Europaschule Gymnasium Stephaneum in Zusammenarbeit mit der Kreativwerkstatt Aschersleben

Fachbereich: Mathematik/Informatik

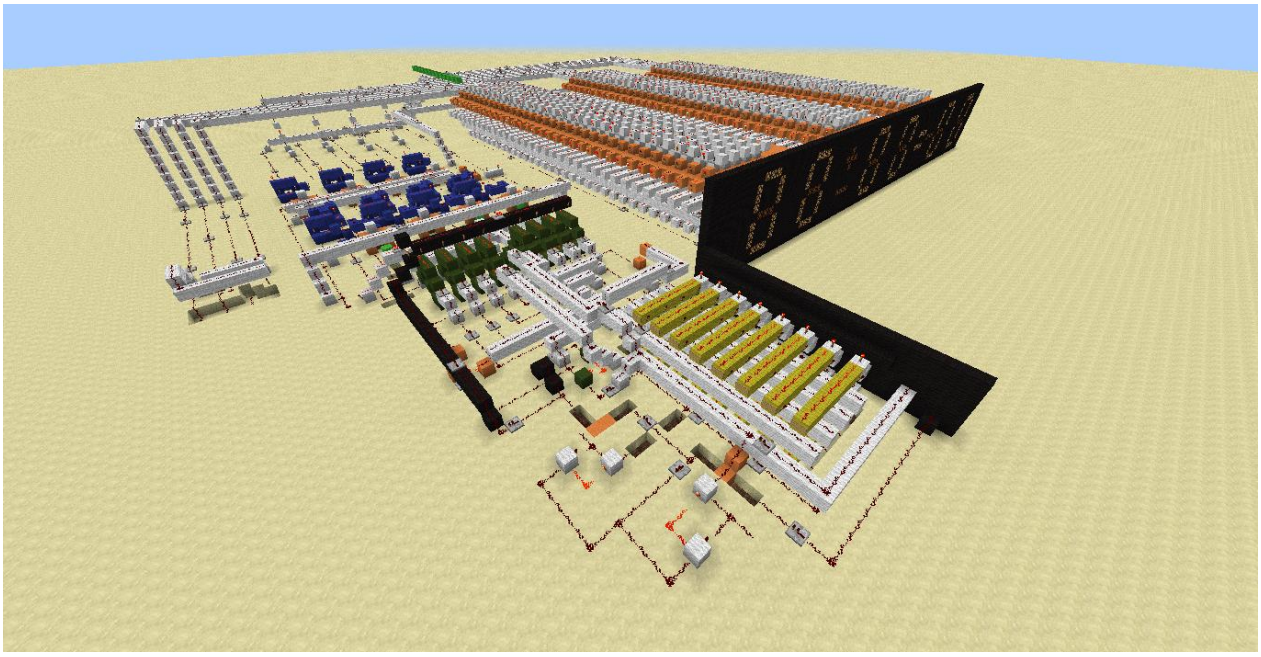
Jugend forscht

Thema:

„Rechenarchitektur in 3D- Simulation von komplexen Vorgängen einfach dargestellt“

Teilnehmer:

Maximilian Schmidt, Hannes Steiner



Betreuer:

Bernd Czech

Gliederung

1. Persönliches Interesse und Motivation an unserem Vortrag

Unser Interesse liegt im Bereich der Simulation und Modellbildung im Schaffen und Bauen von virtuellen Welten in 3D. Besonders fasziniert uns das Spiel Minecraft, wo diese Möglichkeiten relativ einfach und anschaulich gegeben sind und der Kreativmodus eine Menge von Tools beinhaltet. Voraussetzung für solche Simulationen sind leistungsfähige Rechensysteme und ein Betriebssystem, das diese Rechenarchitektur optimal unterstützt. Ziel unserer Arbeit soll eine grafische Veranschaulichung der binären Verarbeitung über die Rechenarchitektur sein. Unsere allgemeine Motivation besteht darin unser Verständnis über informatische Prozesse zu erweitern für ein weiterführendes Studium.

2. Minecraft als Unterrichtsfach?

Neben normalen Simulationsprogrammen, wie z.B. Crocclips, um elektronische Schaltung nachzubauen, kann Minecraft auch im Unterricht genutzt werden, da es möglich ist im Team zu arbeiten. Durch den Mehrspielermodus in Verbindung mit den in Minecraft integrierten Tools kann man zusammen an einem komplexen Rechensystem arbeiten. Nach unserer Vorgehensweise könnten Schüler sinnvoll verschiedene Schaltungen in Minecraft logisch und verständlich nachbauen. Dadurch wird zum einen das logische Denken, vor allem für ein weiterführendes Studium, angeregt, als auch die Zusammenarbeit gefördert, welche den Spaß und das Interesse an verschiedenen Schaltlogiken unterstützt.

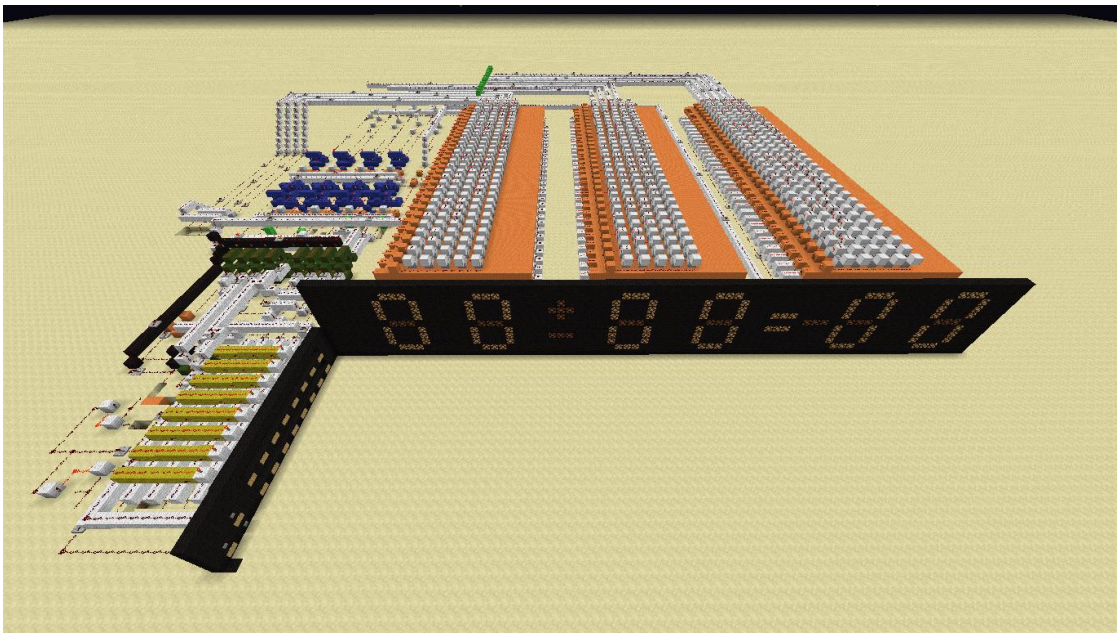


Bild 1: 4-Bit Rechensystem

Das Bild zeigt ein von uns gebautes 4-Bit Rechensystem. Ein Schüler kann sich z.B. ein solches Modell als Vorbild nehmen und weiter entwickeln. In der Schule kann sich ein ähnliches Projekt über mehrere Unterrichtsstunden erstrecken. Wir als Schüler der 10. Klasse haben uns über mehrere Wochen mit allgemeiner Rechenarchitektur beschäftigt und können bestätigen, dass wir unser Wissen als auch unser Interesse ausgebaut haben und nun auch angeregt sind eine komplexere Architektur zu entwickeln.

3. Von-Neumann-Rechner

3.1 Erklärung

Der Von-Neumann-Rechner dient heutzutage vielen Rechensystemen als Vorlage, obwohl er schon im Jahre 1945 von John von Neumann als Konzept für universelle Rechensysteme vorgeschlagen wurde. Doch wie funktioniert diese Architektur und wie ist sie aufgebaut?

Die erste präsentierte Variante bestand aus nur fünf Teilen. Einem Steuerwerk, dem Rechenwerk, sowie Speicher, Eingabe- und Ausgabewerk.

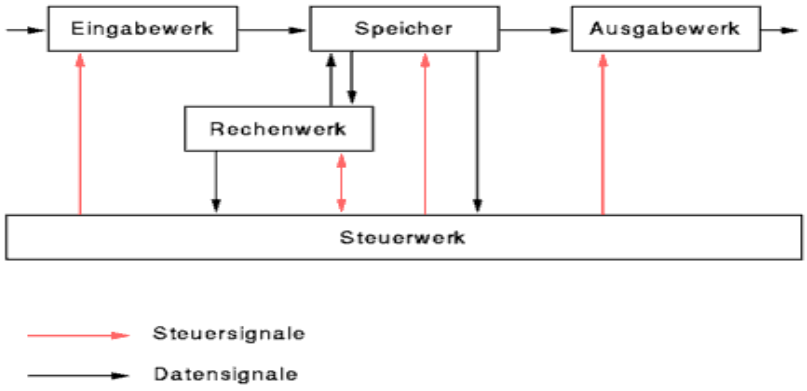


Bild 2: Von-Neumann Prinzip

Trotz der schon fortschrittlichen Architektur benötigte, wie auch heute, die Hardware eine bestimmte Software, um zu arbeiten. Der Benutzer gab von außen eine Bearbeitungsvorschrift (Programm) in den Rechner ein. Diese wurde dann in den Speicher gelegt und machte das Prinzip erst funktionsfähig. Wird nun ein Auftrag über das Eingabewerk eingegeben, werden alle Daten vorerst im Speicher gesichert. Dabei ist zu beachten, dass der Speicher in mehrere Zellen unterteilt ist, welche gleichgroß und adressiert sind. Das Steuerwerk sendet nun ein Steuersignal zum Speicher und erkennt die benötigten Zellen, welche dann die nötigen Informationen an das Rechenwerk weitergibt.

Die arithmetische-logische Einheit (ALU) (siehe Einkernprozessoren) geht dann nach der Rechenlogik vor und gibt das aufgegebene Zwischenergebnis wieder an den Speicher zurück, welcher nun vom Steuerwerk den Befehl erhält, das fertige Ergebnis an das Ausgabewerk weiterzugeben. Das jeweilige Ergebnis wird dann digital oder analog dargestellt, sodass der Benutzer es abgleichen kann.

3.2 Simulation

Um das Prinzip logisch darzustellen haben wir das Spiel Minecraft genutzt um, an einem simplen 4-Bit Rechensystem die Funktionsweise zu erklären. Zuerst beginnt ein Rechenprozess mit der Eingabe. Dabei werden die Dezimalzahlen in Binärzahlen codiert (siehe Bild 4, gelb). Der Benutzer gibt manuell die beiden Summanden ein und bestimmt eine Rechenart (addieren, subtrahieren). Die in Binärzahlen angegebene Rechnung wir dann im Speicher gesichert (siehe Bild 5, grün). Im Speicher wird dann automatisch das Ergebnis abgerufen und weiter in das Rechenwerk (arithmetische-logische Einheit) geleitet. Durch das Tool „Redstone“ können verschiedene Schaltungen realisiert werden. Deshalb besitzt das Rechenwerk auch verschiedene Gatter, die ineinander übergreifen (AND-Gate, OR-Gate, … etc. (siehe 10. Schaltlogik.)) Durch das ähnliche Prinzip wird dann ein Ergebnis wieder in Binärzahlen erzielt. Letztendlich fließen drei Signale in die

drei digitalen Displays (beide Summanden und Summe). Wie im letzten Bild (Bild 7/8) zu sehen, kann das Ergebnis mithilfe von sich einschaltenden Lampen erkannt werden. Das Steuerwerk in diesem System in Kombination mit den automatischen Vorgängen der in Minecraft integrierten Tools. Mehr darüber erfährt man in unserer Simulation.

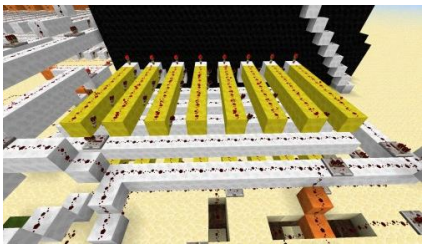



Bild 3/4: Eingabe

Bild 5: Speicher

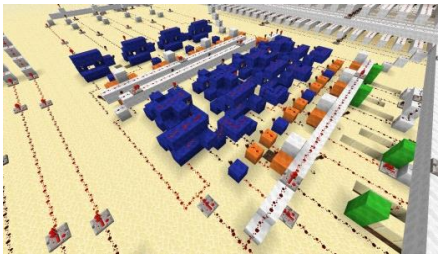


Bild 6: ALU

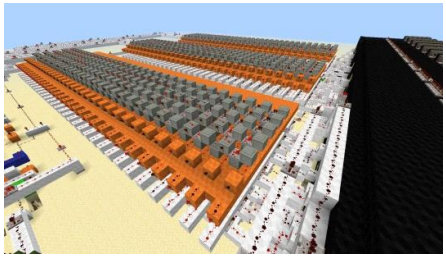
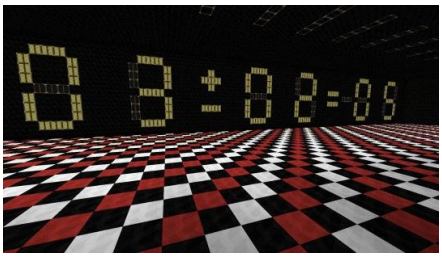



Bild 7/8: Ausgabe

3.3 Geschichtliche Entwicklung

Seinerzeit war die VNA (von Neumann Architektur) spektakulär, auch wenn schon vorher ähnliche, noch nicht ausgereifte Prinzipien entwickelt wurden. Eine ähnliche Idee wurde von Konrad Zuse entwickelt, welcher den ersten programmierbaren vollfunktionsfähigen Computer der Welt baute. Jedoch wurde überliefert, dass von Neumann die Ideen von Konrad Zuse unabhängig von seinen Vorstellungen umsetzte. Seine Ergebnisse präsentierte er dann 1945 und legte somit einen Grundstein für die Informatik. Mit der Zeit wurden auf seinem Prinzip Rechensysteme weiterentwickelt.

4. Einkernprozessor

Ein Prozessor erfüllt den Zweck, Daten mit arithmetischen und logischen Operationen zu verarbeiten. Die grundlegenden Prozessoren-Chips besitzen nur einen Kern, welcher aus einer arithmetischen logischen Einheit (ALU), mehreren Rechenregistern und einem Basis-Steuerwerk besteht, welches den Datentransport leitet. Der Grundlegende Prozessor besteht aus der arithmetischen logischen Einheit, einem Steuerwerk und mehreren Registern, sowie einem Speichermanager.

4.1 Die arithmetische logische Einheit

Die ALU ist ein elektronisches Rechenwerk, welches verwendet wird um, diverse Rechenoptionen durchzuführen. Diese Rechenoptionen sind dabei in arithmetische und logische Operationen gegliedert. Unter arithmetischen Rechenoptionen versteht man die mathematischen Optionen, wie Addition und Subtraktion sowie unter den logischen Optionen die Verknüpfungen wie z.B. AND, OR und NOT. In Bezug auf unser Rechensystem findet man Zahlreiche dieser Funktionen wieder.

4.2 Das Steuerwerk

Die Funktionseinheit Steuerwerk bezieht sich auf die systematische Befehlsverarbeitung. Codierte Befehle können hier zudem entschlüsselt (decodiert) und erfasst werden. Weiterhin dient es der Weitergabe von Steuerbefehlen an das Rechenwerk.

4.3 Die Rechenregister

Ein Register dient der schnellen Datenmanipulation, sowie dem Zwischenspeichern von Befehlen, Speicheradressen und Rechenoperationen. Zudem können Ein- und Ausgabewerte hier gespeichert werden. Die Größe eines Registers richtet sich nach der Grundverarbeitungsdatengröße, welche dieser nicht überschreiten darf. Zumeist haben neuere Rechensysteme eine Grundverarbeitungsdatengröße von 32 Bit oder 64 Bit. Unser Rechner besitzt dabei ein 4 Bit System, welches bei Digitaluhren und Taschenrechnern benutzt wird.

4.4 Der Speichermanager

Ein System benötigt einen Arbeitsspeicher um, gewisse Operationen durchzuführen. Damit dieses System optimal arbeiten kann, übernimmt der Speichermanager die Einteilung des Arbeitsspeichers. Er stellt fest wie viel Kapazität eine Applikation benötigt und teilt ihr genügend RAM zu.

5. Mehrkernprozessor

Im Gegensatz zu Einprozessoren, besitzen Mehrprozessoren mindestens zwei Prozessoren, was Systeme mit z.B. einem Dualcore (2) leistungsfähiger macht als einen Singlecore (1) Rechner. Durch einen oder mehrere Prozessorkerne, werden diese untereinander entlastet und es können mehr Aufgaben gleichzeitig durchgeführt werden.

5.1 Aufbau

Mehrkernprozessoren besitzen einen ähnlichen Aufbau wie ihre Vorgänger, jedoch bestehen sie, wie der Name schon sagt, aus mehreren Prozessoren. Ein einzelner Prozessor unterscheidet sich nicht von den Einkernprozessoren, jedoch arbeiten hierbei beide oder mehrere Prozessorkerne zusammen.

5.2 Bytereihenfolge

Die Bytereihenfolge beschreibt die Reihenfolge, in der einfache Zahlenwerte gespeichert werden soll. Sie tritt ein, wenn der verlangte Betrag mehr Platz benötigt, als in der kleinsten adressierbaren Einheit gegeben ist. Diese ist normalerweise 1 Byte. Ein Byte wiederum sind 8 Bit. Werden nun zum Beispiel 9 Bit benötigt, bestimmt die Reihenfolge nun in welcher Ordnung sie abgearbeitet werden soll. Entweder zuerst 8 Bit und dann die restlichen 1 Bit oder invertiert, da in diesem Fall 1 Byte (also 8 Bit) die maximale Größe ist. Wie die Reihenfolge nun ist, bestimmen zwei Varianten, der Big-Endian und der Little-Endian. Der Big-Endian beginnt dabei mit dem Höchstwertigen, in unserem Fall also 8 Bit. Genau umgekehrt macht es der Little-Endian, welcher sich zuerst auf den kleinsten Wert bezieht. Er würde deshalb mit 1 Bit beginnen. Heute oft verwendete Intel Prozessoren, arbeiten dabei nach dem Little-Endian Prinzip. Mittlerweile gibt es auch Mischformen, welche beide Varianten besitzen können.

6. Quantencomputer

Neben den bis jetzt am häufigsten eingesetzten Rechensystemen, haben wir uns zudem mit den fortschrittlichen Quantencomputern beschäftigt. Im Gegensatz zu den Digitalrechner basieren ihre Berechnungen nicht auf der oft verwendeten binären Form, sondern auf der Quantenmechanik, welche es erlaubt mit materiellen Objekten zu arbeiten um Objekte, vor allem Elementarteilchen, sinnvoll und systematisch aus diversen Teilchen zusammenzusetzen. Sie erlaubt es uns, ein Objekt sehr präzise bis in den subatomaren Teilchenbereich darzustellen.

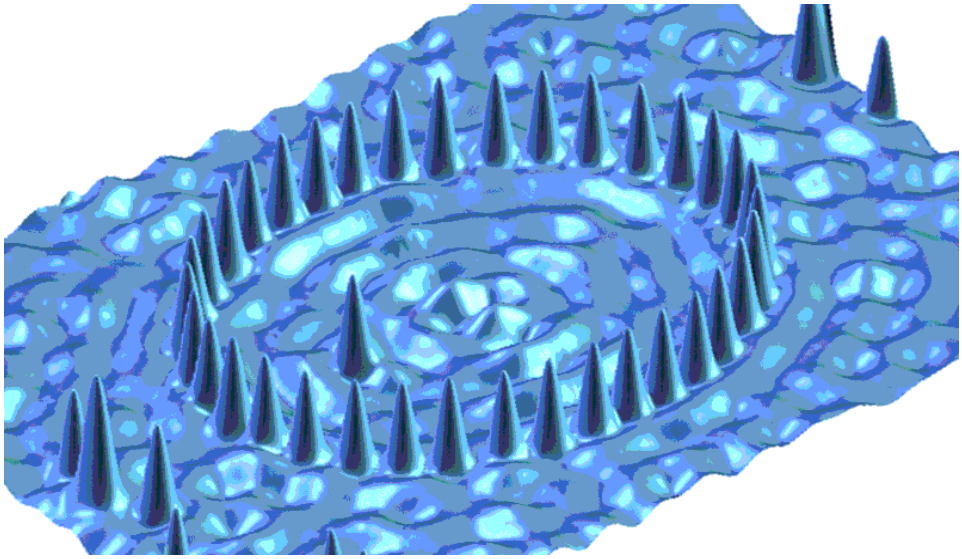


Bild 9: 3D Darstellung

Das hier zu sehende Bild zeigt eine 3-Demensionale Darstellung, welche von einem Atommonteur modelliert wurde. Durch eine derart präzise Darstellung sind solche Systeme revolutionär für die Atom-, Festkörper-, Teilchen- und Kernphysik.

6.1. Aufbau und Funktionsweise

Ein Quantencomputer funktioniert mit QuBit, wobei ein QuBit durch ein beliebiges binäres Quantensystem definiert wird. Wie im binären Zahlensystem können analog und digital die Zustände 0 und 1 dargestellt werden. Durch das Quantensystem können mehrere Systemzustände bestimmt werden. Dabei entstehen Orthogonale Zustände, was beschreibt, dass sich mindestens zwei genau erkennbare Zustände voneinander unterscheiden. Dabei ist zu beachten, dass in fast allen Quantensystemen eine unterschiedliche Zahl an Orthogonalen Zuständen entsteht. Hierbei zeigt sich auch die

Besonderheit der Quantencomputer, da diese unterschiedlichen Quanten nicht als verschiedene, sondern als ein Teil zusammenfassen.

Um nun ein Ergebnis, wie das obere Bild zu erzielen, müssen alle messungsstörenden Objekte, aus dem entfernt werden. Dies wird Mithilfe eines Hochvakuums erzielt. Um andere Teilchenbewegungen extrem zu verlangsamen, wird die Temperatur auf einen Wert möglichst nah an den absoluten Nullpunkt (-273°C) gesenkt. Anschließend sorgt ein Magnetfeld dafür, dass die Ionen eingefangen und in einer Linie aufgereiht werden. Ein Ion steht dabei für ein QuBit. Schließlich werden mithilfe von Photonen, aus einem präzisen Laser, die Energiezustände der Elektronen verändert. Das hier erläuterte Prinzip nennt sich Ionenfalle.

7. Schaltlogik

7.1 Erklärung

Die Schaltlogik bezieht sich auf verschiedene Logikgatter, welche durch unterschiedlichen Aufbau ein bestimmtes Ergebnis erzielen können. Dabei können nur 2 verschiedene Zustände entstehen: 0 für den Zustand false, also falsch und 1 für den Zustand true, also wahr. In unserem Beispiel arbeiten wir mit elektrischen Signalen. Ist das Signal aktiv, steht es für 1. Ist es inaktiv für 0. Durch diese Vorgehensweise können wir unterschiedliche Operationen ausführen und in einem Rechensystem vereinen.

7.2 Gatter (*Gates*)

Die Gatterarten sind: NOT, AND, NAND, OR, NOR, XOR und XNOR. Dabei erfüllen alle eine unterschiedliche Funktionsweise. Um diese Funktionsweise zu erkennen und die dabei entstehenden Zustände ablesen zu können, gibt es eine Wahrheitstabelle für jede Schaltung. Dabei wird die Funktion Y mit jeweils 2 Variablen A und B dargestellt, z.B. Y = A + B. Abzulesen ist nun, welche Kombination erforderlich ist, um Y = 1 zu erzielen. Für die einzelnen Gattertypen sieht das wie folgt aus:

7.3 Beispiel: Adder-Substractor in Minecraft

Da Minecraft viele Tools bietet, um ein solch komplexes System zu modellieren, haben wir uns dabei die verschiedenen Gatter zu Nutze gemacht.

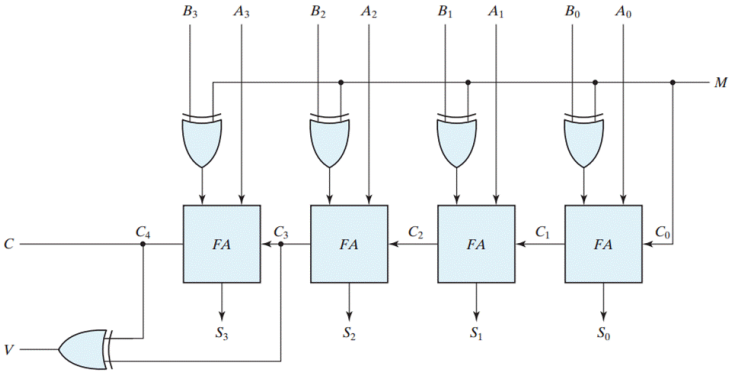


Bild 10: Adder-Substractor Schemata

Das hier aufgeführte Bild zeigt, das Schema der von uns verwendeten „Arithmetischen-logischen Einheit“, einen 4-Bit Adder-Substractor (Addierer-Subtrahierer), welcher es erlaubt Zahlen von 0-15 zu addieren und subtrahieren. Die Eingänge A0 – A3 und B0 – B3 stehen für die Zahlen, welche vorerst im Speicher gespeichert wurden. Sie sind der Schaltlogik entsprechend wahr oder falsch (1 oder 0). Über ein XOR-Gate gelangen sie jeweils in einen Volladdierer (FA). Dieser besteht aus 2 Halbaddierern, welche wiederum aus einem AND- und einem XOR- Gate aufgebaut sind. Darin wird nun das Ergebnis verarbeitet und über die Pfade S0 – S3 zum Display geleitet, wo sie weiter verarbeitet werden. In Minecraft sieht das wie folgt aus:

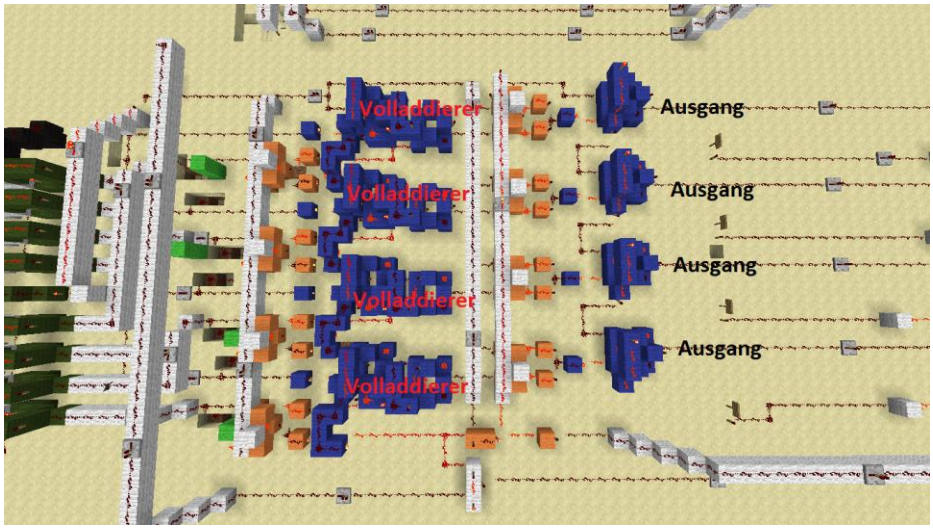


Bild 11: ALU in Minecraft

8. Komparator Parallelschaltung

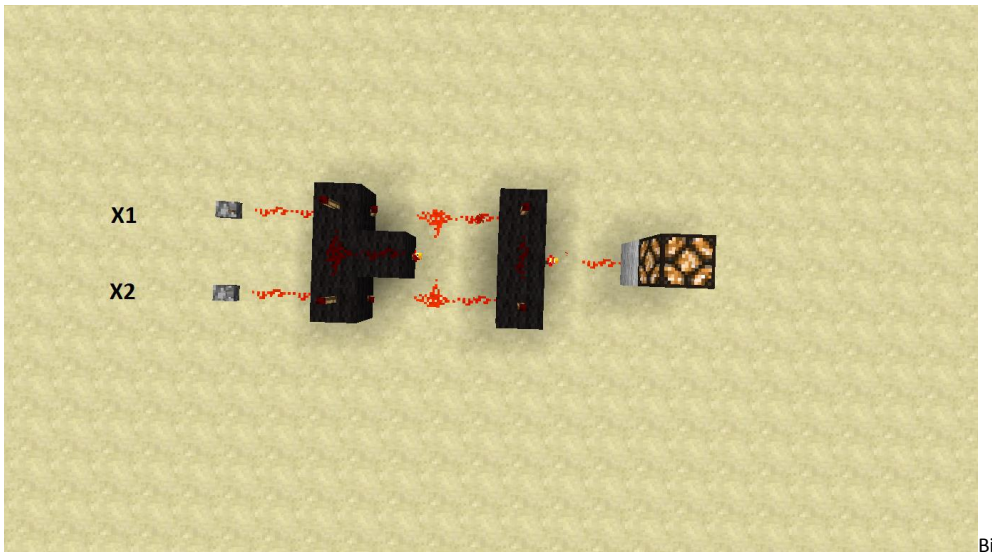


Bild 12: X1, X2 aktiv

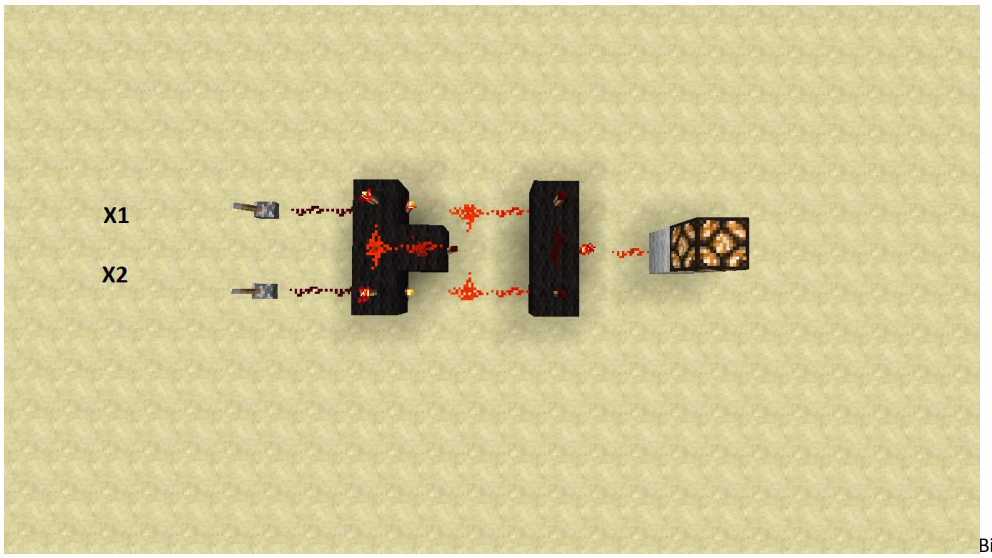


Bild 13: X1,X2 inaktiv

Die oberen Bilder zeigen einen von uns nachmodellierten Vergleicher(Komparator). Wie man erkennen kann, leuchtet die Lampe wenn, beide Zustände gleich sind. Das heißt, wenn X1 und X2 wahr sind bzw. falsch, ist das Ergebnis wahr. Wir haben uns Gedanken gemacht, wie man dieses Prinzip verwenden könnte. Dabei sind wir auf die Idee gekommen eine elektronische Schaltung, wie man sie auch im Haushalt findet, nachzubauen. Nach unserer Vorstellung kann man mithilfe einer solchen Komparator-Schaltung eine Parallelschaltung nachbilden, welche z.B. bei Lampen zum Einsatz kommt. Man aktiviert einen von mehreren Schalter irgendwo in einem Raum und bringt damit die Lampe zu leuchten. Nun kann man einen Schalter auf der anderen Seite des Raumes betätigen, sodass die Lampe wieder erlischt. Dies ist auch umgekehrt möglich.

9. Minimierung von logischen Schaltungen

Karnaugh und Veitch – Verfahren

Das Karnaugh-Veitch-Diagramm, kurz KV-Diagramm, dient der übersichtlichen Darstellung und Vereinfachung Boolescher Funktionen in einen minimalen logischen Ausdruck.

Grundprinzip:

Mit einem KV-Diagramm lässt sich jede beliebige disjunktive Normalform in einen minimalen disjunktiven logischen Ausdruck umwandeln. Der Vorteil gegenüber anderen Verfahren ist, dass der erzeugte Term (meist) minimal ist. Bei einem nicht minimalem Term ist eine weitere Vereinfachung durch Anwenden des Distributivgesetzes (Ausklammern) möglich. Das Umwandeln beginnt mit dem Erstellen einer Wahrheitstabelle, aus der dann die DNF (disjunktive Normalform) abgeleitet wird, die dann wiederum direkt in ein KV-Diagramm umgewandelt wird.

Man erstellt ein KV-Diagramm für *n* Eingangsvariablen mit 2*n* Felder. Das KV-Diagramm wird mit den Variablen an den Rändern beschriftet. Einmal in negierter Form und einmal in nicht-negierter Form. Die Zuordnung der Variablen zu den einzelnen Feldern kann beliebig erfolgen, jedoch ist zu beachten, dass sich horizontal und vertikal benachbarte Felder nur in genau einer Variablen unterscheiden dürfen. Mit Hilfe der Wahrheitstabelle der zu optimierenden Funktion wird in die einzelnen Felder eine 1 eingetragen, wenn ein Minterm (nach Aussagelogik ein spezieller Konjunktionsterm) der Funktion vorliegt, andernfalls eine 0.

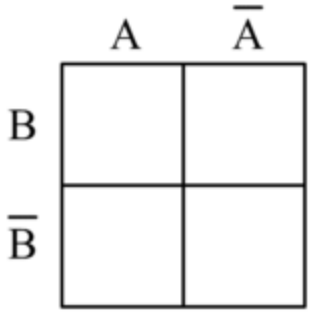


Bild 14: 2x2 KV-Diagramm für 2 logische Variablen (A, B)

10. Quellen

Quellen: http://de.wikipedia.org/wiki/Boolesche_Algebra

http://www.uni.merkertweb.de/p1/schaltlogik_v1.pdf

http://de.wikipedia.org/wiki/Logikgatter

http://de.wikipedia.org/wiki/Quantencomputer

http://www.physik.uni-wuerzburg.de/fileadmin/11030030/Ewelina/quantencomputer/02_QC_AtomeIonen.pdf

http://de.wikipedia.org/wiki/Von-Neumann-Architektur

http://de.wikipedia.org/wiki/John_von_Neumann

http://de.wikibooks.org/wiki/Karnaugh-Veitch-Diagramm:_Beispiele_(Teil_1)

Bildquellen:

http://www.setupsolution.com/wp-content/uploads/2013/08/4-bit-adder-subtractor.png

http://tams-www.informatik.uni-hamburg.de/applets/baukasten/DA/Bilder/von_neumann_rechner_50.gif

http://www.allmystery.de/i/tc3712a_800px-Co_ellipse.gif

http://de.wikibooks.org/wiki/Datei:Karnaugh_map_KV_2Variables_01.png

11. Danksagung

Wir möchten uns hiermit insbesondere bei Herr Czech bedanken, welcher uns mit gutem Rat zur Seite stand und uns die Teilnahme am Jugendforscht Projekt erst ermöglichte. Zudem gilt ein Dank den Lehrern, besonders unserem Schulleiter Herrn Winter, welche uns die Zeit boten, die wir benötigten um unseren Forschungen nachzugehen. Desweiteren bedanken wir uns bei den Veranstaltern von Jugendforscht und der Kreativwerkstadt Aschersleben.

12. Eigenständigkeitserklärung

Hiermit erklären wir, dass alle hier aufgeführten Ideen von uns umgesetzt wurden. Die benötigten Informationen und Bilder sind im Quellenverzeichnis nachzulesen.