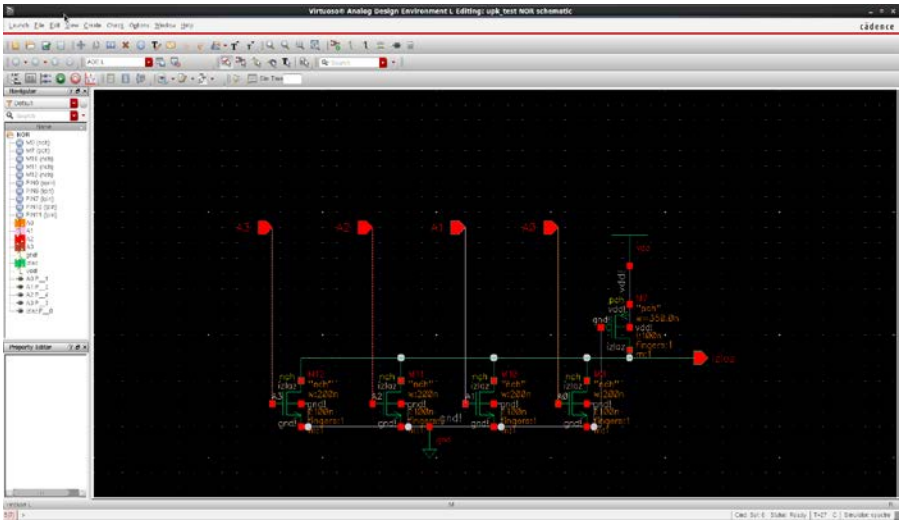
6.5 Korišćenje vektor fajlova (.vec)

Velike šeme sa velikim brojem ulaznih signala je veoma teško simulirati sa stimulus fajlovima, jer se ulazni signali zadaju pojedinačno, pa stimulus postaje nepregledan i zahtevan za pisanje. U takvim situacijama se koriste vektor fajlovi *.vec* sa kojim je moguće zadati veliki broj ulazni signala na jednostavan i pregledan način.

Dalja demonstracija vektor fajlova je rađena u 90nm tehnologiji gde se napon napajanja razlikuje i iznosi 1,2 V (umesto 1,8 V), ali svi načini korišćenja vektora su identični.

Korišćenje vektor fajlova

Demonstracija vektora biće prikazana na jednostavnom dinamičkom četvoroulaznom NOR kolu sa pass tranzistorom i jednim izlazom.



U simulaciji sa vektor fajlovima je neophodno koristiti stimulus fajlove za definisanje konstantnih signala, izvora napajanja (kao što su vdd! i gnd!) i izlaznih kapacitivnosti i otpornosti. Stimulusi se pišu na identičan način kao prethodno opisano. Stimulus pisan u ovoj šemi je

```
simulator lang=spectre
Vdd (vdd! 0) vsource dc=1.2
Gnd (gnd! 0) vsource dc=0
c1 (0 izlaz) capacitor c=2f
```

Procedura je ista kao što je prethodno opisano, u Cadence-u u ADEL simulaciji se u stimulus fajlovima ubaci prethodni .scs fajl i čekira polje za stimulus. **Napon napajanja vdd = 1.2 V jer se radi o 90nm tehnologiji, za 180nm staviti 1.8 V** Sledeći korak je napisati vektor fajl. Demonstrirana su 3 primera.

Primer 1

Primer .vec fajla koji je korišćen za simulaciju :

```
signal A3 A2 A1 A0
io i i i i
period 0.25n
fall 0.01n
rise 0.01n
vih 1.2
vil 0.0
radix 1 1 1 1
1 0 1 0
1 0 1 1
1 1 0 0
1 1 0 1
1 0 0 1
1 0 0 1
0 0 0 0
1 0 0 0
0 0 1 1
```

Prva linija koda predstavlja signale koji se koriste u simulaciji (počinje sa “signal”, a moze da počinje i sa “vname”). Ovde su opisana 4 ulazna signala A3 A2 A1 i A0. Signale je potrebno odvojiti razmakom (moze i vise uzastopnih razmaka, ako je potrebno radi preglednosti) i svakoj narednoj liniji koda se opisuju signali. Svi opisi su razdvojeni razmakom i opisuju signale sa istim redosledom kao i u prvoj liniji koda (na primer u radix-u, druga jedinica odgovara signalu A2). Ukoliko je neki opis signala za sve isti, onda je dovoljno staviti jedan broj (na primer “fall 0.01n” je isti za sve A3 A2 A1 A0)

Druga linija koda opisuje da su signali (počinje sa “io”) ulazni (i), izlazni (o) ili bidirekcioni (b). Ovde se vidi da su sva četri signala ulazna,što će uglavnom i biti.

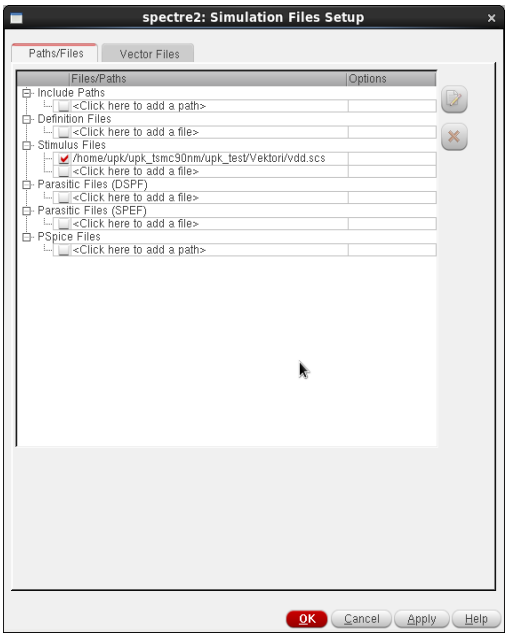
Treća linija koda opisuje način na koji će vektori biti zadati u tabeli podataka, odnosno tabeli vektora (poslednji blok u fajlu sa nulama i jedinicama, ispod radix-a). Ovde je izabrano periodično zadavanje, što znači da svaka linija koda u tabeli vektora predstavlja vrednosti odgovarajućih signala, a te vrednosti traju periodom specificiranoj u tabeli, u ovom slučaju je to 0.25n sekundi. **Na primer**, prva linija u tabeli : 1010 govori da su signali A3=1 A2=0 A1=1 A0=0, i to u vremenskom trajanju od 0ns do 0.25ns, sledeća linija 1011 govori da su signali A3=1 A2=0 A1=1 A0=1 u trajanju od 0.25ns do 0.5ns, sledeća linija 1100 traje od 0.5ns do 0.75ns i tako redom, sve do kraja tabele. Ovakav način predstavljanja je pogoda za “copy paste” jer se vrednosti vektora samo nadovezuju i nije potrebno specificirati pojedinačne vremenske trenutke. Mana je što nije moguće fino podešavati trajanja signala, već samo u rezoluciji glavne periode.

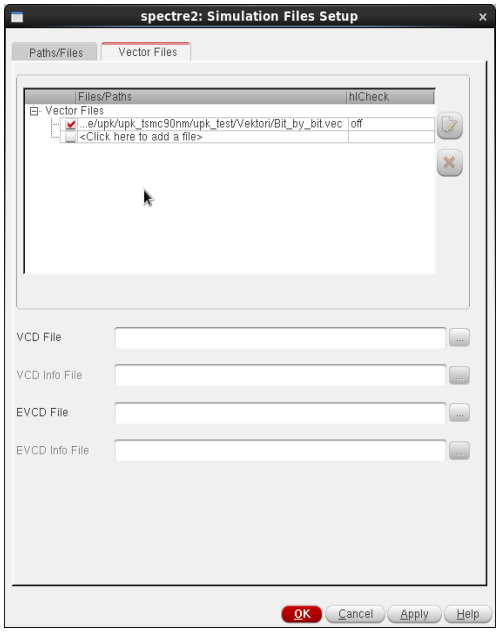
Sledeće dve linije, “fall” i “rise” zadaju trajanje uzlazne i silazne ivice signala, odnoso koliko je vremena potrebno da prednje sa jedinice na nulu i obratno.

Sledeće dve linije ,”vih” i “vil” zadaju naponske nivoe ulaznih signala, high i low. Ovde su 1.2 i 0 .

Poslednja linija pre tabele, opisuje format zadavanja vektora. Pošto su ovde svi signali, tj vektori A3 A2 A1 i A0 jedno bitni, radix za svaki vektor je 1, odnosno svaki vektor je zadatak jednim bitom 0 ili 1. Da su signali višebitni koristio bi se drugačiji radix.

Vektor fajl je neophodno napisati u nekom tekst editoru i promeniti ekstenziju u .vec. Zatim ga dodati u ADEL L simulatoru : **Setup => Simulation Files**
=> Vector files i kao za stimulus fajl, dodati ga u vector files i čekirati box gde je dodat. Ne zaboraviti i stimulus fajl sa napajanjima i kapacitivnostima.

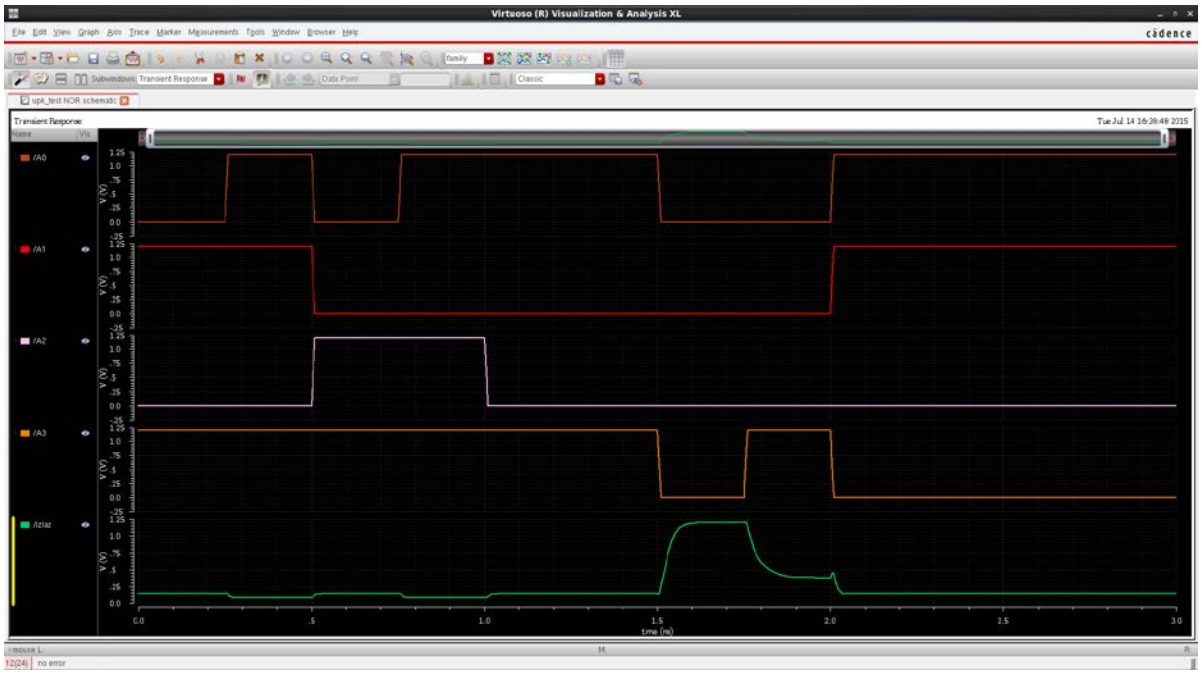




Takođe je moguće dodavati više vektor kao i stimulus fajlova gde su definisani različiti signali.

Dalje se simulacija pokreće na identičan način kao prethodno opisano.

VAŽNO: U nazivu vektor fajla ne smeju da postoje razmaci inače simulator neće da ga otvori, umesto razmaka koristiti donju crtu. Takođe ukoliko prilikom simulacije prijavi grešku u log fajlu da ne može da pronađe vektor fajl, potrebno je promeniti ime vektor fajl u neko drugo dok ne proradi.



Kao što se slike vidi, minimalno trajanje stabilnog signala je 0.25n sekundi.

Primer 2

Ovde je korišćeno isto kolo i stimulus, samo se razlikuje vektor fajl i **pin "izlaz" je promenjen u tristate pin (umesto output-a) to znači da pin može imati tri stanja : jedinicu, nulu, stanje visoke impedanse**. T**ristate se koristi kod ulazno-izlaznih magistrala. Tip pin se menja u tristate desni klikom na pin, pa : propreties=>direction=>tristate.**
Izgled vektor fajla

```
signal A[3:0] izlaz
io i i
tunit ns
slope 0.01
vih 1.2
radix 4 1
0.3         B 1
0.4+0.2     0 1
0.8         F 1
1           5 z
1.3         0 z
1.65        F z
1.9         0 0
2.1         3 0
```

Prva linija koda ima istu ulogu da definiše signale. Ovde je iskorišćena prednost indeksiranja i signali A3 A2 A1 i A0 su spojeni u jedinstveni vektor. Ovakvom indeksiranju treba težiti, jer je mnogo preglednije. Takođe je dodat izlazni signal koji sad može da bude ulazni i izlazni signal (tristate).

Druga linija koda je ista. **Signal "izlaz" je definisan kao ulazni jer se samo preko ulaznog signala može definisati stanje visoke impedanse ("z") u vektorskoj tabeli**.

Treća linija definiše vremenske jedinice u celom vektor fajlu, u ovom slučaju je to nano sekunda.

Linija "slope" zamenjuje rise i fall vremena, i postavlja ih na jednaku vrednost, 0.01 ns.

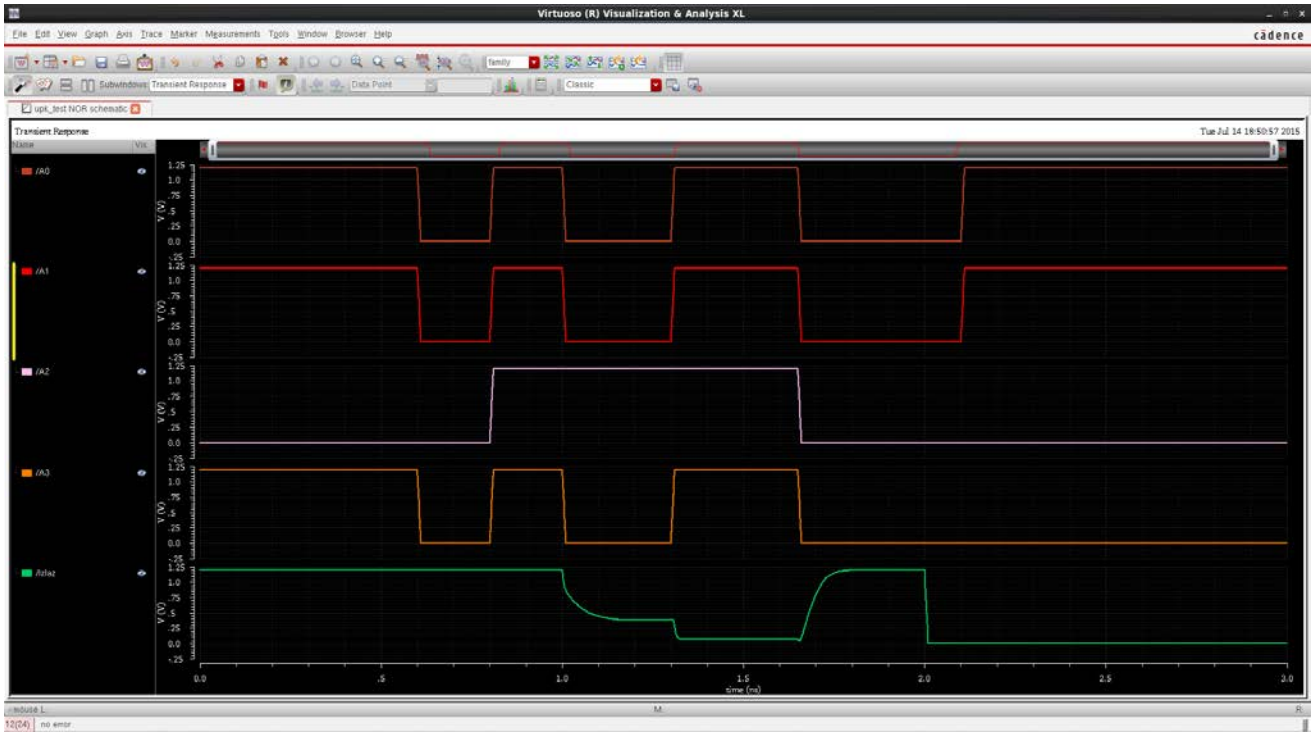
Vih je na 1.2, dok je vil izostavljen jer je po default-u postavljen na nulu.

Radix linija koda u ovom slučaju ima vrednosti "radix 4 1" jer jer prvi vektor (A[3:0])
četvorobitni. Kada radix ima vrednost 4, to znači da se četiri bita odgovarajućeg vektora predstavljaju heksadecimalnim brojem. U ovom primeru B znači A3=1 A2=0 A1=1 A0=1, F znači A3=1 A2=1 A1=1 A0=1 i tako dalje.

Ovde je korišćen apsolutni vremenski mod, način prikazivanja tabele vektora. Tu se definišu samo vremenski trenutci

kada dolazi do **promene** vektora. Na primer u 0.6 ns se vektor promenio na vrednost 0 (A3=0 A2=0 A0=0 A0=0), zatim u 0.8 ns se menja na vrednost F i tako dalje. Takođe je demonstrirano da se u vremenskim intervalima mogu koristiti aritmetičke operacije.

Takođe je moguće definisati stanje viske impedanse, sa "z". U stanju visoke impedanse signal ima skoro beskonačnu otpornost i tada signal efektivno postaje izlazni, jer drugi elementi kola definišu njegovi vrednost.
Kao što se može videti na slici kada je stanje viske impedanse izlaz (u intervalu od 1n do 1.9n), onda se kolo ponaša kao NOR kolu. U ostalom delu intervala, kada nije "z" na izlazu, signal "izlaz" ima definisane vrednosti iz vektor fajla i praktično postaje ulazni signal. Od 0ns do 1ns signal je definisan spolja na 1.2V, a od 2n je definisan spolja sa 0V



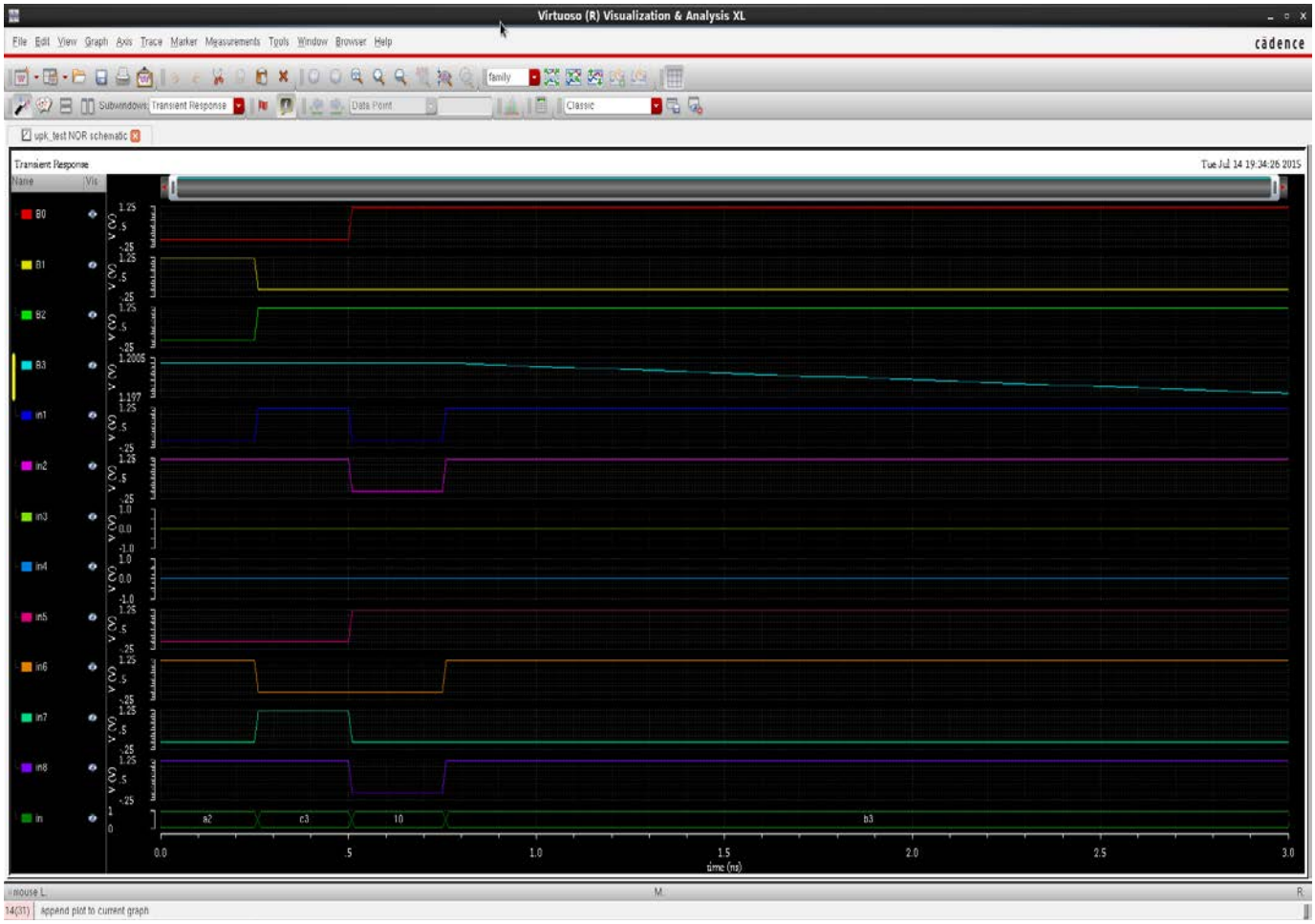
Takođe treba pomenuti da su izvori signala zapravo kao idealni naponski izvori sa 10 milioma rednom otpornošću, zbog toga se javljaju "smetnje" veličine nV (ne mogu se videti na grafiku). Ove otpornosti kao i otpornost viske impedanse je moguće podešavati, detaljnije u priloženom fajl o vektor simulaciji.

Primer 3 Radix vrednosti

Primer izgleda stimulasa sa različitim radixima

signal in[8:1] B[3:0]
io i i
tunit ns
period 0.25
slope 0.01
vih 1.2
radix 44 1111
A2 1010
C3 1100
10 1z01
B3 zzz1

Kada je signal 8-bitni, tj. sa osam ulaza (na primer in[8:1]) , onda mora da se predstavi sa dva ili više simbola. Ovde je osmoulazni signal predstavljen sa dva heksadecimalna broja, to se radi tako što se u radiksu postavi vrednost "radix 44". Takođe je moguće predstaviti 4 bitni signal sa 4 bita, tako što se u radiksu upiše "radix 1111", tada se može upisati visoka impedansa u višebitni vektor.
Grafik prethodnog vektor fajla



Takođe se preporučuje upotreba vektorskog opisa izlaza, kao na grafiku, opcijom analog to digital opisanu u prethodnim poglavljima na strani 45 i 46. Kao što se grafika vidi, vrednosti vektora “in” u .vec fajlu i na grafiku se poklapaju.

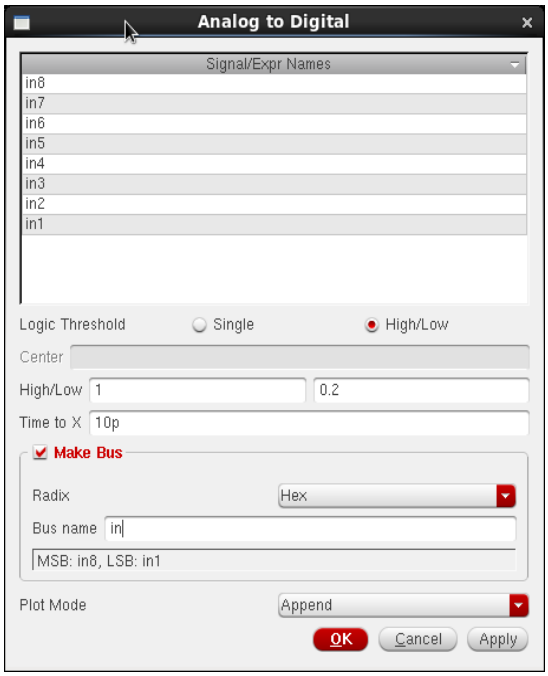


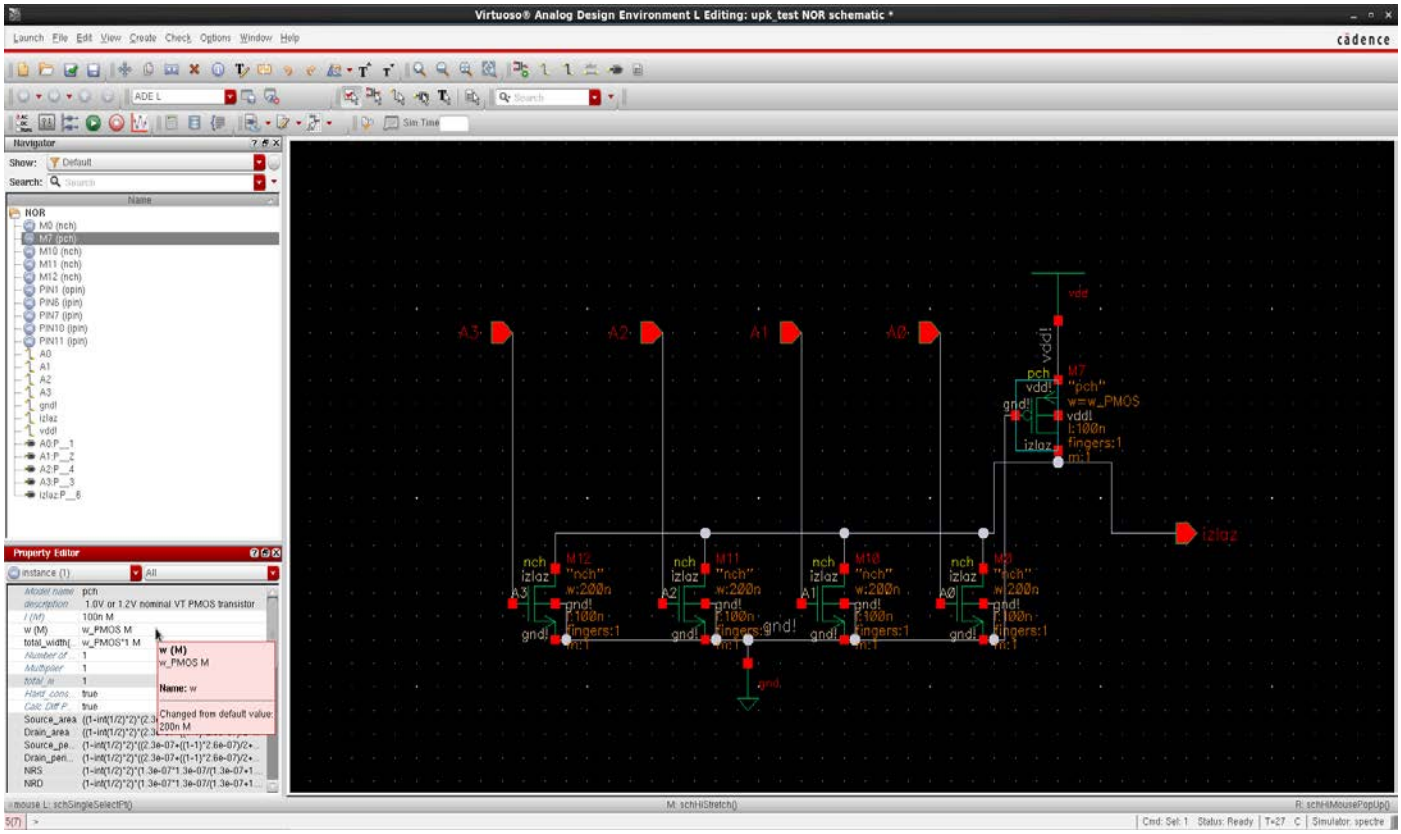
Tabela mogućih vrednosti radix-a

Radix	Vrednost
1	**0, 1**
2	**0-3**
3	**0-7**
4	**0-9, A-F**

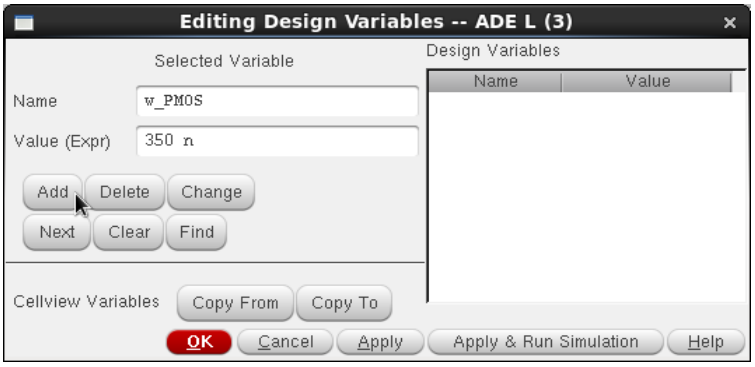
6.6 Korišćenje parametric sweep simulacije

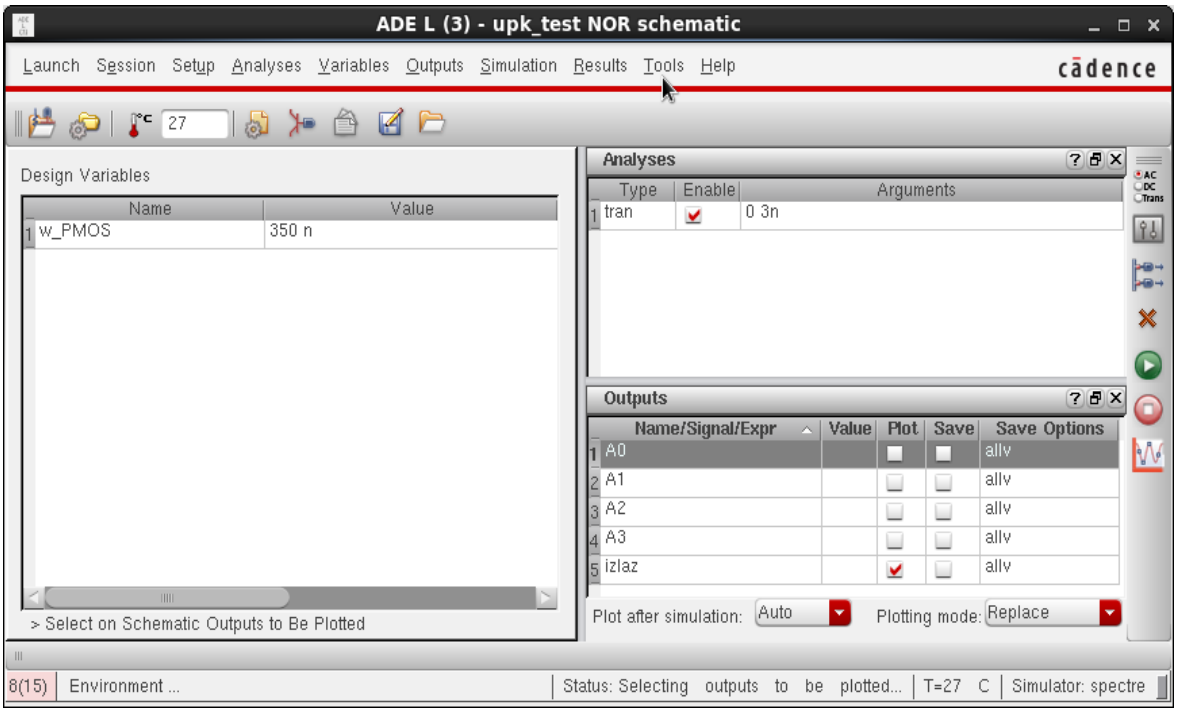
Parametric sweep je korisna simulacija za optimizaciju kola, kada je potrebno meriti izlazne signale u zavisnosti od vrednosti tranzistora. Za demonstraciju je korišćeno NOR kolo gde je menjana vrednost parametara širine kanala pasivnog PMOS tranzistora .U simulaciji je korišćen vektor fajl iz prvog primera.

Prvo je potrebno definisati parametar. U šematiku umesto vrednosti širine PMOS-a upisati neku reč, na primer w_PMOS, kao što je prikazano

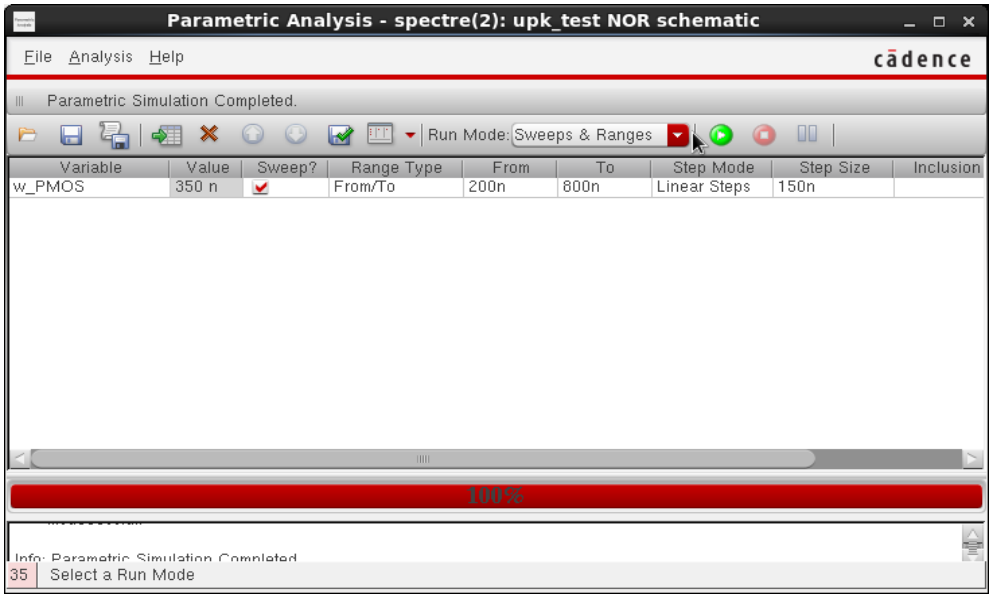


Zatim je potrebno definisati promenljivu w_PMOS u ADEL L simulaciji. To se ostvaruje tako što se ide na V**ariables=>Edit (pored Outputs menija).** Zatim se u polju **Name** stavi ime promenljive, u ovom slučaju je to **w_PMOS.** U polje **Value** upisati neku vrednosti parametra, u ovom slučaju je to 350 n. Zatim kliknuti **Add** i u polju variables u ADEL prozoru će se pojaviti **w_PMOS.**





Dalje je potrebno pokrenuti parametric sweep simulaciju klikom na **Tools=>Parametric Analysis.** U novom prozoru izabrati u **Variable =>w_PMOS** pa zatim odrediti opseg u poljima **From** i **To,** ovde je postavljeno 200n i 800n **(između "200" i "n" ne sme da postoji razmak, mora "200n", inače simulator prijavljuje grešku !!!!)**. Ovde je izabrana **Step Mode=>Linear Steps**, tu se zadaje veličina pojedinačnog koraka u linearnoj razmeri, a moguće je još izabrati da se zadaje broj koraka, logaritamska razmera i ostalo (u help-u je detaljno opisano šta koja stavka radi, po default-u je **Auto**).
Uneti **Step Size,** ovde je to 150n, i klikom na zeleno dugme **Run se pokreće simulacija.**



U simulaciji se samo prikazuje izlazni signal “izlaz” i dobijaju se dijagrami prikazani na slici, gde svaki dijagram predstavlja jednu vrednost tranzistora od 200n do 800n