

## Plan for Learning about Stack Frames

### Typical C stack frame

- gcc calling convention described in A.5 and A.6 in appendix on assembly code and MIPS

### Stack frame the MiniJava compiler will generate

- Need to match the Wisconsin C-- compiler to implement garbage collection

### General stack frame concept

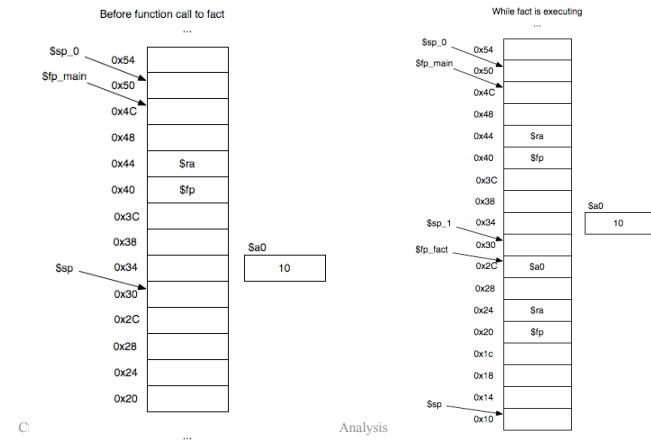
- agreement amongst programmers, procedure call convention
- each agreement has to answer a set of questions
- handling nested procedures
- handling first class functions

CS453 Lecture

Semantic Analysis

1

## Mapping out the stack frame for the fact example



C:

Analysis

## gcc calling convention

### Various rules

- \$sp must stay at 8 byte boundaries
- minimum frame size is 24 bytes
  - four argument registers, \$ra, pad to double word boundary
- first four arguments passed in \$a0-\$a3
- arguments after first four passed on stack, pushed in reverse order
- before a call the caller must save any caller-saved registers that it is using
- at beginning of function, callee must save any callee-saved registers that it will be using
- return value should be in \$v0

CS453 Lecture

Semantic Analysis

3

## A possible implementation of the gcc convention

### Caller

1. Put first four arguments in \$a0-\$a3
2. Push later arguments in reverse order (how do we maintain 8 byte boundary?)
3. Save caller-saved registers that are in use
4. Execute a jal

### Callee

1. Allocate memory for the frame:  $24 + \text{pad}[(\# \text{ params passed over four}) * 4 + (\text{space for callee-saved regs and caller-saved registers}) + (\text{space for locals and temps})]$
2. Save callee-saved registers at highest addresses in the frame
3. Set up frame pointer to top to highest address in the frame
4. Compute ...
5. Put return value in \$v0
6. Restore all callee-saved registers
7. Pop stack frame by adding size of frame
8. Return by using the return address register, \$ra

CS453 Lecture

Semantic Analysis

4