DS 16 – Entwurf logischer Schaltungen

Informatik 10 – MINT-Zweig | Informatiksysteme

Lernziel

Ich entwerfe logische Schaltungen in gegebenen Sachzusammenhängen, auch unter Verwendung geeigneter Software (CircuitVerse).

🕘 Stundenverlaufsplan (Doppelstunde, 90 Minuten)

Orientierungsrahmen für 11 SuS: Die ersten 80 Minuten bilden den Kern. Die letzten 10 Minuten sind bewusst flexibel und werden nicht mit neuem Pflichtstoff belegt.

Phase	Zeit	Inhalt/Aktion	Sozialform
Einstieg und Aktivierung	10'	Eine einfache technische Anforderung in Wenn-dann-Sätzen formulieren.	Plenum; kurze Einzeldenkzeit
Fragend-entwickelnde Erarbeitung	25'	Vorgehen vom Sachtext über Variablen und Belegungstabelle zur Schaltung entwickeln.	Unterrichtsgespräch; Beiträge der SuS werden sichtbar festgehalten
Gemeinsame Umsetzung und Sicherung	20'	Die Alarmanlage gemeinsam modellieren und in CircuitVerse prüfen.	Plenum; Live-Coding, Tafelbild oder gemeinsame Simulation
Kurzer Transfer	15'	Die Fahrstuhltür in Tandems bearbeiten.	Einzelarbeit mit anschließendem Tandemabgleich (4 Tandems + 1 Trio)
Auswertung	10'	Lösungswege vergleichen und jede Verbindung aus der Anforderung begründen lassen.	Plenum; Lösung erklären und Fachsprache sichern
Reserve / Vertiefung	10'	Abstimmung mit Vetorecht als Vertiefung.	Flexibel; bei Bedarf für Fragen, Fehlersuche oder Abschluss nutzen

🗣️ Sprachhilfe

- „Für die Eingangswerte … liefert die Schaltung den Ausgangswert …“
- „Der Term/die Tabelle repräsentiert die Schaltung, weil …“
- „Die Rückkopplung bzw. Gruppierung bewirkt …“

📌 Merksätze

- Beim Schaltungsentwurf übersetzt man eine **Anforderung in Textform** zunächst in eine **Belegungstabelle**, danach in eine passende **Gatterkombination**.
- Vorgehen:
    1. Eingangs- und Ausgangsgrößen benennen
    2. Belegungstabelle aufstellen (alle Fälle durchgehen, in denen der Ausgang `1` sein soll)
    3. Passende Gatter auswählen und verschalten
    4. Schaltung testen (alle Belegungen durchprobieren)

💡 Beispiel: Alarmanlage

Eine Alarmanlage soll auslösen (Ausgang = 1), wenn die **Tür geöffnet** ist **und** die Anlage **scharfgeschaltet** ist, **oder** wenn der **Rauchmelder** anschlägt (unabhängig vom Scharfschaltungsstatus).

Eingänge: `Tuer`, `Scharf`, `Rauch` → Ausgang: `Alarm = (Tuer AND Scharf) OR Rauch`

Baut diese Schaltung zur Kontrolle selbst in CircuitVerse nach, bevor ihr mit den Übungen startet.

```python
from itertools import product

def alarm(tuer, scharf, rauch):
    return (tuer and scharf) or rauch

for t, s, r in product([0, 1], repeat=3):
    print(f"Tuer={t}, Scharf={s}, Rauch={r} -> Alarm={int(alarm(t, s, r))}")
```

```
Tuer=0, Scharf=0, Rauch=0 -> Alarm=0
Tuer=0, Scharf=0, Rauch=1 -> Alarm=1
Tuer=0, Scharf=1, Rauch=0 -> Alarm=0
Tuer=0, Scharf=1, Rauch=1 -> Alarm=1
Tuer=1, Scharf=0, Rauch=0 -> Alarm=0
Tuer=1, Scharf=0, Rauch=1 -> Alarm=1
Tuer=1, Scharf=1, Rauch=0 -> Alarm=1
Tuer=1, Scharf=1, Rauch=1 -> Alarm=1
```

📝 Übung 1 – Fahrstuhltür

Eine Fahrstuhltür darf sich nur schließen (Ausgang `Schliessen = 1`), wenn **beide** Lichtschranken (`LS1`, `LS2`) frei sind (`0` = blockiert, `1` = frei) **und** der Zeitgeber (`Zeit`) abgelaufen ist.

1. Stellt die Belegungstabelle auf.
2. Baut die Schaltung in CircuitVerse.
3. Kontrolliert euer Ergebnis mit Python.

```python
# TODO: Übung 1 - Funktion schliessen(ls1, ls2, zeit) definieren
def schliessen(ls1, ls2, zeit):
    pass

for ls1, ls2, zeit in product([0, 1], repeat=3):
    print(f"LS1={ls1}, LS2={ls2}, Zeit={zeit} -> Schliessen={schliessen(ls1, ls2, zeit)}")
```

```
LS1=0, LS2=0, Zeit=0 -> Schliessen=None
LS1=0, LS2=0, Zeit=1 -> Schliessen=None
LS1=0, LS2=1, Zeit=0 -> Schliessen=None
LS1=0, LS2=1, Zeit=1 -> Schliessen=None
LS1=1, LS2=0, Zeit=0 -> Schliessen=None
LS1=1, LS2=0, Zeit=1 -> Schliessen=None
LS1=1, LS2=1, Zeit=0 -> Schliessen=None
LS1=1, LS2=1, Zeit=1 -> Schliessen=None
```

📝 Übung 2 – Abstimmungsschaltung mit Vetorecht

Drei Personen A, B, C stimmen ab (0 = dagegen, 1 = dafür). Ein Antrag ist angenommen, wenn die **Mehrheit** dafür ist – **außer** Person A stimmt dagegen, dann ist der Antrag **in jedem Fall** abgelehnt (Vetorecht).

1. Überlegt euch zunächst in Worten, wann der Ausgang `1` sein muss.
2. Baut die Schaltung in CircuitVerse.
3. Vergleicht mit der Python-Kontrolle.

```python
# TODO: Übung 2 - Funktion antrag_angenommen(a, b, c) definieren
def antrag_angenommen(a, b, c):
    pass

for a, b, c in product([0, 1], repeat=3):
    print(f"A={a}, B={b}, C={c} -> Angenommen={antrag_angenommen(a, b, c)}")
```

```
A=0, B=0, C=0 -> Angenommen=None
A=0, B=0, C=1 -> Angenommen=None
A=0, B=1, C=0 -> Angenommen=None
A=0, B=1, C=1 -> Angenommen=None
A=1, B=0, C=0 -> Angenommen=None
A=1, B=0, C=1 -> Angenommen=None
A=1, B=1, C=0 -> Angenommen=None
A=1, B=1, C=1 -> Angenommen=None
```

Lösung

✅ Lösungen

Lösung

```python
# Lösung Übung 1
def schliessen(ls1, ls2, zeit):
    return ls1 and ls2 and zeit
```

Lösung

```python
# Lösung Übung 2
def antrag_angenommen(a, b, c):
    mehrheit = (a and b) or (b and c) or (a and c)
    return mehrheit and a
```