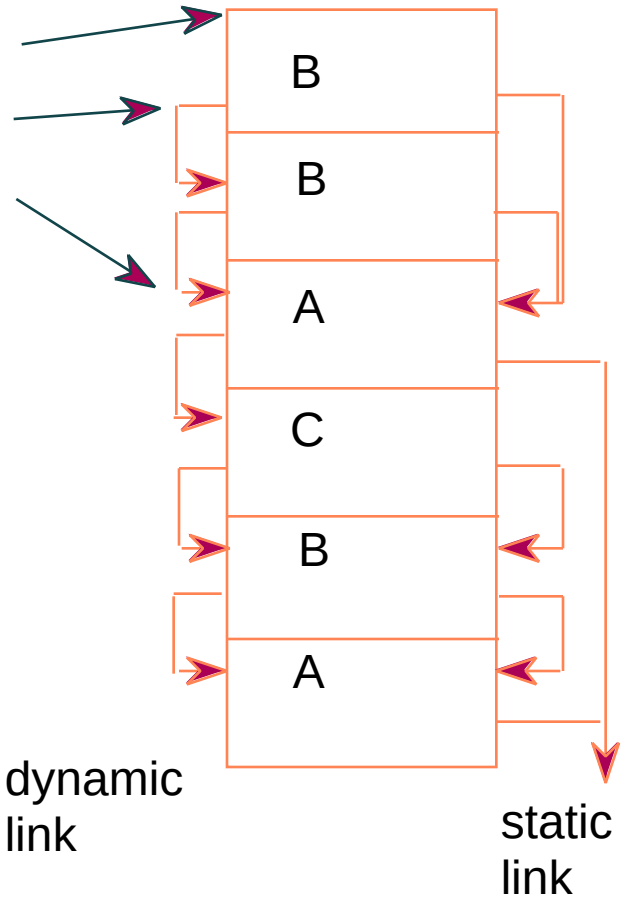
Functions in GCL

Activation Records and Calling

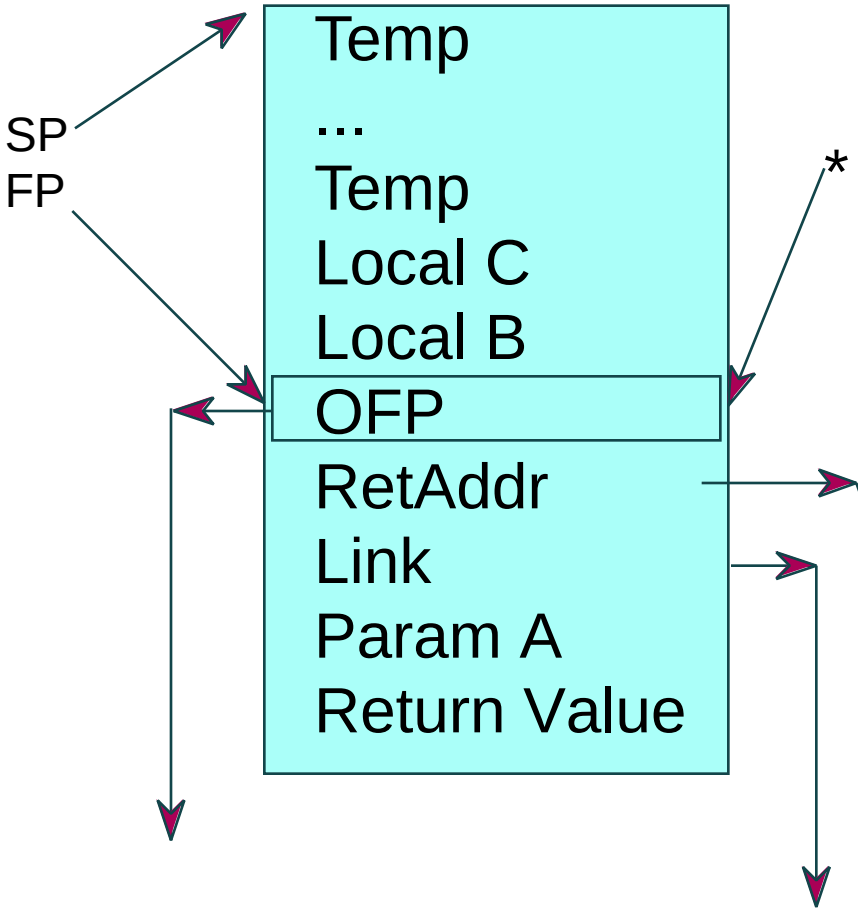
Procedure Call Stack




```
Proc A
    Proc B
        Proc C
        begin
        ...A...
        end
    begin
    ...C....B...
    end
begin
... B...
end
```

Activation Records (Stack Frames)

Activation Record for a function

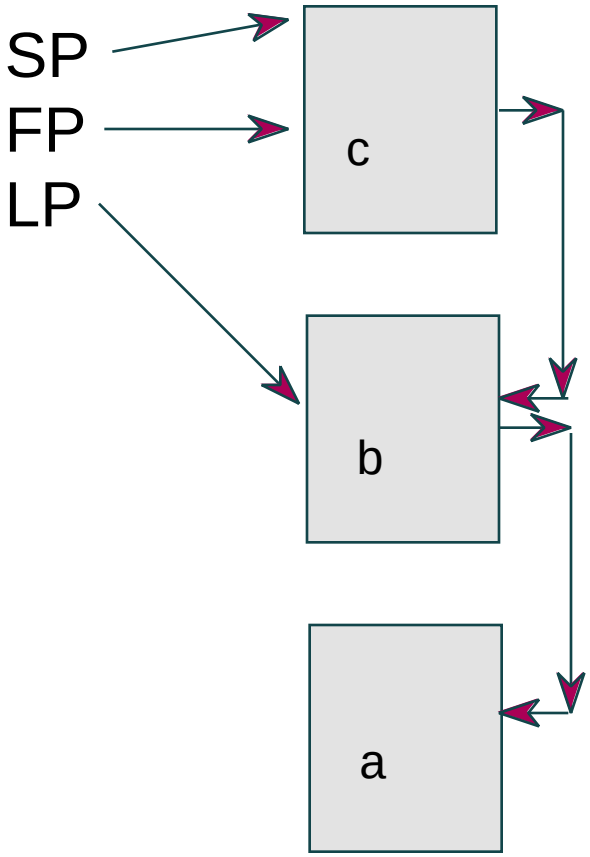



```
proc Foo(Boolean A)
      returns integer
integer B, C;
begin

end;
```

* any pointer to this frame

Accessing Non-Locals



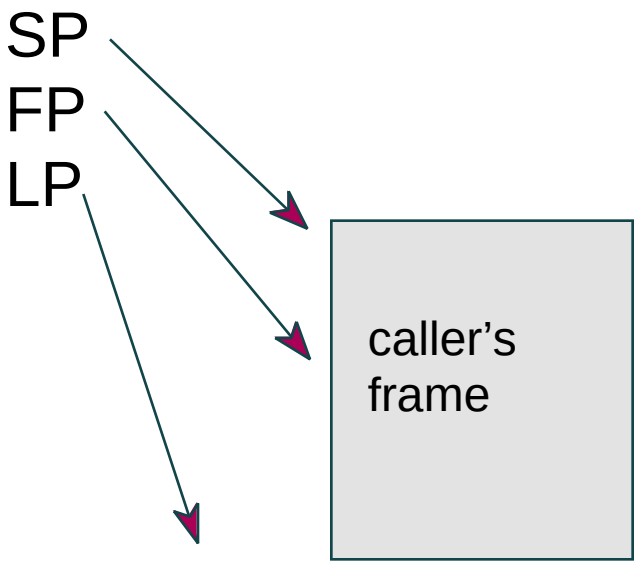

```
Proc A(int...a) -- d l = 2
    Proc B(int...b) -- d l = 3
        Proc C(int...c) -- d l = 4
        begin
        c := a + b;  -- c l = 4
        end
    begin
    ...C....B...
    end
begin
... B...
end
```

c = fp offset, diff=0
b = lp offset, diff=1
a = chain walk,diff>1

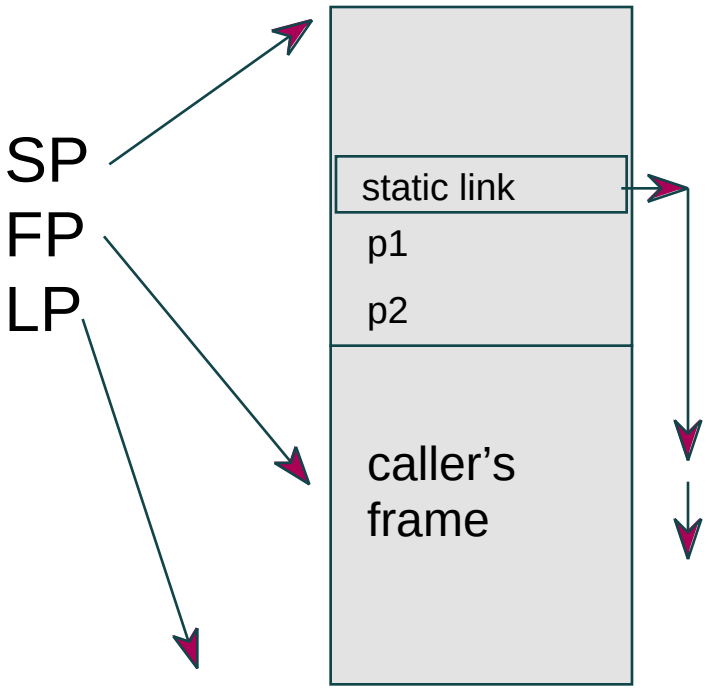
diff = currentLevel - defLevel

Setting Up A Frame part 1 (caller)

Before the procedure call begins.



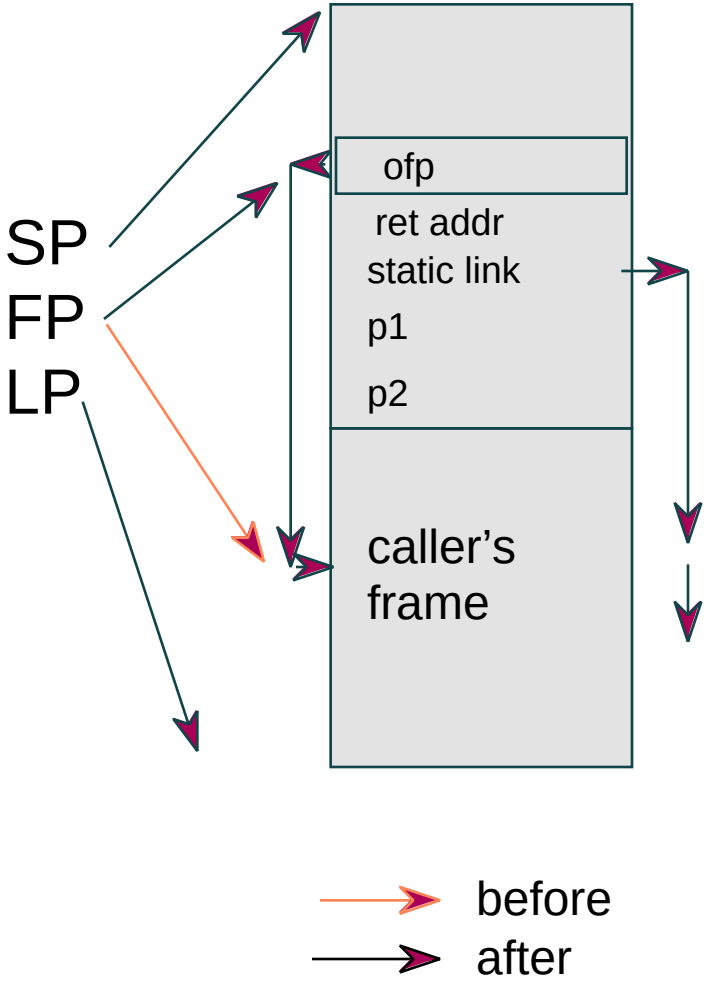
Setting Up A Frame part 1 (caller)



Put up a new frame
Copy in & in-out .
params to it.
Compute its static link
and store it.
JSR to its code--
return address to LP

The setup is relative to SP, the only pointer into this frame at this point.

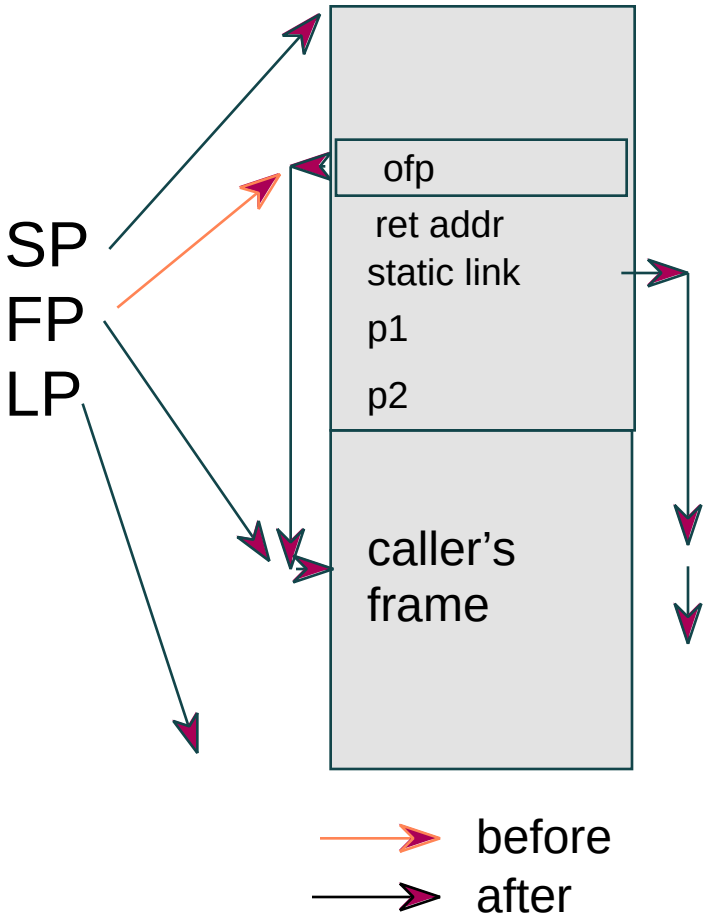
Setting Up A Frame part 2 (called proc)



Save the return address
Save current FP
Set FP to **addr** of ofp slot
Set LP from staticlink slot
Save registers in temp area

You now have a valid frame for the called procedure.
Execute its body.
Note: GCL has an extra step, not shown here.

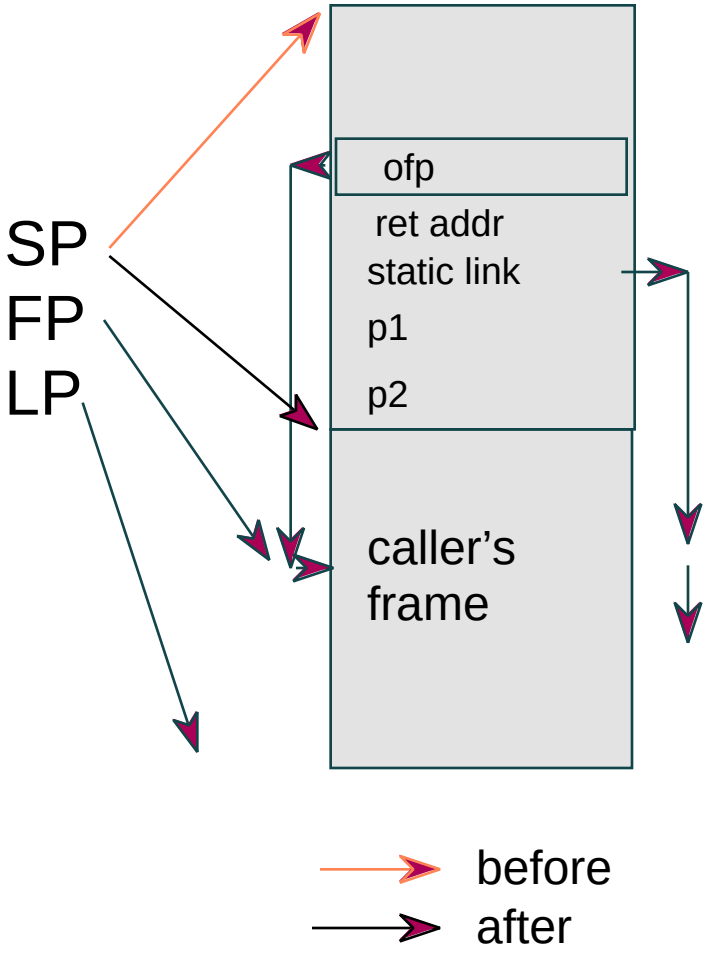
Tearing Down a Frame part 1 (called proc)



Restore the saved regs
Fetch retAddr to LP
Restore the FP from ofp
Jump Indirect thru LP

You will then be executing back at the caller. (Its frame is valid.)

Tearing Down a Frame part 2 (caller)



Restore LP from own StaticLink field.
Copy any out args to final destinations
Reduce SP to original.

You may now continue executing the caller.

Requirements

For this to work we must

a) Give each data item declared in a procedure its own proper offset relative to the OFP slot.

b) Give each declared item its own def level.

c) Clean out the symbol table of all items declared within a proc when we finish compiling the proc.

d) Give each proc a jump label when we declare it.

e) Save the correct size of a proc's stack frame.