VerilogA: Buck Switching Regulator

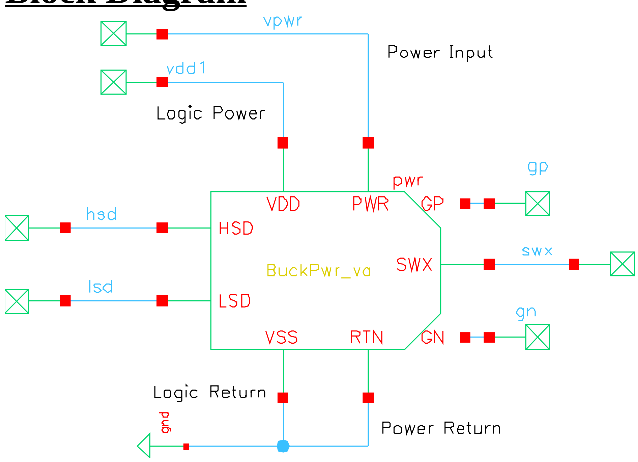
Power Switch Model

September 2009

Description

This Verilog-A model emulates the power stage of a Buck Switching Regulator and is design for use in tope down power management system design. The model includes a high side and low side switch. The high side switch emulates a PMOS device, the low side switch emulates a NMOS device. The switch devices include body diodes with programmable turn on voltage. Switch device on resistance and gate charge are included in the model. The model also includes switch overlap delay and a disable(tri-state) mode.

Block Diagram



Features

- Pmos High Side Switch
- Nmos Low Side Switch
- Programmable Transition Characteristics
- Overlap Error Reporting
- Observable Gate Voltage Behavior

Pin Descriptions

Power Pins;

PWR – Power Switch Input Voltage
VDD – Logic Power
RTN – Power Return
VSS – Logic Return

Logic Inputs

HSD – High Side Driver Enable
LSD – Low Side Driver Enable

Analog Output

SWX – Power Switch Output

Auxilary Outputs

GP – Pmos Gate Voltage
GN – Nmos Gate Voltage

Parameters;

Vtp - Threshold of Pmos Switch.
Gmp - Transconductance of Pswitch
Rdson=gm/(vin-vt)
cgp – Equivalent Pmos Gate capacitance
tgp – Gate transition time
vtn - Threshold of Nmos Rectifier
gmn – Transconductance of Nmos
cgn – Equivalent Nmos gate capacitance
tgn – Nmos gate transition time
vdf – Diode turn on voltage
gdf - Diode conductance when on

VerilogA Code Fragment

```
// Buck Switching Regulator Model: VerilogA
//  Model Emulates Systems where vdsMax == vgsMax
// Power Tranfer Switches;
//  High Side Switch(HSD) is Pmos
//  Low Side Switch(LSD) is Nmos
//  Model has Substrate Diode
`include "constants.vams"
`include "disciplines.vams"
// Pin Definitions
//  I. Power Supply
//   1. VDD,VSS - Bias Reference Power
//   2. PWR,RTN - Power Input/Return
//  B. Output
//   1. SWX - The Switch Output
//  C. Logic Inputs
//   1. HSD - Assert -> High Side Driver On
module buckPwrSwx(VDD,PWR,SWX,RTN,VSS,LSD,HSD,GP,GN);
inout VDD,PWR,RTN,VSS;
inout SWX;
input HSD,LSD;
output GP,GN;
electrical VDD,PWR,RTN,VSS,SWX,HSD,LSD;
// Internal Electrical Signal
//   GP - HSD Gate Voltage
//   GN - LSD Gate Voltage
electrical GP,GN;
// ------------  Module Parameters ----------------------------------------
// 1. gmp - Kp*Wp/Lp in A/V
// 2. vtp - Pmos Threshold
// 3. cgp - Equivalent gate cap. for Pmos
// 4. tgp - Pmos Gate Transition Time
// 5. gmn - Kn*Wn/Ln in A/V
// 6. vtn - Nmos Threshold
// 7. cgn - Equivalent gate cap. for Nmos
// 8. tgn - Nmos Gate Transition Time
// 9. vdf - Forward Diode Volage
// 10. gdf - Diode tranconductance
// .... note that the gate charge is approximated for target Process Node
parameter real gmp=2.0 from (0.01:10.0);
parameter real vtp=0.6 from (0.1:1.0);
parameter real cgp=3.5e-10 from (0:1.0e-7);
parameter real tgp=5.0e-9 from (1.0e-10:1.0e-7);
parameter real gmn=2.0 from (0.01:10.0);
parameter real vtn=0.6 from (0.1:1.0);
parameter real cgn=3.5e-10 from (0:1.0e-7);
parameter real tgn=5.0e-9 from (1.0e-10:1.0e-7);
parameter real vdf=0.7 from (0.2:1.0);
parameter real gdf=20.0 from (1.0:100.0);
```

```
// ------------- Variable Definition  -------------------------------------
// ** Standard Definitions
// errorFlag - Switch to Flag any Error
// state - Current state of system; 0-Sleep
// tsMag - Cross Event Time Base Accuracy
// vsMag - Cross Event Voltage Accuracy
// tsBool - Internal Logic Transition Time
// VthIn - Logic Threshold
// *** Model Specific Definition
integer errorFlag,state,hsdBool,lsdBool;
real tsMag,vsMag,tsBool,VthIn; // Logic Event Variables
real gpMag,td_gp,tr_gp,tf_gp,gnMag,td_gn,tr_gn,tf_gn;
real ipLeakage,inLeakage,ipMag,inMag,tr_o,tf_o;
real gmpDiv2,gmnDiv2;
real vgsP,vdsP,vgsN,vdsN,idp,idn;
real iddGp,iddGn;
analog
 begin
  @(initial_step)  // Initialize
   begin
    errorFlag=0;
    state=0;
    tsMag=1.0e-9;
    vsMag=1.0e-3;
    tsBool=0.25e-9;
    VthIn=1.50;
   // *** Embedded Power Mos Parameters
    td_gp=1n;
    tr_gp=1n;
    tf_gp=1n;
    td_gn=1n;
    tr_gn=1n;
    tf_gn=1n;
    tr_o=1.0e-10;
    tf_o=1.0e-10;
   end
  @(final_step)  // Final Time Step Clean up and Exit
    begin
      if (errorFlag != 0)
        begin
          $strobe(" Application Error ");
          if (errorFlag==1) $strobe("Command both Fet's ON Error");
        end
    end
```


Tel: 972.741.6010 Email: info@icdsn.com web: www.icdsn.com

```
//  -----------------  Logic Interface  ----------------------------------
 // *** High Side Driver
 if(V(HSD,VSS) > VthIn)
    begin     // <-- Enable
      if (V(GN,RTN) < vtn) gpMag=V(RTN);
        else gpMag=V(PWR);
    end
    else gpMag=V(PWR);
 if(V(LSD,VSS) > VthIn)
    begin     // <-- Enable
      if (V(PWR,GP) < vtp) gnMag=V(PWR);
        else gnMag=V(RTN);
    end
    else gnMag=V(RTN);
  // ------------------- Internal Electrical Nodes -------------------------
  // Virtual gate nodes are electrical for slewrate control
  V(GP,RTN) <+ transition(gpMag,0,tgp,tgp);
  V(GN,RTN) <+ transition(gnMag,0,tgn,tgn);
  // -------------------- The Power Fet Models -----------------------------
  vgsP=V(PWR,GP);
  vdsP=V(PWR,SWX);
  vgsN=V(GN,RTN);
  vdsN=V(SWX,RTN);
  // High Side Driver
  if (vgsP <= vtp) ipMag=0; // ipMag=ipLeakage
   else if ( vdsP < 0.5*(vgsP-vtp) ) ipMag=gmp*(vgsP-vtp)*vdsP;
     else ipMag=0.5*gmp*(vgsP-vtp)*(vgsP-vtp);
  // Low Side Driver
  if (vgsN <= vtn) inMag=0; // inMag=inLeakage
  else if ( vdsN < 0.5*(vgsN-vtn) ) inMag=gmn*(vgsN-vtn)*vdsN;
    else inMag=0.5*gmn*(vgsN-vtn)*(vgsN-vtn);
  // Body Diodes: Simple Breakpoint Model to improve simulation time
  if ( vdsP < -vdf ) idp = (vdsP + vdf)*gdf ;
    else idp=0;
  if ( vdsN < -vdf) idn = (vdsN + vdf)*gdf ;
    else idn=0;
  // ------------------------  Charge Loss  ----------------------------------
  // *** Converting charge into current avoids ddt operators and simulates
  //  faster. Also squares up the current into more of an event.
  if ( gpMag - V(GP,RTN) > vtp ) iddGp=0.05*cgp*V(VDD,VSS)/tgp;
    else if ( V(GP,RTN) - gpMag > vtp ) iddGp=cgp*V(VDD,VSS)/tgp;
    else iddGp = 0.0;
  if ( gnMag - V(GN,RTN) > vtn ) iddGn=cgn*V(VDD,VSS)/tgn;
    else if ( V(GN,RTN) - gnMag > vtn ) iddGn=0.05*cgn*V(VDD,VSS)/tgn;
    else iddGn = 0.0;
  // ------------------------ Output Vector ----------------------------------
 I(PWR,SWX) <+ slew(ipMag ,1.0e9,-1.0e9) + idp;
 I(SWX,RTN) <+ slew(inMag ,1.0e9,-1.0e9) + idn
            + 1.0e-6*V(SWX,RTN) + 1.0e-12*ddt(V(SWX,RTN));
 // ---- Add the Dynanic power supply Current
 I(VDD,VSS) <+ slew(iddGp + iddGn,1.0e9,-1.0e9);
```

End ***endmodule***