The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment

The x86 Instruction Set

Lecture 16
Intel Manual, Vols. 2A & 2B

Robb T. Koether

Hampden-Sydney College

Mon, Mar 16, 2009

Outline

The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment

1. The x86 Instruction Set
   - Push and Pop
2. Processing the AST
   - NUM Nodes
   - ID Nodes
   - DEREF Nodes
   - PLUS Nodes
   - ASSIGN Nodes
   - TIMES Nodes
3. Assignment

The x86 Instruction Set

The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment

- There is an online summary of the x86 instruction set.

The Runtime Stack

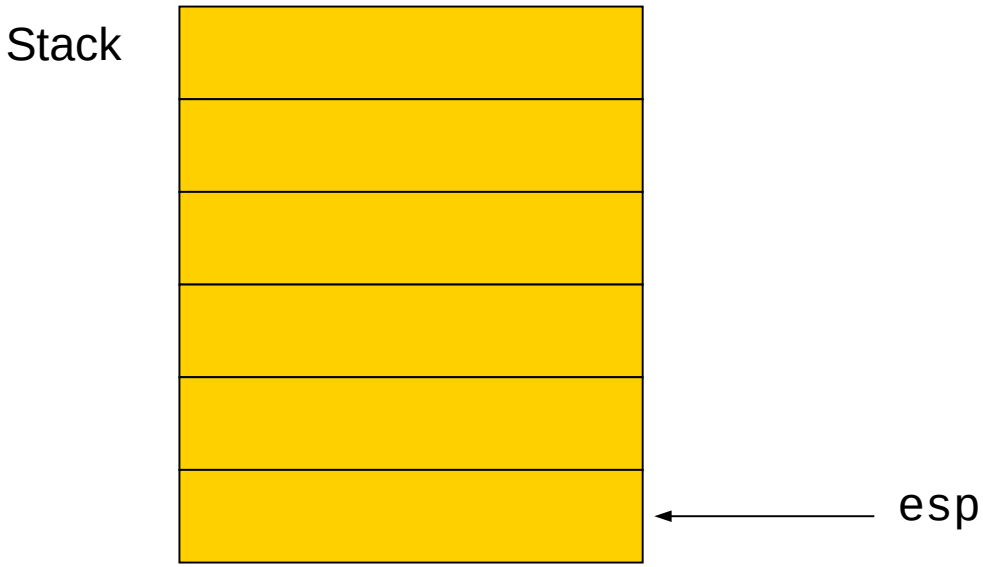
The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment

- The runtime stack is a portion of memory that is used as a stack during program execution.
- The address of the top of the stack is stored in the register `esp`, called the stack pointer.
- The stack grows in a "downward" direction.
  - When values are pushed, `esp` is decremented.
  - When values are popped, `esp` is incremented.

The Runtime Stack

The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment

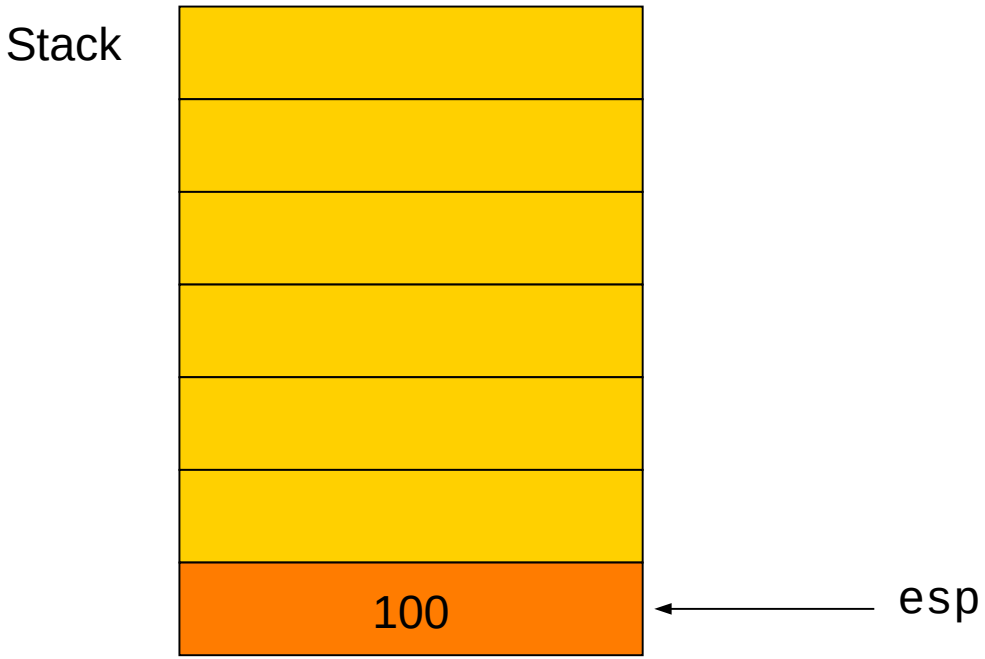
- `esp` points to the "top" of the stack.



The Runtime Stack

The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment

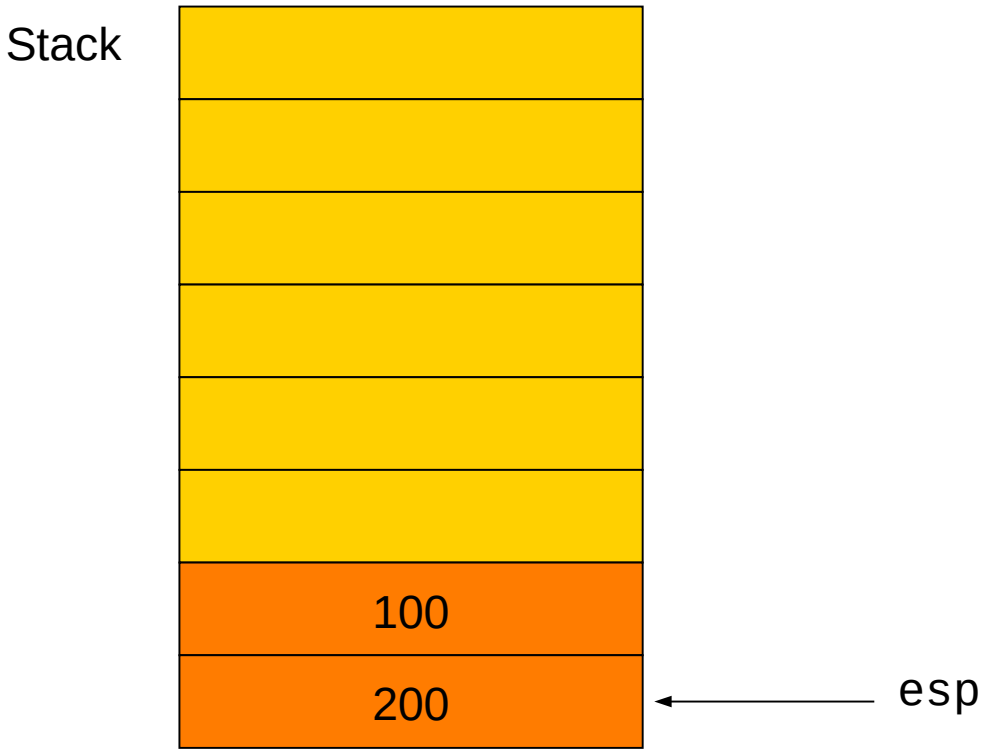
- Push the value 100 and decrement `esp`.



The Runtime Stack

The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment

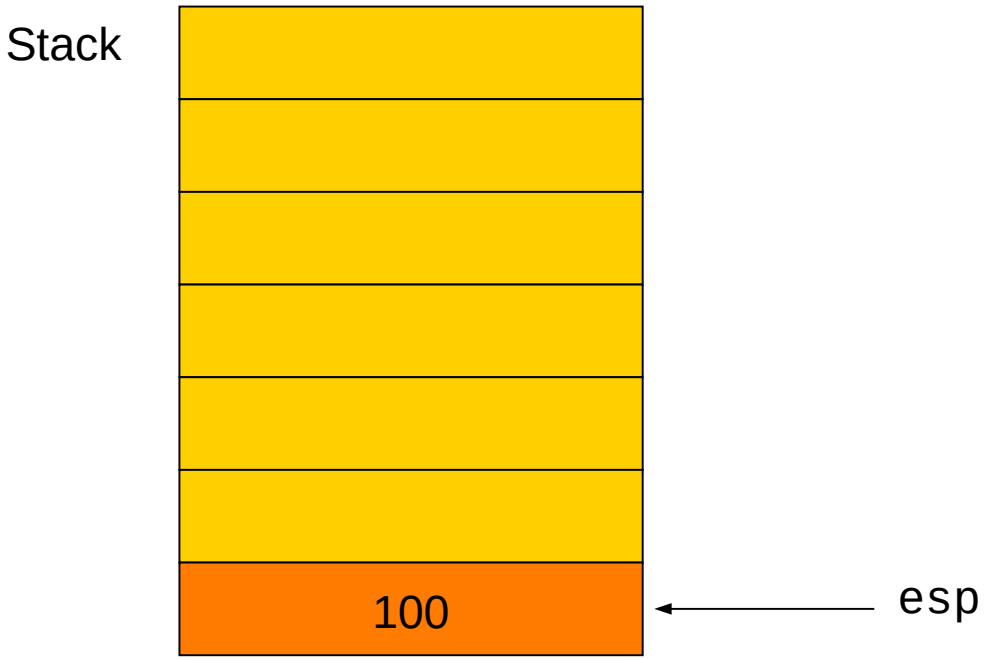
- Push the value 200 and decrement `esp` again.



The Runtime Stack

The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment

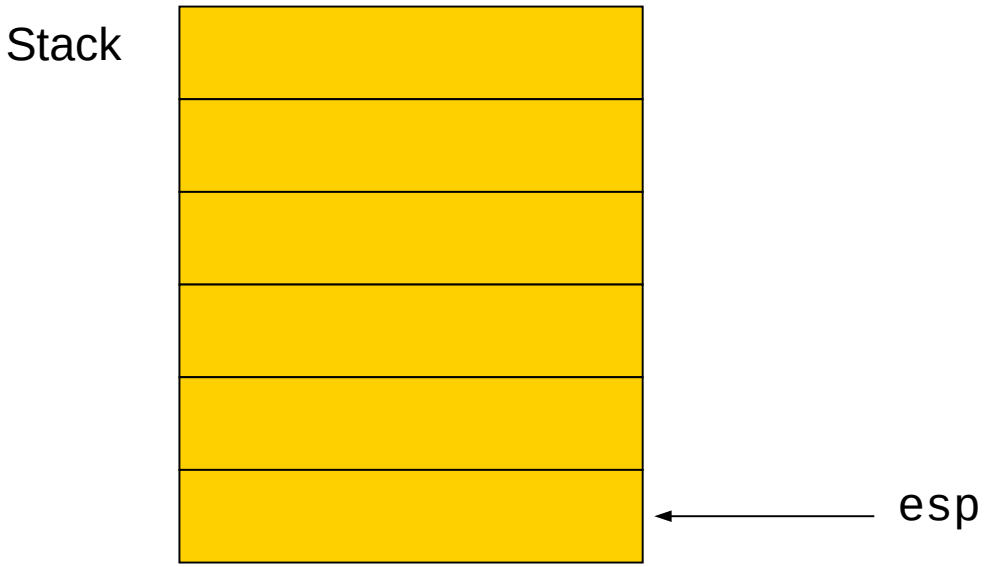
- Pop a value and increment `esp`.



The Runtime Stack

The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment

- Pop another value and increment `esp` again.



The Push and Pop Instructions

The x86 Instruction Set

Robb T. Koether

The x86 Instruction Set

Push and Pop

Processing the AST

NUM Nodes

ID Nodes

DEREF Nodes

PLUS Nodes

ASSIGN Nodes

TIMES Nodes

Assignment

`push` and `pop` Instructions

```
push source
pop  destination
```

where

- *`source`* is a register, a memory address, or an immediate value.
- *`destination`* is a register or a memory address.

The Push and Pop Instructions

The x86 Instruction Set

Robb T. Koether

The x86 Instruction Set
Push and Pop

Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes

Assignment

- The `push` instruction will decrement the stack pointer and then move *`source`* to the stack.
- The `pop` instruction will move the value on the stack to *`destination`* and then increment the stack pointer.

Processing the Syntax Tree

The x86 Instruction Set

Robb T. Koether

The x86 Instruction Set
Push and Pop

Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes

Assignment

- The syntax tree is processed in a left-to-right post-order traversal.
- At each node
  - Process the left subtree.
  - Process the right subtree.
  - Process the node.

Using the Stack

The x86 Instruction Set

Robb T. Koether

The x86 Instruction Set

Push and Pop

Processing the AST

NUM Nodes

ID Nodes

DEREF Nodes

PLUS Nodes

ASSIGN Nodes

TIMES Nodes

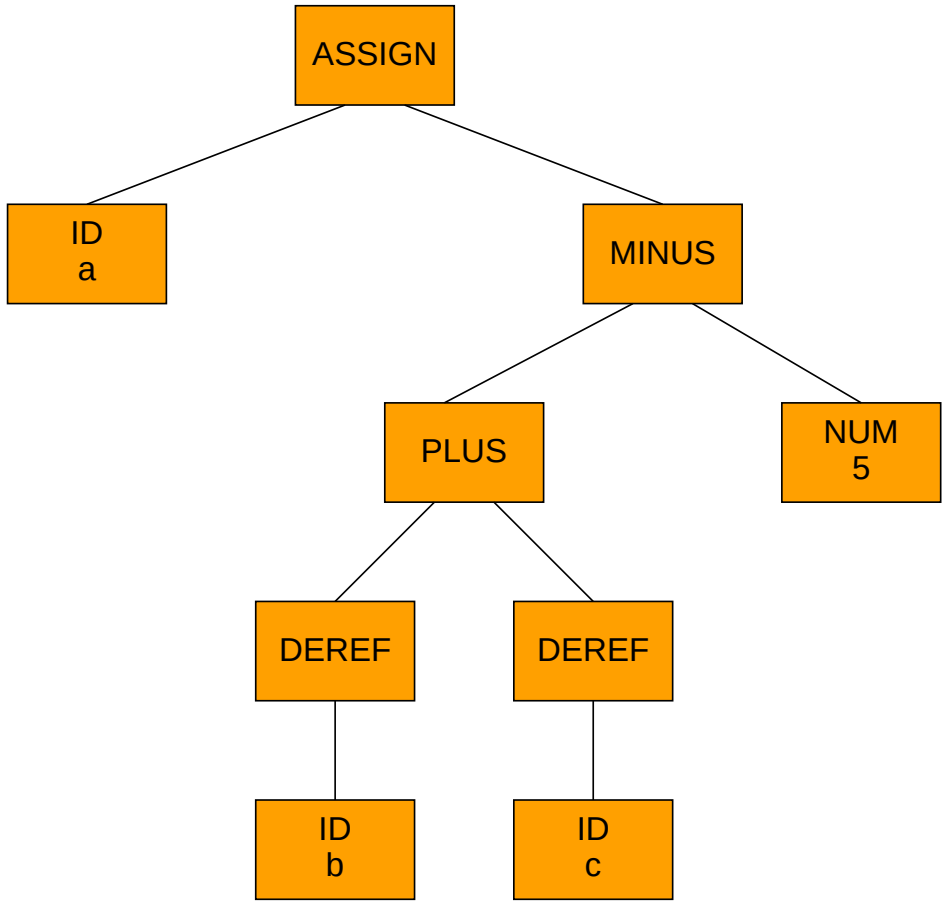
Assignment

- As each node of the syntax tree is executed, it will leave its result on the run-time stack.
- The next node will pop that result off the stack (if it needs it) and then push its own result onto the stack (if there is one), and so on.

Example

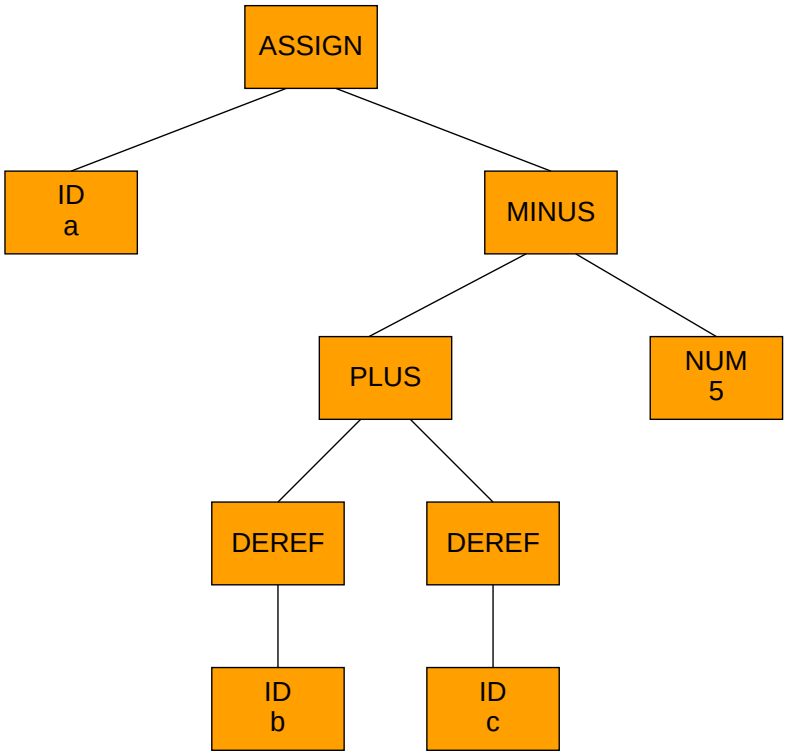
The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment

- The syntax tree for `a = b + c - 5` is



Example

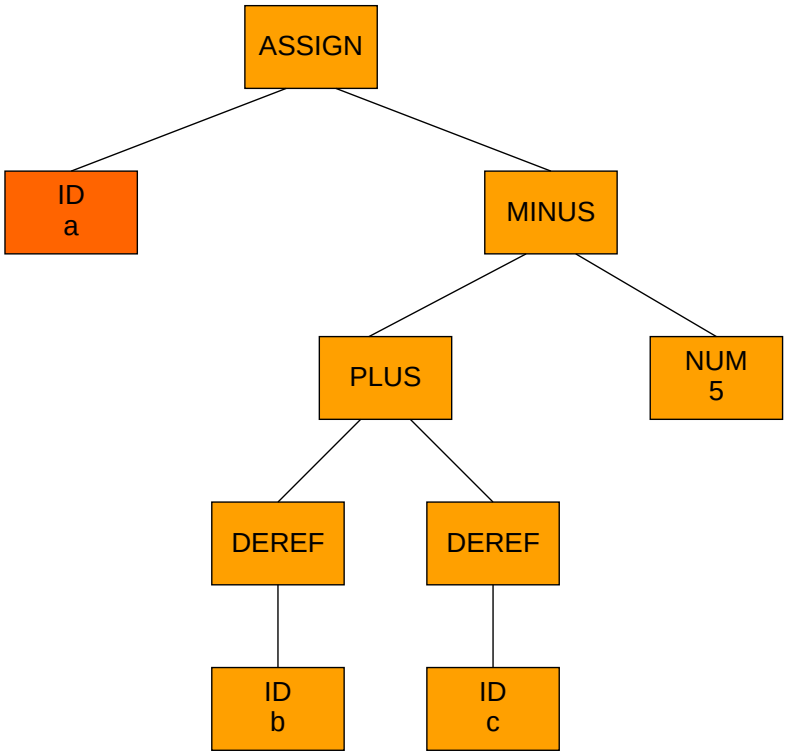
The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment



- `ID` - Push the address of `a`.
- `ID` - Push the address of `b`.
- `DEREF` - Pop the address of `b`, push the value of `b`.
- `ID` - Push the address of `c`.
- `DEREF` - Pop the address of `c`, push the value of `c`.
- `PLUS` - Pop two values, add them, push the result.
- `NUM` - Push `5`.
- `MINUS` - Pop two values, subtract them, push the result.
- `ASSIGN` - Pop the value and the address, store the value at the address, push the result.

Example

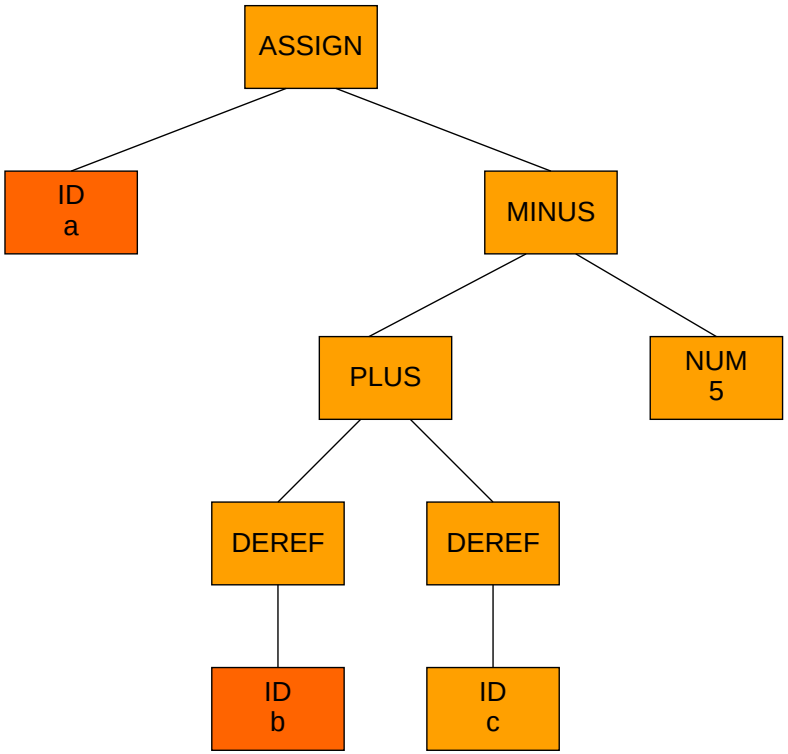
The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment



- `ID` - Push the address of `a`.
- `ID` - Push the address of `b`.
- `DEREF` - Pop the address of `b`, push the value of `b`.
- `ID` - Push the address of `c`.
- `DEREF` - Pop the address of `c`, push the value of `c`.
- `PLUS` - Pop two values, add them, push the result.
- `NUM` - Push `5`.
- `MINUS` - Pop two values, subtract them, push the result.
- `ASSIGN` - Pop the value and the address, store the value at the address, push the result.

Example

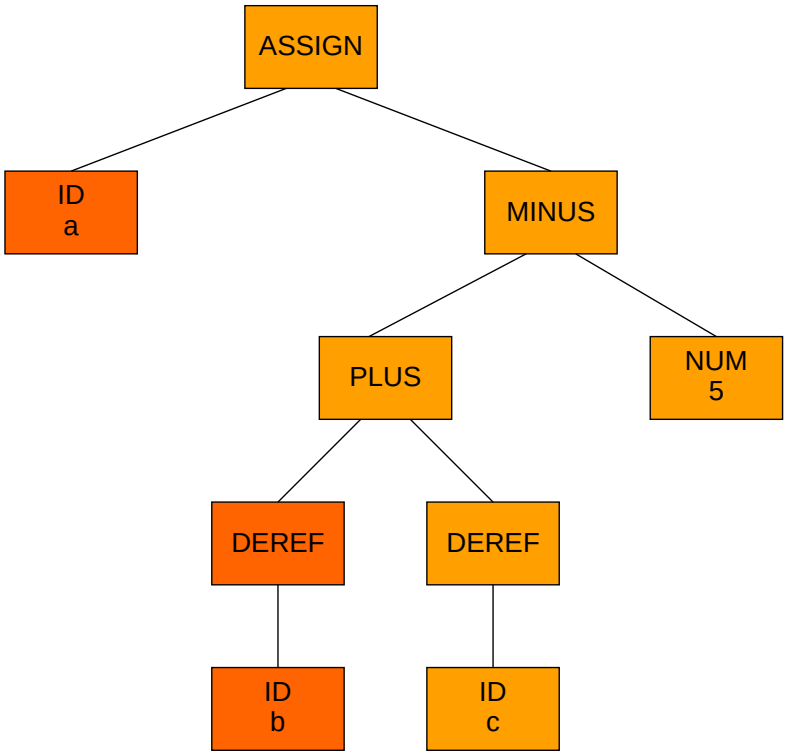
The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment



- `ID` - Push the address of `a`.
- `ID` - Push the address of `b`.
- `DEREF` - Pop the address of `b`, push the value of `b`.
- `ID` - Push the address of `c`.
- `DEREF` - Pop the address of `c`, push the value of `c`.
- `PLUS` - Pop two values, add them, push the result.
- `NUM` - Push `5`.
- `MINUS` - Pop two values, subtract them, push the result.
- `ASSIGN` - Pop the value and the address, store the value at the address, push the result.

Example

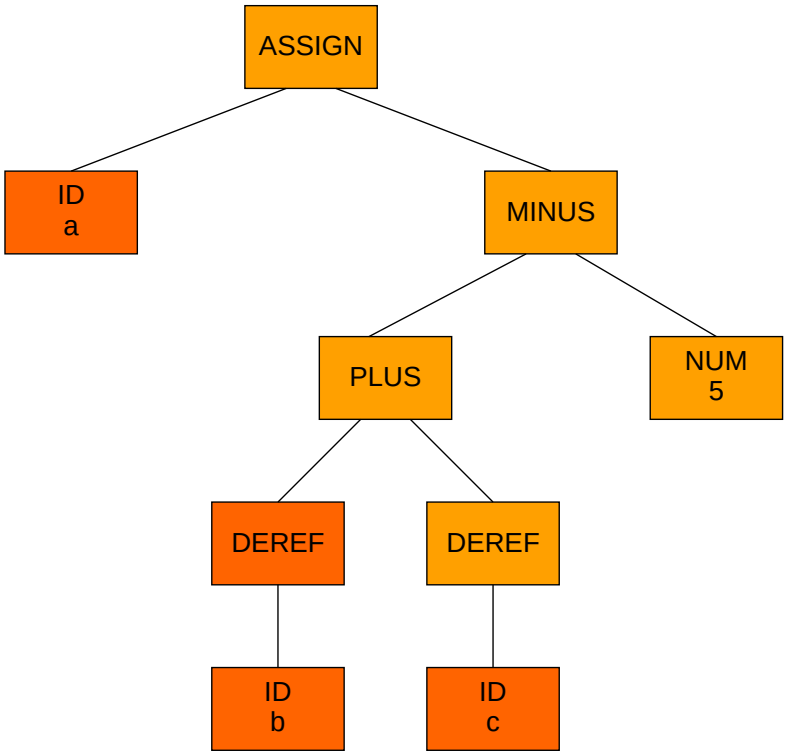
The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment



- `ID` - Push the address of `a`.
- `ID` - Push the address of `b`.
- `DEREF` - Pop the address of `b`, push the value of `b`.
- `ID` - Push the address of `c`.
- `DEREF` - Pop the address of `c`, push the value of `c`.
- `PLUS` - Pop two values, add them, push the result.
- `NUM` - Push `5`.
- `MINUS` - Pop two values, subtract them, push the result.
- `ASSIGN` - Pop the value and the address, store the value at the address, push the result.

Example

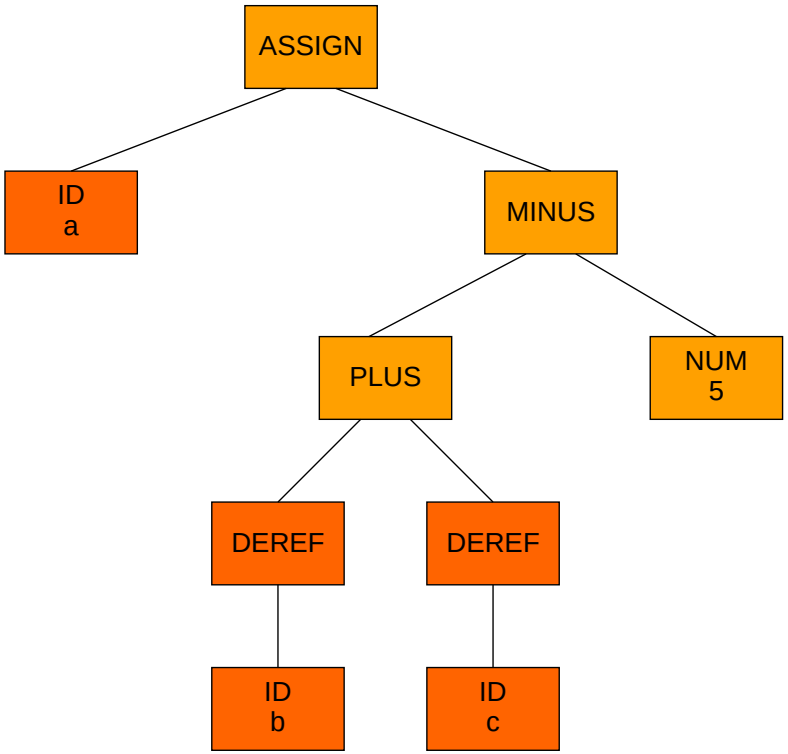
The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment



- `ID` - Push the address of `a`.
- `ID` - Push the address of `b`.
- `DEREF` - Pop the address of `b`, push the value of `b`.
- `ID` - Push the address of `c`.
- `DEREF` - Pop the address of `c`, push the value of `c`.
- `PLUS` - Pop two values, add them, push the result.
- `NUM` - Push `5`.
- `MINUS` - Pop two values, subtract them, push the result.
- `ASSIGN` - Pop the value and the address, store the value at the address, push the result.

Example

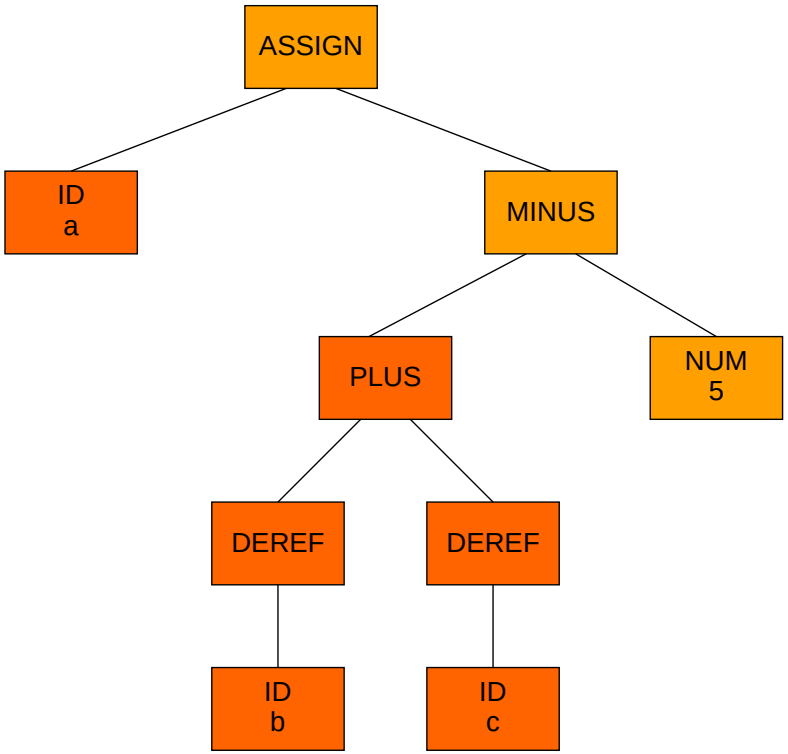
The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment



- `ID` - Push the address of `a`.
- `ID` - Push the address of `b`.
- `DEREF` - Pop the address of `b`, push the value of `b`.
- `ID` - Push the address of `c`.
- `DEREF` - Pop the address of `c`, push the value of `c`.
- `PLUS` - Pop two values, add them, push the result.
- `NUM` - Push `5`.
- `MINUS` - Pop two values, subtract them, push the result.
- `ASSIGN` - Pop the value and the address, store the value at the address, push the result.

Example

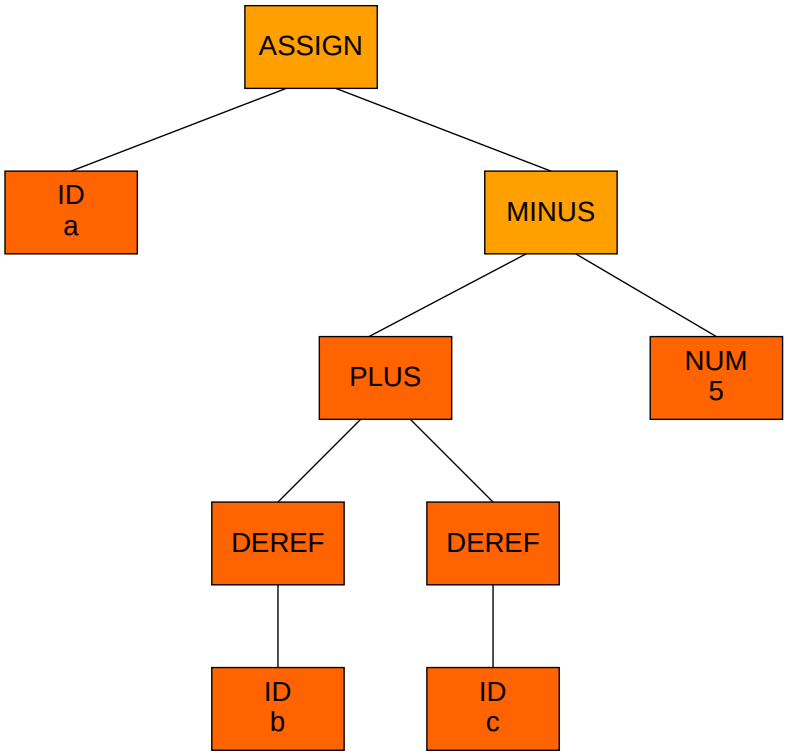
The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment



- `ID` - Push the address of `a`.
- `ID` - Push the address of `b`.
- `DEREF` - Pop the address of `b`, push the value of `b`.
- `ID` - Push the address of `c`.
- `DEREF` - Pop the address of `c`, push the value of `c`.
- `PLUS` - Pop two values, add them, push the result.
- `NUM` - Push `5`.
- `MINUS` - Pop two values, subtract them, push the result.
- `ASSIGN` - Pop the value and the address, store the value at the address, push the result.

Example

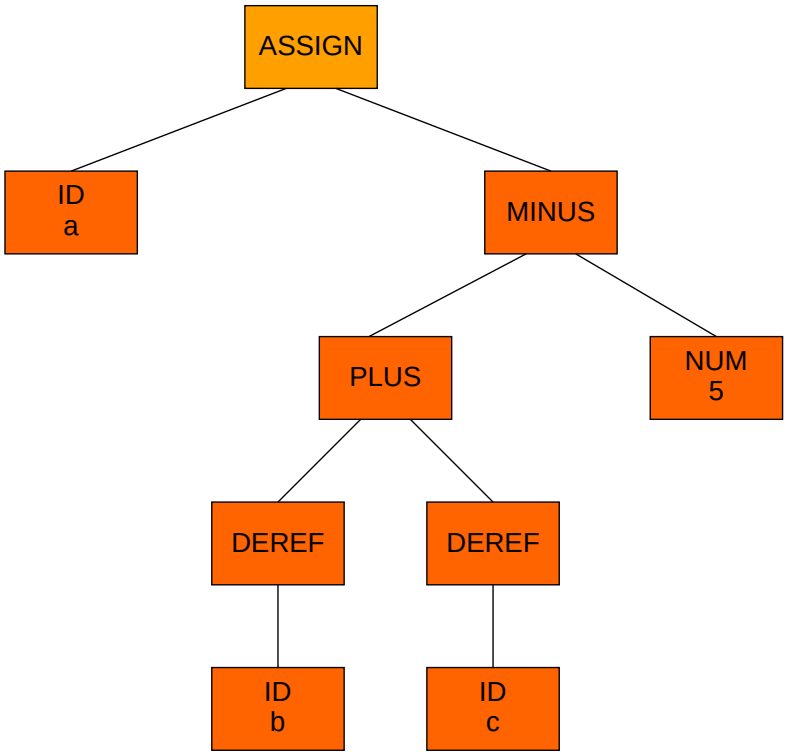
The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment



- `ID` - Push the address of `a`.
- `ID` - Push the address of `b`.
- `DEREF` - Pop the address of `b`, push the value of `b`.
- `ID` - Push the address of `c`.
- `DEREF` - Pop the address of `c`, push the value of `c`.
- `PLUS` - Pop two values, add them, push the result.
- `NUM` - Push `5`.
- `MINUS` - Pop two values, subtract them, push the result.
- `ASSIGN` - Pop the value and the address, store the value at the address, push the result.

Example

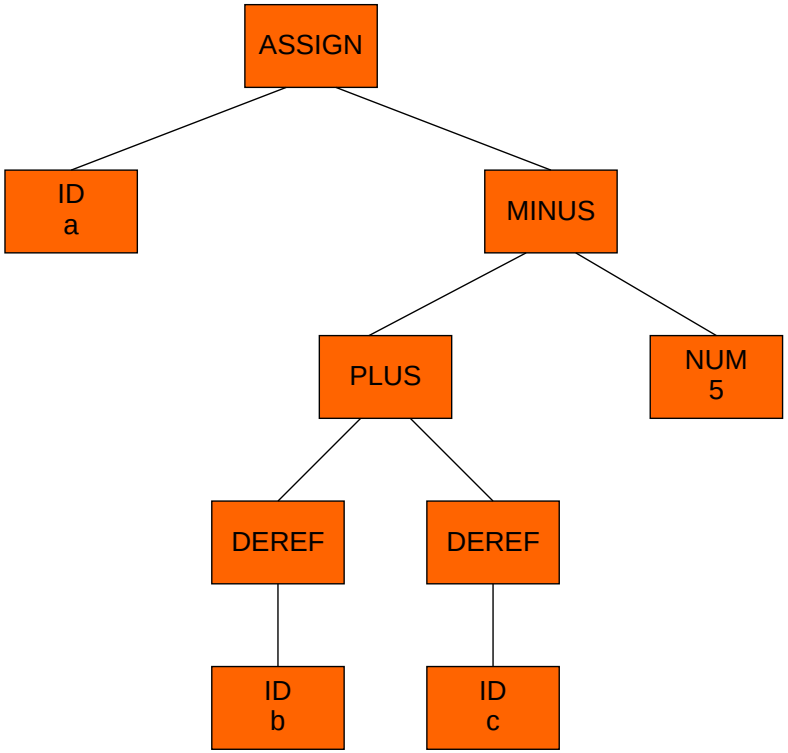
The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment



- `ID` - Push the address of `a`.
- `ID` - Push the address of `b`.
- `DEREF` - Pop the address of `b`, push the value of `b`.
- `ID` - Push the address of `c`.
- `DEREF` - Pop the address of `c`, push the value of `c`.
- `PLUS` - Pop two values, add them, push the result.
- `NUM` - Push `5`.
- `MINUS` - Pop two values, subtract them, push the result.
- `ASSIGN` - Pop the value and the address, store the value at the address, push the result.

Example

The x86 Instruction Set
Robb T. Koether
The x86 Instruction Set
Push and Pop
Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes
Assignment



- `ID` - Push the address of `a`.
- `ID` - Push the address of `b`.
- `DEREF` - Pop the address of `b`, push the value of `b`.
- `ID` - Push the address of `c`.
- `DEREF` - Pop the address of `c`, push the value of `c`.
- `PLUS` - Pop two values, add them, push the result.
- `NUM` - Push `5`.
- `MINUS` - Pop two values, subtract them, push the result.
- `ASSIGN` - Pop the value and the address, store the value at the address, push the result.

A NUM Node

Example (A NUM Node)

```
push $5
```

- A NUM node loads the integer whose value is stored in the node.
- For example, to load 5:
- The $ sign means the "immediate" value 5.

An `ID` Node

Example (An `ID` Node)

```
lea  a,%eax
push %eax
```

- A `ID` node pushes the address of the name that is stored in the node.
- For example, to push the address `a`:
- The instruction "`push a`" would not push the address of `a`; it would push the value at address of `a`, i.e., the value of `a`, which is not what we want.

A DEREF Node

Example (The DEREF Node)

```
pop  %eax
push (%eax)
```

- A DEREF node expects to find a memory address on top of the stack.
- It pushes the value stored at that address.
- The parentheses mean "the value at the address in the register."
- This is the *indirect* addressing mode.

The `add` Instruction

The x86 Instruction Set

Robb T. Koether

The x86 Instruction Set
Push and Pop

Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes

Assignment

The `add` Instruction

```
add source,destination
```

where

- *source* is a register, a memory address, or an immediate value.
- *destination* is a register or a memory address.

- The value at *source* is added to the value at *destination* and the result is stored at *destination*

A PLUS Node

Example (A PLUS Node)

```
pop  %edx
pop  %eax
add  %edx,%eax
push %eax
```

- A PLUS node expects to find two numbers on the stack.
- The right operand should be on top.
- It pops the values, adds them, and pushes the result.

An ASSIGN Node

Example (An ASSIGN Node)

```
pop  %eax
pop  %edx
mov  %eax,(%edx)
push %eax
```

- An ASSIGN node expects to find an address and a value on the stack.
- The value should be on top.
- It pops the value and the address, stores the value at the address, and pushes the value.

The `imul` Instruction

The x86 Instruction Set

Robb T. Koether

The x86 Instruction Set
Push and Pop

Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes

Assignment

- The `imul` instruction performs multiplication of signed integers.
- There are three formats.
- For each format, the destination is `edx:eax`, which holds a 64-bit value.

The `imul` Instruction

The `imul` Instruction (Type 1)

```
imul source
```

where

- *source* is one operand.
- `eax` is the other operand.

The imul Instruction

The x86 Instruction Set

Robb T. Koether

The x86 Instruction Set
Push and Pop

Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes

Assignment

The `imul` Instruction (Type 2)

```
imul register,source
```

where

- *`source`* is a register, a memory address, or an immediate value.
- *`register`* is the destination.

The imul Instruction

The x86 Instruction Set

Robb T. Koether

The x86 Instruction Set
Push and Pop

Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes

Assignment

The `imul` Instruction (Type 3)

```
imul register,source,immediate
```

where

- *`register`* is the destination.
- *`source`* is a register, a memory address, or an immediate value.
- *`immediate`* is a number.

A TIMES Node

Example (A TIMES Node)

```
pop  %eax
pop  %ecx
imul %ecx
push %eax
```

- A TIMES node expects to find two values on the stack.
- It pops them, multiplies them, and pushes the result.

Assignment

The x86 Instruction Set

Robb T. Koether

The x86 Instruction Set
Push and Pop

Processing the AST
NUM Nodes
ID Nodes
DEREF Nodes
PLUS Nodes
ASSIGN Nodes
TIMES Nodes

Assignment

Homework

- Read the descriptions of the above operations in the Intel Manual, Vols. 2A and 2B.
- Also, look up and read about subtraction and division.