Scheduling of time events in the circuit simulator

Shivkumar .V. Iyer

1 Introduction

This short paper will describe how events are scheduled in the circuit simulator. The essential events are the execution of circuit analysis and the storage of simulation data in the output data file. This paper will describe how the time step of circuit analysis is critical to the stability of the simulation. Since the circuit simulator is primarily focused on power electronics applications, a critical component of the simulation is the execution of user-defined control functions. These control functions may be for controlling power converters or for signal processing applications. In any case, implementation of control functions and signal processing in discrete-time requires an accurate control on the frequency of their execution. This paper will describe how this is achieved in the circuit simulator with the provision of time events defined for every control function.

2 Time constants in a circuit

Fig. 1 is a simple circuit consisting of an ac voltage source V feeding a resistor-inductor. With this basic circuit, the importance of simulation time step will be described. The differential equation for the current in the above circuit will be:

$$V - Ri - L\frac{di}{dt} = 0 \tag{1}$$

To solve the above differential equation numerically, let us consider a simple form of numerical integration with a constant time step of ΔT. The simplest solution to the above differential

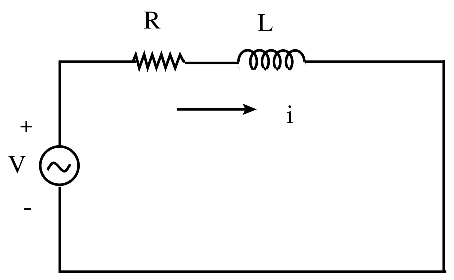

Figure 1: A simple R-L circuit

equation in discrete form will result in the following equation at time instant $t = tk + \Delta T$:

$$i(t = tk + \Delta T) = i(t = tk) + \frac{V(t = tk + \Delta T) - Ri(t = tk)}{L} dt \tag{2}$$

In a circuit such as in Fig. 1, the time constant is defined by the ratio L/R. This factor can also be seen in equation (2). Without jumping too deep into network theory, the time constant of the circuit is an indication of the time delay inherent in the circuit or how fast a circuit can respond to changes in the input. The reader is encouraged to read further on this topic for more details. In simple terms, the inductor in Fig. 1 introduces a time delay due to which the current lags the voltage. If the inductor was not present and $L = 0$, the purely resistive circuit that would result would respond instantly to any changes in the voltage. As an example, in a resistive circuit, the current would be in-phase with the voltage but scaled by a magnitude according to Ohm's Law $V = iR$. Moreover, a circuit without inductance would result in an equation for current that was purely algebraic without a differential term.

Returning to the circuit of Fig. 1. The ratio L/R defines the rate at which the circuit responds to changes in the input voltage. Therefore, in order to simulate this circuit accurately, it would be necessary to capture the response of the circuit to the input voltages. To offer a real world example, if you were to video record an event, the speed of recording has to be significantly faster than the speed of the event or else there will be loss of potentially vital information. In the similar manner, a simulation will need to use a time step significantly smaller than the time constant of the circuit in order to capture the transients of the circuit. This is an extremely crucial factor in the simulation of complex circuits - to choose a time step smaller than the lowest time constant in the circuit.

The choice of the simulation time step can now be discussed. It is possible to vary the simulation time step so as to speed up the simulation. When the circuit is experiencing slow transients to inputs, the simulation time step can be increased without losing accuracy. When the circuit is rapidly responding to inputs, the simulation time step is decreased to be able to capture the fastest transients. Though in theory this concept is valid, in practice it is effective only for fairly simple circuits. For a large and complicated circuit especially with interconnecting impedances as might be the case with a distributed power system, the time step required to simulate it accurately is almost always small. Therefore, the possibility of speeding up the simulation is almost ruled out. However, if a minimum limit on the simulation time step is not imposed, it is possible that the simulation time step may end up becoming so small that the execution may slow down significantly. In large and complex circuits, it may become necessary to ignore some transients that are simply too fast to capture rather than attempt to simulate at an incredibly minute scale. For this reason, the circuit simulator uses a constant time step for simulation.

The choice of the time step for simulating the circuit is left to the user. A fairly simple passive circuit can be simulated at a time step in milliseconds while most power electronic circuits would require a time step of around 1 to 5 microseconds. A high frequency resonant converter might need a time step in nanoseconds. This choice needs to be made by the user. It is advisable to try out a particular time step and then lower it to check the difference in the results. This simulation time step will define the time instants at which the circuit simulator will execute the circuit analysis (loop and nodal analysis) to solve the circuit. For a simulation time step of ΔT, the simulator would solve the circuit at 0, ΔT, $2\Delta T$, $\ldots$, $t_n - \Delta T$, t_n. Here, t_n is the time duration of the simulation which also needs to be specified by the user. Therefore, in this manner, the simulation time step ensures that the circuit is solved at a constant rate. The next section will describe how an event generated in the circuit could cause the circuit analysis to occur at time instants additional to those listed.

3 Time scheduling of control functions

The previous section described the time scheduling of the circuit solver with respect to the simulation time step. This is essential for a stable and accurate simulation. On the other hand, a user-defined control function may have a completely arbitrary execution time step. This may be because the control function may be a hardware implementation. The time step may be extremely small to mimic an analog functionality in the hardware or may be large to mimic the limited computational resources available in real-time with a microcontroller. Additionally, another reason for choosing a particular time step for a control function may be because of the nature of discrete-time control or signal processing implemented. This section will describe using an example how the circuit simulator ensures that each control function is executed at its desired time instant.

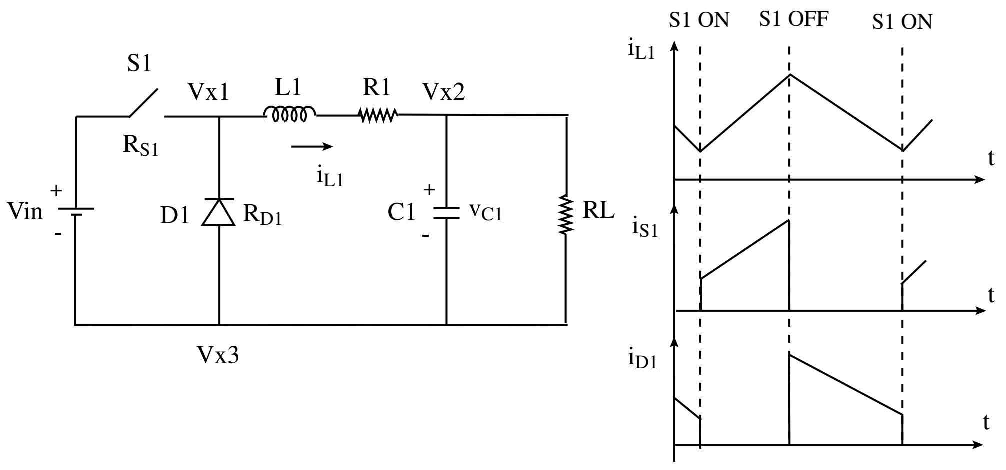

Figure 2: Operation of a buck converter

Fig. 2 shows the circuit of a buck converter with a sample case of switching of the Switch S1. Without going into the details of control design or implementation, let us examine how this converter would be controlled in the circuit simulator. The objective is to control the switching of S1 such that the capacitor voltage V_{C1} is regulated to desired reference. This desired voltage will be lesser than the input voltage V_{in} since the converter is a buck converter. When the switch S1 is turned ON, the input voltage V_{in} charges the inductor L_1 and the capacitor C_1 causing the inductor current i_{L1} and the capacitor voltage V_{C1} to rise. When the switch S1 turns off, the inductor current i_{L1} freewheels through the diode D1 and the capacitor discharges across the load. Therefore, the inductor current i_{L1} and the capacitor voltage V_{C1} decrease. If the switch S1 is switched at a constant frequency, the time for which the switch remains on can be controlled to regulate the output voltage V_{C1}. A simple Proportional-Integral (PI) controller can be used to generate a duty cycle which is then fed to Pulse Width Modulation (PWM) to generate the switching signals. The details of these will not be discussed here.

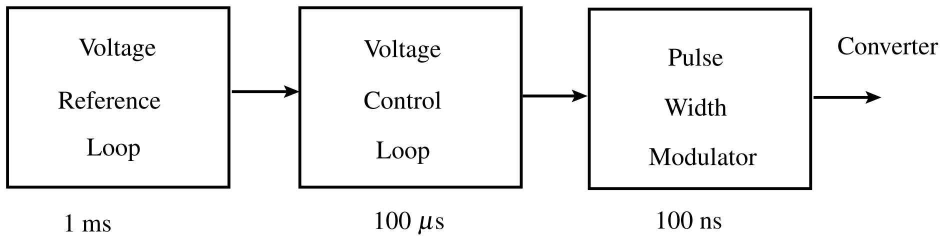

Figure 3: Nested control to describe timing

Fig. 3 shows a possible control strategy for the buck converter in Fig. 2. The control strategy consists of nested control loops with very different execution time steps. A brief description will explain the need for these different execution time steps. The first block on the left named "Voltage Reference Loop" sets the reference for the desired output voltage V_{C1}. An example could be a start-up algorithm that increases the reference from zero to a desired value to prevent large transient spikes in the inductor current. Such a control will be extremely slow since changing the output voltage reference rapidly could result in oscillations. An example of the execution time step is 1 ms as shown in Fig. 3. The next control block is the "Voltage Control Loop". This control loop varies the duty cycle according to the difference between the actual output voltage and the desired reference. A wide variety of control algorithms can be employed - both inductor current and output capacitor voltage can be used to generate the duty cycle. This control loop will be much faster than the outer loop as it needs to regulate the duty cycle which in turn will vary the on time of the switch S1. If the frequency of operation of the switch S1 is 5 kHz, the period of operation will be 200 μs. Therefore, as an example, the execution time step of this control loop is chosen to be 100 μs. The last control loop is "Pulse Width Modulator" which involves comparing the duty cycle with a sawtooth waveform of frequency equal to the switching frequency of 5 kHz. This control loop is quite often achieved either by hardware comparators or by a specialized module in a microcontroller. In order to ensure precise switching, the resolution of this control loop is quite often in nanoseconds. In this example, the execution time step of the PWM is chosen to be 100 ns.

This example shows how a control strategy for a converter can consist of nested loops with very different timing requirements. Let us suppose the simulation time step for this circuit is chosen to be 5 μs which is quite reasonable for a converter that switches at 5 kHz. The question is how will the simulator ensure that every control function executes at its desired time step while ensuring that the circuit solvers are executed at 5 μs? We now introduce the concept of time events. A time event is a special variable associated with a user-defined control function. When a user creates a control function, one or more time events for that control function can be defined. The user can update them in any way desired within the control function. Therefore, let us say the following time events are defined for the three control functions in Fig. 3 - tvolref for “Voltage Reference Loop”, tvolcon for “Voltage Control Loop”, tpwm for “Pulse Width Modulator”.

A time event serves a dual purpose. As an example, the time event tpwm can be used to define time instants that are updated every 100 ns with the following statement in the control function corresponding to “Pulse Width Modulator”:

```
tpwm = tpwm + 100.0e−9
```

With the above statement, the time event variable tpwm will change in discrete steps by 100 ns. This is just an example, the variable could be updated by any arbitrary step which could even be random. This variable allows the user defined control function to ensure that the control algorithm is not executed unless the simulation time instant is equal to this current value of tpwm. Moreover, by setting a value of tpwm, the control function gives a command to the simulator to ensure that the simulator will execute the function at this desired time step. This second concept can be explained in detail as follows.

The circuit simulator creates a vector called TimeVector to arrange all the time events generated. One time event that will be added by default is the time instant generated by the circuit solver. Let us call this time instant tode. The time vector will start as follows:

```
TimeVector = [tode]
```

If a simulator has no user-defined control functions, then tode will be the only element in TimeVector and therefore, the circuit solver will execute at every update of tdode which in this example will be 5 μs. With the control functions described above, the time events tvolref, tvolcon and tpwm will be generated by the respective control functions. They will be added to TimeVector as follows:

```
TimeVector = [tode, tvolref, tvolcon, tpwm]
```

The time events will be updated in the control functions as shown in Fig. 3. To decide the next time instant, the above TimeVector will be arranged in ascending order - lowest to highest. The lowest time instant will be the next time instant of execution. At this point, the emphasis should be placed on this word - “execution”.

The simulator will perform circuit analysis (loop and nodal analysis) to solve the circuit when an “event” occurs. An event is said to occur under two circumstances. The first is when the time instant of execution which is the lowest time instant in TimeVector is tode or the time step of the circuit solver itself. The reason is that this ensures that the circuit analysis occurs at minimum rate to ensure the simulation is stable. Failure to do so could result in an unstable or an inaccurate simulation. The second reason for an event to occur is if there is a change in the physical state of the circuit. Examples of change are if the resistance, inductance or capacitance of a branch change. This change could happen due to either control - a control function could turn on or turn off the switch S1 or an element can be modelled to reflect possible changes with time - as an example, the load resistance could change in value to show heating effect. Therefore, to sum it up, circuit analysis will happen if the solver commands it or if the control function changes the circuit and therefore commands it.

The time of “execution” is therefore not the next time instant when the circuit analysis is performed. Because the choice of a time instant need not generate an event. If the time instant chosen is tode, an event is generated automatically. But if the time instant is tvolref, tvolcon or tpwm, an event need not be generated. The circuit simulation now executes all the control functions provided by the user. If any of the control functions changes the state of the circuit

- resistance, inductance or capacitance - an event is generated. Now the circuit analysis is performed at this time instant which need not be equal to tode but could be equal to tvolref, tvolcon or tpwm. In this manner, the circuit simulator ensures accurate timing of control functions and if necessary co-ordinates circuit analysis.

4 Conclusions

One of the primary objectives in developing the circuit simulator was to able to simulate complex power electronic circuits. A essential component of such circuits is nested control loops and a combination of software driven and hardware driven functions. It is not uncommon now to have multiple microcontrollers for a single power converter that perform separate functions and share information when needed. Or to use a high-speed microcontroller for advanced mathematical functions and a low-speed microcontroller for the simple hardware interface. In either case, to mimic this circuit, it is essential that the simulator be able to mimic the time events as they occur in hardware. This paper has described how the circuit simulator processes control functions at the exact time instant when desired ensuring accuracy and stability. This provides a user a powerful platform to test every aspect of a potential hardware implementation.