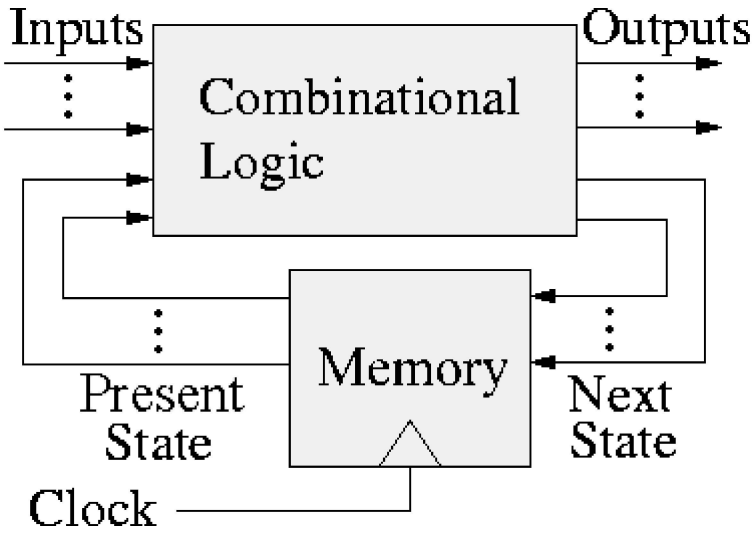
Laboratorul 04 - Logică secvențială

Logică secvențială

Pentru simularea circuitelor vom folosim tot simulatorul din primele laboratoare, http://www.falstad.com/circuit [http://www.falstad.com/circuit].



Toate circuitele studiate până acum au fost circuite combinaționale, adică outputul lor este determinat doar de starea intrărilor. Ele nu au memorie. Având funcția booleană implementată și starea intrărilor vom putea întotdeauna deduce starea ieșirilor.

În circuitele secvențiale însă output-urile nu mai depind exclusiv de starea curentă a intrărilor, ci și de stările anterioare ale intrărilor. Cunoscând funcția booleană a unui circuit secvențiale, nu mai putem deduce output-ul fără să știm și o istorie a stării interne, care la rândul ei depinde de istoria intrărilor.

Folosind circuite cu memorie, se pot construi circuite ce îndeplinesc o funcție utilă precum numărătoare, acumulatoare aritmetice, etc…

Flip Flop

Flip-flopul este un circuit digital capabil să funcționeze ca un bit de memorie. Pini:

- semnal de clock
- una sau două intrări
- un semnal de ieșire
- complementul ieșirii(opțional)
- semnal de clear
- Vcc si ground

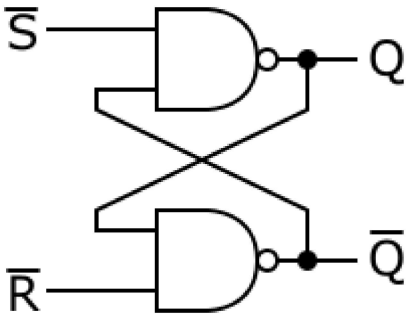
Diferența dintre **Latch** și **Flip-Flop** este aceea că un flip-flop, față de latch, primește la intrare și un semnal de ceas, deci flip-flop-ul este un circuit sincron. Totuși in literatura, acești termeni sunt adesea folosiți interschimbabil. In continuare, vom utiliza doar termenul de **Flip-Flop** si vom nota cu **Q** starea inițială a flip-flop-ului și cu **Q'** starea viitoare, după aplicarea semnalelor de intrare.

Tipuri de flip-flopuri:

- flip-flop SR
- flip-flop T
- flip-flop JK

- flip-flop D

Flip-flop SR



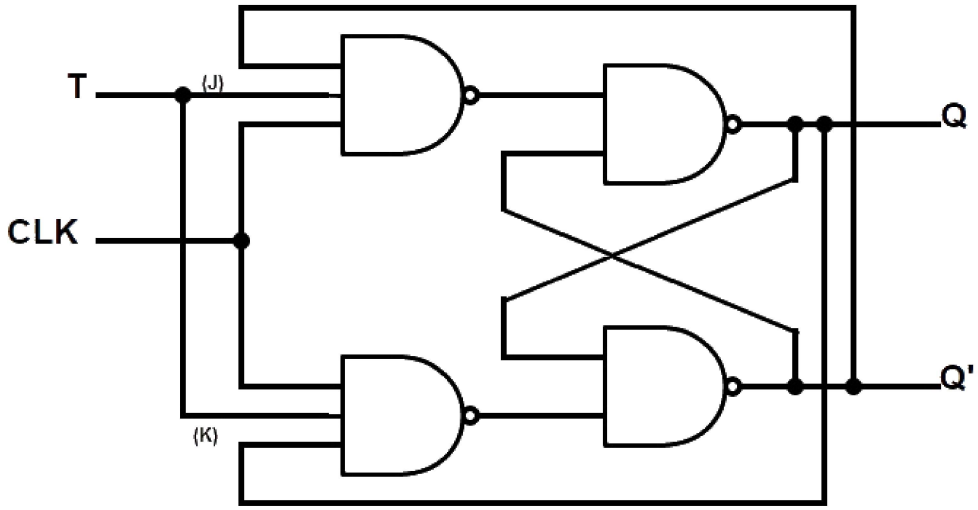
S	R	Q	Q'
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	X
1	1	1	X

Flip-flopul SR are doua inputuri S(set) si R(reset):

- Nici R, nici S nu e activ, outputul rămâne neschimbat
- Când R este activ, outputul este 0
- Când S este activ, outputul este 1
- Când sunt amândouă active starea este nedeterminată

Observatie: Pentru flip-flop-ul S-R există atât o implementare cu porți NAND (ca mai sus) unde intrările sunt **active LOW** (o intrare S sau R se consideră activă când are valoarea 0 logic), precum si o implementare cu porți NOR, cu intrări **active HIGH**.

Flip-flop T

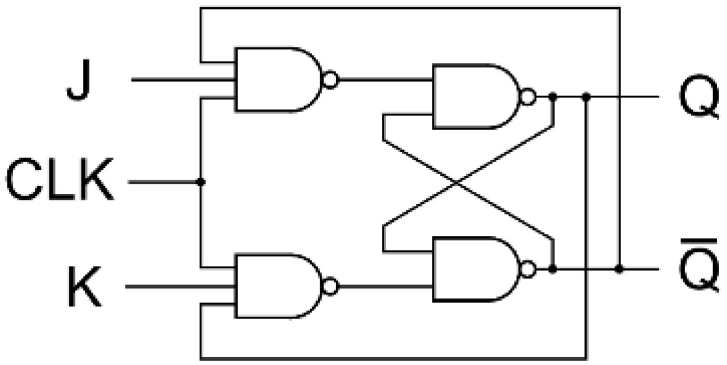


T	Q	Q'
0	0	0
0	1	1
1	0	1
1	1	0

Are un singur pin de intrare:

- Când T este activ ieșirea se schimbă la fiecare puls al ceasului.
- Când T nu este activ ieșirea rămâne neschimbată

Flip-flop JK



J	K	Q	Q'
0	0	0	0
0	0	1	1
0	1	X	0
1	0	X	1
1	1	0	1
1	1	1	0

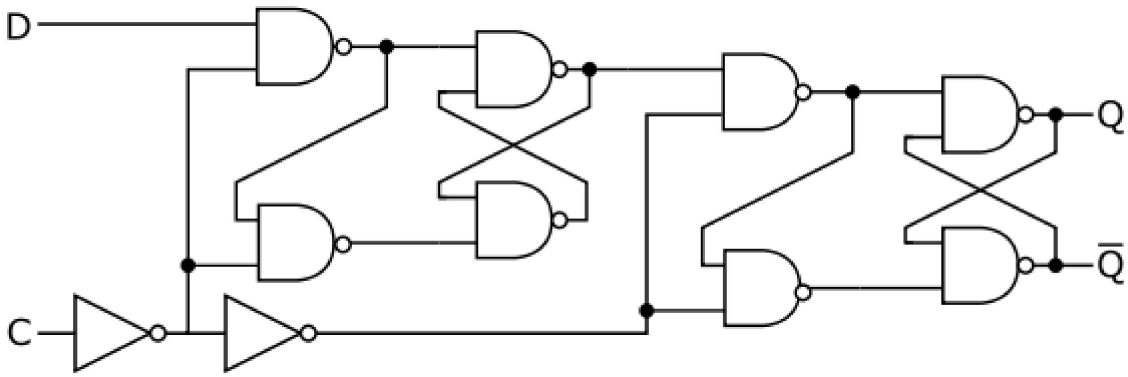
Flip-flopul JK are două intrări J și K:

- Când J==K==1, outputul se schimbă
- Când J!=K, outputul este J

- Când J==K==0, outputul rămâne la fel.

Acesta este un flip-flop universal, putând fi configurat ca orice alt tip de flip flop.

Flip-flop D



Acest tip de circuit basculant bistabil are o intrare de tip D (date) şi o intrare de ceas C (Clk). Pe lângă acestea, el mai poate avea şi două intrări asincrone R şi S care sunt prioritare. Valoarea de la intrare, la momentul tn, apare la ieşire la momenul tn+1, aşa cum se observă şi din tabelul de adevăr de mai jos.

tn		**tn+1**
D	**Q**	**Q'**
0	X	0
1	X	1

Conform tabelului se observă că Qn+1=Dn. Deci CBB tip D întârzie starea, adică ieşirea la momentul tn+1 este aceeaşi cu intrarea la momentul tn (celulă de întârziere sau de memorare). Acest gen de circuit este folosit la realizarea memoriilor RAM statice, a regiştrilor, dar şi la realizarea numărătoarelor.

Numărătoare

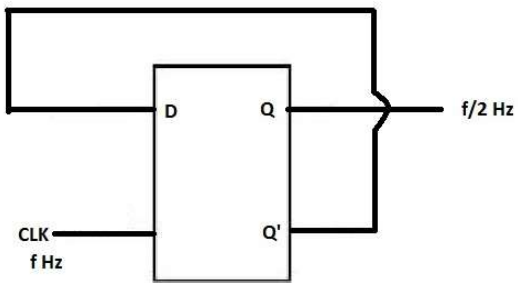
Un numărător este un circuit digital care reține numărul de apariții al unui eveniment.

Tipuri de numărătoare:

- asincrone(ripple) - bistabilii sunt conectați în cascadă.
- sincrone - fiecare bistabil are intrarea sa de ceas, se folosesc porți logice pentru a furniza intrările pentru următoarele circuite.
- Johnson - este o configurație de tip inel care transmite ultima ieșire inversată ca intrare în primul bistabil, astfel se realizează un inel de lungime 2 * numărul de bistabile.
- zecimale - este un numărător care folosește o codare binară pentru a reprezenta cifre zecimale, se va reseta după ce se trece peste 9.

Divizoare de ceas

Un divizor de ceas, numit și **prescaler** este un circuit care primește o frecvență de ceas **f** la intrare, iar la ieșire ne dă aceeași frecvență, divizată cu **n**. Un divizor de ceas ce împarte frecvența la doi se poate realiza

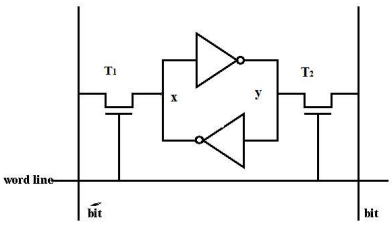


FREQUENCY DIVIDER USING D FLIP FLOP

ușor folosind un **flip-flop D**.

Putem generaliza ușor implementarea, astfel că pentru a face un divizor **f/2^n** vom cascada **n** flip-flop-uri.

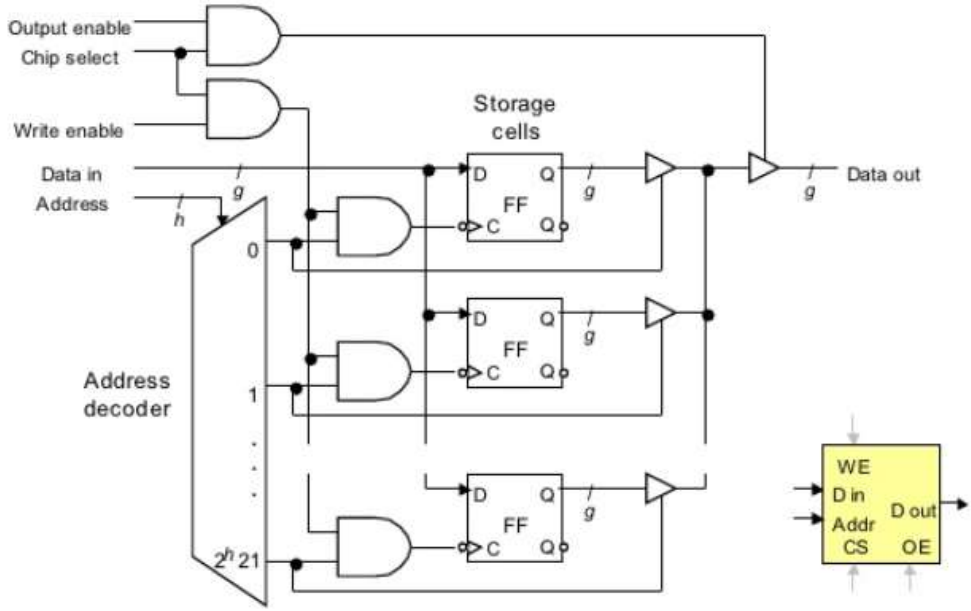
Bit de SRAM



In general, un bit de memorie SRAM se implementează folosind două porți de negare (NOT),necesitând în total 6 tranzistori MOS (2 pentru fiecare poarta NOT și 2 pentru selecție). Pentru simplitate, in cadrul acestui laborator vom folosi câte un flip-flop D pentru fiecare bit de memorie.

Memorite SRAM

In continuare vom considera că vrem să selectăm din memorie biți individuali, dar conceptul se poate extinde ușor la linii (grupari de biți, cum sunt octeții). De asemenea, biții individuali vor fi reținuți folosind



flip-flop-uri D.
O memorie trebuie să ne ofere posibilitatea de a selecta un bit pentru a putea efectua operații de scriere și citire. **Selecția** se realizează cu un decodor care primește o adresă din memorie și selectează un singur bit (exemplu [http://elf.cs.pub.ro/ic/wiki/2018-sem2/lab3#encoderedecodere]). Un bit selectat este singurul a cărei ieșire ajunge pe linia de data out (singurul pe care îl citim), dar și singurul la care poate ajunge un semnal

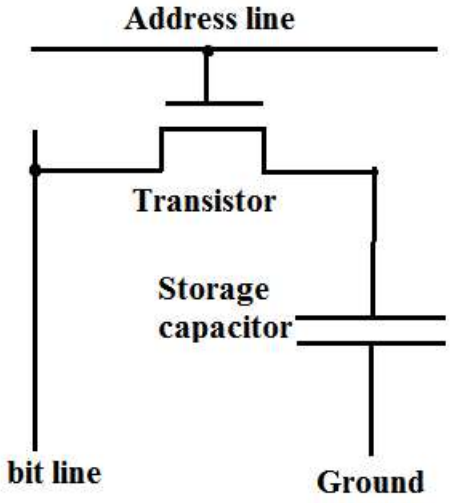
de scriere (**write enable**). **Datele** vor fi trimise la toți flip-flopii simultan. Acest lucru nu ne deranjează, deoarece scrierea efectivă a unui bit se face doar la apariția semnalului de write care va fi primit doar de bitul selectat. **Scrierea** este declanșată de un edge pozitiv al semnalului **write enable**. După cum se poate observa în figură, acest semnal aunge doar la bitul selectat. **Cititrea** necesită doar selectarea unui anumit bit. Acest lucru permite stării acestuia să ajungă pe linia de data out. **Observatie:** In implementarea noastră, nu vom folosi semnalele **Chip select** și **Output Enable**. Practic le vom considera mereu **1** logic. De asemenea, vom inlocui portile tri-state din figură (simbol [https://en.wikipedia.org/wiki/Three-state_logic#/media/File:Tristate_buffer.svg]) cu poți **AND** sau **OR** dupa caz, astfel: Vrem ca doar ieșirea bitului selectat să ajungă pe linia data out, deci vom realiza un **AND** logic intre ieșirea fiecărui bit și semnalul corespunzștor de selectie. Iesirea acestor porți pot fi apoi trimise într-o poartă **OR**. Astfel, logica de output va fi urmaroarea: **data_out = sel_0*out_0 + sel_1*out_1 + .. + sel_n*out_n**, unde **sel_i** este linia de selecție a bitului **i**, iar **out_i** este ieșirea corespunzătoare.

Mai multi biti pe linie

Până în acest punct am selectat un singur bit odată. Dacă ne dorim să realizăm o memorie care selectează, de exemplu, un octet la un moment dat, trebuie să realizăm urmatoarele schimbări:
Un semnal de selecție **sel_i**, în loc să ajugă la un singur bit (atât pentru **write enable** ,cât și pentru **data out**), va ajunge la o grupare de 8 biti.
Linia de **data out** va fi alcatuită din 8 fire.

Memorie DRAM

Un dezavantaj al memorie SRAM este faptul că aceasta ocupă mult spatiu datorită numarului mare de tranzistori. Astfel, nu este avantajos să producem memorii de dimensiuni mari in aceasta tehnologie. Ca un inlocuitor scalabil a aparut memoria DRAM ce necesită un tranzistor si un condensator pentru fiecare bit de

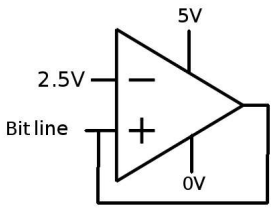


memorie.

Exista totusi un dezavantaj al acestei abordări. Condensatorul care retine bitul de memorie îți poate schimba starea atât cu trecerea timpului (materialele, fiind imperfecte, permit trecerea unor curenți prin dielectricul condensatorului), cat și atunci când este citită starea acestuia. Drept urmare, starea condensatorului trebuie reîmprospătată la intervale regulate de timp.

Un circuit de refresh

Pentru a reîmprospăta starea unui bit este necesar să citim această stare și apoi să o scriem din nou.



pentru aceasta, putem folosi un amplificator opertational astfel:
Tensiunea de pe condensatorul bitului selectat (aceasta fiind fie aproape de 0V fie aproape de 5V) este comparată cu tensiunea de 2.5V. Diferența dintre cele doua, **tensiune_bit - 2.5V** va fi aplificată cu mai multe ordine de marime iar rezultatul va fi limitat intre tensiunile de alimenatre ale amplificatorului operațional(0V-5V). Deci pentru o tensiune mică (aproape de 0V) pe condensator, vom avea ieșirea **max(0V, (0V - 2.5V) * N) = 0V**, unde N este un număr foarte mare. Iar pentru o tensiune de aproape 5V pe condensator vom avea **min(5V, (5V - 2.5V) * N) = 5V**.

Exerciții

În cadrul exercitiilor cu memorii, acestea trebuie să conțină linii de write, data input, data output, selecție pentru linie și pentru coloană. Memoria DRAM va cuprinde de asemenea un semnal de refresh ce permite reîmprospătarea stării bitului selectat (inchide circuitul între bit line si intrarea neinversoare a amplificatorului operațional).

1. (4p) Realizaţi un divizor de ceas 1 la 8.
2. (4p) Realizaţi o memorie SRAM de dimensiune de 4 biți (2 linii) cu cuvânt de 2 biţi folosind flip-flop-uri de tip D.
3. (2p) Realizaţi o memorie DRAM pe 2 linii si 2 coloane.

icalc/laboratoare/lab4.txt · Last modified: 2021/03/23 11:27 by giorgiana.vlasceanu