Modeling a 4-Level DC-DC Converter Using EEnets

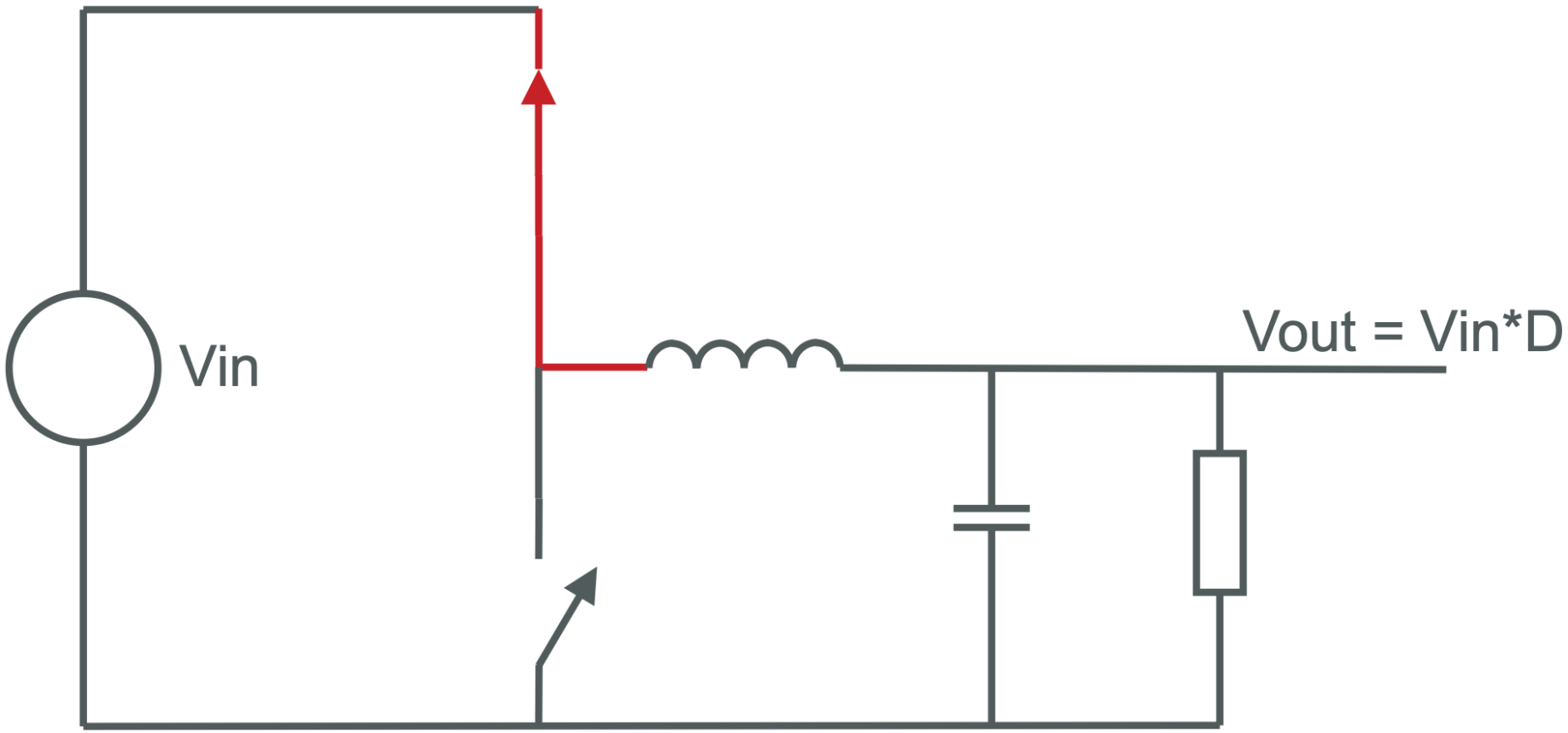
Abhijit Madhu Kumar (on behalf of Paul Denny, PSemi)
Solutions Engineer

cadence®

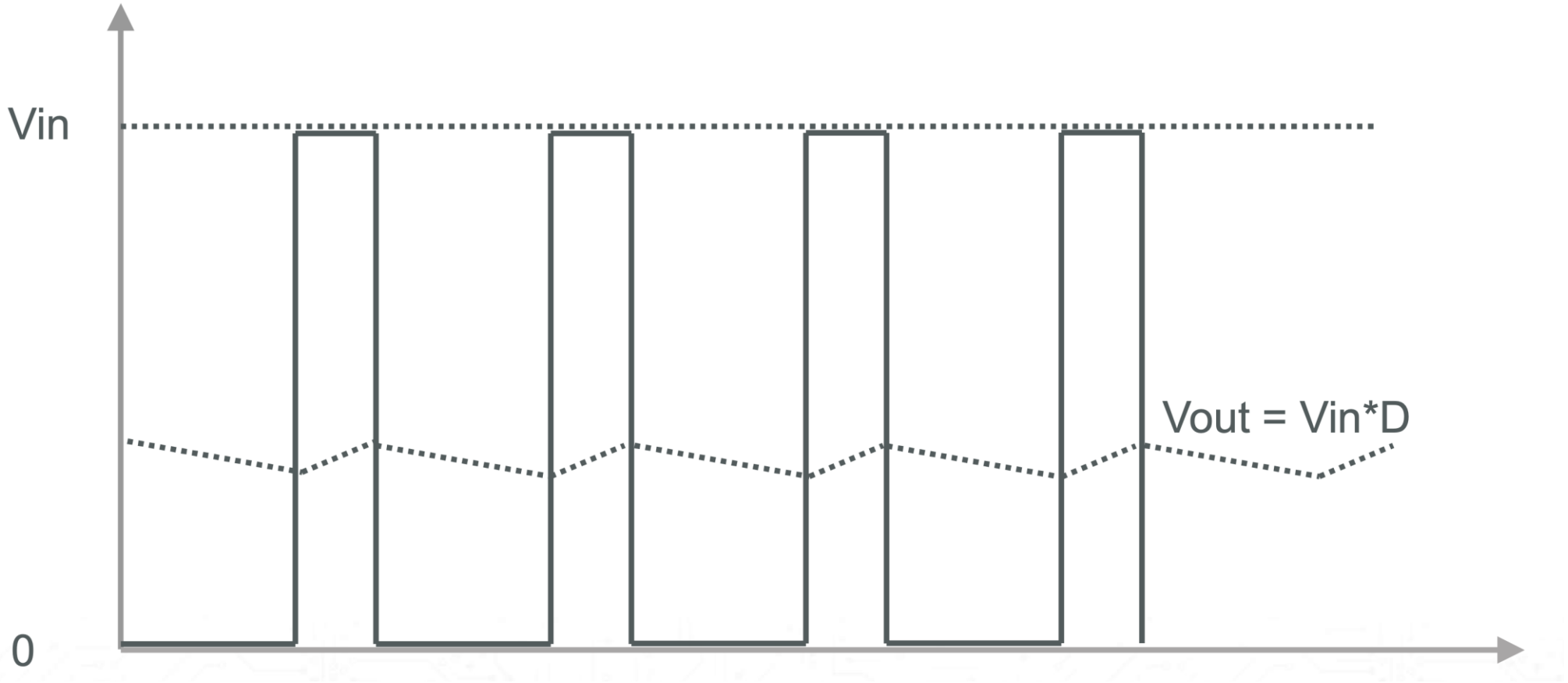
Contents

- What is a Multi-level DCDC Converter
- The Verification Challenge
- Limitations of Traditional Modelling methodologies
- What are EEnet models?
- Accuracy of EEnet methodology
- Speed of EEnet methodology
- Pros and Cons
- Q&A

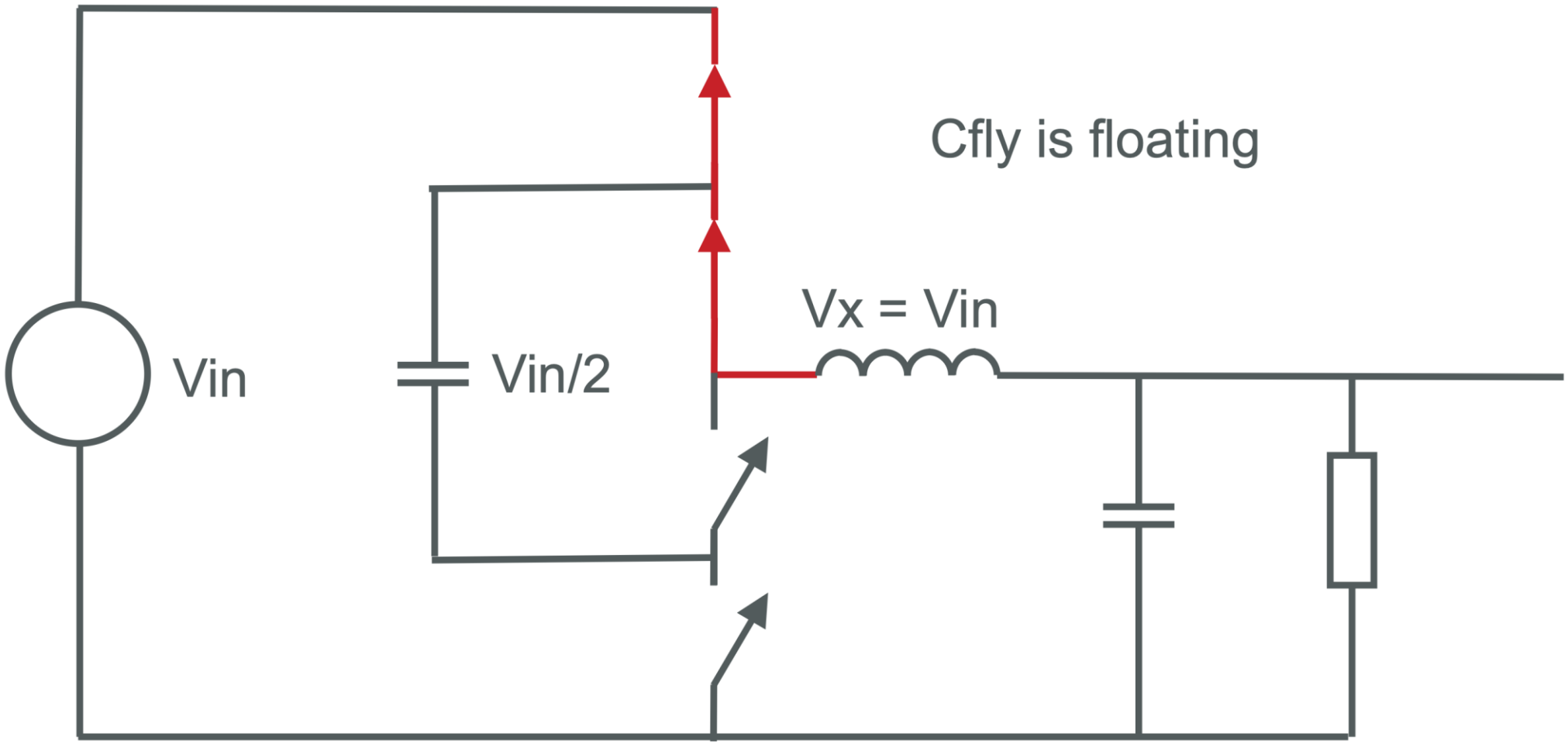
Basic Buck Converter Output Stage



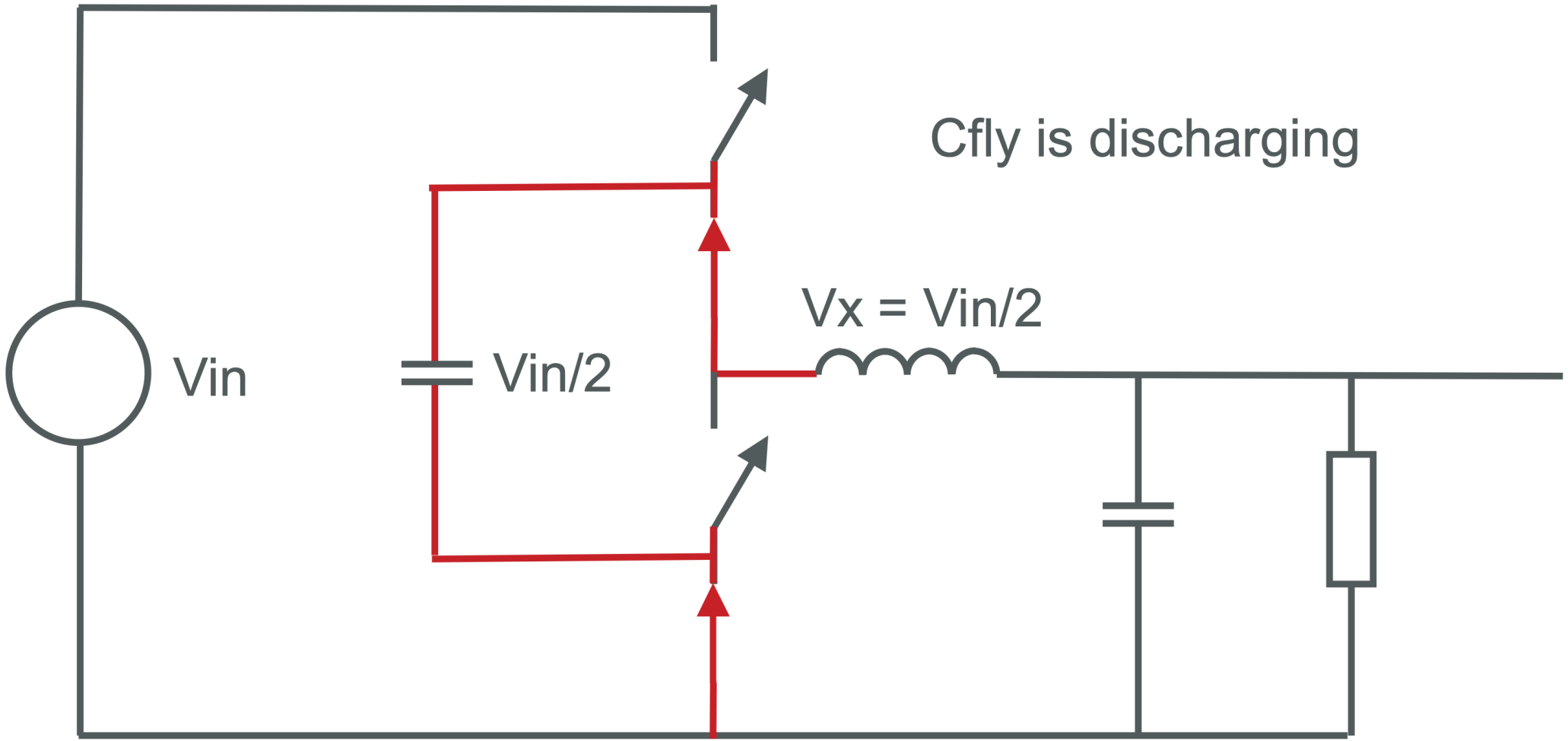
Basic Buck Converter Output Switching Waveform



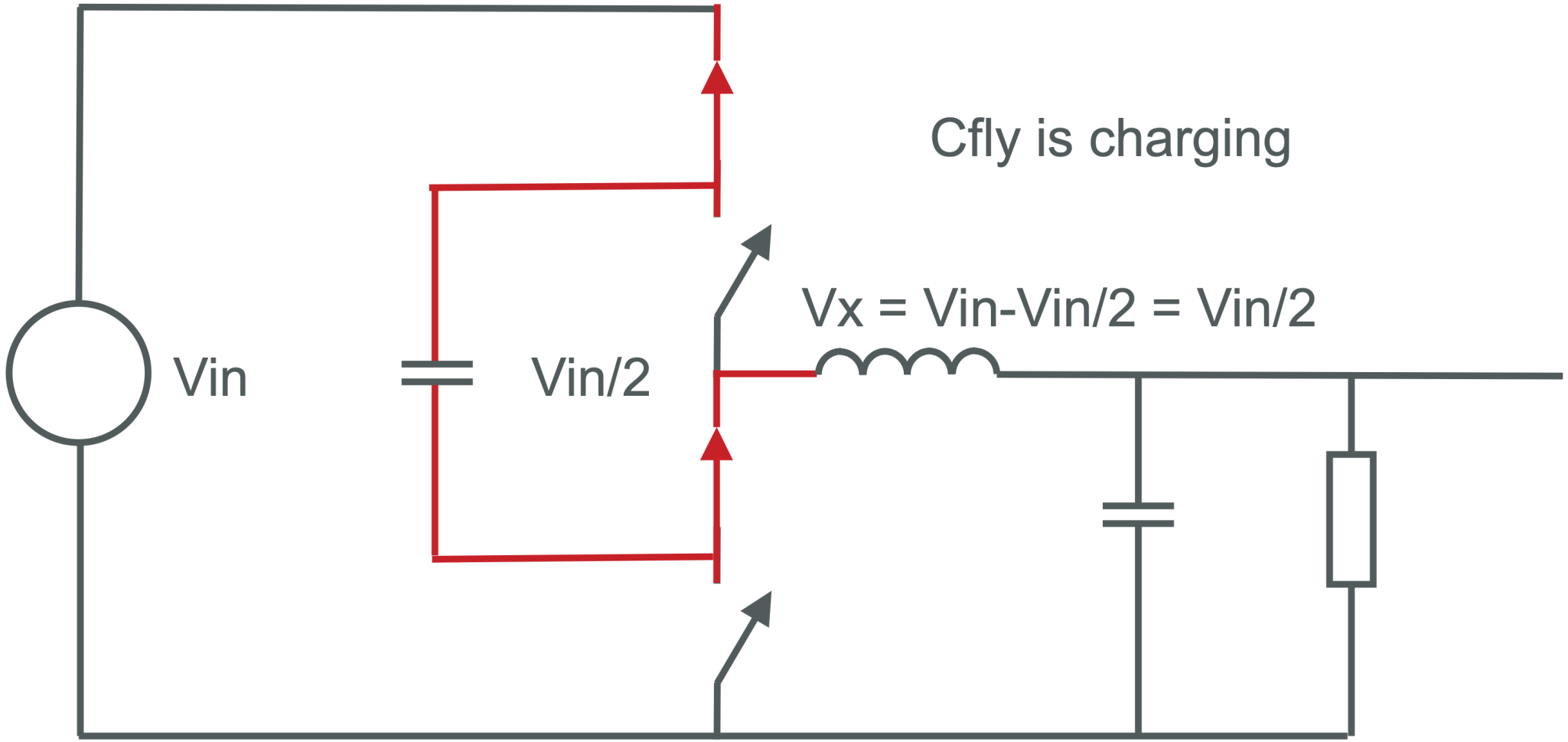
3-Level Buck Converter Output Stage



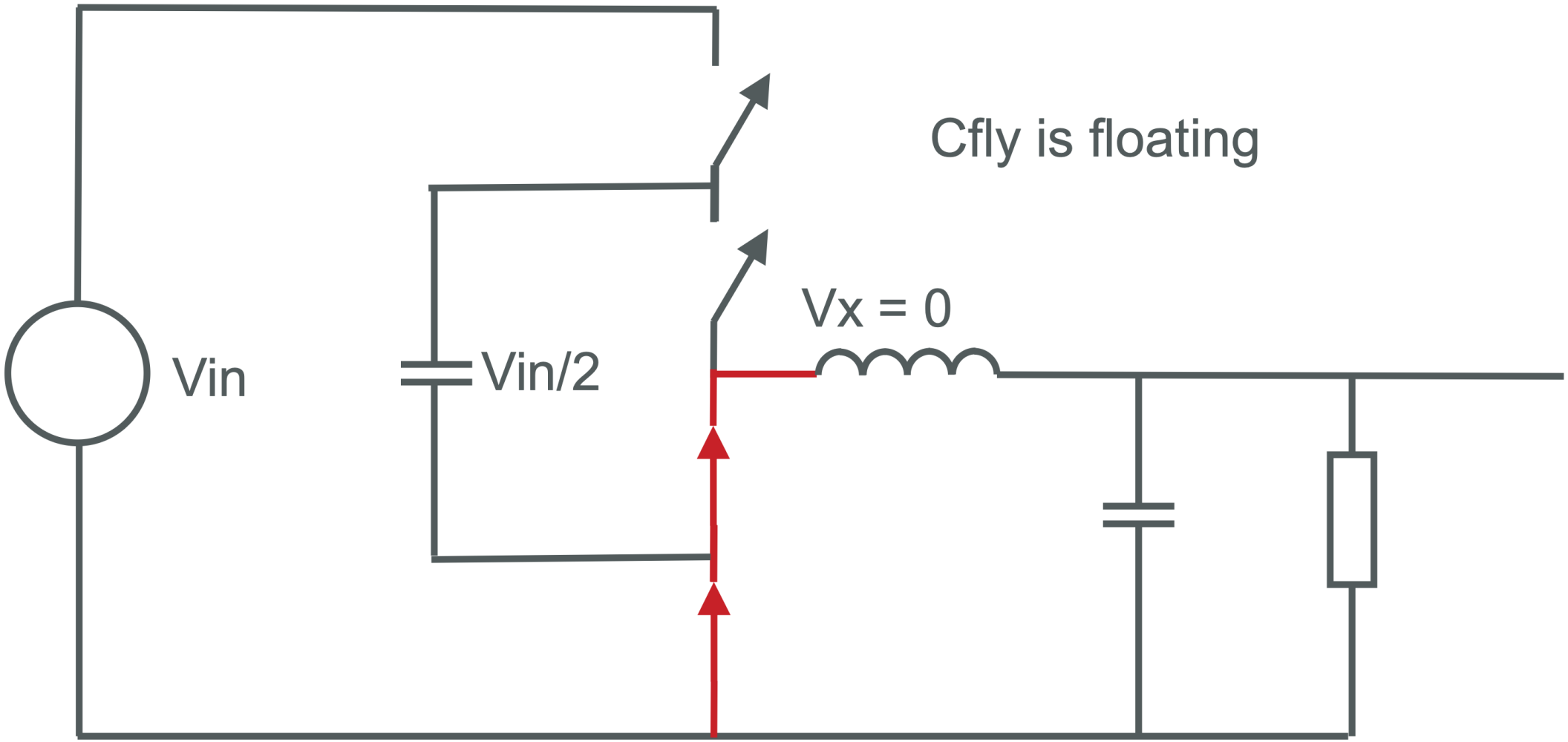
3-Level Buck Converter Output Stage



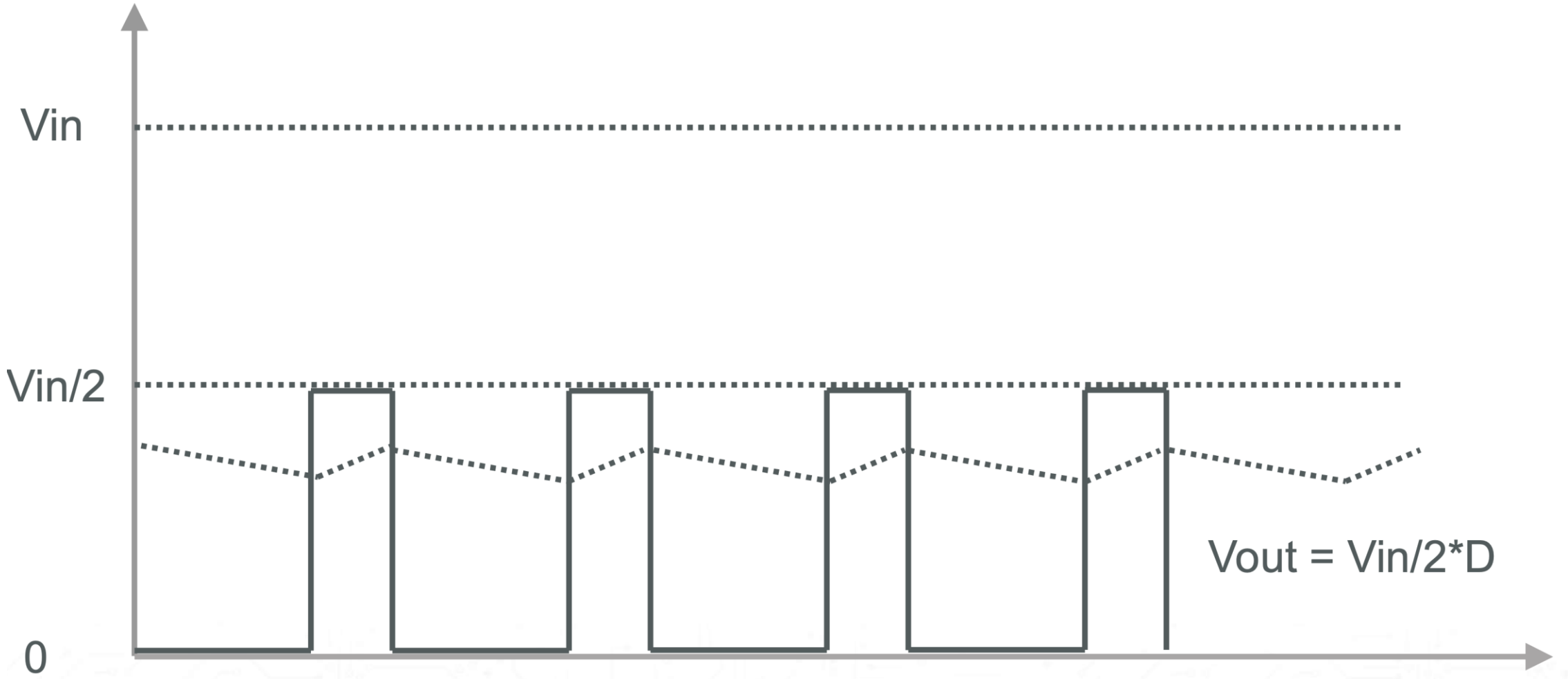
3-Level Buck Converter Output Stage



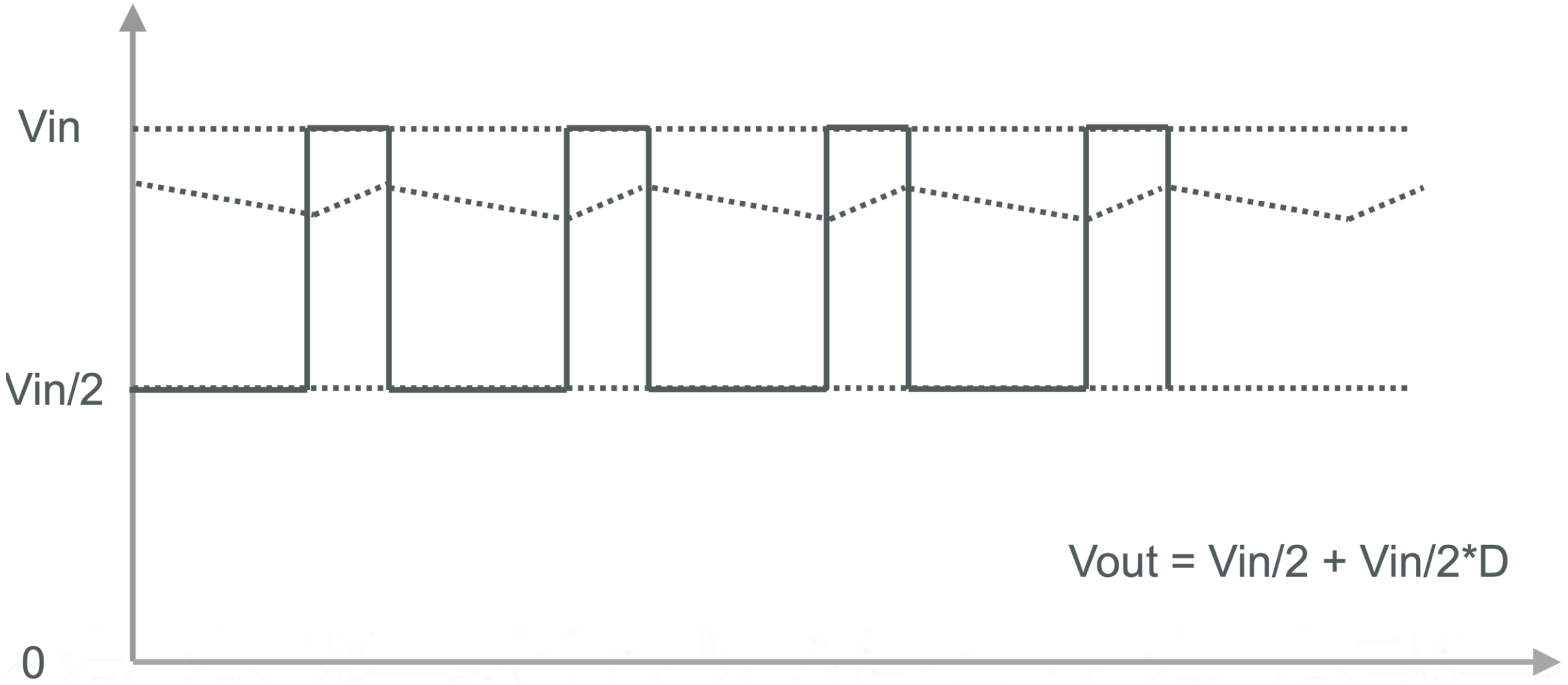
3-Level Buck Converter Output Stage



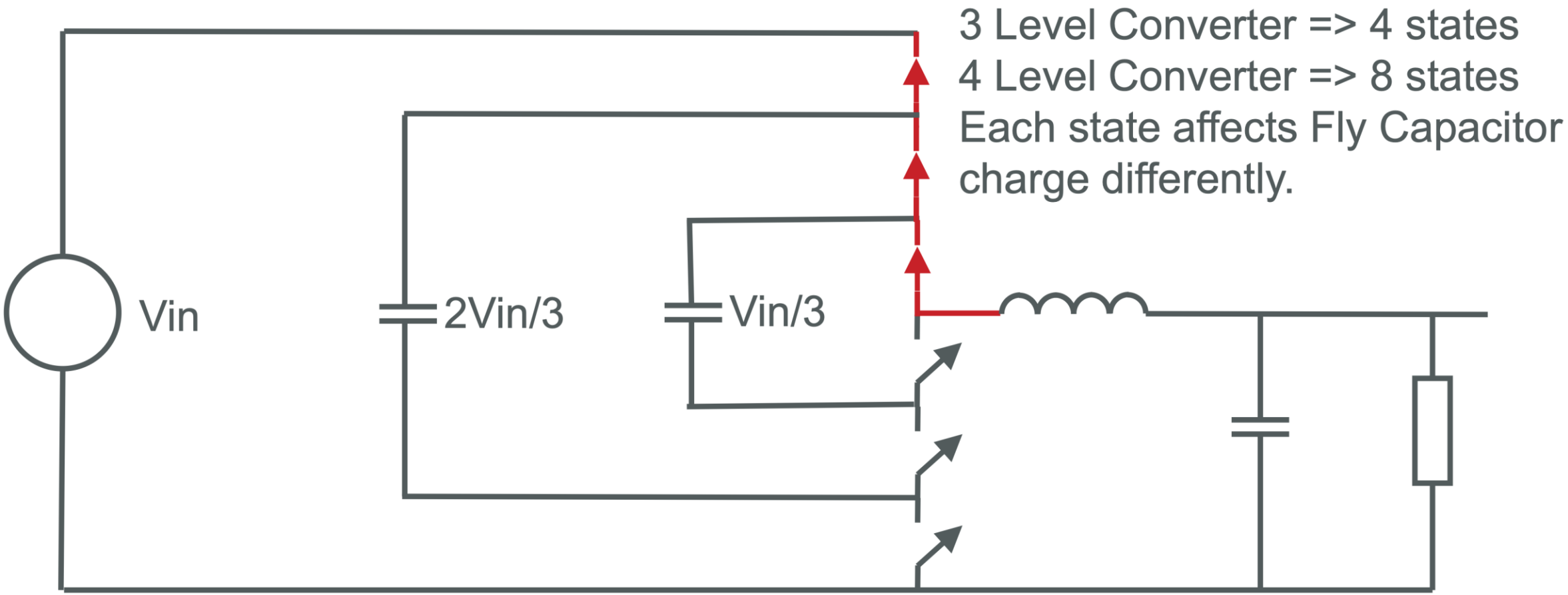
3-Level Buck Converter Output Switching Waveforms



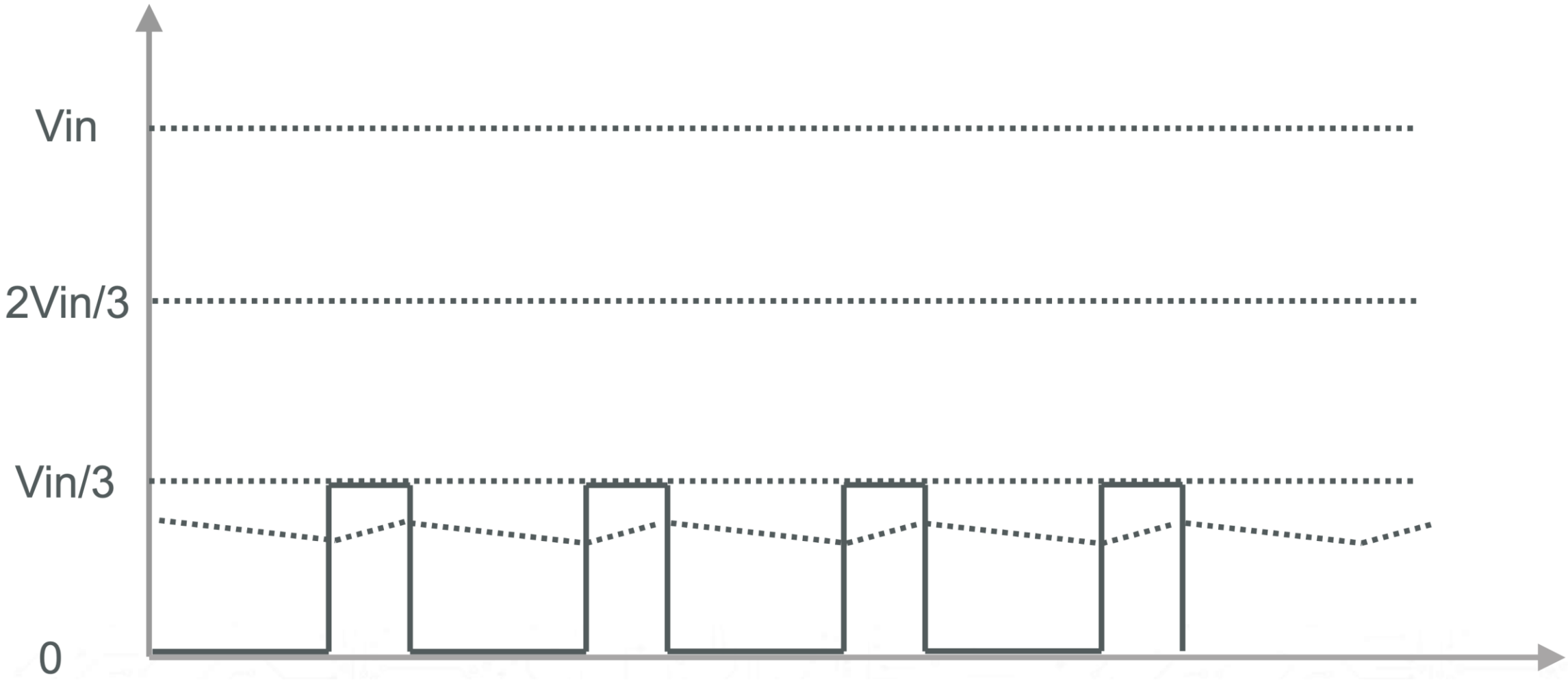
3-Level Buck Converter Output Switching Waveforms



4-Level Buck Converter Output Stage

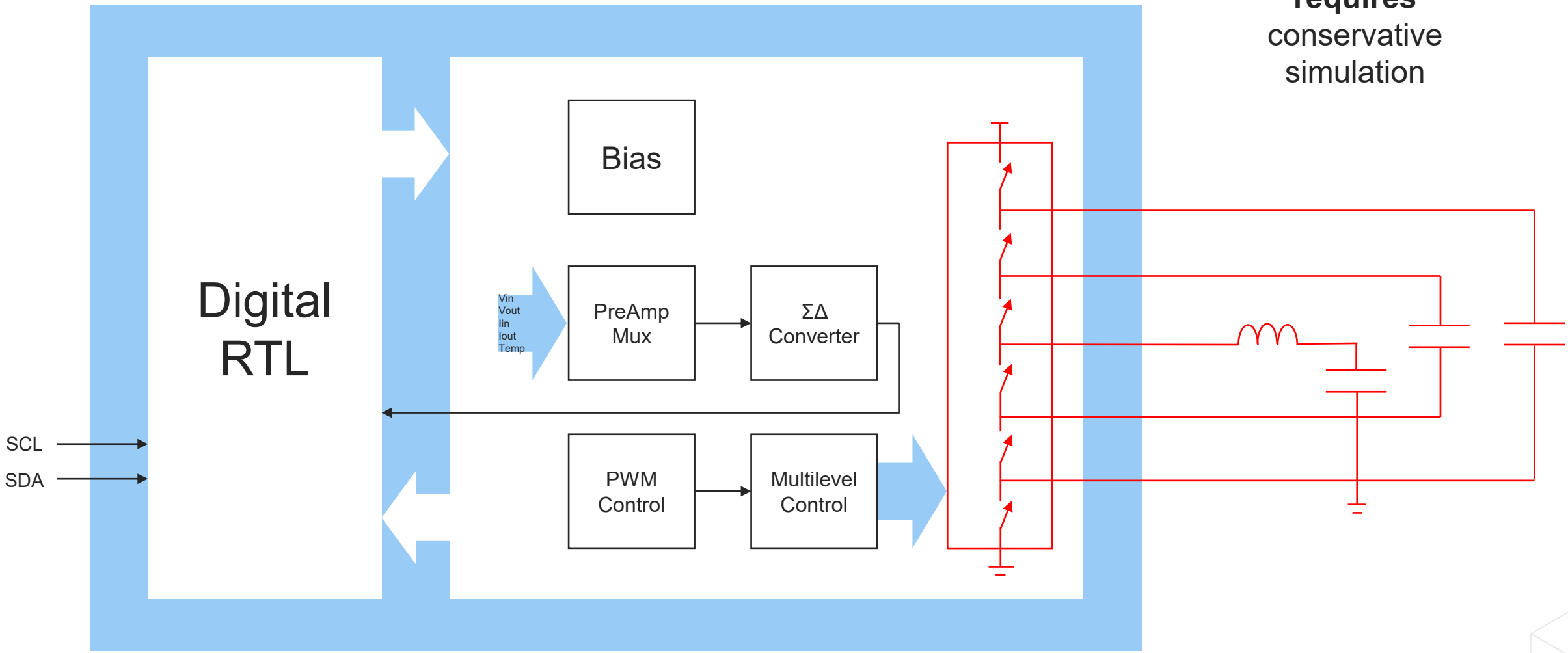


4-Level Buck Converter Output Switching Waveforms

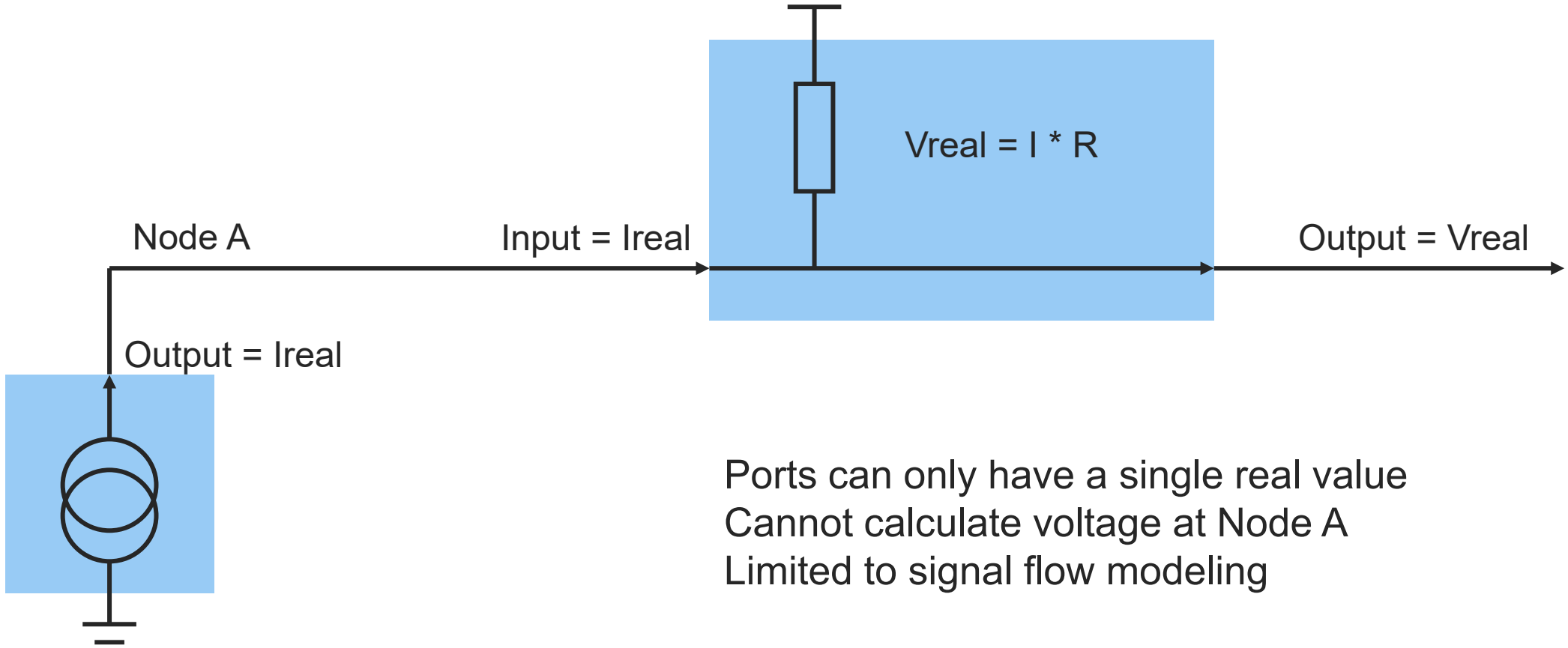


The Verification Challenge

Red circuitry **requires** conservative simulation

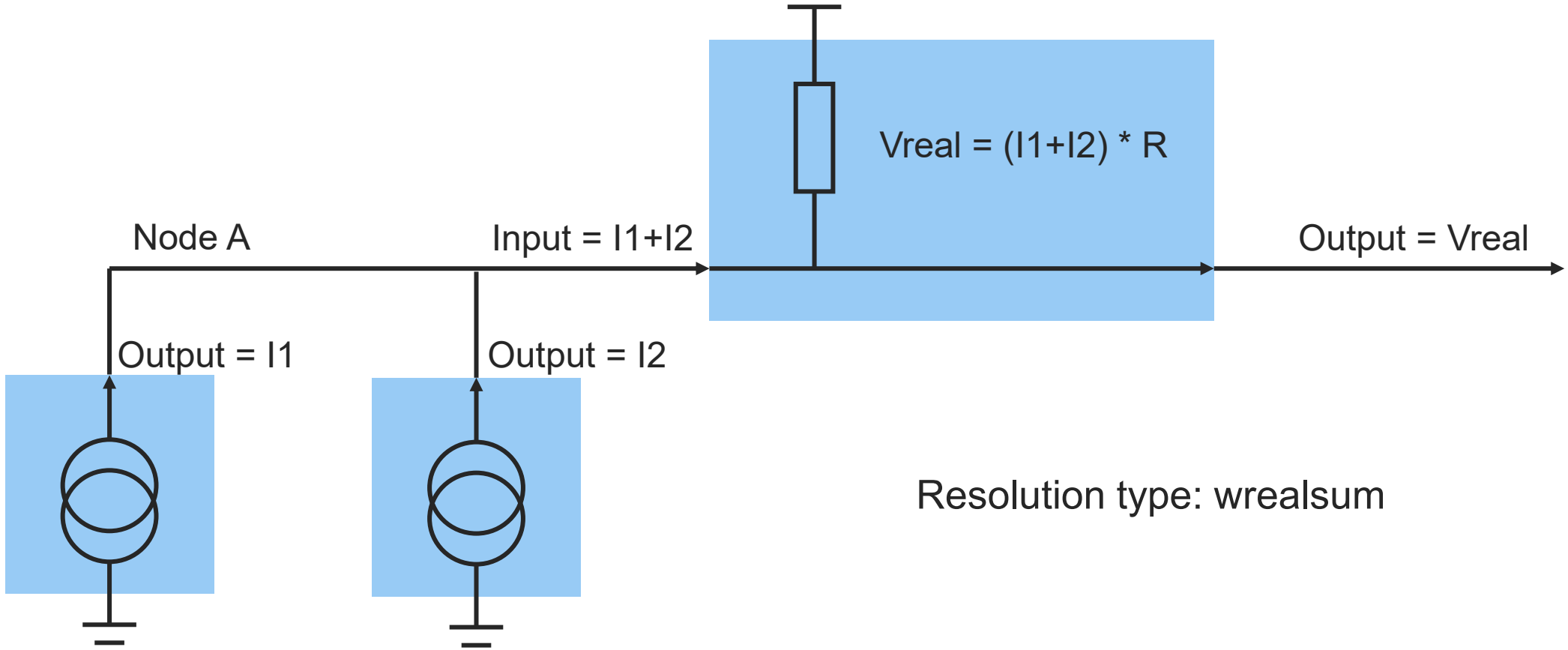


Limitations of WREAL Models - Single Value Per Port



Ports can only have a single real value
Cannot calculate voltage at Node A
Limited to signal flow modeling

Limitations of WREAL Models - Limited Resolution Functions



Resolution type: wrealsum

What are EEnet Models?

- EEnets employ SystemVerilog language features:
  - User-Defined Types (UDT)
  - User-Defined Nets (UDN)
  - User-Defined Resolution Functions (UDR)

SystemVerilog UDN and UDR

- Integral part of SV language definition
- User Defined Nets (UDN)
  - Allow any number of independent variables to be assigned to a port
    - real
    - logic
    - int
- User Defined Resolution Functions (UDR)
  - Allow any function to resolve multiple drivers on net

What are EEnet Models?

- User (Cadence) Defined Type (UDT)

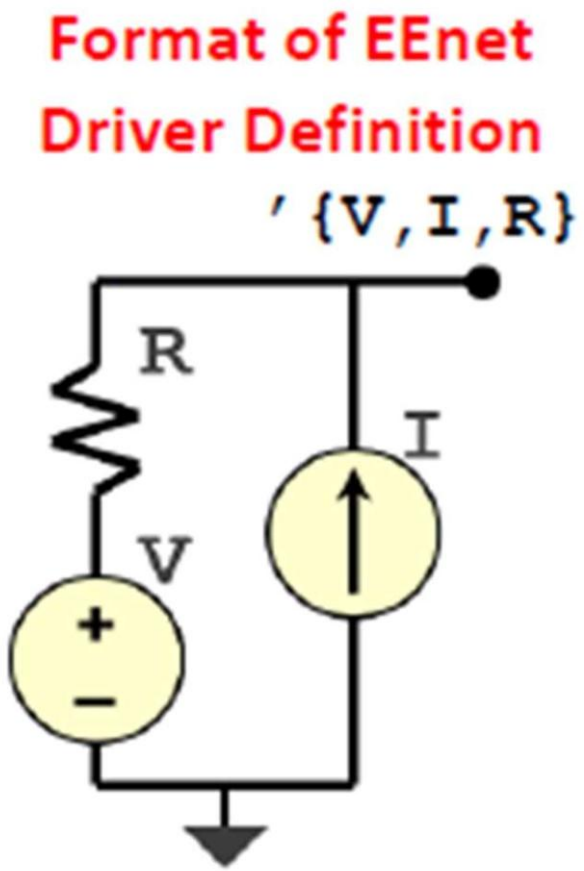
```
typedef struct {
    real V;
    real I;
    real R;
} EEstruct;
```

- User Defined Resolution Function (UDR)

```
function automatic Eestruct res_EE
(input EEstruct driver[ ]);
    <lots of function code here>
```

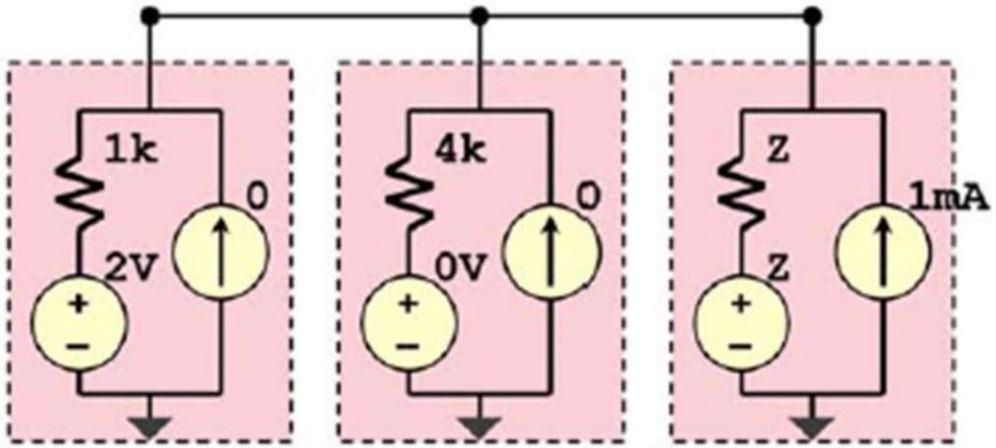
- User Defined Net (UDN)

```
nettype EEstruct EEnet with res EE;
```



EEnet Resolution Function

- Reproduced from Cadence's EEnet Rapid Adoption Kit



Output Evaluation:

ΣI = 2mA+0mA+1mA = 3mA

ΣG = 1m + 0.25m + 0 = 1.25m

Vout = ΣI / ΣG = 3m/1.25m = 2.4V

Rout = 1/ΣG = 800 Ω

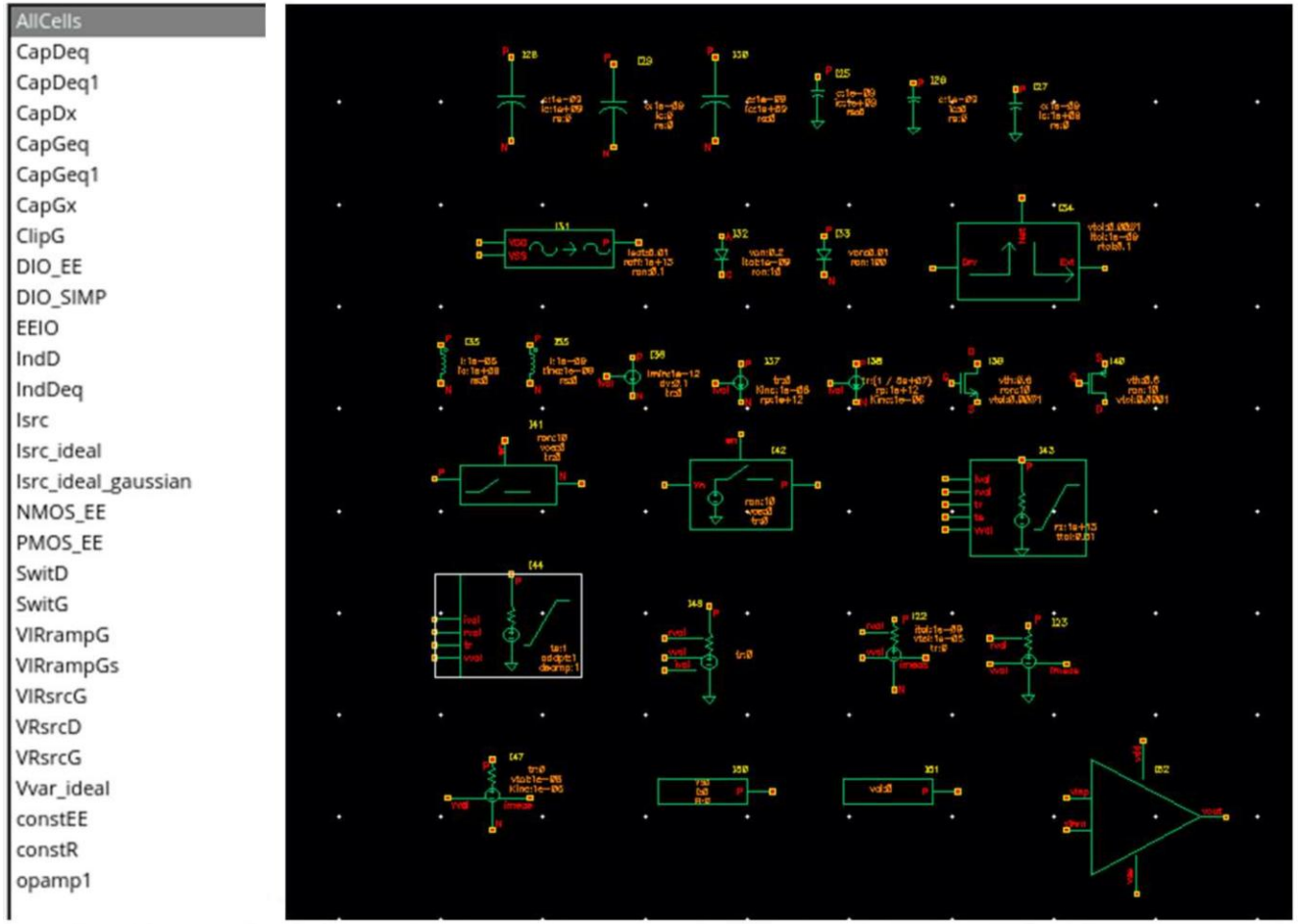
Resulting EEnet value: '{2.4, 0, 800}

What About Capacitors?

- V = Integ(I * dT) / C
- V1 = V0 + ((I0 + I1)/2 * dT) / C
- V1 = [V0 + I0 * dT/2C] + [I1 * dT/2C]
- V1 = [Veq] + [I1 * Req]
- This simple difference equation fits directly into the EEnet driver model
- Similar arithmetic allows for Inductors
- NB – this provides an exact solution equivalent to Spice
- Need to provide suitable internal event clock

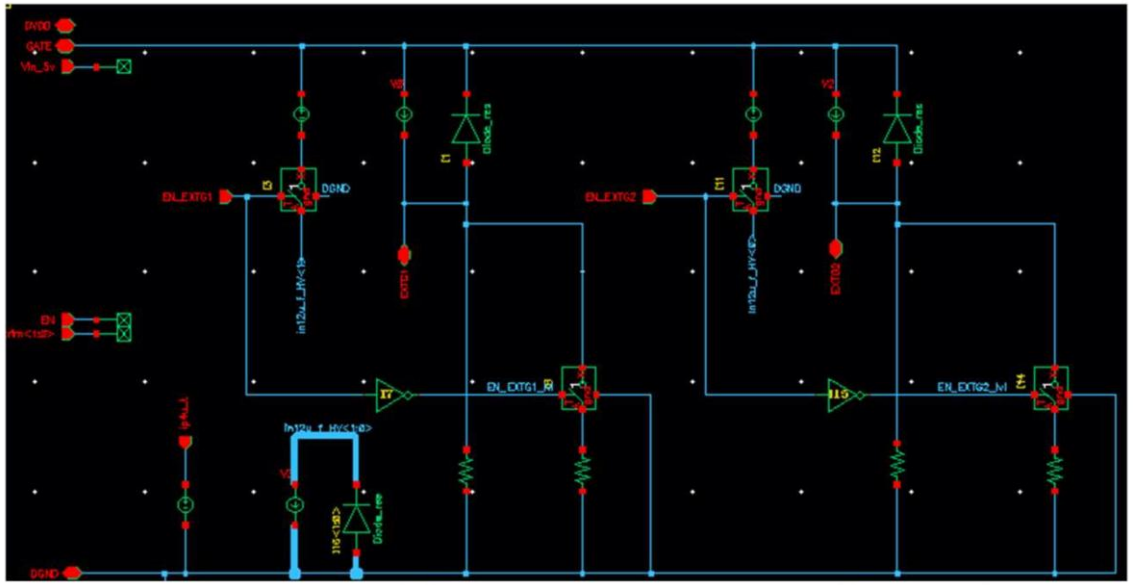
EEnet Library

- Library is included in Cadence Xcelium installation
- Models are pre-written and include advanced modeling features
  - Convergence limits
  - Automatic timestep control
  - Floating (differential) components
- Covers vast majority of analog modelling needs
- Greatly eases adoption



EEnet Simulation Speed

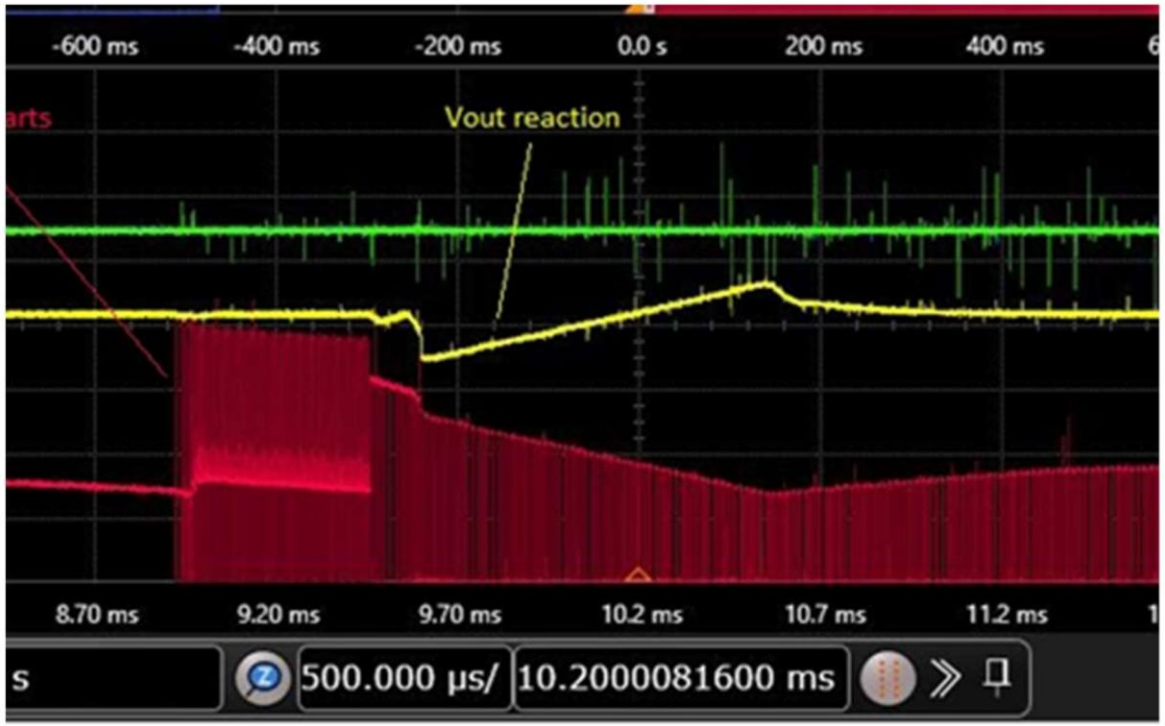
- Extremely fast ~2000x in like-for-like comparison
  - Top level SOC simulation in minutes, not days

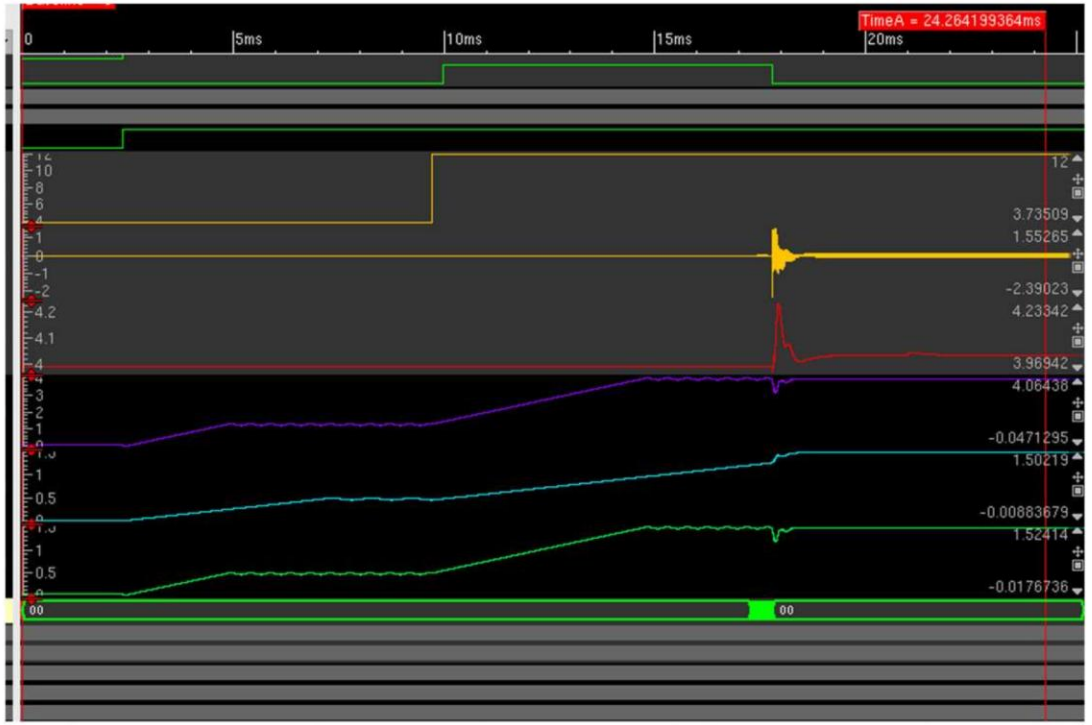



```
module GateDriveSW_mirr_ES7
import EE_pkg::*; import cds_rnm_pkg::*;
( DGND, DVDD, EXTG1, EXTG2, GATE, in4u_f, EN_EXTG1,
EN_EXTG2, ip4u_t, PSUB, EN, Vin_5v, Itrim );

  input logic EN_EXTG2;
  inout EEnet DGND;
  input logic EN_EXTG1;
  input logic EN;
  inout EEnet PSUB;
  inout logic EXTG2;
  input real Vin_5v;
  input real  ip4u_t;
  inout EEnet GATE;
  inout EEnet DVDD;
  input logic [1:0] Itrim;
  inout wreal4state  in4u_f;
  inout logic EXTG1;

// Shorthand for standard real constants:
`define Z `wrealZState
`define X `wrealXState

parameter real R4   = 2.0e3;
parameter real R1   = 1.0e6;
parameter real R0   = 49.1;
parameter real R5   = 2.0e3;
parameter real R3   = 1.0e6;
parameter real R2   = 49.1;

real iref;

assign EXTG1  =  ((real'(EN_EXTG1) * (GATE.V - (iref * R4))) > ((DVDD.V - DGND.V) / 2)) ? 1 : 0;
assign EXTG2  =  ((real'(EN_EXTG2) * (GATE.V - (iref * R5))) > ((DVDD.V - DGND.V) / 2)) ? 1 : 0;

always @*
  begin
    case(Itrim)
      2'b00:   iref = real'(EN) * ip4u_t * (6/8)  * 4;
      2'b01:   iref = real'(EN) * ip4u_t * (6/4)  * 4;
      2'b10:   iref = real'(EN) * ip4u_t * (6/16) * 4;
      2'b11:   iref = real'(EN) * ip4u_t * (6/12) * 4;
      default: iref = 0;
    endcase
  end

function real max(input real a,b); max = (a>b) ? a : b;
endfunction

endmodule
```

Accuracy of EEnet Methodology

- Extremely accurate modelling of behaviour
  - Allows high confidence in algorithms, system stability and functionality





Pros and Cons

Advantages	Challenges
Very accurate - SPICE-level accuracy possible	Requires expert modeling ability (timesteps, convergence)
Very wide range of abstraction possible	Verification engineer must straddle Analog and Digital domains
Extremely fast system simulation speeds	Project team and database must be managed as top-down – Digital on Top
Only requires single simulation kernel (and license)	Who to lead methodology in mixed-signal team?