Contact Steve Shultz about contributing to this group (Brian Bailey)
** Need examples to demonstrate the need for Z,X etc (All)
** Clarify definitions of the levels at which we need to define the semantics. What is implicit and explicit in these models.
** Collect examples. Asa to collect from people by Tuesday (All)

Next Meetings

September 21st phone conference 9:00 till 10:00
October 5th. Face to face 9:00 till 4:00 Host required !

Generalized FSMD



Transformation Expression Types Classified