ERA-Tutorium 4

Thomas Kilian

Organisatorisches

- Gibt es Fragen zum letzten Tutorium?

- Wünsche und Anregungen?

Fragen zur Vorlesung

Fragen zur Vorlesung

Größe von Nibble, Byte, Word, Doubleword, Quadword?

Fragen zur Vorlesung

Größe von Nibble, Byte, Word, Doubleword, Quadword?

➡ Nibble: 4, Byte: 8, Word: 16, Doubleword: 32, Quadword: 64

Fragen zur Vorlesung

Größe von Nibble, Byte, Word, Doubleword, Quadword?

- ➡ Nibble: 4, Byte: 8, Word: 16, Doubleword: 32, Quadword: 64

Wie viel Speicher kann der x86 adressieren?

Fragen zur Vorlesung

Größe von Nibble, Byte, Word, Doubleword, Quadword?

➡ Nibble: 4, Byte: 8, Word: 16, Doubleword: 32, Quadword: 64

Wie viel Speicher kann der x86 adressieren?

➡ 2^{32} Byte

Fragen zur Vorlesung

Fragen zur Vorlesung

Sind folgende Assembler-Befehle legal?

Fragen zur Vorlesung

Sind folgende Assembler-Befehle legal?

- MOV EAX, [EBX + ECX * 5]
- MOV AL, [5 + ECX]
- MOV [ESI], EDX + 5

Fragen zur Vorlesung

Sind folgende Assembler-Befehle legal?

- ~~MOV EAX, [EBX + ECX * 5]~~
- MOV AL, [5 + ECX]
- MOV [ESI], EDX + 5

Fragen zur Vorlesung

Sind folgende Assembler-Befehle legal?

- ~~MOV EAX, [EBX + ECX * 5]~~
- MOV AL, [5 + ECX]
- ~~MOV [ESI], EDX + 5~~

Fragen zur Vorlesung

Fragen zur Vorlesung

Wodurch unterscheidet sich eine RISC LOAD/ STORE CPU- Architektur von der CISC-Architektur x86?

Fragen zur Vorlesung

Wodurch unterscheidet sich eine RISC LOAD/ STORE CPU- Architektur von der CISC-Architektur x86?

- Bei LOAD/STORE-Architektur dürfen nur diese beiden Befehle auf den Speicher zugreifen

Fragen zur Vorlesung

Wodurch unterscheidet sich eine RISC LOAD/ STORE CPU- Architektur von der CISC-Architektur x86?

- Bei LOAD/STORE-Architektur dürfen nur diese beiden Befehle auf den Speicher zugreifen

Nach welchem Prinzip arbeitet der Stack im x86?

Fragen zur Vorlesung

Wodurch unterscheidet sich eine RISC LOAD/ STORE CPU- Architektur von der CISC-Architektur x86?

- Bei LOAD/STORE-Architektur dürfen nur diese beiden Befehle auf den Speicher zugreifen

Nach welchem Prinzip arbeitet der Stack im x86?

- LIFO -> Last In First Out

Stack

Stack

- entwickelt von F. L. Bauer und K. Samelson

Stack

- entwickelt von F. L. Bauer und K. Samelson
- selbstverwaltende Speicherverwaltung

Stack

- entwickelt von F. L. Bauer und K. Samelson
- selbstverwaltende Speicherverwaltung
- LIFO-Struktur

Stack

- entwickelt von F. L. Bauer und K. Samelson
- selbstverwaltende Speicherverwaltung
- LIFO-Struktur
- zwei Befehle

Stack

- entwickelt von F. L. Bauer und K. Samelson
- selbstverwaltende Speicherverwaltung
- LIFO-Struktur
- zwei Befehle
  - PUSH: Legt Element auf Stack

Stack

- entwickelt von F. L. Bauer und K. Samelson
- selbstverwaltende Speicherverwaltung
- LIFO-Struktur
- zwei Befehle
  - PUSH: Legt Element auf Stack
  - POP: Holt Element vom Stack

Stack

Stack

- zwei Stackpointer

Stack

- zwei Stackpointer
  - **EBP** (Frame oder Base Pointer) und **ESP** (Stack Pointer)

Stack

- zwei Stackpointer
  - **EBP** (Frame oder Base Pointer) und **ESP** (Stack Pointer)
- Nachbilden von ***PUSH*** / ***POP***

Stack

- zwei Stackpointer
  - **EBP** (Frame oder Base Pointer) und **ESP** (Stack Pointer)
- Nachbilden von ***PUSH*** / ***POP***
- ***JUMP*** vs. ***CALL***

Stack

- zwei Stackpointer
  - **EBP** (Frame oder Base Pointer) und **ESP** (Stack Pointer)
- Nachbilden von ***PUSH*** / ***POP***
- ***JUMP*** vs. ***CALL***
- ***CALL*** immer mit ***RET***

Aufgabe 1

Aufgabe 1

An welche Adresse springt *ret*? Werte in Register und Stack?

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1					

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1				

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?			

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?		

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2					

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1				

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2			

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?		

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3					

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1				

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2			

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3		

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4					

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1				

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2			

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3		

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX					

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1				

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2			

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3		

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX					

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1				

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2			

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3		

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX					

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2				

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2			

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3		

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX					

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2				

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2			

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3		

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX					

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2				

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2			

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3		

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX					

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2				

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2			

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3		

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX					

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2				

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2			

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4		

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4	4	

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4	4	2, 3, 1, …
POP EDX					

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4	4	2, 3, 1, …
POP EDX	2				

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4	4	2, 3, 1, …
POP EDX	2	2			

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4	4	2, 3, 1, …
POP EDX	2	2	4		

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4	4	2, 3, 1, …
POP EDX	2	2	4	2	

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4	4	2, 3, 1, …
POP EDX	2	2	4	2	3, 1, …
POP EBX					

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4	4	2, 3, 1, …
POP EDX	2	2	4	2	3, 1, …
POP EBX	2				

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4	4	2, 3, 1, …
POP EDX	2	2	4	2	3, 1, …
POP EBX	2	3			

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4	4	2, 3, 1, …
POP EDX	2	2	4	2	3, 1, …
POP EBX	2	3	4		

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4	4	2, 3, 1, …
POP EDX	2	2	4	2	3, 1, …
POP EBX	2	3	4	2	

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4	4	2, 3, 1, …
POP EDX	2	2	4	2	3, 1, …
POP EBX	2	3	4	2	1, …
RET					

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4	4	2, 3, 1, …
POP EDX	2	2	4	2	3, 1, …
POP EBX	2	3	4	2	1, …
RET	2				

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4	4	2, 3, 1, …
POP EDX	2	2	4	2	3, 1, …
POP EBX	2	3	4	2	1, …
RET	2	3			

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4	4	2, 3, 1, …
POP EDX	2	2	4	2	3, 1, …
POP EBX	2	3	4	2	1, …
RET	2	3	4		

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4	4	2, 3, 1, …
POP EDX	2	2	4	2	3, 1, …
POP EBX	2	3	4	2	1, …
RET	2	3	4	2	

;	EAX	EBX	ECX	EDX	Stack
MOV EAX,1	1	?	?	?	…
MOV EBX, 2	1	2	?	?	…
MOV ECX, 3	1	2	3	?	…
MOV EDX, 4	1	2	3	4	…
PUSH EAX	1	2	3	4	1, …
PUSH EBX	1	2	3	4	2, 1, …
POP EAX	2	2	3	4	1, …
PUSH ECX	2	2	3	4	3, 1, …
PUSH EAX	2	2	3	4	2, 3, 1, …
PUSH EDX	2	2	3	4	4, 2, 3, 1, …
POP ECX	2	2	4	4	2, 3, 1, …
POP EDX	2	2	4	2	3, 1, …
POP EBX	2	3	4	2	1, …
RET	2	3	4	2	…

Aufgabe 2

Aufgabe 2

Welche Möglichkeiten gibt es für die Parameter-übergabe an Funktionen?

Aufgabe 2

Welche Möglichkeiten gibt es für die Parameter-übergabe an Funktionen?

- Übergabe über Register

Aufgabe 2

Welche Möglichkeiten gibt es für die Parameter-übergabe an Funktionen?

- Übergabe über Register
- Übergabe über feste Speicherstellen

Aufgabe 2

Welche Möglichkeiten gibt es für die Parameter-übergabe an Funktionen?

- Übergabe über Register
- Übergabe über feste Speicherstellen
- Übergabe über Stack

Aufgabe 2: Variante 1

```
; Aufruf
MOV EAX, 1
MOV EBX, 2
MOV ECX, 3
MOV EDX, 4
CALL fn_reg
```

```
fn_reg:
ADD EAX, EBX
ADD EAX, ECX
ADD EAX, EDX
RET
```

Aufgabe 2: Variante 2

```
; Speicher alloziieren
fn_p1: RESD 1
fn_p2: RESD 1
fn_p3: RESD 1
fn_p4: RESD 1
```

```
; Aufruf
MOV [fn_p1], 1
MOV [fn_p2], 2
MOV [fn_p3], 3
MOV [fn_p4], 4
CALL fn_static
```

```
fn_static:
MOV EAX, [fn_p1]
ADD EAX, [fn_p2]
ADD EAX, [fn_p3]
ADD EAX, [fn_p4]
RET
```

Aufgabe 2: Variante 3

```
; Aufruf
PUSH dword 4
PUSH dword 3
PUSH dword 2
PUSH dword 1
CALL fn_stack
ADD ESP, 16
```

```
fn_stack:
MOV EAX, [ESP + 4]
ADD EAX, [ESP + 8]
ADD EAX, [ESP + 12]
ADD EAX, [ESP + 16]
RET
```

Aufgabe 2

Vor- und Nachteile dieser Varianten?

Register	Speicher	Stack
+ schnellste Möglichkeit	+ keine Stacknutzung	+ beliebige (auch variable) Anzahl an Parametern
- Anzahl Parameter begrenzt	+ einfache Adressierung	+ beliebige Verschachtelung und Rekursion
- Parameter “flüchtig”, keine Rekursion	+ einfaches Debugging	- umständlichere Adressierung
	+ Parameter immer auslesbar	o EBP als fester Anker zur relativen Adressierung im Stack
	- keine Rekursion möglich	
	- keine variable Anzahl an Parametern	
	- Speicher immer reserviert, auch wenn Funktion nicht aufgerufen wird	

Aufgabe 3

Mit welcher Formel kann man allgemein die Speicheradresse eines bestimmten RGB- Bildpunkts (x, y) aus den Werten Bildschirmspeicherstart, Breite und Höhe berechnen?

Aufgabe 3

Grafikkarten mit 8 Bit pro Farbintensität (Rot/Grün/Blau, d.h. “true color”) benötigen mindestens 24 Bit pro Pixel. Diese werden inzwischen fast immer auf 32 Bit aufgerundet.

Mit welcher Formel kann man allgemein die Speicheradresse eines bestimmten RGB- Bildpunkts (x, y) aus den Werten Bildschirmspeicherstart, Breite und Höhe berechnen?

Aufgabe 3

Aufgabe 3

Entwickeln eines Assembler-Programms für *set_pixel*

Aufgabe 3

Entwickeln eines Assembler-Programms für *set_pixel*

set_pixel:

Aufgabe 3

Entwickeln eines Assembler-Programms für *set_pixel*

set_pixel:
MOV EDX, 4

Aufgabe 3

Entwickeln eines Assembler-Programms für *set_pixel*

set_pixel:
MOV EDX, 4
MUL EDX

Aufgabe 3

Entwickeln eines Assembler-Programms für *set_pixel*

set_pixel:
MOV EDX, 4
MUL EDX
MOV ESI, EAX

Aufgabe 3

Entwickeln eines Assembler-Programms für *set_pixel*

set_pixel:
MOV EDX, 4
MUL EDX
MOV ESI, EAX
MOV EAX, 7680

Aufgabe 3

Entwickeln eines Assembler-Programms für *set_pixel*

set_pixel:
MOV EDX, 4
MUL EDX
MOV ESI, EAX
MOV EAX, 7680
MUL EBX

Aufgabe 3

Entwickeln eines Assembler-Programms für *set_pixel*

set_pixel:
MOV EDX, 4
MUL EDX
MOV ESI, EAX
MOV EAX, 7680
MUL EBX
ADD EAX, ESI

Aufgabe 3

Entwickeln eines Assembler-Programms für *set_pixel*

set_pixel:
MOV EDX, 4
MUL EDX
MOV ESI, EAX
MOV EAX, 7680
MUL EBX
ADD EAX, ESI
ADD EAX, [display_start]

Aufgabe 3

Entwickeln eines Assembler-Programms für *set_pixel*

set_pixel:
MOV EDX, 4
MUL EDX
MOV ESI, EAX
MOV EAX, 7680
MUL EBX
ADD EAX, ESI
ADD EAX, [display_start]
MOV [EAX], ECX

Aufgabe 3

Entwickeln eines Assembler-Programms für *set_pixel*

set_pixel:
MOV EDX, 4
MUL EDX
MOV ESI, EAX
MOV EAX, 7680
MUL EBX
ADD EAX, ESI
ADD EAX, [display_start]
MOV [EAX], ECX
RET