UNIVERSITÄT LEIPZIG

Enterprise Computing
Einführung in das Betriebssystem z/OS

Prof. Dr. Martin Bogdan
Prof. Dr.-Ing. Wilhelm G. Spruth

WS2012/13

Einführung Teil 3

Mainframe Architektur

Was ist eine Rechner Architektur ?

Als Rechner Architektur bezeichnen wir das Erscheinungsbild eines Rechners wie ihn der Assembler (oder Maschinensprachen-) Programmierer sieht.

Heute Prozessoren verwenden nur noch wenige Rechner Architekturen. Die wichtigsten sind:

- **x86 (PCs und Laptops, größtenteils von Intel und AMD hergestellt)**
- **ARM (Mobiltelefone, viele embedded Systems, wachsende Bedeutung)**
- **PowerPC (IBM Unix “System P” Rechner, Sony Playstation, CISCO Internet Router, Microsoft X-Box, Nintendo, embedded Systems in der Automobilindustrie, werden von Freescale und IBM hergestellt)**
- **Sparc (wird in großen Sun/Oracle und Fujitzu Rechnern eingesetzt)**
- **Itanium (wird in großen Hewlett Packard Unix Rechnern eingesetzt)**
- **System z (ausschließlich in Mainframes eingesetzt, heute fast nur von IBM Hergestellt)**

Für die Entwicklung der Mainframes ist eine Betrachtung des historischen Hintergrundes interessant. Am 7. April 1964 kündigte IBM die S/360-Rechnerfamilie (System 360) an. Die Ziffer „360“ bezog sich dabei auf alle Richtungen eines Kompasses, um die universelle Anwendbarkeit, den weitgefächerten Bereich von Performance und Preis der Produkte sowie die Entwicklungsrichtungen des Unternehmens zu demonstrieren. Die Familie bestand ursprünglich aus 6 Zentraleinheiten und etwa 45 weiteren Ein-/Ausgabeeinheiten.

Die Existenz der S/360-Architektur ist den drei genialen Wissenschaftlern *)

Gene Amdahl,
Gerry Blaauw,
Fred Brooks

und der damaligen IBM-Entwicklungs-Organisation unter der Leitung von IBM-Vizepräsident B.O. Evans zu verdanken, siehe http://www.informatik.uni-leipzig.de/cs/Literature/History/index.html .

Die Einführung der S/360-Architektur stellt einen Meilenstein in der Entwicklung des Computers dar. Zum damaligen Zeitpunkt waren die Unterschiede in den Rechner-Architekturen der einzelnen Hersteller sehr viel größer als dies heute der Fall ist. Viele Eigenschaften, die heute als selbstverständlich gelten, entstanden mit der Einführung der S/360-Architektur. Einige von vielen Beispielen sind:

- **Die Entscheidung, dass ein Byte eine Länge von 8 Bit hat (und nicht z.B. 6 oder 7 Bit),**
- **die Tatsache, dass die Einheit der Hauptspeicheradressierung das Byte ist (und nicht ein 24-, 36- oder 48-Bit langes Wort),**
- **die Einführung von Mehrzweckregistern, die gleichzeitig als Adressregister und als Datenregister dienen,**
- **der Verzicht auf die direkte Hauptspeicher-Adressierung,**
- **der Unterschied zwischen Kernel (Supervisor)- und User (Problem)-Status,**
- **die Einführung des S/360-Kanals, der noch heute in der Form der SCSI und Ficon-Interfaces weiterlebt.**

***) Amdahl, G.M., G.A. Blaauw, and F.P. Brooks, Jr. "Architecture of the IBM System/360." IBM Journal of Research and Development, vol. 8, no. 2 (April 1964): 87-101.**

An Architecture to stand the Test of Time

Ein weiterer Meilenstein war die bewusste Entscheidung, die S/360-Architektur für eine sehr lange Lebenszeit auszulegen. Diesem dienten vor allem zwei Maßnahmen:

- **Die Verpflichtung und Garantie, dass Maschinencode auf allen damaligen sowie zukünftigen Rechnermodellen unverändert lauffähig sein würde. Diese Strategie wurde bis zu den heutigen System z Modelle eingehalten.**

- **Die Einrichtung eines Architektur-Boards mit der Aufgabenstellung, die S/360-Architektur nach wissenschaftlichen Grundsätzen weiterzuentwickeln.**

Diese und viele andere Entscheidungen von Amdahl, Blaauw und Brooks erwiesen sich als außerordentlich tragfähig und haben dazu geführt, dass sich die System z Architektur auch heute noch fortschrittlich und zukunftsorientiert darstellt.

Es existieren nur wenige Architektureigenschaften, die man mit dem heutigen Wissensstand anders und/oder besser machen würde. Dies ist besonders bemerkenswert, weil viele später entwickelte Rechnerarchitekturen mit neuartigen Entwicklungen versprochene Verbesserungen nicht liefern konnten, und in der Zwischenzeit wieder von der Bildfläche verschwunden sind. Ein Beispielhierfür ist die Entwicklung der VAX-Architektur durch die Firma Digital Equipment Corporation (DEC).

DEC war in den 80er Jahren der weltweit zweitgrößte Computer Hersteller nach IBM. Im Jahre 1976 entwickelte DEC eine komplett neue Rechner-Architektur, die erheblich besser als die S/370 Architektur sein sollte. Nach größeren Anfangserfolgen musste DEC die VAX Architektur aufgeben, weil sie gegenüber S/370 schlicht nicht wettbewerbsfähig war. Die Firma Digital Equipment Corporation hat sich von diesem Disaster nie wieder erholen können.

1991 löste die Firma DEC ihre VAX-Architektur durch die Alpha-Architektur ab.

In dem Vorwort des Alpha-Architektur-Handbuches *) wurde explizit darauf hingewiesen, dass man die gleichen Entwurfsprinzipien angewendet habe, die von Amdahl, Blaauw und Brooks 1964 für die Einführung von S/360 entwickelt wurden:

> **The Alpha architecture is a RISC architecture that was designed for high performance and longevity. Following Amdahl, Blaauw, and Brooks, we distinguish between architecture and implementation:**
>
> - **Computer architecture is defined as the attributes of a computer seen by a machinelanguage programmer. This definition includes the instruction set, instruction formats, operation codes, addressing modes, and all registers and memory locations that may be directly manipulated by a machine-language programmer.**
>
> - **Implementation is defined as the actual hardware structure, logic design, and datapath organization.**
>
> **This architecture book describes the required behavior of all Alpha implementations, as seen by the machine-language programmer.**

***) Alpha Architecture Reference Manual, Digital Press, Digital Equipment Corporation, 1992**

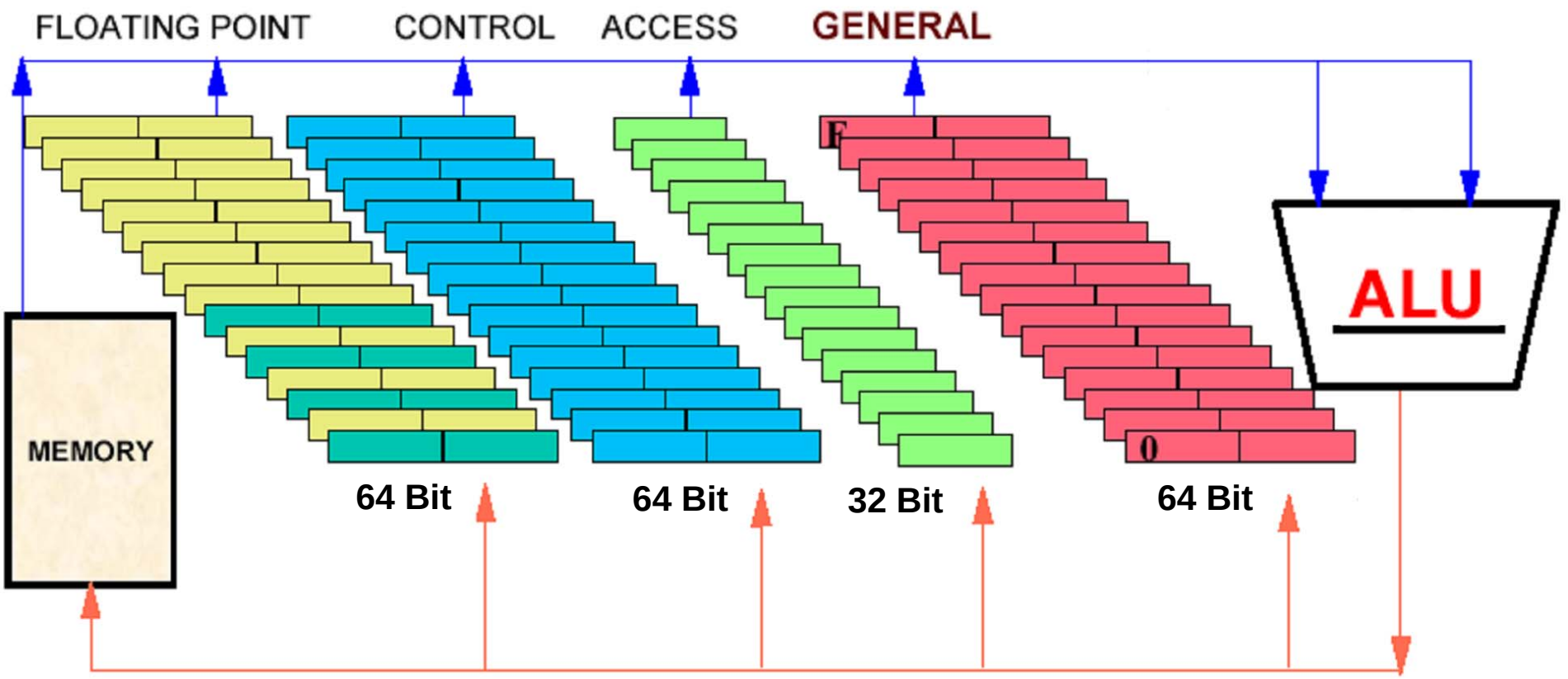
Lebensdauer von Anwendungssoftware

Ein wichtiger Unterschied zwischen Enterprise Computing und anderen Einsatzgebieten der Informatik ist die erwartete Lebensdauer der damit entwickelten Anwendungen. Bei manchen Smartphones und Tablets wird akzeptiert, dass Software für ältere Modelle aufgrund deren begrenzten Langzeitsupports nicht mehr verfügbar ist. Der ideale Kunde eines Smartphone Herstellers ersetzt sein Mobiltelefon alle 24 Monate durch ein neues, verbessertes Modell, obwohl das alte Modell noch voll funktionsfähig ist. Inkompatibilitäten zwischen aufeinander folgenden Software Versionen sind in dieser Situation wegen des schnelllebigen Marktes kein gravierendes Problem.

Im Gegensatz dazu hat Enterprise Software eine Lebensdauer, die in Jahrzehnten gemessen wird. Ein Testfall waren die „Jahr 2000“ (y2k) Umstellungen von zweistelligen auf vierstellige Jahreszahlen. Obwohl der erforderliche Umstellungsaufwand von weltweit 300 Mrd. $ ein guter Anlass gewesen wäre, veraltete Software durch neuere, moderne Versionen zu ersetzen, ist dies fast nirgendwo geschehen.

In der Vergangenheit verstand man unter dem Begriff „Legacy Software“ viele Jahre alte Anwendungen, die auf Mainframe Rechnern ausgeführt wurden. In zunehmendem Maße existieren in den Unternehmen jedoch Legacy Anwendungen, die auf Windows (in unterschiedlichen Versionen), Linux, AIX, HP_UX, Solaris, Tru64 UNIX und vielen weiteren Betriebssystemen laufen.

Es wird geschätzt, dass Unternehmen etwa 50 % ihres jährlichen Budgets für Anwendungssoftware in Neuentwicklungen und 50 % in die Administration und Pflege von Legacy Software investieren.



System z Programmiermodell

Die System z Architektur ist eine 64 Bit Architektur und verfügt über je 16 Mehrzweckregistern, Gleitkomma- und Steuerregistern mit einer Länge von je 64 Bit, und 16 Access Registern mit einer Länge von je 32 Bit. Ähnlich wie x86, PowerPC und Sparc Architekturen ist sie aus einer ursprünglichen 32 Bit Version weiterentwickelt worden. Hierbei unterstützen die System z-Rechner einen 32-Bit- und einen 64-Bit-Modus. Das Umschalten zwischen beiden Modi ist sehr einfach.

Architecture	Integer registers	Double FP registers
x86	8	8
x86-64	16	16
IBM/360	16	4
Z/Architecture	16	16
Itanium	128	128
UltraSPARC	32	32
POWER	32	32
Alpha	32	32
6502	3	0
ARM	16	16

Anzahl der Register mehrerer Mainstream Architekturen

Eine zu große Anzahl von Registern verringert die Performance, weil bei jedem Prozesswechsel der Inhalt der Register abgespeichert (saved) und neue Inhalte geladen werden müssen. Optimal sind durchschnittlich 25 Register, abhängig vom Profile der bearbeiteten Anwendungen. Da 25 keine Binärziffer ist, sind 16 oder 32 Register optimal.

16 und 32 Bit Maschinenbefehle

Die System z Maschinenbefehle haben alle eine einheitliche Länge von entweder 16 Bit, 32 Bit oder 48 Bit. Vor allem der geschickte Entwurf der 16 Bit Maschinenbefehle führt dazu, dass ein kompiliertes Programm auf einem System z Rechner fast immer 20 – 30 % weniger Platz im Hauptspeicher benötigt, als bei fast allen anderen Rechner Architekturen. Dies gilt spezifisch auch bei einem Vergleich der System z Architektur gegenüber der x86 Architektur.

Vor allem auf Grund der reduzierten Cache Bandbreite ist dies ein Performance Vorteil.

Es ist interessant zu sehen, dass die Firma Arm Holdings mit der Einführung der Thumb Technologie Erweiterung für die ARM Architektur ein ähnliches Ziel anstrebt. Vor allem die jüngste Thumb-2 Technologie Erweiterung weist an dieser Stelle überraschende konzeptuelle Ähnlichkeiten mit der 1964 entstandenen S/360 Architektur auf.

Hiermit wiederholt sich die Frage: Wie konnte es sein, dass Amdahl, Blaauw und Brooks beim Entwurf der S/360 Architektur im Jahre 1964 alles richtig gemacht haben ?

Decimal Floating Point (DFP)

System z unterstützt drei Gleitkommaformate: Hexadezimal, IEEE 754 und Dezimal.

Das hexadezimale Format ist ein IBM proprietärer Binär-Standard und fast nur auf Mainframe Rechnern anzutreffen. Die meisten auf Mainframes gespeicherten Gleitkommadaten verwenden das hexadezimale Format.

Das IEEE 754 Format ist international weit verbreitet, und wird u.a. von der x86 Architektur benutzt. Auf Mainframes benutzt vor allem das zLinux Betriebssystem diesen Standard.

Das Dezimale Gleitkommaformat (DFP)ist erst seit wenigen Jahren verfügbar. Im Gegensatz zu den beiden anderen Formaten werden Mantisse und Exponent mit Dezimalziffern dargestellt. Dies führt zu einer gewissen Performance- und Genauigkeitseinbuße.

Die Benutzung von Dezimal Arithmetik ist in der Wirtschaft weit verbreitet. DFP vermeidet Rundungsfehler und andere Probleme bei der Konvertierung von Dezimaldaten in Binärdaten und umgekehrt, und gewinnt deshalb an Bedeutung.

Die Gleitkomma Einheit des System z Mikroprozessors unterstützt alle drei Gleitkomma-Arten. Ebenso speichert die z/OS DB2 Datenbank Daten in allen drei Formaten.

ASCII- und EBCDIC

Es existieren drei weit verbreitete Standards für die Darstellung alphanumerischer (Buchstaben, Ziffern, Sonderzeichen) Daten.

Die Darstellung von alphanumerischen Zeichen geht bei allen Rechnerarchitekturen auf uralte Wurzeln zurück. Bei vielen Rechnern ist dies die 7-Bit-ASCII-Darstellung (American Standard Code for Information Interchange), die ihren Ursprung in den Lochstreifen der Teletype-Maschinen hat und nachträglich auf 8 Bit erweitert wurde. Bei den Mainframe- (und einigen anderen) Rechnern ist dies die 8-Bit-EBCDIC-Darstellung (Extended Binary Coded Decimal Interchange Code, ausgesprochen [ebsidik]), die ihren Ursprung in den Lochkarten hat. Das Ergebnis ist eine Spaltung der IT-Welt. Etwa 60 % aller geschäftlich relevanten alphanumerischen Daten sind im EBCDIC-Format gespeichert und etwa 40 % im ASCII-Format. Wenn immer ASCII - und EBCDIC-Rechner miteinander kommunizieren, sind unschöne Konvertierungsverfahren erforderlich.

Mainframes unterstützen neben dem EBCDIC- auch das ASCII-Format. Letzteres wird z.B. von dem zLinux Betriebssystem genutzt, ist aber sonst nur wenig gebräuchlich. Man kann davon ausgehen, dass alphanumerische Daten auf einem Mainframe in der Regel im EBCDIC-Format vorliegen.

Neben EBCDIC und ASCII können Mainframes auch Daten im UTF-8 bzw. UTF-16 Format verarbeiten. Die Java Plattform verwendet UTF16 in Character Arrays und Strings

ASCII-Tabelle

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	SP	!	"	#	S	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	Ä	Ö	Ü	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	ä				

EBCDIC-Tabelle

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	PF	HT	LC	DEL			SMM	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	TM	RES	NL	BS	IL	CAN	EM	CC	CU1	IFS	IGS	IRS	IUS
2	DS	SOS	FS		BYP	LF	ETB	ESC			SM	CU2		ENQ	ACK	BEL
3			SYN		PN	RS	UC	EOT				CU3	DC4	NAK		SUB
4	SP										¢	.	<	(	+	|
5	&										!	$	*	)	;	¬
6	-	/										,	%	_	>	?
7											:	#	@	'	=	"
8		a	b	c	d	e	f	g	h	i						
9		j	k	l	m	n	o	p	q	r						
A		~	s	t	u	v	w	x	y	z						
B										`						
C		A	B	C	D	E	F	G	H	I						
D		J	K	L	M	N	O	P	Q	R						
E	\		S	T	U	V	W	X	Y	Z						
F	0	1	2	3	4	5	6	7	8	9						

ASCII- und EBCDIC Zeichentabellen

Beispiele: **ASCII** **R = Hex 52 ;**
EBCDIC R = Hex D9 ;

Weltweit sind etwa 60% aller wirtschaftlich relevanten Daten sind im 8 Bit EBCDIC Standard (Extended Binary Coded Decimal Interchange Code) abgespeichert. Etwa 40% aller wirtschaftlich relevanten Daten sind im 7 Bit ASCII Standard (American Standard Code for Information Interchange) bzw. seiner 8 Bit Erweiterung abgespeichert.

Big Endian versus Little Endian

Eine Hauptspeicheradresse adressiert ein einziges Byte im Hauptspeicher. Unter der „Endianess“ eines Rechners versteht man die Reihenfolge, in der eine Gruppe von Bytes im Hauptspeicher abgespeichert werden.

Nehmen wir einen Maschinenbefehl an, der 4 aufeinanderfolgende Bytes in ein Mehrzweckregister der Zentraleinheit ladet. Werden die Bytes in aufsteigender oder in absteigender Reihenfolge geladen ?

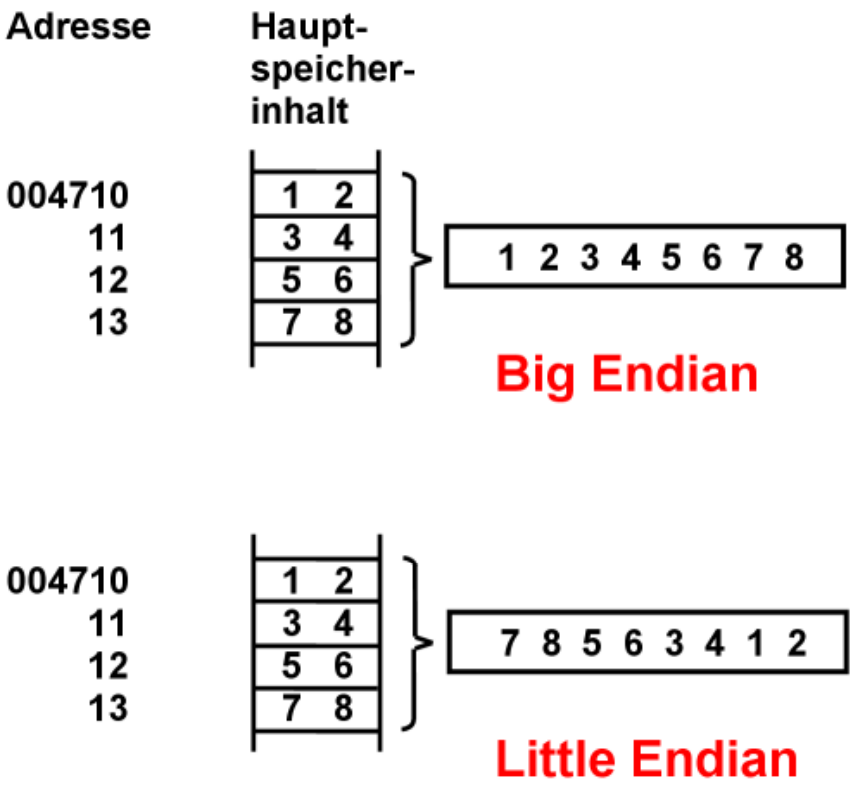
Wenn Halbworte oder Worte im Hauptspeicher gespeichert sind, dann befindet sich an der adressierten Hauptspeicherstelle:

- **Das wertniedrigste Byte bei Little Endian Rechnern**
- **Das werthöchste Byte bei Big Endian Rechnern**

Die Bytes eines Halbwortes oder Wortes werden bei Little Endian Rechnern in umgekehrter Reihenfolge abgespeichert wie bei Big Endian Rechnern.

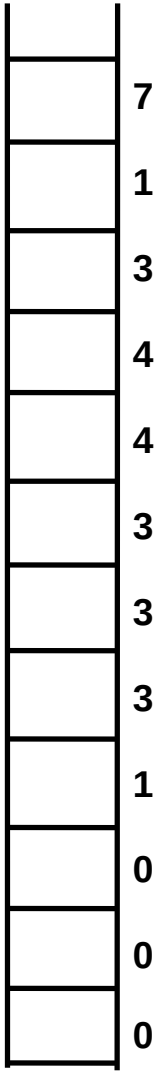
Dies ist in der folgenden Abbildung dargestellt. Es enthält die Hauptspeicheradresse 004710

- **das werthöchste Byte bei der Big Endian Architektur**
- **das wertniedrigste Byte bei der Little Endian Architektur**



Mainframes, die meisten Unix Rechner und das Internet verwenden die Big Endian Architektur.
x86 und Ethernet verwenden die Little Endian Architektur.

Für Konvertierungszwecke enthält die x86 Architektur einen „Byte Swap“ Maschinenbefehl, mit dessen Hilfe die Endianess eines Feldes im Hauptspeicher umgedreht werden kann.

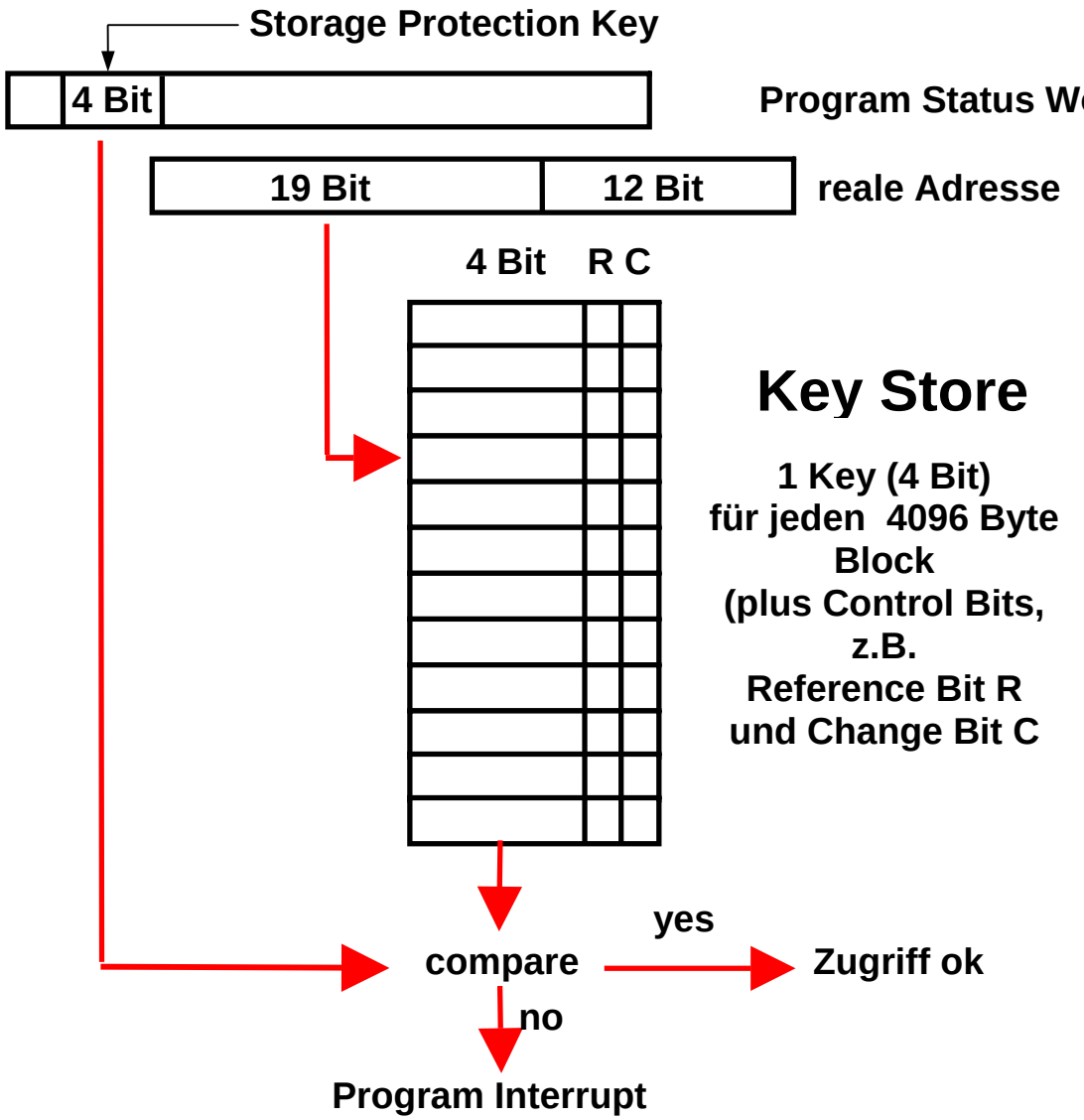


System z Speicherschutz

Hardware-Speicherschutz (Hardware Protection Keys) sind eine einzigartige Einrichtung der Mainframes. Der Hardware-Speicherschutz teilt den Hauptspeicher in 4096 Byte große Blöcke auf und ordnet jedem dieser Blöcke einen 4 Bit Speicherschutzschlüssel zwischen 0 ... 15 zu, der in einem getrennten Schnellspeicher abgespeichert wird. Der Speicherschutzschlüssel wird in einem 4-Bit-Feld des CPU Status Registers (Program Status Word, PSW*)* gespeichert. Bei jedem Speicherzugriff wird aus einem Schnellspeicher dieser Speicherschutzschlüssel ausgelesen und mit dem 4-Bit-Feld im PSW verglichen. Nur wenn dieser Vergleich positiv ausfällt (4-Bit-Felder sind identisch), erfolgt der Zugriff.

Die einzelnen Prozesse haben unterschiedliche Speicherschutzschlüssel. Somit wird verhindert, dass die Programme eines Prozesses auf Speicherbereiche eines anderen Prozesses zugreifen können.

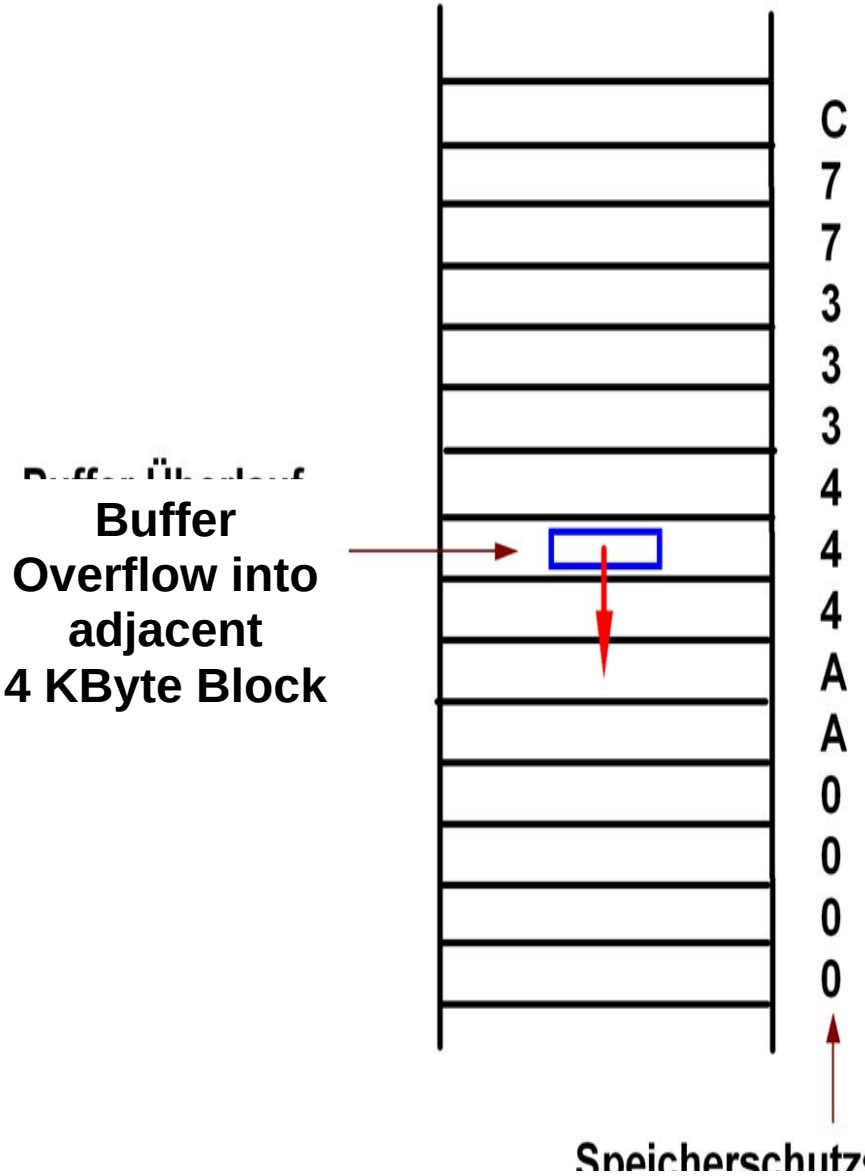
Die hierfür erforderliche Logik ist in der folgenden Abbildung dargestellt.



Storage Protection Keys

Bei jedem Hauptspeicher-zugriff wird der Wert im Key Store für den adressierten 4 KByte Block mit dem Wert im PSW verglichen. Nur bei Übereinstimmung wird der Zugriff durchgeführt.

Literatur: ABCs of z/OS System Programming, Volume 10 http://www.redbooks.ibm.com/redbooks/pdfs/sg246990.pdf
Section 1.24, S. 43



Buffer Overflow Prevention

Der Hauptspeicher ist in 4 KByte große Blöcke mit unterschiedlichen Protection Keysaufgeteilt. Dies verhindert einen bekanntes Sicherheitsproblem, den Buffer-Overflow.

Versucht ein Prozess auf den (realen) Hauptspeicherbereich eines anderen Prozesses zuzugreifen, so verhindert die Hardware Protection Einricht den Zugriff, wenn die Protection Keys ungleich sind.

Vor allem der Zugriff auf Betriebssystem-Kernel Funktionen wird damit ausgeschlossen.

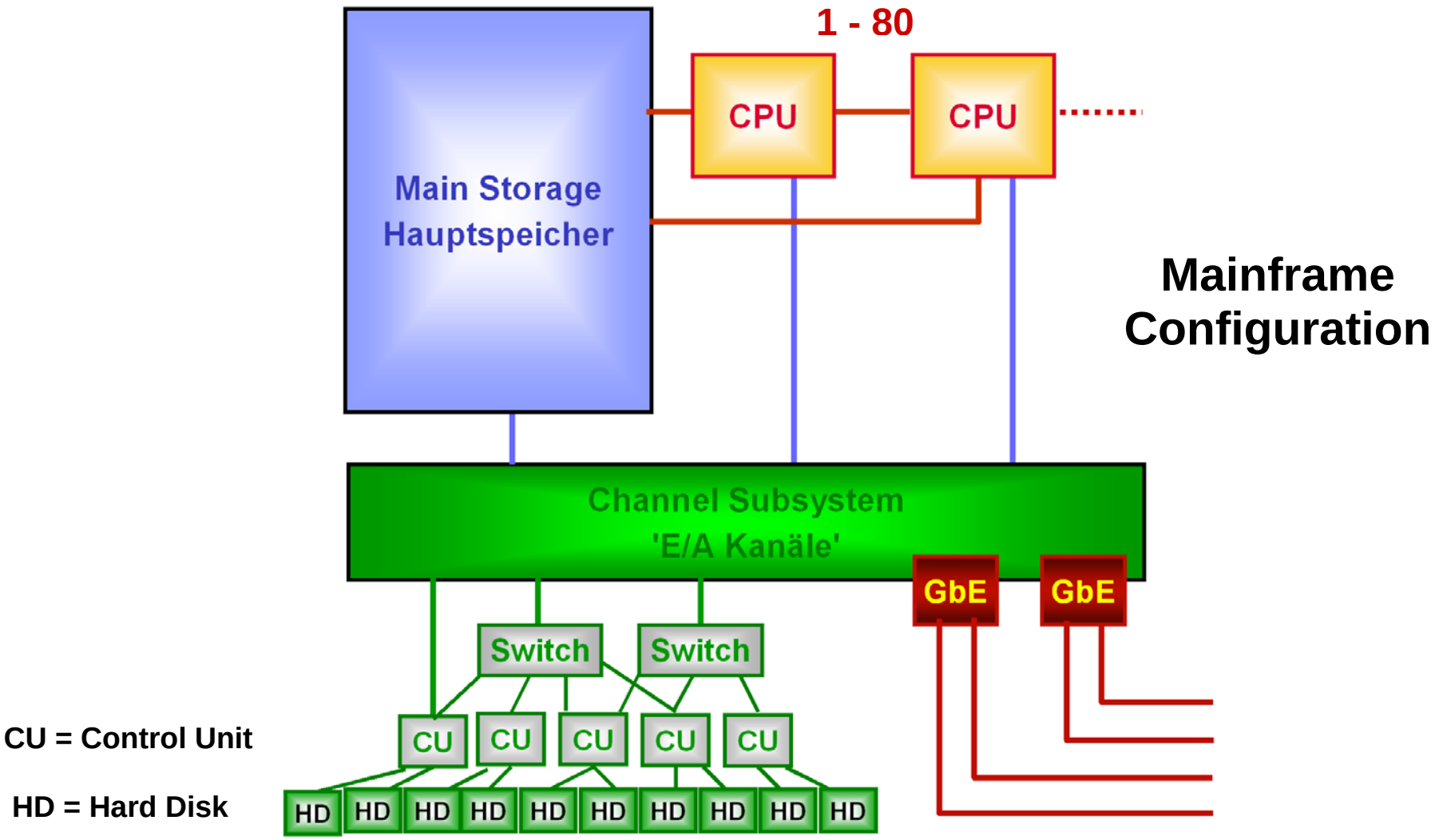
Hardware Management Console

Ein PC verfügt über ein BIOS. Letzteres implementiert Maschinencode, der in einem Teil des Hauptspeichers liegt, auf den ein Benutzerprozess nicht zugreifen kann.

Eine äquivalente Mainframe-Funktion heißt Firmware , häufig auch als LIC (Licensed Internal Code) bezeichnet. Der Firmware-Bereich ist außerhalb des Adressbereichs der Maschinenbefehle im Hauptspeicher untergebracht. Er wird beim Hochfahren eines Rechners durch einen als IML (Initial Microcode Load) bezeichneten Vorgang in den Speicher geladen.

Firmware hat zahlreiche Funktionen. Er implementiert z.B. manche komplexe Maschineninstruktionen oder umfangreiche Diagnostik- und Fehlerbehandlungs-Funktionen. Die später beschriebene PR/SM-Einrichtung wird ebenfalls mittels Firmware implementiert.

Wenn man beim Hochfahren eines PC eine Funktionstaste drückt, kann man BIOS-Funktionen aufrufen. Beim Mainframe ist diese Funktionalität sehr viel umfangreicher, während des laufenden Betriebs verfügbar, und wird als „Operator Facilities“ bezeichnet. Ein System-Administrator kommuniziert mit Hilfe der Operator Facilities mit dem Rechner. Er erledigt damit Aufgaben wie das Setzen von Datum und Zeit, das Reset von Subsystemen, Architectural Mode Selection, das Eingreifen in einen ausführenden Programmablauf oder ein Reagieren auf Maschinenfehler-Unterbrechungen. Ebenfalls dazu gehört die Funktion eines Boot-Managers, beim Mainframe als Initial Program Load (IPL) bezeichnet. Mainframe-Rechner verfügen über ein als „Hardware Management Console“ (HMC) bezeichnetes Bildschirm-Gerät, das ausschließlich der Nutzung der Operator Facilities durch den System-Administrator dient.



Ein Mainframe System besteht zumindest aus einer oder mehreren (bis zu 96) Prozessoren, einem Hauptspeicher, und Anschlüssen für Ein/Ausgabe (E/A) Geräte, besonders Plattenspeicher (Hard Disk) und Netzwerkanschlüsse, häufig Gigabit oder 10 GBit Ethernet (GbE). Plattenspeicher und andere Input/Output Geräte (I/O) werden grundsätzlich über Control Units (CU) angeschlossen.

Control Unit

Bei einem PC erfolgt die Ansteuerung eines Plattenspeichers durch eine als I/O Driver bezeichnete Komponente des Betriebssystems. Die Ausführung des I/O Driver Codes benötigt CPU Zeit.

An viele PCs ist nur ein einziger Plattenspeicher angeschlossen. An einen Mainframe können Tausende, zehntausende oder hunderdtausende von Plattenspeichern angeschlossen sein. Zur Entlastung der CPU(s) wird die I/O Driver Abarbeitung auf dezentrale Spezialrechner ausgelagert, die als Control Units bezeichnet werden. Ohne diese Auslagerung wäre ein Mainframe Rechner niemals in der Lage, die erforderliche hohe Ein/Ausgabeleistung, besonders Plattenspeicherdurchsatz, zu erbringen.

Zum Anschluss der Control Units und Plattenspeicher wird ein Netzwerk von Verbindungen benötigt, die als Kanäle (Channels) bezeichnet werden. Kanäle werden den einzelnen I/O Anforderungen dynamisch zugeordnet, was ebenfalls sehr verarbeitungsaufwendig ist. Um hiervon die CPU(s) zu entlasten, erfolgt die Ansteuerung der Kanäle durch eine getrennte Komponente, das „Channel Subsystem“.

Wir werden uns die Ein/Ausgabe Ansteuerung später noch genauer ansehen.

Magnetbandspeicher

Fast alle Mainframe Installetionen verwenden Magnetbänder um weniger häufig gebrauchte Daten zu speichern, und/oder um Daten zu archivieren.

Auf Magnetbänder (Tapes) wird mit Hilfe von Robotern zugegriffen. Diese verfügen über eigene Control Units, die von z/OS angesteuert werden. Ein Magnetband-Roboter ist in der Lage, eine Magnetbandkassette aus einem Regal zu entnehmen und in eine Magnetband Lese Schreibstation einzulegen, um einen automatischen Zugriff zu ermöglichen. Der Umfang dieser Magnetbanddaten übertrifft den Umfang der Plattenspeicherdaten typischerweise um einen Faktor 10.

Eine typischerweise nochmals um einen Faktor 10 – 100 größere Datenmenge ist zu Archivierungszwecken ausgelagert.

Auch Magnetbandspeicher oder Magnetbandroboter werden über Control Units an den Mainframe Rechner angeschlossen.

Unterschiedliche Begriffe

Die Mainframe Welt benutzt häufig andere Begriffe als die PC oder Linux Welt. Hier ist ein kleiner Auszug aus meinem Wörterbuch:

z/OS, OS/390	Windows/Unix
Problem State	User Mode
Supervisor State	Kernel Mode
Region	Virtueller Adressenraum
Data Set	File
DASD	Plattenspeicher
Program Status Word	Status Register

DASD = Direct Access Storage Device