Lernzettel

Schaltnetze und Schaltwerke: Synchron-/Asynchron-Entwurf, Register- und Zählerwerke

Universität: Technische Universität Berlin
Kurs/Modul: Technische Grundlagen der Informatik (TechGI) - Digitale Systeme
Erstellungsdatum: September 6, 2025

Lernzettel: Schaltnetze und Schaltwerke – Synchron-/Asynchron-Entwurf, Register- und Zählerwerke

(1) Grundbegriffe. Schaltnetze liefern Ausgangswerte ausschließlich aus den Eingangsgrößen und besitzen keinen Speicher. Schaltwerke enthalten Speicherbausteine (Flipflops) und können Zustände über die Zeit hinweg speichern und verändern. In der Praxis realisieren Schaltnetze Funktionen, die Schaltwerke als Sequenzen ausführen.

(2) Synchron- vs. Asynchron-Entwurf.

- Synchroner Entwurf: Alle Flipflops werden durch ein gemeinsames Taktsignal getaktet. Der Zustand wird nahezu gleichzeitig aktualisiert.
- Asynchroner (Ripple) Entwurf: Die Zustandsänderung erfolgt sequentiell, jeder Flipflop wird durch den Ausgang des vorherigen Flipflops getaktet. Schneller in der Schaltung, aber größere Verzögerungen durch Pfad-Skew und Setup-/Hold-Zeiten.

(3) Flipflops als Bausteine. Flipflops speichern einen Bitwert Q und werden durch Taktimpulse aktualisiert. Typische Typen:

- D-Flipflop: $Q^+ = D$. Der nächste Zustand entspricht dem Eingang D zum Takt.
- JK-Flipflop: $Q^+ = J\,\overline{Q} + Q\,\overline{K}$. Mit J/K–Eingängen lassen sich Setzen, Resetten oder Taktieren realisieren.
- T-Flipflop: $Q^+ = Q \oplus T$. Bei T = 1 wird der Zustand invertiert.
- SR-Flipflop: Nur S und R sind relevant; S=1,R=0 setzt, S=0,R=1 löscht, S=R=1 ist ungültig (je nach Implementierung).

(4) Register- und Speicherbausteine. Register bündeln mehrere Flipflops, um mehrere Bits parallel zu speichern. Typische Funktionen:

- Parallel laden vs. serielle Eingabe
- Verschieberegister (Shift-Register) für serielle/gleiche Richtung Verschiebung
- Ladeverhalten: Laden (load) vs. Halten (hold)

(5) Zählerwerke. Zähler realisieren zyklische Zustandsfolgen:

- Ripple-Zähler (asynchron): Die Taktung erfolgt stufenweise von LSB zu MSB; schnelle Realisierung, größere Verzögerungen nach oben.
- Synchrone Zähler: Alle FFs erhalten denselben Taktsignal; die Zählerlogik bestimmt die jeweiligen J/K- oder D-Eingänge.
- Up-/Down-Zähler: Zählrichtung wird über einen Modus-Eingang gesteuert.
- Binär- vs. BCD-Zähler: Unterschiedliche Zählstrukturen, z. B. BCD benötigt nach 9 einen Reset.

- Reset- und Preset-Funktionen: Startwerte oft 0 oder spezielle Werte.

(6) Entwurfsmethoden für Sequenzen.

- Zustandsgleichungen: Der nächste Zustand Q^+ ist Funktion des aktuellen Zustands Q und der Eingaben.
- Moore- vs. Mealy-Zustandsautomaten: Moore-Ausgabe hängt nur vom Zustand ab; Mealy-Ausgabe hängt zusätzlich von Eingaben ab.
- Vorgehen: Spezifikation → Zustandsdiagramm → Zustands- und Logikimplementation mit Flipflops.

(7) Beispielschaltungen.

- Beispiel A: 3-Bit binärer Up-Counter (synchron oder asynchron realisiert). LSB toggelt; MSBs folgen mit Pfadbzwang.
- Beispiel B: Sequencer mit 4 Bits, der nacheinander verschiedene Zustände durchläuft (Moore/Mealy-Variante erläutern).

(8) Formeln zu Schlüsselbausteinen. *D-Flipflop*:

$$Q^+ = D$$

JK-Flipflop:

$$Q^+ = J\,\overline{Q} + Q\,\overline{K}$$

T-Flipflop:

$$Q^+ = Q \oplus T$$

(9) Beispielhafte Zählergleichungen (synchroner Binärzähler). Für einen 3-Bit synchronen Zähler mit D-Flipflops gilt typischerweise eine Zuweisung der nächsten Zustände über D-Inputs, z. B.:

$$D_0 = \overline{Q_0}, \quad D_1 = Q_1 \oplus Q_0, \quad D_2 = Q_2 \oplus (Q_0 \wedge Q_1)$$

Dies realisiert eine Zählfolge von 000, 001, 010, 011, 100, 101, 110, 111, 000, ...

(10) Timing-Hinweise. Beachte bei synchronen Systemen: - Setup-Zeit: Eingang muss vor dem Takt stabil sein. - Hold-Zeit: Eingang muss nach dem Takt stabil bleiben. - Taktdominanz: Pfadverzögerungen müssen unterhalb der Setup-/Hold-Grenzen liegen.