Phase Change Memory Modeling Using Verilog-A

Yi-Bo Liao
Department of Electronic
Engineering, National Ilan
University, Taiwan, R.O.C.
r9542015@niu.edu.tw

Yan-Kai Chen
Department of Electronic
Engineering, National Ilan
University, Taiwan, R.O.C.
r9426009@niu.edu.tw

Meng-Hsueh Chiang
Department of Electronic
Engineering, National Ilan
University, Taiwan, R.O.C.
mhchiang@niu.edu.tw

ABSTRACT

In this paper, we successfully develop a compact phase change memory (PCM) model using Verilog-A. As PCM has shown its potential for next generation memory device, a predictive, yet simple-to-use circuit model is crucial to the development. Since the Verilog-A modeling is flexible and portable for many circuit simulators, the proposed modeling technique can be widely used, as compared with conventional modeling schemes.

1. INTRODUCTION

Non-volatile memory (NVM) has been very popular for data storage due to its high density feature. As the demand for data amount increases rapidly, especially for multimedia applications, memory density has become one of the key factors for technology development. However, continuous scaling of MOSFET is a challenging task due to the physical limitations. Among the emerging NVM technologies, the PCM has been a promising candidate for its fast read/write and high scalability. PCM was also called Ovonic Unified Memory (OUM), which was first presented by S. Ovshinsky in 1968. The material of PCM is based on $Ge_2Sb_2Te_5$ (GST) [1][2][3].

2. BACKGROUND

PCM has two stable phase states: crystalline phase (Set) and amorphous phase (Reset). In **Figure. 1**, it presents the difference of the two states: the set state is low-resistance (~7 KΩ) and the reset state is high-resistance (~200 KΩ).

The phase state can be changed by the current pulse. In this example [4], the current pulse width was kept at 100 ns, and if the current pulse amplitude achieves 700 μA, defined as I_{Set}, the phase of PCM will be changed to the crystalline state. In another situation, if current pulse amplitude is over 1200 μA, defined as I_{Reset}, the phase of PCM will become the amorphous state. When the programming current pulse falls between zero and I_{Set} or between I_{Set} and I_{Reset}, PCM is in the incomplete (partial) transition. The relationship between the resistance and the current pulse amplitude is shown in **Figure. 2** [4].

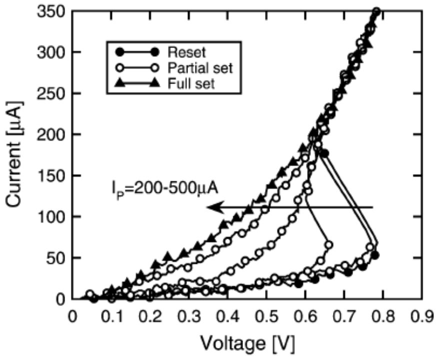

Figure. 1 I-V curves of PCM [4].

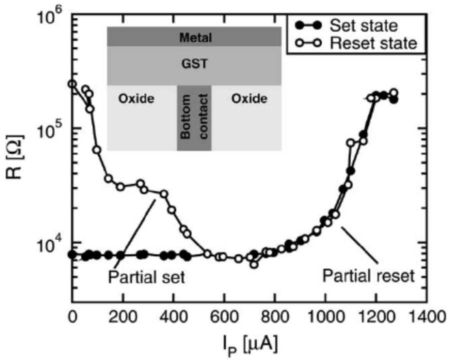

Figure. 2 R-I curves of PCM [4].

The proposed PCM modeling is based on the Verilog-A feature of HSPICE [5]. However the presented modeling technique can be also applied to other simulators as long as the Verilog-A option is available. To demonstrate the modeling concept, we first extract the related model parameters from **Figure. 1** and **Figure. 2**. The values of the model parameters are shown in **Table. 1**. In the next section, we will discuss the model in detail.

978-1-4244-1567-0/07/$25.00 © 2007 IEEE 159

Table. 1 The values of PCM parameters

Parameter	Value
Static resistance of Set (R_{Set})	7KΩ
Programming current of Set(I_{Set})	600μA
Programming time of Set (T_{Set})	100ns
Static resistance of Reset (R_{Resett})	200KΩ
Programming current of Reset (I_{Reset})	1200μA
Programming time of Reset (T_{Reset})	100ns
Holding voltage (V_h)	0.45V
Threshold voltage (V_{th})	0.78V
Dynamic-on resistance (R_{on})	1KΩ

3. MODELING USING VERILOG-A

We present a simple modeling technique accounting for only two stable states with the partial set and partial reset conditions neglected. The model consists of three modules: (1) PCMR module, (2) Decision module, and (3) Memory module, as shown in **Figure. 3**.

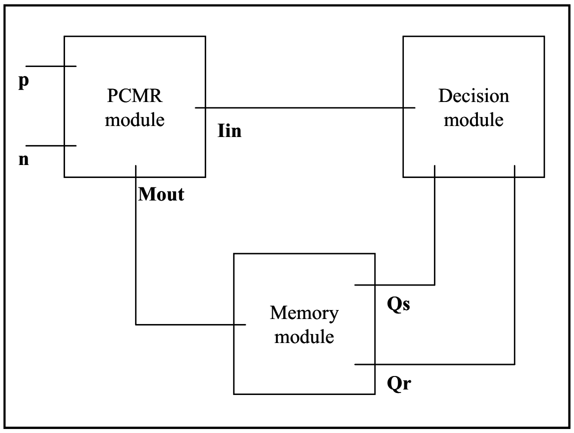

Figure. 3 The model block chart.

3.1. PCMR module

This module is composed of two independent I-V curves (**Figure. 4**), only one of which can be selected by the "Memory module" (section 3.3). Depending on the current pulse width and amplitude, the PCMR module will then respond to the next decision module.

3.1.1 Set curve

This curve combines two resistance states: R_{Set} and R_{on}. R_{Set} represents the resistance of the crystalline state and R_{on} represents the (ON) resistance during programming. We set the boundary at I_x. The (V_x, I_x) is an intersection of R_{Set} and R_{on} curves. When the input current pulse amplitude is higher than I_x, the resistance will be set to R_{on}. The equation for calculating R_{on} is based on the following relation:

$$V(I) = V_h + I \times R_{on}. \qquad (1)$$

3.1.2 Reset curve

Reset curve consists of three resistance states: R_{Reset}, R_{sn} and R_{on}. R_{Reset} represents the resistance of amorphous state and R_{sn} represents the negative resistance during snapback. We set two boundaries at I_{th} and I_x, respectively. The R_{sn} is calculated based on the linear relation between (V_{th}, I_{th}) and (V_x, I_x) as

$$V(I) = (I - I_{th}) \times \left(\frac{V_x - V_{th}}{I_x - I_{th}} \right) + V_{th}. \qquad (2)$$

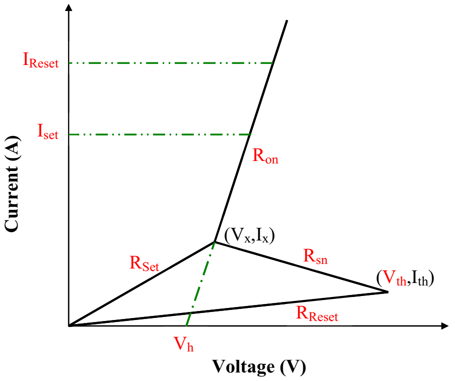

Figure. 4 The set and reset I-V curves of the PCMR module.

3.2. Decision module

When the decision module receives the memory element current pulse, it checks its width and amplitude. If the pulse achieves the condition for Set, a high voltage (high state) will appear at the node of Qs. On the other hand, the node of Qr will show a high voltage, when the pulse width and amplitude reach the reset condition. We utilize the current control resistance (CCR) to determine the pulse amplitude, and use an RC circuit to estimate the pulse width.

3.2.1 Current control resistance (CCR) module

Figure. 5 shows the module characteristic. The CCR module is implemented with a Fermi equation for numerical continuity, given by

$$\mathrm{R}(I) = \mathrm{BS} + \frac{\mathrm{AS} - \mathrm{BS}}{1 + exp[(\mathrm{I_s} - I) \times \mathrm{F}]}, \qquad (3)$$

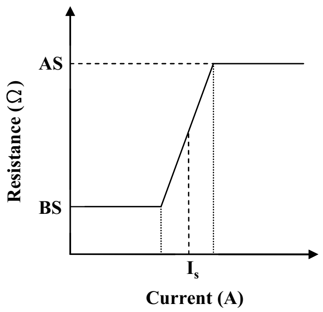

Figure. 5 R-I curves of CCR.

where $\mathrm{I_s}$ is the switch current, BS is the resistance for $\mathrm{I} < \mathrm{I_s}$, AS is the resistance for $\mathrm{I} > \mathrm{I_s}$, and F is the factor which controls the slope between BS and AS.

3.2.2 RC circuit model

We separate this sub-module in two parts for calculating the pulse width and amplitude because of the two different set and reset state criteria. The RC circuit module provides a simple mean to evaluate the pulse width via RC integral in time. **Figure. 6** shows the conceptual RC circuit; we use this idea to develop the decision module.

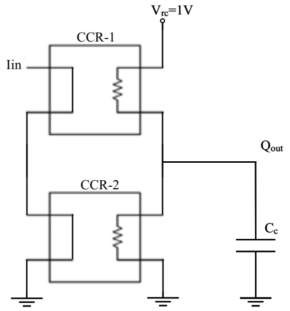

Figure. 6 RC circuit model with CCR.

The CCR-1 has two different resistances: $\mathrm{R_{rc}}$ and $\mathrm{R_{max}}$. The CCR-2 also has two different resistances: $\mathrm{R_{max}}$ and $\mathrm{R_{min}}$. Here, $\mathrm{R_{max}}$ is set to 1 GΩ, $\mathrm{R_{min}}$ is set to 1 mΩ, $\mathrm{R_{rc}}$ is set to 50 KΩ, and $\mathrm{V_{rc}}$ is set to 1 V. We use the RC equation for the module calculation as

$$\mathrm{V_{C_c}}(\mathrm{t}) = \mathrm{V_{rc}}\left[1 - exp\left(-\frac{\mathrm{t}}{\mathrm{R_{rc}C_c}}\right)\right]. \qquad (4)$$

We set the values of t in (5) using the model parameters $\mathrm{T_{Set}}$ and $\mathrm{T_{Reset}}$ for the set and reset state modules, respectively, and then the values of $\mathrm{C_c}$ can be determined based on the following relation with given t:

$$\mathrm{t} = \mathrm{R_{rc}C_c}\, ln\left[\frac{\mathrm{V_{rc}}}{\mathrm{V_{rc}} - \mathrm{V_{C_c}}(\mathrm{t})}\right]. \qquad (5)$$

When Iin (shown in **Figure. 6**) reaches the condition of $\mathrm{I_{Set}}$, the resistances of CCR-1 and CCR-2 are set to $\mathrm{R_{rc}}$ and $\mathrm{R_{max}}$, respectively, and the model begins the calculation of charging time. When the pulse width reaches the condition of $\mathrm{T_{Set}}/\mathrm{T_{Reset}}$, the voltage across $\mathrm{C_c}$ is changed to $\mathrm{V_{rc}}$ (1 V) and then this value will be delivered to the next memory module.

3.3. Memory module

The memory module must maintain the PCM state after the previous PCMR module determines the phase state. We utilize a capacitor to store the data. In addition, voltage control resistors (VCRs) are included, which is also implemented with a Fermi equation for numerical reason (as discussed in section 3.2.1). The equivalent circuit model of the memory module is illustrated in **Figure. 7**.

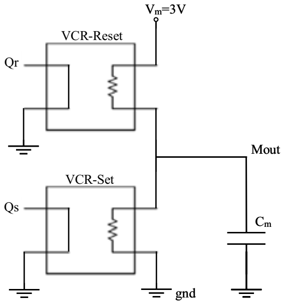

Figure. 7 Memory module equivalent circuit.

Qr and Qs control the VCR by varying the VCR resistance to charge or discharge the capacitor $\mathrm{C_m}$. When Qr is at high voltage, the VCR-Reset resistance becomes low ($\mathrm{R_{min}}$), VCR-Set resistance becomes high ($\mathrm{R_{max}}$), and $\mathrm{C_m}$ is charged to $\mathrm{V_m}$ (3 V). On the other hand, when Qs is at high voltage, the VCR-Set resistance becomes low, VCR-Reset resistance becomes high, and $\mathrm{C_m}$ is discharged. After charging or discharging, all VCRs will turn to high resistance to hold the charge of $\mathrm{C_m}$.

Another important feature of the model is that the initial condition of the PCM can be set by users. When calculating the operation point in simulation, the capacitance is assumed open. Therefore, the voltage across C_m is defined by a voltage divider composed of VCR-Reset and VCR-Set. We add another parameter IC to control the initial condition. The voltage of C_m (Mout) is given by

$$V_{C_m} = V(\mathrm{Mout}, \mathrm{gnd}) = \frac{R_{Max} + IC \times R_{Max}}{R_{Max}} \times V_m. \tag{6}$$

If IC = 1, the voltage at Mout is 2 V and the PCM model presents amorphous (Reset) state. Otherwise, the voltage at Mout is 1.5 V and the PCM model presents the crystalline (Set) state.

4. RESULTS

We have developed the PCM model using Verilog-A, parameters of which can be set by users, as shown **Table. 1**. The proposed model using Verilog-A is verified with HSPICE. There are two approaches to test the model: DC analysis and transient analysis.

4.1. DC analysis

We scan the PCM model by sweeping current from 0 A to 1 mA for 0.1 µA step. The reset and set states are tested respectively. The test results are shown in **Figure. 8** where the simulation results are compared with real device data [4]. Though the model does not match data exactly everywhere, it shows a good agreement with data in the low and high current regimes for practical read and write operations, respectively.

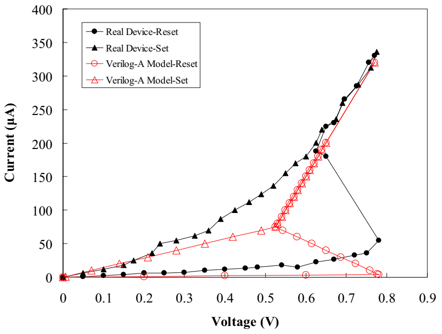

Figure. 8 DC analysis and comparison.

4.2. Transient analysis

We input a write/read current pulse to the PCM, and measure the resistance calculated by the model, as shown **Figure. 9**. The first programming current pulse is for the reset state of 1200 µA/100 ns. The second programming current pulse is for the set state of 600 µA/100 ns. Outside the programming current pulses, it is the read current of 0.1µA. The initial condition of the PCM model is assumed to be crystalline state (7KΩ). Resistance from the PCM model varies from crystalline value to amorphous one. After the set programming pulse, the resistance returns to the crystalline value. When the pulse is programming the PCM cell, the resistance from the model is set to R_{on}.

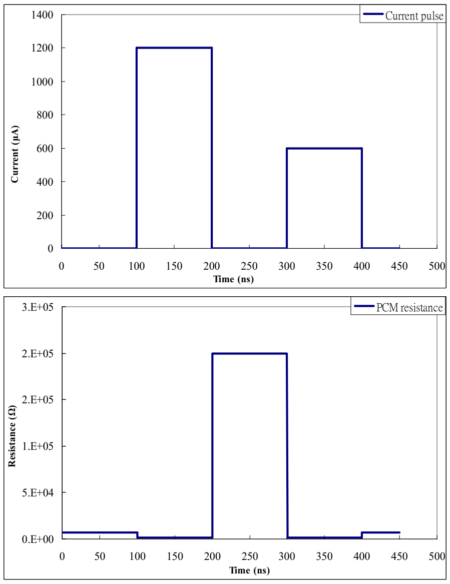

Figure. 9 Transient analysis.

4.3. R-I curve comparison

Figure. 10 shows the relationship between the PCM resistance and the programming current. The model maintains the initial state until the current pulse goes beyond I_{Set} (600 µA). Above I_{Reset} (1200 µA), the model changes the state to an amorphous one. The inconsistency between model and data is mainly due to the ignored partial set and partial reset conditions in the model.

4.4. Practical model usage and limitations

To ensure the model accuracy and predictivity, the model parameters, listed in Table 1, have to be calibrated first. Arbitrary setting of model parameters could cause a simulation failure. For instance, the calculated V_x, based on the given model parameters, must be smaller than V_{th}, as indicated in **Figure. 4**.

5. CONCLUSIONS

We have presented a compact PCM model using Verilog-A. The model is predictive, yet simple to use and more importantly the model can be implemented in any circuit simulators with Verilog-A option.

ACKNOWLEDGEMENTS

The authors would like to thank the Electronics and Opto-Electronics Research Laboratories of the Industrial Technology Research Institute for technical support and National Science Council of Taiwan, R.O.C. (under contract NSC-95-2221-E-197- 021) for financial support.

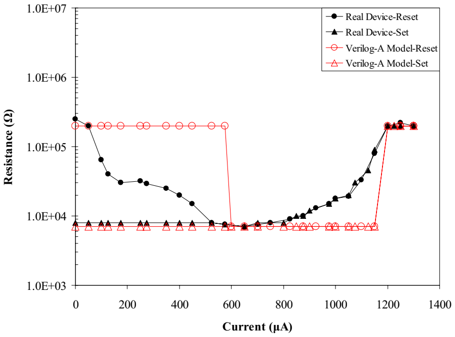

Figure. 10 The R-I curve comparison with real device data.

6. REFERENCES

[1] G. Atwood and R. Bez, "Current status of Chalcogenide phase change memory," *Device Research Conference Tech. Dig.*, Vol. 1, 2005, pp. 29-33.

[2] A. L. Lacaita, "Physics and Performance of Phase Change Memories," *Simulation of Semiconductor Processes and Devices Tech. Dig.*, 2005, pp. 267-270.

[3] M. Gh. Mohammad, L. Terkawi, and M. Albasman, "Phase Change Memory Faults," *Proceedings of the 19th International Conference on VLSI Design*, 2006.

[4] D. Ielmini, A. L. Lacaita, A. Pirovano, F. Pellizzer, and R. Bez, "Analysis of phase distribution in phase-change nonvolatile memories," *IEEE Electron Device Letters*, Vol. 25, pp. 507-509, July 2004.

[5] HSPICE user's manual version 2006.03.

APPENDIX: Verilog-A Models

Listing 1. PCMR module.

```
module PCMR(p,n,Iin,Mout);
input p,n,Mout;
output Iin;
electrical p,n,Iin,Mout,gnd;
ground gnd;
parameter real  Rset=7K from[0:inf);
parameter real  Rreset=200k from[0:inf);
parameter real  Ron=0.5k from[0:inf);
parameter real  Vh=0.45 from[0:inf);
parameter real  Vth=0.8 from[0:inf);
real   Ix,Vx,Ith;

analog begin

     I(Iin) <+ I(n,p);
     Vx=(Vh*Rset)/(Rset-Ron);
     Ix=Vh/(Rset-Ron);
     Ith=Vth/Rreset;
//Set curve
If ( V(Mout)<=1.75 ) begin
     If ( I(p,n)<Ix ) begin
          V(p,n) <+ I(p,n)*Rset;
     end
     else begin
          V(p,n) <+ Vh+I(p,n)*Ron;
     end
end
//Reset curve
else begin
     if ( I(p,n) <= Ith ) begin
          V(p,n) <+ I(p,n)*Rreset;
     end
//Rsn curve
     else  if ( I(p,n)>Ith && I(p,n)<Ix ) begin
          V(p,n) <+ (I(p,n)-Ith)*((Vx-Vth)/(Ix-Ith))+Vth;
     end
     else  begin
          V(p,n) <+ Vh+I(p,n)*Ron;
     end
end
end
endmodule
```

Listing 2. CCR module.

```
module CCR(In_p,In_n,Out_p,Out_n);
input  In_p,In_n;
output Out_p,Out_n;
electrical In_p,In_n,Out_p,Out_n;
parameter real BS=1e+9 from[0:inf);
parameter real AS=1e-3 from[0:inf);
parameter real Is=6e-4 from[0:inf);
real    SW,SW_Max,Rsw;
real    X_BS,Y_BS,X_AS,Y_AS;
real    delta_I,percent,F;

analog begin

        F=1e+3*(1/Is);
        percent=1e-10;
if (AS>BS)  begin
        Y_BS=BS+((AS-BS)*percent);
        Y_AS=AS-((AS-BS)*percent);
        X_BS=Is-((1/F)*(ln((AS-Y_BS)/(Y_BS-BS))));
        X_AS=Is-((1/F)*(ln((AS-Y_AS)/(Y_AS-BS))));
        delta_I=(X_AS-X_BS)/2;
        SW=(Is-delta_I-I(In_p,In_n))*F;         //Left Shift
end
else if (AS<BS) begin
        Y_BS=BS-((BS-AS)*percent);
        Y_AS=AS+((BS-AS)*percent);
        X_BS=Is-((1/F)*(ln((AS-Y_BS)/(Y_BS-BS))));
        X_AS=Is-((1/F)*(ln((AS-Y_AS)/(Y_AS-BS))));
        delta_I=(X_BS-X_AS)/2;
        SW=(Is+delta_I-I(In_p,In_n))*F;         //Right Shift
end
        SW_Max=50;
if (SW<SW_Max) begin
        Rsw=BS+((AS-BS)/(1+exp(SW)));
end
else begin
        Rsw=BS+((AS-BS)/(1+exp(SW_Max)));
end
        V(In_p,In_n) <+ 0;
        V(Out_p,Out_n) <+ I(Out_p,Out_n)*Rsw;
end
endmodule
```

Listing 3. RC module.

```
module RC(Iin,Qout);
inout Iin,Qout;
electrical Iin,Iin1,Qout,rc;
ground gnd;
parameter    real Trc=100e-9 from[0:inf);
parameter    real Irc=6e-4 from[0:inf);
real Rmax,Rmin,Rrc,Cc,Vrc,;

analog begin

        Rmax=1e+9;
        Rmin=1e-3;
        Vrc=1;
        Rrc=50k;
        Cc=Trc/(Rrc*ln((Vrc/(Vrc-(Vrc*0.99)))));
end

Vrc    #(.Vrc(Vrc))  Vrc(rc,gnd);
CCR   #(.BS(Rmax), .AS(Rrc), .Is(Irc))
CCR1(Iin,Iin1,rc,Qout);

CCR   #(.BS(Rmin), .AS(Rmax), .Is(Irc))
CCR2(Iin1,gnd,Qout,gnd);

Cap    #(.C(Cc))                Cc(Qout,gnd);
endmodule
```

Listing 4. Memory module.

```
module Memory(QS,QR,Mem);
inout QS,QR,Mem;
electrical QS,QR,Mem,m,gnd;
ground gnd;
parameter real Vs=3 from[0:inf);
parameter real IC=0 from[0:inf);
real Rmax,Rmin,R1,R2;

analog    begin

        Rmax=1e+9;
        Rmin=1e-3;
        R1=Rmax;
        R2=Rmax+IC*Rmax;
end

Vrc    #(.Vrc(3))
Vrc(rc,gnd);

VCR  #(.BV(R1),.AV(Rmin),.Vs(Vs))
Msw1(QR,gnd,m,Mem);

VCR  #(.BV(R2),.AV(Rmin),.Vs(Vs))
Msw2(QS,gnd,Mem,gnd);

Cap   #(.C(20p))
Cm(Mem.gnd);

endmodule
```