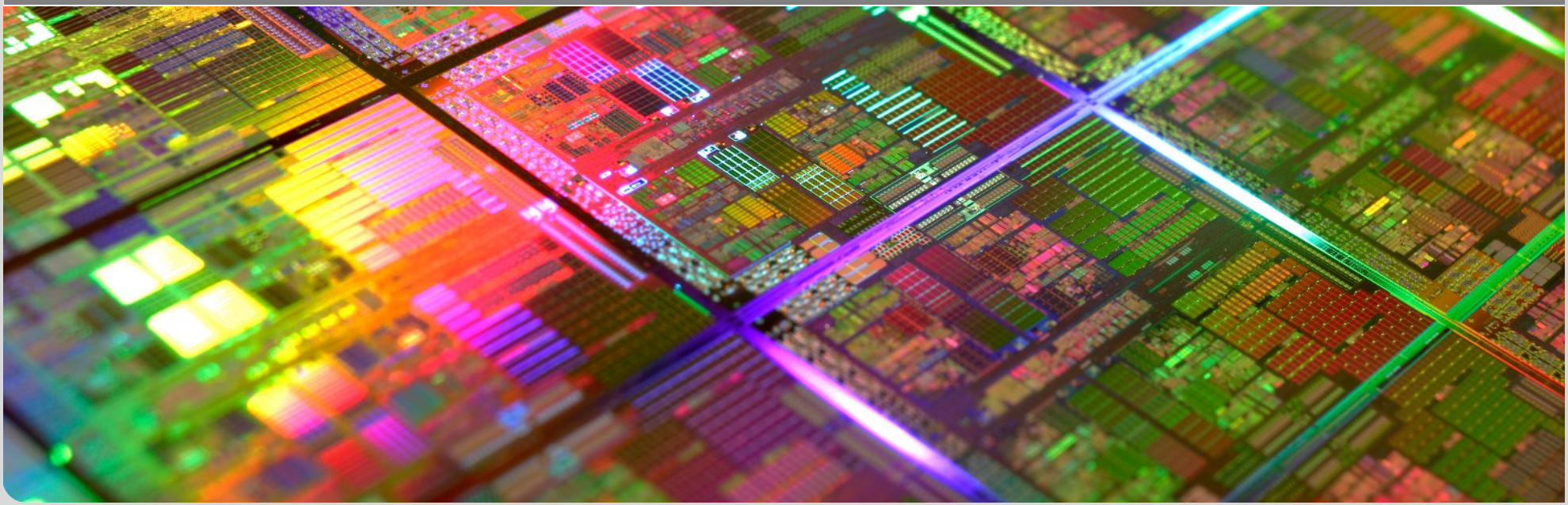
Karlsruher Institut für Technologie

DuE-Tutorien 17 und 18

Tutorien zur Vorlesung “Digitaltechnik und Entwurfsverfahren”

Christian A. Mandery

TUTORIENWOCHE 10 AM 20.01.2012



Heute

- Automaten
- Schaltwerke
- Übungsaufgaben
- Informationen zur Probeklausur

Automaten: Definition

- Kurze Wiederholung aus “Grundbegriffe der Informatik” (1. Semester) und “Theoretische Grundlagen der Informatik” (3. Semester)
- In dieser Vorlesung deterministische endliche Automaten (DEA)
- Tupel $(E, A, Q, \delta, \omega, q_0)$ bestehend aus:
  - dem Eingabealphabet E
  - dem Ausgabealphabet A
  - der Zustandsmenge Q
  - der Überführungsfunktion $\delta : Q \times E \to Q$
  - der Ausgabefunktion ω
  - dem Startzustand q_0

Automaten: Definition

- Kurze Wiederholung aus “Grundbegriffe der Informatik” (1. Semester) und “Theoretische Grundlagen der Informatik” (3. Semester)
- In dieser Vorlesung deterministische endliche Automaten (DEA)
- Tupel $(E, A, Q, \delta, \omega, q_0)$ bestehend aus:
  - dem Eingabealphabet E
  - dem Ausgabealphabet A
  - der Zustandsmenge Q
  - der Überführungsfunktion $\delta : Q \times E \to Q$
  - der Ausgabefunktion ω
  - dem Startzustand q_0

Automaten: Definition

- Kurze Wiederholung aus “Grundbegriffe der Informatik” (1. Semester) und “Theoretische Grundlagen der Informatik” (3. Semester)
- In dieser Vorlesung deterministische endliche Automaten (DEA)
- Tupel $(E, A, Q, \delta, \omega, q_0)$ bestehend aus:
  - dem Eingabealphabet E
  - dem Ausgabealphabet A
  - der Zustandsmenge Q
  - der Überführungsfunktion $\delta : Q \times E \to Q$
  - der Ausgabefunktion ω
  - dem Startzustand q_0

Automaten: Definition

- Kurze Wiederholung aus “Grundbegriffe der Informatik” (1. Semester) und “Theoretische Grundlagen der Informatik” (3. Semester)
- In dieser Vorlesung deterministische endliche Automaten (DEA)
- Tupel $(E, A, Q, \delta, \omega, q_0)$ bestehend aus:
  - dem Eingabealphabet E
  - dem Ausgabealphabet A
  - der Zustandsmenge Q
  - der Überführungsfunktion $\delta : Q \times E \to Q$
  - der Ausgabefunktion ω
  - dem Startzustand q_0

Automaten: Definition

- Kurze Wiederholung aus “Grundbegriffe der Informatik” (1. Semester) und “Theoretische Grundlagen der Informatik” (3. Semester)
- In dieser Vorlesung deterministische endliche Automaten (DEA)
- Tupel $(E, A, Q, \delta, \omega, q_0)$ bestehend aus:
  - dem Eingabealphabet E
  - dem Ausgabealphabet A
  - der Zustandsmenge Q
  - der Überführungsfunktion $\delta : Q \times E \to Q$
  - der Ausgabefunktion ω
  - dem Startzustand q_0

Automaten: Definition

- Kurze Wiederholung aus “Grundbegriffe der Informatik” (1. Semester) und “Theoretische Grundlagen der Informatik” (3. Semester)
- In dieser Vorlesung deterministische endliche Automaten (DEA)
- Tupel $(E, A, Q, \delta, \omega, q_0)$ bestehend aus:
  - dem Eingabealphabet E
  - dem Ausgabealphabet A
  - der Zustandsmenge Q
  - der Überführungsfunktion $\delta : Q \times E \to Q$
  - der Ausgabefunktion ω
  - dem Startzustand q_0

Automaten: Definition

- Kurze Wiederholung aus “Grundbegriffe der Informatik” (1. Semester) und “Theoretische Grundlagen der Informatik” (3. Semester)
- In dieser Vorlesung deterministische endliche Automaten (DEA)
- Tupel $(E, A, Q, \delta, \omega, q_0)$ bestehend aus:
  - dem Eingabealphabet E
  - dem Ausgabealphabet A
  - der Zustandsmenge Q
  - der Überführungsfunktion $\delta : Q \times E \to Q$
  - der Ausgabefunktion ω
  - dem Startzustand q_0

Automaten: Definition

- Kurze Wiederholung aus “Grundbegriffe der Informatik” (1. Semester) und “Theoretische Grundlagen der Informatik” (3. Semester)
- In dieser Vorlesung deterministische endliche Automaten (DEA)
- Tupel $(E, A, Q, \delta, \omega, q_0)$ bestehend aus:
  - dem Eingabealphabet E
  - dem Ausgabealphabet A
  - der Zustandsmenge Q
  - der Überführungsfunktion $\delta : Q \times E \to Q$
  - der Ausgabefunktion ω
  - dem Startzustand q_0

Automaten: Mealy vs. Moore

- **Mealy-Automat**:
  - Ausgabe hängt vom Zustand des Automaten und der Eingabe ab, d.h. $\omega : Q \times E \to A$
  - Zwei Typen:
    1. Zuerst Ausgabe bilden, dann Zustand wechseln (verwende alten Zustand)
    2. Zuerst Zustand wechseln, dann Ausgabe bilden (verwende neuen Zustand)
- **Moore-Automat**:
  - Ausgabe hängt nur vom Zustand des Automaten ab, d.h. $\omega : Q \to A$
- Mealy- und Moore-Automaten sind gleichmächtig und können systematisch ineinander überführt werden
- Meist ist die Modellierung eines gegebenen Problems mit einem der beiden Typen deutlich intuitiver

Automaten: Mealy vs. Moore

- **Mealy-Automat**:
  - Ausgabe hängt vom Zustand des Automaten und der Eingabe ab, d.h. $\omega : Q \times E \to A$
  - Zwei Typen:
    1. Zuerst Ausgabe bilden, dann Zustand wechseln (verwende alten Zustand)
    2. Zuerst Zustand wechseln, dann Ausgabe bilden (verwende neuen Zustand)
- **Moore-Automat**:
  - Ausgabe hängt nur vom Zustand des Automaten ab, d.h. $\omega : Q \to A$
- Mealy- und Moore-Automaten sind gleichmächtig und können systematisch ineinander überführt werden
- Meist ist die Modellierung eines gegebenen Problems mit einem der beiden Typen deutlich intuitiver

Automaten: Mealy vs. Moore

- **Mealy-Automat**:
  - Ausgabe hängt vom Zustand des Automaten und der Eingabe ab, d.h. $\omega : Q \times E \to A$
  - Zwei Typen:
    1. Zuerst Ausgabe bilden, dann Zustand wechseln (verwende alten Zustand)
    2. Zuerst Zustand wechseln, dann Ausgabe bilden (verwende neuen Zustand)
- **Moore-Automat**:
  - Ausgabe hängt nur vom Zustand des Automaten ab, d.h. $\omega : Q \to A$
- Mealy- und Moore-Automaten sind gleichmächtig und können systematisch ineinander überführt werden
- Meist ist die Modellierung eines gegebenen Problems mit einem der beiden Typen deutlich intuitiver

Automaten: Mealy vs. Moore

- **Mealy-Automat**:
  - Ausgabe hängt vom Zustand des Automaten und der Eingabe ab, d.h. $\omega : Q \times E \to A$
  - Zwei Typen:
    1. Zuerst Ausgabe bilden, dann Zustand wechseln (verwende alten Zustand)
    2. Zuerst Zustand wechseln, dann Ausgabe bilden (verwende neuen Zustand)
- **Moore-Automat**:
  - Ausgabe hängt nur vom Zustand des Automaten ab, d.h. $\omega : Q \to A$
- Mealy- und Moore-Automaten sind gleichmächtig und können systematisch ineinander überführt werden
- Meist ist die Modellierung eines gegebenen Problems mit einem der beiden Typen deutlich intuitiver

Automaten: Mealy vs. Moore

- **Mealy-Automat**:
  - Ausgabe hängt vom Zustand des Automaten und der Eingabe ab, d.h. $\omega : Q \times E \to A$
  - Zwei Typen:
    1. Zuerst Ausgabe bilden, dann Zustand wechseln (verwende alten Zustand)
    2. Zuerst Zustand wechseln, dann Ausgabe bilden (verwende neuen Zustand)
- **Moore-Automat**:
  - Ausgabe hängt nur vom Zustand des Automaten ab, d.h. $\omega : Q \to A$
- Mealy- und Moore-Automaten sind gleichmächtig und können systematisch ineinander überführt werden
- Meist ist die Modellierung eines gegebenen Problems mit einem der beiden Typen deutlich intuitiver

Automatengraphen

- Repräsentation eines Automaten als gerichteter Graph zur grafischen Darstellung
- Knoten des Graphen entsprechen Zuständen des Automaten
- Kanten des Graphen entsprechen Zustandsübergängen und werden mit der/den Eingabe(n) beschriftet, die diesen Übergang auslösen
- Angabe der Ausgabe entweder an den Kanten (Mealy-Automat) oder den Zuständen (Moore-Automat)

Automatengraphen

- Repräsentation eines Automaten als gerichteter Graph zur grafischen Darstellung
- Knoten des Graphen entsprechen Zuständen des Automaten
- Kanten des Graphen entsprechen Zustandsübergängen und werden mit der/den Eingabe(n) beschriftet, die diesen Übergang auslösen
- Angabe der Ausgabe entweder an den Kanten (Mealy-Automat) oder den Zuständen (Moore-Automat)

Automatengraphen

- Repräsentation eines Automaten als gerichteter Graph zur grafischen Darstellung
- Knoten des Graphen entsprechen Zuständen des Automaten
- Kanten des Graphen entsprechen Zustandsübergängen und werden mit der/den Eingabe(n) beschriftet, die diesen Übergang auslösen
- Angabe der Ausgabe entweder an den Kanten (Mealy-Automat) oder den Zuständen (Moore-Automat)

Automatengraphen

- Repräsentation eines Automaten als gerichteter Graph zur grafischen Darstellung
- Knoten des Graphen entsprechen Zuständen des Automaten
- Kanten des Graphen entsprechen Zustandsübergängen und werden mit der/den Eingabe(n) beschriftet, die diesen Übergang auslösen
- Angabe der Ausgabe entweder an den Kanten (Mealy-Automat) oder den Zuständen (Moore-Automat)

Ablauftabelle

- Tabellarische Darstellung der Überführungs- und Ausgabefunktion in einer Tabelle
- Gibt zu jeder möglichen Kombination aus Zustand und Eingabe den Folgezustand und die Ausgabe an
- Beispiel:

z^t	e^t	z^{t+1}	y^t
0	0	0	0
0	1	1	0
1	0	1	1
1	1	0	1

Was macht dieser Automat?

Ablauftabelle

- Tabellarische Darstellung der Überführungs- und Ausgabefunktion in einer Tabelle
- Gibt zu jeder möglichen Kombination aus Zustand und Eingabe den Folgezustand und die Ausgabe an
- Beispiel:

z^t	e^t	z^{t+1}	y^t
0	0	0	0
0	1	1	0
1	0	1	1
1	1	0	1

Was macht dieser Automat?

Ablauftabelle

- Tabellarische Darstellung der Überführungs- und Ausgabefunktion in einer Tabelle
- Gibt zu jeder möglichen Kombination aus Zustand und Eingabe den Folgezustand und die Ausgabe an
- Beispiel:

z^t	e^t	z^{t+1}	y^t
0	0	0	0
0	1	1	0
1	0	1	1
1	1	0	1

Was macht dieser Automat?

Automatentafel

- Darstellung der Ablauftabelle in zweidimensionaler Form:
  - Zeilen entsprechen möglichen Zuständen
  - Spalten entsprechen möglichen Eingaben
  - Tabellenzellen geben Folgezustand an
- Angabe der Ausgabe...
  - in den Tabellenzellen der Automatentafel (Mealy-Automat)
  - als zusätzliche Spalte der Automatentafel (Moore-Automat)

Automatentafel

- Darstellung der Ablauftabelle in zweidimensionaler Form:
  - Zeilen entsprechen möglichen Zuständen
  - Spalten entsprechen möglichen Eingaben
  - Tabellenzellen geben Folgezustand an
- Angabe der Ausgabe...
  - in den Tabellenzellen der Automatentafel (Mealy-Automat)
  - als zusätzliche Spalte der Automatentafel (Moore-Automat)

Flussmatrix

- Flussmatrix: Automatentafel bei asynchronen Schaltwerken
  - Besonderheit: Stabile Zustände werden durch Einkreisen gekennzeichnet
  - Ein Zustand ist stabil bezüglich einer Eingabe, wenn er unter dieser Eingabe auch der Folgezustand ist
- Mögliche Typen eines Übergangs in der Flussmatrix:
  - Direkter Übergang: Direkter Folgezustand ist stabil
  - Indirekter Übergang: Direkter Folgezustand ist instabil, aber stabiler Zustand wird nach endlich vielen Zwischenzuständen erreicht
  - Oszillation: Es wird kein stabiler Zustand erreicht (Zyklus aus instabilen Zuständen)
- Erregungsmatrix: Flussmatrix mit eingesetzter Zustandskodierung

Flussmatrix

- Flussmatrix: Automatentafel bei asynchronen Schaltwerken
  - Besonderheit: Stabile Zustände werden durch Einkreisen gekennzeichnet
  - Ein Zustand ist stabil bezüglich einer Eingabe, wenn er unter dieser Eingabe auch der Folgezustand ist
- Mögliche Typen eines Übergangs in der Flussmatrix:
  - Direkter Übergang: Direkter Folgezustand ist stabil
  - Indirekter Übergang: Direkter Folgezustand ist instabil, aber stabiler Zustand wird nach endlich vielen Zwischenzuständen erreicht
  - Oszillation: Es wird kein stabiler Zustand erreicht (Zyklus aus instabilen Zuständen)
- Erregungsmatrix: Flussmatrix mit eingesetzter Zustandskodierung

Flussmatrix

- Flussmatrix: Automatentafel bei asynchronen Schaltwerken
  - Besonderheit: Stabile Zustände werden durch Einkreisen gekennzeichnet
  - Ein Zustand ist stabil bezüglich einer Eingabe, wenn er unter dieser Eingabe auch der Folgezustand ist
- Mögliche Typen eines Übergangs in der Flussmatrix:
  - Direkter Übergang: Direkter Folgezustand ist stabil
  - Indirekter Übergang: Direkter Folgezustand ist instabil, aber stabiler Zustand wird nach endlich vielen Zwischenzuständen erreicht
  - Oszillation: Es wird kein stabiler Zustand erreicht (Zyklus aus instabilen Zuständen)
- Erregungsmatrix: Flussmatrix mit eingesetzter Zustandskodierung

Flussmatrix

- Flussmatrix: Automatentafel bei asynchronen Schaltwerken
  - Besonderheit: Stabile Zustände werden durch Einkreisen gekennzeichnet
  - Ein Zustand ist stabil bezüglich einer Eingabe, wenn er unter dieser Eingabe auch der Folgezustand ist
- Mögliche Typen eines Übergangs in der Flussmatrix:
  - Direkter Übergang: Direkter Folgezustand ist stabil
  - Indirekter Übergang: Direkter Folgezustand ist instabil, aber stabiler Zustand wird nach endlich vielen Zwischenzuständen erreicht
  - Oszillation: Es wird kein stabiler Zustand erreicht (Zyklus aus instabilen Zuständen)
- Erregungsmatrix: Flussmatrix mit eingesetzter Zustandskodierung

Flussmatrix

- Flussmatrix: Automatentafel bei asynchronen Schaltwerken
  - Besonderheit: Stabile Zustände werden durch Einkreisen gekennzeichnet
  - Ein Zustand ist stabil bezüglich einer Eingabe, wenn er unter dieser Eingabe auch der Folgezustand ist
- Mögliche Typen eines Übergangs in der Flussmatrix:
  - Direkter Übergang: Direkter Folgezustand ist stabil
  - Indirekter Übergang: Direkter Folgezustand ist instabil, aber stabiler Zustand wird nach endlich vielen Zwischenzuständen erreicht
  - Oszillation: Es wird kein stabiler Zustand erreicht (Zyklus aus instabilen Zuständen)
- Erregungsmatrix: Flussmatrix mit eingesetzter Zustandskodierung

Flussmatrix

- Flussmatrix: Automatentafel bei asynchronen Schaltwerken
  - Besonderheit: Stabile Zustände werden durch Einkreisen gekennzeichnet
  - Ein Zustand ist stabil bezüglich einer Eingabe, wenn er unter dieser Eingabe auch der Folgezustand ist
- Mögliche Typen eines Übergangs in der Flussmatrix:
  - Direkter Übergang: Direkter Folgezustand ist stabil
  - Indirekter Übergang: Direkter Folgezustand ist instabil, aber stabiler Zustand wird nach endlich vielen Zwischenzuständen erreicht
  - Oszillation: Es wird kein stabiler Zustand erreicht (Zyklus aus instabilen Zuständen)
- Erregungsmatrix: Flussmatrix mit eingesetzter Zustandskodierung

Flussmatrix

- Flussmatrix: Automatentafel bei asynchronen Schaltwerken
  - Besonderheit: Stabile Zustände werden durch Einkreisen gekennzeichnet
  - Ein Zustand ist stabil bezüglich einer Eingabe, wenn er unter dieser Eingabe auch der Folgezustand ist
- Mögliche Typen eines Übergangs in der Flussmatrix:
  - Direkter Übergang: Direkter Folgezustand ist stabil
  - Indirekter Übergang: Direkter Folgezustand ist instabil, aber stabiler Zustand wird nach endlich vielen Zwischenzuständen erreicht
  - Oszillation: Es wird kein stabiler Zustand erreicht (Zyklus aus instabilen Zuständen)
- Erregungsmatrix: Flussmatrix mit eingesetzter Zustandskodierung

Schaltwerke: Definition

- Bisher haben wir nur Schaltnetze betrachtet
  - Ausgabe eines Schaltnetzes hängt nur von der aktuellen Eingabebelegung ab (nach Abklingen von Laufzeiteffekten)
- Heute: Schaltwerke
  - Ausgabe eines Schaltwerks hängt auch von vorhergehenden Eingabebelegungen ab
  - Ein Schaltwerk muss also eine Art “Gedächtnis” haben, damit es sich an frühere Eingabebelegungen “erinnern” kann
  - Realisierung dieses “Gedächtnisses” z.B. durch Flipflops
- Zusammenhang zu Automaten: Jedes Schaltwerk kann als DEA dargestellt und umgekehrt jeder DEA mit einem Schaltwerk realisiert werden

Schaltwerke: Definition

- Bisher haben wir nur Schaltnetze betrachtet
  - Ausgabe eines Schaltnetzes hängt nur von der aktuellen Eingabebelegung ab (nach Abklingen von Laufzeiteffekten)
- Heute: Schaltwerke
  - Ausgabe eines Schaltwerks hängt auch von vorhergehenden Eingabebelegungen ab
  - Ein Schaltwerk muss also eine Art “Gedächtnis” haben, damit es sich an frühere Eingabebelegungen “erinnern” kann
  - Realisierung dieses “Gedächtnisses” z.B. durch Flipflops
- Zusammenhang zu Automaten: Jedes Schaltwerk kann als DEA dargestellt und umgekehrt jeder DEA mit einem Schaltwerk realisiert werden

Schaltwerke: Definition

- Bisher haben wir nur Schaltnetze betrachtet
  - Ausgabe eines Schaltnetzes hängt nur von der aktuellen Eingabebelegung ab (nach Abklingen von Laufzeiteffekten)
- Heute: Schaltwerke
  - Ausgabe eines Schaltwerks hängt auch von vorhergehenden Eingabebelegungen ab
  - Ein Schaltwerk muss also eine Art “Gedächtnis” haben, damit es sich an frühere Eingabebelegungen “erinnern” kann
  - Realisierung dieses “Gedächtnisses” z.B. durch Flipflops
- Zusammenhang zu Automaten: Jedes Schaltwerk kann als DEA dargestellt und umgekehrt jeder DEA mit einem Schaltwerk realisiert werden

Synchrone und asynchrone Schaltw.

- Schaltwerke können synchron und asynchron realisiert werden
- Synchrone Schaltwerke:
  - Alle Zustandsspeicher werden von einem Taktgeber gesteuert
  - Zustandswechsel findet immer nur bei einem Taktsignal statt
    - Flankensteuerung: Schaltvorgang findet bei steigender oder fallender Flanke (oder beiden) statt
    - Zustandssteuerung: Schaltvorgang findet bei $\mathit{Takt} = 1$ oder $\mathit{Takt} = 0$ statt
- Asynchrone Schaltwerke:
  - Mindestens ein Teil des Zustandsspeichers ist nicht getaktet
- Was sind die Vor- und Nachteile der beiden Realisierungsarten?

Synchrone und asynchrone Schaltw.

- Schaltwerke können synchron und asynchron realisiert werden
- Synchrone Schaltwerke:
  - Alle Zustandsspeicher werden von einem Taktgeber gesteuert
  - Zustandswechsel findet immer nur bei einem Taktsignal statt
    - Flankensteuerung: Schaltvorgang findet bei steigender oder fallender Flanke (oder beiden) statt
    - Zustandssteuerung: Schaltvorgang findet bei $Takt = 1$ oder $Takt = 0$ statt
- Asynchrone Schaltwerke:
  - Mindestens ein Teil des Zustandsspeichers ist nicht getaktet
- Was sind die Vor- und Nachteile der beiden Realisierungsarten?

Synchrone und asynchrone Schaltw.

- Schaltwerke können synchron und asynchron realisiert werden
- Synchrone Schaltwerke:
  - Alle Zustandsspeicher werden von einem Taktgeber gesteuert
  - Zustandswechsel findet immer nur bei einem Taktsignal statt
    - Flankensteuerung: Schaltvorgang findet bei steigender oder fallender Flanke (oder beiden) statt
    - Zustandssteuerung: Schaltvorgang findet bei $\mathit{Takt} = 1$ oder $\mathit{Takt} = 0$ statt
- Asynchrone Schaltwerke:
  - Mindestens ein Teil des Zustandsspeichers ist nicht getaktet
- Was sind die Vor- und Nachteile der beiden Realisierungsarten?

Synchrone und asynchrone Schaltw.

- Schaltwerke können synchron und asynchron realisiert werden
- Synchrone Schaltwerke:
  - Alle Zustandsspeicher werden von einem Taktgeber gesteuert
  - Zustandswechsel findet immer nur bei einem Taktsignal statt
    - Flankensteuerung: Schaltvorgang findet bei steigender oder fallender Flanke (oder beiden) statt
    - Zustandssteuerung: Schaltvorgang findet bei $\mathit{Takt} = 1$ oder $\mathit{Takt} = 0$ statt
- Asynchrone Schaltwerke:
  - Mindestens ein Teil des Zustandsspeichers ist nicht getaktet
- Was sind die Vor- und Nachteile der beiden Realisierungsarten?

Synchrone und asynchrone Schaltw.

- Schaltwerke können synchron und asynchron realisiert werden
- Synchrone Schaltwerke:
  - Alle Zustandsspeicher werden von einem Taktgeber gesteuert
  - Zustandswechsel findet immer nur bei einem Taktsignal statt
    - Flankensteuerung: Schaltvorgang findet bei steigender oder fallender Flanke (oder beiden) statt
    - Zustandssteuerung: Schaltvorgang findet bei $\mathit{Takt} = 1$ oder $\mathit{Takt} = 0$ statt
- Asynchrone Schaltwerke:
  - Mindestens ein Teil des Zustandsspeichers ist nicht getaktet
- Was sind die Vor- und Nachteile der beiden Realisierungsarten?

Probleme asynchroner Schaltwerke

- Problematik: Wettläufe
  - Wenn sich bei einem Übergang mehrere Zustandsvariablen ändern, geschieht diese Änderung der Zustandsvariablen nicht exakt gleichzeitig
  - Schaltwerk nimmt implizite Zwischenzustände an, in denen eine Zustandsvariable geändert wurde, aber die andere noch nicht (**Wettlauf**)
  - Falls der Zwischenzustand einen anderen Folgezustand liefert oder selbst stabil ist, wird das Ziel des ursprünglichen Übergangs nicht erreicht (**kritischer Wettlauf**)
- Lösung: Wettlauffreie Zustandskodierung
  - Bei jedem Übergang darf sich maximal eine Zustandsvariable ändern
  - Eventuell mehr Zustandsspeicher notwendig als mit minimaler Zustandskodierung

Probleme asynchroner Schaltwerke

- Problematik: Wettläufe
  - Wenn sich bei einem Übergang mehrere Zustandsvariablen ändern, geschieht diese Änderung der Zustandsvariablen nicht exakt gleichzeitig
  - Schaltwerk nimmt implizite Zwischenzustände an, in denen eine Zustandsvariable geändert wurde, aber die andere noch nicht (**Wettlauf**)
  - Falls der Zwischenzustand einen anderen Folgezustand liefert oder selbst stabil ist, wird das Ziel des ursprünglichen Übergangs nicht erreicht (**kritischer Wettlauf**)
- Lösung: Wettlauffreie Zustandskodierung
  - Bei jedem Übergang darf sich maximal eine Zustandsvariable ändern
  - Eventuell mehr Zustandsspeicher notwendig als mit minimaler Zustandskodierung

Probleme asynchroner Schaltwerke

- Problematik: Wettläufe
  - Wenn sich bei einem Übergang mehrere Zustandsvariablen ändern, geschieht diese Änderung der Zustandsvariablen nicht exakt gleichzeitig
  - Schaltwerk nimmt implizite Zwischenzustände an, in denen eine Zustandsvariable geändert wurde, aber die andere noch nicht (**Wettlauf**)
  - Falls der Zwischenzustand einen anderen Folgezustand liefert oder selbst stabil ist, wird das Ziel des ursprünglichen Übergangs nicht erreicht (**kritischer Wettlauf**)
- Lösung: Wettlauffreie Zustandskodierung
  - Bei jedem Übergang darf sich maximal eine Zustandsvariable ändern
  - Eventuell mehr Zustandsspeicher notwendig als mit minimaler Zustandskodierung

Probleme asynchroner Schaltwerke

- Problematik: Wettläufe
  - Wenn sich bei einem Übergang mehrere Zustandsvariablen ändern, geschieht diese Änderung der Zustandsvariablen nicht exakt gleichzeitig
  - Schaltwerk nimmt implizite Zwischenzustände an, in denen eine Zustandsvariable geändert wurde, aber die andere noch nicht (**Wettlauf**)
  - Falls der Zwischenzustand einen anderen Folgezustand liefert oder selbst stabil ist, wird das Ziel des ursprünglichen Übergangs nicht erreicht (**kritischer Wettlauf**)
- Lösung: Wettlauffreie Zustandskodierung
  - Bei jedem Übergang darf sich maximal eine Zustandsvariable ändern
  - Eventuell mehr Zustandsspeicher notwendig als mit minimaler Zustandskodierung

Entwurf von Schaltwerken

1. Spezifikation des Problems
2. Auswahl des Automaten-Typs (Mealy oder Moore)
3. Entwurf des Automaten (Zustände, Ablauftabelle oder Automatentafel, evtl. Automatengraphen)
4. Festlegen einer Zustandskodierung
5. Entwurf der Schaltnetze der Ansteuerungsfunktionen für die Zustandsspeicher und der Ausgabefunktionen
6. Aufbauen des Schaltwerks aus Zustandsspeichern und Ansteuerungs- und Ausgabefunktionen

Entwurf von Schaltwerken

1. Spezifikation des Problems
2. Auswahl des Automaten-Typs (Mealy oder Moore)
3. Entwurf des Automaten (Zustände, Ablauftabelle oder Automatentafel, evtl. Automatengraphen)
4. Festlegen einer Zustandskodierung
5. Entwurf der Schaltnetze der Ansteuerungsfunktionen für die Zustandsspeicher und der Ausgabefunktionen
6. Aufbauen des Schaltwerks aus Zustandsspeichern und Ansteuerungs- und Ausgabefunktionen

Entwurf von Schaltwerken

1. Spezifikation des Problems
2. Auswahl des Automaten-Typs (Mealy oder Moore)
3. Entwurf des Automaten (Zustände, Ablauftabelle oder Automatentafel, evtl. Automatengraphen)
4. Festlegen einer Zustandskodierung
5. Entwurf der Schaltnetze der Ansteuerungsfunktionen für die Zustandsspeicher und der Ausgabefunktionen
6. Aufbauen des Schaltwerks aus Zustandsspeichern und Ansteuerungs- und Ausgabefunktionen

Entwurf von Schaltwerken

1. Spezifikation des Problems
2. Auswahl des Automaten-Typs (Mealy oder Moore)
3. Entwurf des Automaten (Zustände, Ablauftabelle oder Automatentafel, evtl. Automatengraphen)
4. Festlegen einer Zustandskodierung
5. Entwurf der Schaltnetze der Ansteuerungsfunktionen für die Zustandsspeicher und der Ausgabefunktionen
6. Aufbauen des Schaltwerks aus Zustandsspeichern und Ansteuerungs- und Ausgabefunktionen

Entwurf von Schaltwerken

1. Spezifikation des Problems
2. Auswahl des Automaten-Typs (Mealy oder Moore)
3. Entwurf des Automaten (Zustände, Ablauftabelle oder Automatentafel, evtl. Automatengraphen)
4. Festlegen einer Zustandskodierung
5. Entwurf der Schaltnetze der Ansteuerungsfunktionen für die Zustandsspeicher und der Ausgabefunktionen
6. Aufbauen des Schaltwerks aus Zustandsspeichern und Ansteuerungs- und Ausgabefunktionen

Entwurf von Schaltwerken

1. Spezifikation des Problems
2. Auswahl des Automaten-Typs (Mealy oder Moore)
3. Entwurf des Automaten (Zustände, Ablauftabelle oder Automatentafel, evtl. Automatengraphen)
4. Festlegen einer Zustandskodierung
5. Entwurf der Schaltnetze der Ansteuerungsfunktionen für die Zustandsspeicher und der Ausgabefunktionen
6. Aufbauen des Schaltwerks aus Zustandsspeichern und Ansteuerungs- und Ausgabefunktionen

Übungsaufgabe 1

Gegeben sei die Ablauftabelle eines endlichen Automaten mit den symbolischen Zuständen a, b und c.

Z^t	e^t	Z^{t+1}	y^t_{Mealy}	y^t_{Moore}
a	0	a		
a	1	b		
b	0	a		
b	1	c		
c	0	a		
c	1	b		

1. Füllen Sie die mit y^t_{Mealy} bezeichnete Spalte mit einer *nicht* konstanten Ausgabefunktion Ihrer Wahl so aus, dass die Ablauftabelle einem Mealy-Automaten entspricht. Begründen Sie Ihre Wahl.

Übungsaufgabe 1

Gegeben sei die Ablauftabelle eines endlichen Automaten mit den symbolischen Zuständen a, b und c.

Z^t	e^t	Z^{t+1}	y^t_{Mealy}	y^t_{Moore}
a	0	a		
a	1	b		
b	0	a		
b	1	c		
c	0	a		
c	1	b		

2. Füllen Sie die mit y^t_{Moore} bezeichnete Spalte mit einer *nicht* konstanten Ausgabefunktion Ihrer Wahl so aus, dass die Ablauftabelle einem Moore-Automaten entspricht. Begründen Sie Ihre Wahl.

Übungsaufgabe 1

Gegeben sei die Ablauftabelle eines endlichen Automaten mit den symbolischen Zuständen a, b und c.

Z^t	e^t	Z^{t+1}	y^t_{Mealy}	y^t_{Moore}
a	0	a		
a	1	b		
b	0	a		
b	1	c		
c	0	a		
c	1	b		

3. Geben Sie den Automatengraphen für Ihren Mealy-Automaten aus Aufgabenteil 1 an. Vergessen Sie nicht, die Kanten zu beschriften.

Übungsaufgabe 2

Gegeben sei die nachstehende Ablauftabelle eines Automaten mit den symbolischen Zuständen A, B, C und D.

Zustand	Eingabe		Folgezustand	Ausgabe	
Z^t	e_1^t	e_2^t	Z^{t+1}	y_1	y_0
A	1	-	D	0	0
A	0	0	B	0	1
A	0	1	A	1	1
B	-	0	C	1	1
B	-	1	A	0	1
C	-	0	A	1	0
C	-	1	C	0	1
D	-	-	A	0	0

1. Um welchen Automatentyp handelt es sich bei dem angebenen Automaten?

Übungsaufgabe 2

Gegeben sei die nachstehende Ablauftabelle eines Automaten mit den symbolischen Zuständen A, B, C und D.

Zustand Z^t	Eingabe e_1^t	Eingabe e_2^t	Folgezustand Z^{t+1}	Ausgabe y_1	Ausgabe y_0
A	1	-	D	0	0
A	0	0	B	0	1
A	0	1	A	1	1
B	-	0	C	1	1
B	-	1	A	0	1
C	-	0	A	1	0
C	-	1	C	0	1
D	-	-	A	0	0

2. Wie viele Flipflops würden Sie für die Realisierung des Automaten als synchrones Schaltwerk mindestens benötigen?

Übungsaufgabe 2

Gegeben sei die nachstehende Ablauftabelle eines Automaten mit den symbolischen Zuständen A, B, C und D.

Zustand	Eingabe		Folgezustand	Ausgabe	
Z^t	e_1^t	e_2^t	Z^{t+1}	y_1	y_0
A	1	-	D	0	0
A	0	0	B	0	1
A	0	1	A	1	1
B	-	0	C	1	1
B	-	1	A	0	1
C	-	0	A	1	0
C	-	1	C	0	1
D	-	-	A	0	0

3. Geben Sie den Automatengraphen des Automaten an.

Probeklausur

- Termin: **01.02.2011** (Mi), 11.30 Uhr, im Audimax
- Probeklausur:
  - Nur DuE (TI 1), kein RO (TI 2)
  - In der Regel fünf Aufgaben (= TI 1-Anteil an einer TI-Klausur)
  - Vorbereitung: Alte Klausuren rechnen!
- Was bringt die Probeklausur?

Note	Bonuspunkte
1	2
2	1,5
3	1
4	0,5
5	0

- Bonuspunkte werden nur auf eine bestandene Klausur angerechnet (siehe Merkblatt auf Vorlesungshomepage)

Probeklausur

- Termin: **01.02.2011** (Mi), 11.30 Uhr, im Audimax
- Probeklausur:
  - Nur DuE (TI 1), kein RO (TI 2)
  - In der Regel fünf Aufgaben (= TI 1-Anteil an einer TI-Klausur)
  - Vorbereitung: Alte Klausuren rechnen!
- Was bringt die Probeklausur?

Note	Bonuspunkte
1	2
2	1,5
3	1
4	0,5
5	0

- Bonuspunkte werden nur auf eine bestandene Klausur angerechnet (siehe Merkblatt auf Vorlesungshomepage)

Probeklausur

- Termin: **01.02.2011** (Mi), 11.30 Uhr, im Audimax
- Probeklausur:
  - Nur DuE (TI 1), kein RO (TI 2)
  - In der Regel fünf Aufgaben (= TI 1-Anteil an einer TI-Klausur)
  - Vorbereitung: Alte Klausuren rechnen!
- Was bringt die Probeklausur?

Note	Bonuspunkte
1	2
2	1,5
3	1
4	0,5
5	0

- Bonuspunkte werden nur auf eine bestandene Klausur angerechnet (siehe Merkblatt auf Vorlesungshomepage)

Probeklausur

- Termin: **01.02.2011** (Mi), 11.30 Uhr, im Audimax
- Probeklausur:
  - Nur DuE (TI 1), kein RO (TI 2)
  - In der Regel fünf Aufgaben (= TI 1-Anteil an einer TI-Klausur)
  - Vorbereitung: Alte Klausuren rechnen!
- Was bringt die Probeklausur?

Note	Bonuspunkte
1	2
2	1,5
3	1
4	0,5
5	0

- Bonuspunkte werden nur auf eine bestandene Klausur angerechnet (siehe Merkblatt auf Vorlesungshomepage)

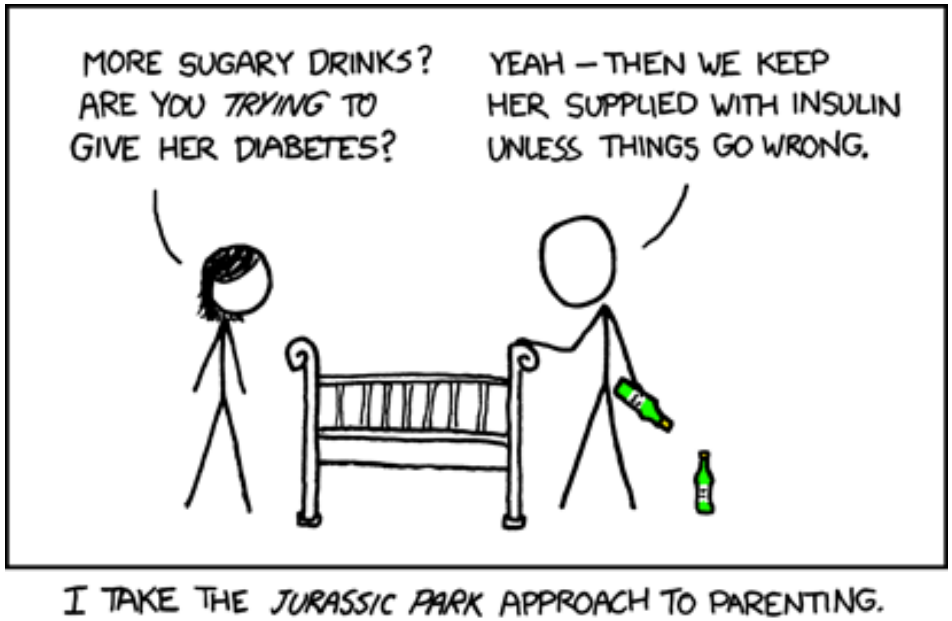
Probeklausur

- Rückgabe der Probeklausur im Tutorium
- Wer nicht anwesend ist: Ergebnis per E-Mail
- Anmeldung zur Probeklausur **jetzt** im Tutorium
  $\rightarrow$ Erforderlich, damit ausreichend Klausuren gedruckt werden können

Probeklausur

- Rückgabe der Probeklausur im Tutorium
- Wer nicht anwesend ist: Ergebnis per E-Mail
- Anmeldung zur Probeklausur **jetzt** im Tutorium
  $\rightarrow$ Erforderlich, damit ausreichend Klausuren gedruckt werden können

Fertig!



Quelle: http://xkcd.com/531/