Z transform elements – Part 1:

About the writer: Harvey Morehouse is a contractor/consultant with many years of experience using circuit analysis programs. His primary activities are in Reliability, Safety, Testability and Circuit Analysis. He may be reached at harvey.annie@verizon.net. Simple questions regarding my articles for which I know the answer are free. Complex questions, especially where I am ignorant of the answers, are costly!!!

Summary:

This document will be in several parts, where this is part 1. Difference equations involve expressions that have meaning at discrete time intervals – sampled data systems. Generally this involves the use of A/D converters, digital processing functions, and D/A converters. It is also useful in describing switched capacitor systems.

To develop models that enable one to analyze such systems in a (SPICE) continuous time domain (transient analysis) is one goal, and also in the frequency (AC analysis) with one set of models is the goal. Here we develop one device model for the z^{-1} function.

Sampled Data Systems:

There are several different ways to solve difference (not differential) equations. These implementations often are combined in a system with analog circuitry. It would be useful to be able to able to accommodate both sampled data and continuous time circuitry analysis within one solution environment such as SPICE. And with the proper Z transform elements this is possible and feasible. Like the familiar Laplace 's' transform, the Z transform element can also be used within SPICE.

Z transform definition:

A sampled data system is usually described as an equation or set of equations in 'nT'. The transformation of this system into the Z domain is accomplished by means of the following equation.

$$V(z) = \sum_{n=0}^{\infty} V(nT)\, Z^{-n}$$

Interesting as this might be, to those who are interested, it really tells us nothing. What is needed is a means to relate the V(z) expressions into a form that SPICE can use. And there is. The 'z' variable can be shown to be equivalent to:

$$z = e^{sT}$$

Now the $\mathbf{e^{-sT}}$ function can be shown to be a delay in the time domain. The function delays the associated product of this function by time T. So we can associate z^{-1} with a time delay of T seconds. But what is the inverse of a time delay? What is the meaning of z? Now we get involved with the quasi-function devised by P.A.M. Dirac, who invented quantum mechanics, the so-called Dirac delta function. This function is defined as:

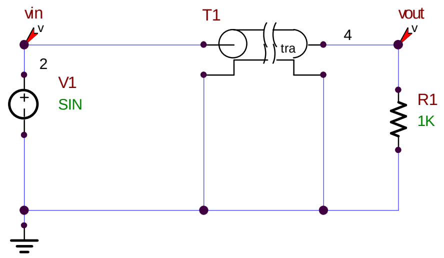
$$\delta(t-a) = 1/e^{as}$$

This function is equal to zero when $t \neq a$, one when $t = a$. It is of infinitesimally when it is at zero argument time and infinitely large in amplitude at time $t = a$, however its integral is equal to one. Remember now that we are dealing with sampled data systems whose values are only meaningful at the instant of sampling time. With that restriction, a simple delay represents the z^{-1} operator.

Now often sampled data systems are written in terms of powers of z or z^n, as well as in powers of z^{-1} or z^{-n}. Without attempting to describe what the inverse of a Dirac delta function looks like, we can note two things for now. The first is that one can always convert a system written in terms of z^n to one in terms of z^{-n} rather easily, And secondly there is a function that will allow a solution to the ratio of two polynomials in z^n, the Discrete Time Transfer Function.

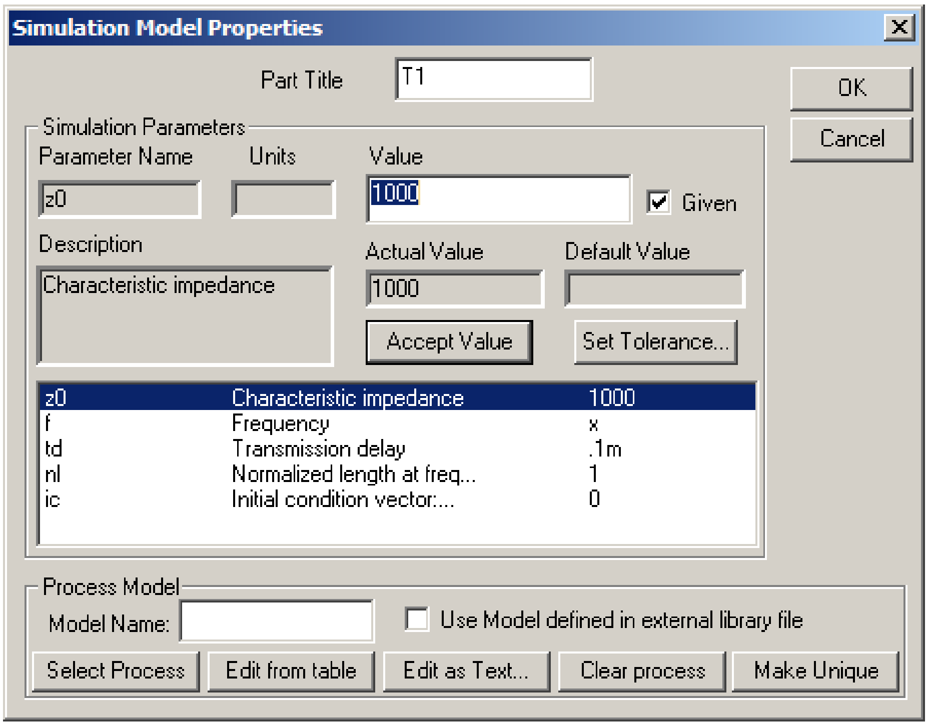
z^{-1} device modeling 1

The first device to be created is based on a lossless delay line implementation. The circuit to do this is shown in Figure 1 following.



Zm1 test circuit1
Figure 1

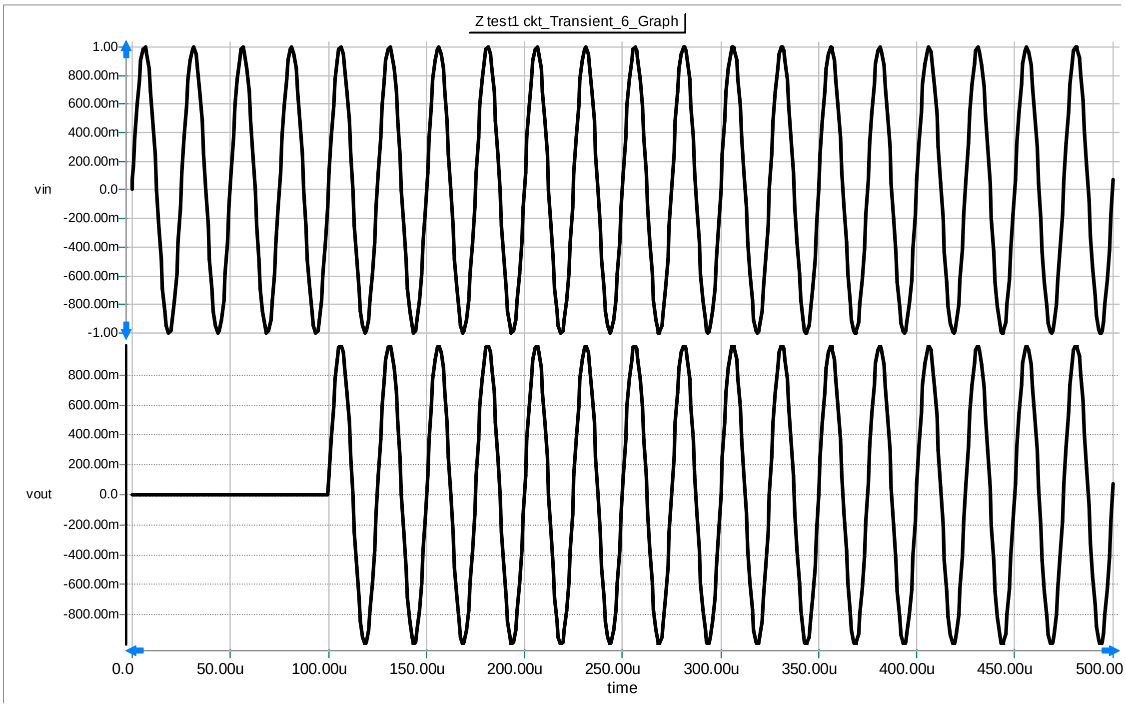
In Figure 1 a simple delay line and a buffer amplifier is used to create a $\mathbf{z^{-1}}$ device model. The parameters passed to the delay line are shown in Figure 2.



Circuit 1 passed parameters
Figure 2

The important parameters to be passed are those noted. (In order to create this part the value of frequency should be set to its default value of zero.) The transmission delay is set equal to the reciprocal of the sampling frequency. The value shown corresponds to a 10Kc sampling rate.

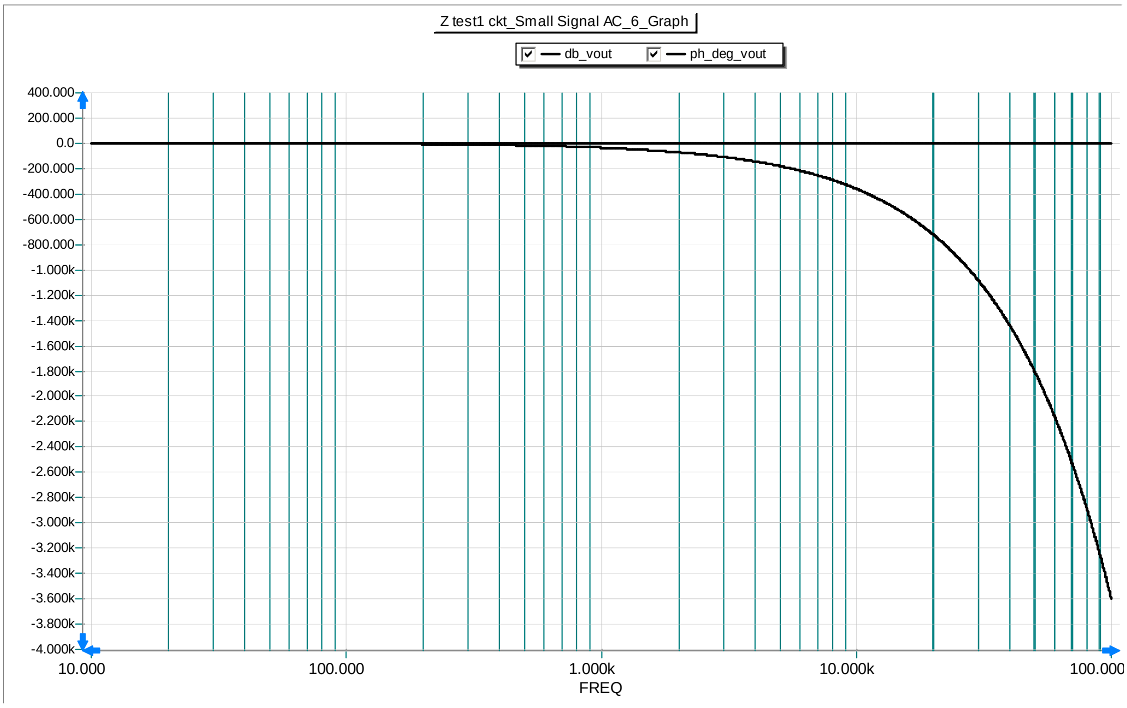
A transient graph of the circuit performance is shown in Figure 3.



Circuit1 zm1 circuit transient graph
Figure 3

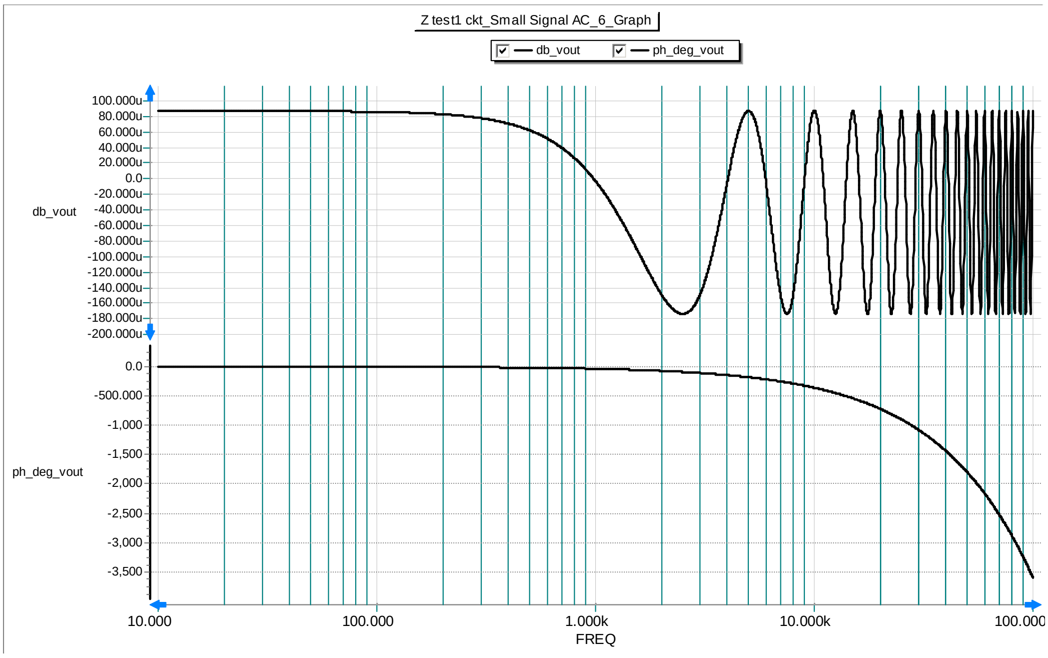
In Figure 3 it is seen that the circuit does indeed perform as intended. Now it is desired that the circuit model work in an AC analysis also. An AC frequency sweep was performed on the circuit. The results are shown in Figure 4.

Now it is desired to see how this will react to an AC sweep. A sweep was performed with v1 as the AC signal source. The results are shown in Figure 5.



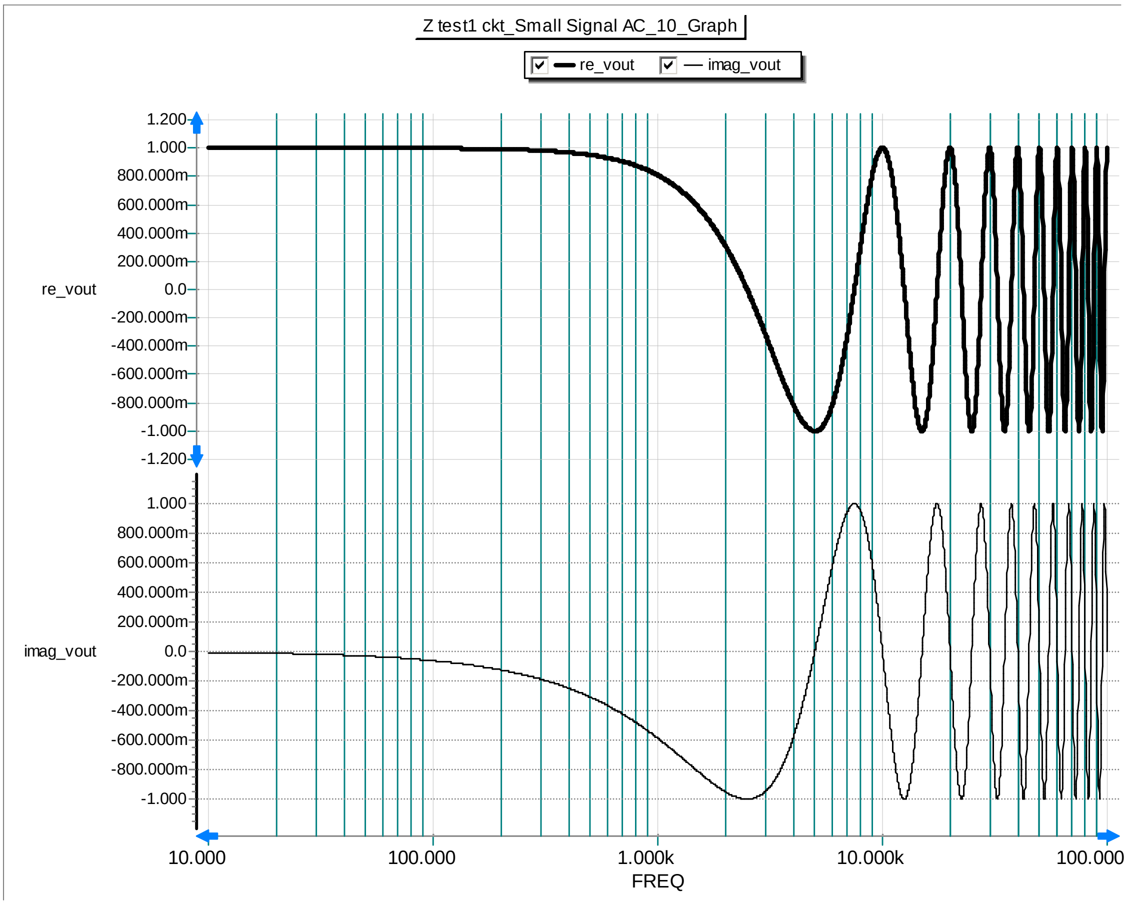
zm1 AC sweep
Figure 4

Here it is seen that the output gain is uniform over from zero to about zero to infinity frequency range. However, just out of curiosity, let us separate the two curves. This is shown in Figure 5 following.



zm1 AC sweep2
Figure 5

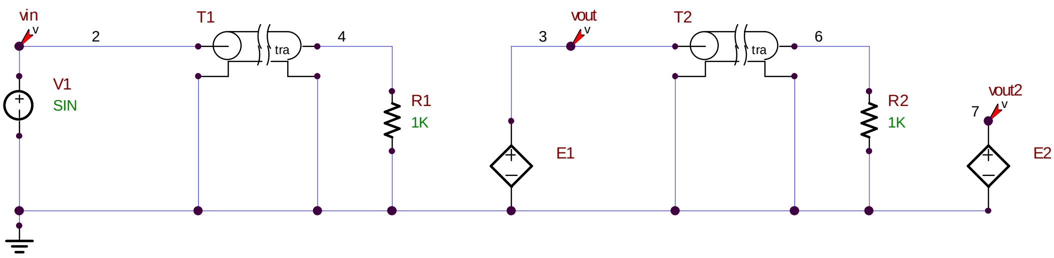
This is a most interesting set of curves! Until one notices that the db output is essentially constant at zero db. Now what we are looking at is the small signal gain at particular frequencies, it must be remembered. At all frequencies there is no effective attenuation, however with increasing frequency the delayed output is seeing more and more time delay per cycle, an effective phase shift. If there is a phase shift, there must be a gain change as the output signal real and complex parts are not perfectly in synchronism due to numerical errors and SPICE solution parameters that indicate an acceptable solution has been found. This is validated by a plot of the real and imaginary parts of vout, shown in Figure 6 following.



zm1 AC sweep3

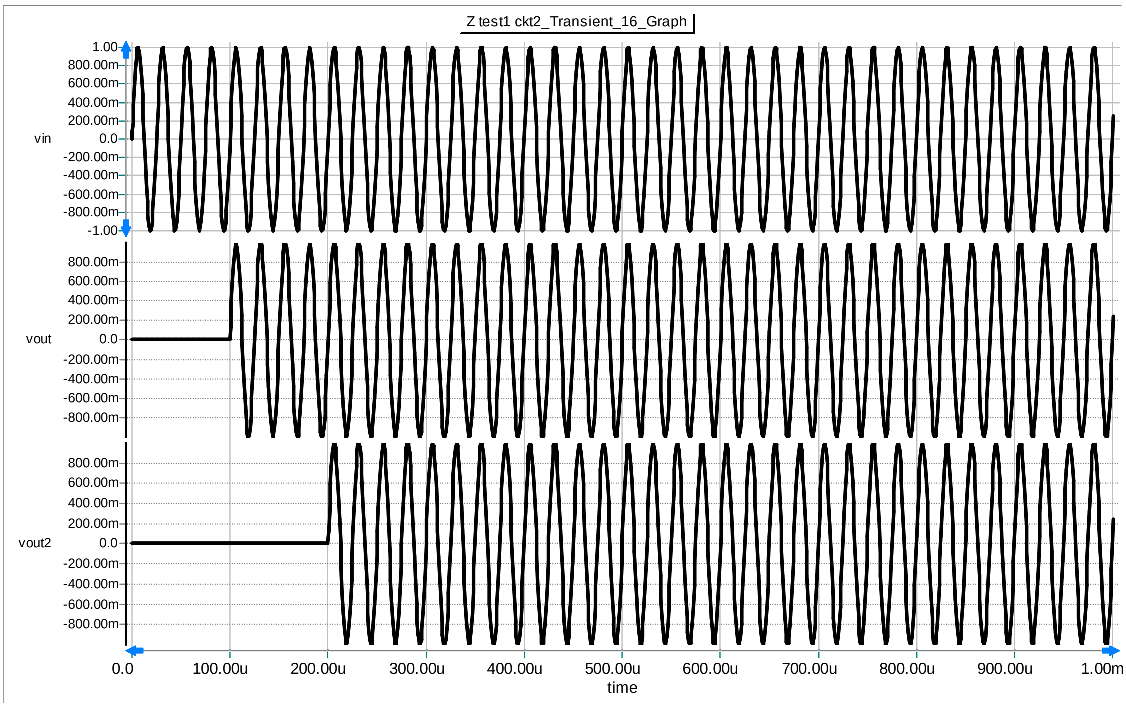
Figure 6

The next circuit we wish to examine is the effect of cascading these elements. Refer to Figure 7 following.



Cascade zm1 circuit 2
Figure 7

Figure 7 is straightforward. Devices E1 and E2 are unity gain buffers to eliminate circuit loading. The transient graph of this circuit is shown in Figure 8.

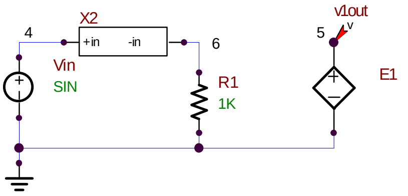


Cascade zm1 circuit 2 cascade connection
Figure 8

The circuits do not interact and the delays of each stage accumulate.
We are not done with this circuit yet, but first we will create another realization for the function.

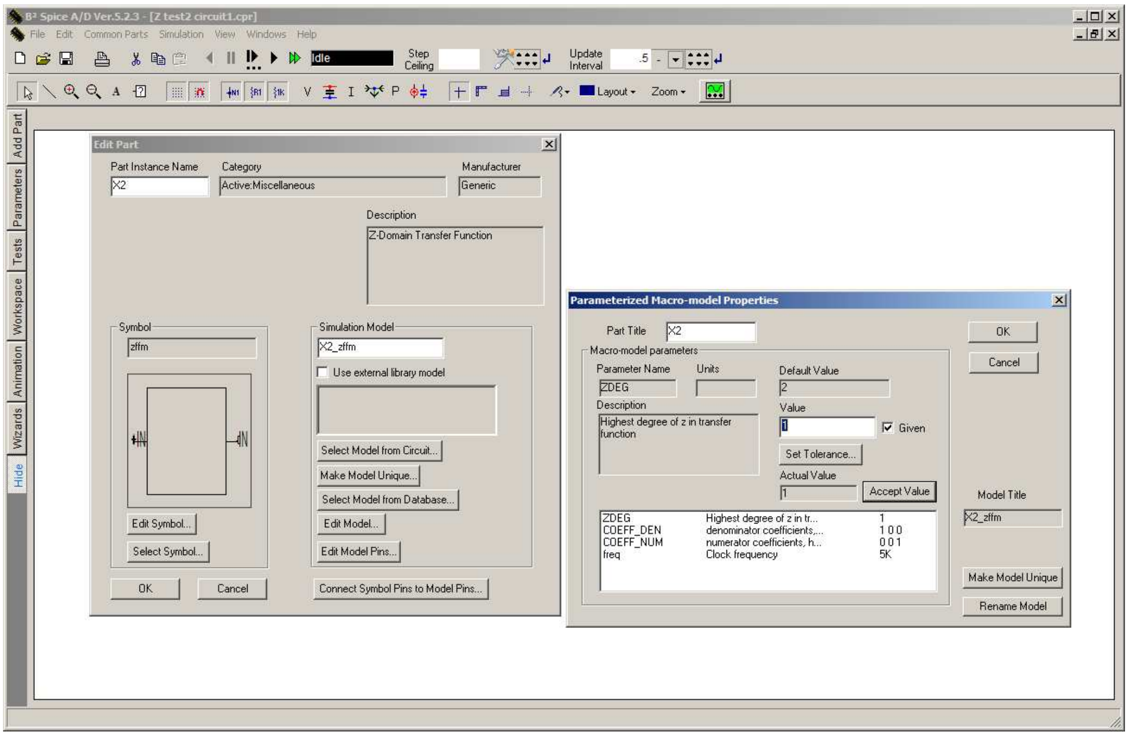
z^{-1} device modeling 2

B2Spice contains a model for a discrete time transfer function. This consists of simulating the transfer function of the ratio of two polynomials in z^n. This is a 'hard' coded function in C, and not directly accessible for modification. Could this function be used to create a z^{-n} as well as a z^n function? We will first consider making a z^{-n} function, as the result of the internal expression equaling (1)/(z) . Consider the circuit of Figure 9 following.



zm1 circuit 2 device model
Figure 9

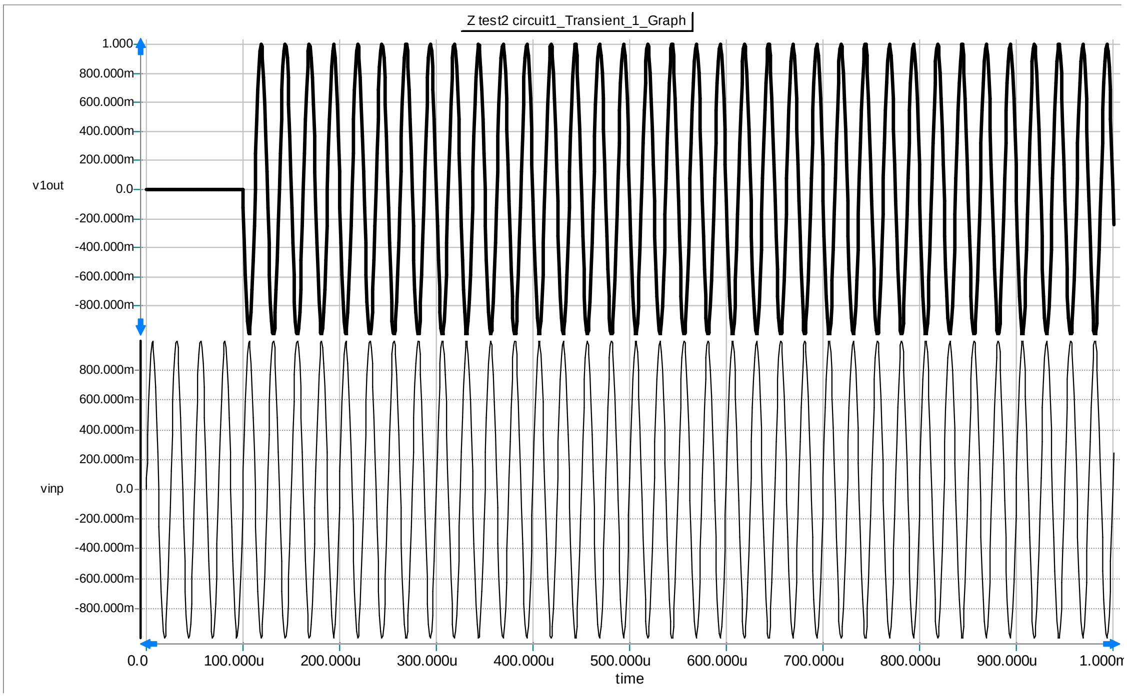
The circuit is straight forward, with the transmission line of the previous model replaced by X2, a discrete time transfer function model. Now this device is unusual, in that to get to the parameter settings, then after the Edit window opens, click the Edit model button. This is shown in Figure 10.



zm1 circuit 2 device model
Figure 10

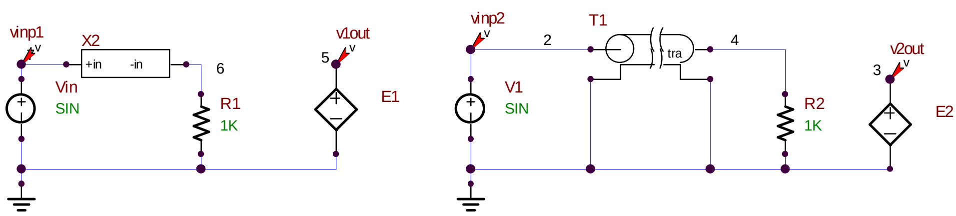
The X2 numerator is set to unity, (0 0 1) and the denominator to z, (1 0 0). The sampling frequency is set to 5K, or a period of 100 usec. Note that the creator of the circuit used half periods of delay for his realization. Thus, a denominator of (0 1 0) would correspond to a half delay of the input frequency period, or $e^{s*/2f}$.

A transient graph of this circuit is shown in Figure 11.



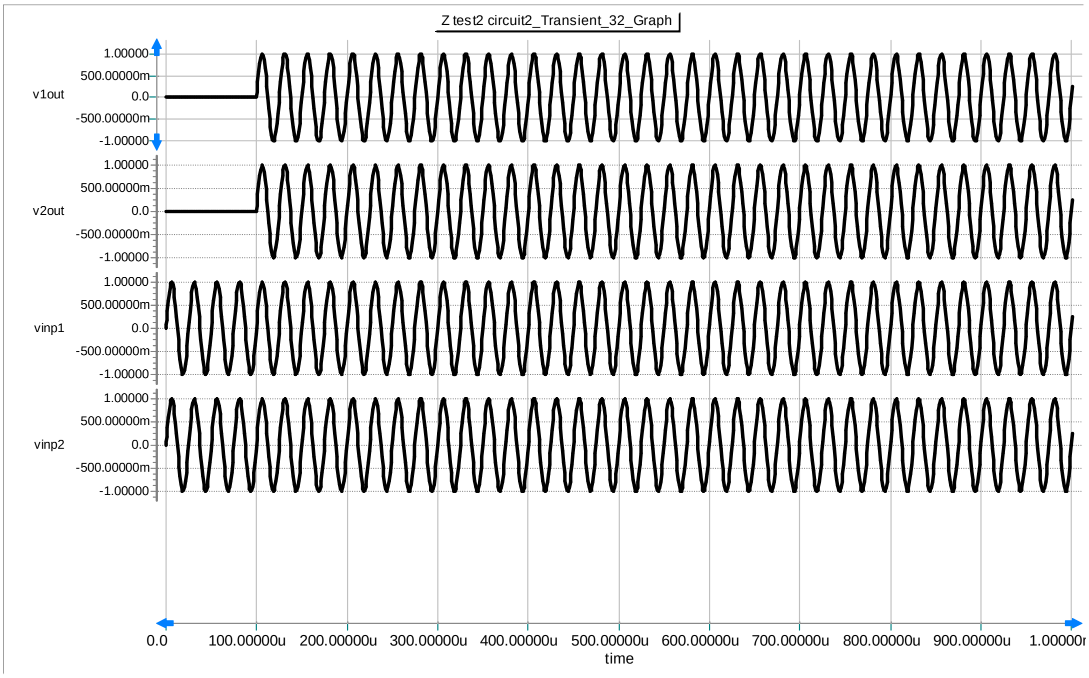
zm1 circuit 2 device transient test
Figure 11

The circuit behaves nicely. Now that we have two working models, they may be compared. This is done in the circuit of Figure 12.



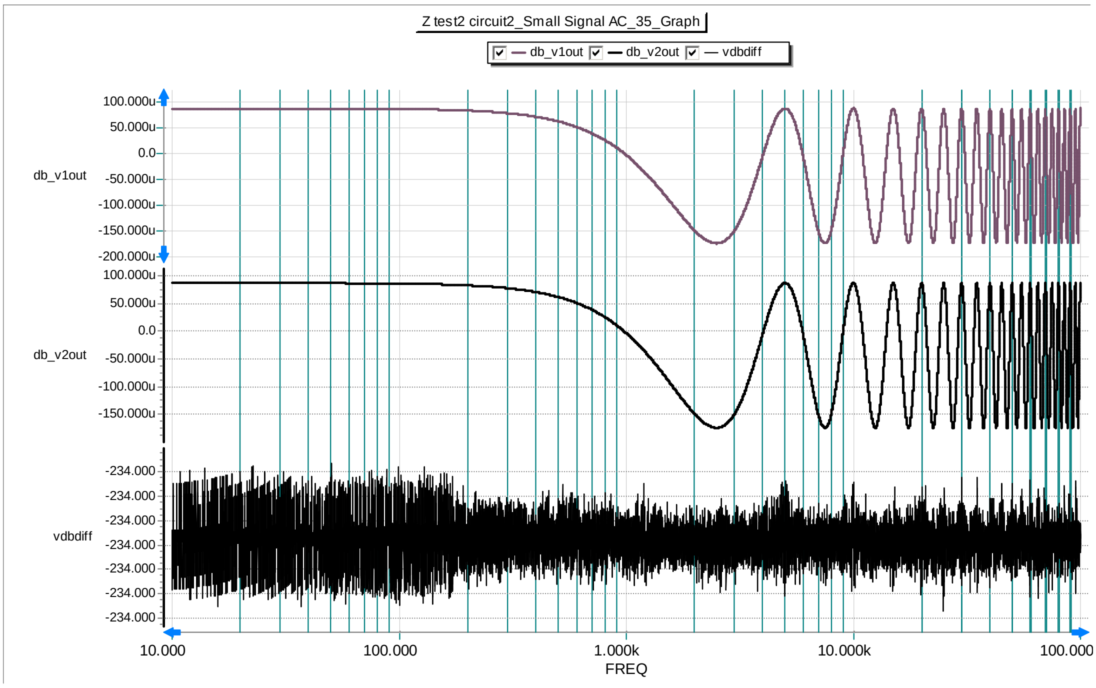
zm1 circuit 2 test2 comparison test
Figure 12

In Figure 12 the two delay circuits are brought together for a comparison test. The results of a transient test are shown in Figure 13.



zm1 circuit 2 test3 transient comparison test
Figure 13

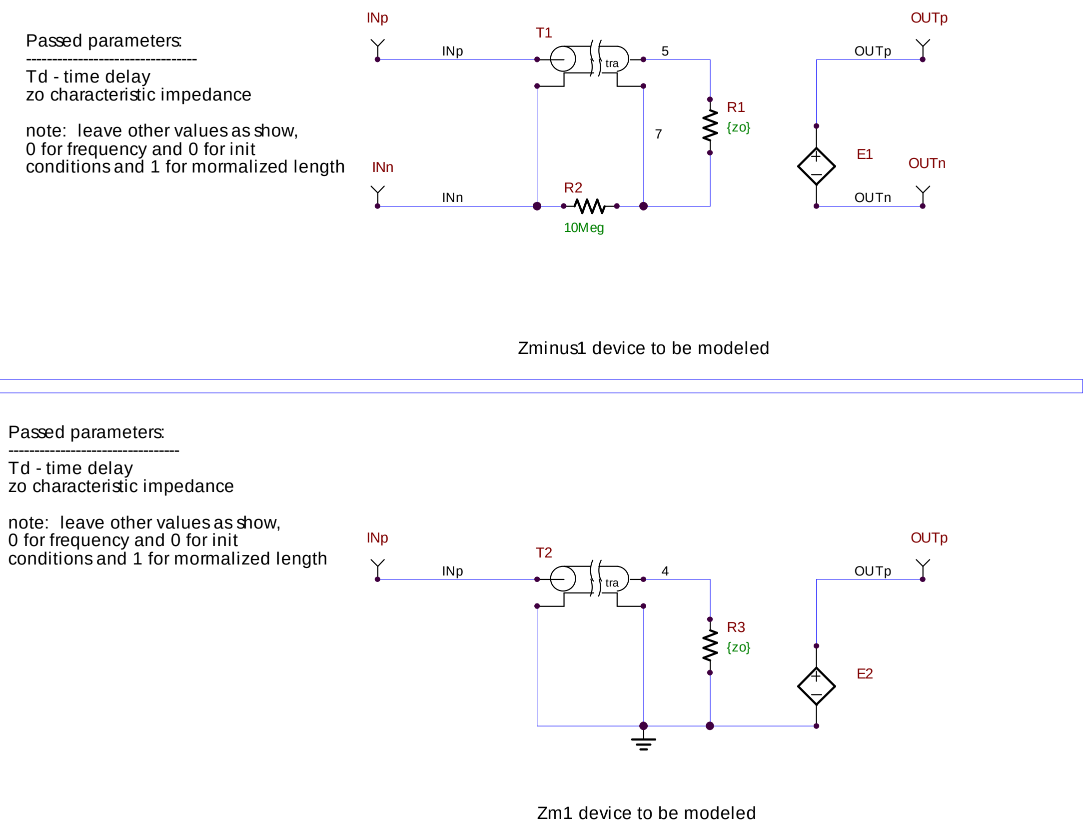
There is no apparent difference in the results. Figure 14 shows the results of an AC sweep comparison test.



zm1 circuit 2 test3 sweep comparison test
Figure 14

In Figure 14 it is seen that the difference between the two outputs is 234 dB less than the input, which is zero for all purposes.

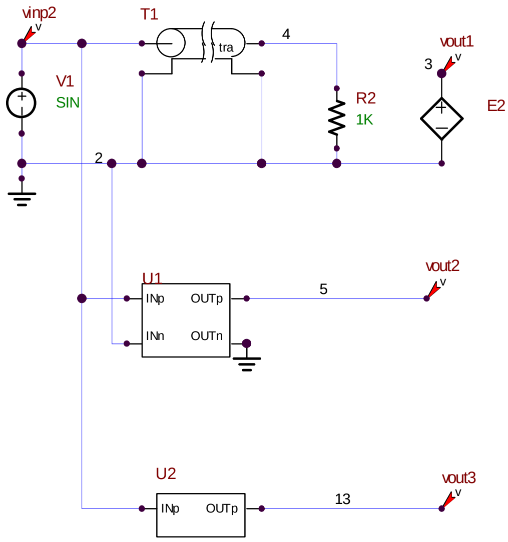
The question now remains which of the two implementations should be used to create the z^{-1} device? The answer is that the transmission line based model is simpler and more trouble free and so it will be used. We will create two versions of the circuit, with slightly different topologies. One is floating, and the other is grounded. These are shown in Figure 14 following.



z^{-1} device model circuits

Figure 14

The above circuits were turned into parameterized subcircuit devices and embedded in a circuit as shown in Figure 15.



z^{-1} device model circuit3
Figure 15

Passed parameters are z0 = 1K and Td = 0.1ms. (Actually, the value passed for z0 is meaningless, as the internal transmission lines are assumed perfectly terminated in the models.) The two devices perform almost identically, however there are very very slight differences due to the floating implementation used for U1.

The zmimus1 netlist (of device U1) is:

```
*************************
* B2 Spice Subcircuit
*************************
* Pin #              Pin Name
* INp                INp
* INn                INn
* OUTp               OUTp
* OUTn               OUTn
.Subckt Zminus1 INp INn OUTp OUTn

***** main circuit
R1 5  7 9.999999959040e+002
T1 INp  INn  5  7  z0 = 1k  td = 1.000000000000e-004  nl = 1  ic = 0
E1 OUTp  OUTn 5 7 1
R2 INn  7 10Meg

.ends
```

The zm1 netlist (of device U2) is:

```
*************************
* B2 Spice Subcircuit
*************************
* Pin #              Pin Name
* INp                INp
* OUTp               OUTp
.Subckt Zm1 INp OUTp
```

```
***** main circuit
T2 INp  0  4  0  z0 = 1k  f = 0  td = {Td}  nl = 1  ic = 0
R3 4  0 1K
E2 OUTp  0 4 0 1

.ends
```

Conclusions:

Two versions of a z transform delay element are presented, however the reader must by cautioned that in the sampled data domain the values have meaning only at the sample times.