

---

---

---

---

---



## Blatt 5

Wir verwenden die sequentiellen Schaltungen für z.B. endliche Automaten

Damit die Prozessoren komplexere Programme effizient ausführen können ist es notwendig, Daten wie Variablen und Anweisungen zu zwischenspeichern

### Sequentielle Schaltungen

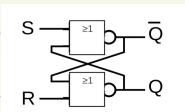
das sind komplizierter als kombinatorische Schaltungen, weil ihre Ausgabe nicht nur von den aktuellen Eingaben sondern auch von vorherigen Eingaben abhängt

### SR-Latch (eine der grundlegenden sequentiellen Schaltungen)

enthält 2 Nor-Gatter, 2 Eingaben S und R, 2 Ausgaben Q und  $\bar{Q}$   
Problem und Nachteil

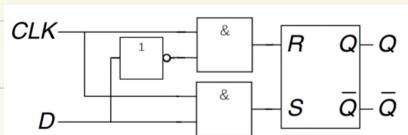
c) Im echten Leben, wenn beide Eingänge des SR-Latches 1 sind, basiert das Ergebnis auf den Herstellungsbedingungen. Das heißt, langsamer Gatter kann das Endergebnis ändern. Das Ergebnis wird dann unstabil.

Irregulärer Zustand: S und R sind beide 1



Setzustand:  $S = 1, R = 0$

Resetzustand:  $S = 0, R = 1$



### D-Latch

eine Erweiterung des SR-Latches (Eingattem mit 3 zusätzlichen Gatter, 1 Not und 2 And). Die Eingaben gehen durch diese Gatter, dann sind 2 Ausgaben davon die Eingaben von SR-Latch. besitzt 2 Eingänge (D und Takteingang CLK für clock). Die Eingabe D bestimmt den nächsten Ausgabewert, der Takteingang kontrolliert, wann sich die Ausgabe ändert.

$CLK = 0 \Rightarrow$  speichern vorherige Werte

$CLK = 1 \Rightarrow Q = D$

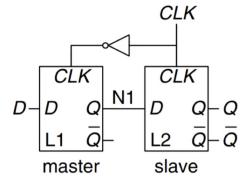
### Vorteil im Vergleich zum SR-Latch

c. Falls  $CLK = 0$ , wird die beide and-Gatter 0 sein und das Latch ist im Speicherzustand  $S = R = 0$ .

Falls  $CLK = 1$ , weil wir D invertieren mit not-Gatter, soll der irreguläre Zustand  $S = R = 1$  nie erreicht werden

$\rightarrow$  D-Latch ist stabiler als SR-Latch

# D - F L i p - F l o p



Unterschied (D-ff und D-Latch): Ausgang Q des D-Latches kann sich irgende wann ändern. Dagegen beim D-ff kann sich der Wert Q ändern kann, nur wenn Taktflanke aufsteigt.

## Funktion von Master-Slave

a)  $L_1 + L_2$  hat keinen Einfluss ??

Für ein D-Latch, falls  $CLK=1$  dann wird immer  $D=Q$ . d.h. Datenwert wird weitergegeben

Falls  $CLK=0$ , wird master  $CLK=1$  und slave  $CLK=0$ . Der Wert von master D wird zu  $N_1$  und slave D weitergegeben. Da slave  $CLK=0$  ist, hat slave Q jetzt den Wert des vorherigen Taktes ( $Q_{prev}$ )

Falls  $CLK=1$ , wird master  $CLK=0$  und slave  $CLK=1$ . Die Ausgabe  $N_1$  hat jetzt den Wert von Q des vorherigen Taktes ( $Q_{prev}$ ) und dieser Wert wird mit dem slave-D-Latch weitergegeben

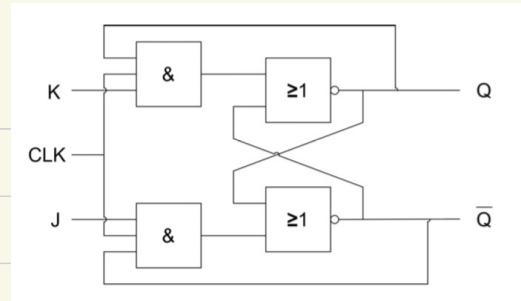
→ in jedem Fall wird der Wert Q immer weitergegeben ohne Einfluss von D

→ wie Abbildung 4 eine Signaländerung von D während  $CLK=1$  beeinflusst Q nicht.

## Unterschied zwischen synchronem und asynchronem

c). Beim synchronen Reset wird Reset-Signal mit einem Takt-Signal synchronisiert. Das Reset-Signal wird wirksam, nur wenn das Takt-Signal (Clock) in einer bestimmten Phase ist (steigende oder fallende Flanke). Dadurch wird Flip-Flop stabil und Timing-Probleme vermeiden. Im Gegensatz dazu wird ein asynchroner Reset-Signal unabhängig vom Takt-Signal angewendet. Es wirkt sofort und setzt den Flip-Flop Zustand zurück, unabhängig von der Taktphase.

# JK - Flip - Flop



Im Vergleich zum D-Flip-Flop kann das JK-FF die Ausgabe außer setzen, zurücksetzen und halten noch umschalten