**Zu öffnende Dateien:**

* PowerPoint: Logikschaltungen und Entwurf CPU
* Logisim: Schaltungen Präsentation
* PDF: SAP-Architektur

**Weitere Medien:**

* ICs
* Tafel

**Intro:**

* **Seite 1**
* Begrüßung, BLL vorstellen
* Worum es geht, gleich kurze Einführung
* Zunächst: Bei Fragen einfach reinrufen, wobei straffer Zeitplan
* Bei Langeweile nicht zu offensichtlich zeigen
* Grundverständnis, wie digitale Geräte Daten verarbeiten und verrechnen
* Ziel: Funktionsweise CPU („Gehirn“ solcher Geräte)

**Einführung 1: Digitale Signale und Datenverarbeitung**

* **Seite 2**
* **Seite 3**
* Digital bezeichnet eine Art der Datendarstellung und -verarbeitung
* Handy, Computer als Prime Example digitaler Geräte
* Prinzipiell sind heutzutage aber überall digitale elektronische Schaltkreise zum Steuern verschiedenster Geräte verbaut
* Digital Gegenstück zu Analog
* Analogsignal, wie z.B. Tonsignale dargestellt werden: kontinuierlich und durch Spannung
* Übertragung von Musik an Lautsprecher
* Datenverarbeitung hat diverse Nachteile, technische und wirtschaftliche
* Am Ende ist es bei Strom am einfachsten, zwischen „Strom an“ und „Strom aus“ zu unterscheiden, Beispiel: Steckdose fassen, Spannung ist egal ob 200 oder 230
* **Seite 4**
* Bei Digitaltechnik bzw digitalen Datenverarbeitung wird also zwischen 2 Zuständen unterschi
* Strom an = hohes elektrisches Potential = 1 = wahr
* Strom aus = niedriges elektrisches Potential = 0 = falsch
* Das ist die (vereinfachte) Definition von digital: Darstellen aller Informationen durch 0 und 1
* Zum Verarbeiten von digitalen Signalen bzw. Daten werden log. Verknüpfungen verwendet

**Einführung 2: Logische Verknüpfungen:**

* **Seite 5**
* Logische Verknüpfungen verknüpfen mehrere Eingangssignale zu einem Ausgangssignal
* **Seite 6**
* Entsprechend nach den Operatoren NICHT, UND, ODER, XOR
* **Seite 10**
* Die logischen Operatoren sind in elektr. Schaltungen durch sog. Logikgatter implementiert
* Ein Logikgatter ist ein aktives elektr. Bauteil, das eine logische Operation durchführt
* Die Realisation von solchen Logikgattern ist aber Aufgabe der Digitalen Elektronik
* Bleiben auf der Logikebene:
* Vernachlässigung aller elektronischen Aspekte
* Es geht allein um die logische Funktionalität eines Schaltkreises
* Frage nach Exkurs zur Implementierung? Tafel, **Seite 11**

**Einführung 3: Digitale Informationsdarstellung:**

* **Seite 12**
* **Seite 13**
* Daten werden allgemein als Abfolge von Nullen und Einsen dargestellt, parallel oder seriell
* Datenlängen: Bit, Byte, Word, Double Word
* **Seite 14**
* Zahlen werden im Binärsystem dargestellt: Analogie zum Dezimalsystem
* An der Tafel zeigen
* Negative Zahlen im Zweierkomplement (hat sich etabliert)
* Invertierung, +1, signed Bit drangeklatscht (1 = neg,0 = pos), ohne Erklärung warum
* An der Tafel zeigen
* Zeichenkodierungssatz, der Zeichen einen Binärcode zuordnet: ASCII hat 1 Byte
* Mit einem Byte lässt sich alles Mögliche darstellen:

Unsigned int 0-255; Signed int -128-127; Standardschriftzeichen Latein und Steuerzeichen

* Byteinterpretation hängt immer vom Programm ab

**Einführung 4: CPU:**

* **Seite 15**
* Nach Grundlagen CPU, es geht um Realisierung CPU auf Logikebene, d.h. aus Logikgattern
* **Seite 16**
* CPU = Central Processing Unit, d.h. zentrale Verarbeitungseinheit oder Hauptprozessor
* Sozusagen Gehirn von digitalen Geräten: Rechen- und Steuereinheit, verarbeitet Daten und steuert andere Komponenten
* Besonderheit: Programmierbar.
* **Seite 17**
* Manche CPUs in eingebetteten Systemen sind auf eine Aufgabe bzw. ein spezielles Programm zugeschnitten
* Roboterarme in Industrieanalagen
* PC, Smartphone CPU frei programmierbar, sodass Anwendungen frei programmiert werden können
* In keinen speziellen technischen Kontext eingebettet
* Da wird zwischen zwei Systemen unterschieden

1. die CPU allein als Rechen- und Steuereinheit mit zusätzlichen Coprozessoren:

* Speichercontroller, IO-Controller oder Grafikprozessoren
* so ist es in PCs
* Intel Core 2 Q8200 mit 4 Cores à 2.33 GHz ist eine CPU für einen PC, 456 Millionen verbaute Transistoren (CPU zeigen)

1. SoC (Handy, RPi):

* Im Chip sind neben CPU noch andere Dinge integriert
* Grafikprozessoren, Speichercontroller, IO Controller
* Beispiel:
* RaspberryPi Pico ist ein Mikrocontroller, man kann damit also beispielsweise Elektronikgeräte steuern (RPi Pico zeigen)
* ganz abgespaced in heutigen SoCs:
* Bildprozessoren, die direkt die Bildnachbearbeitung ausführen und damit nicht CPU oder Grafikprozessor belästigen
* Coprozessoren für Matrixoperationen, die für KI-Anwendungen wichtig sind und somit Performance Boosts für solche sind
* Beispiel Apple A15 Bionic Chip im IPhone 13, der eben genau das enthält mit 15 Milliarden Transistoren auf 107,68 mm2
* **Seite 18**
* Jetzt wissen wir, was eine CPU also ist: Ein frei programmierbares Rechen- und Steuerwerk in verschiedenen digitalen Geräten
* Jede CPU besteht aus vielen unterschiedlichen Komponenten, die für verschiedene Aufgaben verantwortlich sind
* Am Ende sind diese durch eine Kontrolleinheit verknüpft und werden durch diese gesteuert, so dass Befehle ausgeführt werden können
* Es gibt verschieden Architekturen für CPUs, die deren inneren Aufbau festlegen und auch diverse Spezifikationen
* Bsp.: wo Befehle und Daten gespeichert werden, welche Befehle sie interpretieren können, wie sie mit anderen Komponenten außerhalb der CPU kommunizieren
* -SAP Architektur ist eine vereinfachte Architektur, an der man eben den Aufbau und die Funktionsweise leicht veranschaulichen kann
* Zunächst die verschiedenen Komponenten in der CPU anschauen

**Komponente 1: Rechenwerk der CPU und ALU:**

* **Seite 19**
* Wie rechnet eine CPU?
* Schriftliches addieren analog zu Binärzahlen
* Tafel: 11001 (25) + 11100 (28) = 110101 (53)
* Ohne c\_in: welche Verknüpfung für sum (XOR), welche Verknüpfung für c\_out (AND)
* Logisim Halbaddierer zeigen
* Halbaddierer, zu dem noch c\_in addiert werden muss
* Übertrag bei einem = Übertrag, also OR
* Logisim Volladdierer zeigen
* Verschaltung von Addierern mit c\_in und c\_out
* Logisim 8-Bit Addierer zeigen
* Subtraktion: Zweierkomplement addieren
* B muss in das Zweierkomplement umgewandelt werden
* Logisim 1-Bit Rechenwerk
* Negation auf Befehl (SUB Signal) durch XOR (Tafel XOR Tabelle)
* Logisim 8-Bit Rechenwerk zeigen
* und +1 durch c\_in am ersten
* (Logisim zeigen) 1-Bit Rechenwerk um logische Verknüpfungen ergänzen
* Gibt alles gleichzeitig aus => Auswahlschaltung
* Durch 4 Bits soll die Auszugebende Operation angewählt werden
* Operationsübersicht in Logisim zeigen
* Auswahlschaltung ist ein MUX, der einen Eingang auswählt und ausgibt
* MUX in Logisim zeigen und minimal das Prinzip von AND Verschaltungen
* Fertige ALU in Logisim zeigen
* Dann kurz Parameter erläutern, die gesetzt werden müssen
* Flags außerdem noch als Statusanzeigen der ALU Operationen

**Komponente 2: Speicherbausteine der CPU:**

* **Seite 20**
* In der CPU müssen oft Daten zwischengespeichert werden
* sog Register laden Daten und können diese bei Bedarf ausgeben
* SR Latch als bistabile Schaltung, d.h. es kann zwei verschiedene stabile Zustände annehmen.
* NOR Gatter erklären, Set und Reset (In Logisim zeigen)
* Dann RS setzen durch Datenbit und ein Load Signal (In Logisim zeigen)
* Timing in der CPU, Taktgeber Signal, d.h. es muss an einen Takt gekoppelt sein
* (Logisim) Fertiges Register: Reset und CLK nur sagen, dass es da ist, aber nicht wie
* (Logisim) 8 Bit Register mit Rising Edge und Enable als fertigen Baustein

**Komponente 3: Arbeitsspeicher:**

* **Seite 21**
* Nächste Kompontente ist der Arbeitsspeicher
* Register nur als Speicher von einer Information / Datenwort. Es muss aber noch irgendwo das Programm und dazugehörige Daten gespeichert werden
* Generell auch größere Datenmengen
* Arbeitsspeicher (RAM == Random Access Memory) ist ein Cluster von Registern, wobei jedem eine bestimmte Adresse zugewiesen ist
* Über einen Adresseingang wird in Binär ein Register bzw. eine Adresse angewählt, die nun aktiv ist
* Diese kann beschrieben oder ausgelesen werden
* Adressdekodierer dekodiert die Adresse und aktiviert eine Zelle
* Steuerleitung, die über Lese- und Schreibvorgänge bestimmt
* Datenleitung, über die die aus- und eingehenden Daten zu den einzelnen Zellen geleitet werden
* RAM Bausteine mit Registern, d.h. Flipflops sind aber groß, da ein Speicherbaustein aus mehreren Transisoren besteht
* Die reale Implementierung (RAM Chip zeigen) sieht daher anders aus (SDRAM)
* Nur ganz kurz: Über einen Transistor wird ein Kondensator immer wieder neu aufgeladen, so werden die Daten gespeichert (Dynamic)
* Im Logiksimulator ist DRAM aber nicht realisierbar und SRAM mit einer gewissen Größe von Grund auf zu bauen ist viel zu aufwändig
* RAM Baustein in Logisim zeigen
* vorgefertigter RAM-Baustein mit 8 Bit Adresse, 8 Bit Datensignal und Steuersignale Read und Load
* Normalerweise ist der RAM nicht in der CPU selbst, sondern extern und auch nicht direkt von der CPU, sondern von Speichercontrollern angesteuert
* Die CPU führt jegliche RAM Operationen über den Speichercontroller aus
* Hier in dem Fall der SAP-Architektur ist der RAM aber in der CPU, da er halt einfach benötigt wird, um Programm und Daten zu speichern

**Zwischenübersicht:**

* Aus diesen Komponenten kann ein Grundgerüst der CPU gebaut werden
* Kann Daten verarbeiten, rechnen, speichern
* SAP Architektur PDF zeigen
* Datenbus, über den die Komponenten Daten austauschen
* A-Register für Operanden und Ergebnis der CPU, TMP Register für zweiten Operanden
* A und TMP direkt mit der ALU und an den Bus, ALU an den Bus, Flags in Register
* B- und C-Register für weitere Datenspeicher
* RAM Modul mit Adressregister, um die angewählt Adresse festzuhalten, während der RAM über den Bus Daten austauscht
* Ausgabe Register, das auszugebende Werte einfach speichert und direkt an ein einziges Peripheriegerät ausgibt: Die Dezimalanzeige
* Logisim Zwischenstand zeigen
* CLK Signal: CPU ist getaktet, damit Daten synchron zueinander vom und auf den Bus
* Reset Signal: CPU in Grundzustand, Register leeren z.B.
* Steuersignale an die Seite, über diese können die Komponenten gesteuert werden
* Beispielhaft Daten verschieben im Schaubild PDF
* In Logisim zeigen:
* 0 in A, 2 in B, 3 in C, dann immer zwei und 3 addieren auf das A Register => Steuersignale

**Thema 4: Befehlszähler:**

* **Seite 22**
* Jetzt: Programmierung der CPU bzw. Steuerung der Komponenten
* Wenn ein Programm ausgeführt wird, dann ist das Programm in Form von einzelnen Befehlen gespeichert und zwar im RAM
* Alle Befehle sind aufeinanderfolgend
* Die Befehle sind auch logischerweise jeweils 1 Byte bei einer 8 Bit CPU, wobei aber zu einem Befehl manchmal ein Parameterwert übergeben werden muss
* Befehl LVA lädt einen Wert in das A-Register, der Wert wird als Parameter mitgegeben
* Der Befehl ist einer RAM Adresse (LVA in bin), der Parameter ist in der Folgenden Adresse (0b00101010) (Tafel LVA und Parameter)
* Das heißt, jetzt wird eine Komponente benötigt, die den Überblick darüber behält, bei welchem Teil des Programms wir uns befinden
* Das übernimmt der sogenannte Befehlszähler (auf ihn in PDF zeigen)
* recht kompliziert auf Logikebene, daher nur das Grundprinzip
* Binärzähler, der, wenn über ein Steuersignal IC aktiviert, bei jedem Taktsignal hochzählt
* Der Befehlszähler ist sozusagen ein Register, das die Adresse des nächsten, auszuführenden Befehls speichert
* und zählen kann, damit ein Befehl nach dem anderen abgearbeitet wird
* außerdem muss der Befehlszähler für bestimmte Befehle (Sprungbefehle) bestimmte Werte laden können
* LC Signal, sodass der Wert vom Bus in den Befehlszähler geladen wird und von wo aus dann weiter gezählt werden kann
* Die inneren Vorgänge sind dabei egal
* Eingebauten Befehlszähler in Logisim fertige CPU zeigen
* kurzen Einblick über Logisim geben: JK Als Erweiterung RS, Verknüpfen zum Zählen, dann Logik zum Setzen von JK zum 1 Bit Pointer und daraus 8 Bit Pointer

**Thema 5: Kontrolleinheit:**

* **Seite 23**
* Finale Komponente, die die Befehle aus dem RAM dekodiert und die einzelnen Komponenten der CPU über die Steuersignale entsprechend ansteuert
* Kurzen Überblick Steuersignale in Schaubild und was ist das Steuerwort
* Kontrolleinheit in der gesamten CPU zeigen:
* Hauptbestandteil der Kontrolleinheit ist ein ROM
* ROM ist eine Art Nachschlagwerk: Wie RAM, nur dass die Inhalte der Adressen hardcoded sind. Über den Adresseingang kann man also eine bestimme Ausgabe zu einer Adresse anfordern
* Als Eingabe geben wir also den Befehl, der dekodiert werden soll (Logisim)
* In der zugehörigen Adresse zu einem Befehl befindet sich dann das entsprechende Steuerwort für die Steuersignale
* So werden Befehle dekodiert.
* Es werden aber mehrere Zyklen benötigt, um einen Befehl auszuführen, da die Maschinenbefehle komplexer sind als Lade vom Bus in das A Register oder so
* Wenn schon: Lade einen bestimmten Wert, der als Parameter übergeben wurde, in das A Register, dazu aber gleich mehr
* Über mehrere Zyklen d.h. in die Adressleitung wird auch noch mit einbezogen, in welchem Schritt wir uns befinden über einen Binärzähler (Logisim)
* So findet sich eine Adresse für jeden Einzelschritt eines Befehls
* SAP-Schaubild
* Kontrolleinheit operiert nach FETCH DECODE EXECUTE
* Fetch: Unabhängig des Befehls muss in den ersten zwei Schritten folgendes Geschehen:
* EP, LM
* RR, LI, IP
* Dann kann es losgehen mit der eigentlichen Execution des Befehls. Ab Schritt drei wird dann auch der tatsächliche Befehl in der Adresse des ROM berücksichtigt
* Auszugebende Steuerworte für LVA
* EP, LM
* IP, RR, LA
* CC

**Thema 6: Programmierung der CPU:**

* Excel Datei: Kurzen Überblick über alle Befehle: nur was diese machen, in Gruppen aufgeteilt
* Programm schreiben für die Zweierpotenzen, erläutern
* Dann andere Programme zeigen
* Wenn Zeit ist, Compiler vorführen