

Memristor-Based IMPLY Logic Gate Design Procedure

Shahar Kvatinsky, Eby G. Friedman,^{*}
Avinoam Kolodny, and Uri C. Weiser

Technion – Israel Institute of Technology

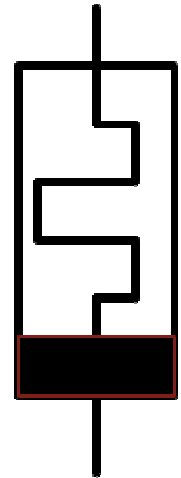
^{*}University of Rochester

ICCD 2011



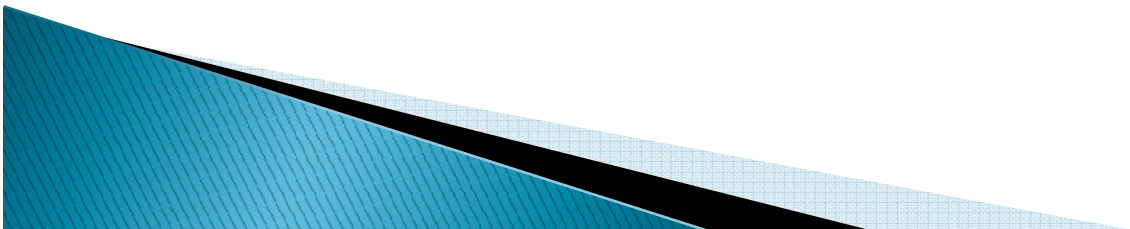
Logic Design with Memristors

- ▶ Our contributions:
 - Methodology for logic design with memristors
 - Performance – robustness tradeoff
 - Simplified memristor models
 - Demonstration that the widely used memristor model is impractical

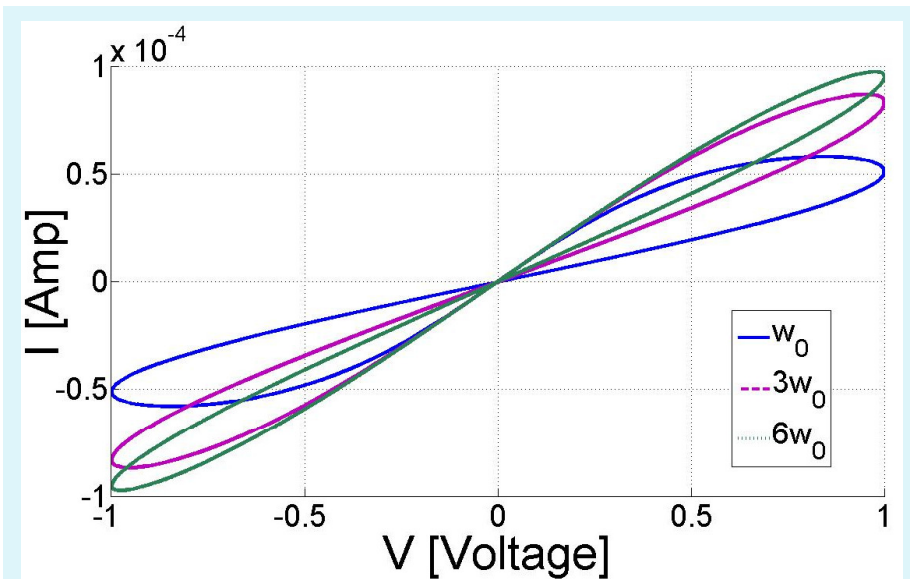
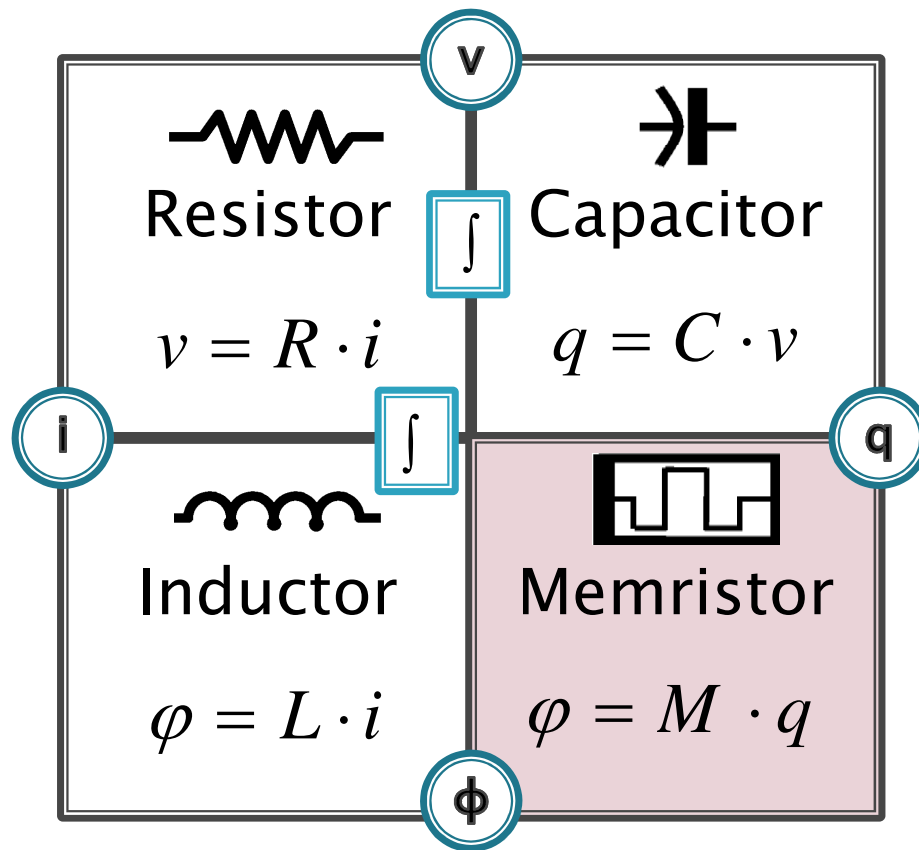


Outline

- ▶ **Memristors**
- ▶ IMPLY logic Gate
- ▶ Design procedure
- ▶ Conclusions and future work



Memristors

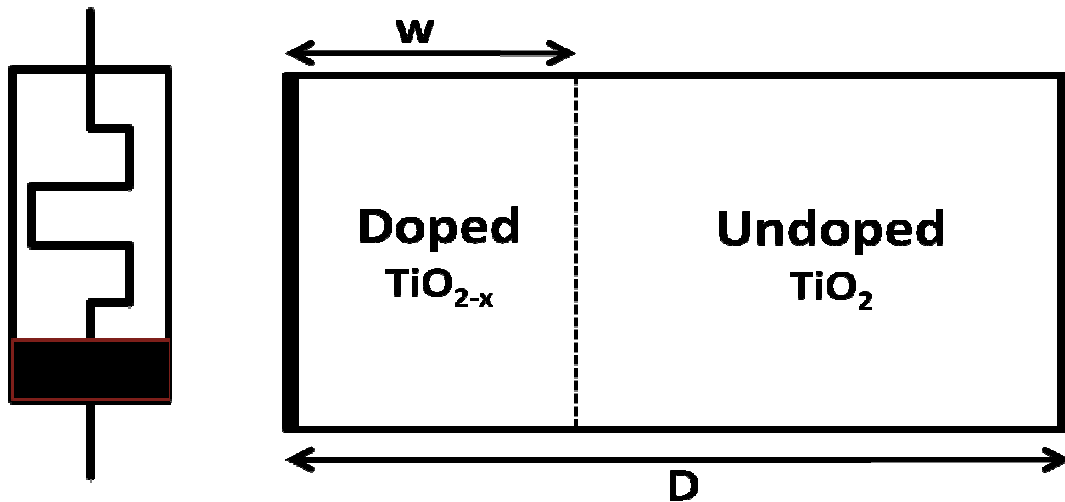


$$v = M(x, i) \cdot i$$

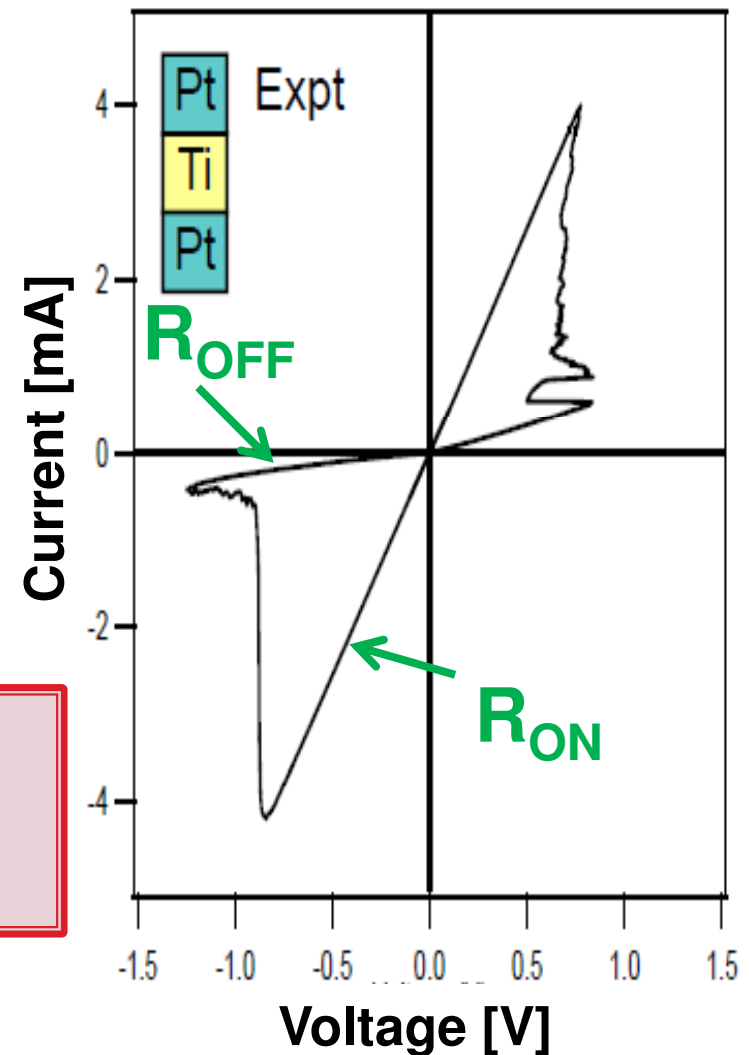
$$\frac{dx}{dt} = f(x, i)$$

Practical Memristors

► 2008 Hewlett Packard

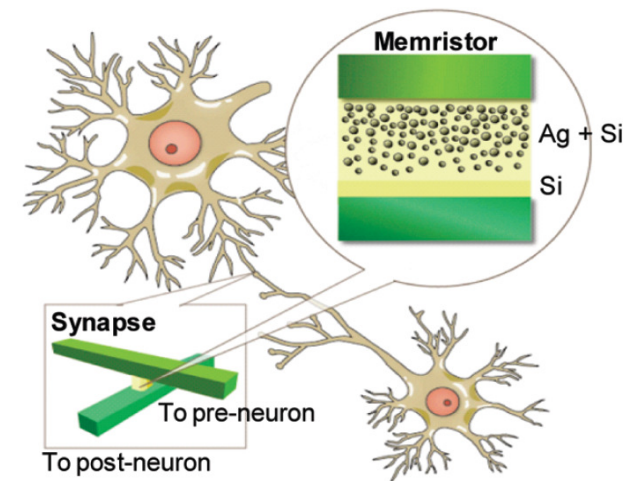
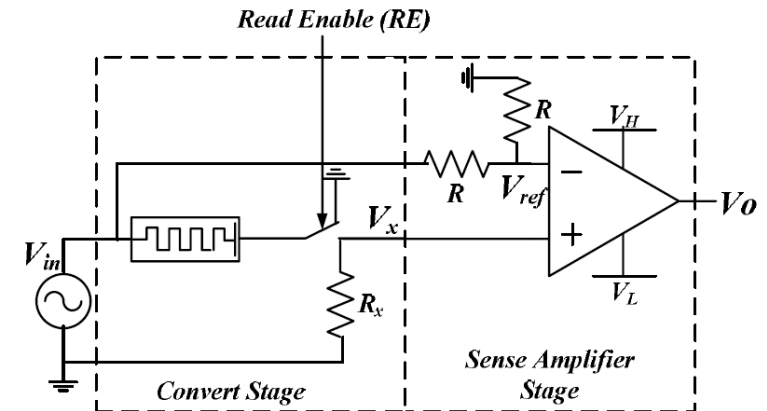


$$M(q) = R_{OFF} \left(1 - \frac{\mu_v R_{ON}}{D^2} q(t) \right)$$



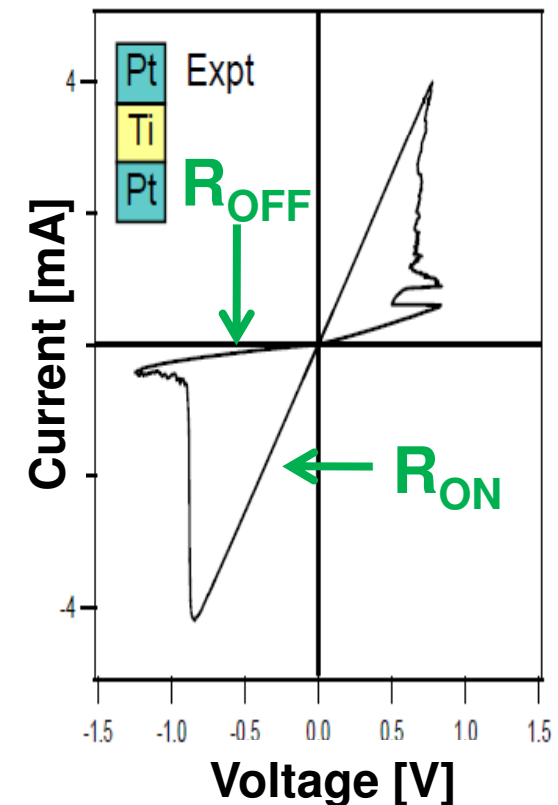
Memristor-Based Applications

- ▶ Memory
- ▶ Analog circuits
- ▶ Neuromorphic systems
- ▶ Logic



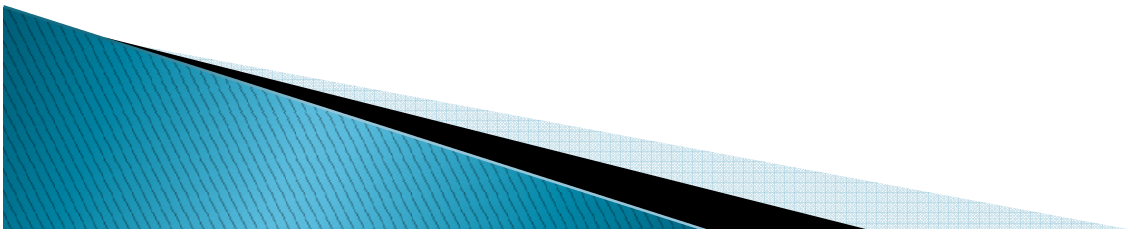
Logic with Memristors

- ▶ $R_{ON} \rightarrow$ logic '1', $R_{OFF} \rightarrow$ logic '0'
- ▶ Memristors as the primary building block
- ▶ Memristor can be:
 - Input
 - Output
 - Computational logic element
 - Latch



Outline

- ▶ Memristors
- ▶ **IMPLY logic gate**
- ▶ Design procedure
- ▶ Conclusions and future work

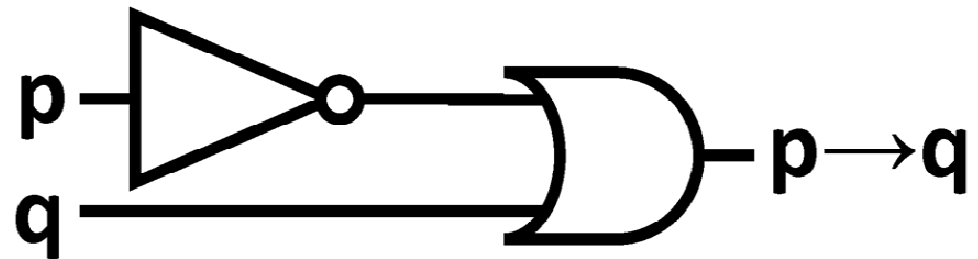



IMPLY Logic Gate

p	q	$p \text{ IMP } q$
0	0	1
0	1	1
1	0	0
1	1	1

$$p \rightarrow q$$

If p then q



IMPLY + FALSE  **Complete logic**

IMPLY Logic with Memristors

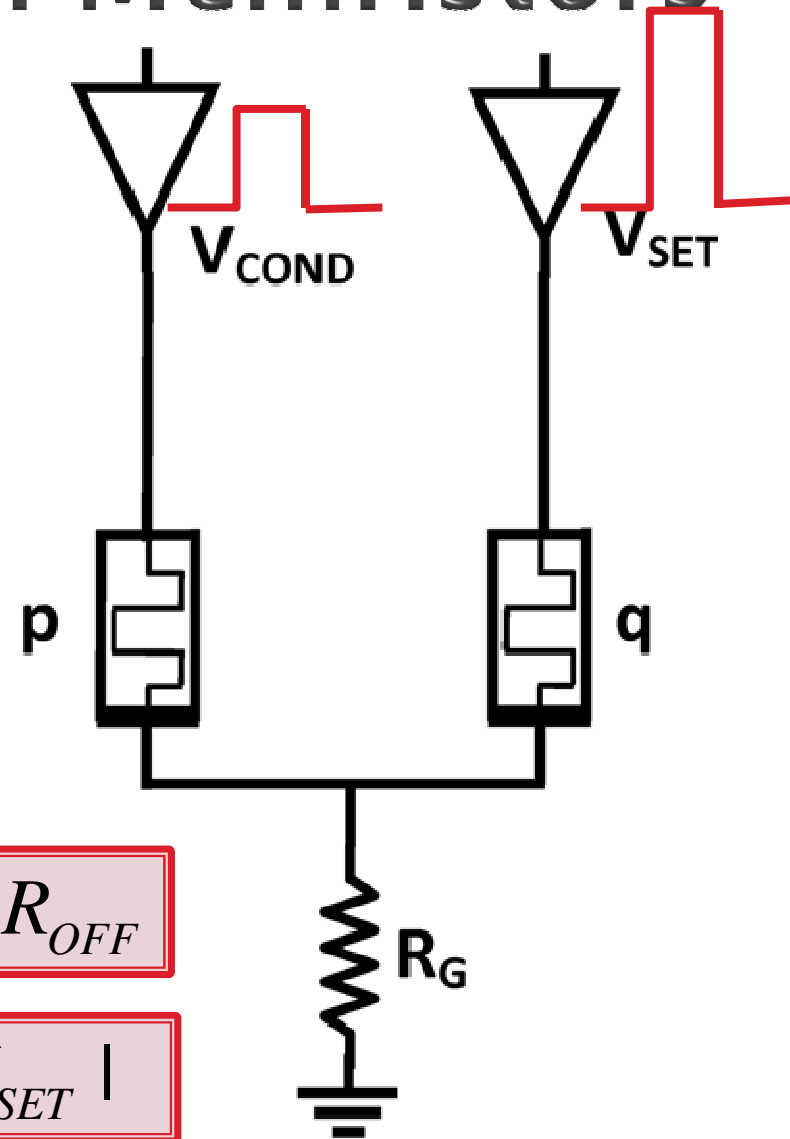
Logic 0 $\rightarrow R_{OFF}$

Logic 1 $\rightarrow R_{ON}$

p	q	$p \text{ IMP } q$
0	0	1
0	1	1
1	0	0
1	1	1

$$R_{ON} < R_G < R_{OFF}$$

$$|V_{COND}| < |V_{SET}|$$



IMPLY Logic with Memristors

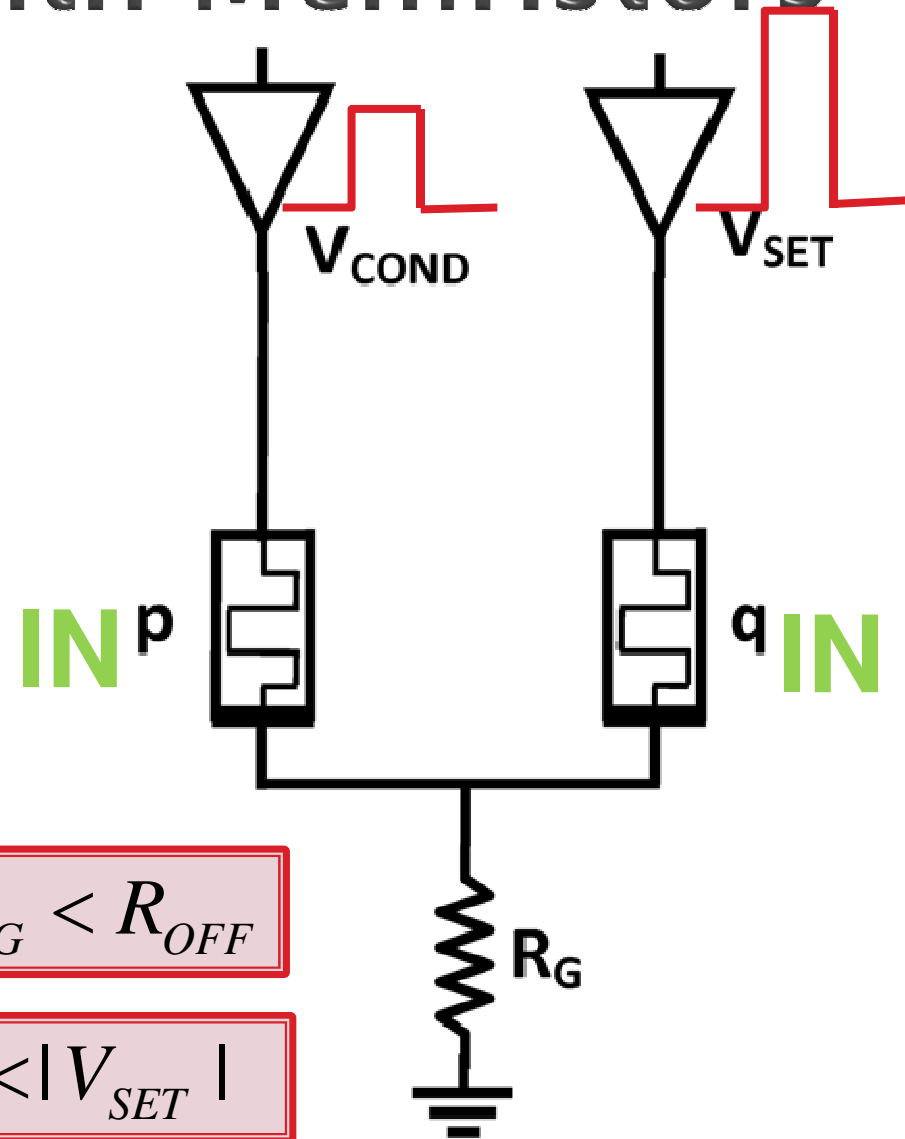
Logic 0 $\rightarrow R_{OFF}$

Logic 1 $\rightarrow R_{ON}$

p	q	$p \text{ IMP } q$
0	0	1
0	1	1
1	0	0
1	1	1

$$R_{ON} < R_G < R_{OFF}$$

$$|V_{COND}| < |V_{SET}|$$



IMPLY Logic with Memristors

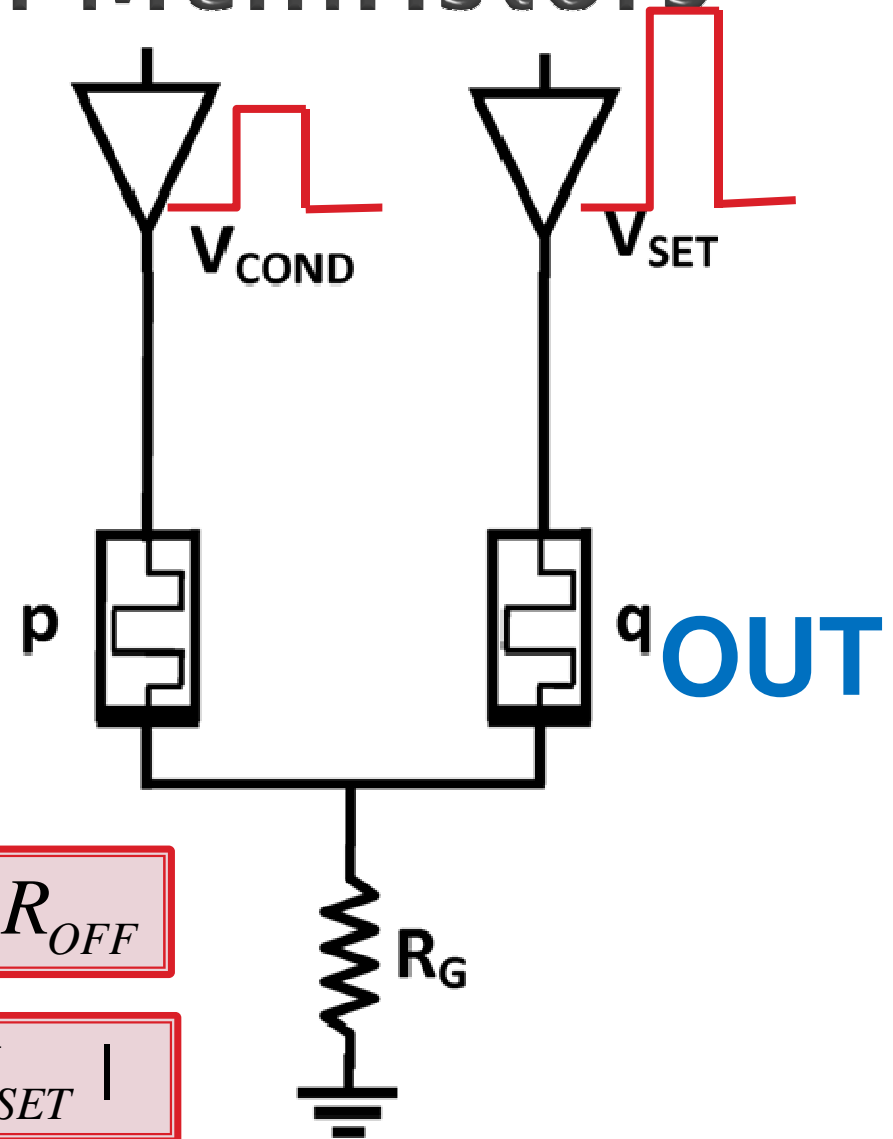
Logic 0 $\rightarrow R_{OFF}$

Logic 1 $\rightarrow R_{ON}$

p	q	$p \text{ IMP } q$
0	0	1
0	1	1
1	0	0
1	1	1

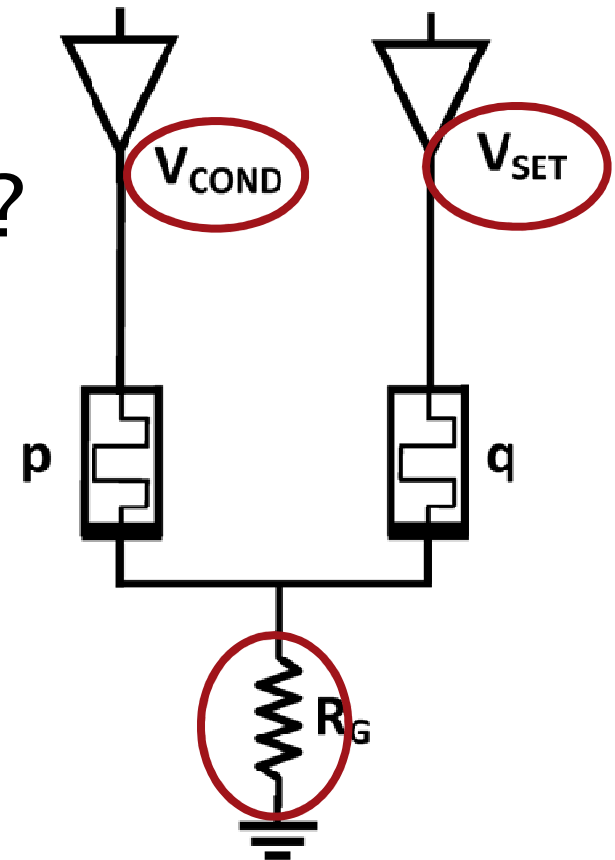
$$R_{ON} < R_G < R_{OFF}$$

$$|V_{COND}| < |V_{SET}|$$



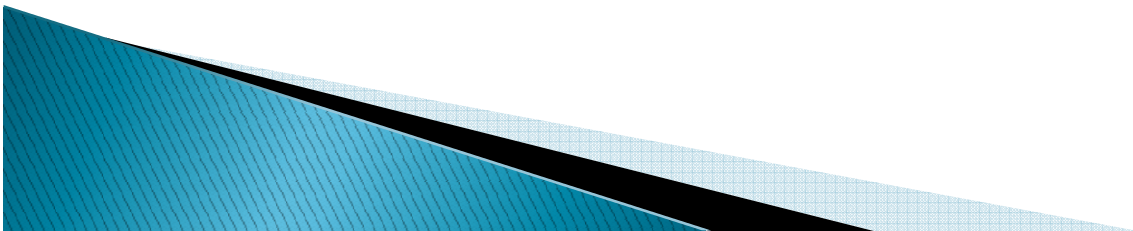
Need Design Methodology

- ▶ Determine proper circuit parameters
 - R_G ?
 - Voltage levels? V_{COND} ? V_{SET} ?
 - Logic gate delay?

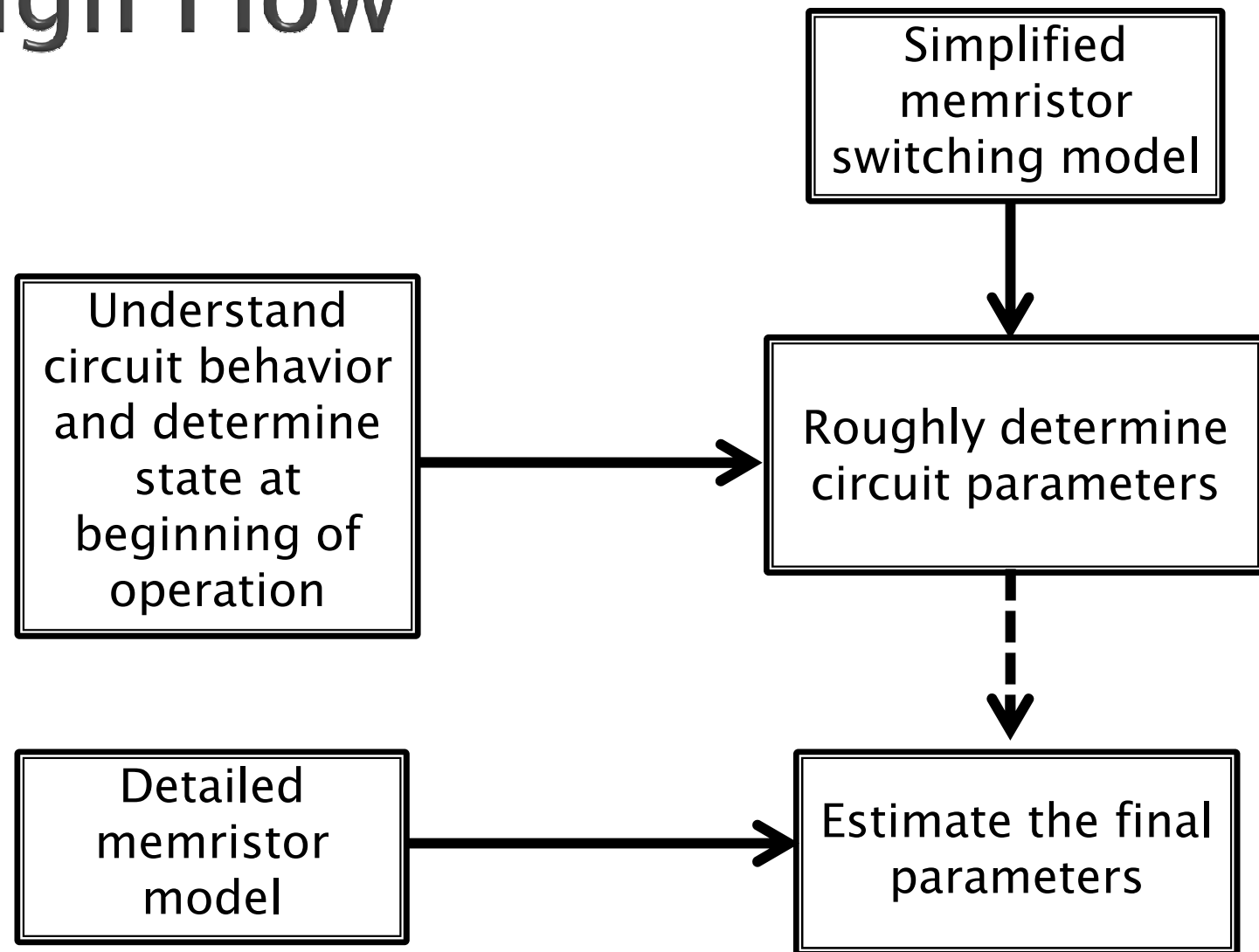


Outline

- ▶ Memristors
- ▶ IMPLY logic gate
- ▶ **Design procedure**
- ▶ Conclusions and future work



Design Flow



Behavior for Different Input Cases

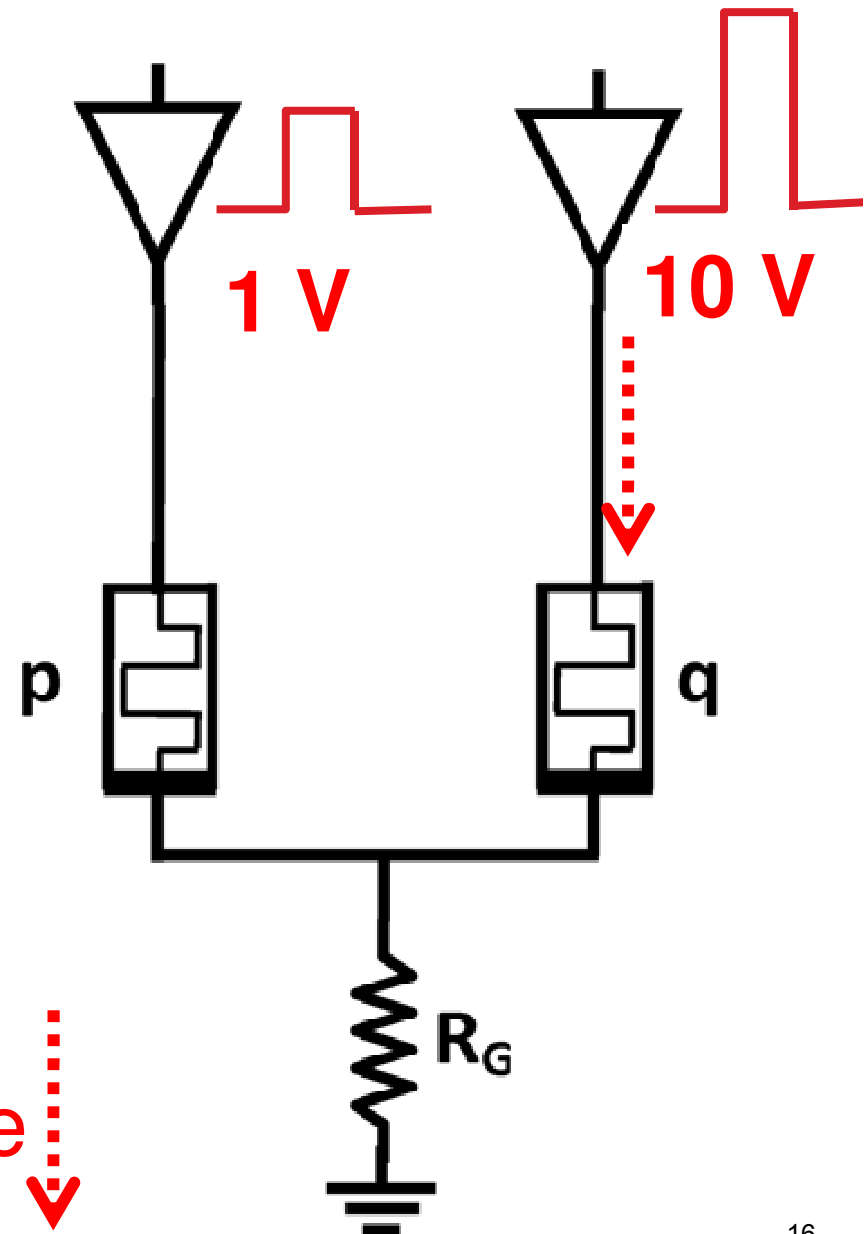
$$R_{ON} = 1 \Omega$$

$$R_{OFF} = 1 M\Omega$$

Case	p	q	$p \text{ IMP } q \rightarrow q$
1	0	0	1
2	0	1	1
3	1	0	0
4	1	1	1

IN OUT

Lower resistance



Performance and Robustness Tradeoff

Case	p	q	$p \text{ IMP } q \rightarrow q$
1	0	0	1
2	0	1	1
3	1	0	0
4	1	1	1

Write time



TRADEOFF

State drift



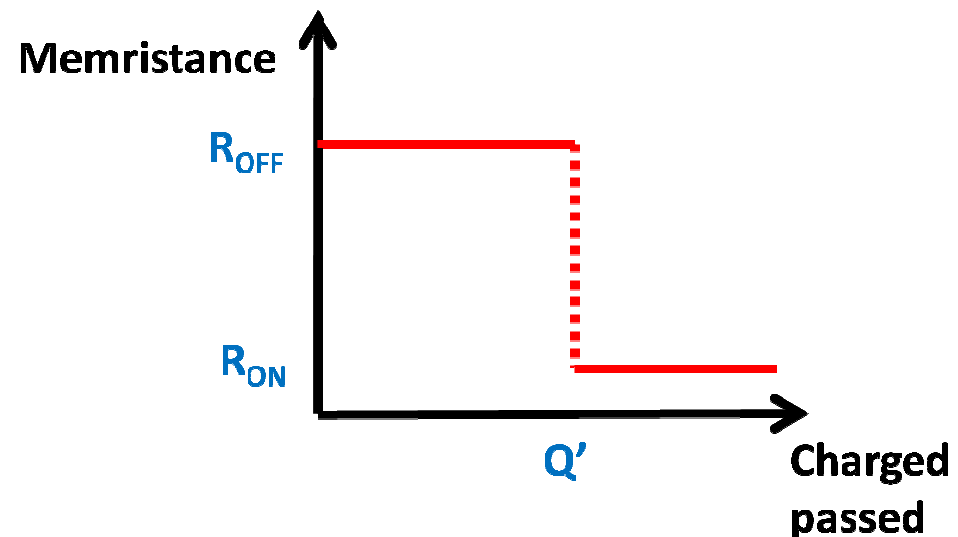
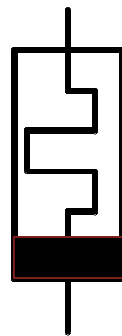
IN OUT Refreshing the gate



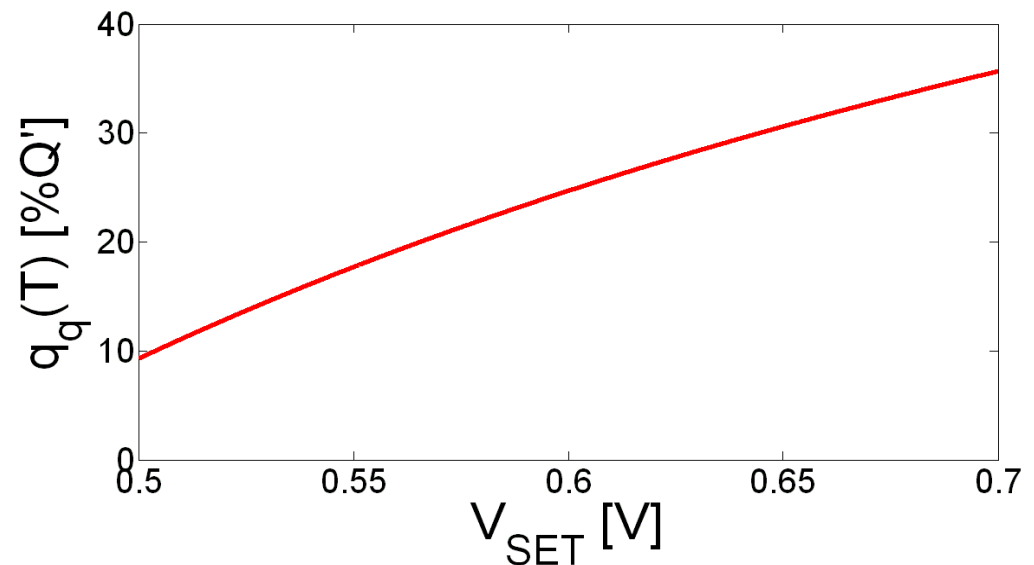
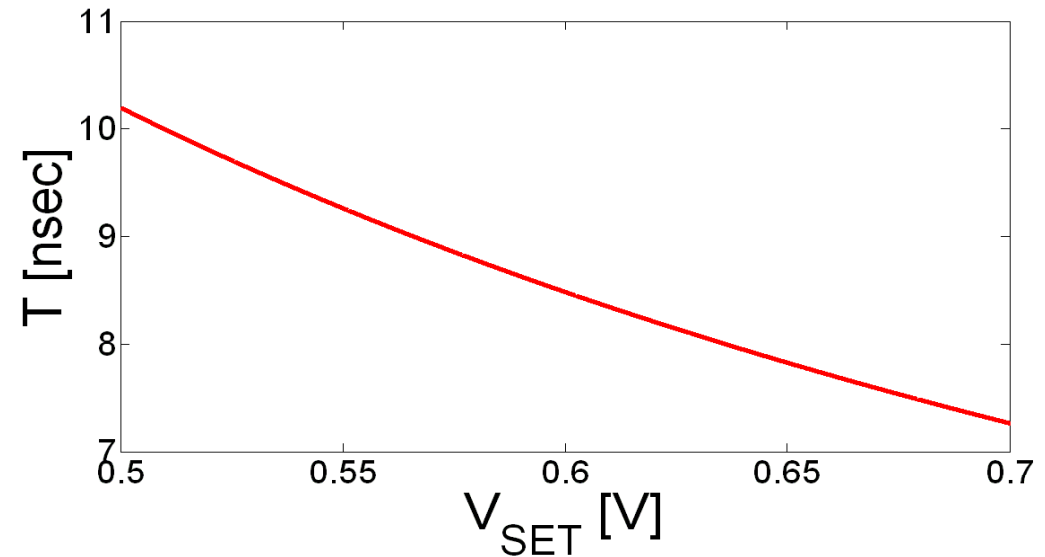
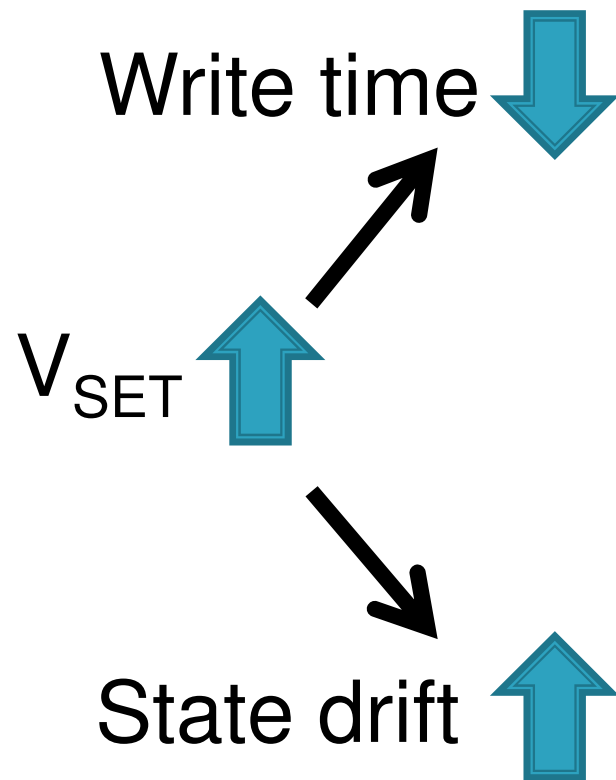
Binary Memristance Model

- For computing write time and state drift
- Two memristance states – R_{ON} and R_{OFF}
- Charge of Q' must flow to switch from

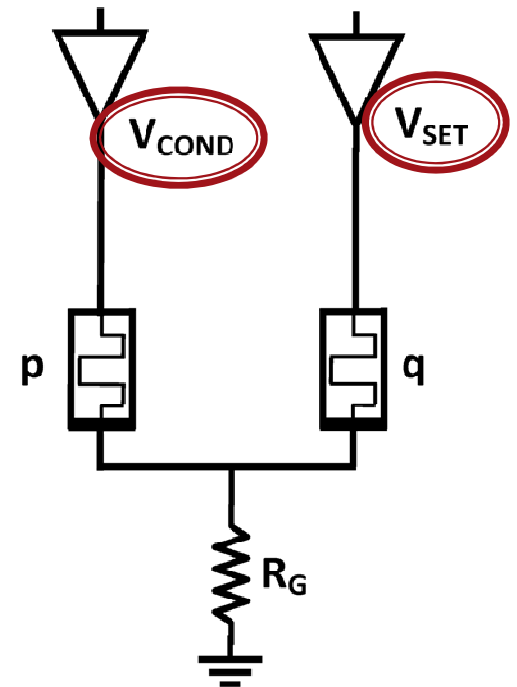
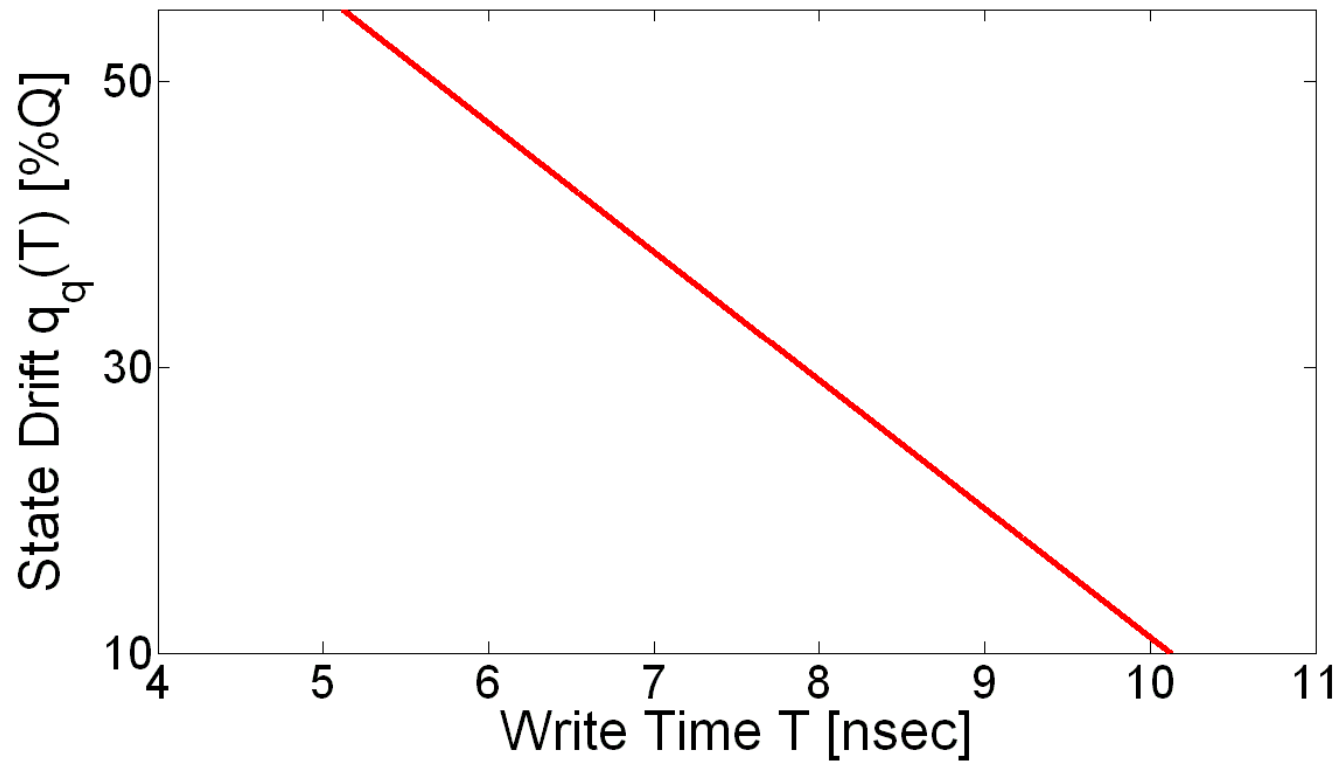
R_{OFF} to R_{ON}



Write Time and State Drift



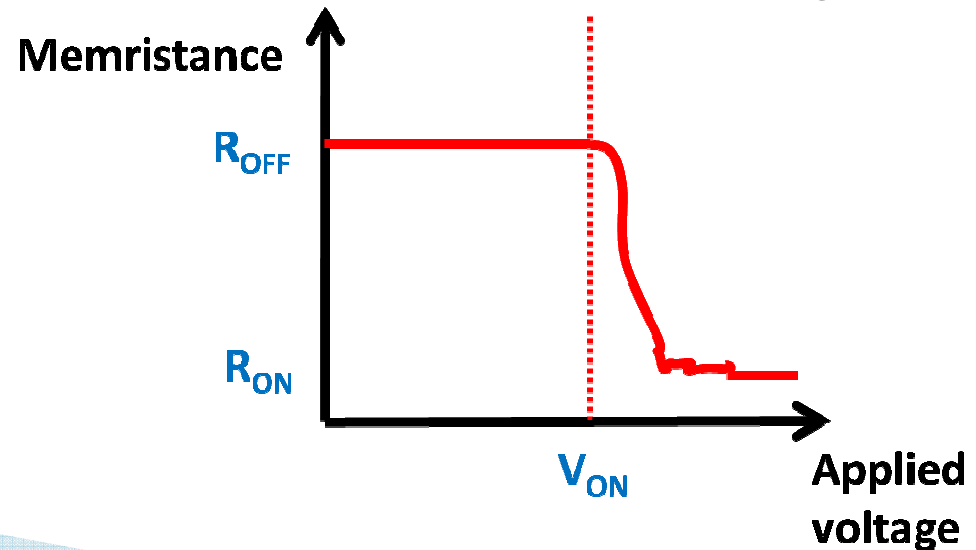
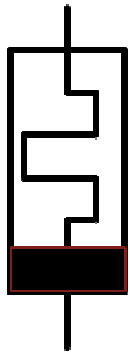
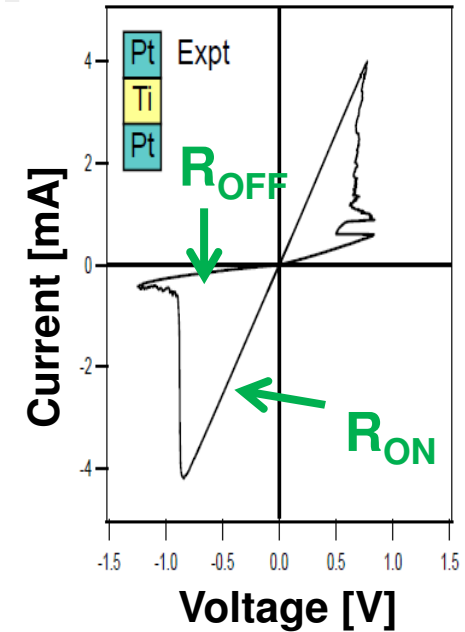
Performance and Robustness Tradeoff



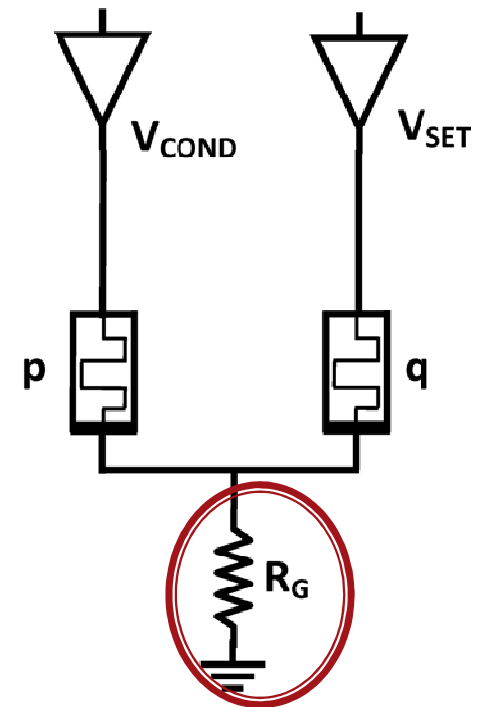
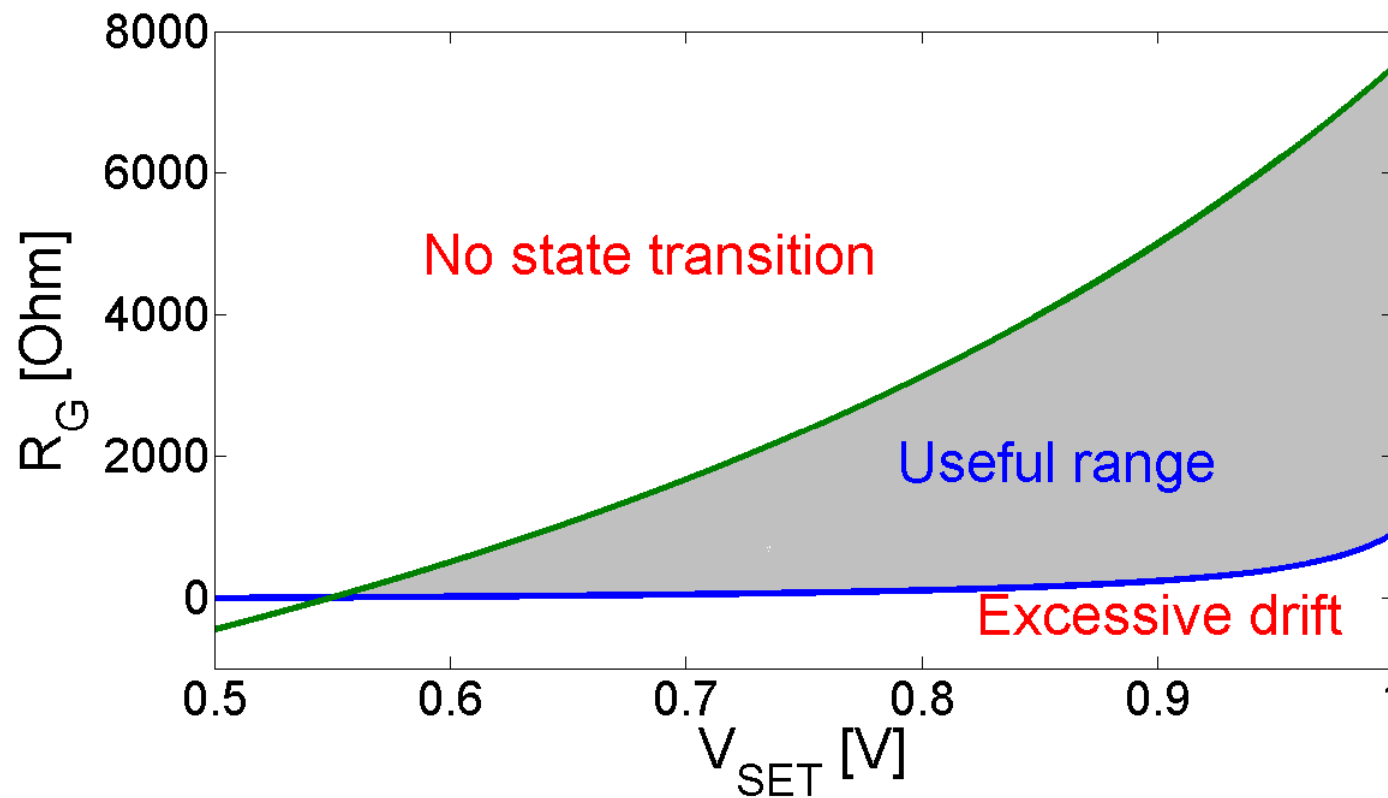
Roughly determine voltage levels

Fixed Threshold Model

- ▶ For choosing the value of R_G
- ▶ Nonlinear behavior
- ▶ Fixed threshold voltage V_{ON}



R_G Design Constraints

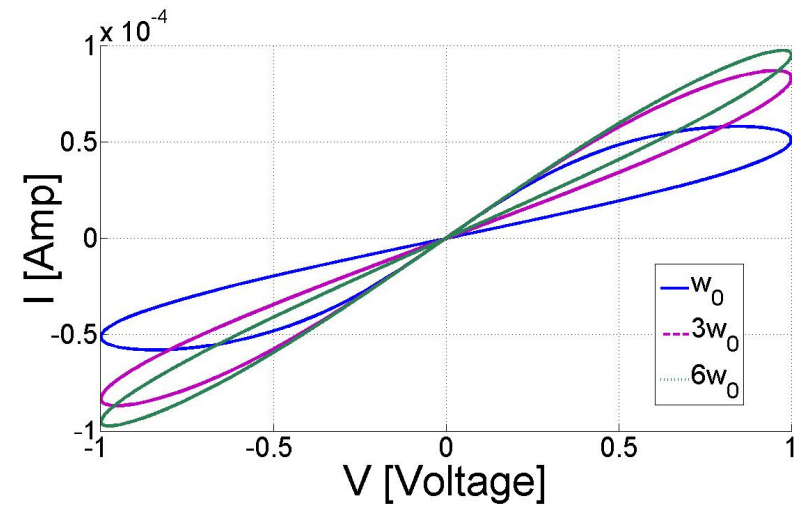
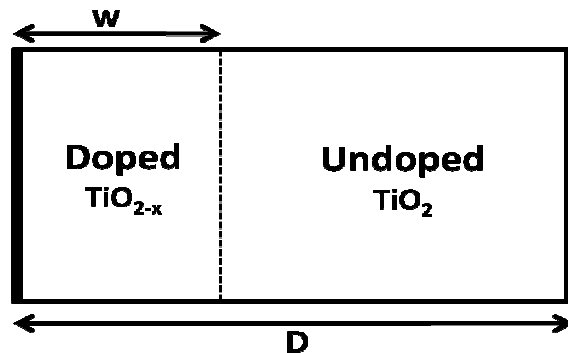


Roughly determine R_G

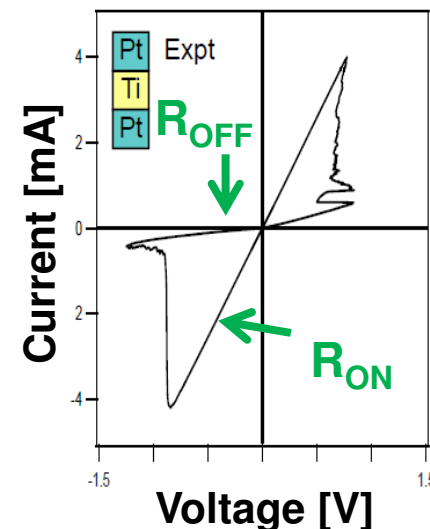
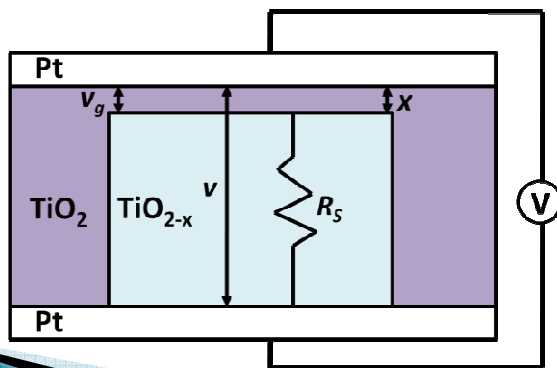


Detailed Memristor Models

▶ Linear ion drift model



▶ Nonlinear ion drift with threshold model

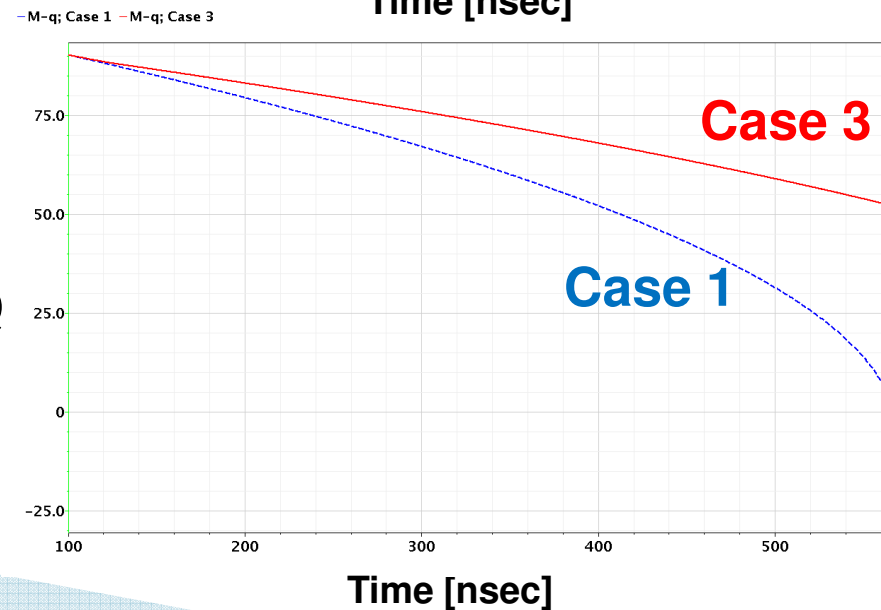
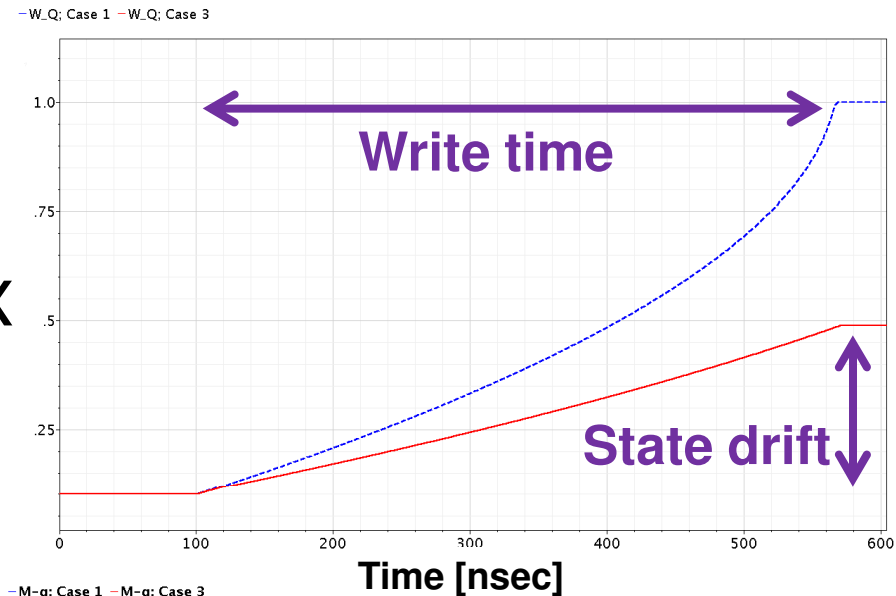


Linear Ion Drift Model

State variable - x

Case	p	q	$p \text{ IMP } q$
1	0	0	1
3	1	0	0

Memristance M_Q

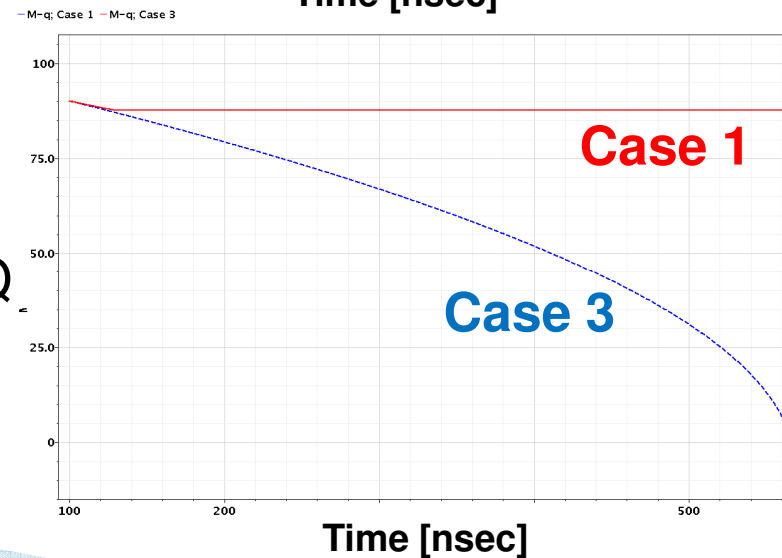


Nonlinear Ion Drift with Threshold Model

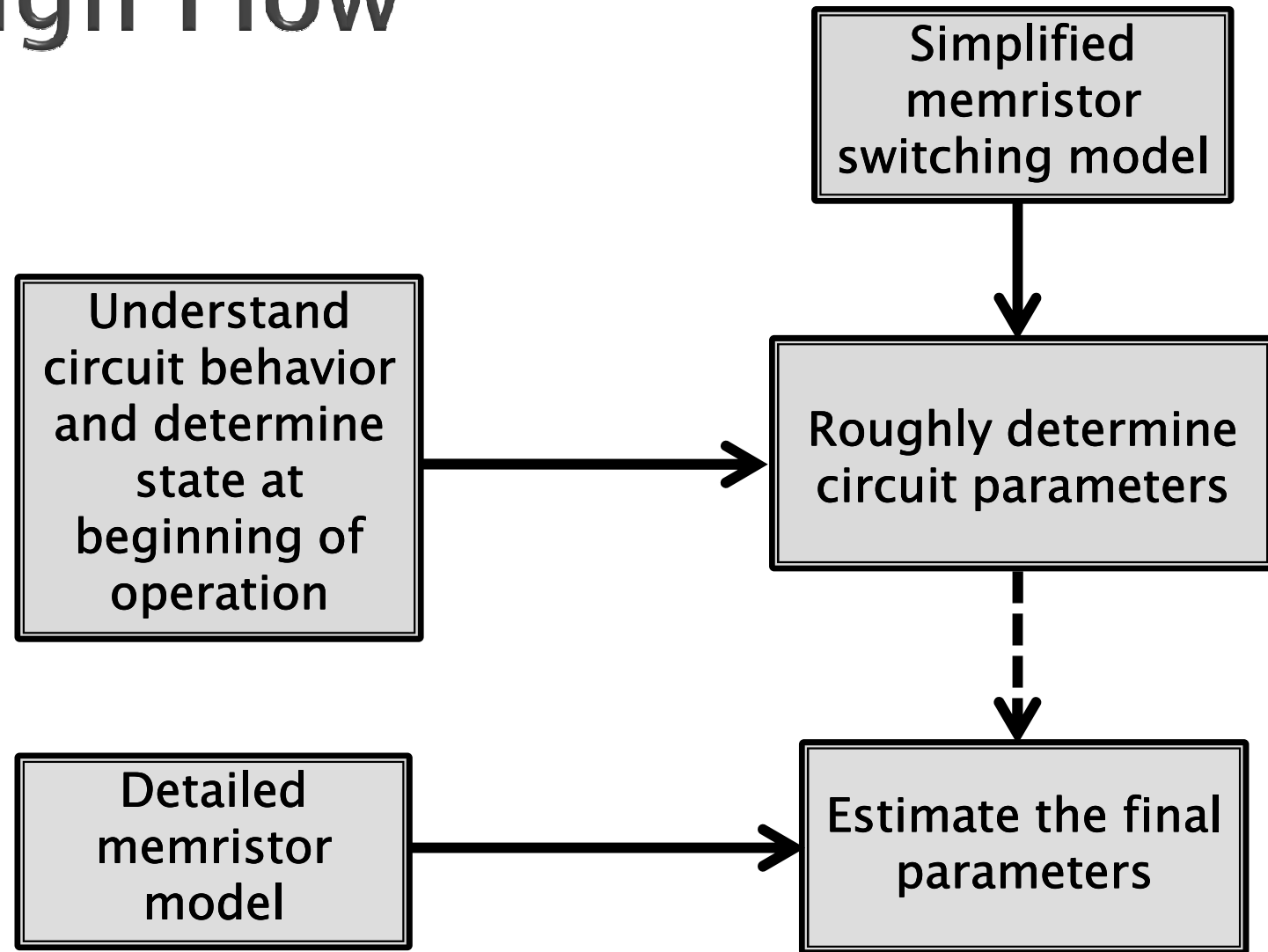
State variable x



Memristance M_Q

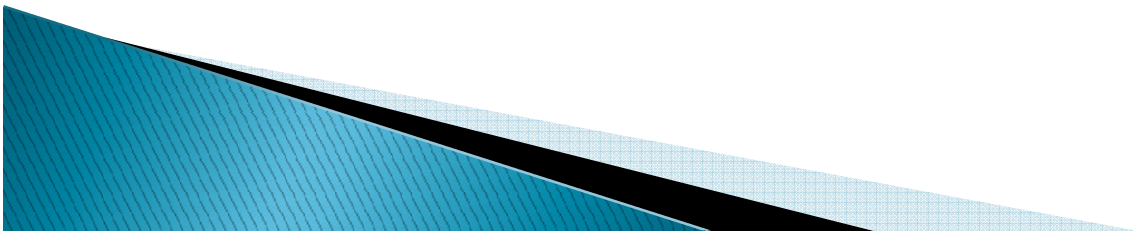


Design Flow



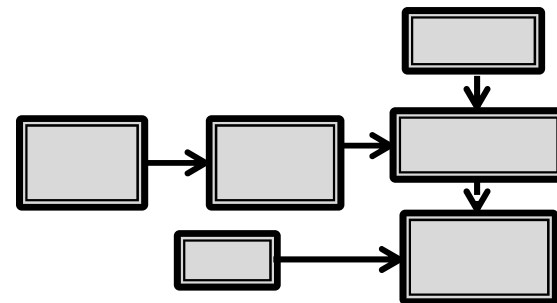
Outline

- ▶ Memristors
- ▶ IMPLY logic gate
- ▶ Design procedure
- ▶ **Conclusion and future work**



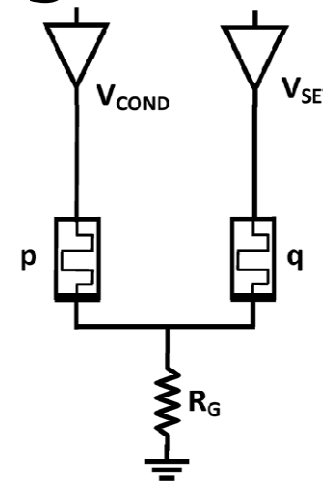
Summary – Logic Design with Memristors

- ▶ IMPLY logic gate was presented and analyzed
- ▶ We offered **procedures** for designing IMPLY logic gates with memristors and organized it into a **design flow**



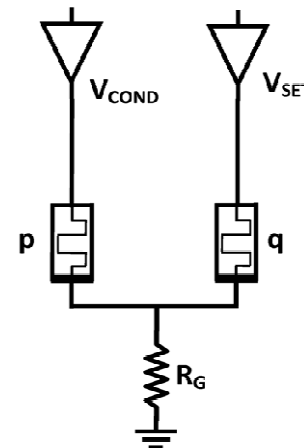
Conclusions

- ▶ Performance and robustness tradeoff
- ▶ Need for refresh because of state drift
- ▶ Widely used linear ion drift memristor model is **incompatible** for logic design



Conclusions – Beyond This Talk

- ▶ More in the paper – a detailed design example
- ▶ Future work – complete design methodology for memristor-based logic



Thanks!

<http://memristor.shorturl.com>

