

A FINITE-DIFFERENCE TIME-DOMAIN (FDTD) SOFTWARE FOR SIMULATION OF PRINTED CIRCUIT BOARD (PCB) ASSEMBLY

F. Kung and H. T. Chuah

Faculty of Engineering, Multimedia University
Jalan Multimedia, 63100 Cyberjaya, Selangor, Malaysia

Abstract—This paper describes the design of a three-dimensional (3D) finite-difference time-domain (FDTD) simulation software for printed circuit board (PCB) modeling. The flow, the dynamics and important algorithms of the FDTD simulation engine will be shown. The software is developed using object-oriented programming (OOP) approach, to enable code reuse and ease of upgrade in future. The paper begins by looking at how a 3D PCB structure is created using cubes, and proceed to show the inclusion of various lumped components such as resistors, capacitor, inductor and active semiconductor components into the model. The architecture of the FDTD simulation program is then carefully explained. Finally a few sample simulation examples using the software will be illustrated at the end of the paper.

1 Introduction

2 Building a 3D Model from Cubes

2.1 Incorporating Conducting Structures

2.2 Lumped Components

2.2.1 Linear Lumped Components

2.2.2 Nonlinear Lumped Components

3 The Issue of Stability and Absorbing Boundary Condition

4 The Model Definition File

5 The FDTD Simulation Engine — An Overview

5.1 Software Flow

5.2 Pre-Computation and Look-up Table

5.3 Other Important Objects in the Simulation Engine

5.4 The Complete Package

6 Sample Simulation Examples

7 Conclusion

References

1. INTRODUCTION

FDTD method has been widely used to model interaction of electromagnetic waves in complicated PCB structures [2–7]. The classical FDTD approach, which is based on Yee’s explicit formulation [1] is adopted for this paper. Mathematical theorems for the FDTD formulation, concerning issues such as accuracy, convergence, dispersion, computational complexity and stability are provided in [3, 8] and [9]. A 3D model can be considered as a combination of cubes, which are called Yee’s Cells in FDTD method [3]. The cubes can be variable in sizes but we will consider the size of all cubes to be similar in this paper. The discretization of 3D space and associated field components in a Yee’s Cell is shown in Fig. 1, following the convention of [7] and [10]. The electric and magnetic field components are updated in discrete time-steps, at an interval of $\frac{1}{2}\Delta t$, where Δt is the time-step. The update equations for the electric and magnetic fields usually fulfill the following forms, for non-magnetic dielectric materials (called the

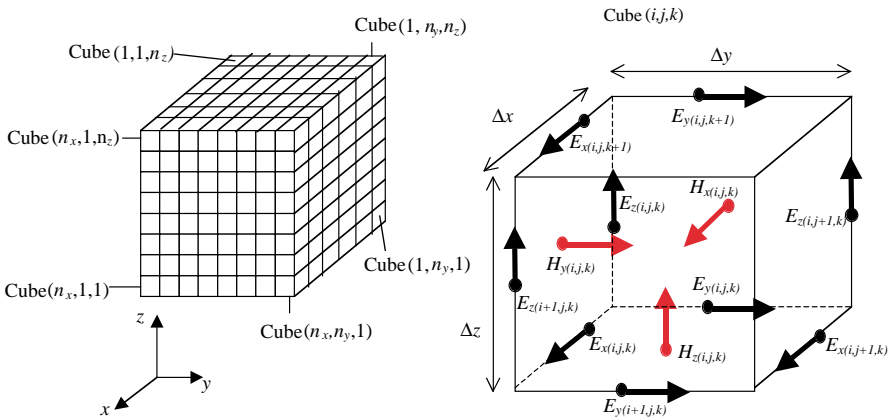


Figure 1. Discretization of the 3D space, and the standard Yee’s Cell (i, j, k) with associated field components.

Canonical FDTD Form [10]):

$$H_{r(i,j,k)}^{n+\frac{1}{2}} = H_{r(i,j,k)}^{n-\frac{1}{2}} - \frac{\Delta t}{\mu} \nabla \times E_{r(i,j,k)}^n \quad (1)$$

$$E_{r(i,j,k)}^{n+1} = E_{r(i,j,k)}^n + \frac{\Delta t}{\varepsilon_{r(i,j,k)}} \left[\nabla \times H_{r(i,j,k)}^{n+\frac{1}{2}} - J_{r(i,j,k)}^{n+\frac{1}{2}} \right] \quad (2)$$

where $r = x, y, z$. Here $\nabla \times H_{r(i,j,k)}^{n+\frac{1}{2}}$ and $\nabla \times E_{r(i,j,k)}^n$ are shorthand, for instance for x component these are [6, 7]:

$$\nabla \times E_{x(i,j,k)}^n = \frac{E_{z(i,j+1,k)}^n - E_{z(i,j,k)}^n}{\Delta y} - \frac{E_{y(i,j,k+1)}^n - E_{y(i,j,k)}^n}{\Delta z} \quad (3a)$$

$$\nabla \times H_{x(i,j,k)}^{n+\frac{1}{2}} = \frac{H_{z(i,j,k)}^{n+\frac{1}{2}} - H_{z(i,j-1,k)}^{n+\frac{1}{2}}}{\Delta y} - \frac{H_{y(i,j,k)}^{n+\frac{1}{2}} - H_{y(i,j,k-1)}^{n+\frac{1}{2}}}{\Delta z} \quad (3b)$$

Equation (1) and (2) are derived from Maxwell's Curl equations of E and H fields respectively by replacing the differential operators with finite difference. The term $\varepsilon_{r(i,j,k)}$ is the effective permittivity of the medium while $J_{r(i,j,k)}^{n+\frac{1}{2}}$ is the effective current density. Both terms can be dependent on location, orientation and previous field component values. The detailed derivation of the update equations for the field components will not be repeated here, but the reader is referred to the appropriate references [2–12].

The focus of this paper is on the designing of a versatile FDTD software for PCB modeling. This paper summarizes the works of [2–12], particularly [6, 7, 10] and shows in a systematic way how a full fledged FDTD software for 3D PCB modeling can be developed using object-oriented programming (OOP) language. Section 2 describes in detail how a 3D PCB model with complicated conducting structures can be modeled by using dielectric cubes with various conducting patches on the surface. A summary of incorporating lumped linear and nonlinear components is also presented. Issues of stability and absorbing boundary condition (ABC) are briefly discussed in Section 3. An efficient syntax to describe a 3D model is then shown in Section 4, followed by an overview of the architecture of the FDTD software in Section 5. Finally Section 6 shows some simulation examples to validate the work.

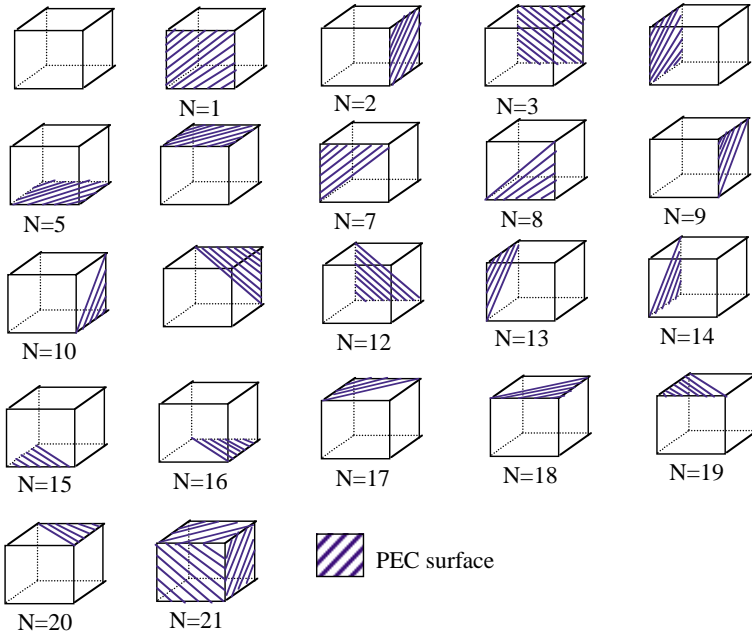


Figure 2. Perfect electric conductor (PEC) patches on cube surface with integer identification.

2. BUILDING A 3D MODEL FROM CUBES

Since a PCB assembly contains metallic structures, a range of cubes, with different conducting patches on the surface, is proposed, as illustrated in Fig. 2. Notice that in Fig. 2 an integer is assigned to each cube pattern, this allows the user to differentiate between the distinct cubes. Combining and stacking these cubes allows one to achieve a realistic approximate 3D model of a PCB assembly with various conducting structures. Since the conducting traces on a PCB is usually made of gold-plated copper-beryllium alloy with very large conductivity, it will be assumed as perfect electric conductor (PEC) here for simplicity. Moreover each cube can be assigned with some properties, which include the dielectric constant ϵ_r and the conductivity σ of the cube. These two parameters can be constant, or change with frequency and field intensity. This allows modeling of dielectric properties such as (1) linear, lossless, non-dispersive (2) linear, lossy, non-dispersive (3) linear, lossless and dispersive and (4) nonlinear. In a homogeneous, linear and lossless dielectric, the general electric field update equation of (2) would be simplified to:

$$E_r^{n+1}(i,j,k) = E_r^n(i,j,k) + \frac{\Delta t}{\epsilon_r} \left[\nabla \times H_r^{n+\frac{1}{2}}(i,j,k) \right] \quad (4)$$

If the dielectric is non-homogeneous, the permittivity ϵ_r in (4) will be the average permittivity of the 4 cubes surrounding the electric field component [9]. Linear dispersive dielectric can be accounted for by Recursive Convolution Method [11] (also described extensively in Chapter 9 of [3]) or the Z -transform method [12]. The Z -transform method can also handle nonlinear dielectric quite effectively. In both approaches the electric update equation will again assume the general form of (2), with the effective current density $J_r^{n+\frac{1}{2}}(i,j,k)$ due to the contribution from previous field components. $J_r^{n+\frac{1}{2}}(i,j,k)$ will be a nonlinear function of the field components for a nonlinear dielectric.

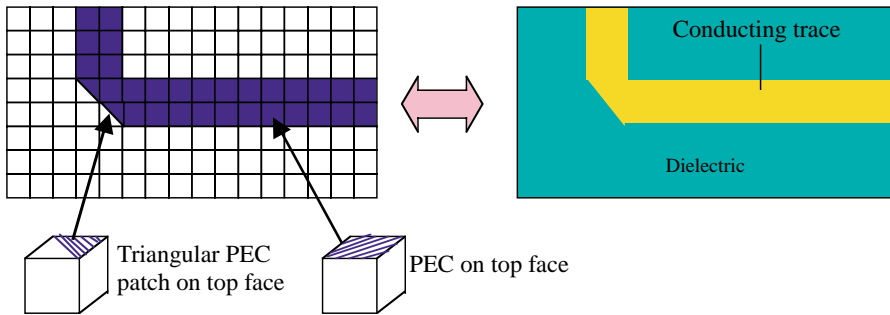


Figure 3. Top view of a section of a PCB surface and the equivalent FDTD model.

2.1. Incorporating Conducting Structures

Using the cubes of Fig. 2, a conducting trace with bend can easily be constructed as shown in Fig. 3. To model a PEC patch on the top face of a cube, all the corresponding E field components on the edges of the face will be permanently set to zero. To model a triangular patch on the surface of a cube, a method called ‘Diagonal split-cell model for PEC surface’ ([3], Chapter 10) is employed. The formulation is based on application of integral form of Faraday’s Law, the detail is discussed in [3] and [9]. Here only the result is given. Consider Fig. 4 where the top face of the cube is a PEC conducting patch, the update equation for $H_{z(i,j,k+1)}^{n+\frac{1}{2}}$ is:

$$H_{z(i,j,k+1)}^{n+\frac{1}{2}} = H_{z(i,j,k+1)}^{n-\frac{1}{2}} - \left(\frac{2\Delta t}{\mu\Delta y} E_x^n(i,j,k+1) + \frac{2\Delta t}{\mu\Delta x} E_y^n(i+1,j,k+1) \right) \quad (5)$$

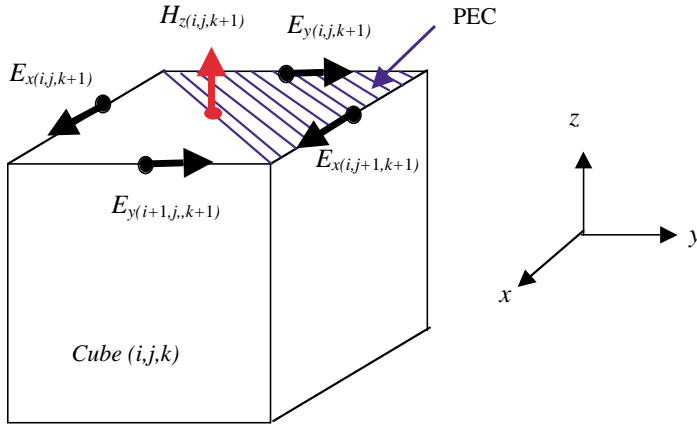


Figure 4. Top face of Cube (i, j, k) being partially covered with triangular PEC patch.

Equation (5) together with $E_{x(i,j+1,k+1)}^{n+1} = 0$ and $E_{y(i,j,k+1)}^{n+1} = 0$ account for the triangular PEC patch on the top surface of Cube (i, j, k) . The update equations for $E_{x(i,j,k+1)}^{n+1}$ and $E_{y(i+1,j,k+1)}^{n+1}$ remain unchanged. Fig. 5 shows the other orientations for triangular PEC patches and the corresponding update equations for H_z field component. This can be easily extended to triangular conducting patches in other surfaces.

2.2. Lumped Components

In FDTD method, a lumped component is assumed to coincide with an E field component. Associated with each lumped element is the current-voltage (I-V) relation. A potential difference imposed across the lumped element will result in a current flowing through the lumped element with negligible delay. Maxwell's Curl equation for H is then used to derive the update equation for the corresponding E field component. Update equation for H field components surrounding the lumped component is not affected.

2.2.1. Linear Lumped Components

Linear lumped components encompass elements such as short length of wire, resistor, capacitor, inductor and resistive voltage source. The I-V relation for the lumped component is linear. Here it is assumed the lumped element coincides with E field component $E_{z(i,j,k)}^n$,

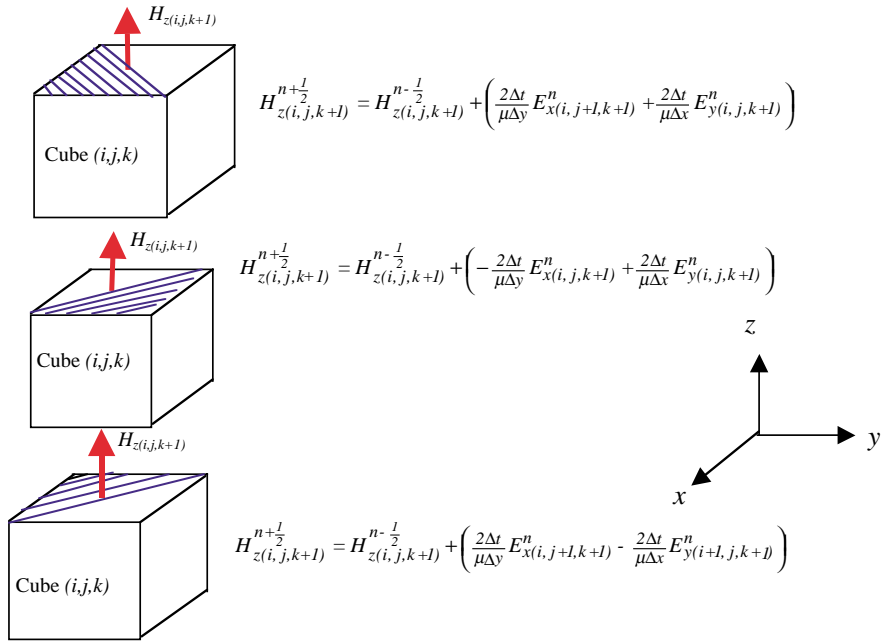


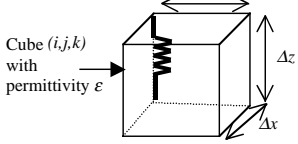
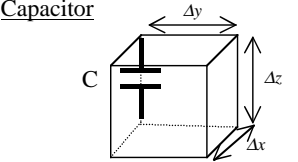
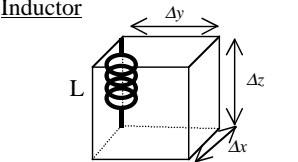
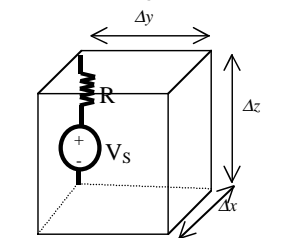
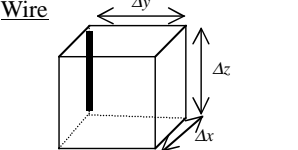
Figure 5. Update equations for H_z field component for various triangular PEC patches on top surface of Cube (i, j, k) .

generalization to other E field components can be easily carried out. Table 1 provides the update equations for the $E_{z(i,j,k)}^{n+1}$ field component when it is assumed that the above lumped components coincide with it. The derivation is described in [2]. Update equations for $H_{x(i,j,k)}^{n+\frac{1}{2}}$, $H_{x(i,j-1,k)}^{n+\frac{1}{2}}$, $H_{y(i,j,k)}^{n+\frac{1}{2}}$ and $H_{y(i-1,j,k)}^{n+\frac{1}{2}}$ remain unchanged.

2.2.2. Nonlinear Lumped Components

In this section, the technique of incorporating lumped diode and bipolar junction transistor (BJT) will be described. The I–V relations for diode and BJT are nonlinear. The discrete diode or BJT inclusive of the packaging may be larger than the cubes used to discretized the computational domain, but the actual size of the PN junction within the diode and BJT is usually smaller than the length of the cubes, thus can be considered as lumped. In this section, modeling of field effect transistor (FET) will not be covered as the method proposed for the diode and BJT can be directly extended to include the FET, see [5]

Table 1. Update equations for $E_z^{n+1}(i,j,k)$ of various linear lumped elements.

<p>Resistor</p> 	$E_z^{n+1}(i,j,k) = \left(\frac{1 - \frac{\Delta t \Delta z}{2Re\Delta x \Delta y}}{1 + \frac{\Delta t \Delta z}{2Re\Delta x \Delta y}} \right) E_z^n(i,j,k) + \left(\frac{\frac{\Delta t}{\epsilon}}{1 + \frac{\Delta t \Delta z}{2Re\Delta x \Delta y}} \right) \nabla \times H_z^{n+\frac{1}{2}}(i,j,k)$ <p>where $\nabla \times H_z^{n+\frac{1}{2}}(i,j,k) = \frac{H_y^{n+\frac{1}{2}}(i,j,k) - H_y^{n-\frac{1}{2}}(i-1,j,k)}{\Delta x} - \frac{H_x^{n+\frac{1}{2}}(i,j,k) - H_x^{n-\frac{1}{2}}(i,j-1,k)}{\Delta y}$</p>
<p>Capacitor</p> 	$E_z^{n+1}(i,j,k) = E_z^n(i,j,k) + \left(\frac{\frac{\Delta t}{\epsilon}}{1 + \frac{C\Delta z}{\epsilon\Delta x \Delta y}} \right) \nabla \times H_z^{n+\frac{1}{2}}(i,j,k)$
<p>Inductor</p> 	$E_z^{n+1}(i,j,k) = E_z^n(i,j,k) + \frac{\Delta t}{\epsilon} \nabla \times H_z^{n+\frac{1}{2}}(i,j,k) - \frac{\Delta z (\Delta t)^2}{\epsilon L \Delta x \Delta y} \sum_{m=1}^n E_z^m(i,j,k)$
<p>Resistive Voltage Source</p> 	$E_z^{n+1}(i,j,k) = \left(\frac{1 - \frac{\Delta t \Delta z}{2Re\Delta x \Delta y}}{1 + \frac{\Delta t \Delta z}{2Re\Delta x \Delta y}} \right) E_z^n(i,j,k) + \left(\frac{\frac{\Delta t}{\epsilon}}{1 + \frac{\Delta t \Delta z}{2Re\Delta x \Delta y}} \right) \nabla \times H_z^{n+\frac{1}{2}}(i,j,k)$ $- \left(\frac{\frac{\Delta t \Delta z}{Re\Delta x \Delta y}}{1 + \frac{\Delta t \Delta z}{2Re\Delta x \Delta y}} \right) \left(\frac{V_s^{n+\frac{1}{2}}}{\Delta z} \right)$ <p>V_s^n can be a constant to represent d.c. source, a sinusoidal function, a pulse function or any other arbitrary function of n.</p>
<p>Wire</p> 	$E_z^{n+1}(i,j,k) = 0$

for instance.

The diode model used here is similar to the model used in SPICE circuit simulator [13]. Here Taylor expansion is used to approximate the current-voltage (I–V) relationship across the PN junction locally. The Taylor expansion will result in a quadratic expression for the I–V relation, which changes dynamically as the operating point of the PN junction shifts during simulation. The advantage of this approach is simplicity in that the update equation for the electric field is explicit. Full details are presented in [6]. Unlike the linear components the PN junction is a polarized device, i.e., the current flow in the positive and negative direction is not the same. Assuming the lumped diode to be oriented along z -axis, it can be oriented in two modes along the z direction. We call these the positive and negative orientation, denoted by integer N . This is illustrated in Fig. 6. The positive orientation corresponds to forward bias current flowing in the positive z -axis and vice-versa for the negative orientation. The effective voltage across the PN junction is then given by:

$$V = N \cdot E_{z(i,j,k)} \Delta z \tag{6}$$

The bipolar junction transistor (BJT) is modeled as two back-to-back PN junctions. In this paper Gummel-Poon model is used instead as it is more accurate over the older Ebers-Moll BJT model [13]. Following the spirit of incorporating the diode into FDTD formulation, the I–V relations across the base-emitter (BE) and base-collector (BC) junctions of the BJT are approximated by two variables Taylor expansion. Ignoring higher order terms leads to two update equations for the electric fields across the BE and BC junctions. Full details are presented in [7]. Table 2A and Table 2B summarize the formulation for diode and BJT into the FDTD framework based on the methods described in [6] and [7]. This is shown for diode that is oriented along z -axis and BJT along x -axis, again orientation in other directions can easily be accommodated.

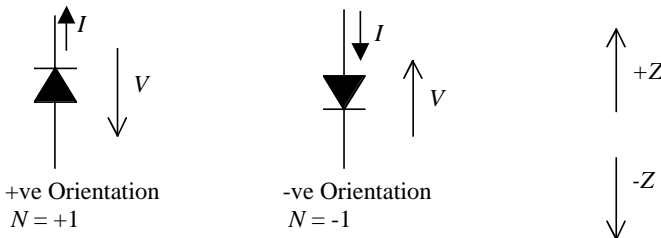


Figure 6. Positive and negative orientation of the diode.

Table 2A. Incorporating a diode into FDTD framework.

<p><u>Diode</u></p> <p>I_s = saturation current. n = emission coefficient. V_j = junction potential. C_{jo} = depletion capacitance at zero bias. τ_p = sum of transit time for holes and electrons. m = junction grading coefficient. FC = coefficient which determines when the junction is heavily forward biased. $F_2 = (1-FC)^{1+m}$ $F_3 = 1-FC(1+m)$ Please see [13] or [6] for detailed explanation of the model.</p>	<p style="text-align: center;">$I_D(V) = I_S \left(\exp\left(\frac{V}{nV_T}\right) - 1 \right), V_T = \frac{kT}{q}$</p> <p style="text-align: center;">$C(V) = \frac{\tau_D I_S}{nV_T} \left(\exp\left(\frac{V}{nV_T}\right) \right) + C_{jo} \left(1 - \frac{V}{V_j} \right)^{-m}, V < (FC \cdot V_j)$</p> <p style="text-align: center;">$\frac{\tau_D I_S}{nV_T} \left(\exp\left(\frac{V}{nV_T}\right) \right) + \frac{C_{jo}}{F_2} \left(F_3 + \frac{mV}{V_j} \right), V \geq (FC \cdot V_j)$</p>
$A = \frac{N\Delta t\Delta z^2}{2\epsilon\Delta x\Delta y} \left(\frac{1}{4} \frac{\partial^2 I_D}{\partial V^2} \Big _{V^n} + \frac{1}{\Delta t} \frac{\partial C}{\partial V} \Big _{V^n} \right) \quad B = \frac{\Delta t\Delta z}{\epsilon\Delta x\Delta y} \left(\frac{1}{2} \frac{\partial I_D}{\partial V} \Big _{V^n} + \frac{1}{\Delta t} C^n \right) + 1$ $D = \frac{N\Delta t\Delta y^n}{\epsilon\Delta x\Delta y} - \frac{\Delta t}{\epsilon} \nabla \times H_{z(i,j,k)}^{n+\frac{1}{2}}$ $\nabla \times H_{z(i,j,k)}^{n+\frac{1}{2}} = \frac{H_{y(i,j,k)}^{n+\frac{1}{2}} - H_{y(i-1,j,k)}^{n+\frac{1}{2}}}{\Delta x} - \frac{H_{x(i,j,k)}^{n+\frac{1}{2}} - H_{x(i,j,1,k)}^{n+\frac{1}{2}}}{\Delta y}$ $E_{z(i,j,k)}^{n+1} = E_{z(i,j,k)}^n + \frac{-B + \sqrt{B^2 - 4AD}}{2A} \quad \text{or} \quad E_{z(i,j,k)}^{n+1} = E_{z(i,j,k)}^n - \frac{D}{B} \quad (\text{lower accuracy})$ <p>The dielectric is assumed lossless and the permittivity can be set to a value reflecting the package of the diode.</p>	

To model packaged surface-mounted components, we have to take into account the physical size of the component. The lumped components must have physical dimension corresponding to the edges of the cube. This implies that lumped components as defined in FDTD framework have to be combined in series or parallel to commensurate with the physical dimension of the component. Fig. 7 and 8 illustrate how typical surface-mounted device (SMD) packages such as SOT23, SOT143 for BJT and resistor or capacitor in 0805 and 0603 packages [14] are modeled. Here the solder joint and solder pads are also taken into account. The spatial discretization used is $\Delta x = 0.8$ mm, $\Delta y = 0.8$ mm and $\Delta z = 0.52$ mm. Application to smaller packages such as 0402 and 0201 proceeds in similar manner by using smaller spatial discretization.

Table 2B. Incorporating a BJT into FDTD framework.

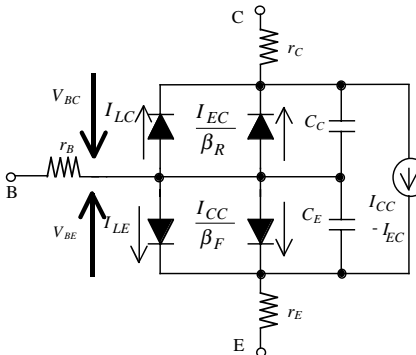
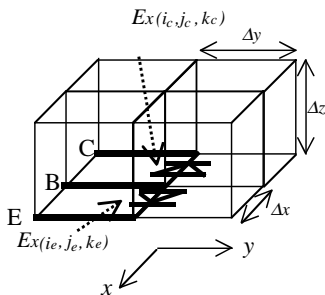
<p>Bipolar Junction Transistor β_F = Ideal maximum forward beta β_R = Ideal maximum reverse beta n_F = Forward current emission coefficient n_R = Reverse current emission coefficient V_A = Forward Early voltage I_{KF} = Corner for forward beta high-current roll-off I_{SE} = Base-emitter leakage saturation current n_E = Base-emitter leakage current emission coefficient V_B = Reverse Early voltage I_{KR} = Corner for reverse beta high-current roll-off I_{SC} = Base-collector leakage saturation current n_C = Base-collector leakage current emission coefficient V_{JE} = Base-emitter built-in potential m_E = Base-emitter P-N grading factor C_{JE} = Base-emitter zero-bias P-N capacitance τ_F = Ideal forward transit time V_{JC} = Base-collector built-in potential m_C = Base-collector P-N grading factor C_{JC} = Base-collector zero-bias P-N capacitance τ_R = Ideal reverse transit time FC = Forward bias depletion capacitance coefficient r_B, r_C, r_E = Base-spreading resistance, collector and emitter resistance.</p> <p>Please see [13] or [7] for detailed explanation of the model.</p>	 <p> $I_{CC} = \frac{I_{SS}}{q_b} \left(\exp\left(\frac{V_{BE}}{V_{TE}}\right) - 1 \right), V_{TE} = n_F \frac{kT}{q}$ $I_{EC} = \frac{I_{SS}}{q_b} \left(\exp\left(\frac{V_{BC}}{V_{TC}}\right) - 1 \right), V_{TC} = n_R \frac{kT}{q}$ $I_{LE} = I_{SE} \left(\exp\left(\frac{V_{BE}}{V_{TEL}}\right) - 1 \right), V_{TEL} = n_E \frac{kT}{q}$ $I_{LC} = I_{SC} \left(\exp\left(\frac{V_{BC}}{V_{TCL}}\right) - 1 \right), V_{TCL} = n_C \frac{kT}{q}$ $q_b = \frac{q_1}{2} + \sqrt{\left(\frac{q_1}{2}\right)^2 + q_2}, q_1 = 1 + \frac{V_{BE}}{V_E} + \frac{V_{BC}}{V_A}$ $q_2 = \frac{I_{SS}}{I_{KF}} \left(\exp\left(\frac{V_{BE}}{V_{TE}}\right) - 1 \right) + \frac{I_{SS}}{I_{KR}} \left(\exp\left(\frac{V_{BC}}{V_{TC}}\right) - 1 \right)$ $C_E(V_{BE}) = \tau_F \frac{\partial I_{CC}}{\partial V_{BE}} + C_{JE} \left(1 - \frac{V_{BE}}{V_{JE}}\right)^{-m_E}, V_{BE} < (FC \cdot V_{JE})$ $\tau_F \frac{\partial I_{CC}}{\partial V_{BE}} + \frac{C_{JE}}{F_{2E}} \left(F_{3E} + \frac{m_E V_{BE}}{V_{JE}}\right), V_{BE} \geq (FC \cdot V_{JE})$ $F_{2E} = (1 - FC)^{1+m_E}, F_{3E} = 1 - FC \cdot (1 + m_E)$ $C_C(V_{BC}) = \tau_R \frac{\partial I_{EC}}{\partial V_{BC}} + C_{JC} \left(1 - \frac{V_{BC}}{V_{JC}}\right)^{-m_C}, V_{BC} < (FC \cdot V_{JC})$ $\tau_R \frac{\partial I_{EC}}{\partial V_{BC}} + \frac{C_{JC}}{F_{2C}} \left(F_{3C} + \frac{m_C V_{BC}}{V_{JC}}\right), V_{BC} \geq (FC \cdot V_{JC})$ $F_{2C} = (1 - FC)^{1+m_C}, F_{3C} = 1 - FC \cdot (1 + m_C)$ </p>
---	---

Table 2B. continued.



$$\nabla \times \vec{H}_{C(i_c, j_c, k_c)}^{n+\frac{1}{2}} = \frac{H_{z(i_c, j_c, k_c)}^{n+\frac{1}{2}} \cdot H_{z(i_c, j_c, l, k_c)}^{n+\frac{1}{2}}}{\Delta y} - \frac{H_{y(i_c, j_c, k_c)}^{n+\frac{1}{2}} \cdot H_{y(i_c, j_c, k_c-1)}^{n+\frac{1}{2}}}{\Delta z}$$

$$\nabla \times \vec{H}_{E(i_e, j_e, k_e)}^{n+\frac{1}{2}} = \frac{H_{z(i_e, j_e, k_e)}^{n+\frac{1}{2}} \cdot H_{z(i_e, j_e, l, k_e)}^{n+\frac{1}{2}}}{\Delta y} - \frac{H_{y(i_e, j_e, k_e)}^{n+\frac{1}{2}} \cdot H_{y(i_e, j_e, k_e-1)}^{n+\frac{1}{2}}}{\Delta z}$$

$$I_{ES} = I_{LE} + \left(1 + \frac{1}{\beta_F}\right) I_{CC} - I_{EC} \quad I_{CS} = I_{LC} + \left(1 + \frac{1}{\beta_R}\right) I_{EC} - I_{CC}$$

$$B_1 = \frac{\Delta t \Delta x}{\varepsilon \Delta y \Delta z} \left(\frac{1}{2} \frac{\partial I_{CS}}{\partial V_{BC}} \Big|_{V_{BC}^n} + \frac{C_C^n}{\Delta t} \right) + 1 \quad C_1 = \frac{\Delta t N_C}{\varepsilon \Delta y \Delta z} I_{CS}^n - \frac{\Delta t}{\varepsilon} \left(\nabla \times \vec{H}_{C(i_c, j_c, k_c)}^{n+\frac{1}{2}} \right)$$

$$E_1 = \frac{\Delta t \Delta x N_C N_E}{2 \varepsilon \Delta y \Delta z} \frac{\partial I_{CS}}{\partial V_{BE}} \Big|_{V_{BE}^n} \quad B_2 = \frac{\Delta t \Delta x N_C N_E}{2 \varepsilon \Delta y \Delta z} \cdot \frac{\partial I_{ES}}{\partial V_{BC}} \Big|_{V_{BC}^n}$$

$$C_2 = \frac{\Delta t N_E}{\varepsilon \Delta y \Delta z} I_{ES}^n - \frac{\Delta t}{\varepsilon} \left(\nabla \times \vec{H}_{E(i_e, j_e, k_e)}^{n+\frac{1}{2}} \right) \quad E_2 = \frac{\Delta t \Delta x}{\varepsilon \Delta y \Delta z} \left(\frac{1}{2} \frac{\partial I_{ES}}{\partial V_{BE}} \Big|_{V_{BE}^n} + \frac{C_E^n}{\Delta t} \right) + 1$$

$$E_{x(i_c, j_c, k_c)}^{n+1} = E_{x(i_c, j_c, k_c)}^n + \frac{C_1 E_2 - C_2 E_1}{B_2 E_1 - B_1 E_2} \quad E_{x(i_e, j_e, k_e)}^{n+1} = E_{x(i_e, j_e, k_e)}^n + \frac{B_1 C_2 - B_2 C_1}{B_2 E_1 - B_1 E_2}$$

*For the example illustrated in the figure above, $N_C = -1$ and $N_E = 1$.

3. THE ISSUE OF STABILITY AND ABSORBING BOUNDARY CONDITION

The proposed scheme of incorporating nonlinear diode and BJT into FDTD simulation is efficient. Since it is a one-step method and explicit, the scheme can be easily integrated into an FDTD program. The issue of using too large a time step, rendering the result inaccurate does not occur. Since in requiring the space and time discretization to fulfill acceptable dispersion and CFL Stability Criterion [3, 8, 9], the time step is usually limited to one tenth of the period of highest harmonics encountered. This itself is sufficient to guarantee adequate accuracy. Furthermore, in [10], a new general stability criterion has been developed which can accommodate nonhomogeneous dielectric and lumped components (both linear and nonlinear). It is shown in [10] if the lumped components formulation does not contribute ‘numerical energy’ to the system then the FDTD formulation is stable

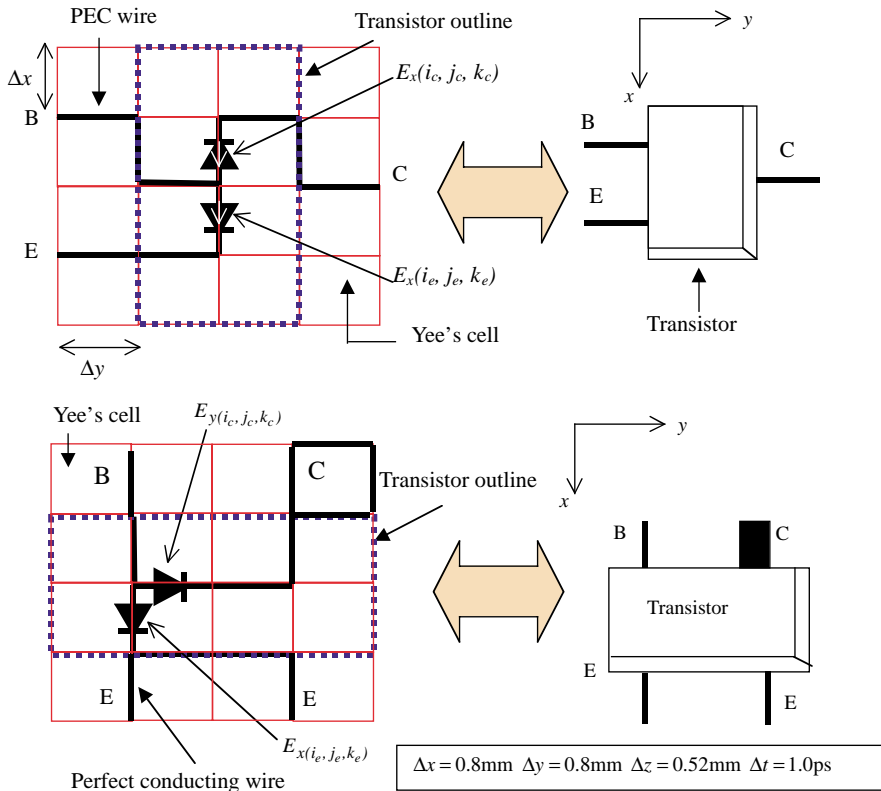


Figure 7. Representing an NPN transistor in SOT23 and SOT143 packages in FDTD framework.

(a condition known as proper formulation). For the case of PCB assembly, the space and time discretization fulfills almost similar condition as the CFL Stability Criterion. It is illustrated in [10] that all the lumped component formulations in Section 2 are proper or conditionally proper, thus the FDTD system so developed will be stable.

In this paper the Mur's first order absorbing boundary condition (ABC) ([3]) is used for the following reasons:

1. Mur's first order ABC can be applied to boundary with dielectric discontinuities. For boundary with dielectric discontinuity it is discovered that the application of Mur's second order ABC will cause large spurious wave component to be generated in the computational domain, which defeat the purpose of having an

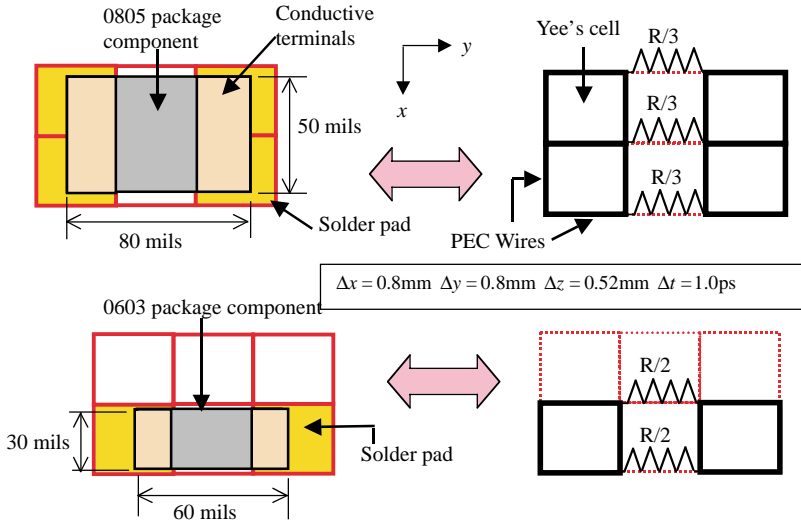


Figure 8. Representing a resistor R in 0805 and 0603 packages in FDTD framework.

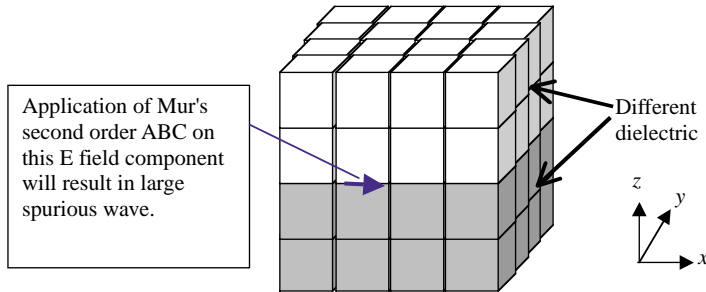


Figure 9. Boundary E field component between two layers of different dielectrics.

ABC with better absorbing property. The reason for this is electromagnetic wave travels at different velocity for dielectric with different permittivity. Mur's second order ABC ([3]) assumes all associated E field components to have similar permittivity and velocity c , which is not true when the boundary E field component is sandwiched between two different dielectrics. This is shown in Fig. 9 for $y = 0$ boundary with a discontinuous dielectric along z -axis. This results in substantial error during the interpolation process, which then propagates back to the computational domain.

2. Computationally Mur's first order ABC is much easier to implement as compare to other ABCs.
3. For many PCB modeling problems, the spurious reflection from Mur's first order ABC is still acceptable as long as sufficient buffer cubes are allocated.

4. THE MODEL DEFINITION FILE

With the method to model a 3D PCB assembly in place, a syntax is needed to describe the 3D model. Here we propose a syntax called the Model Definition File (MDF). The MDF is a text file, consisting of lines of text which describe a single cube, or a collection of similar cubes, until all cubes are taken into account in the model. Each line starts with one of the following keywords: "cube", "region", "comp", "probe", "param" and "*". A brief description of the keywords is provided in Table 3.

A simple MDF file listing is shown in Table 4. The MDF content of Table 4 describes the physical model illustrated in Fig. 10. The corresponding electrical schematic is shown in Fig. 11. Fig. 10 shows

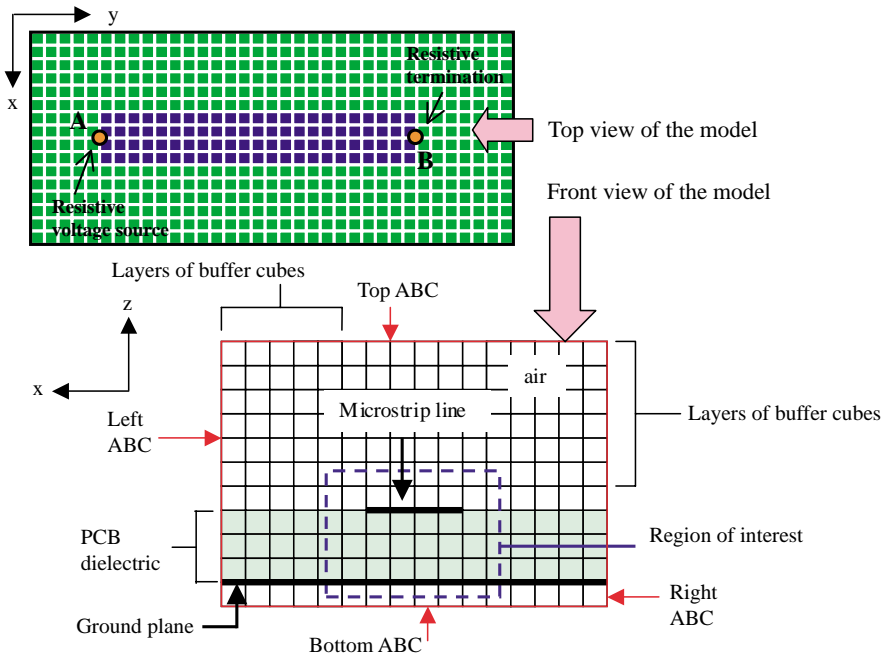


Figure 10. Top view and front view of a simple model.

Table 3. The model definition file keywords.

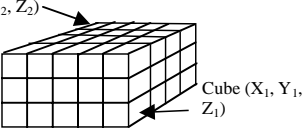
<p>cube X Y Z Er Sigma D N [...parameter list..] Purpose: Specifies the characteristics of a single cube. Example: cube 5 .8 9 4.2 0 0 0 cube 4 5 10 2.7 100.0 1 1</p>	<p>X, Y, Z = Coordinate of the cube. Er = The dielectric constant of the cube. Sigma = The conductivity of the cube. D = Cube characteristic i.e. whether it is a lossless dielectric (0), lossy dielectric (1), linearly dispersive (2) etc. N = Cube surface characteristic, corresponding to the integer N in Fig. 2. [parameter list...] = Optional parameters, for used with dispersive and nonlinear dielectric.</p>
<p>region X1 Y1 Z1 X2 Y2 Z2 Er Sigma D N [...parameter list..] Purpose: Specifies the characteristics of a group of cube bounded by 2 end cubes. Example: region 2 3 4 10 15 16 2.7 0 0 1</p>	<p>(X1, Y1, Z1) and (X2, Y2, Z2) denote the coordinates of the two end cubes in a region, as shown below. The rest of the parameters are similar to keyword "cube".</p> 
<p>probe X Y Z Field Purpose: Specifies the field components to print into the output file (in binary format). Example: probe 5 8 7 ez</p>	<p>X, Y, Z = Coordinate of the field component, see Fig. 1. Field = ex, ey, ez, hx, hy, hz for x, y or z components of electric and magnetic fields.</p>
<p>comp X Y Z Field Type Ref [...parameter list..] Purpose: Specifies a lumped component in the FDTD spatial grid. Example: comp 7 45 5 ey 150 0 1 (2.682E-09 1.836 0.5 0.333 4.0E-12 11.54E-09 0.5) comp 8 45 5 ey 150 0 1 (2.682E-09 1.836 0.5 0.333 4.0E-12 11.54E-09 0.5) Or comp 7 45 5 ey 150 10 comp 8 45 5 ey 150 10 param 10 (1 2.682E-09 1.836 0.5 0.333 4.0E-12 11.54E-09 0.5)</p>	<p>X, Y, Z = Coordinate of the field component, see Fig. 1. Field = ex, ey, ez for x, y or z components of electric field. Type = The type of components. For instance 100 represents a resistor, 101 represents a capacitor and 200 represents an inductor. Please refer to [9] for complete listing. [...parameter list...] = a list of optional parameters. Ref = The parameter list reference. The user has the choice of adding the component parameter list immediately after the component declaration, or having the parameter list declared elsewhere in the MDF file.</p>
<p>param Ref [... parameter list..] Purpose: Specifies a parameter list. This is useful when many components or cubes share similar parameter list.</p>	<p>Ref = An integer to identify the parameter list. Parameter list = A sequence of real values such as 2.978, 11.54E-09 and 0.5 separated by space or tab.</p>
<p>* Comments</p>	<p>Anything after * is treated as comments.</p>

Table 4. Model definition file for the model shown in Fig. 10.

```

16 35 11
* ### Begins model definition ###
* --- PCB dielectric definition (linear and lossless dielectric) ---
region 1 1 3 16 35 4 4.2 0 0 0
* --- PCB ground plane ---
region 1 1 2 16 35 2 4.2 0 0 5
* --- Microstrip line ---
region 7 6 4 10 28 4 4.2 0 0 6
* --- Load termination (Type 100 is a resistor, Type 1 is a PEC wire) ---
comp 9 29 4 ez 100 0 50
comp 9 29 3 ez 1 0
comp 9 29 2 ez 1 0
* --- Source definition (Type 50 is a trapezoidal pulse resistive voltage source) ---
* pulsed voltage source: This corresponds to 50Ohm source resistance, Vlow=0,
* Vhigh=3V , Tdelay=0, Trise/fall=50ps, Thigh=100ps and Tlow=200ps
comp 9 6 4 ez 50 0 (50 0 3 0 50 100 200)
comp 9 6 3 ez 1 0
comp 9 6 2 ez 1 0
* --- definition of field components to probe ---
probe 9 6 4 ez
probe 9 29 4 ez
* ### End of File ###

```

a short length of microstrip line, on a PCB with linear, lossless and non-dispersive dielectric with $\epsilon_r = 4.2$. The thickness of the PCB is equivalent to three cubes. On one end of the microstrip line there is a resistive voltage source with $R_s = 50 \Omega$. The other end of the microstrip line is terminated with a 50Ω lumped resistor. The detailed connection of the resistive voltage source and lumped resistive termination is shown in Fig. 12. Note that in Fig. 10 there should be a few layers of buffer cubes between the region of interest and the boundaries of the model as mentioned (which uses Mur's first order ABC). No buffer cubes are required for the ground plane as it covers the whole of the bottom face of the model boundary.

The first line in Table 4 declares that the size of this model is 16 cubes along x -axis ($x_n = 16$), 35 cubes along y -axis ($y_n = 35$) and 11 cubes along z -axis ($z_n = 11$). The size of each cube ($\Delta x, \Delta y, \Delta z$) is not declared in the MDF file, but will be defined in the FDTD software together with Δt . Typically for a FR4 double-sided PCB, the suggested size of each cube is:

$$\Delta x = 0.8 \text{ mm}, \quad \Delta y = 0.75 \text{ mm}, \quad \Delta z = 0.52 \text{ mm}$$

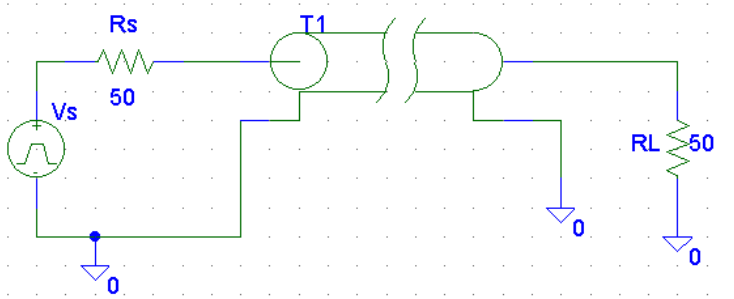


Figure 11. The electrical schematic of the model of Figure 10.

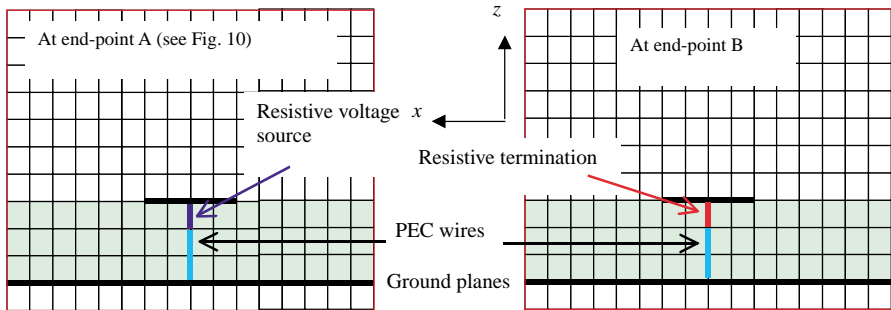


Figure 12. End-points of the microstrip line.

Every line beginning with the keyword “*” is a comment and will be ignored.

5. THE FDTD SIMULATION ENGINE — AN OVERVIEW

The basic simulation engine is divided into two modules: the interface to the WindowsTM operating system and the actual simulation engine. This is shown in Fig. 13. In this paper only an overview of the program will be given. Emphasis is on the concepts of organizing the data in the program systematically so that many different conditions can be modeled.

The WindowsTM Interface Block is coded in C Language, as there are many standard templates for creating a WindowsTM based program in C. The FDTD Simulation Engine is coded in C++ Language

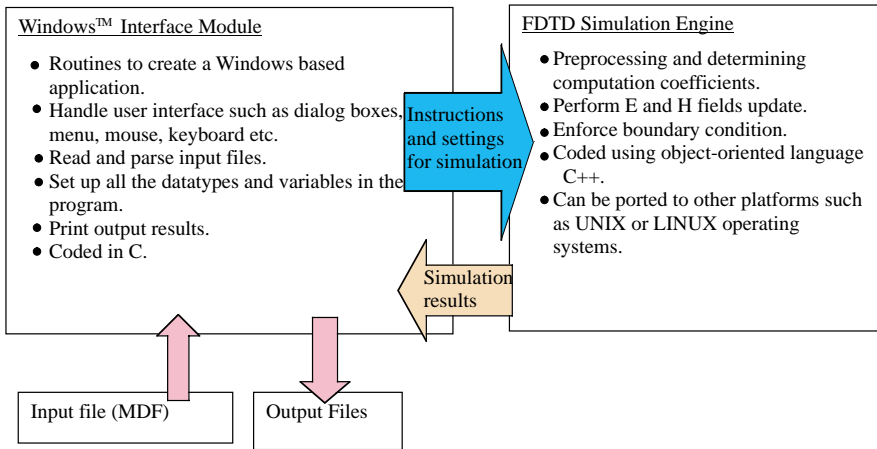


Figure 13. Major software blocks for the FDTD simulation program.

with future upgrade and expansion in mind. Using object-oriented programming language such as C++ allows efficient code reuse and project management [15]. Standard C++ syntax's complying with ANSI (American National Standard Institute) are used, thus ensuring that the FDTD Simulation Engine can be compiled and run in other platforms such as UNIX or LINUX. The output of the software contains two files, an output file (data.out) containing the electric field or magnetic field components and a header file (data.hdr) containing a listing of the field components plotted out in the output file. The 'data.out' file is stored in binary format as this saves disk space while the 'data.hdr' file is stored in text format. Thus visualizing software is required to view the output file. Software such as MATLAB or MATHCAD can be employed in visualizing the output file. Due to space constraint, the details of the output file will not be discussed, but full description can be obtained from the software online help [16].

5.1. Software Flow

Fig. 14 shows the high-level flow of the FDTD Simulation Engine module. The flow is typical of a FDTD algorithm except for the initialization part. There are more steps in the initialization portion as compare to standard FDTD flow as the FDTD simulation engine needs to accommodate different types of elements with various update functions. Furthermore frequently used values are pre-computed and stored as computational coefficients. A system is therefore needed

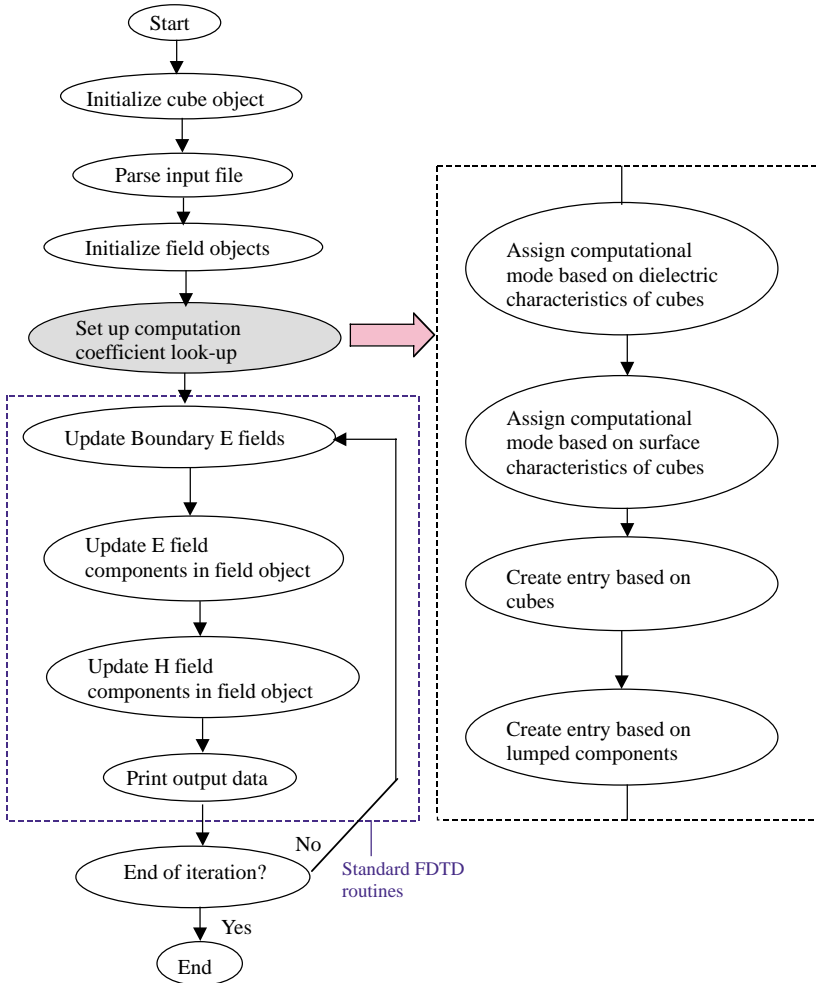


Figure 14. Computation flow of the FDTD program.

to track the location of each coefficient in the computer memory. The following sections will attempt to provide in more details the key components of the software flow. Some familiarity with C/C++ language syntax and datatype are assumed.

5.2. Pre-Computation and Look-up Table

In order to hasten the update of all electric and magnetic field components, regularly used expressions are pre-computed and stored

in a look-up table. Each look-up table can be implemented as a two dimensional array of floating point values. The row index to the array corresponds to a certain update mode. An example will help to clarify the idea. The update equation for E_x field component, in an isotropic region with linear, lossless dielectric is repeated here ([3]):

$$E_x^{n+1}(i,j,k) = E_x^n(i,j,k) + \frac{\Delta t}{\varepsilon \Delta y} \left[H_z^{n+\frac{1}{2}}(i,j,k) - H_z^{n+\frac{1}{2}}(i,j-1,k) \right] - \frac{\Delta t}{\varepsilon \Delta z} \left[H_y^{n+\frac{1}{2}}(i,j,k) - H_y^{n+\frac{1}{2}}(i,j,k-1) \right] \quad (7a)$$

This can be written as:

$$E_x^{n+1}(i,j,k) = E_x^n(i,j,k) + C_{0X} \left[H_z^{n+\frac{1}{2}}(i,j,k) - H_z^{n+\frac{1}{2}}(i,j-1,k) \right] - C_{1X} \left[H_y^{n+\frac{1}{2}}(i,j,k) - H_y^{n+\frac{1}{2}}(i,j,k-1) \right] \quad (7b)$$

where

$$C_{0X} = \frac{\Delta t}{\varepsilon \Delta y}, \quad C_{1X} = \frac{\Delta t}{\varepsilon \Delta z} \quad (8)$$

The computation coefficients C_{0X} and C_{1X} are frequently used as other E field components in similar dielectric will use similar update function too. These coefficients are stored in a two-dimensional array and assigned a certain row index, for example row r . Hence all E_x computations that share similar update form as Equation (7a) and similar dielectric constant ε will use the computation coefficients C_{0X} and C_{1X} stored in row r of the array. The computation coefficients will be different for E_x in region of different dielectric constant ε . Another row entry will have to be created for that particular region. Sometimes the update equation form also varies, for instance when E_x component coincide with a lumped resistor of resistance R Ohms, the update equation is given by (Table 1):

$$E_x^{n+1}(i,j,k) = \left(\frac{1 - C_X}{1 + C_X} \right) E_x^n(i,j,k) + \frac{\Delta t}{\varepsilon(1 + C_X)\Delta y} \left[H_z^{n+\frac{1}{2}}(i,j,k) - H_z^{n+\frac{1}{2}}(i,j-1,k) \right] - \frac{\Delta t}{\varepsilon(1 + C_X)\Delta z} \left[H_y^{n+\frac{1}{2}}(i,j,k) - H_y^{n+\frac{1}{2}}(i,j,k-1) \right], \quad C_X = \frac{\Delta x}{R\Delta y\Delta z} \quad (9)$$

In this case the computation coefficients will be:

$$C_{0X} = \frac{1 - C_X}{1 + C_X}, \quad C_{1X} = \frac{\Delta t}{\varepsilon(1 + C_X)\Delta y}, \quad C_{2X} = \frac{\Delta t}{\varepsilon(1 + C_X)\Delta z} \quad (10)$$

In the software, the actual computation coefficient table is implemented as an array of structures instead of a two-dimensional floating-point array (note: the term ‘Structure’ here is a form of datatype in C/C++ Language [15]). Each field component, E_x , E_y , E_z , H_x , H_y , H_z has its own look-up table. The structures are named $stCCE_x$, $stCCE_y$, $stCCE_z$, $stCCH_x$, $stCCH_y$ and $stCCH_z$. Each structure contains ten **double** type variables $fc0$ to $fc9$, a **char** type variable $umode$ and a **void** pointer $plist$. Fig. 15 shows an example of the look-up table for E_x field components:

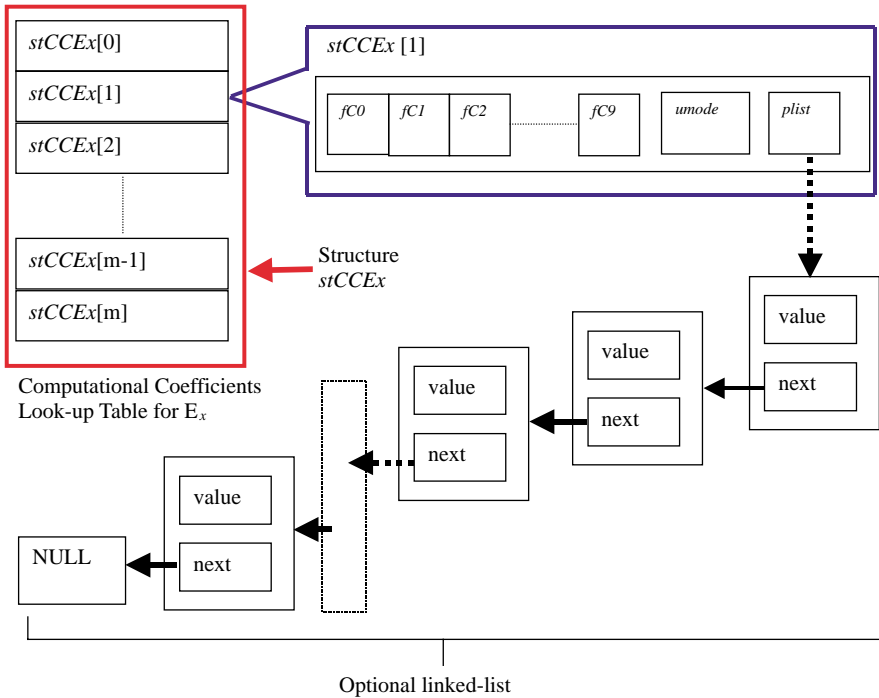


Figure 15. Computational coefficients look-up table for E_x field components.

Each entry $stCCE_x[i]$ can store up to ten coefficients. The variable $umode$ is an 8-bit unsigned integer (known as **char** datatype in C/C++) used to identify the type of lumped element associated with the E_x field component using these coefficients. For instance $umode = 0$ means the E_x field is in a linear, lossless and non-dispersive dielectric, $umode = 100$ means the E_x field component coincide with a lumped resistor and so forth. The values assigned to $umode$ are the same as that used to differentiate various lumped components [16].

The void pointer *plist* can point to a linked-list [15]. This is used when the ten variables *fC0–fC9* are not sufficient to store the computation coefficients, and extra memory locations are needed. This happens for instance for update equations of *E* fields in dispersive dielectric. The linked-list provides a mechanism of variable length storage. It will be questioned why not do away with *fC0–fC9*, having just *plist* and *umode*. The reason is retrieving information from a linked-list is computationally slower and more cumbersome than directly reading from an array. To access *fC2* in *stCCEx[1]*, the following statement is used in C/C++: *stCCEx[1].fC2*. To access the third elements in the linked-list, we have to access the first element, then the second and finally the third in a sequential manner [15]. Most update functions for *E* and *H* field components usually do not require so many coefficients, so having ten variables in the structure and the rest in an optional linked-list provide a compromise between computer memory requirement and speed.

5.3. Other Important Objects in the Simulation Engine

In the parlance of object-oriented programming, there is no variable or datatype. Instead there are only objects or class as it is called in C++ [15]. Each object consists of a number of members, which can be a data storage element such as **integer**, floating-point (**float**), long floating-point (**double**), character (**char**), structures etc. and functions/subroutines. The functions within an object are known as methods. The members of an object have different visibility scope, some members are accessible by functions outside of the object, while others are only accessible by other members within the object. An object can be inherited by other objects, making all its member data and functions available to the child object. The inherited object is usually called the parent object. The child object can add its own members on top of the parent's members, this features form the basis of code reuse in object-oriented technology.

A number of objects are declared for the FDTD Simulation Engine as shown in Fig. 16. Among the more important objects are the electric field object (*tEfield*), magnetic field object (*tHfield*) and Absorbing Boundary Condition object (*tABC*). The electric field object contains the variables to store all the *E* field components in the model. Similarly the magnetic field object contains the variables to store all *H* field components in the model. The Absorbing Boundary Condition object contains some variables and functions for updating the *E* fields tangential to the model boundary. These three are the base objects, sometimes also called the parent objects. A child object, the FDTD field object (*tEHfdd*) inherits both *tEfield* and *tHfield* objects.

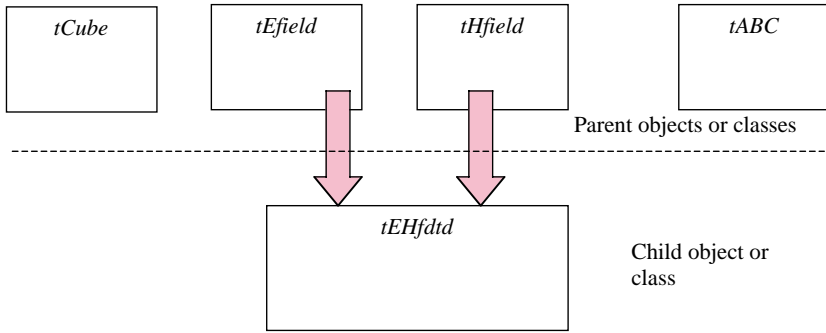


Figure 16. Various objects declared in the FDTD simulation engine.

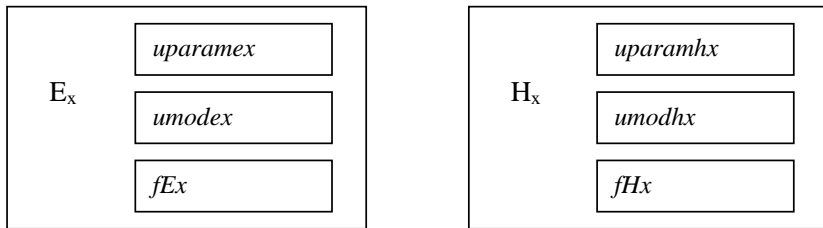


Figure 17. Three data storage locations are needed for each field component.

There is also another base object, the cube object (*tCube*) which is used during the initialization stage and for creating the computation coefficient look-up table. The cube object *tCube* is used to store the characteristics of every cube in the model as the MDF file is read and parsed. Information contained in *tCube* is then used to assist in constructing the look-up table depicted in Fig. 15 for E and H field components. The object *tCube* will be removed from the computer memory once initialization is completed.

Associated with each field component are three variables. A **char** variable to specify the update mode for the field such as Equations (7a) or (9), another **char** variable specifies the row index for the associated computation coefficients look-up table, and lastly a **double** variable to store the current value of the field component. Fig. 17 illustrates this concept for E_x and H_x components. In Fig. 17, *uparamex* and *uparamhx* are the variables associated with the row index of the look-up tables for E_x and H_x field respectively. The variables *umodex* and *umodhx* are associated with the update mode. Lastly the current values for the fields are stored in *fEx* and *fHx*. All these variables are

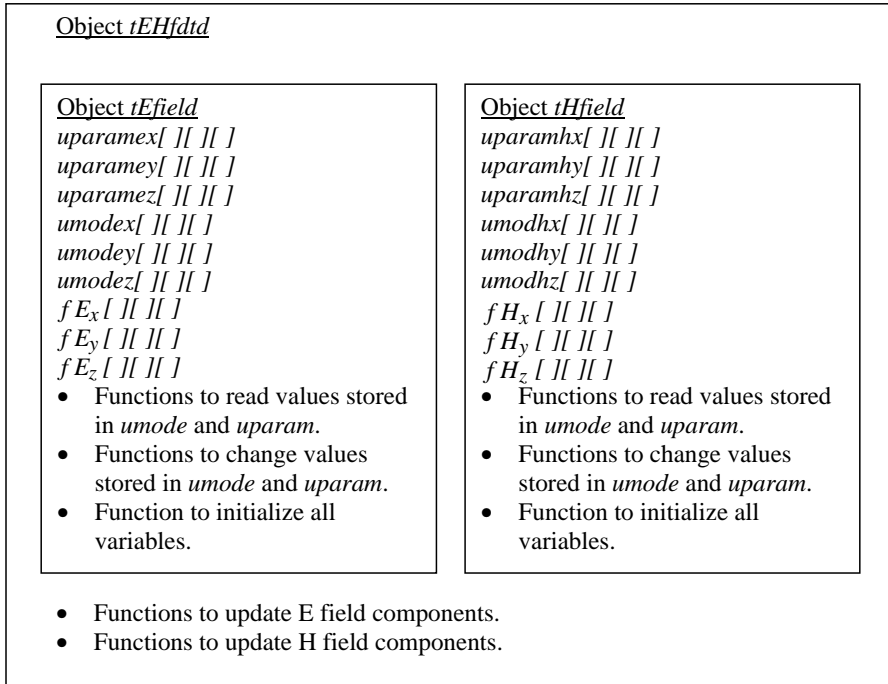


Figure 18. Relationship between classes *tEHfdd*, *tEfield* and *tHfield*.

contained in *tEfield* and *tHfield* objects.

Since the *tEfield* object contains storage locations for all *E* field components, it must have three-dimensional arrays of the variables *uparamex*, *umodex* and *fEx* for *E_x*. Only *fEx* is accessible by external functions. Variables *uparamex* and *umodex* are private to the *tEfield* and can only be accessed by methods declared within the object. This is known as encapsulation in object-oriented programming. It serves to provide transparency to external functions while preventing accidental modification of important variables, reducing the chances of programming error. Another two similar sets of arrays have also been declared in *tEfield* for *E_y* and *E_z*. The same can be said for *tHfield*, where nine three-dimensional arrays are declared for *H_x*, *H_y* and *H_z*. The *tEHfdd* object inherits both *tEfield* and *tHfield* objects. It also introduces all functions required to update *E* and *H* field components under various modes. This relationship is shown in Fig. 18. Finally, Fig. 19 illustrates the details of a typical field component update sequence. This example is for *E_x* field components. Further information for creating the computation coefficient look-up table

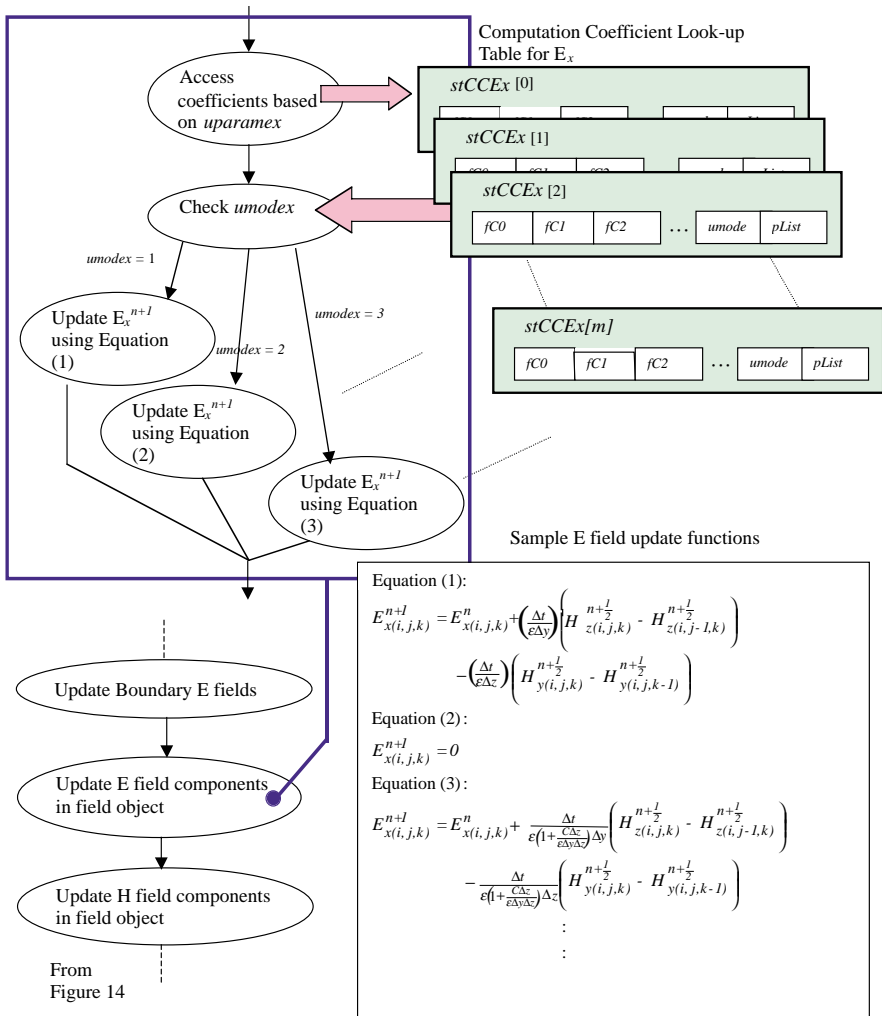


Figure 19. Update E field component sequence, as seen for E_x component.

during the initialization stages can be found in [9]. The procedures shown in Fig. 19 are then repeated for E_y , E_z , H_x , H_y and H_z field components. There are also other objects or data structures not described by this short discussion. These includes a linked-list to store the information for field components to ‘probe’ or save to the output file, and a linked-list for storing the information for various lumped components.

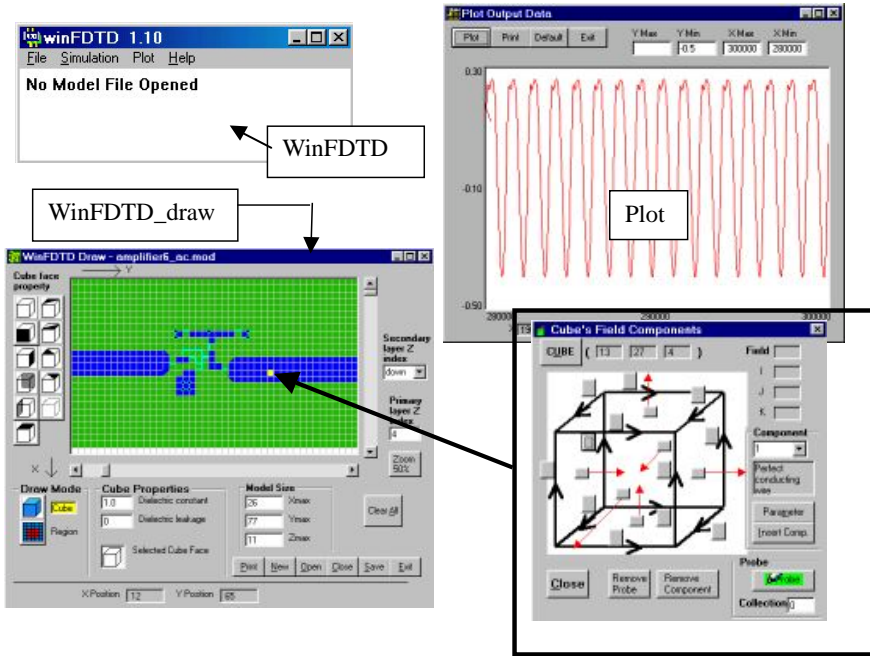


Figure 20. Screen shots of “WinFDTD”, “WinFDTD_draw” and “Plot”.

5.4. The Complete Package

The complete FDTD simulation package developed for this project consists of three separate programs whose screen shots are shown in Fig. 20.

1. The FDTD simulation engine called “WinFDTD”.
2. A utility called “WinFDTD_draw” which is a program for the user to draw the three-dimensional model layer-by-layer. The program then automatically generates the model definition file (MDF) which is then read by WinFDTD.
3. A simple utility called “Plot”, to read the output files and plot the field components, or the equivalent voltages and currents.

Both “WinFDTD_draw” and “Plot” are also Windows™ based programs that are created using the programming language Visual Basic 6.0 by Microsoft Corp. The details will not be described, as these are merely helper programs.

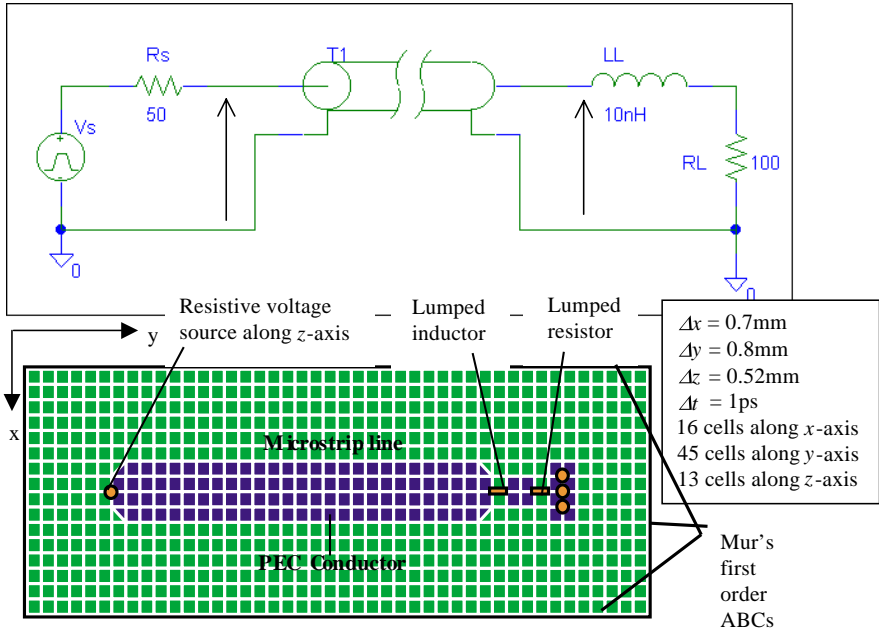


Figure 21. Schematic and top view of the model.

6. SAMPLE SIMULATION EXAMPLES

Transmission Line Loaded with Series RL Impedance

The first simulation example shows a resistive voltage source driving a short length of transmission line with a series RL load. The schematic and top view of the model is shown in Fig. 21. V_s is a trapezoidal pulse source, with rise/fall time of 200 ps, amplitude of 3.3 V, high period of 600 ps and low period of 1000 ps. Transmission line T_1 is a lossless microstrip line, it is designed to have a characteristic impedance of $50\ \Omega$, and propagation delay of 150 ps. The printed circuit board (PCB) is assumed to be linear, lossless with $\epsilon_r = 4.2$. Thickness of the PCB is three cells along the z -axis. A PEC conducting plane covers the bottom of the PCB. At the top and four sides of the model, Mur's first order ABC is employed. The simulation results are shown in Fig. 22. The effective voltages are obtained by performing a numerical line integration of the corresponding E field components. For instance the lumped resistor R_L coincides with $E_{y(9,37,5)}$ and the lumped inductor L_L coincides with $E_{y(9,34,5)}$. Thus the effective voltage

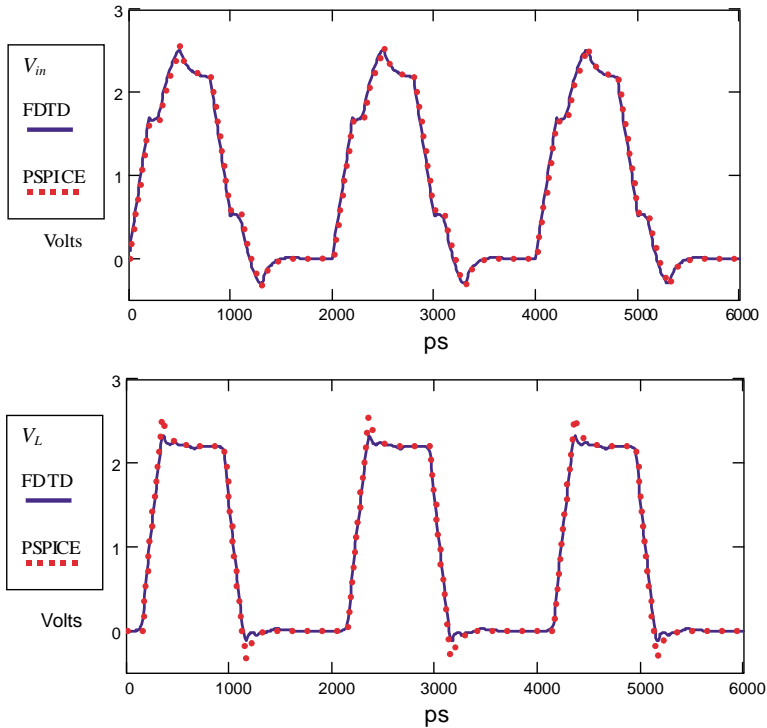


Figure 22. The effective voltage across the series RL load.

across the series RL load can be computed as:

$$V_L^n = - \int \vec{E} \cdot \vec{dl} \cong -N\Delta y \left(E_{y(9,37,5)}^n + E_{y(9,34,5)}^n \right) \quad (11)$$

where $N = -1$ because positive current flows in positive y direction. The schematic of Fig. 21 is also implemented in PSPICE circuit simulator and the results are superimposed with the FDTD results.

Crosstalk Simulation

The second example is a simulation of electromagnetic coupling between two adjacent microstrip lines. This is the classic crosstalk simulation. The schematic and top view of the model are shown in Fig. 23. In the schematic, both the active and passive transmission lines are terminated with its own characteristic impedance of 50Ω . V_s is a trapezoidal pulse source, rise/fall time of 200 ps, amplitude of 3.3 V, high period of 600 ps and low period of 1000 ps. As the electromagnetic energy from V_s travels along the active transmission

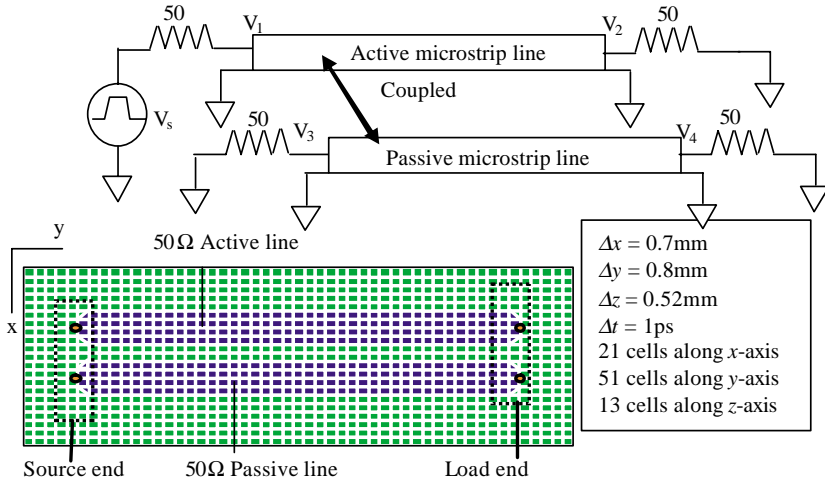


Figure 23. Schematic and top view of the coupled microstrip line model.

line, energy is also coupled to the passive transmission line via electric and magnetic field coupling. The coupled electromagnetic energy appears as voltages across the source and load end of the passive transmission line, known as Near-End Crosstalk (NEXT) and Far-End Crosstalk (FEXT) respectively in the literature [17, 18]. FDTD simulated voltages on the source and load ends of the active and passive transmission lines are shown in Fig. 24, together with waveforms obtained using the RF/microwave CAD software Advance Design System (ADSTM) from Agilent Technologies. The coupled microstrip line model used by ADSTM is based on the model proposed in [24]. As observed there is good agreement of both results.

Oscillator Simulation

A simple UHF oscillator is designed and constructed on FR4 substrate printed circuit board using the schematic shown in Fig. 25. The oscillator configuration is based on the Clapp-Gouriet design [19] and is operational from 800 MHz to 1900 MHz. For demonstration purpose the oscillator frequency is tuned to around 1300 MHz. Again the reasons for using this frequency is similar to that of [7], in that the FR4 dielectric can be considered lossless and non-dispersive at lower frequency. The discretization used is $\Delta x = 0.8$ mm, $\Delta y = 0.8$ mm, $\Delta z = 0.52$ mm and $\Delta t = 1.0$ ps, 24 cells along x -axis, 43 cells along y -axis and 14 cells along z -axis. Thickness of the printed circuit board

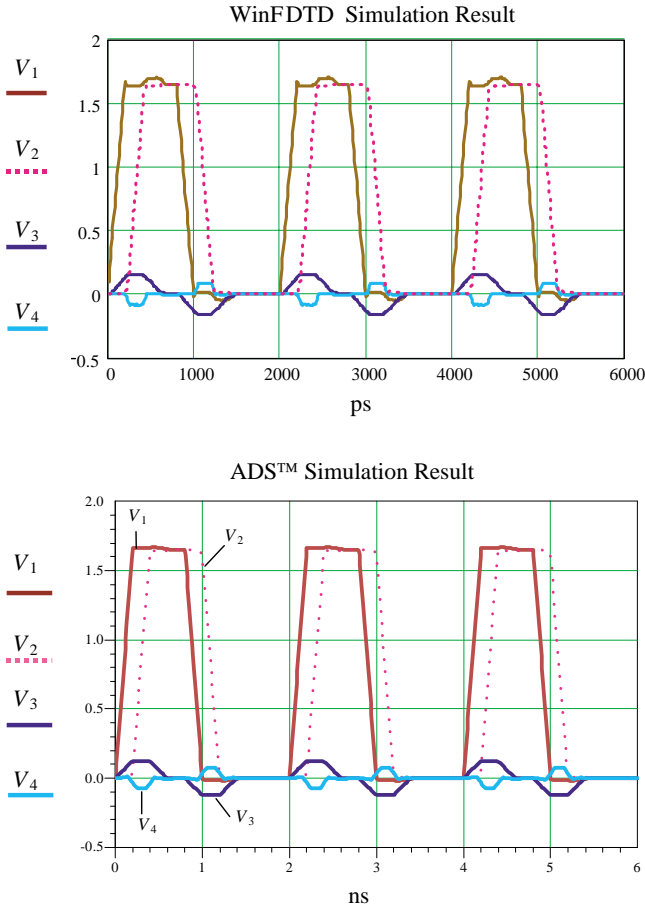


Figure 24. Voltages on the active and passive transmission lines. Near-End Crosstalk (V_3), Far-End Crosstalk (V_4).

is 3 cells. First order Mur ABC is employed at the model boundaries. The BJT employed in the oscillator is BFG520, a wide-band 9 GHz NPN transistor in SOT-143 plastic package [20]. The diode D_1 is a schottky diode, HSMS-2820 from Agilent Technologies [21]. It is housed in SOT-23 plastic package. The SPICE models for both the BJT and schottky diode can be obtained from the manufacturer's website. Inductor L_1 is a ceramic-core SMD inductor in 0805 package. From the datasheet of L_1 , the minimum self-resonating frequency and Q factor of the inductor are 4000 MHz and 40 (measured at 250 MHz) respectively (www.epcos.com). Thus for operating frequency below

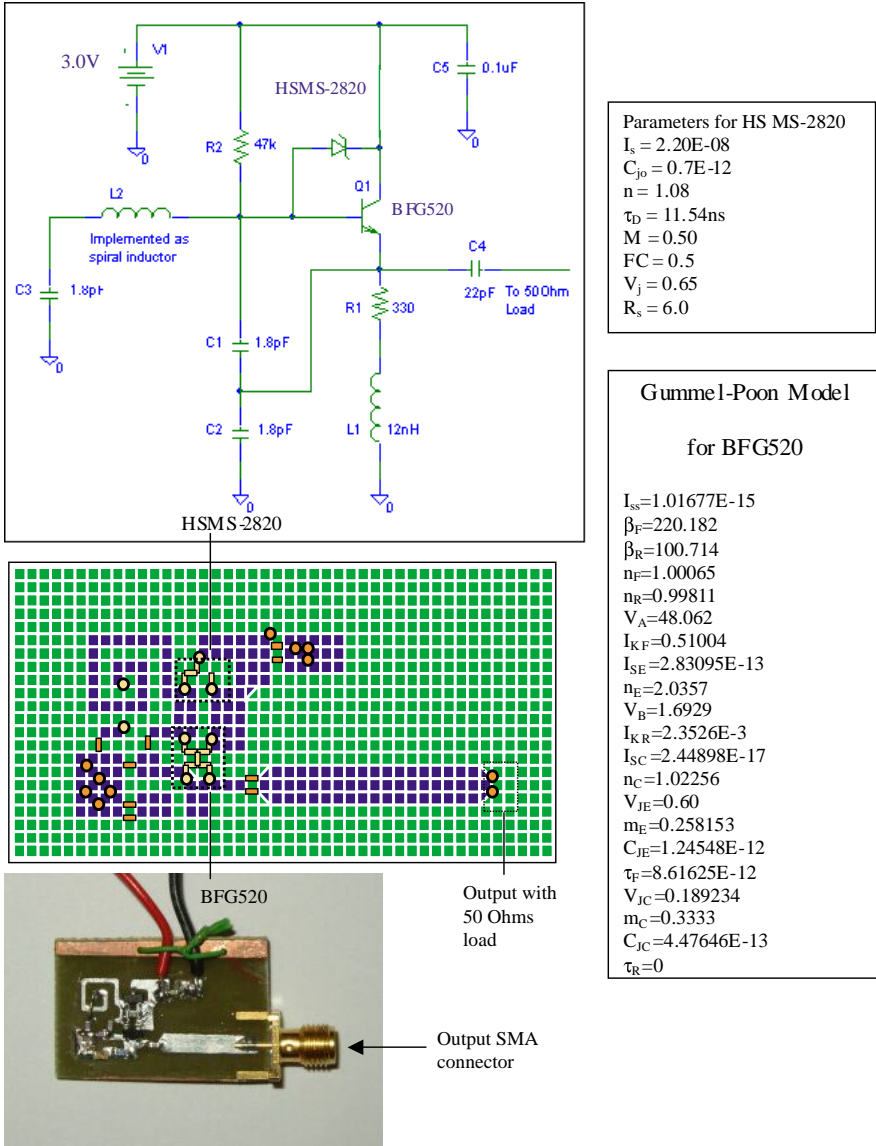


Figure 25. Top view of the oscillator, FDTD model and actual printed circuit board.

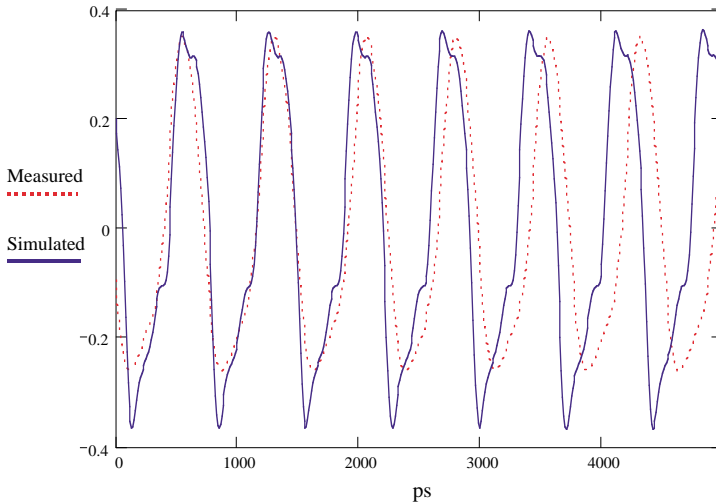


Figure 26. Comparison between simulation and measurement voltage waveforms at the output of the oscillator.

2000 MHz, the inductor can be considered ideal with very little skin-effect loss. Inductor L_2 is implemented on the printed circuit board as spiral inductor [22]. The capacitors are high-quality SMD multi-layer ceramic capacitor in 0603 package while the resistors are high quality SMD resistors in 0805 package. The connector at the output is a $50\ \Omega$ 3.5 mm SMA connector.

A two-step simulation scheme similar to that in [7] is employed. Initially all the lumped capacitors are removed and the lumped inductor is replaced with a short circuit. Simulation is carried out for 70000 time-steps ($\Delta t = 1.0$ ps) until d.c. steady state is achieved. After this all the lumped capacitors and inductor are inserted back to the model. The E and H field components at time-step $n = 70000$ are used as the initial conditions, and the simulation is then rerun until transient steady state is achieved. This two-step scheme shortens the time required to achieve transient steady state [7]. A comparison between simulation and measured waveform at the output port is shown in Fig. 26 for transient steady state. The measured waveform is obtained using the TDS8000 digital-sampling oscilloscope from Tektronix. Full measurement details are presented in [9]. It is observed that the measured oscillation period is approximately 760 ps while the simulated oscillation period is approximately 710 ps. Translated into frequencies, the measured frequency is 1.32 GHz while the simulated frequency is 1.41 GHz. The difference could be attributed to a number of factors:

1. Tolerance and nonideal characteristics of the SMD capacitors and inductor.
2. The SPICE models of the schottky diode and transistor also play a role in the discrepancy.
3. Discontinuity and reflection from the measurement setup.

7. CONCLUSION

A simulation framework for modeling electromagnetic wave propagation in a PCB environment using FDTD approach has been described in this paper. FDTD is chosen, as it is a time-domain method, which made it suitable for incorporating nonlinear components. Moreover being a time-domain method allows examination of the behaviors of the model during transient state. This is advantageous for studying the transient behavior for microwave oscillator circuits and circuits containing high-speed digital signals. Section 4 introduces a convenient notation to describe a complex three-dimensional model cube-by-cube. This notation is very flexible and compact, with ample capacity to include new features in the future. In Section 5, the algorithms presented in Sections 2–4 are systematically converted into a computer program. Object-oriented programming (OOP) technology is used to enable code reuse for future upgrade. Emphasis is on how to store the information describing the model, how to speed up the computation by performing pre-calculation and cross-referencing each field components with the appropriate update functions and coefficients. The software is coded using C++ programming language as it is widely supported at the moment. Conversion to other OOP languages such as JAVA can be easily done as both share many similarities in the syntax. The architecture and data structures (or objects as it is called in OOP) of the software allow a lot of room for expansion in the future. Berenger Perfectly Matched Layer (PML) ABC, ([3, 23]) can also be employed in place of Mur's first order ABC. This can be easily done by modifying the *tABC* object of Section 5. Again this demonstrate the ease with which software written in OOP language can be modified while maintaining the integrity of the whole software. Furthermore, digital integrated circuits can also be incorporated by using the approach of Analog Behavioral Modeling (ABM) [25]. Finally, it must be mentioned that the FDTD formulation does have its limitation at present. For circuits, which emphasize on steady-state sinusoidal response, the FDTD approach would be considerably slower than Method of Moments (MoM) or Finite Element Methods (FEM).

REFERENCES

1. Yee, K. S., "Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media," *IEEE Trans. Antennas and Propagation*, Vol. 14, 302–307, May 1966.
2. Piket-May, M. J., A. Taflove, and J. Baron, "FD-TD modeling of digital signal propagation in 3-D circuits with passive and active loads," *IEEE Trans. Microwave Theory and Techniques*, Vol. 42, 1514–1523, August 1994.
3. Taflove, A., *Computational Electrodynamics — The finite-difference time-domain method*, Artech House, 1995.
4. Ciampolini, P., P. Mezzanotte, L. Roselli, and R. Sorrentino, "Accurate and efficient circuit simulation with lumped-element FDTD technique," *IEEE Trans. Microwave Theory and Techniques*, Vol. 44, 2207–2214, Dec. 1996.
5. Kuo, C., B. Houshmand, and T. Itoh, "Full-wave analysis of packaged microwave circuits with active and nonlinear devices: An FDTD approach," *IEEE Trans. Microwave Theory and Techniques*, Vol. 45, 819–826, May 1997.
6. Kung, F. and H. T. Chuah, "Modeling a diode in FDTD," *J. of Electromagnetic Waves and Appl.*, Vol. 16, No. 1, 99–110, 2002.
7. Kung, F. and H. T. Chuah, "Modeling of bipolar junction transistor in FDTD simulation of printed circuit board," *Progress in Electromagnetic Research*, PIER 36, 179–192, 2002.
8. Strikwerda, J. C., *Finite Difference Schemes and Partial Differential Equations*, Wadsworth & Brooks/Cole Mathematics Series, 1989.
9. Kung, F., "Modeling of electromagnetic waves propagation in printed circuit board and related structures," Ph.D. thesis, Multimedia University, May 2003. <http://pesona.mmu.edu.my/~wllkung/phd>.
10. Kung, F. and H. T. Chuah, "Stability of classical finite-difference time-domain (FDTD) formulation with nonlinear elements — a new perspective," *Progress in Electromagnetic Research*, PIER 42, 49–89, 2003.
11. Luebbers, R. J. and F. Hunsberger, "FDTD for Nth-order dispersive media," *IEEE Trans. Antenna and Propagation*, Vol. 40, 1297–1301, Nov. 1992.
12. Sullivan, D. M., "Z-transform theory and the FDTD method," *IEEE Trans. Microwave Theory Techniques*, Vol. 43, No. 3, 676–682, March 1995.

13. Massobrio, G. and P. Antognetti, *Semiconductor Device Modeling with SPICE*, 2nd edition, McGraw-Hill, 1993.
14. Strauss, R., *Surface Mount Technology*, Butterworth-Heinemann, 1994.
15. Murray III, W. H. and C. H. Pappas, *Borland C++ in Depth*, McGraw-Hill, 1996.
16. http://pesona.mmu.edu.my/~wltkung/winfdtd_exe.zip.
17. Blood, W. R., *MECL System Design Handbook*, 3rd edition, Motorola Semiconductor Products Inc., 1980.
18. Johnson, H. W. and M. Graham, *High-Speed Digital Design — A handbook of black magic*, Prentice Hall, 1993.
19. Smith, J. R., *Modern Communication Circuits*, 2nd edition, McGraw-Hill, 1998.
20. Phillips Semiconductor, “BFG520; BFG520/X, BFG520/XR, NPN 9 GHz wideband transistor,” Technical Data, www.semiconductors.com, Oct. 1997.
21. Agilent Technologies, “Surface mount RF Schottky barrier diodes: HSMS-282x series,” Technical Data, www.semiconductor.agilent.com, 2000.
22. Robertson, I. D. and S. Lucyszyn, *RFIC and MMIC Design and Technology*, The Institution of Electrical Engineers, 2001.
23. Taflove, A. (ed.), *Advances in Computational Electrodynamics — The finite-difference time-domain method*, Artech House, 1998.
24. Kirschning, M. and R. H. Jansen, “Accurate wide-range design equations for frequency-dependent characteristic of parallel coupled microstrip lines,” *IEEE Trans. Microwave Theory and Techniques*, Vol. 32, 83–90, Jan. 1984 (Note, a correction of the paper appears on Vol. 33 on the same journal, page 288).
25. Duran, P. A., *A Practical Guide to Analog Behavioral Modeling for IC System Design*, Kluwer Academics Publishers, 1998.

F. Kung obtained his B.Eng. and M.Eng.Sc. in electrical engineering from University of Malaya in 1994 and 1997 respectively. He worked in Intel Technology in Penang, Malaysia from May 1994–May 1996, and in GMS Technology, in Selangor, Malaysia from June 1997–June 1999. He obtained his Ph.D. in electrical engineering, from Multimedia University, Malaysia, in 2003.

H. T. Chuah obtained his B.Eng., M.Eng.Sc. and Ph.D., all in electrical engineering, from University of Malaya, Malaysia. He is currently the Dean of Faculty of Engineering, Multimedia University. Chuah was the recipient of the inaugural Young Engineer Award by the Institution of Engineers, Malaysia in 1991, and the National Young Scientist Award (Industrial Sector) by the Malaysian Ministry of Science, Technology and the Environment. His research interests are in applied electromagnetic, microwave remote sensing, and device physics.