Übungsaufgabe

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

Aufgabe 1: Schaltnetze und Schaltwerke: Synchron-/Asynchron-Entwurf

Betrachten Sie Schaltnetze, Synchron- und Asynchron-Entwurf sowie Register- und Zählerwerke. Beantworten Sie die folgenden Unteraufgaben in klarem Textformat.

a) Definieren Sie in knappen Worten den Unterschied zwischen synchronen und asynchronen Schaltnetzen. Nennen Sie je zwei typische Bausteine bzw. Konfigurationen für jeden Entwurfstyp.

b) Welche Bedeutung haben Setup- und Hold-Zeiten in synchronen Schaltungen? Skizzieren Sie grob, wie diese Größen das Design beeinflussen.

c) Erläutern Sie, warum ein asynchrones Reset-Verhalten in Registern zu Problemen führen kann. Welche Designansätze helfen, solche Probleme zu vermeiden?

Aufgabe 2: Register- und Zählerwerke

Diese Aufgabe behandelt Entwurf und Analyse von Registern und Zählern.

a) 3-Bit Up-Counter mit synchronem Reset. Geben Sie die nächsten-Zustand-Funktionen D0, D1, D2 in Abhängigkeit von Q0, Q1, Q2 und Reset an. Formulieren Sie eine Gleichung pro Bit getrennt voneinander.

b) Beschreiben Sie kurz die Zählfolge dieses Zählers bei Reset=0. Welche Zustände durchläuft er zyklisch?

c) Synchroner vs. asynchroner Reset. Geben Sie die nächsten-Zustand-Funktionen für einen Zähler an, der zusätzlich einen asynchronen Reset besitzt. Verwenden Sie dazu die typische Notation, dass bei Reset=1 alle Bits sofort auf 0 gesetzt werden, sonst gelten die Funktionen aus Teil a). Formulieren Sie die drei Ausdrücke analog zu Teil a), aber mit der asynchronen Reset-Semantik.

d) Skizzieren Sie eine kurze Testsequenz, mit der der Zähler von 000 aus gestartet wird, dann zyklisch weiterläuft, und zuletzt ein Reset-Zustand erreicht wird. Beschreiben Sie die Zustandsübergänge in Textform.

Aufgabe 3: Timing-Parameter und Zyklen in Schaltnetzen

In dieser Aufgabe geht es um grundlegende Timing-Parameter und deren Auswirkungen auf das Design von Registerketten.

a) Definieren Sie die folgenden Begriffe und geben Sie deren typische Einheiten an: - Clock-Periodenzeit T, Ta, $t_pd(Propagationsverzgerung), t_su(Setup - Zeit), t_h(Hold - Zeit)$.

b) Gegeben sei eine Anordnung mit zwei hintereinandergeschalteten D-Flip-Flops. Die Parameter sind: - $t_pd1 = 1,2ns, t_pd2 = 1,0ns, -t_su = 0,5ns, t_h = 0,2ns, -TaktperiodeT = 5ns$.

Berechnen Sie das minimale erforderliche $T_mingemderBedingungT_mint_pd1 + t_pd2 + t_su$.

Prüfen Sie anschließend, ob der gegebene T = 5 ns ausreicht, um das Setup-/Hold-Verhalten zu gewährleisten. Begründen Sie Ihre Einschätzung.

Lösungen

Lösung zu Aufgabe 1: Schaltnetze und Schaltwerke

a) Synchron-/Asynchron-Entwurf

- **Synchrones Schaltnetz**:
  - Zustand wird nur zum Zeitpunkt eines Clock-Edges aktualisiert (edge-triggered).
  - Typische Bausteine: D-Flip-Flops, Register, synchroner Zähler.
  - Typische Konfigurationen: synchroner Reset (Reset wird im Clock-Fall wirksam).
- **Asynchrones Schaltnetz**:
  - Zustand kann sich unmittelbar durch Änderungen an Signalen aktualisieren (kein globaler Clock).
  - Typische Bausteine: SR-Latch, JK-Latch, Ripple-Zähler (asynchron).
  - Typische Konfigurationen: asynchroner Reset bzw. asynchrone Set-/Reset-Anforderungen; kein gemeinsamer Clock-Pfad.

b) Setup-/Hold-Zeiten in synchronen Schaltungen

- **Setup-Zeit** t_{su}: Die Eingangssignale müssen vor dem Clock-Edge stabil sein, damit der Flip-Flop korrekte Daten sammelte.
- **Hold-Zeit** t_h: Die Eingangssignale müssen nach dem Clock-Edge weiterhin stabil bleiben, um eine fehlerhafte Übernahme zu verhindern.
- Einfluss auf das Design:
  - Höhere t_{su} oder t_h schränken die maximale Taktrate ein (minimale Taktperiode höher).
  - Einflussgrößen: maximale Propagationsverzögerungen, Clock-Skew, Weglängen in der Schaltung.
  - Designmaßnahmen: selektive Wahl von Flankenarten (edge-triggered), Pipeline-Ansätze, Clock-Tree-Design, Retiming, Nutzung von Flip-Flops mit ausreichenden t_{su}/t_h.

c) Asynchrones Reset-Verhalten in Registern

- **Problem**: Wenn das Reset-Signal asynchron wirkt, können Register in unterschiedlicher Zeit zurückgesetzt werden, was zu inkonsistenten Zuständen im System führt, insbesondere beim Start oder beim Release des Resets.
- Typische Designansätze zur Vermeidung:
  - Verwendung eines *synchronen Resets* (Reset wirkt nur zum Clock-Edge), um Release-Phasen zu koordinieren.
  - Einsatz eines *Reset-Synchronizers* (D-Flip-Flop-Kette) zur Synchronisierung der Reset-Entlassung an alle Register.
  - Sicherstellung einer *single-reset-Quelle* und ggf. Gehäuse eines Conditional Reset, der erst nach einer synchronen Entnahme des Resets aktiv wird.
  - Vermeidung von Reset-Glitches durch saubere Reset-Logik und Entkopplung von Reset-Pfaden.

Lösung zu Aufgabe 2: Register- und Zählerwerke

a) 3-Bit Up-Counter mit synchronem Reset

$$D_0 = \overline{R} \cdot \overline{Q_0}$$
$$D_1 = \overline{R} \cdot (Q_1 \oplus Q_0)$$
$$D_2 = \overline{R} \cdot (Q_2 \oplus (Q_1 \cdot Q_0))$$

Dabei ist R der synchrone Reset (bei Reset=1 setzen alle Bits auf 0).

Alternative äquivalente Schreibweise (als Konditionalform):

$$D_0 = (R = 0) \wedge \neg Q_0, \quad D_1 = (R = 0) \wedge (Q_1 \oplus Q_0), \quad D_2 = (R = 0) \wedge (Q_2 \oplus (Q_1 \wedge Q_0)) .$$

b) Zählfolge bei Reset=0

- Die Zählrichtung ist nach oben, die Folge zyklisch durch alle 3-Bit-Zustände von 000 bis 111.
- Typische Sequenz: 000 → 001 → 010 → 011 → 100 → 101 → 110 → 111 → 000 ...

c) Synchroner vs. asynchroner Reset Für einen Zähler mit zusätzlich asynchronem Reset gilt bei asynchronem Reset die folgende Semantik: Bei Reset=1 werden alle Bits sofort auf 0 gesetzt, ansonsten gelten die aus Teil a) bekannten Funktionen.

$$D_0 = \overline{R_{\text{async}}} \cdot \overline{Q_0}, \quad D_1 = \overline{R_{\text{async}}} \cdot (Q_1 \oplus Q_0), \quad D_2 = \overline{R_{\text{async}}} \cdot (Q_2 \oplus (Q_1 \cdot Q_0)) .$$

Hierbei steht $R_{\text{async}} = 1$ für aktives asynchrones Reset-Verhalten.

d) Kurze Testsequenz (Start 000; zyklisch weiter; danach Reset)

- Startzustand: 000 (Reset=0)
- Takt 1: 001
- Takt 2: 010
- Takt 3: 011
- Takt 4: 100
- Takt 5: 101
- Takt 6: 110
- Takt 7: 111
- Takt 8: 000 (Zyklus geht weiter)
- Danach Apply asynchronen Reset (kurz Reset=1): Q wird sofort 000
- Reset entfernen (Reset=0); nächster Takt erzeugt 001 und der zyklische Betrieb geht weiter.

Lösung zu Aufgabe 3: Timing-Parameter und Zyklen in Schaltnetzen

a) Definitionen und typische Einheiten

- **Clock-Periodenzeit** T: Zeitspanne zwischen zwei aufeinanderfolgenden identischen Clock-Edges (z. B. von einem positiven Flankenwechsel bis zum nächsten). Typische Einheiten: s, ns, ps.
- **Ta**: Typischerweise die Clock-Skew-/Arrival-Time-Variation zwischen verschiedenen Teilen des Systems (Clock-Arrival-Time oder Clock-Skew). Einheiten: ns oder ps.
- t_{pd} (Propagationsverzögerung): Verzögerung vom Clock-Ereignis am Eingang eines Bausteins bis zum Ausgangs-Signal. Einheiten: ns.
- t_{su} (Setup-Zeit): Mindestens benötigte Stabilität der Eingangsdaten vor dem Clock-Edge. Einheit: ns.
- t_h (Hold-Zeit): Mindestens notwendige Stabilität der Eingangsdaten nach dem Clock-Edge. Einheit: ns.

b) Gegeben sei eine Anordnung mit zwei hintereinandergeschalteten D-Flip-Flops

- Parameter: $t_{pd1} = 1{,}2$ ns, $t_{pd2} = 1{,}0$ ns, $t_{su} = 0{,}5$ ns, $t_h = 0{,}2$ ns, $T = 5$ ns.

minimale erforderliche T ($T_{\min}$) gemäß Setup-Anforderung:

$$T_{\min} \geq t_{pd1} + t_{pd2} + t_{su} = 1{,}2 + 1{,}0 + 0{,}5 = 2{,}7 \text{ ns}.$$

Damit gilt: $T = 5$ ns $\geq T_{\min} = 2{,}7$ ns; das Setup-Verhalten ist erfüllt.

Hold-Überprüfung: Für den Hold-Fall gilt typischerweise

$$t_h \leq t_{pd1,min},$$

dabei ist t_{pd1} hier als Typ-/Min-Wert zu interpretieren. Mit $t_h = 0{,}2$ ns und $t_{pd1} = 1{,}2$ ns gilt $0{,}2$ ns $\leq 1{,}2$ ns; die Hold-Beanstandung ist erfüllt. Der Wert der Clock-Periode T beeinflusst die Hold-Bedingung in diesem einfachen Modell zunächst nicht (keine Skew-Berücksichtigung).

Somit ist der gegebene Takt von $T = 5$ ns sowohl für Setup- als auch für Hold-Parameter ausreichend.