Are locals inevitably slow?
M. Anton Ertl, TU Wien

How to code 3dup?

: 3dup.3 {: a b c :} a b c a b c ;

Instr. bytes System
41 158 Gforth AMD64
16 44 iforth 5.0.27 (plus 20 bytes entry and return code)
7 19 lxf 1.6-982-823 32-bit
41 149 SwiftForth 3.11.0 32-bit (calls LSPACE)
26 92 VFX Forth 64 5.11 RC2

\ lxf code
mov eax , [ebp]
mov [ebp-Ch] , eax
mov eax , [ebp+4h]
mov [ebp-8h] , eax
mov [ebp-4h] , ebx
lea ebp , [ebp-Ch]
ret near

VICHECK from Nick Nelson’s "Better Values"

: VICHECK {: pindex paddr -- pindex’ paddr :} 
Checks for valid index
\ paddr is the address of the data, the first cell of which contains the array size
  pindex 0 paddr @ WITHIN IF \ Index is valid
  pindex paddr
ELSE \ Index is invalid
  \ code for reporting the error elided
THEN ;

: VICHECKs ( pindex paddr -- pindex’ paddr ) \ Checks for valid index
\ paddr is the address of the data, the first cell of which contains the array size
  over 0 2 pick @ WITHIN IF \ Index is valid
  the stack already contains the stuff
ELSE \ Index is invalid
  \ code for reporting the error elided
THEN ;
DICHAJICK from Nick Nelson's "Better Values"

<table>
<thead>
<tr>
<th>instructions</th>
<th>bytes</th>
<th>system</th>
</tr>
</thead>
<tbody>
<tr>
<td>locals stack</td>
<td>locals stack</td>
<td>32-bit</td>
</tr>
<tr>
<td>21 9</td>
<td>68 27</td>
<td>lxf 1.6-982-823</td>
</tr>
<tr>
<td>22 5</td>
<td>78 19</td>
<td>VFX Forth 64 5.11 RC2</td>
</tr>
</tbody>
</table>

Discussion and Conclusion

- Are locals inevitably slow? **No**
- lxf is analytical about the return stack (including locals) but only in straight-line code
- C compilers have been register-allocating locals for decades Even on architectures like IA-32 with 8 registers

Counterarguments

- Locals are against the Forth spirit
- Locals are not used enough to justify optimizing them