

x86

Data Movement Instructions

Arithmetic and Logic Instructions

Control Flow Instructions

A calls B calls C calls D

Subroutine Calling Issues

How should a programming language/compiler handle these?

- Return to call site
- Return Values
- Pass Parameters
- Manage Registers
- Local Variables
- Allow Recursion?

A calls B(c,d,e)

3/23/2005

CS216, Spring 2005

7

C/C++ Calling Convention: Caller

Caller (*before you call the callee*)

- Save caller-saved registers (EBX, ECX, EDX)
- Push parameters on stack (in inverted order)
- Call !!

Caller (*when you return from the callee*)

- Pop parameters off stack
- Return value will be in EAX
- Restore caller-saved registers

3/23/2005

CS216, Spring 2005

8

C/C++ Calling Convention: Callee

Callee (*prologue*)

- Push caller's EBP onto stack, copy ESP into EBP
- Allocate local variables on stack
- Save callee-saved registers (EDI, ESI)
- [then actually do the stuff in the callee function]

Callee (*epilogue*)

- Put return value in EAX
- Restore callee-saved registers
- De-allocate local variables `mov esp, ebp`
- Restore caller's EBP `pop ebp`
- `ret`

3/23/2005

CS216, Spring 2005

9

Caller-Saved Registers

- (EBX, ECX, EDX)
- Caller saves these registers if it cares about the values currently in those registers
- (The compiler will tend to put temporary values in these registers)

3/23/2005

CS216, Spring 2005

10

Callee-Saved Registers

- (ESI, EDI)
- Callee saves these registers if it needs more registers than just EBX, ECX, EDX
- (The compiler will tend to put long-lived values in these registers)

3/23/2005

CS216, Spring 2005

11