Field Programmable Gate Array

Geschichte/Allgemeines, Funktionsweise, Anwendung

Alexander Neubacher
Günther Pritz
Benjamin Traubenek

Paris Lodron Universität Salzburg

WAP WS23/24
12.01.2024

Überblick

Einführung

Was sind FPGAs?

- FPGA steht für Field Programmable Gate Array und gehört zur Familie der Programmable Logic Devices (PLD).

Was sind FPGAs?

- FPGA steht für Field Programmable Gate Array und gehört zur Familie der Programmable Logic Devices (PLD).
- Ist ein Halbleiterchip.

Was sind FPGAs?

- FPGA steht für Field Programmable Gate Array und gehört zur Familie der Programmable Logic Devices (PLD).
- Ist ein Halbleiterchip.
- Ging aus Fortschritten von Programmable Array Logic (PAL) und General Array Logic (GAL) hervor.

Was sind FPGAs?

- FPGA steht für Field Programmable Gate Array und gehört zur Familie der Programmable Logic Devices (PLD).
- Ist ein Halbleiterchip.
- Ging aus Fortschritten von Programmable Array Logic (PAL) und General Array Logic (GAL) hervor.
- Die Hauptmerkmale von FPGAs sind
  - Gate Array,
  - Programmierbarkeit und
  - Vielseitigkeit.

Unterschiede zu anderen digitalen Chips

- Allgemeine Einteilung von digitalen Chips:
  - Application Specific Standard Products (ASSP).
  - Programmable Logic Devices (PLD).

Unterschiede zu anderen digitalen Chips

- Allgemeine Einteilung von digitalen Chips:
  - Application Specific Standard Products (ASSP).
  - Programmable Logic Devices (PLD).

- FPGAs bieten eine größere Designfreiheit im Vergleich zu anderen PLDs.

Unterschiede zu anderen digitalen Chips

- Allgemeine Einteilung von digitalen Chips:
  - Application Specific Standard Products (ASSP).
  - Programmable Logic Devices (PLD).
- FPGAs bieten eine größere Designfreiheit im Vergleich zu anderen PLDs.
- Überwinden die Einschränkungen die mit benutzerdefinierten Schaltungen und der endlichen Anzahl von Gattern in ursprünglich programmierbaren Geräten verbunden sind.

Entwicklung

- FPGAs stellen seit den 1980er Jahren eine bedeutende Entwicklung in der Community der Application-Specific Integrated Circuits (ASIC) dar.

Entwicklung

- FPGAs stellen seit den 1980er Jahren eine bedeutende Entwicklung in der Community der Application-Specific Integrated Circuits (ASIC) dar.
- Xilinx und Altera als Schlüsselspieler bei der Verbreitung der FPGA-Technologie.

Entwicklung

- FPGAs stellen seit den 1980er Jahren eine bedeutende Entwicklung in der Community der Application-Specific Integrated Circuits (ASIC) dar.
- Xilinx und Altera als Schlüsselspieler bei der Verbreitung der FPGA-Technologie.
- Umprogrammierbare Alternative zu ASICs die eine schnellere Entwicklung und Tests ermöglichen.

Entwicklung

- FPGAs stellen seit den 1980er Jahren eine bedeutende Entwicklung in der Community der Application-Specific Integrated Circuits (ASIC) dar.
- Xilinx und Altera als Schlüsselspieler bei der Verbreitung der FPGA-Technologie.
- Umprogrammierbare Alternative zu ASICs die eine schnellere Entwicklung und Tests ermöglichen.
- FPGAs gewinnen in der Telekommunikationsbranche aufgrund der sich schnell ändernden Standards und des Bedarfs an agilen Lösungen an Bedeutung.

Technische Entwicklung

- FPGAs folgen dem Mooreschen Gesetz.

Technische Entwicklung

- FPGAs folgen dem Mooreschen Gesetz.
- Pionierarbeit von Xilinx im Jahr 1984 durch die Einführung des XC2064 mit 64 programmierbaren Logikzellen.

Technische Entwicklung

- FPGAs folgen dem Mooreschen Gesetz.
- Pionierarbeit von Xilinx im Jahr 1984 durch die Einführung des XC2064 mit 64 programmierbaren Logikzellen.
- Durch den Fortschritt der Halbleiterindustrie stieg die Kapazität von FPGAs.

Technische Entwicklung

- FPGAs folgen dem Mooreschen Gesetz.
- Pionierarbeit von Xilinx im Jahr 1984 durch die Einführung des XC2064 mit 64 programmierbaren Logikzellen.
- Durch den Fortschritt der Halbleiterindustrie stieg die Kapazität von FPGAs.
- Dies führte zur Entwicklung spezialisierter Design-Tools:
  - ISE- und Vivado-Serie von Xilinx.
  - Qartus-Serie von Intel.

Moderne Bedeutung von FPGAs

- Spielen heute als programmierbare System-on Chips mit komplexen Funktionen eine entscheidende Rolle.

Moderne Bedeutung von FPGAs

- Spielen heute als programmierbare System-on Chips mit komplexen Funktionen eine entscheidende Rolle.
- FPGAs werden leistungsstark beschleunigt, indem sie wiederverwendbare IP-Cores und dedizierte Logikeinheiten integrieren. Das ermöglicht ihnen, spezifische Aufgaben effizienter auszuführen als herkömmliche Prozessoren.

Moderne Bedeutung von FPGAs

- Spielen heute als programmierbare System-on Chips mit komplexen Funktionen eine entscheidende Rolle.
- FPGAs werden leistungsstark beschleunigt, indem sie wiederverwendbare IP-Cores und dedizierte Logikeinheiten integrieren. Das ermöglicht ihnen, spezifische Aufgaben effizienter auszuführen als herkömmliche Prozessoren.
- In der KI-Ära werden FPGAs mit spezialisierten IP-Cores und Logikeinheiten als Hochleistungsbeschleuniger weiterentwickelt, was ihre Marktposition stärkt.

Marktsituation

- FPGAs wurden ursprünglich im Software Defined Radio (SDR) für vielseitige Kommunikation verwendet.

Marktsituation

- FPGAs wurden ursprünglich im Software Defined Radio (SDR) für vielseitige Kommunikation verwendet.
- Aufgrund der hohen Kosten, die bei einer Umstellung der Produktionslinien entstehen würden, zögerte die Verteidigungsindustrie, FPGA-basierte SDRs einzuführen.

Marktsituation

- FPGAs wurden ursprünglich im Software Defined Radio (SDR) für vielseitige Kommunikation verwendet.
- Aufgrund der hohen Kosten, die bei einer Umstellung der Produktionslinien entstehen würden, zögerte die Verteidigungsindustrie, FPGA-basierte SDRs einzuführen.
- Mittlerweile werden diese jedoch auch hier eingesetzt.

Vorteile am Markt

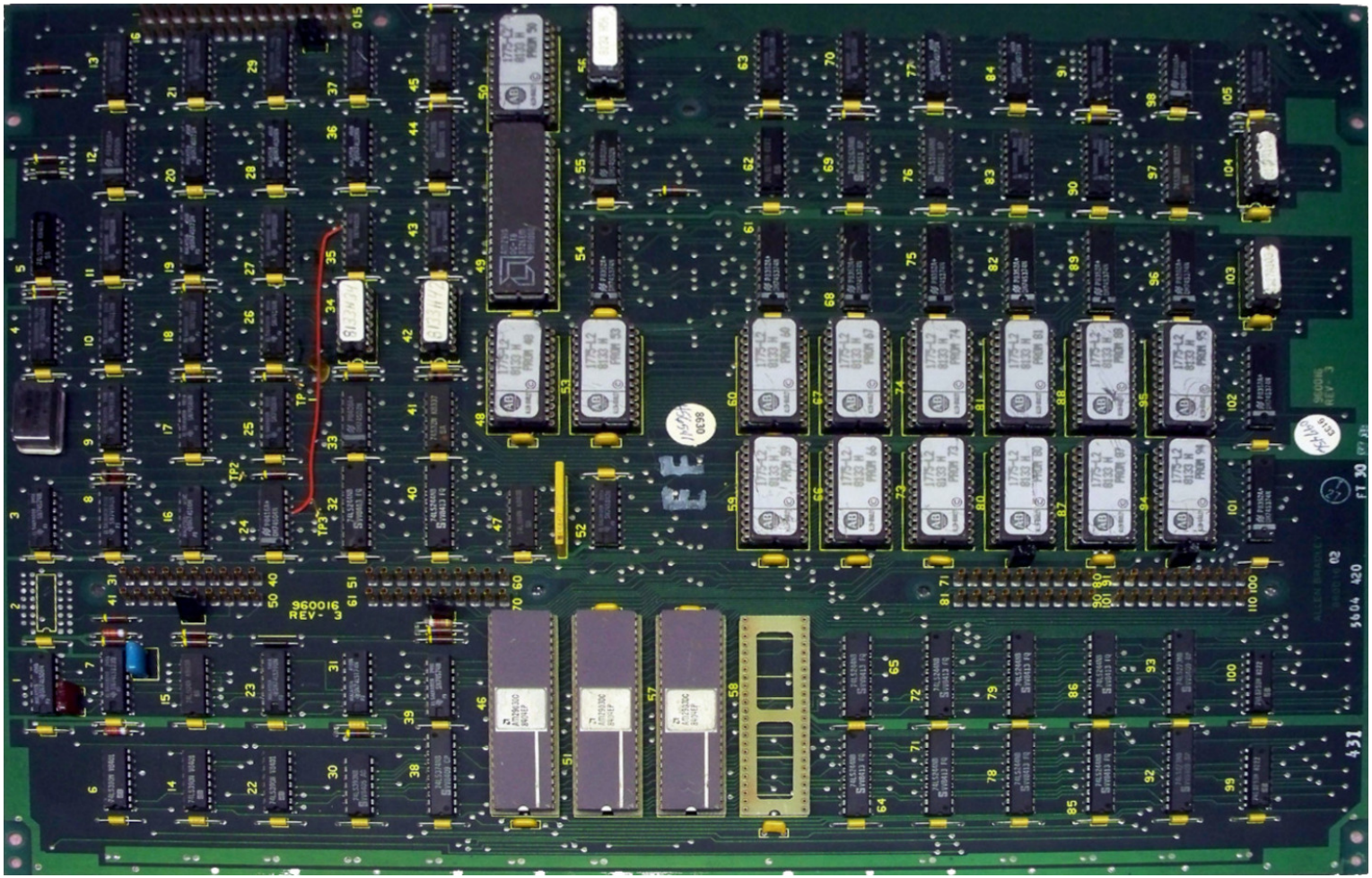
- Trotz der Herausforderungen bleiben FPGAs wertvoll für einen schnellen Markteintritt, Wettbewerbsvorteile und kritische Anwendungen, insbesondere bei High Performance Computing (HPC) und in Rechenzentren.

Vorteile am Markt

- Trotz der Herausforderungen bleiben FPGAs wertvoll für einen schnellen Markteintritt, Wettbewerbsvorteile und kritische Anwendungen, insbesondere bei High Performance Computing (HPC) und in Rechenzentren.
- FPGAs bieten Betriebskostenvorteile gegenüber CPUs und GPUs und tragen zu kleineren und effizienteren Rechenzentren bei.

Funktion

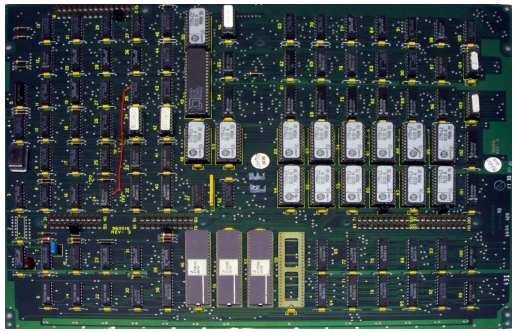
Ausgangsbasis - Motivation

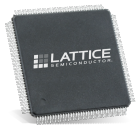


Quelle: Pixabay

Ziel

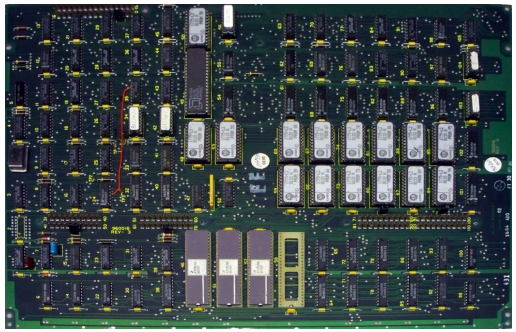
- Umsetzung der Funktion einer digitalen Baugruppe als integrierter Schaltkreis (IC).

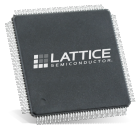




Ziel

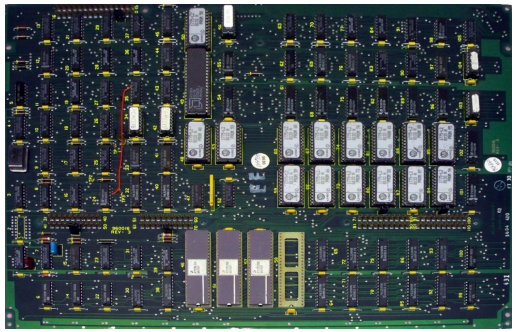
- Umsetzung der Funktion einer digitalen Baugruppe als integrierter Schaltkreis (IC).
- **Zusätzlich:** Funktionalität frei programmierbar!

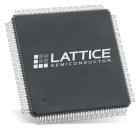




Ziel

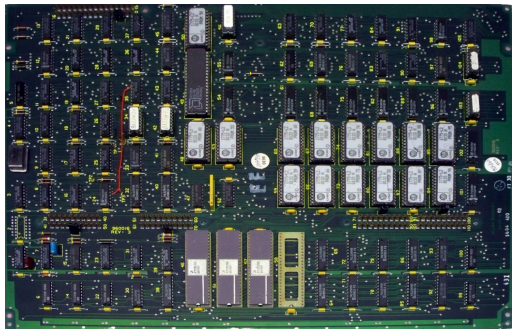
- Umsetzung der Funktion einer digitalen Baugruppe als integrierter Schaltkreis (IC).
- **Zusätzlich:** Funktionalität frei programmierbar!
- Welche Funktionsblöcke finden in digitalen Baugruppen Anwendung?

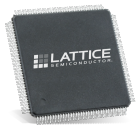




Ziel

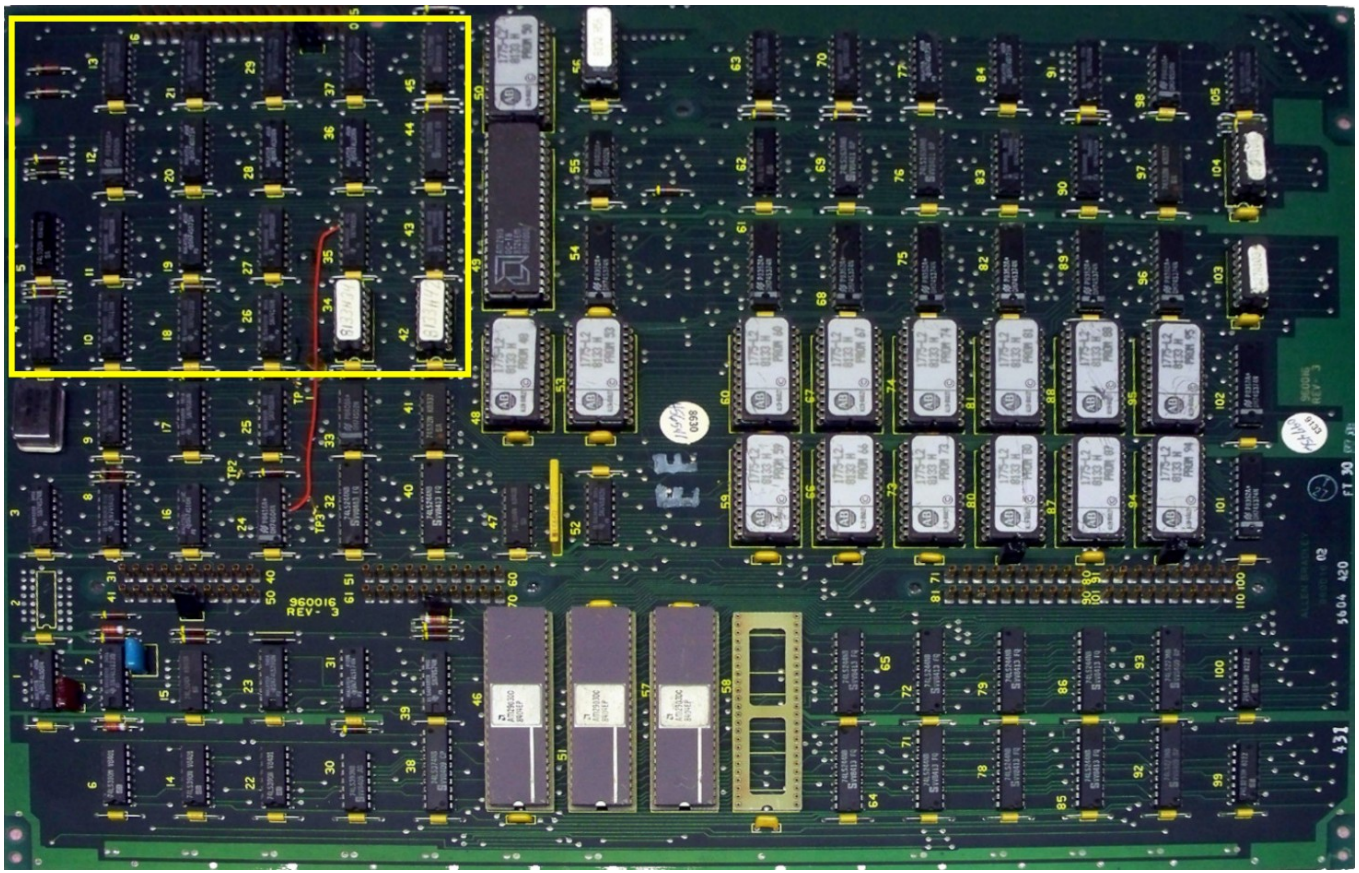
- Umsetzung der Funktion einer digitalen Baugruppe als integrierter Schaltkreis (IC).
- **Zusätzlich:** Funktionalität frei programmierbar!
- Welche Funktionsblöcke finden in digitalen Baugruppen Anwendung?
- Wie können diese frei programmierbar realisiert werden?





Funktionsblöcke digitaler Schaltungen

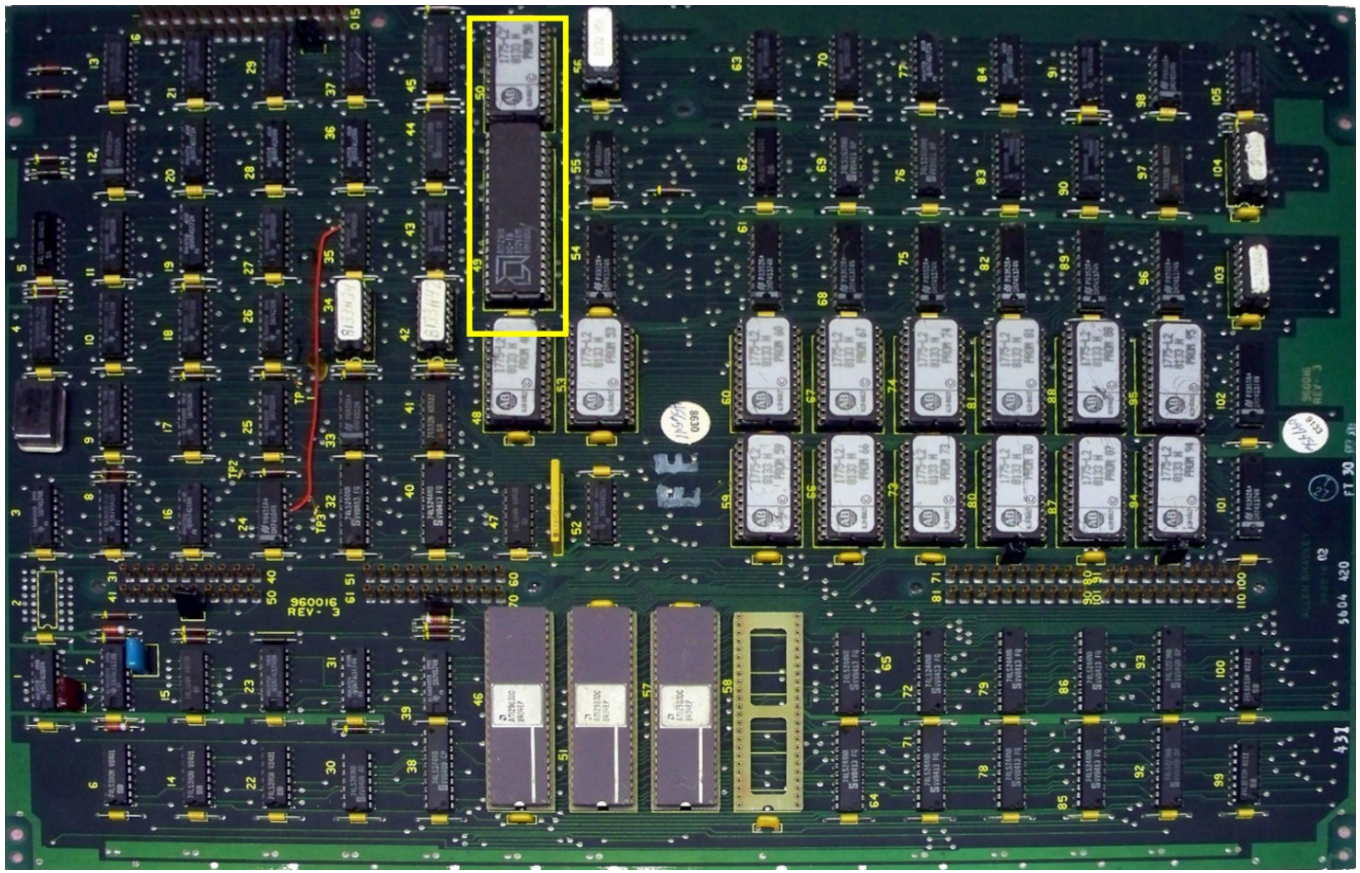
Kombinatorische Logik (Gatter)



Quelle: Pixabay

Funktionsblöcke digitaler Schaltungen

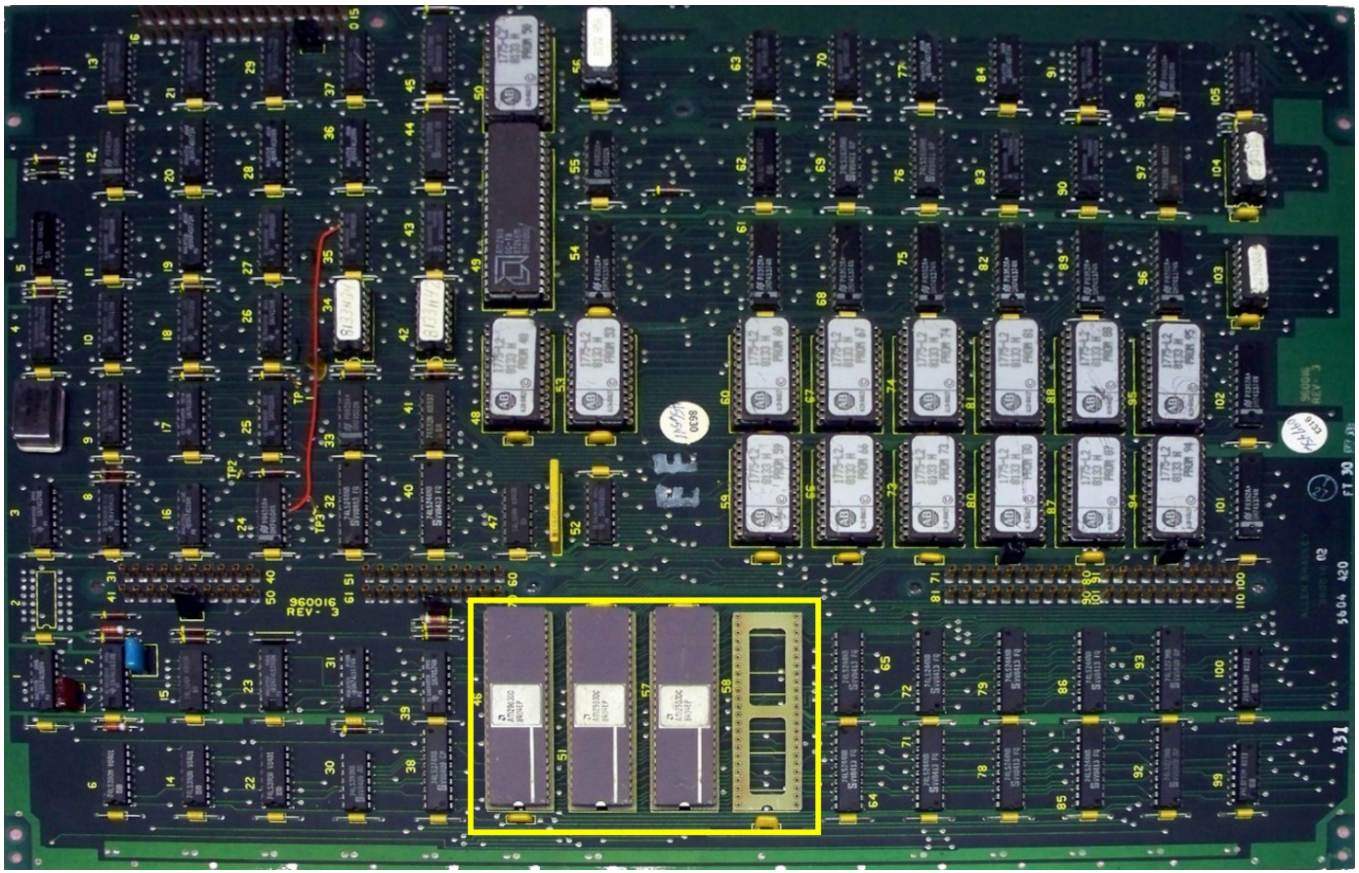
Sequentielle Logik (Flipflop)



Quelle: Pixabay

Funktionsblöcke digitaler Schaltungen

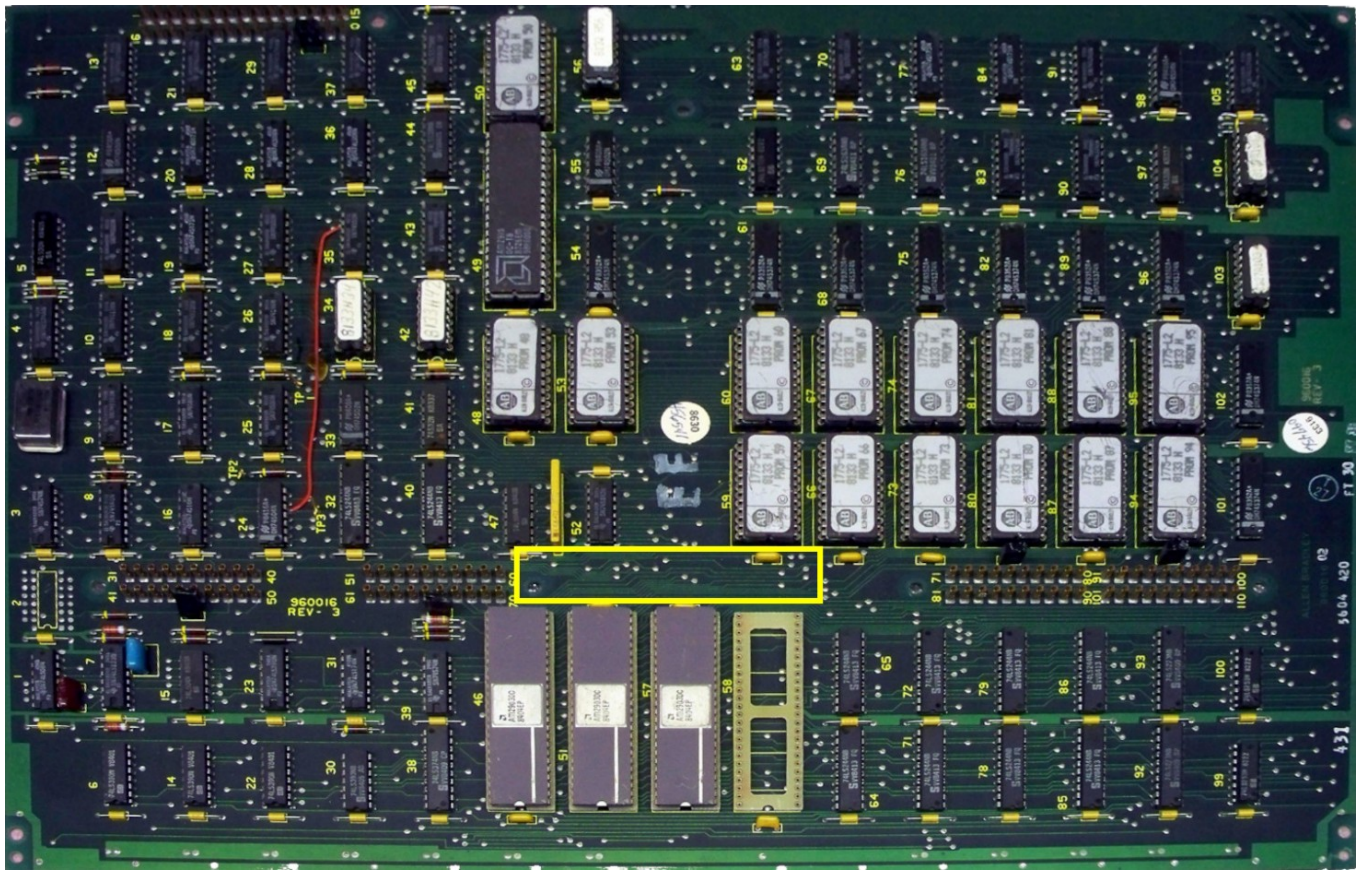
Speicher



Quelle: Pixabay

Funktionsblöcke digitaler Schaltungen

Verbindungen



Quelle: Pixabay

Umsetzung: Kombinatorische Logik

- Logikgatter repräsentieren Operatoren einer logischen Funktion.

Umsetzung: Kombinatorische Logik

- Logikgatter repräsentieren Operatoren einer logischen Funktion.
- Ungeeignet für freie Programmierung, weil nicht mehr veränderbar.

Umsetzung: Kombinatorische Logik

- Logikgatter repräsentieren Operatoren einer logischen Funktion.
- Ungeeignet für freie Programmierung, weil nicht mehr veränderbar.
- Alternativ kann eine logische Funktion auch über ihre Funktionswerte beschrieben werden.

Umsetzung: Kombinatorische Logik

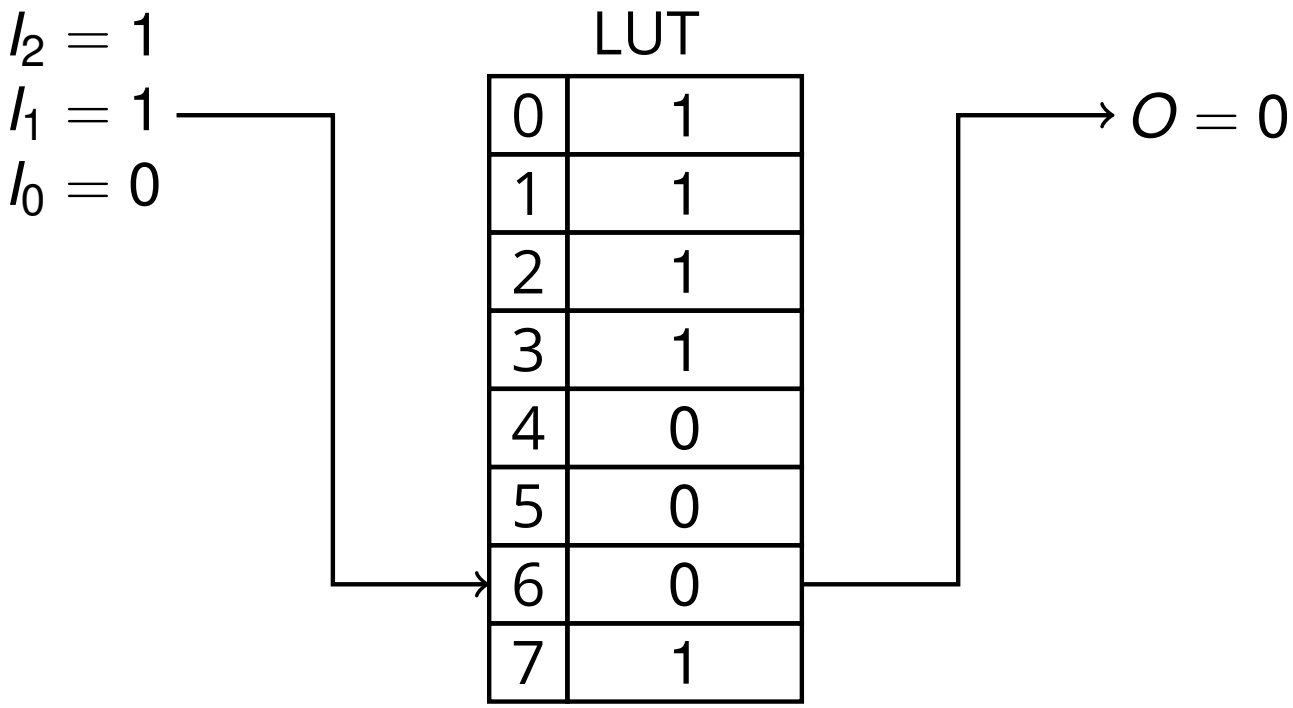
- Logikgatter repräsentieren Operatoren einer logischen Funktion.
- Ungeeignet für freie Programmierung, weil nicht mehr veränderbar.
- Alternativ kann eine logische Funktion auch über ihre Funktionswerte beschrieben werden.
- Implementierung erfolgt als Lookup-Tabelle (LUT).

Umsetzung: Kombinatorische Logik

- Logikgatter repräsentieren Operatoren einer logischen Funktion.
- Ungeeignet für freie Programmierung, weil nicht mehr veränderbar.
- Alternativ kann eine logische Funktion auch über ihre Funktionswerte beschrieben werden.
- Implementierung erfolgt als Lookup-Tabelle (LUT).
- Variablen (Eingänge) werden als Adresse aufgefasst.

Umsetzung: Kombinatorische Logik

Beispiel LUT: $O = (I_0 \wedge I_1) \vee \neg I_2$



Umsetzung: Sequentielle Logik

- Um möglichst universell zu sein werden folgende Eigenschaften gefordert:

Umsetzung: Sequentielle Logik

- Um möglichst universell zu sein werden folgende Eigenschaften gefordert:
  - Operation muss synchron zu einem Eingangstakt sein,

Umsetzung: Sequentielle Logik

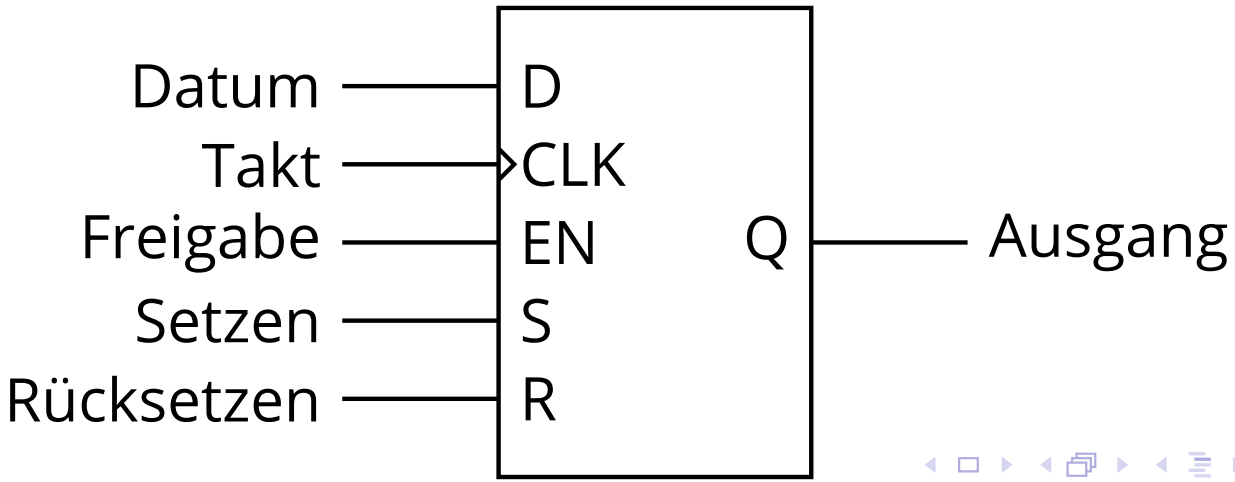
- Um möglichst universell zu sein werden folgende Eigenschaften gefordert:
  - Operation muss synchron zu einem Eingangstakt sein,
  - über einen Freigabeeingang für den Eingangstakt und

Umsetzung: Sequentielle Logik

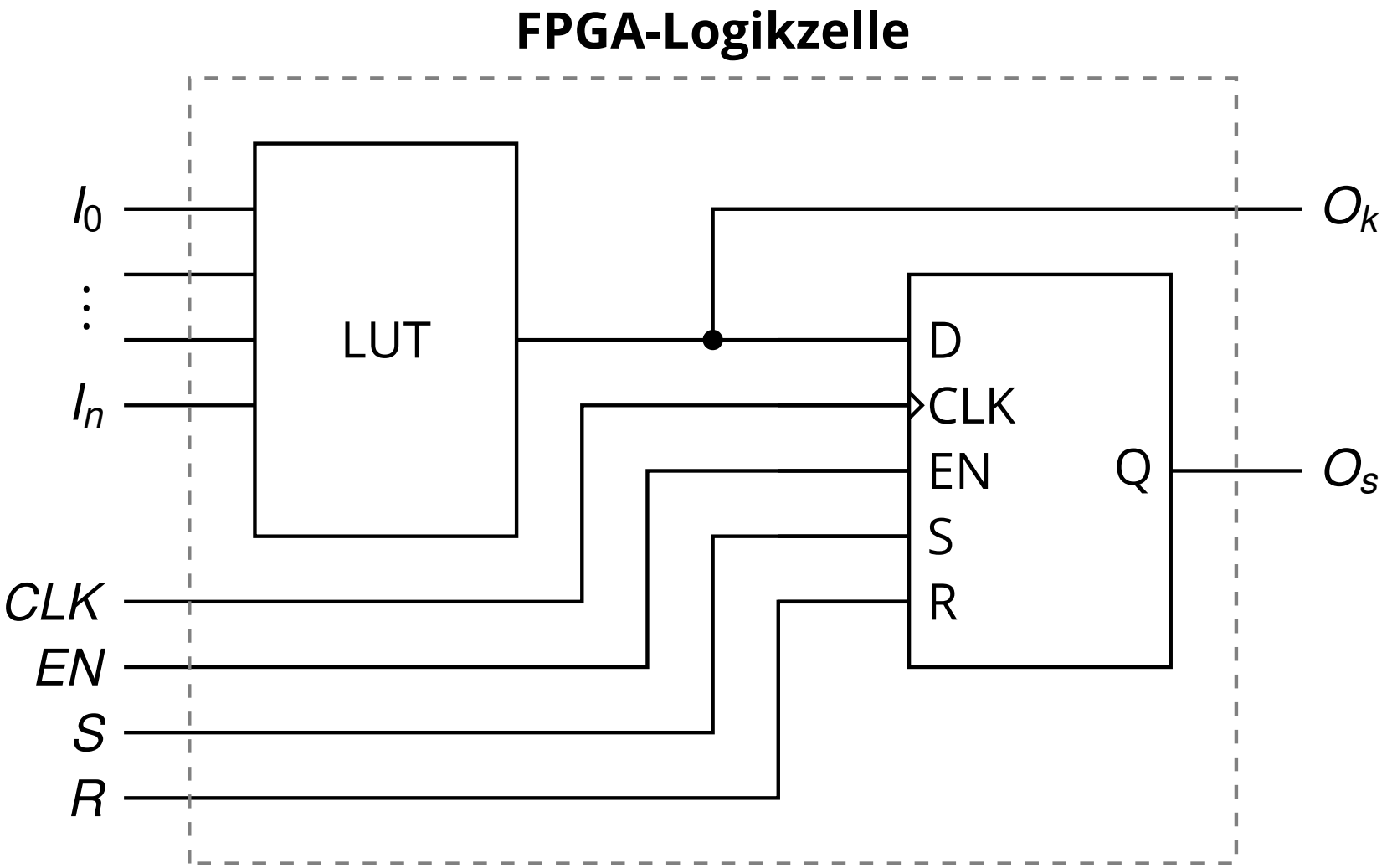
- Um möglichst universell zu sein werden folgende Eigenschaften gefordert:
  - Operation muss synchron zu einem Eingangstakt sein,
  - über einen Freigabeeingang für den Eingangstakt und
  - über einen asynchronen Set- und Reseteingang verfügen.

Umsetzung: Sequentielle Logik

- Um möglichst universell zu sein werden folgende Eigenschaften gefordert:
  - Operation muss synchron zu einem Eingangstakt sein,
  - über einen Freigabeeingang für den Eingangstakt und
  - über einen asynchronen Set- und Reseteingang verfügen.
- Diese geforderten Charakteristika lassen sich mittels eines erweiterten D-Flipflops realisieren.



Umsetzung: Logikzelle



Umsetzung: Speicher

- Aufgrund der Eigenschaften des D-Flipflops können Logikzellen auch als 1-Bit-Speicher verwendet werde.

Umsetzung: Speicher

- Aufgrund der Eigenschaften des D-Flipflops können Logikzellen auch als 1-Bit-Speicher verwendet werde.
- Nachteil: LUT ist Overhead. Deshalb ist dies ein sehr teurer Speicher.

Umsetzung: Speicher

- Aufgrund der Eigenschaften des D-Flipflops können Logikzellen auch als 1-Bit-Speicher verwendet werde.
- Nachteil: LUT ist Overhead. Deshalb ist dies ein sehr teurer Speicher.
- Aus diesem Grund werden in modernen FPGAs deshalb dedizierte RAM-Speicherblöcke verbaut.

Umsetzung: Speicher

- Aufgrund der Eigenschaften des D-Flipflops können Logikzellen auch als 1-Bit-Speicher verwendet werde.
- Nachteil: LUT ist Overhead. Deshalb ist dies ein sehr teurer Speicher.
- Aus diesem Grund werden in modernen FPGAs deshalb dedizierte RAM-Speicherblöcke verbaut.
- Zusätzlich werden heutige FPGAs mit nichtflüchtigem Speicher ausgestattet, der die Konfiguration enthält.

Umsetzung: Verbindungen

- Logikblöcke werden in einer Matrix angeordnet.

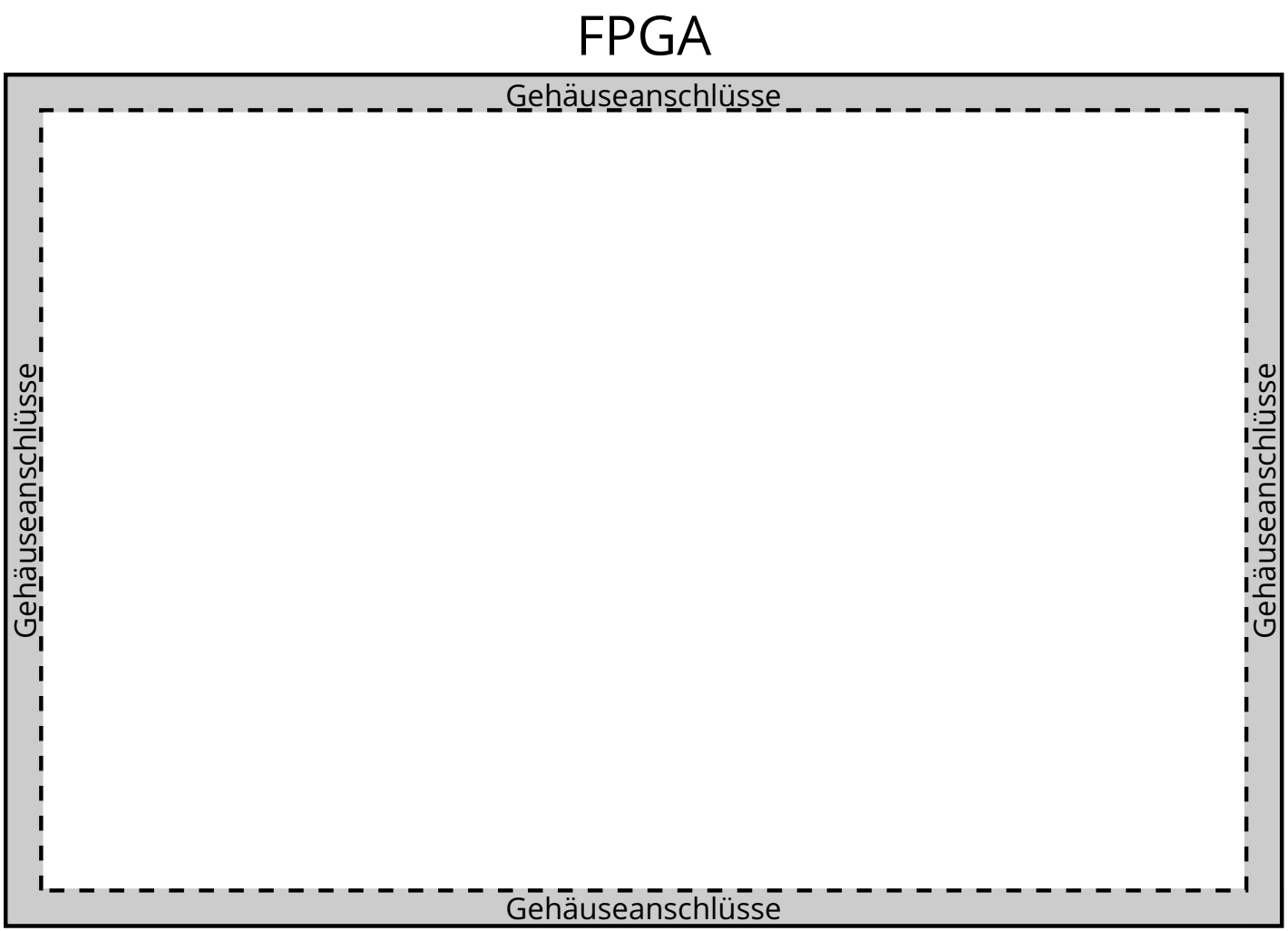
Umsetzung: Verbindungen

- Logikblöcke werden in einer Matrix angeordnet.
- Zwischen den Blöcken werden Verbindungsleitungen gitterförmig verlegt.

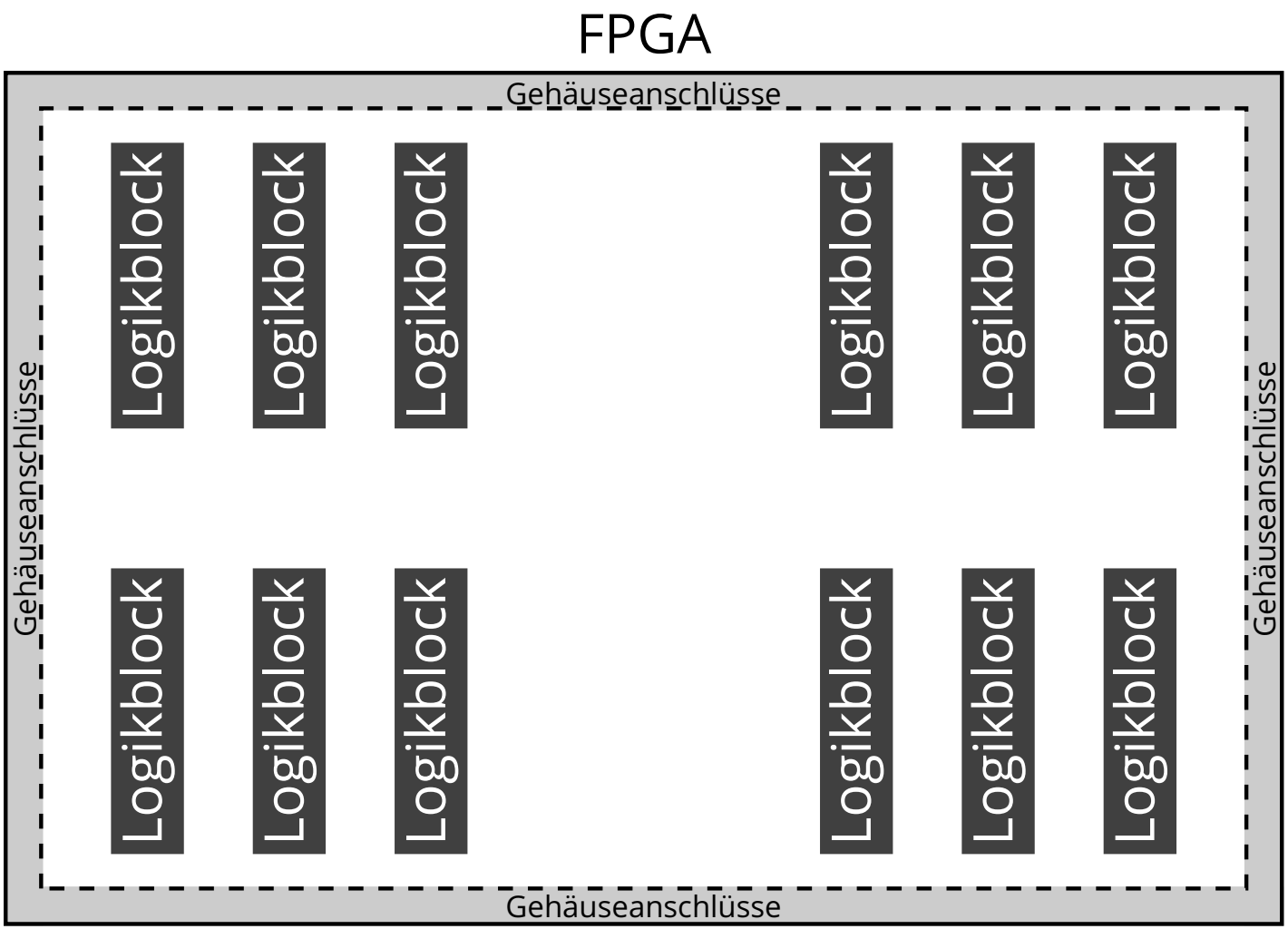
Umsetzung: Verbindungen

- Logikblöcke werden in einer Matrix angeordnet.
- Zwischen den Blöcken werden Verbindungsleitungen gitterförmig verlegt.
- An den Kreuzungspunkten verbaut man steuerbare Schalter mit denen Ein- und Ausgänge zusammengeschaltet werden können.

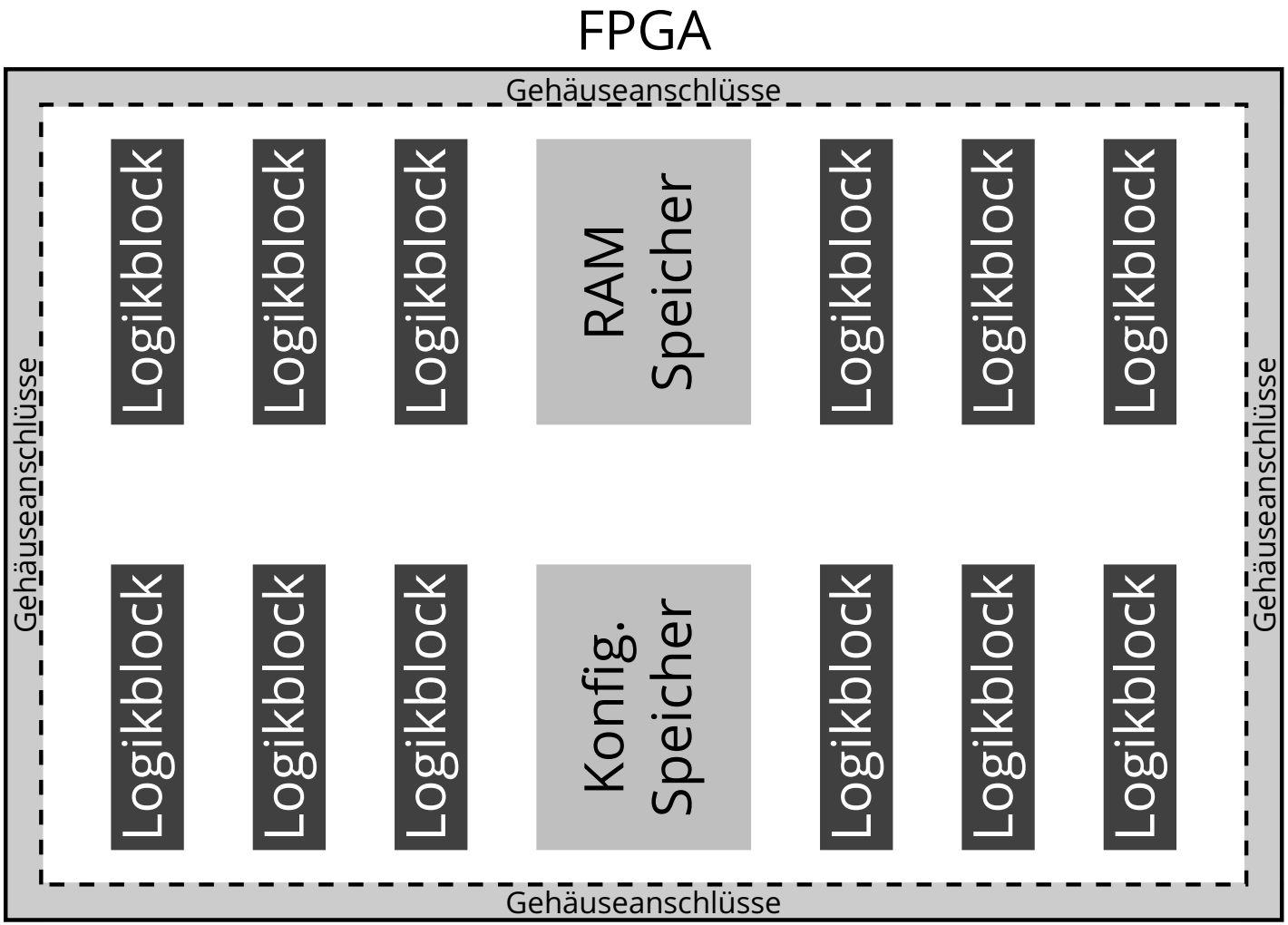
FPGA Gesamtbild



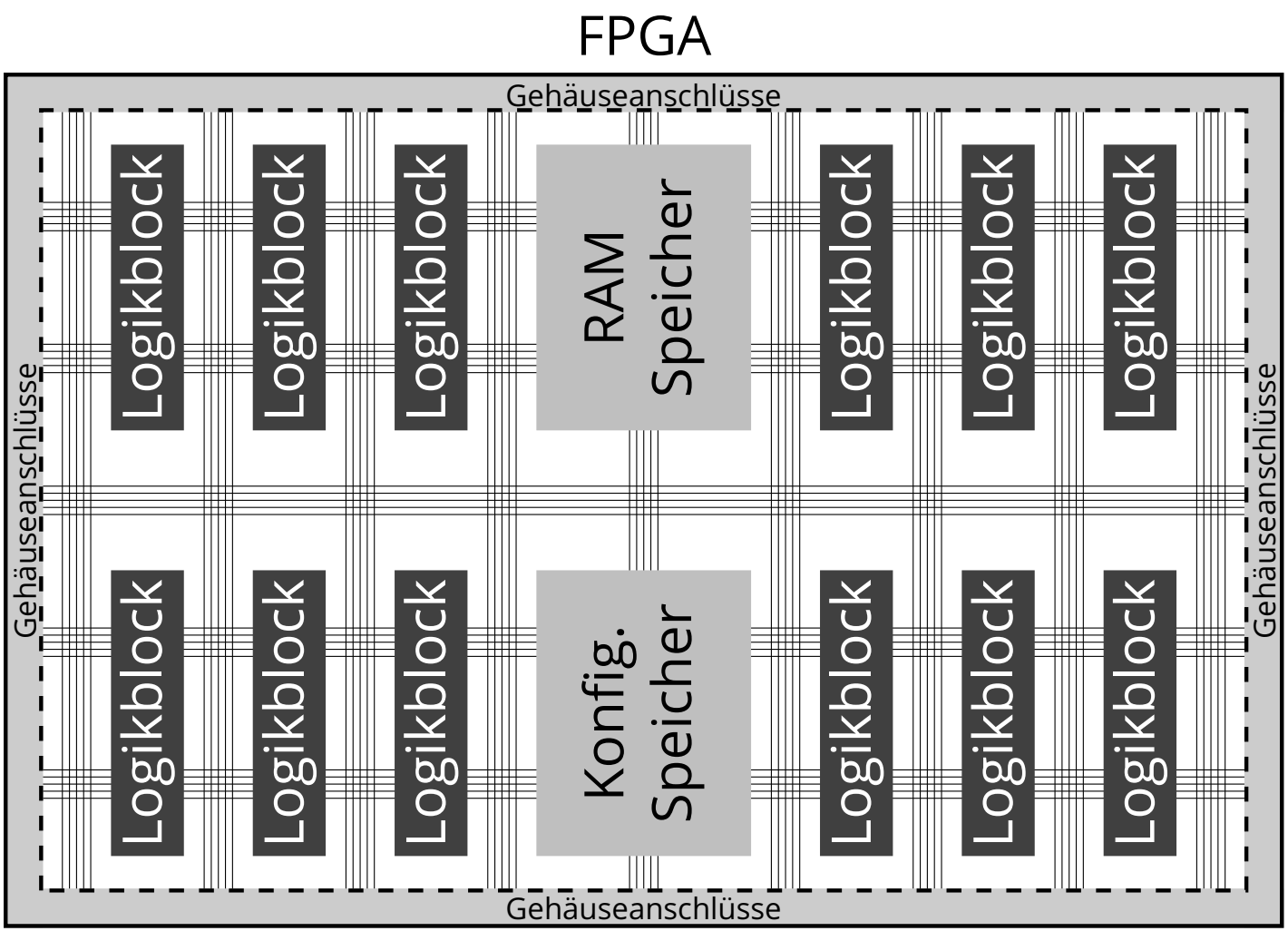
FPGA Gesamtbild



FPGA Gesamtbild



FPGA Gesamtbild



Vorteile und Anwendung

Vorteile

Vorteile im Wesentlichen:

1. Flexibilität

Vorteile

Vorteile im Wesentlichen:

1. Flexibilität
2. Geschwindigkeit/Parallelisierung

Flexibilität

- Field Programmable, d.h. FPGA wird im „Feld“ von Kunden konfiguriert.

Flexibilität

- Field Programmable, d.h. FPGA wird im „Feld“ von Kunden konfiguriert.
- FPGA kann nach spezifischen Anforderungen konfiguriert werden.

Flexibilität

- Field Programmable, d.h. FPGA wird im „Feld“ von Kunden konfiguriert.
- FPGA kann nach spezifischen Anforderungen konfiguriert werden.
- Es bestehen Möglichkeiten beliebige Projekte mit beliebiger Komplexität zu realisieren.

Flexibilität

- Field Programmable, d.h. FPGA wird im „Feld“ von Kunden konfiguriert.
- FPGA kann nach spezifischen Anforderungen konfiguriert werden.
- Es bestehen Möglichkeiten beliebige Projekte mit beliebiger Komplexität zu realisieren.
- Bezüglich der Implementierungsmöglichkeiten bestehen kaum Beschränkungen.

Flexibilität

- Bei FPGA besteht die Möglichkeit zur Rekonfiguration.

Flexibilität

- Bei FPGA besteht die Möglichkeit zur Rekonfiguration.
- Dadurch kann ein FPGA dynamisch angepasst werden.

Flexibilität

- Bei FPGA besteht die Möglichkeit zur Rekonfiguration.
- Dadurch kann ein FPGA dynamisch angepasst werden.
- Dies ist vor allem dann praktisch, wenn im Nachhinein Funktionalitäten hinzugefügt/gelöscht werden sollen.

Flexibilität

- Bei FPGA besteht die Möglichkeit zur Rekonfiguration.
- Dadurch kann ein FPGA dynamisch angepasst werden.
- Dies ist vor allem dann praktisch, wenn im Nachhinein Funktionalitäten hinzugefügt/gelöscht werden sollen.
- Daraus ergibt sich die Möglichkeit Fehler einfach und schnell zu beheben.

Flexibilität

- Bei FPGA besteht die Möglichkeit zur Rekonfiguration.
- Dadurch kann ein FPGA dynamisch angepasst werden.
- Dies ist vor allem dann praktisch, wenn im Nachhinein Funktionalitäten hinzugefügt/gelöscht werden sollen.
- Daraus ergibt sich die Möglichkeit Fehler einfach und schnell zu beheben.
- All dies erfordert keine Änderungen an der FPGA-Hardware.

Geschwindigkeit/Parallelisierung

- Mehrere Operationen gleichzeitig ausführen (im Gegensatz zu sequentiell arbeitenden Microcontrollern).

Geschwindigkeit/Parallelisierung

- Mehrere Operationen gleichzeitig ausführen (im Gegensatz zu sequentiell arbeitenden Microcontrollern).
- Die Parallelisierung bringt eine sehr hohe Geschwindigkeit bei der Abarbeitung von Operationen mit sich.

Geschwindigkeit/Parallelisierung

- Mehrere Operationen gleichzeitig ausführen (im Gegensatz zu sequentiell arbeitenden Microcontrollern).
- Die Parallelisierung bringt eine sehr hohe Geschwindigkeit bei der Abarbeitung von Operationen mit sich.
- Deshalb sehr niedrige Latenzzeit.

Geschwindigkeit/Parallelisierung

- Mehrere Operationen gleichzeitig ausführen (im Gegensatz zu sequentiell arbeitenden Microcontrollern).
- Die Parallelisierung bringt eine sehr hohe Geschwindigkeit bei der Abarbeitung von Operationen mit sich.
- Deshalb sehr niedrige Latenzzeit.
- Vor allem gut für komplexe Berechnungen und Verarbeitung großer Datenmengen.

Anwendung

Parallelisierung vs. sequentieller Ablauf

Verilog (FPGA)

```
1) reg a;
2) reg b;
3)
4) a <= 0;
5) b <= 1;
6)
7) a <= b;
8) b <= a;
```

Java

```
1) int a;
2) int b;
3)
4) a = 0;
5) b = 1;
6)
7) a = b;
8) b = a;
```

Anwendung

- Kryptographie

Anwendung

- Kryptographie
- Kommunikationstechnologie

Anwendung

- Kryptographie
- Kommunikationstechnologie
- Luft- und Raumfahrttechnik

Anwendung

- Kryptographie
- Kommunikationstechnologie
- Luft- und Raumfahrttechnik
- Und viele weitere

Anwendung im Bereich der Kryptographie

- Hohe Flexibilität.
  - Algorithmen sind flexibel aktualisierbar,
  - dies bringt die Möglichkeit mit sich, Algorithmen zu ergänzen.

Anwendung im Bereich der Kryptographie

- Hohe Flexibilität.
  - Algorithmen sind flexibel aktualisierbar,
  - dies bringt die Möglichkeit mit sich, Algorithmen zu ergänzen.

- Bessere Performance.
  - FPGA kann genau auf bestimmtes Verschlüsselungsverfahren optimiert werden.
  - FPGA-Bearbeitung ist bei Berechnung von Verschlüsselungsverfahren viel schneller als Software-basierte Implementierung.

Vielen Dank für Ihre Aufmerksamkeit!