# 1. Allgemeines

## 1.1 Die wichtigsten Architekturen

Die wichtigsten Architekturen im Bereich der Computerprozessoren sind: der **x86** (IBM AT oder kompatibel), **PPC** (Apple PowerMac, IBM Mainframes), **MIPS** (Sun) und **ALPHA** (Compaq Server).

## 1.2 Geschichte des x86

Der erste Prozessor der als Chip auf den Markt kam war 1971 der **4004** von **Intel,** mit einem Prozessortakt von 108 kHz. Alle seine Vorgänger arbeiteten noch mit Röhren und Transistoren.
Von diesem Prozessor aus ging die Weiterentwicklung über einige andere Typen (8008, 8080...) zum **8086**, zu dem noch alle aktuellen Pentium-Modelle und deren Ableger (Athlon, Cyrix...) kompatibel sind. Daher auch der Name **x86**.
Der 8086 arbeitete schon mit einem 16 Bit Datenbus und war damit seiner Zeit voraus. Deshalb brachte Intel die „Weiterentwicklung“ des 8086, den **8088**, auf den Markt. Dieser arbeitete nur noch intern mit 16 Bit, extern aber nur mit 8 Bit. Der 8088 wurde im IBM PC-XT (extended Technology) zum vollen Erfolg.
Alle weiteren Prozessoren die Intel entwickelte waren kompatibel zum 8086, der 80286, der 80386, der 80486 und auch der Pentium (80586).
Mit dem 80486 wurde erstmals ein Mathematischer Coprozessor auf dem Chip integriert. Mit dem Pentium verließ Intel die Schiene der x86-Namensgebung, obwohl der Pentium eigentlich ein 80586 ist. Doch Namen lassen sich Markenrechtlich besser schützen als Zahlen.
Intel brachte bis zum 80486 auch immer die günstige Einsteigervariante **...SX** auf den Markt, was aber eigentlich immer eine schlechte Wahl war. Beim 486 SX zum Beispiel wurde auf den Coprozessor verzichtet, was ihn langsamer machte als einen 386 mit externem Coprozessor.
Natürlich gab es auch Konkurrenz für Intel. AMD, Nexgen, IBM/Cyrix versuchten immer wieder die x86-Prozessoren von Intel, die schon zum Standard geworden sind, nachzubauen und neue Features zu integrieren, sie hinkten dem Marktführer aber in der Entwicklung immer hinterher. Erstmals konnte **AMD** mit dem Kauf von **Nexgen** und der daraus resultierenden Entwicklung des **K6** Intel die Leistungskrone, zumindest bei bestimmten Anwendungen, die hauptsächlich Integerberechnungen durchführten, streitig machen, und das zu einem wesentlich besseren Preis als der Pentium zu haben war. Intel reagierte darauf mit dem **Pentium II**. Intel hatte damit die Leistungskrone zwar wieder inne, der PII war aber noch wesentlich teurer als der Pentium. Außerdem ruhte AMD sich nicht auf seinen Lorbeeren aus und entwickelte den **Athlon** (K7), der jetzt dem **Pentium III** in allen Bereichen überlegen war.
Seither streiten sich Intel und AMD abwechselnd um die Leistungskrone. Zurzeit hat zwar Intel im Bereich der Taktfrequenzen die Nase vorn, AMD kann aber mit gleichem Takt eine höhere Leistung erzielen. Dies ist auf Architekturunterschiede zurückzuführen.

# 2. Grundlagen der Prozessorarchitektur und Herstellung

## 2.1 Der prinzipielle Aufbau eines MOSFET

Der MOSFET (Metalloxid-Feldeffekt-Transistor) ist das wichtigste Bauteil eines Prozessors. Der Grundsätzliche Aufbau ist sehr einfach: „Zwischen 2 stark dotierten Elektroden liegt ein Kanal der entgegengesetzten Dotierung – bei n-Kanal-MOSFETs liegt also zwischen den n-dotierten Source- und Drain-Elektroden ein p-dotierter Kanal. Über diesem Kanal und durch eine Oxid-Schicht elektrisch davon isoliert befindet sich eine dritte, die so genannte Gate-Elektrode.“[1]

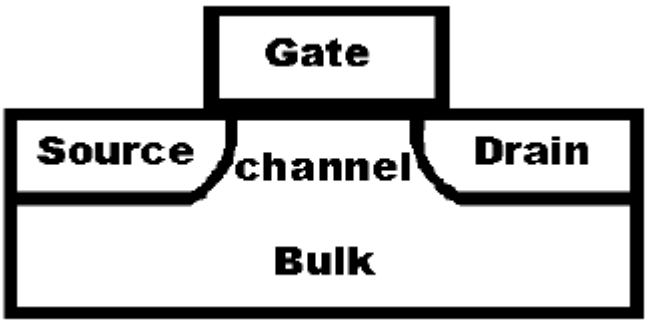


**Abbildung 1: Aufbau eines MOSFET**

Die Funktion: „Liegt an der Gate-Elektrode eine hinreichend große positive Spannung an, werdenElekronen aus dem p-dotierten Halbleiter angezogen; unter dem Oxid bildet sich ein dünner, leitender Kanal aus. (...) Die Leitfähigkeit des Kanalgebietes (channel) wird durch das auf die Landungsträger einwirkende elektrische Feld der Gate-Spannung gesteuert. Oberhalb einer spezifischen Schwellenspannung (...) wird der Kanal leitend und der Transistor schaltet durch.“[2] Der tatsächliche Aufbau eines MOSFET sieht allerdings etwas anders aus.

[1] c´t 5/2000 Seite 260: Unter der Haube. Optimierung von CMOS-Transistoren
[2] c´t 5/2000 Seite 260: Unter der Haube. Optimierung von CMOS-Transistoren (gekürzt)

### *2.2 Tatsächlicher Aufbau eines MOSFET*

Durch die zunehmende Verkleinerung der Strukturen treten einige Probleme auf, die gelöst werden müssen. Das größte Problem sind „heiße“ Elektronen, diese sind besonders Energiereich und müssen abgebremst werden um den Transistor nicht zu beschädigen. Dazu werden schwächer dotierte Zonen, die LDD (Lightly Doped Drain) eingeführt, die wie eine Art Vorwiderstand wirken. Weiterhin muss noch dafür gesorgt werden, die Schaltgeschwindigkeit zu erhöhen. Dazu muss man die Kapazität verringern, was durch den Einbau einer „Retrogade Well“ erreicht wird. Das Gebiet „Pocket“ dient der Verringerung der Leckströme.

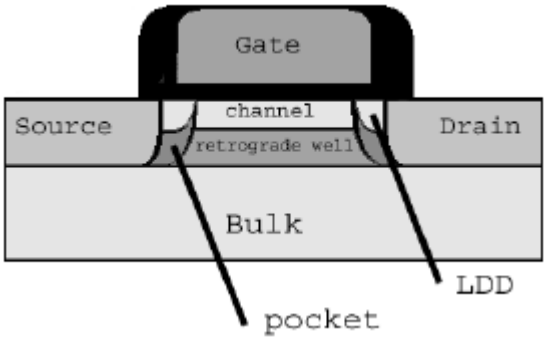


**Abbildung 2: Tatsächlicher Aufbau MOSFET**

### *2.3 Gründe für Verkleinerung von Srukturen*

Um höhere Geschwindigkeiten zu erreichen ist es zwingend notwendig die Strukturen ständig zu verkleinern. Dafür gibt es 2 Gründe: Je größer das Die, desto schwieriger ist es zu kühlen. Zweitens: Je kleiner die Strukturen, desto schneller können die Transistoren schalten. Folglich sollte das Die möglichst klein sein und die Packungsdichte sehr hoch.

### *2.4 Herstellung der Prozessoren*

Über die logische Umsetzung des Designs auf dem Chip schweigen die Hersteller. Bevor es jedoch zu einer logischen Implementation auf einem real existierenden Chip kommt, werden die das neue Design und die Architektur erstmal mit einer Software simuliert. Dies ist ein sehr wichtiger Schritt, denn in der ausgedachten Logik stecken noch sehr viele Fehler (bugs). Trotz der zahlreichen und ausführlichen Tests bleiben Fehler immer wieder unentdeckt, oder zeigen sich erst später in der Serie. Beispiele dafür wäre die Einführung des Pentium, der bei bestimmten Gleitkommaoperationen Fehler produzierte. Oder die Einführung des P III 1,1 GHz, den Intel wieder vom Markt nehmen musste, weil er bei diesen hohen Taktraten reproduzierbare Fehler zeigte.
Wie ja allgemein bekannt ist werden Prozessoren auf Basis von Silizium hergestellt. Dies ist zwar in der Natur in großen Mengen vorhanden, muss aber für die Weiterverarbeitung aufwendig veredelt werden. So werden für die Prozessorherstellung Monokristalle benötigt, da diese ein viel regelmäßigere und feinere Oberfläche haben wie Polykristalle. Diese müssen jedoch erst aufwendig „gezüchtet“ werden.

**Abbildung 3: Monokristallines Silizium**

Danach wird der Siliziumzylinder in Scheiben, die so genannten Wafer, geschnitten. Diese werden dann noch aufwendig geschliffen und poliert. Danach sind sie die Grundlage für die Chipproduktion. Jetzt werden die Wafer dotiert, beschichtet, belichtet und geätzt, bis alle Strukturen auf der Siliziumscheibe vorhanden sind.

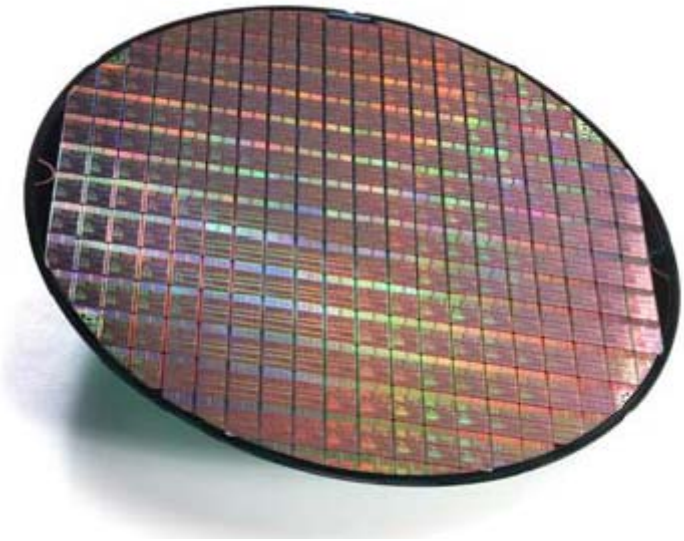

**Abbildung 4: Wafer mit Athlon-Prozessoren**

Dieser Wafer enthält die fertigen Prozessoren, sie müssen jetzt nur noch ausgeschnitten, in Gehäuse gepackt und kontaktiert werden. Abschließend müssen die Prozessoren dann noch auf den Prüfstand um festzustellen, mit welcher Taktung der Prozessor stabil läuft, mit dieser oder einer niedrigeren wird er dann auch verkauft.
Natürlich ist irgendwann die Grenze jeder Architektur erreicht und man kann diese nicht mehr höher Takten. Dann muss an einer neuen Architektur gefeilt werden und andere Technologien verwendet werden. So wurden zum Beispiel im Lauf der Zeit die Strukturen immer feiner, zurzeit sind **130 nm** erreicht, der neue **Pentium V** soll mit **90 nm** gefertigt werden. Ein weiterer Ansatz ist die Verwendung von neuen Materialien. So werden alle heute üblichen Prozessoren in Kupfertechnik, nicht mehr in Aluminiumtechnik, gefertigt. Kupfer hat gegen Aluminium den Vorteil, dass es einen geringeren spezifischen Widerstand hat. Früher konnte man es aber nicht verwenden, weil es das Silizium stark verunreinigt. Erst als man Technologien entwickelte um das Kupfer vom Silizium zu isolieren, konnte man in Kupfertechnik fertigen.

**Abbildung 5: Mikroskopaufnahme Aluminiumtechnik**

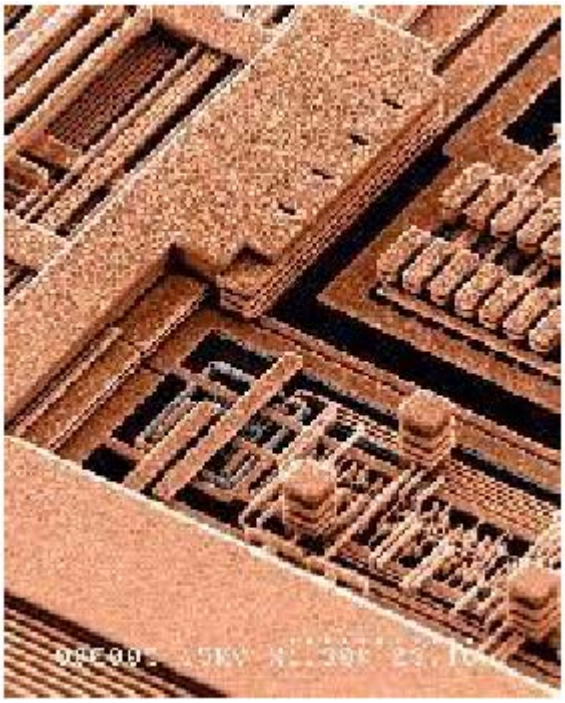

**Abbildung 6: Mikroskopaufnahme Kupfertechnik**

Obwohl man auf den Aufnahmen mehrere Schichten erkennen kann, bei modernen Prozessoren 6 Stück, liegen die Transistoren jedoch nur in der Untersten Schicht. Schicht 2 – 6 sind nur Leiterbahnen, die die einzelnen Halbleiter miteinander verbinden.

Eine weitere Technologie, die zur weiteren Verkleinerung der Chips beitragen könnte wäre SOI (Silicon On Insulator). Mithilfe dieser Technologie würden Kapazitäten verringert, wodurch man Strukturen verkleinern und Taktraten erhöhen könnte.

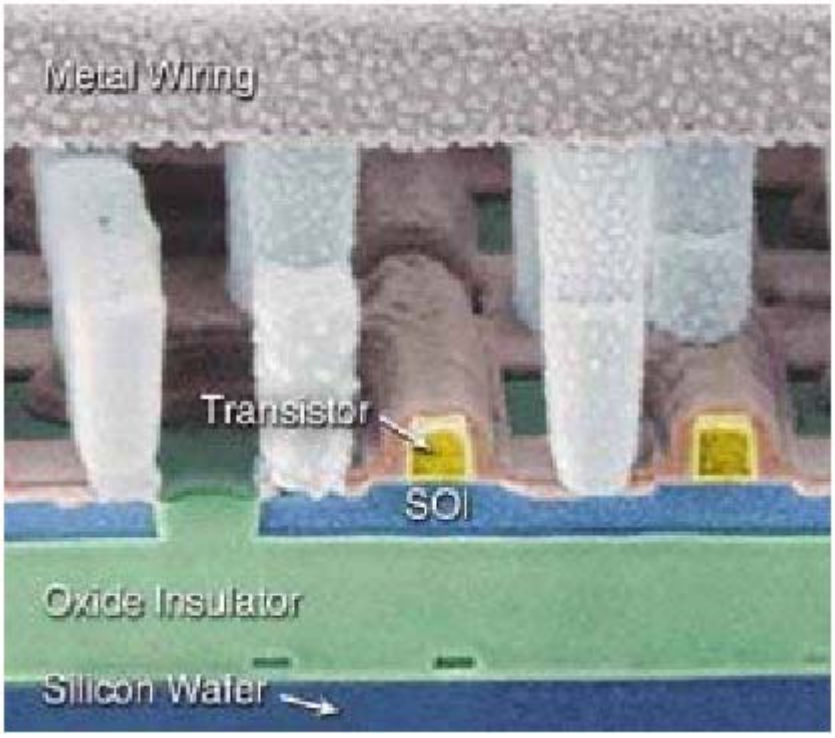


**Abbildung 7: Prozessor in SOI-Technik**

Diese Technik ist aber noch nicht serienreif und sei hier nur der Vollständigkeit wegen erwähnt.

# 3. Architektur von x86 Prozessoren

## 3.1 Das Von-Neumann-Prinzip

Auf diesem Prinzip beruht die Architektur nahezu aller modernen Rechner.

Ein Rechner besteht aus 3 Grundbestandteilen, der CPU (Central Processing Unit), dem Speicher (Memory) und der Ein-/Ausgabeeinheit (I/O).
Die Struktur des Rechners ist unabhängig vom zu bearbeitenden Problem. Die Anpassung an die Aufgabenstellung erfolgt durch Speicherung eines eigenständigen Programms für jedes neue Problem im Speicher des Rechners. Dieses Programm enthält die notwendigen Informationen für die Steuerung des Rechners. Dieses Grundkonzept der Anpassung hat zu der Bezeichnung „Programmgesteuerter Universalrechner“ (engl. „stored-program machine”) geführt.
Der Speicher besteht aus Plätzen fester Wortlänge, die einzeln mit Hilfe einer festen Adresse angesprochen werden können. Innerhalb des Speichers befinden sich sowohl Programmanteile als auch Daten, zwischen denen - als Speicherinhalt - grundsätzlich nicht unterschieden wird.

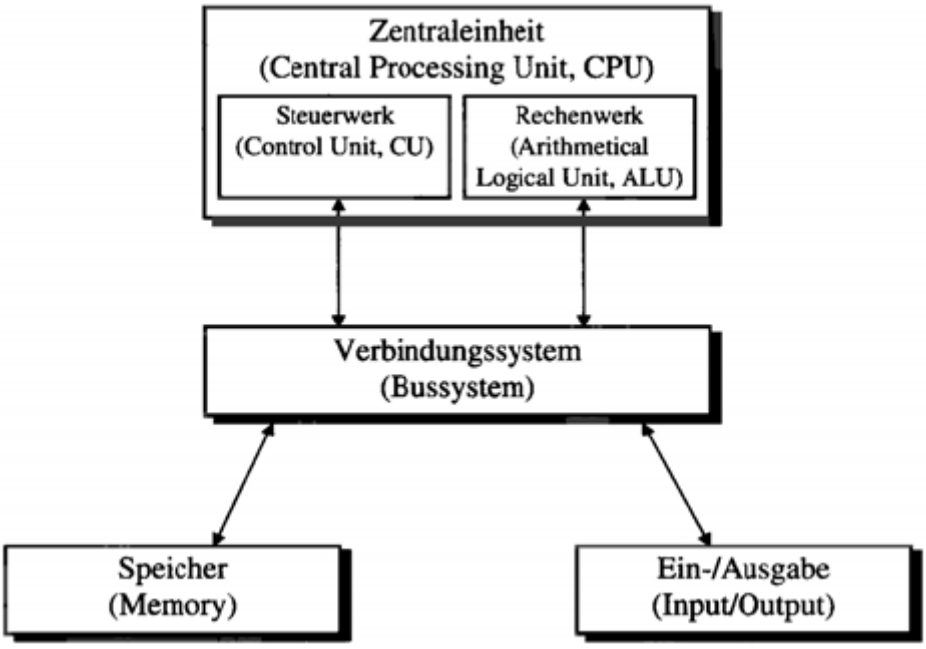


**Abbildung 8: Von-Neumann-Prinzip**

Aus den genannten Grundprinzipien können die wesentlichen Charakteristika des Von-Neumann-Rechners abgeleitet werden:
1. Zu jedem Zeitpunkt führt die CPU exakt einen Befehls aus. Die Steuerung der Bearbeitung liegt im Steuerwerk (Control Unit, CU), das in der Lage sein muss, alle notwendigen Schritte zur vollständigen Behandlung einleiten zu können. Innerhalb eines Befehls kann höchstens ein Datenwert bearbeitet,

d.h. neu berechnet werden. Dieses Prinzip wird Single Instruction - Single Data (SISD) genannt.
2. Alle Inhalte von Speicherzellen, im Folgenden Speicherwörter genannt, sind prinzipiell als Daten oder Befehle interpretierbar. Die Daten wiederum können als „eigentliche“ Daten oder als Referenzen auf andere Speicherzellen (Adressen) genutzt werden. Die jeweilige Verwendung eines Speicherinhalts richtet sich allein nach dem momentanen Kontext des laufenden Programms.
3. Als Konsequenz aus der vorgenannten Eigenschaft können Daten und Befehle nicht gegen ungerechtfertigten Zugriff geschützt werden, da sie gemeinsam ohne Unterscheidungsmöglichkeit im Speicher untergebracht sind.
Ein Von-Neumann-Rechner zeigt unter diesen Voraussetzungen einen typischen Verlauf einer Befehlsbearbeitung. Exemplarisch ist der Ablauf, insbesondere der Signale auf dem Bussystem, für einen Transferbefehl zwischen Register und Speicher dargestellt.

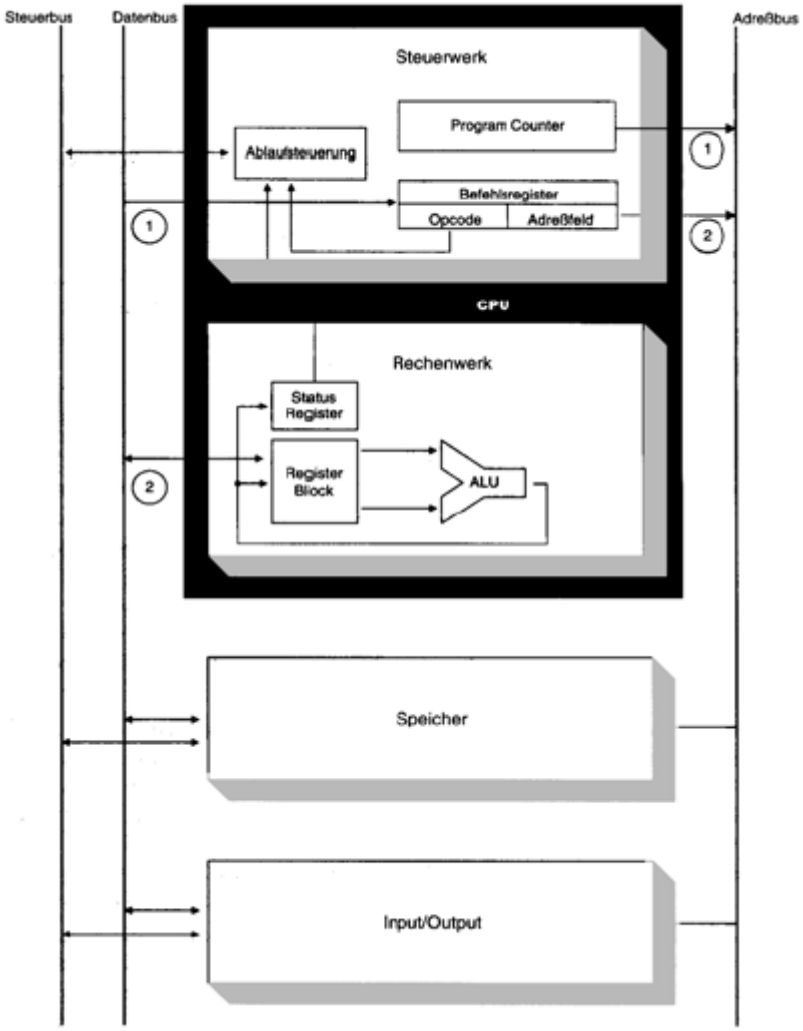


**Abbildung 9: Transferbefehl**

In der Phase 1 lädt die CPU, gesteuert durch Abläufe in dem Steuerwerk CU, das Befehlswort aus dem Speicher einschließlich aller benötigten Zusatzinformationen wie Operanden, in diesem Fall aus der Speicheradresse

bestehend. Diese Phase zeigt sich am Bussystem, indem zwei lesende Zugriffe - je einer für das Befehlswort und den Operanden - sequentiell ablaufen. Das Steuerwerk speichert die Inhalte in dafür vorgesehene Befehlsregister und interpretiert sie für die weiteren Aktionen.
In der zweiten Phase wird nun der Inhalt des Registers in die adressierte Speicherzelle kopiert. Hierzu wird der Inhalt aus dem Registerblock des Rechenwerks auf den Datenbus kopiert, während der Adressbus die Information über die gewünschte Speicherzelle trägt. Adress- und Datenbus sind Teilsysteme der Verbindungseinrichtung im Von-Neumann-Rechner. Das hier exemplarisch gezeigte Verfahren zweier Phasen in der Befehlsbearbeitung ist die technische Lösung für das grundsätzliche Problem des Von-Neumann- Rechners, nur eine Verbindungseinrichtung zu zwei verschiedenen Speicherinhalten - Programm und Daten - zu besitzen. Im Allgemeinen gilt hier folgendes Konzept des zeitlichen Multiplex des Bussystems:
1. In der sogenannten Fetch- und Interpretationsphase wird aufgrund der durch den Befehlszähler angezeigten Adresse der Inhalt einer Speicherzelle geladen und als Befehl interpretiert. Zu dieser Phase zählt im Allgemeinen auch das Laden von Operanden, zumeist von Adressen oder unmittelbaren Daten. Dieser Vorgang wird durch die Interpretation des Befehls gesteuert.
2. In der darauf folgenden Ausführungsphase wird der Befehl nunmehr vollständig interpretiert und ausgeführt. Die Ausführung kann verschiedene Teile der CPU und des Bussystems sowie der angeschlossenen Einheiten in Anspruch nehmen, dies wird durch das Steuerwerk entsprechend gesteuert.
Gemeinsam ist diesen Vorgängen, dass alle Speicherzelleninhalte, auch aus dem Ein-/Ausgabesystem, als Daten interpretiert werden. Dieser zweistufige Ablauf muss für einen Befehl streng sequentiell ablaufen, da eine Abhängigkeit zwischen den verschiedenen Phasen existiert. Spätere Variationen der Von-Neumann-CPU, z.B. die RISC-Architekturen, beinhalten ein Phasenpipelining, das eine scheinbare Parallelität der Aktionen zueinander bewirkt. Dies stellt keinesfalls einen Widerspruch zu dem bisher Gesagten dar, denn die Parallelität im Phasenpipelining bezieht sich auf verschiedene Befehle, während die Bearbeitung eines Befehls weiterhin streng sequentiell bleibt. Der Zeitmultiplex der Busnutzung ist notwendig geworden, da - wie aus dem Beispiel bereits ersichtlich wurde - das Bussystem für den Zugriff auf mehrere Arten von Speicherzelleninhalten genutzt wird. Dies wiederum hat seine Ursache darin, dass der gleiche Speicher fast immer im Mittelpunkt der Operationen steht, unabhängig davon, ob es Programm- oder Dateninhalte sind, auf die zugegriffen werden soll. Die CPU-Speicherkommunikation wird daher die Leistungsfähigkeit des Gesamtsystems entscheidend beeinflussen, was auch als Von-Neumann-Flaschenhals (Von Neuman bottleneck) bezeichnet wird. Das von Neumann-Rechnermodell kann als optimal im Sinn von minimal bezeichnet werden. Die in- und externen Ressourcen, die hierbei zur Anwendung kommen, sind nicht weiter minimierbar, ohne eine sehr wesentliche Einschränkung der Funktionalität bis hin zur Sinnlosigkeit des Einsatzes hinzunehmen.[3]

[3] Christian Siemers: Prozessorbau. Eine konstruktive Einführung in das Hardware/Software – Interface Seite 33ff

### *3.2 Design moderner x86 Prozessoren*

Grundsätzlich ist bei Prozessoren eine Einteilung nach ihrem Befehlssatz möglich. Hier gibt es 2 wichtige Typen: **RISC und CISC**.
**RISC** ist die Abkürzung für Reduced Instruction Set Computing. Dieser Begriff meint nur, dass es sich bei den Befehlen um eine reduzierte Anzahl handelt. Diese Befehle können dann schneller abgearbeitet werden; der Nachteil ist aber, dass manche komplizierte Aufgaben nur mit einer Aneinanderreihung von Befehlen abgearbeitet werden kann. Hier kommt die CISC Architektur zum Zug:
Die **CISC** CPU(analog zu RISC: Complex Instruction Set Computing) besitzt wesentlich mehr Befehle als ihre Konkurrenten, und kann dadurch komplexe Befehle schneller ausführen. Doch hier kommt eine oft zitierte Regel zur Anwendung: rund 80 % der durch die Programme aufgerufenen Befehle benötigt lediglich rund 20 % des Befehlsumfangs.
Prozessoren der x86-Familie sind eigentlich CISC-CPUs, seit dem P III bzw. dem Athlon stimmt dies jedoch nicht mehr ganz. Sie sind sozusagen Mischlinge aus RISC und CISC Architektur. Extern, also nach außen hin und damit für den Rest der Peripherie und dem Programm, das der Prozessor ausführt, verhält er sich wie ein CISC Prozessor. Intern bearbeitet er die Befehle mit einem RISC Befehlssatz.
Dieses System hat einen bestechenden Vorteil: die Software, die für ältere Prozessoren geschrieben wurde kann weiter verwendet werden und muss nicht komplett neu kompiliert werden; wenn man dann ältere Software noch nützen will, muss die CISC Umgebung aufwendig emuliert werden.

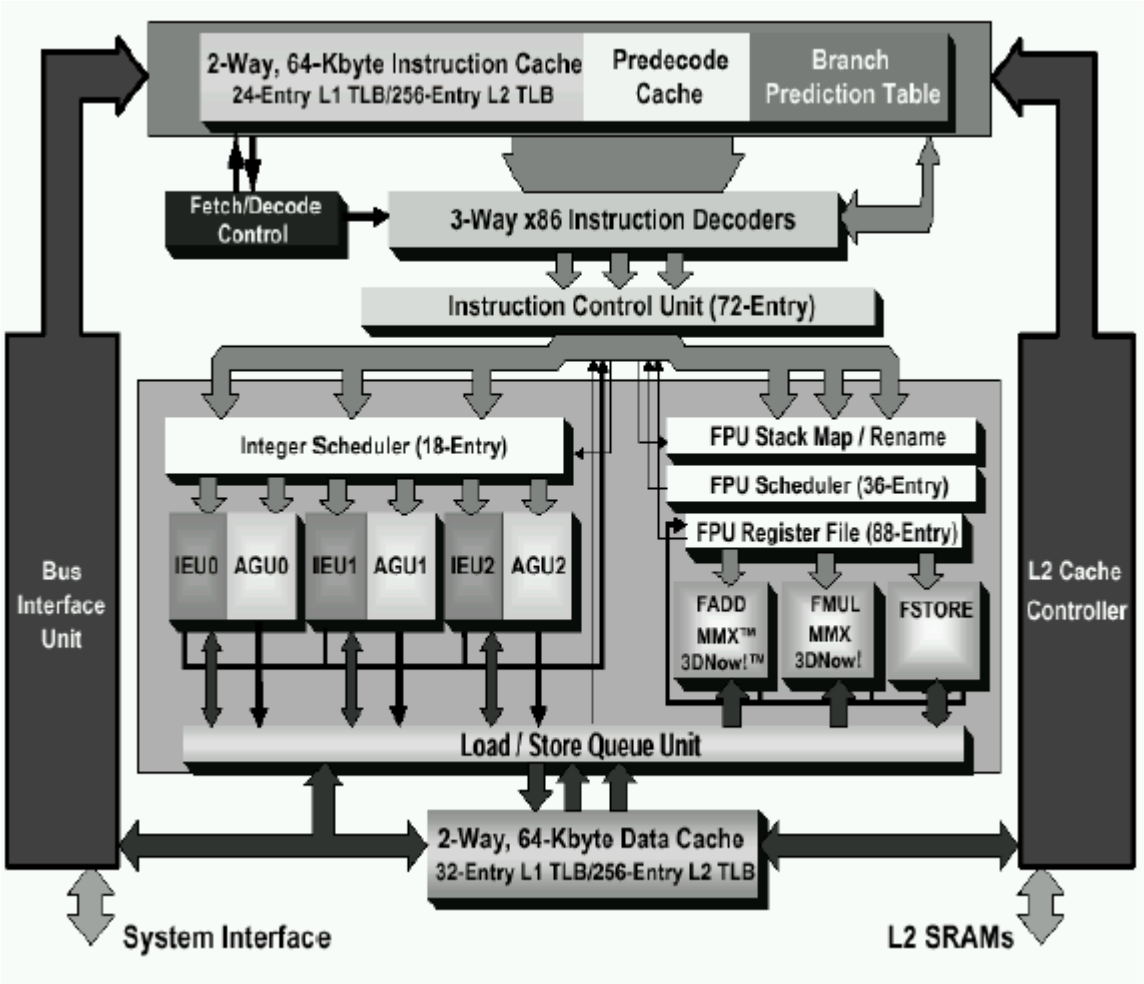


**Abbildung 10: Blockschaltbild Athlon Modell 2**

Was in diesem Blockschaltbild zu erkennen ist, sind die wichtigen Verarbeitungseinheiten, von denen es im AMD Athlon 9 Stück gibt. Auf dem Schaltbild sind sie bezeichnet mit: IEU 0, IEU 1, IEU 2, AGU 0, AGU 1, AGU 2, FStore, Fadd/MMX/3Dnow! und Fmul/MMX/3Dnow!.
Diese neun Einheiten sind für die Verarbeitung der Befehle mit den dementsprechenden Daten zuständig. Darüber auf der Abbildung befinden sich die Einheiten, die die Befehle an die Ausführungseinheiten verteilen: ICU (Instruction Control Unit), Integer Scheduler und die FPU Stack/Scheduler/Register Einheiten. Noch mal darüber sind die Decoder zu sehen, die den x86 Code in RISC wandeln.
Moderne Prozessoren besitzen eine Superscalare Architektur und einen Pipeline Betrieb. Superscalar bedeutet, dass der Prozessor viele verschiedene Ausführungseinheiten besitzt und deshalb mehrere Operationen parallel verarbeiten kann. Eine Pipeline kann man sich wie ein Fließband in einer Fabrik vorstellen. Die Abarbeitung eines Befehls wird in viele kleine Unterschritte zerlegt. Jeder der Arbeiter bekommt eine kleine und einfache Aufgabe, die er schnell erfüllen kann, dann rollt das Band weiter. Genauso im Prozessor: Die Abarbeitung eines Befehl wird in viele kleine Unterschritte zerlegt, um diese dann schneller abarbeiten zu können. Weiterhin analog: In der Fabrik ist es möglich, eine einzelne Aufgabe (zum Beispiel die Produktion eines PKW) in unterschiedlich viele Abschnitte zu gliedern. Je kleiner die

Abschnitte, desto schneller kann das Förderband fließen. Auch beim Prozessor ist es so: Je mehr Zwischenschritte desto mehr Takt ist möglich. Es gibt nur einen Haken. Was passiert wenn einem der Mitarbeiter ein Fehler unterläuft? Die ganze Produktion muss komplett abgebrochen und neu gestartet werden. Genauso passiert es im Prozessor. Wenn ein Befehl abgebrochen werden muss, da er aufgrund einer falschen „**Vermutung**“ schon bevor der Befehl kam bearbeitet wurde, muss die gesamte Operation abgebrochen und das Ergebnis verworfen werden.
Man hat einmal theoretische Überlegungen darüber angestellt, wie lang die optimale Pipeline für einen Prozessor ist. Experten haben ein theoretisches Optimum von etwa acht bis neun einzelner Stufen für klassische Integer Berechnungen herausgefunden. Geringere Piplinelängen würden wie gesagt nur in einem Maße Parallelität der Befehlsabarbeitungen erlauben, wobei die zu fein abgestufte Pipeline viel zu lange Wartezeiten bei einer falschen Vorhersage eines Befehls bedingt.