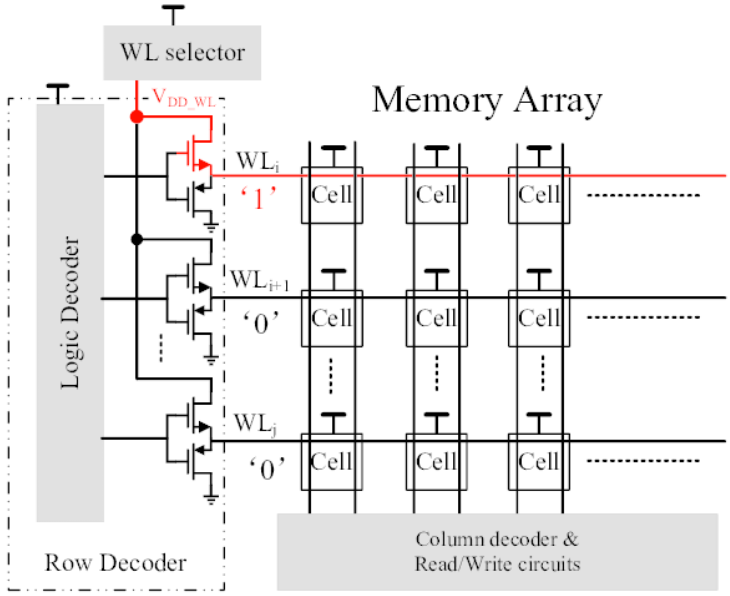
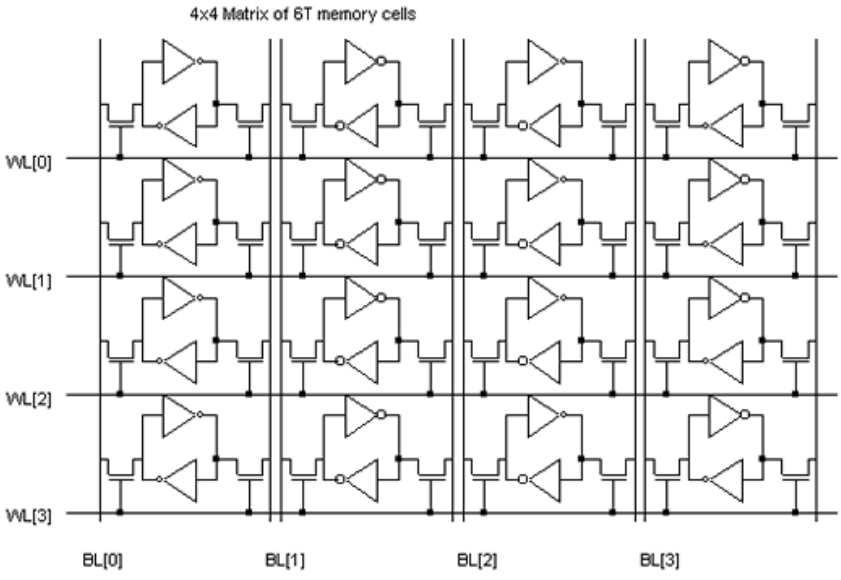
Struktura memorije

Adresni dekoder generira linije riječi: dekodira koja je memorijska lokacija tražena i na nju postavlja logičku jedinicu



Detaljnija struktura

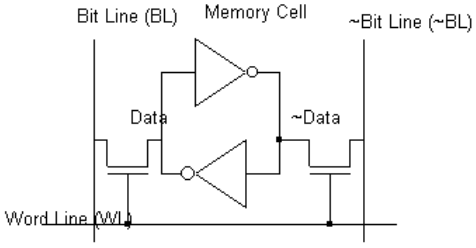
Svaka memorijska ćelija kod statičke je memorije izvedena kao jedan bistabil uparen s odvojnim tranzistorima kojima upravlja linija riječi. Sve ćelije koje čuvaju isti bit po memorijskim lokacijama povezane su linijama bita (na slici to su okomite linije)



Jedna memorijska ćelija statičke memorije: koncept

Kada je na liniji riječi logička 0, odvojni tranzistori (sklopke!) izoliraju podatak zapisan u bistabilu od linije bita.

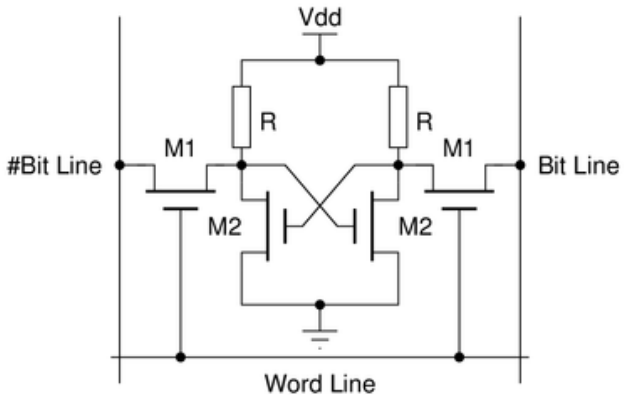
Kada je na liniji riječi logička 1, odvojni tranzistori (sklopke!) dovode podatak zapisan u bistabilu na liniju bita.



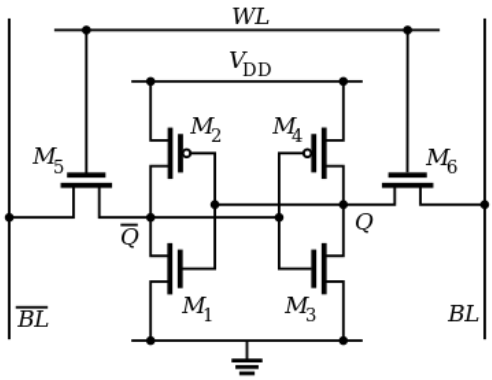
Kako linije riječi generira adresni dekoder, u jednom stupcu samo će jedna ćelija biti omogućena (ona koja odgovara traženoj memorijskoj lokaciji) i ta će ćelija podatak dovesti na liniju bita.

Izvedba memorijske ćelije

Kod statičke memorije, radi se o bistabilu. Kako za pamćenje jednog bita trebamo dosta sklopovlja, statičke memorije imaju manju gustoću (broj bitova po površini memorijskog čipa) od dinamičkih memorija.

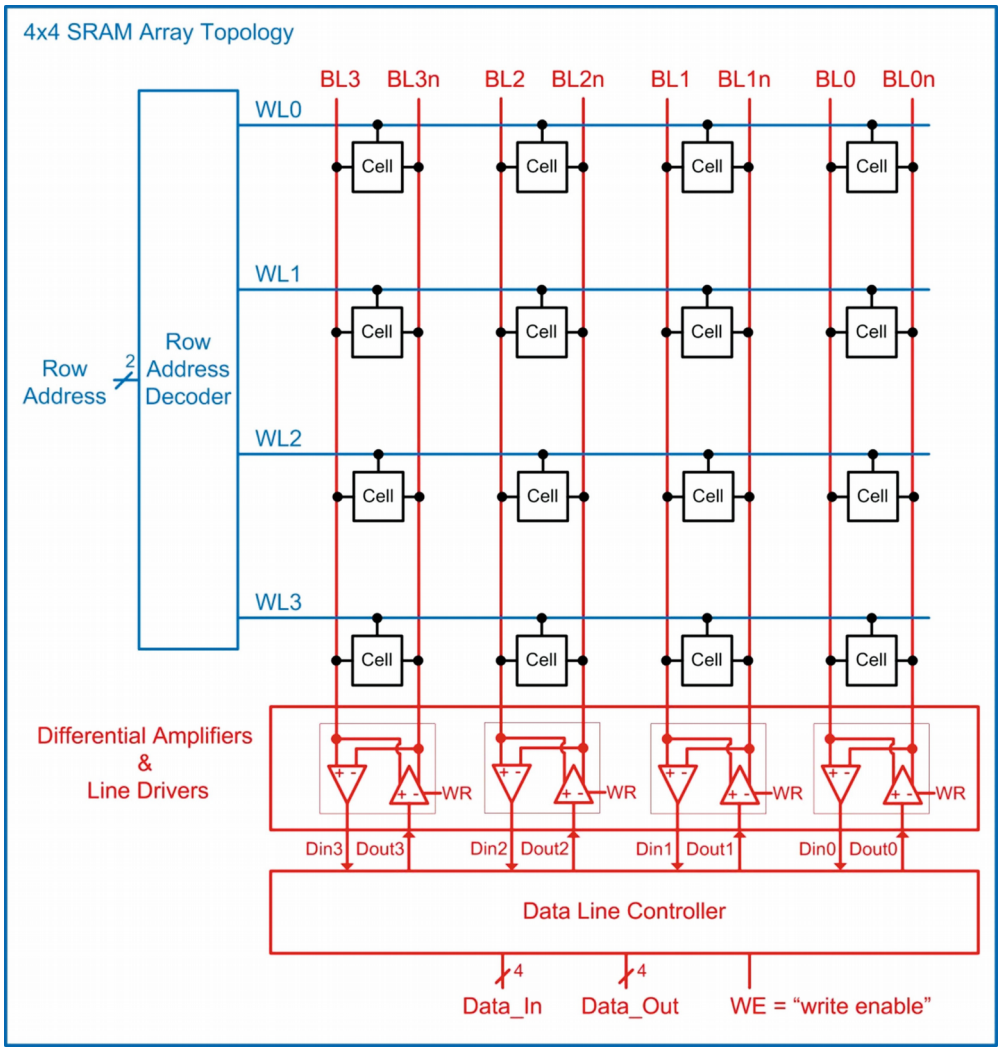


(4-tranzistorska izvedba)



(6-tranzistorska izvedba)

Detaljniji primjer: SRAM 4x4



U memorijskom polju adresni dekoder određuje koliko imamo fizičkih memorijskih lokacija: onoliko koliko adresni dekoder ima izlaza.

Korisnik memoriju promatra kroz logičke memorijske lokacije: primjerice, može koristiti memoriju 8x2, koja čuva dvobitne podatke i ima 8 (logičkih) memorijskih lokacija. Fizička implementacija memorije doista može imati 8 fizičkih riječi, a može ih imati i manje, ali tada na svakoj treba pohraniti bitove više logičkih riječi pa potom odabrati što daje korisniku (kapacitet u svakom slučaju mora biti prikladan).

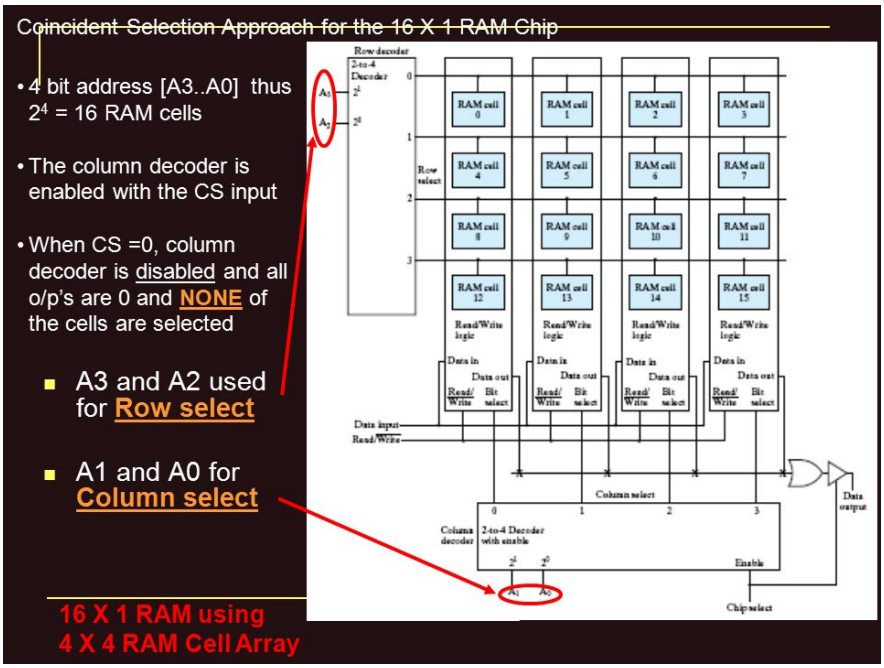
S obzirom na to, memorijsko polje možemo organizirati na sljedeći način:

- 2D organizacija: broj fizičkih i logičkih riječi je jednak, tj. adresni dekoder dobiva sve adresne bitove, a svaka fizička riječi čuva podatke upravo jedne logičke riječi
- 2½D organizacija: skraćuje duljinu linije bita na način da fizičkih riječi ima manje, ali svaka tada čuva podatke više logičkih riječi; na izlazu trebamo multipleksor/demultipleksor koji će odabrati koja od pohranjenih logičkih riječi ide korisniku (dio adresnih bitova ide na adresni dekoder; ostatak ide na izlazni multipleksor)
- 3D organizacija: pohranu jednog bita po lokacijama riješava uporabom dva dekodera i koincidentnim adresiranjem; potom se to ponavlja za svaki potreban bit.

Koincidentno adresiranje

Koristi se kako bismo smanjili utrošak sklopova potreban za izradu dekodera.

U nastavku je primjer pohrane jednog bita, kroz 16 memorijskih lokacija:



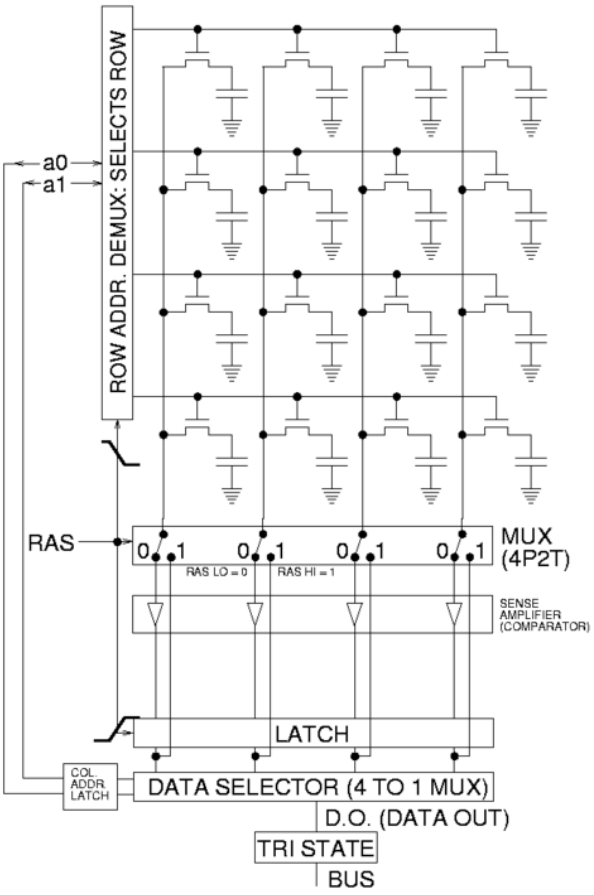
Smanjuje se veličina potrebnog dekodera:

- kada bi bio jedan dekoder 4/16, za ostvarenje bih trebao 16 logičkih sklopova I
- za jedan dekoder 2/4 trebam 4 logička sklopa I; stoga za dva dekodera trebam 2x4=8 što je manje od 16
  - što je broj adresnih bitova veći, dobit je veća; npr.
    - za 10 adresnih bitova (kilobit) imamo 1024 naprema 2x32=64
    - za 20 adresnih bitova (megabit) imamo 1048576 naprema 2x1024=2048

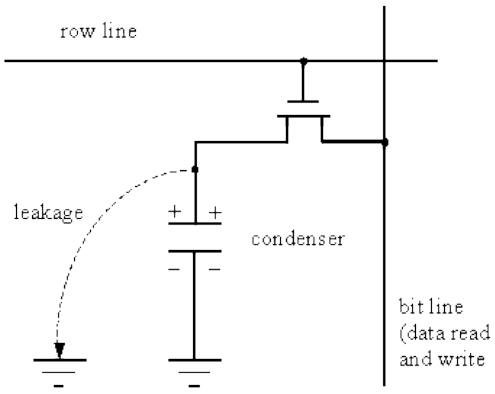
Dinamička memorija: DRAM

Uobičajeno koristi jednotranzistorsku memorijsku ćeliju koja podatak čuva kao naboj zapisan na parazitnom kapacitetu.

Primjer 4x4 memorije:



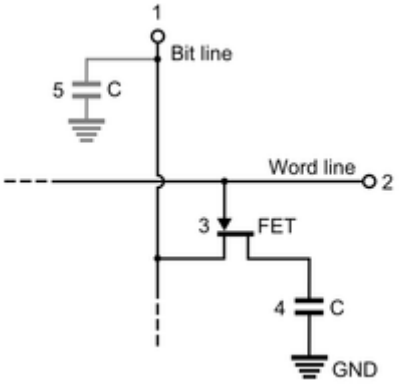
Podatak se čuva na kondenzatoru kao pohranjeni naboj koji s vremenom “curi”:



stoga je zapisani podatak potrebno periodički čitati i ponovno zapisivati u ćeliju – tih problema nismo imali sa statičkom memorijom koja je podatak čuvala u bistabilu.

Čitanje podatka je destruktivno:

- kako je linija bita (na slici vertikala: *bit line*) vrlo dugačka (povezuje ćelije koje čuvaju podatak po svim lokacijama), ona ima popriličan parazitni kapacitet i taj je bitno veći od kapaciteta na kojem se čuva zapisani podatak
- kada se memorijska ćelija omogući (na horizontali: *word line* se postavi 1), naboj zapisan u ćeliji se preraspodijeli i većina ga završi na liniji bita, čime je upisani sadržaj izgubljen
  - stoga nakon svakog čitanja treba osigurati da se pročitani podatak nanovo zapiše u ćeliju



Osvježavanje dinamičke memorije:

- može biti zadaća procesora koji koristi memoriju, pa je njegova odgovornost da pravovremeno čita redom sve memorijske lokacije i ponovno zapisuje pročitani sadržaj (uz glavni posao koji inače radi, a to je izvođenje zadanog programa)
- može biti sklopovski ugrađeno u samu dinamičku memoriju
  - u tom slučaju, takve memorije nazivamo pseudostatičke, jer ih procesor ne treba osvježavati već ih tretira kao statičke memorije

Dinamičke memorije nude bitno veću gustoću zapisa podataka od statičkih. Stoga se danas dominantno koriste u računalima.

Slaganje memorijskih modula

Idemo još pogledati kako od manjih memorijskih modula izgraditi veće memorije (ploča!). Neka imamo potreban broj memorijskih modula 8x2. Trebamo:

- memoriju 8x4
- memoriju 32x2
- memoriju 32x4

Koliko nam memorijskih modula treba i kako ćemo ih spojiti?

Važno: pretpostavka je da svaki memorijski čip ima ulaz CE (engl. *Chip Enable*) i podatkovne izlaze s tri stanja (kad je čip onemogućen, izlazi su u trećem stanju – visoki otpor).