Assembler und Mikroprozessoren

Bauteile

Register

Register sind sehr kleine Speichereinheiten (in Mikrosim nur 1 Byte), welche sich im Prozessor oder Bus-System befinden. Der Prozessor kann sehr schnell auf sie zugreifen, jedoch gibt es auch in modernen Prozessoren nur sehr wenige. In Mikrosim gibt es diese Register:

- AX, BX: Speichern von Zwischenergebnissen
- AR: Adresse des Bytes im RAM welches geschrieben oder gelesen wird
- DR (Data Register): Wert der in der ALU zusammen mit einem anderen Wert verarbeitet wird. Werte die aus dem RAM gelesen werden oder dorthin geschrieben werden solle landen hier. Mikroprogramme nutzen den Inhalt dieses Registers als Parameter.
- OP (Operation Pointer): Addresse der nächsten Operation im MPS
- IP (Instruction Pointer): Addresse der nächsten Operation im RAM

Busse

Busse sind die Leitungen auf welchen die Daten transportiert werden

Tore

Tore regulieren den Datenfluss auf Bussen. Es können niemals mehrere geöffnete Ausgangstore auf den gleichen Bus führen. Dies führt zu einem Kurzschluss

ALU

Die artihmetisch-logische Einheit ist der "wahre" Prozessor. Sie performt Operationen mit Eingangswerten. Welche Operation ausgeführt wird, bestimmen Steuerleitungen. In Mikrosim beherrscht die ALU nur Addition und Subtraktion (gesteuert durch Tor D).

Takt

Kein Bauteil im engeren Sinn aber trotzdem ein wichtiges Konzept sind Takte. Ein ALU-Takt besteht aus zwei Phasen: Zuerst werden die Ausgangstore geöffnet und geschlossen (Bus Takt 1) und dann die Eingangstore geöffnet und geschlossen (Bus Takt 2)

RAM

Der RAM ist der Speicher, in dem Daten und Programme gespeichert werden. Alle aus dem RAM gelesene Daten müssen zuerst im DR Datenregister zwischengespeichert werden. Der RAM kann maximal 256*Größe Byte der Register enthalten, da jedes Byte im RAM einer Addresse zugerodnet werden muss. In Mikrosim ist die Registergröße 1 Byte. Daher gibt es 256 (2^8) mögliche Addressen und der RAM fasst 256 Byte.

Von-Neumann-Rechner

Ein Von Neumann Rechner zeichnet ich dadurch aus, dass im RAM Programme *und* benötigte Daten liegen. Dadurch kann ein Rechner im Gegensatz zu anderen Werkzeugen für alle möglichen Aufgaben verwendet werden.

Assemblerbefehle

Ein Assemblerbefehl drückt eine Operation aus, die in einem oder mehreren ALU-Takten durch Toröffnungen realisiert wird. Da diese Toröffnung je nach Architektur des Prozessors verschieden sind, werden diese Befehle direkt im Prozessor implementiert. Dazu werden für die Operationen Mikroprogramme im Mikroprogramm-Speicher (MPS) des Prozessors gespeichert. Diese sind von Programmierern nicht veränderbar und werden bei der Herstellung festgelegt. Jeder Assemblerbefehl besitzt maximal einen Paramter. So wären die Befehle *mov AX, BX* und *mov BX, AX* verschiendene Befehle. (In Pseudocode: *function mov_ax(bx: register)* und *function mov_bx(ax: register)*) Programmierer schreiben Programme indem sie diese Befehle kombinieren. Schreibt man ein Programm in einer kompilierten Sprache wie C oder Java (obwohl es bei Java in Wahrheit ein wenig komplizierter ist), so wandelt ein Programm namens *Compiler* den geschriebenen Quellcode in Maschinencode um, welcher eine Abfolge von Assemblerbefehlen ist.

Von-Neumann-Zyklus

Der Von Neumann Zyklus definiert, wie befehle und Parameter aus dem RAM geladen und ausgeführt werden. Jeder Assemblerbefehl besteht in Mikrosim aus 2 Byte. Das erste ist die Addresse des Befehls im MPS, das zweite der Parameter.
Der Von Neumann Zyklus besteht aus mehreren Schritten.
Im FETCH Schritt wird das auszuführende Programm geladen.
Zu Beginn eines Mikroprogramms muss der Parameter im DR Register liegen. Daher wird der Parameter im FETCH OPERANDS Schritt dorthin geschoben.
Im EXECUTE Schritt wird das Programm mit dem Parameter in DR ausgeführt.
Auf Prozessor Ebene geschieht dies: (Es empfiehlt sich das Mikrosim Skript auf Seite 9 nebenbei anzuschauen)

1. FETCH: Die Addresse des Mikroprogramms, welche an der Stelle des Instruction Pointers (IP) im RAM liegt, wird in den Operation Pointer (OP) geladen. Hiermit wird der auszuführende Befehl festgelegt. Der Instruction Pointer wird um eins erhöht.

2. FETCH OPERANDS: Die Parameter des Befehls werden geladen. Hierzu wird der Wert auf den der IP verweist in das Data Register (DR) geladen. Der IP wird um eins erhöht und verweist nun auf den nächsten Befehl.

3. EXECUTE: Der Prozessor wählt als Folgeaddresse den Wert im OP und springt dorthin. Dies geschieht indem die Folgeaddrese auf 0 gesetzt und das Tor F geöffnet wird. Somit ist die neue Addresse 00 (FA) + OP. Das dortige Mikroprogramm wird ausgeführt. Danach springt das Programm an den Anfang zurück.

Sprungbefehle

Es gibt unbedingte Sprungbefehle (JMP), welche zu einem Sprung an eine gewisse Addresse im MPS führen, und bedingte Sprünge (in Mikrosim nur JS) welche nur unter gewissen Umständen funktionieren. JS löst in Mikrosim nur aus, wenn der in AX gespeicherte Wert negativ ist.

Netzwerke

IPV4

Eine IP Addressse besteht aus 4 Byte, welche durch Punkte getrennt werden. Somit kann jede Ziffer 0-255 betragen. Eine IP Addresse besteht aus einem Netzwerkteil, welcher das Netzwerk definiert, und einem Geräteteil, welcher das Gerät im Netzwerk identifiziert. Wie viele Bit Teil des Netzwerkteils sind, wird durch die (Sub)netzmaske bestimmt. Eine Subnetzmaske von 255.255.0.0 (binär: 11111111.11111111.00000000.00000000) bedeutet, dass die ersten beiden Ziffern der Netzwerkteil sind. Eine Netzmaske von 255.240.0.0 (11111111.11110000.00000000.00000000) teilt die ersten 12 Bit dem Netwerkteil zu. Man gibt diese Netzmasken auch in Kurzschreibweise /Anzahl_der_Bits_der_Netzmaske an. Die beiden Beispiele wären somit /16 und /12. Diese Kurzschreibweise setzt man hinter die IP Addresse: 192.168.3.232 /24. Subnetzmasken können keine Löcher enthalten. In Binärschreibweise müssen somit alle Einsen in Reihe stehen. Netzmasken wie 11110011.x.x.x sind dicht möglich. Die IP x.x.x.255 ist in einem Netz mit /24 Netmaske die Broadcast Addresse. Alle Pakete, die an sie gesendet werden, werden an alle Geräte im Netzwerk versandt. Die Anzahl der IPV4 Addressen ist begrenzt. Da wir sie fast ausgeschöpft haben, ist ein Wechsel auf IPV6 nötig.

E-Mail

- MTA (Mail-Transfer-Agent): Beim Verschicken sendet der Absender die Email an seinen MTA. Dieser versendet die Mail an andere MTAs, bis einer von ihnen den MDA des Empfängers kennt. Die Mail wird dann an den MDA gesendet. Zur Kokmmunikation zwischen Absender und MTA wird das SMTP (Simple-Mail-Transfer-Protocol) verwendet
- MDA (Mail-Delivery-Agent): Der MDA speichert das Postfachs des Besitzers und sendet sie auf Anfrage an den Nutzer. Dies geschieht entweder durch das veraltete POP3 (Post Office Protocol) Protokoll, bei dem die Daten nach dem Abrufen vom MDA gelöscht werden und dem IMAP (Internet Message Access Protocol) Protokoll, bei dem die Nachrichten auf dem MDA verbleiben.

DNS Resolution / Auflösung (keine Ahnung ob das drankommt)

Wird ein Domain Name aufgelöst, so geschieht dies Rückwärts. Zuerst wird die Anfrage an einen Core Nameserver gesendet, welcher basierend auf der Top-Level-Domain (.de, .com, etc.) die Anfrage an den Nameserver des Betreibers dieser TLD weiterleitet (.de: Denic, .com: Verisign). Dieser Server leitet nun die Anfrage an den Nameserver des Betreibers des Domain Namens (schiller-og, google, youtube) weiter. Dieser lokale Namerserver gibt basierend auf der Subdomain (www., ftp.) die passende Ip zurück. Diese IP wandert die Serverkette hoch und landet am Ende beim User