Laboratorij za načrtovanje integriranih vezij

Univerza *v Ljubljani*
Fakulteta *za elektrotehniko*

Digitalni Elektronski Sistemi

Delovanje realnega vezja

Omejitve modela vezja

Model v VHDLu je poenostavljeno realno vezje

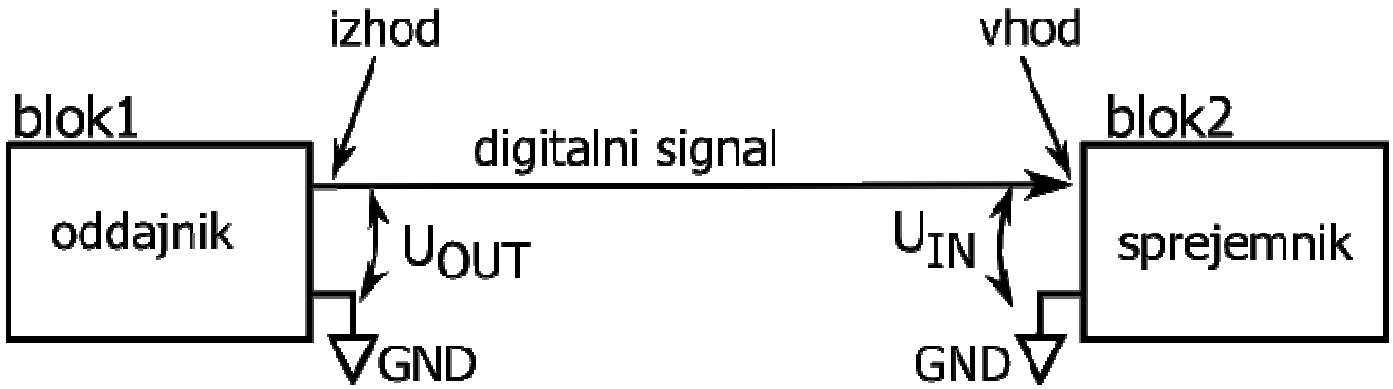
- V modelu imamo logične vrednosti 0 in 1, v vezju pa imamo napetosti
  - 0 = 0V ali nekoliko višja napetost
  - 1 = Vdd (napajanje) ali nekoliko nižja napetost

- V modelu ne upošteva pojavo v realnih flip-flopih
  - če se vhod spremeni istočasno s fronto ure ne moremo napovedati kaj se bo zgodilo z izhodom
    - dobimo eno ali drugo logično stanje, ne glede na vhod ali
    - preide v metastabilno stanje, ki preide počasi v eno izmed stabilnih stanj

Statični red

- Statični red določa pravila za uspešen prenos digitalnih vrednosti med različnimi vezji
  - Povezovanje dig. integriranih vezij v različnih izvedbah (TTL, CMOS, LVCMOS…)

- Dogovor za potenciale, ki določajo nizko ali visoko stanje

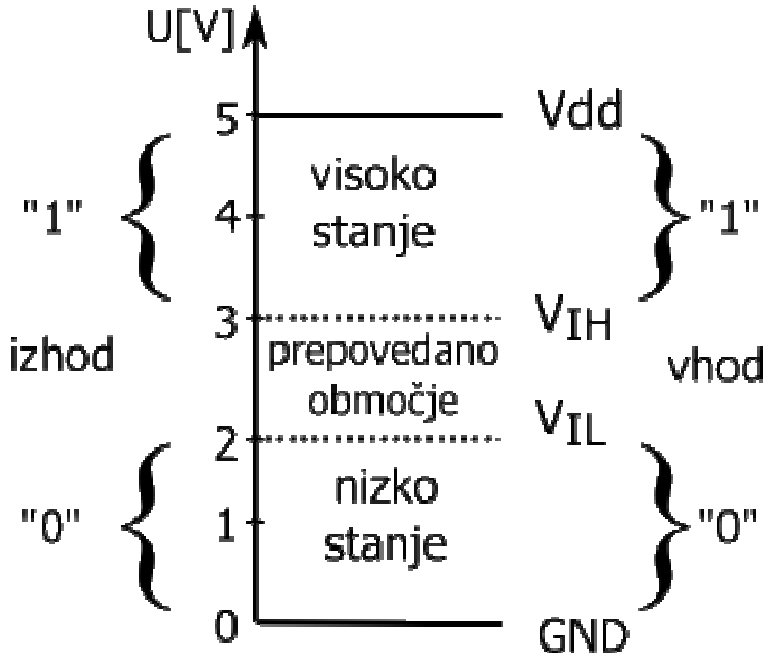


Prepovedano območje potencialov

- Kako intepretiramo signal s potencialom Vdd / 2 ?
  - da se izognemo dvoumni interpretaciji uvedemo prepovedano območje, npr.:

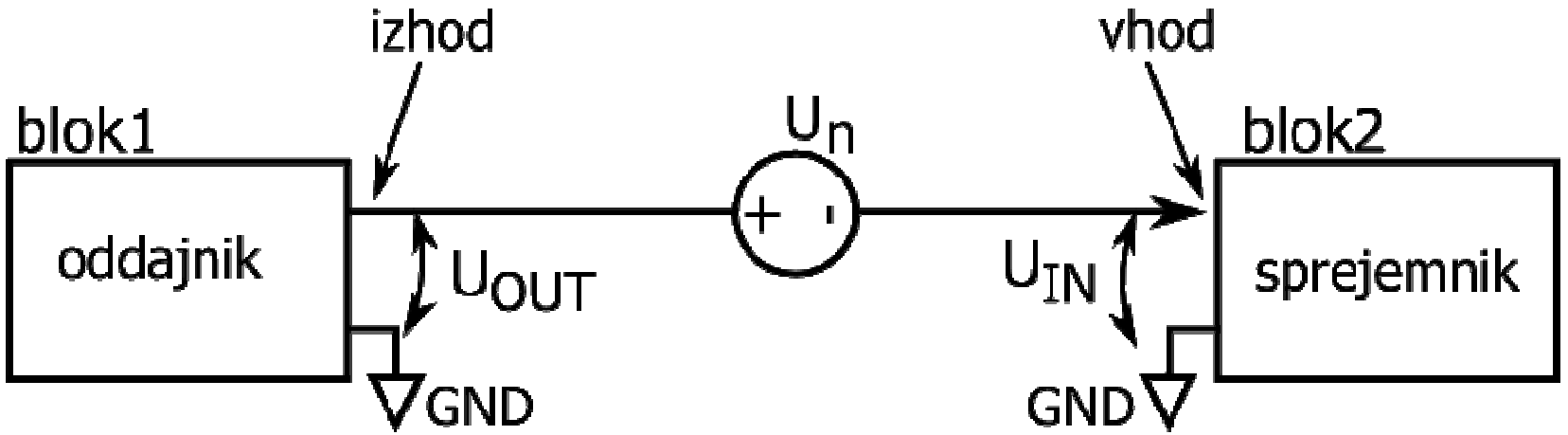
    logična '0': $0V \leq V_L \leq 2V$

    logična '1': $3V \leq V_H \leq 5V$



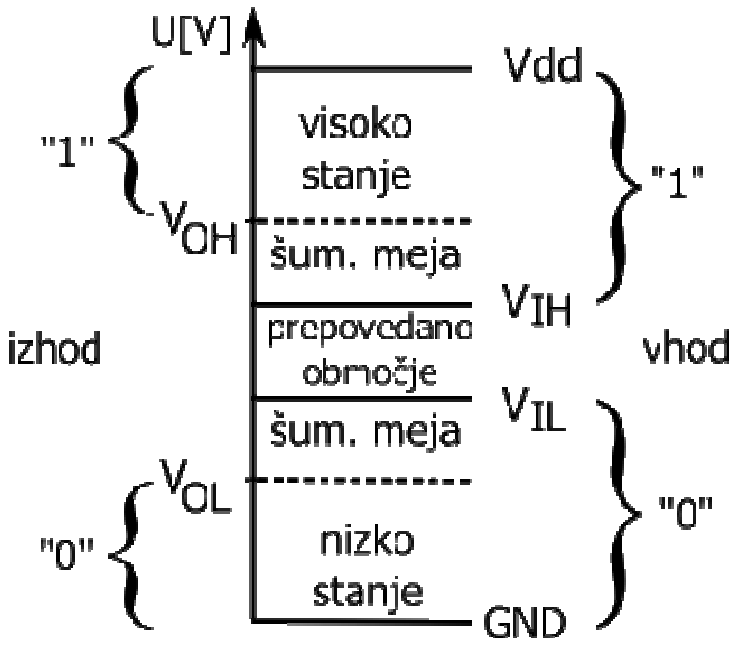
Šum na signalnih povezavah

- Na povezavah v vezju se lahko pojavi šum
  - Šum predstavimo kot dodatno napetost, ki se prišteje ali odšteje od napetosti signala
  - Npr. Un = 0.5V šumne napetosti povzroči
    - CMOS nizko stanje 1V se poveča na 1.5V, kar je še vedno '0'
    - CMOS nizko stanje 2V se poveča na 2.5V, kar je prepovedano območje !



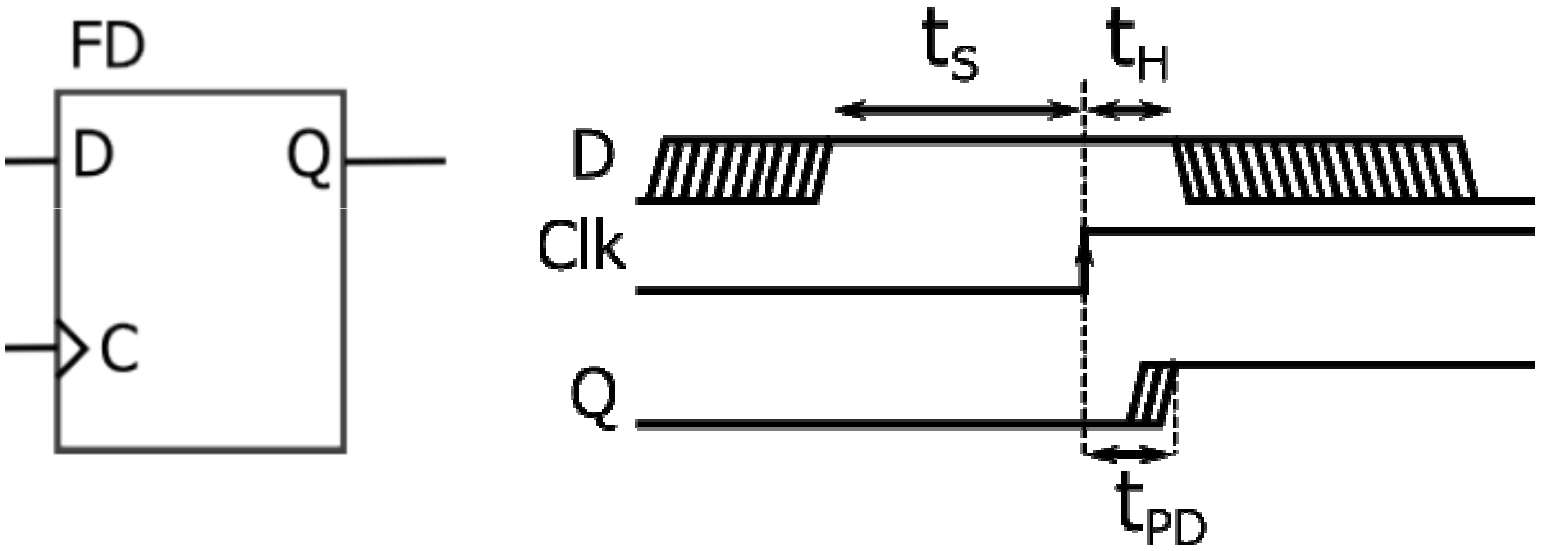
Rešitev: dodamo šumno mejo

- Za oddajnik signala (izhod) določimo manjše dovoljeno območje kot za sprejemnik (vhod)
  - razlika je šumna meja, ki dovoljuje določen nivo šuma brez vpliva na kakovost komunikacije
  - Npr. podatki za 3.3V LVCMOS:



oznaka	pomen	min	max
Vdd	napajalna napetost	$3.0V$	$3.6V$
V_{IH}	vhodni visok nivo	$2V$	$Vdd + 0.3V$
V_{IL}	vhodni nizek nivo	$-0.3V$	$+0.8V$
V_{OH}	izhodni visok nivo	$Vdd - 0.2V$	
V_{OL}	izhodni nizek nivo		$+0.2V$

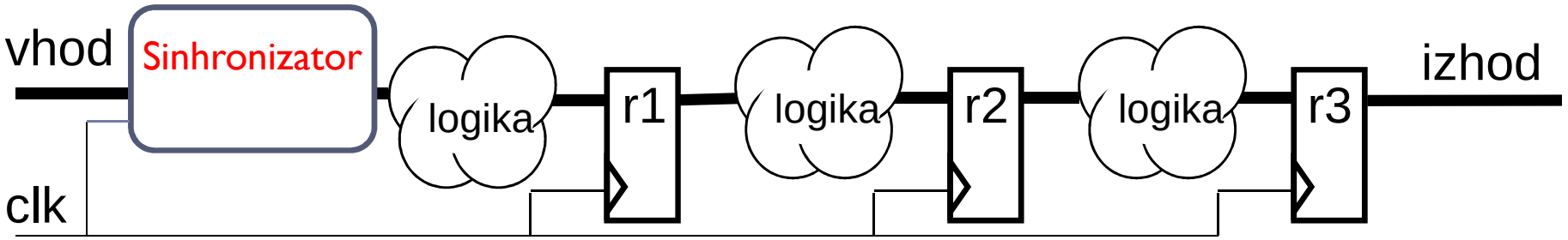
Delovanje flip-flopov



- Kateri signal se je prej spremenil ?
- Dinamični red
  - izogibanje "tekmovanju" med signali

Pravila dinamičnega reda

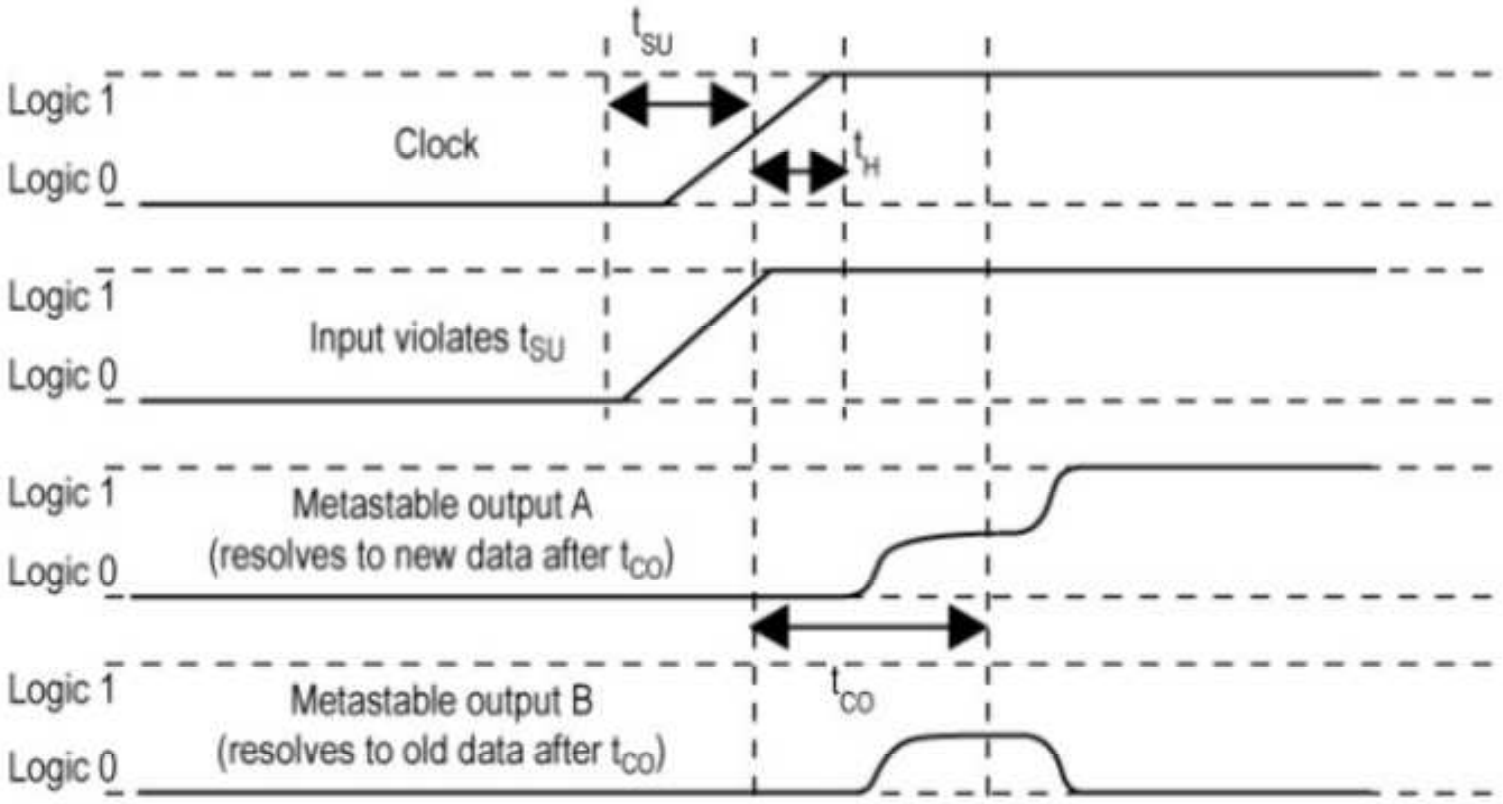
- Ali lahko zagotovimo, da bo vezje vedno delovalo?



- S skrbnim načrtovanjem bo vedno izpolnjen dinamični red
  - ura mora priti do vseh pomnilnih elementov istočasno
  - sinhrono sekvenčno vezje
- Na asinhronem vhodu ne moremo upoštevati pravil !
  - vhodi se ne spreminjajo po zakonitostih naše ure in ne moremo zagotavljati dinamičnega reda (t_S in t_H)

Vezje za sinhronizacijo

- Slaba novica: asinhroni arbiter ne obstaja!
- Poskus rešitve: uporabimo D flip-flop
  - obstaja možnost, da gre v metastabilno stanje
  - čez (nedoločen) čas gre izhod v eno ali drugo stabilno stanje

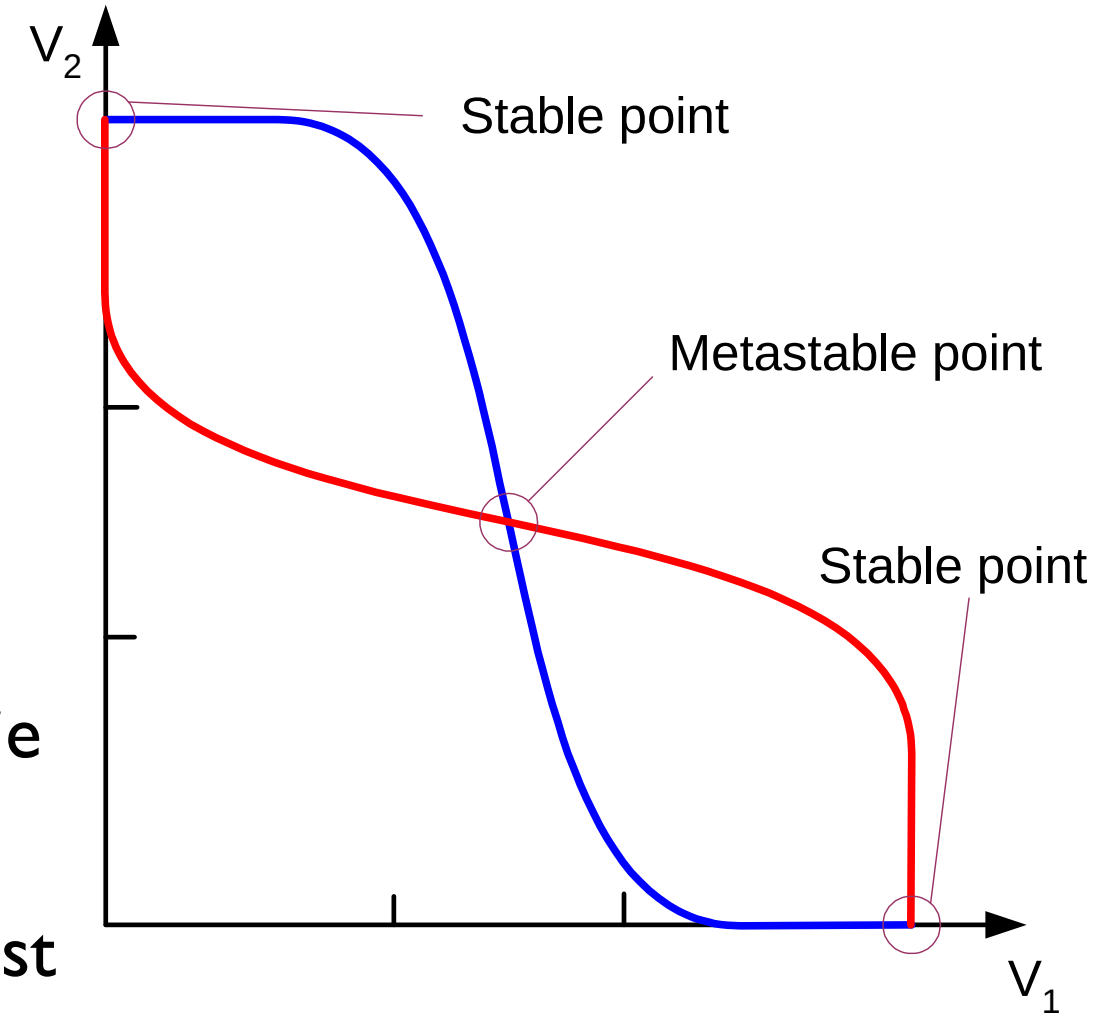


Metastabilno stanje

“0”

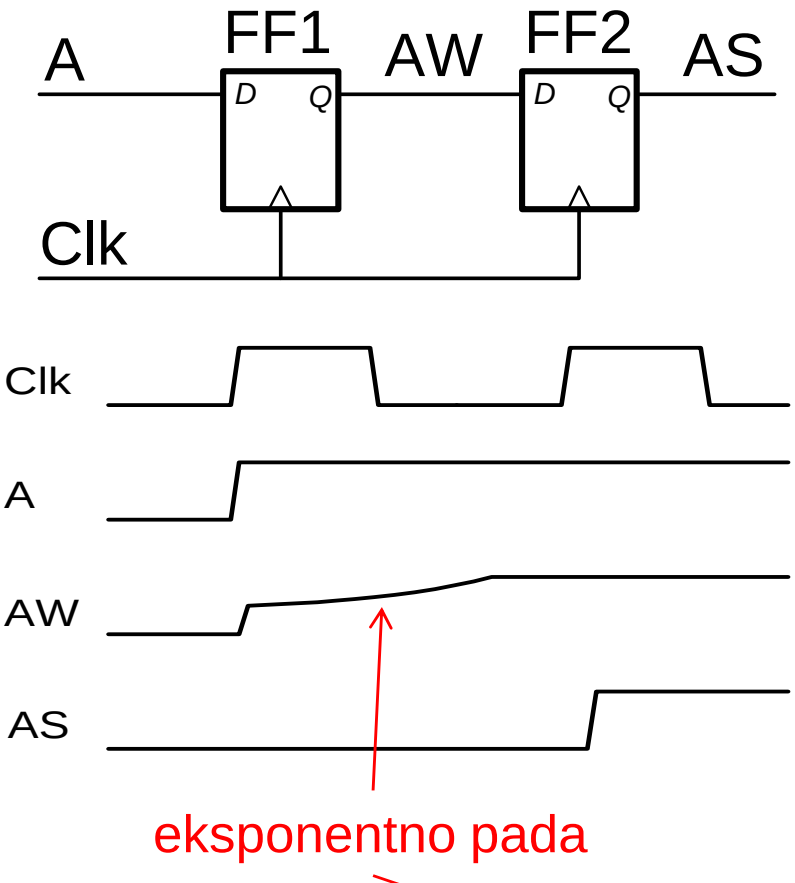
“1”

- lastnost bistabilnih vezij
- čez čas gre v stabilno stanje
- nedoločen čas okrevanja
- verjetnost za metastabilnost eksponentno pada s časom



Sinhronizacija z dvema D flip-flopoma

- metastabilno stanje na izhodu FF1
- stanje se stabilizira na izhodu FF2, če je na voljo dovolj časa
- pri višjih frekvencah ure uporabimo več zaporednih D flip-flopov

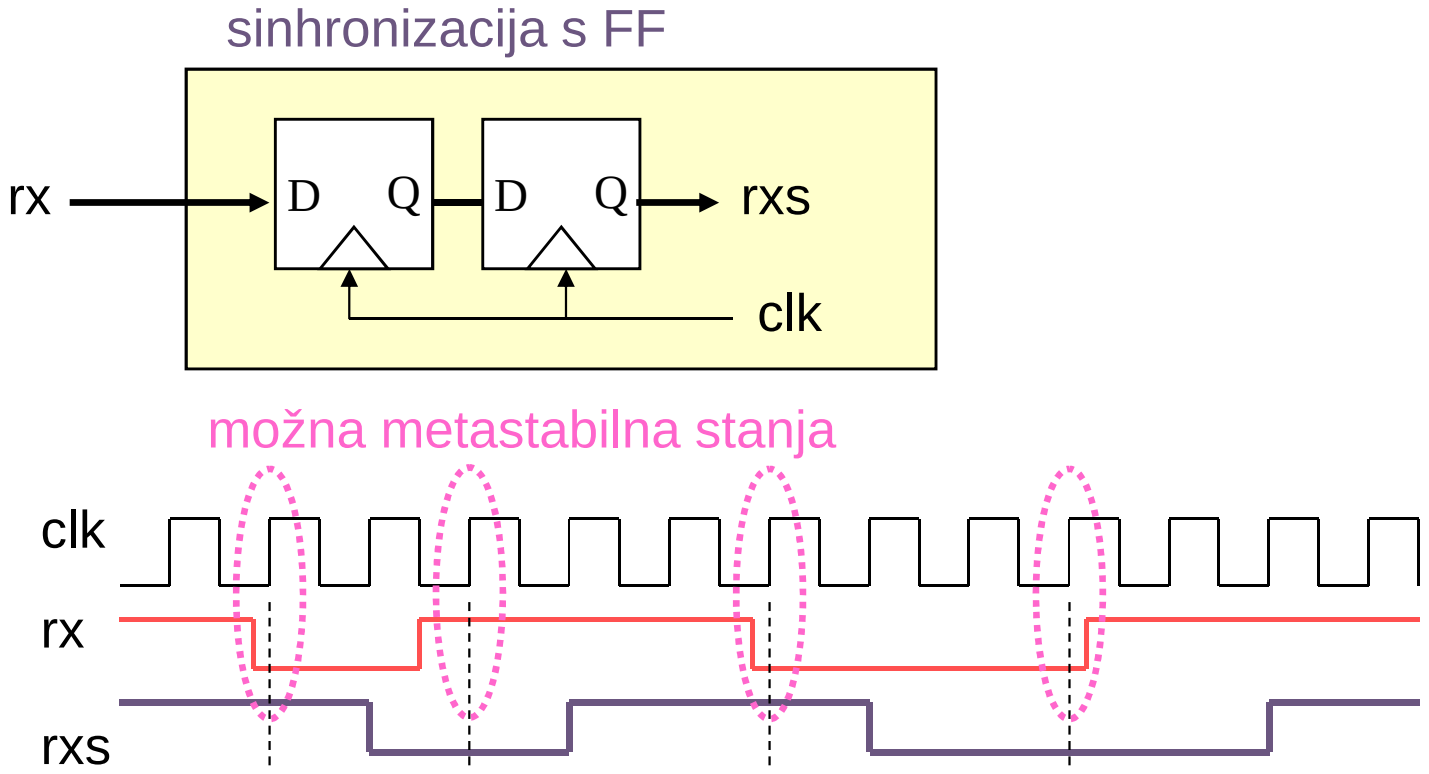


P(napake) = P(metastab stanja) x P(ni še stabilno po t_w)

- Zakasnitev signala je cena za sinhronizacijo, ki se ji ne moremo izogniti !

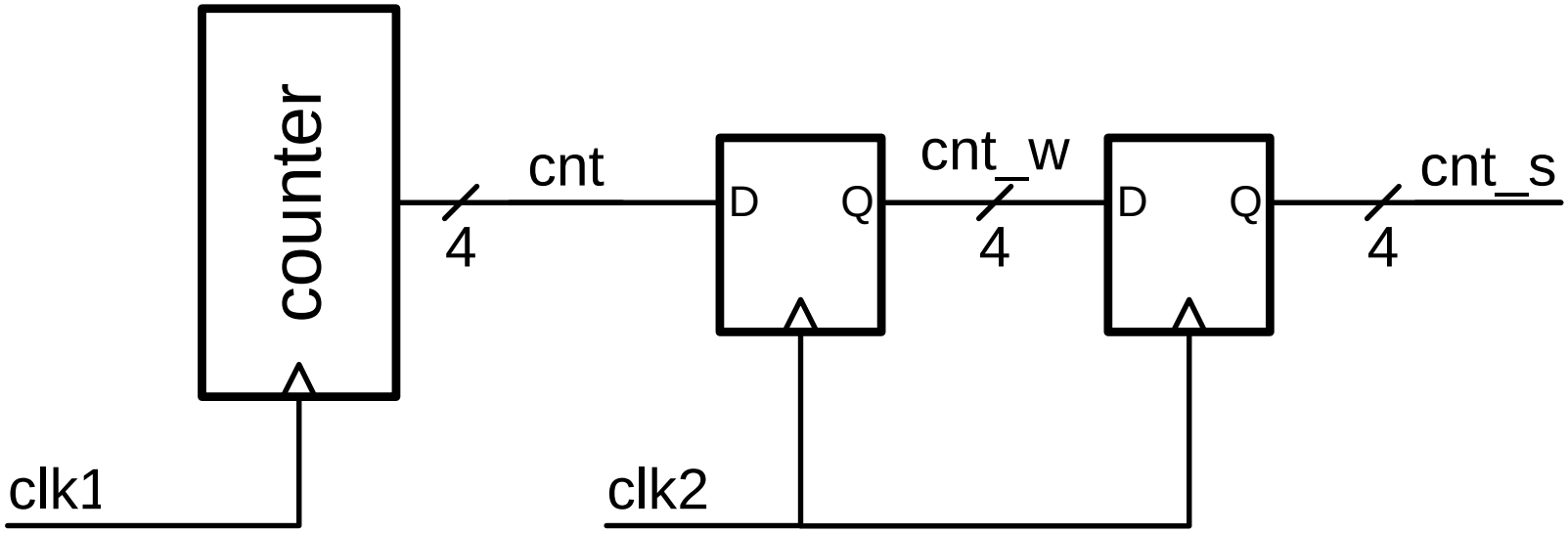
Načrtovanje vmesnikov

- problem komunikacijskih vmesnikov je **sinhronizacija**
  - asinhrone signale vzorčimo z višjo frekvenco ure in jih peljemo čez sinhronizacijsko vezje
- težav z metastabilnostjo ne vidimo na simulaciji !



Sinhronizacija večbitnih signalov

- Ali bo prikazano vezje delovalo ?
  - 4-bitni števec dela z uro clk1, vrednost potrebujemo v vezju, ki preklaplja z uro clk2



- Vsak bit je posebej sinhroniziran
  - Kaj se zgodi ob prehodu števca iz 0111 v 1000 ?
  - Spremenijo se vsi biti, na izhodu je lahko napačen katerikoli…

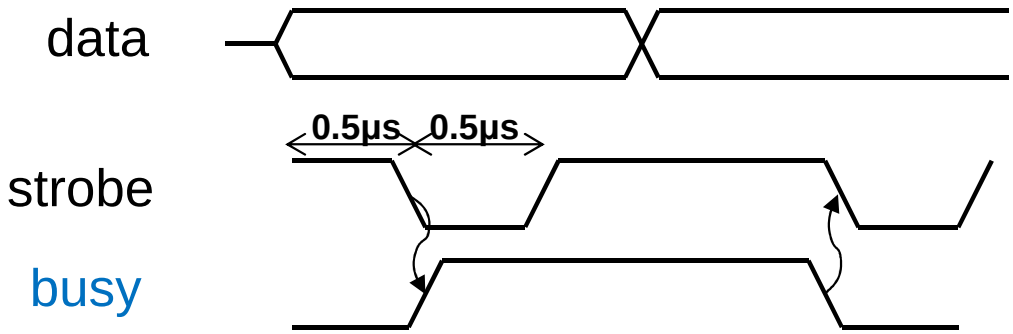
Rešitev 1: Grayev števec

- Binarni števec zamenjamo z Grayevim, ki naenkrat spreminja le en bit
  - v najslabšem primeru bomo prebrali prejšnjo vrednost števca in v naslednjem ciklu branja dobili pravo vrednost
  - **0111** => 1. cikel: **0101**, 2. cikel: **0101** (brez napak)
  - **0111** => 1. cikel : **0111**, 2. cikel : **0101** (napaka)
- Takšno vezje se uporablja na naslovne kazalce v pomnilnikih FIFO
  - kazalec za pisanje šteje z uro clk1
  - Kazalec za branje šteje z uro clk2
  - Razlika kazalcev pove koliko je pomnilnik poln

```
# 0000
# 0001
# 0011
# 0010
# 0110
# 0111
# 0101
# 0100
# 1100
# 1101
# 1111
# 1110
# 1010
# 1011
# 1001
# 1000
# 0000
# 0001
# 0011
# 0010
```

Rešitev 2: komunikacija z usklajevanjem

- Usklajevalni protokol (handshaking) določa časovno okno v katerem je mogoč prenos podatka
- Primer: paralelni vmesnik Centronics
  - 8-bitni asinhroni prenos podatkov za zunanje naprave
  - PC z negativnim impulzom (strobe) označi nov podatek
  - ko naprava zazna strobe, postavi busy in prebere podatek



  - Sinhroniziramo le enobitne kontrolne signale (strobe in busy)