MÉTODOS DE SIMULACIÓN Y MODELADO

Solución al Trabajo Práctico - Enero de 2024

Ejercicio 1

Lea el artículo citado a continuación, que puede descargar de la página web de la asignatura, y conteste **detalladamente** a las preguntas.

> Åström, K.J., Elmqvist, H., Mattsson, S.E. *Evolution of continuous-time modeling and simulation*. The 12^{th} European Simulation Multiconference, ESM'98, June 16–19, 1998, Manchester, UK.

1. ¿Qué analogías pueden establecerse entre el modelado basado en diagramas de bloques y el paradigma de la simulación analógica?
2. ¿Qué es el paradigma de modelado físico? ¿Qué tipo de modelos matemáticos se obtienen de aplicar el paradigma del modelado físico?
3. ¿Qué diferencias hay entre el paradigma de la simulación analógica y el paradigma del modelado físico?
4. Explique detalladamente la Figura 1 basándose en el artículo.

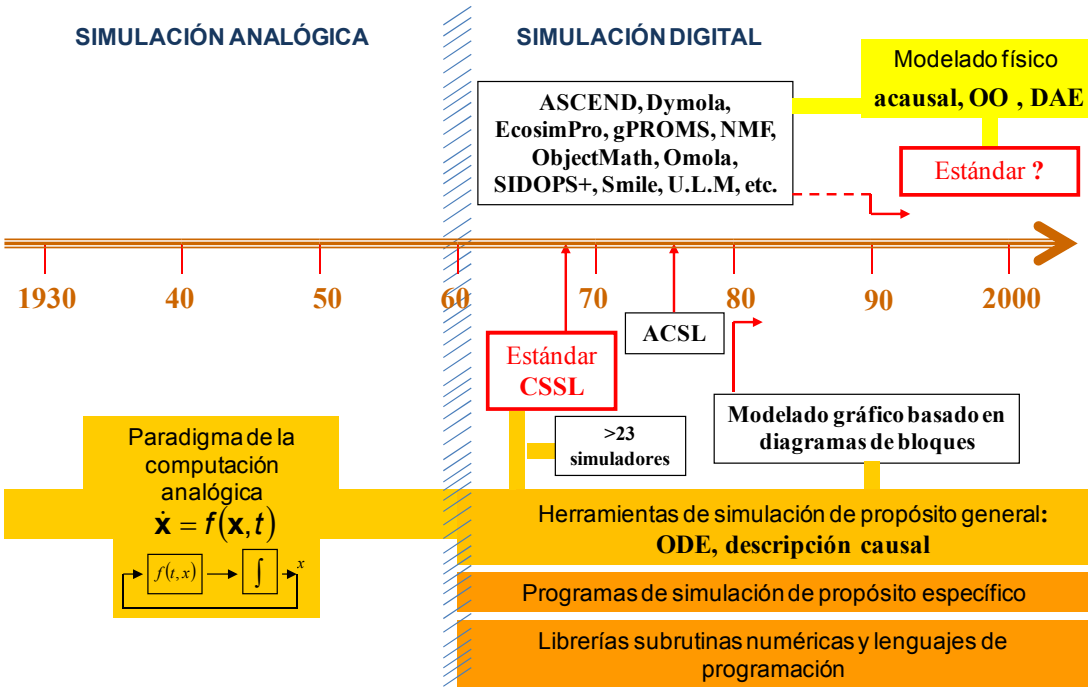

Figura 1: Evolución del modelado y simulación de tiempo continuo.

Solución al Ejercicio 1

En este ejercicio el alumno debe contestar, con sus propias palabras, a las cuestiones planteadas.

Ejercicio 2

El circuito mostrado en la Figura 2 está compuesto por los componentes siguientes: una batería de voltaje constante U_S, dos resistencias de valores constantes R_S y R_L, un condensador de valor constante C, dos interruptores y un circuito de control.

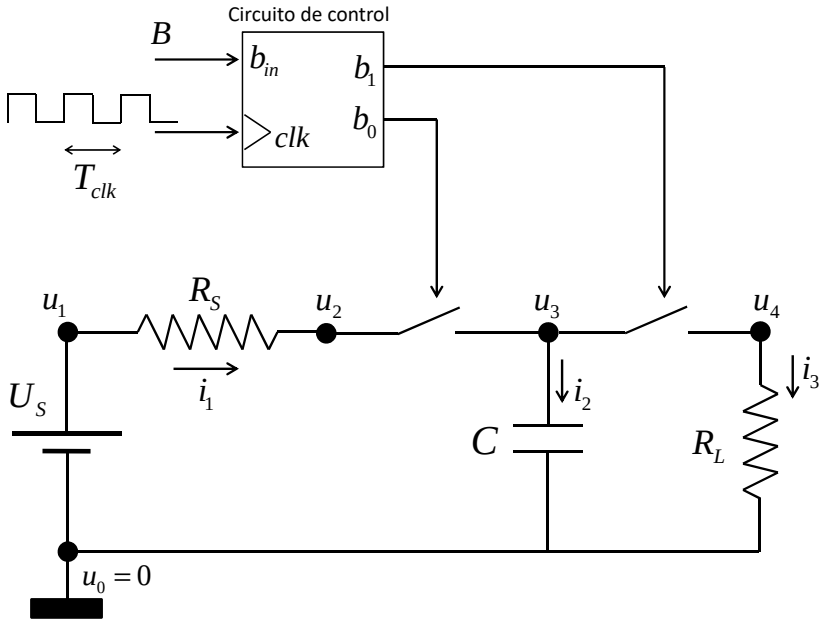

Figura 2: Diagrama del circuito.

El circuito de control tiene dos entradas (b_{in} y la entrada de la señal de reloj clk) y dos salidas (b_0 y b_1). Las entradas aplicadas y las salidas obtenidas son señales binarias. Las señales binarias se describen en este modelo mediante variables Booleanas. Las transiciones en las señales de salida se producen en el flanco de subida de la señal de reloj (clk), es decir, en el instante en que ésta pasa de valer *false* a valer *true*. La señal de reloj tiene un periodo (T_{clk}) constante conocido. La tabla de la verdad del circuito de control es la siguiente:

b_{in}	b_0	b_1
false	true	false
true	false	true

Las dos salidas (b_0 y b_1) del circuito de control están conectadas al puerto de control de sendos interruptores. El interruptor se modelada como una resistencia de valor variable, de la forma descrita a continuación.

- Mientras la señal Booleana aplicada al puerto de control del interruptor vale *true*, la resistencia del interruptor vale $R_{closed} = 10^{-8}$ Ω.
- Mientras la señal Booleana aplicada al puerto de control del interruptor vale *false*, la resistencia del interruptor vale $R_{open} = 10^{8}$ Ω.

El modelo debe permitir calcular la evolución de las variables Booleanas b_0, b_1; de los voltajes u_1, u_2, u_3 y u_4; y de las corrientes i_1, i_2 e i_3.

Se asignan los valores siguientes a los parámetros: $U_S = 12$ V, $R_S = 8000$ Ω, $R_L = 5000$ Ω, $C = 0.0001$ F, $T_{clk} = 1$ s. En el instante inicial, el voltaje u_3 vale 5 V. En el instante inicial, se produce un flanco de subida de la señal de reloj. El modelo se simula durante 12 s. La señal B, que es aplicada al puerto de entrada b_{in} del circuito de control, es conocida. Su evolución se muestra en la Figura 3.

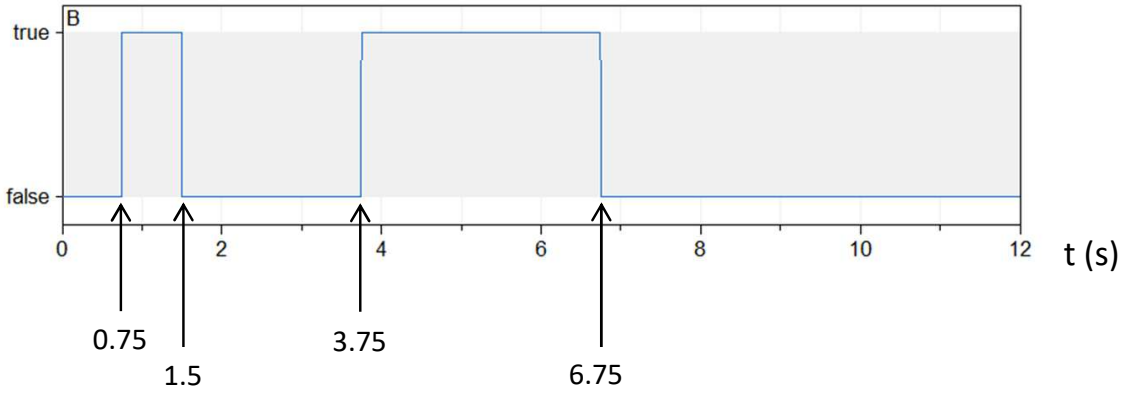

Figura 3: Evolución de la variable B.

1. Escriba las ecuaciones del modelo.
2. Asigne la causalidad computacional. Si el modelo que ha planteado tuviera lazos algebraicos lineales, manipúlelos "a mano" a fin de obtener el modelo ordenado y resuelto.
3. Escriba el diagrama de flujo del algoritmo para la simulación de este modelo. Emplee el método de integración de Euler explícito. La condición de finalización de la simulación es que el tiempo alcance el valor 12 s.
4. Programe el algoritmo anterior en lenguaje R y ejecute la simulación. Explique detalladamente cómo ha escogido el tamaño del paso de integración. Represente gráficamente frente al tiempo las variables siguientes: B, clk, b_0, b_1, u_1, u_2, u_3, u_4, i_1, i_2, i_3.

Solución al Ejercicio 2

Las ecuaciones del modelo se muestran a continuación. No se muestran las asignaciones de valor a las constantes y parámetros. El operador % que aparece en la expresión derecha de la segunda ecuación representa el operador módulo, esto es, el resto obtenido al realizar la división entera. La función sample y la cláusula when se definen como en Modelica. La variable t representa el tiempo medido en segundos.

$$
\begin{aligned}
B &= t \geq 0.75 \textbf{ and } t < 1.5 \textbf{ or } t \geq 3.75 \textbf{ and } t < 6.75 \\
clk &= (t \ \% \ T_{clk}) < 0.5 \\
b_{in} &= B \\
&\textbf{when } \text{sample}(0,\ 0.5 \cdot T_{clk}) \textbf{ then} \\
&\quad b_0 := \textbf{not } b_{in} \\
&\quad b_1 := b_{in} \\
&\textbf{end when} \\
u_1 &= U_S \\
R_{sw1} &= \begin{cases} R_{closed} & \text{si } b_0 = true \\ R_{open} & \text{en caso contrario} \end{cases} \\
R_{sw2} &= \begin{cases} R_{closed} & \text{si } b_1 = true \\ R_{open} & \text{en caso contrario} \end{cases} \\
u_1 - u_2 &= i_1 \cdot R_S \\
u_2 - u_3 &= i_1 \cdot R_{sw1} \\
i1 &= i2 + i3 \\
C \cdot \frac{du_3}{dt} &= i_2 \\
u_3 - u_4 &= i_3 \cdot R_{sw2} \\
u_4 &= i_3 \cdot R_L
\end{aligned}
$$

Asumiendo que la variable que aparece derivada (u_3) pueden ser seleccionada como variable de estado, las variables del modelo pueden clasificarse de la forma siguiente:

- Parámetros y constantes: U_S, R_S, R_L, C, T_{clk}, R_{closed}, R_{open}
- Variables de estado: u_3
- Variables algebraicas: B, clk, b_{in}, b_0, b_1, R_{sw1}, R_{sw2}, u_1, u_2, u_4, i_1, i_2, i_3

Para asignar la causalidad computacional al modelo, se sustituye la derivada de la variable de estado por una variable muda: $\frac{du_3}{dt} \rightarrow der u_3$.

Omitiendo las ecuaciones en las que se asigna valor a los parámetros y las constantes, se obtiene el modelo siguiente:

$$B = t \geq 0.75 \textbf{ and } t < 1.5 \textbf{ or } t \geq 3.75 \textbf{ and } t < 6.75 \tag{1.1}$$

$$clk = (t \ \% \ T_{clk}) < 0.5 \tag{1.2}$$

$$b_{in} = B \tag{1.3}$$

when sample$(0,\ 0.5 \cdot T_{clk})$ **then**

$$b_0 := \textbf{not } b_{in} \tag{1.4}$$

$$b_1 := b_{in} \tag{1.5}$$

end when

$$u_1 = U_S \tag{1.6}$$

$$R_{sw1} = \begin{cases} R_{closed} & \text{si } b_0 = true \\ R_{open} & \text{en caso contrario} \end{cases} \tag{1.7}$$

$$R_{sw2} = \begin{cases} R_{closed} & \text{si } b_1 = true \\ R_{open} & \text{en caso contrario} \end{cases} \tag{1.8}$$

$$u_1 - u_2 = i_1 \cdot R_S \tag{1.9}$$

$$u_2 - u_3 = i_1 \cdot R_{sw1} \tag{1.10}$$

$$i1 = i2 + i3 \tag{1.11}$$

$$C \cdot deru_3 = i_2 \tag{1.12}$$

$$u_3 - u_4 = i_3 \cdot R_{sw2} \tag{1.13}$$

$$u_4 = i_3 \cdot R_L \tag{1.14}$$

Las variables del modelo pueden clasificarse en conocidas (el tiempo, los parámetros, las constantes y las variables de estado) y desconocidas (las variables algebraicas y las derivadas de las variables de estado). A continuación se muestra dicha clasificación.

– Conocidas: t
U_S, R_S, R_L, C, T_{clk}, R_{closed}, R_{open}
u_3

– Desconocidas: B, clk, b_{in}, b_0, b_1, u_1, u_2, u_4, i_1, i_2, i_3, R_{sw1}, R_{sw2}
$deru_3$

Con el fin de analizar si el modelo es *estructuralmente singular*, se comprueba que:

1. El número de ecuaciones y de variables desconocidas (incógnitas) es el mismo. Este modelo tiene 14 ecuaciones y 14 incógnitas.

2. Cada incógnita puede emparejarse con una ecuación en que aparezca y con la cual no se haya emparejado ya otra incógnita.

B	$\rightarrow$	Ec. (1.1),	clk	$\rightarrow$	Ec. (1.2),	b_{in}	$\rightarrow$	Ec. (1.3),
b_0	$\rightarrow$	Ec. (1.4),	b_1	$\rightarrow$	Ec. (1.5),	u_1	$\rightarrow$	Ec. (1.6),
R_{sw1}	$\rightarrow$	Ec. (1.7),	R_{sw2}	$\rightarrow$	Ec. (1.8),	i_1	$\rightarrow$	Ec. (1.9),
u_2	$\rightarrow$	Ec. (1.10),	i_2	$\rightarrow$	Ec. (1.11),	$deru_3$	$\rightarrow$	Ec. (1.12),
u_4	$\rightarrow$	Ec. (1.13),	i_3	$\rightarrow$	Ec. (1.14)			

Aplicando el algoritmo de la partición, se obtiene el siguiente modelo ordenado (existen otras posibles formas de ordenar el modelo que son igualmente válidas), con la causalidad computacional señalada:

$$
\begin{aligned}
u_3 \quad & \text{Variable de estado} \\
[B] &= t \geq 0.75 \ \textbf{and} \ t < 1.5 \ \textbf{or} \ t \geq 3.75 \ \textbf{and} \ t < 6.75 \\
[clk] &= (t \ \% \ T_{clk}) < 0.5 \\
[b_{in}] &= B \\
&\textbf{when} \ \text{sample}(0, \ 0.5 \cdot T_{clk}) \ \textbf{then} \\
&\quad [b_0] := \textbf{not} \ \ b_{in} \\
&\quad [b_1] := b_{in} \\
&\textbf{end} \ \ \textbf{when} \\
[u_1] &= U_S \\
[R_{sw1}] &= \begin{cases} R_{closed} & \text{si} \ \ b_0 = true \\ R_{open} & \text{en caso contrario} \end{cases} \\
[R_{sw2}] &= \begin{cases} R_{closed} & \text{si} \ \ b_1 = true \\ R_{open} & \text{en caso contrario} \end{cases} \\
\left.\begin{aligned} u_1 - u_2 &= i_1 \cdot R_S \\ u_2 - u_3 &= i_1 \cdot R_{sw1} \end{aligned}\right| & \quad \text{Lazo algebraico lineal}, [u_2], [i_1] \\
\left.\begin{aligned} u_3 - u_4 &= i_3 \cdot R_{sw2} \\ u_4 &= i_3 \cdot R_L \end{aligned}\right| & \quad \text{Lazo algebraico lineal}, [u_4], [i_3] \\
i1 &= [i2] + i3 \\
C \cdot [deru_3] &= i_2
\end{aligned}
$$

Manipulando simbólicamente las ecuaciones, se obtiene el modelo ordenado y resuelto (existen otras posibles ordenaciones de las ecuaciones que son igualmente válidas).

$$
\begin{aligned}
u_3 \quad & \text{Variable de estado} \\
[B] &= t \geq 0.75 \textbf{ and } t < 1.5 \textbf{ or } t \geq 3.75 \textbf{ and } t < 6.75 \\
[clk] &= (t \ \% \ T_{clk}) < 0.5 \\
[b_{in}] &= B \\
&\textbf{when } \text{sample}(0,\ 0.5 \cdot T_{clk}) \textbf{ then} \\
&\quad [b_0] := \textbf{not } b_{in} \\
&\quad [b_1] := b_{in} \\
&\textbf{end when} \\
[u_1] &= U_S \\
[R_{sw1}] &= \begin{cases} R_{closed} & \text{si } b_0 = true \\ R_{open} & \text{en caso contrario} \end{cases} \\
[R_{sw2}] &= \begin{cases} R_{closed} & \text{si } b_1 = true \\ R_{open} & \text{en caso contrario} \end{cases} \\
[i_1] &= \frac{u_1 - u_3}{R_S + R_{sw1}} \\
[u_2] &= u_1 - i_1 \cdot R_S \\
[i_3] &= \frac{u_3}{R_L + R_{sw2}} \\
[u_4] &= i_3 \cdot R_L \\
[i2] &= i1 - i3 \\
[deru_3] &= \frac{i_2}{C}
\end{aligned}
$$

En la Figura 4 se muestra el diagrama de flujo para la simulación del modelo, empleando el método de integración de Euler explícito y considerando como condición de finalización que el tiempo simulado alcance el valor 12 s. El modelo posee una variable de estado de tiempo continuo: u_3. Su valor inicial está indicado en el enunciado del ejercicio.

Se ha seleccionado un tamaño del paso igual a 0.01 s. Para ello, se ha repetido la simulación empleando diferentes tamaños del paso, encontrándose que el error cometido escogiendo 0.01 s es admisible para el propósito de este estudio.

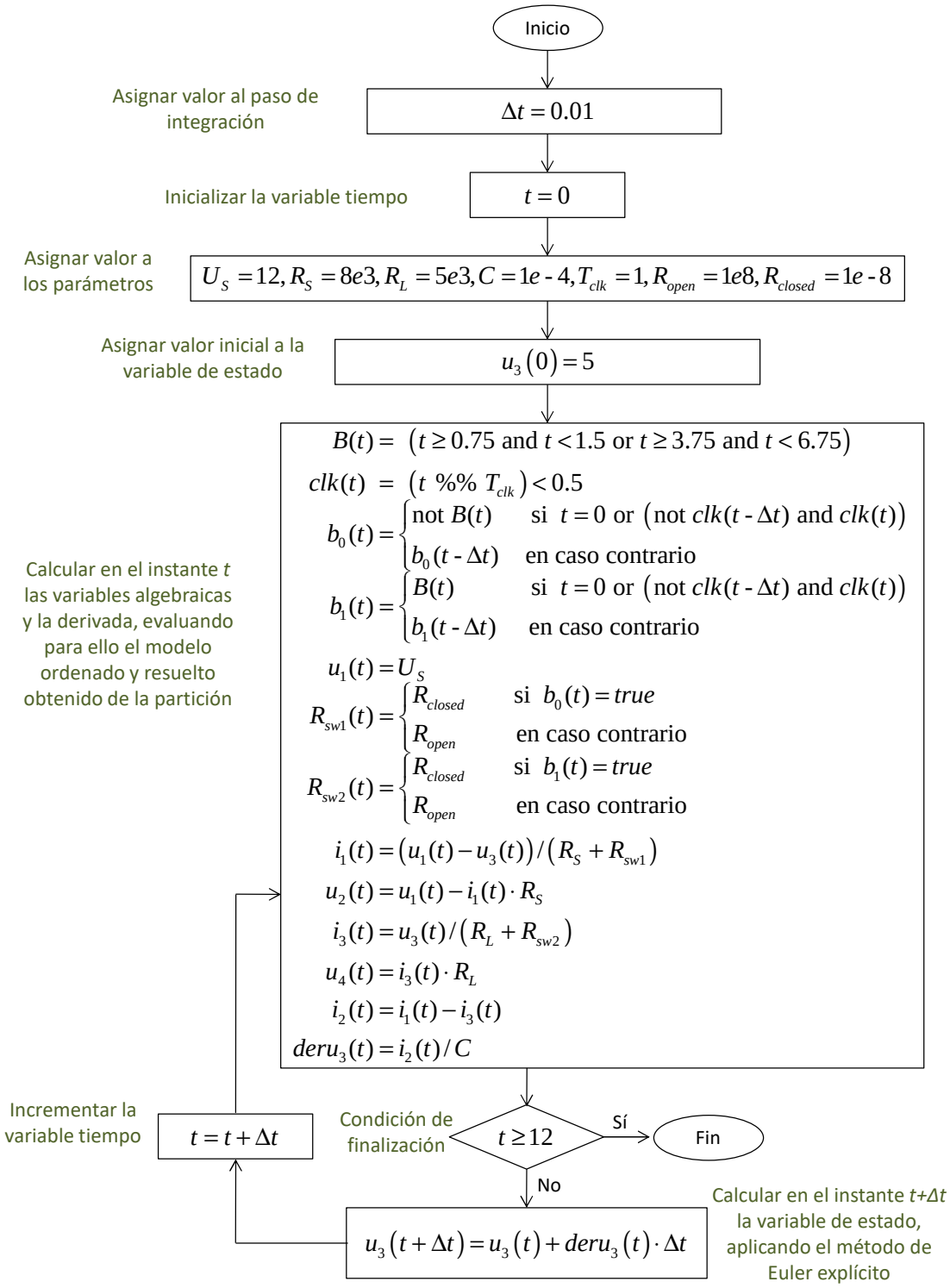

Figura 4: Diagrama de flujo de la simulación del modelo.

Se muestra a continuación el código R que implementa el algoritmo mostrado en la Figura 4, con un intervalo de comunicación igual al tamaño del paso de integración, esto es, 0.01 s. En las Figuras 5 y 6 se muestran las gráficas obtenida al ejecutar el código R.

```
rm( list = ls() )
tiempoFin   <- 12     # Valor final del tiempo
incTiempo   <- 0.01   # Paso de integracion

# Parametros
US       <- 12
RS       <- 8e3
RL       <- 5e3
C        <- 1e-4
Tclk     <- 1
Ropen    <- 1e8
Rclosed  <- 1e-8

# Vector que contiene los instantes de tiempo simulado
tiempo <- seq(0, tiempoFin, by = incTiempo)

# Inicializacion vectores que almacenan las variables del modelo
nPoints <- length(tiempo)
u1   <- numeric(nPoints)
u2   <- numeric(nPoints)
u3   <- numeric(nPoints)
u4   <- numeric(nPoints)
i1   <- numeric(nPoints)
i2   <- numeric(nPoints)
i3   <- numeric(nPoints)
B    <- logical(nPoints)
clk  <- logical(nPoints)
b0   <- logical(nPoints)
b1   <- logical(nPoints)

# Valor inicial del estado de tiempo continuo
u3[1]  <- 5

# Bucle de la simulacion
for ( nStep in c(1:nPoints) ) {

   # Calculo variables algebraicas en t
   B[nStep]    <- tiempo[nStep] >= 0.75 && tiempo[nStep] < 1.5   ||
                  tiempo[nStep] >= 3.75 && tiempo[nStep] < 6.75
   clk[nStep]  <- ( tiempo[nStep] %% Tclk ) < 0.5
   if ( nStep == 1 || !clk[nStep-1] && clk[nStep] ) {
      b0[nStep] <- !B[nStep]
      b1[nStep] <-  B[nStep]
   } else {
      b0[nStep] <- b0[nStep-1]
      b1[nStep] <- b1[nStep-1]
   }
```

```
   u1[nStep]   <- US
   Rsw1 <- if ( b0[nStep] ) Rclosed else Ropen;
   Rsw2 <- if ( b1[nStep] ) Rclosed else Ropen;
   i1[nStep] <- ( u1[nStep] - u3[nStep] ) / ( RS + Rsw1 )
   u2[nStep] <- u1[nStep] - i1[nStep] * RS
   i3[nStep] <- u3[nStep] / ( RL + Rsw2 )
   u4[nStep] <- i3[nStep] * RL
   i2[nStep] <- i1[nStep] - i3[nStep]
   deru3     <- i2[nStep] / C

   # Calculo en el siguiente instante de la variable de estado
   if ( nStep < nPoints ) {
      u3[nStep+1] <- u3[nStep] + incTiempo * deru3
   }

}  # fin del bucle de la simulacion

# Representacion grafica
# Para poder usar minor.tick es necesario instalar el paquete Hmisc
# Configuracion de la representacion grafica

dev.new()
plot(tiempo, clk, xlab = "t [s]", ylab = "clk",
     type="l", col="blue", lwd=1.5,
     xaxp = c(0, tiempoFin, 2*tiempoFin), yaxp = c(0, 1, 1),
     panel.first = abline(v = seq(0,tiempoFin,0.5), h = seq(0,1),
                          lwd = 0.5, lty = 3, col="grey") )
dev.new()
plot(tiempo, B, xlab = "t [s]", ylab = "B",
     type="l", col="blue", lwd=1.5,
     xaxp = c(0, tiempoFin, 2*tiempoFin), yaxp = c(0, 1, 1),
     panel.first = abline(v = seq(0,tiempoFin,0.5), h = seq(0,1),
                          lwd = 0.5, lty = 3, col="grey") )
dev.new()
plot(tiempo, b0, xlab = "t [s]", ylab = "b0",
     type="l", col="blue", lwd=1.5,
     xaxp = c(0, tiempoFin, 2*tiempoFin), yaxp = c(0, 1, 1),
     panel.first = abline(v = seq(0,tiempoFin,0.5), h = seq(0,1),
                          lwd = 0.5, lty = 3, col="grey") )
dev.new()
plot(tiempo, b1, xlab = "t [s]", ylab = "b1",
     type="l", col="blue", lwd=1.5,
     xaxp = c(0, tiempoFin, 2*tiempoFin), yaxp = c(0, 1, 1),
     panel.first = abline(v = seq(0,tiempoFin,0.5), h = seq(0,1),
                          lwd = 0.5, lty = 3, col="grey") )

dev.new()
plot(tiempo, u1, xlab = "t [s]", ylab = "u1 [V]",
     type="l", col="blue", lwd=1.5,
     xaxp = c(0, tiempoFin, 2*tiempoFin),
     panel.first = abline(v = seq(0,tiempoFin,0.5), h = seq(0,20,1),
                          lwd = 0.5, lty = 3, col="grey") )

dev.new()
```

```
plot(tiempo, u2, xlab = "t [s]", ylab = "u2 [V]",
     type="l", col="blue", lwd=1.5,
     xaxp = c(0, tiempoFin, 2*tiempoFin),
     panel.first = abline(v = seq(0,tiempoFin,0.5), h = seq(0,20,1),
                          lwd = 0.5, lty = 3, col="grey") )

dev.new()
plot(tiempo, u3, xlab = "t [s]", ylab = "u3 [V]",
     type="l", col="blue", lwd=1.5,
     xaxp = c(0, tiempoFin, 2*tiempoFin),
     panel.first = abline(v = seq(0,tiempoFin,0.5), h = seq(0,20,1),
                          lwd = 0.5, lty = 3, col="grey") )

dev.new()
plot(tiempo, u4, xlab = "t [s]", ylab = "u4 [V]",
     type="l", col="blue", lwd=1.5,
     xaxp = c(0, tiempoFin, 2*tiempoFin),
     panel.first = abline(v = seq(0,tiempoFin,0.5), h = seq(0,20,1),
                          lwd = 0.5, lty = 3, col="grey") )

dev.new()
plot(tiempo, i1, xlab = "t [s]", ylab = "i1 [A]",
     type="l", col="blue", lwd=1.5,
     xaxp = c(0, tiempoFin, 2*tiempoFin),
     panel.first = abline(v = seq(0,tiempoFin,0.5),
                          h = seq(0,0.002,0.0001),
                          lwd = 0.5, lty = 3, col="grey") )

dev.new()
plot(tiempo, i2, xlab = "t [s]", ylab = "i2 [A]",
     type="l", col="blue", lwd=1.5,
     xaxp = c(0, tiempoFin, 2*tiempoFin),
     panel.first = abline(v = seq(0,tiempoFin,0.5),
                          h = seq(-0.003,0.003,0.0002),
                          lwd = 0.5, lty = 3, col="grey") )

dev.new()
plot(tiempo, i3, xlab = "t [s]", ylab = "i3 [A]",
     type="l", col="blue", lwd=1.5,
     xaxp = c(0, tiempoFin, 2*tiempoFin),
     panel.first = abline(v = seq(0,tiempoFin,0.5),
                          h = seq(0,0.003,0.0001),
                          lwd = 0.5, lty = 3, col="grey") )
```

Código 1: Programación en R del algoritmo de la simulación.

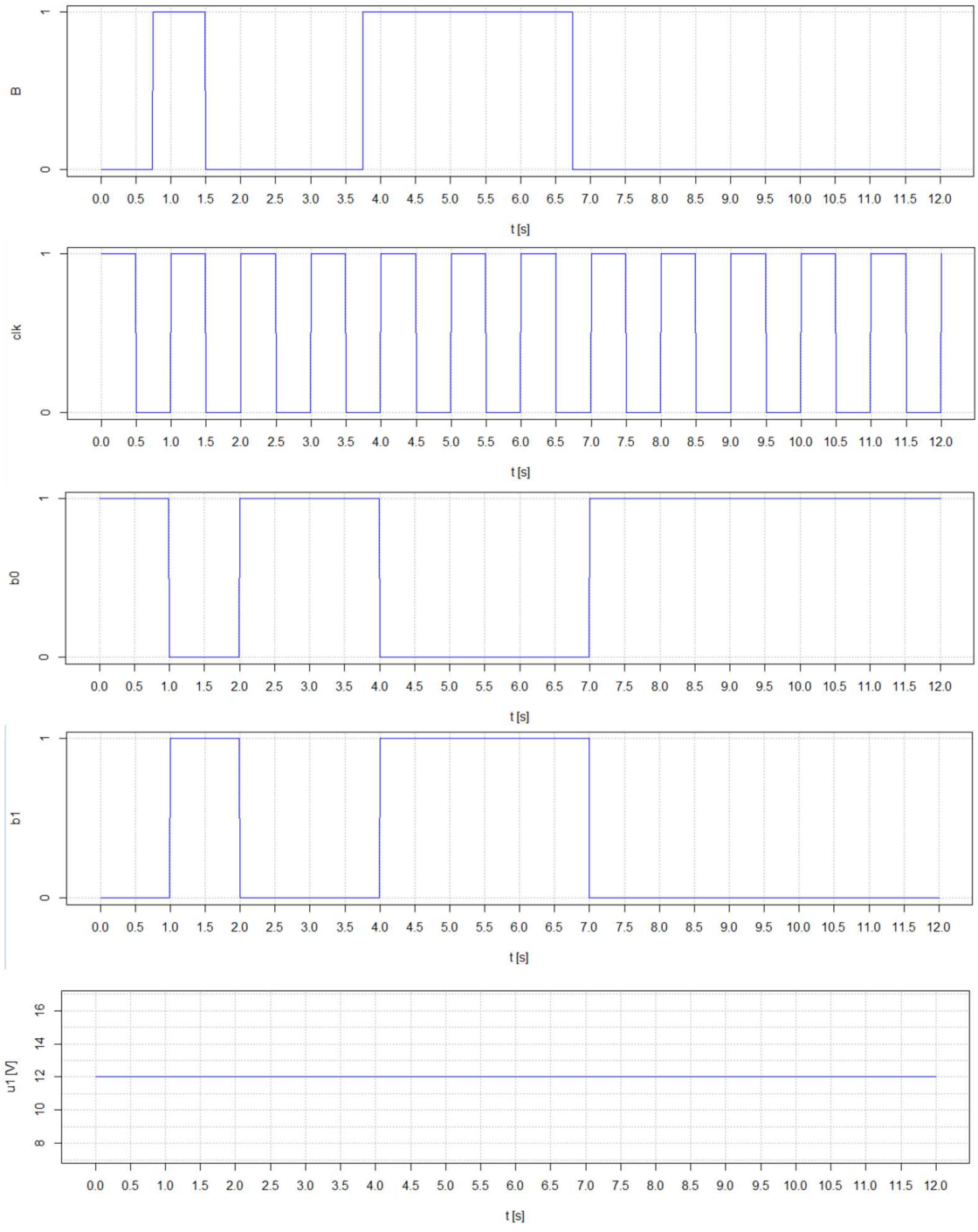

Figura 5: Resultado de la ejecución del código R.

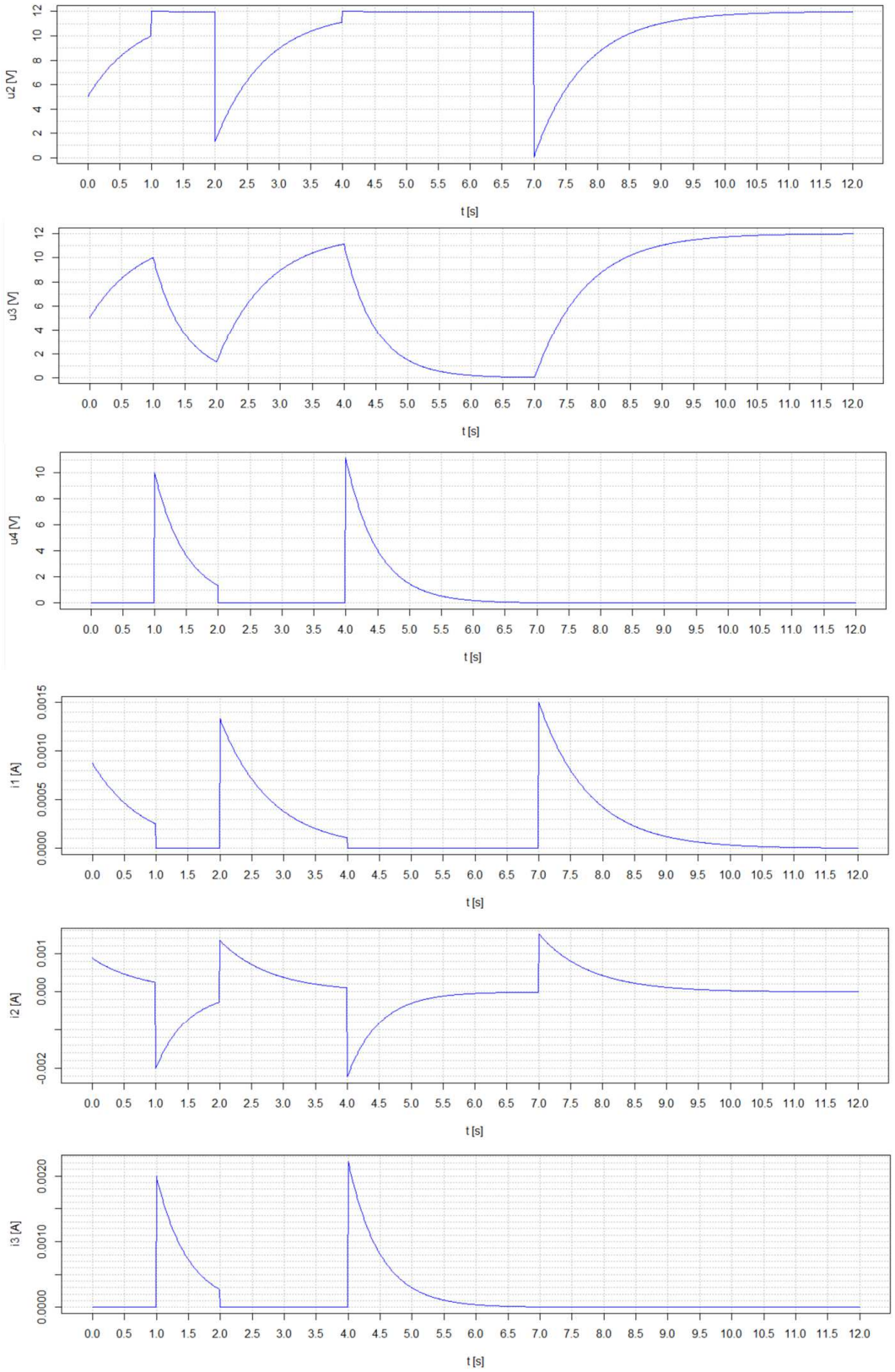

Figura 6: Resultado de la ejecución del código R.

Ejercicio 3

Describa en lenguaje Modelica el sistema del ejercicio anterior, de las dos maneras siguientes:

1. Como un modelo atómico, que viene descrito por las ecuaciones que usted ha planteado al contestar a la pregunta anterior.
2. Programe una librería que contenga los componentes necesarios para componer el sistema descrito en el Ejercicio 2. A continuación, defina dicho sistema como un modelo compuesto, instanciando y conectando componentes de la librería que ha creado.

Asigne a los parámetros los valores indicados en el ejercicio anterior y simule los dos modelos anteriores durante 12 s. Compruebe que las simulaciones de ambos modelos proporcionan el mismo resultado. Para ello, represente gráficamente frente al tiempo las variables siguientes: B, clk, b_0, b_1, u_1, u_2, u_3, u_4, i_1, i_2, i_3.

Solución al Ejercicio 3

El modelo atómico en lenguaje Modelica se muestra en Código 2. El resultado de la simulación está representado en las Figuras 7 – 9.

En el Código 3 se muestra una posible forma de modelar los componentes, organizándolos en una librería, y de componer el sistema mediante dichos componentes. En este caso, se asigna valor inicial al voltaje del condensador en la interfaz gráfica del entorno de modelado, una vez traducido el modelo.

```
model circ
  import SI = Modelica.Units.SI;
  parameter SI.Voltage US = 12;
  parameter SI.Resistance RS = 8e3, RL = 5e3;
  parameter SI.Capacitance C = 1e-4;
  parameter SI.Time T_clk = 1;
  parameter SI.Resistance Ropen = 1e8, Rclosed = 1e-8;
  SI.Voltage u_1, u_2, u_3(start=5, fixed=true), u_4;
  SI.Current i_1, i_2, i_3;
  Boolean B, b_in, clk(start=false, fixed=true), b_0, b_1;
  SI.Resistance Rsw1, Rsw2;
equation
  // Entradas al circuito de control
  when sample(0, 0.5*T_clk) then
     clk = not pre(clk);
  end when;
  B = time >= 0.75 and time < 1.5 or time >= 3.75 and time < 6.75;
  // Circuito de control
  b_in = B;
  when clk then
     b_0 = not b_in;
     b_1 = b_in;
  end when;
  // Circuito analogico
  u_1 = US;
  u_1 - u_2 = i_1 * RS;
  Rsw1 = if b_0 then Rclosed else Ropen;
  u_2 - u_3 = i_1 * Rsw1;
  C * der(u_3) = i_2;
  i_1 = i_2 + i_3;
  Rsw2 = if b_1 then Rclosed else Ropen;
  u_3 - u_4 = i_3 * Rsw2;
  u_4 = i_3 * RL;
  annotation (experiment(StopTime=12));
end circ;
```

Código 2: Modelo atómico en Modelica.

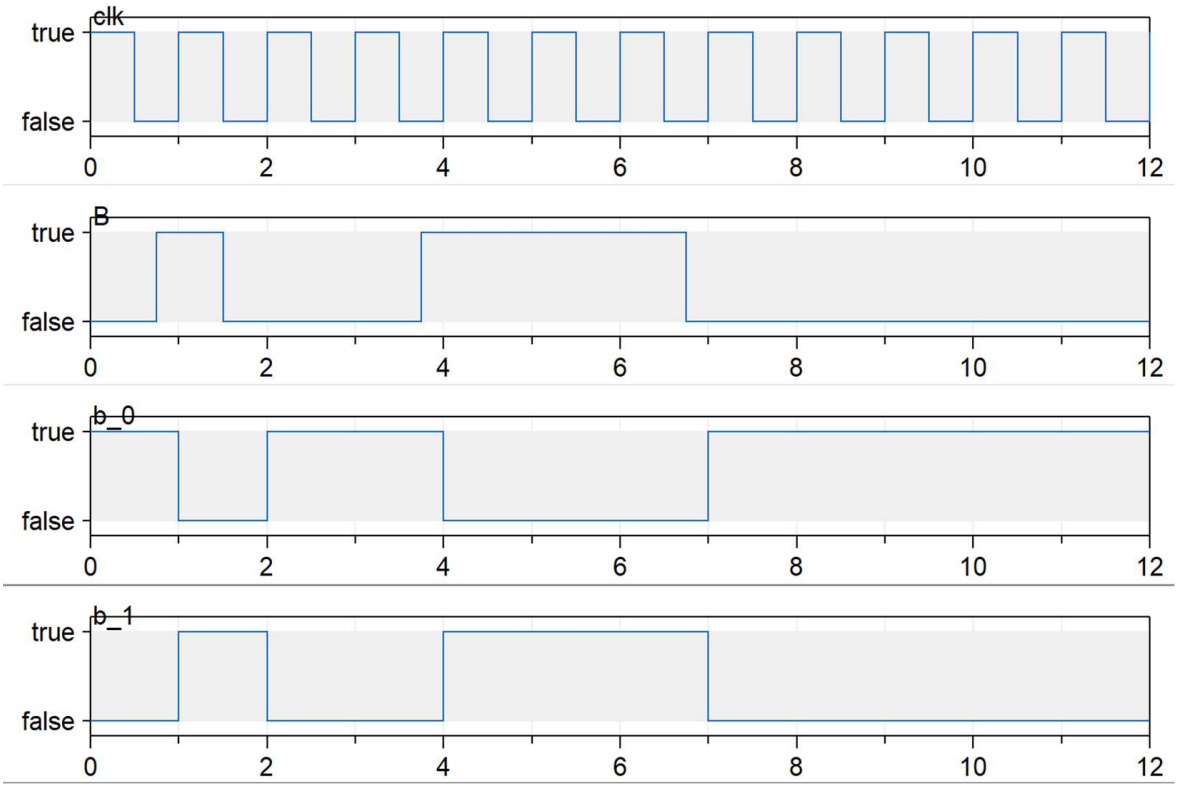

Figura 7: Resultado de la ejecución del código Modelica.

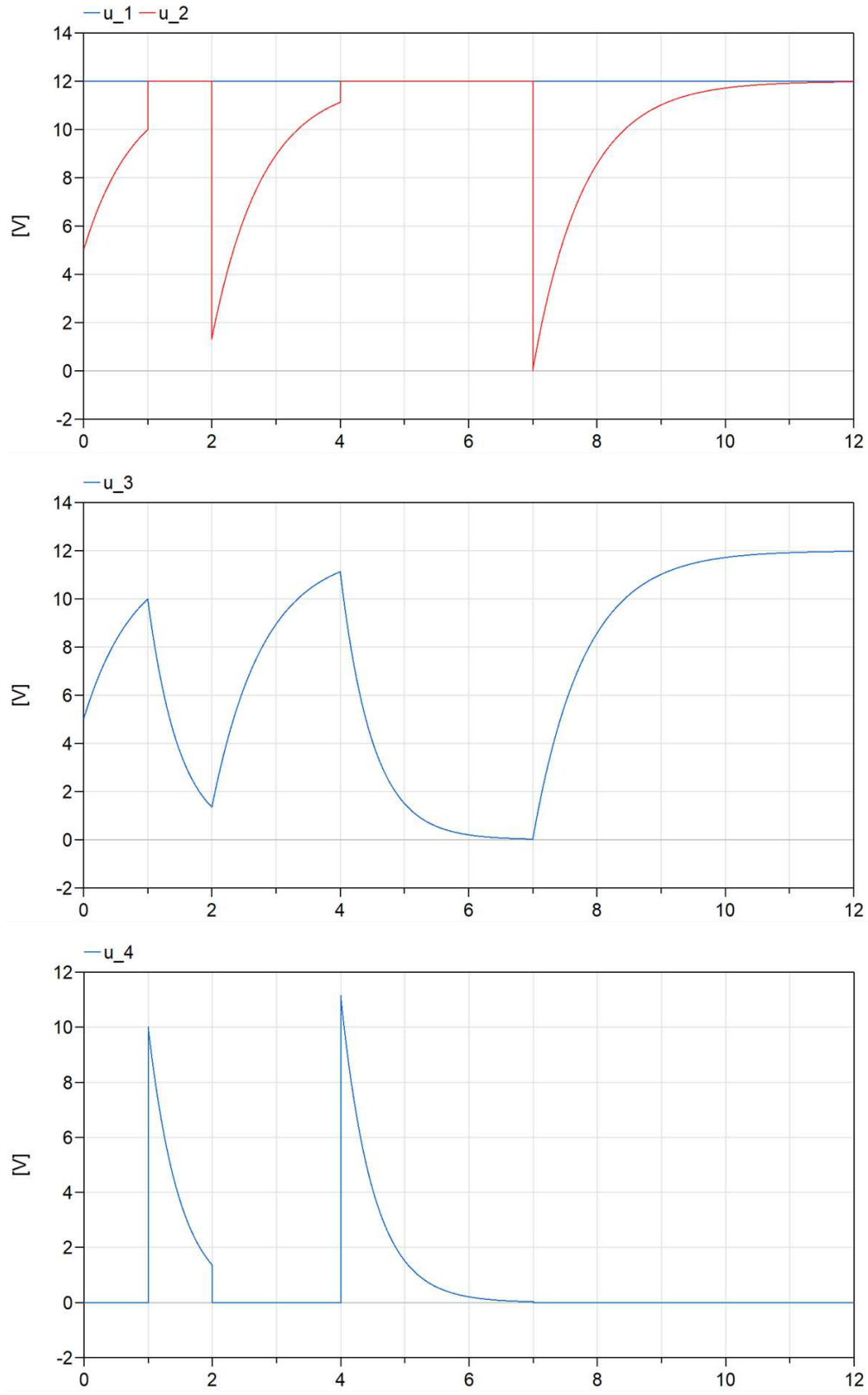

Figura 8: Resultado de la ejecución del código Modelica.

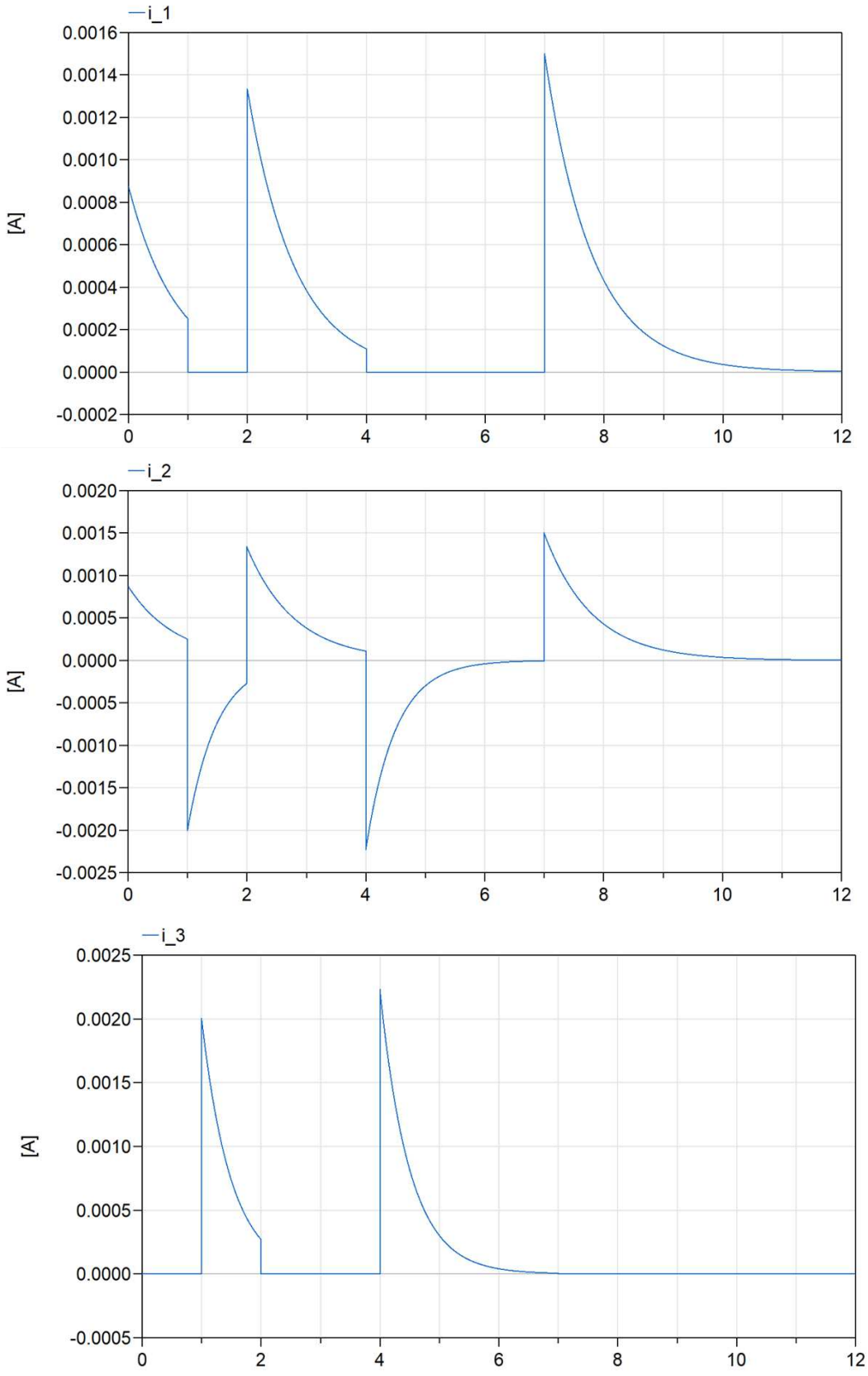

Figura 9: Resultado de la ejecución del código Modelica.

```
package LibElectrica

  import SI = Modelica.Units.SI;


  package Interfaces

    connector Pin
      SI.Voltage u;
      flow SI.Current i;
    end Pin;

    connector BinaryInput
      input Boolean b;
    end BinaryInput;

    connector BinaryOutput
      output Boolean b;
    end BinaryOutput;

    model OnePort
      Interfaces.Pin p, n;
    protected
      SI.Voltage u "Voltaje entre pines (= p.u - n.u)";
    equation
      u = p.u - n.u;
      p.i = -n.i;
    end OnePort;

  end Interfaces;


  package BooleanInputSignals

    model Signal_B
       Interfaces.BinaryOutput B;
    equation
       B.b = time >= 0.75 and time < 1.5 or
             time >= 3.75 and time < 6.75;
    end Signal_B;

    model Signal_clk
       Interfaces.BinaryOutput Clk;
       parameter SI.Time T_clk = 1;
    equation
       when sample(0, 0.5*T_clk) then
          Clk.b = not pre(Clk.b);
       end when;
    end Signal_clk;

  end BooleanInputSignals;


```

```
54
55  package Components
56
57    model ControlCircuit
58      Interfaces.BinaryInput  B_in, Clk;
59      Interfaces.BinaryOutput B_0, B_1;
60    equation
61      when Clk.b then
62        B_0.b = not B_in.b;
63        B_1.b = B_in.b;
64      end when;
65    end ControlCircuit;
66
67    model Battery
68      extends Interfaces.OnePort;
69      parameter SI.Voltage U;
70    equation
71      u = U;
72    end Battery;
73
74    model Resistor
75      extends Interfaces.OnePort;
76      parameter SI.Resistance R;
77    equation
78      u = R * p.i;
79    end Resistor;
80
81    model Switch
82      extends Interfaces.OnePort;
83      Interfaces.BinaryInput  B;
84      parameter SI.Resistance Ropen = 1e8, Rclosed = 1e-8;
85      SI.Resistance Rsw;
86    equation
87      Rsw = if B.b then Rclosed else Ropen;
88      u = Rsw * p.i;
89    end Switch;
90
91    model Capacitor
92       extends Interfaces.OnePort;
93       parameter SI.Capacitance C "Capacidad";
94    equation
95       C * der(u) = p.i;
96    end Capacitor;
97
98    model Ground
99      Interfaces.Pin p;
100   equation
101     p.u = 0;
102   end Ground;
103
104 end Components;
105
106
107
```

```
108
109   package Circuits
110
111     model circ
112       Components.Battery Bat(U=12);
113       Components.Resistor RS(R=8e3), RL(R=5e3);
114       Components.Capacitor C(C=1e-4);
115       Components.Switch Sw1, Sw2;
116       Components.Ground Ground;
117       Components.ControlCircuit Cntrl;
118       BooleanInputSignals.Signal_B Signal_B;
119       BooleanInputSignals.Signal_clk Signal_clk;
120     equation
121       connect(Signal_B.B, Cntrl.B_in);
122       connect(Signal_clk.Clk, Cntrl.Clk);
123       connect(Cntrl.B_0, Sw1.B);
124       connect(Cntrl.B_1, Sw2.B);
125       connect(Bat.p, RS.p);
126       connect(Bat.n, Ground.p);
127       connect(RS.n, Sw1.p);
128       connect(Sw1.n, C.p);
129       connect(C.n, Ground.p);
130       connect(C.p, Sw2.p);
131       connect(Sw2.n, RL.p);
132       connect(RL.n, Ground.p);
133       annotation (experiment(StopTime=12));
134     end circ;
135
136   end Circuits;
137
138 end LibElectrica;
```

Código 3: Librería y modelo compuesto en Modelica.

Ejercicio 4

Consideremos el problema en dos dimensiones de un tubo cilíndrico macizo, fabricado con un material aislante térmico, de conductividad térmica κ, que contiene en su interior dos tubos cilíndricos de diferente radio que transportan agua a diferente temperatura. En la Figura 10 se muestra el sistema descrito en dos dimensiones, que es simétrico respecto al eje X.

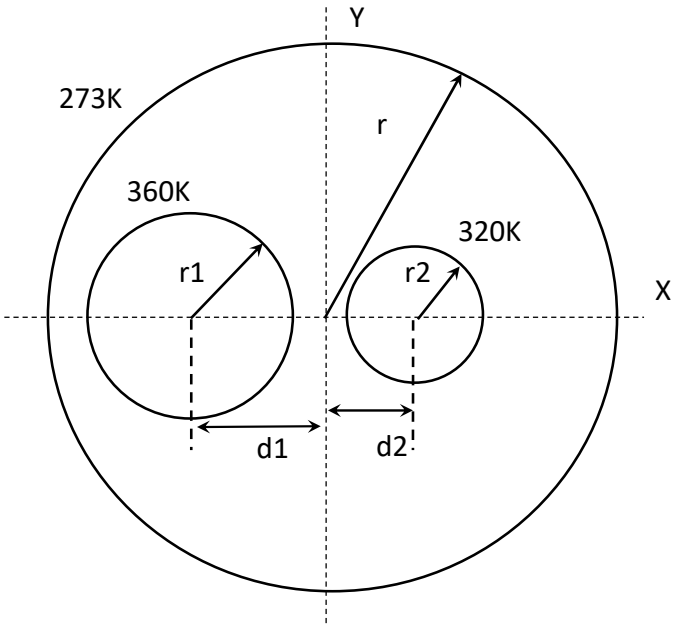

Figura 10: Sistema térmico.

El radio del tubo exterior es $r = 0.5$ m, y los radios de los tubos internos miden $r1 = 0.15$ m y $r2 = 0.1$ m. La distancia $d1$ vale 0.2 m y la distancia $d2$ vale 0.15 m.

La parte exterior del tubo de radio r se mantiene a una temperatura de 273 K, y los tubos interiores izquierdo y derecho se mantienen a unas temperaturas de 360 K y 320 K respectivamente. La conductividad térmica del material aislante vale $\kappa = 0.03\ \frac{W}{K \cdot m}$.

Obtenga un gráfico con las curvas de nivel de la temperatura, empleando FlexPDE. Escriba el código del *script* de FlexPDE correspondiente y muestre el gráfico generado por FlexPDE.

Solución al Ejercicio 4

El *script* se muestra en el Código 4, el gráfico generado por FlexPDE se muestra en la Figura 11.

```
TITLE 'Tuberias'
SELECT      errlim=3e-4     ngrid=1     spectral_colors
VARIABLES   temp  { unidades SI : m, K, W }
DEFINITIONS
   r=0.5    r1=0.15    r2 =0.1    d1=0.2    d2 = 0.15
   k=0.03          {Conductividad termica }
EQUATIONS
   div( -k*grad(temp))=0
BOUNDARIES
region 'dominio'
   start 'exterior' (0,-r) value(temp)=273      { Suelo congelado }
   arc( center=0,0)  angle=360
   start 'izquierda' (-d1-r1,0) value(temp)=360 { Tuberia izquierda }
   arc( center=-d1,0) angle=360
   start 'derecha' (d2-r2,0) value(temp)=320    { Tuberia derecha }
   arc( center=d2,0) angle=360
PLOTS
   contour( temp)
END
```

Código 4: Modelo descrito en FlexPDE.

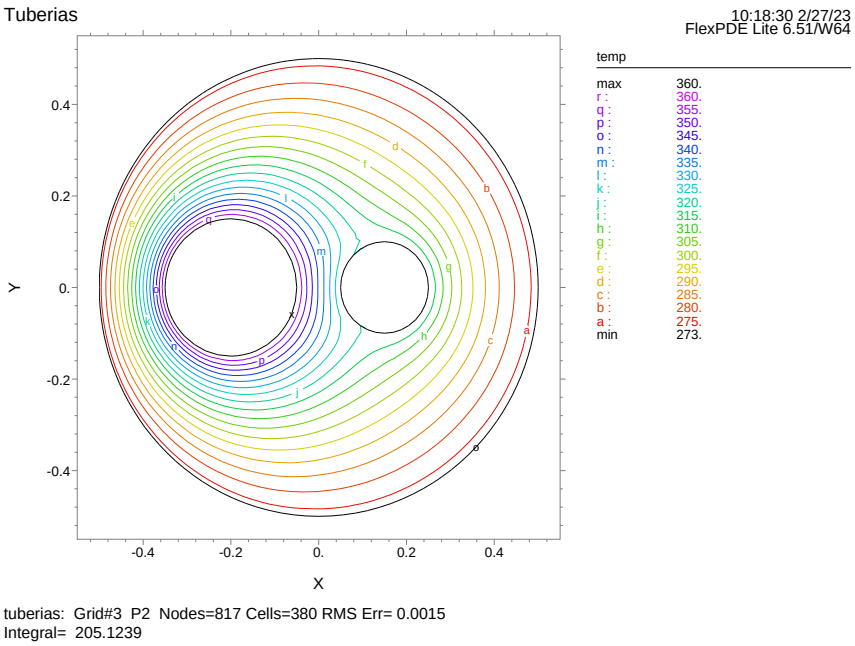

Figura 11: Gráfico generado por FlexPDE.