\SMASHLINK(CALLER,ALINK,CLINK)

; Smashes caller's ALINK and/or CLINK
 with ALINK and CLINK

| caller == NIL | caller=myakink() |

oldalink=caller->alink
oldclink=caller-> clink

| alink !=NIL | alink != clink<br>\|\| alink != oldclink | incusecount(alink) |

; Don't increment twice
if ALINK comes out same as CLINK

caller->alink=alink

| clink!=NIL | alink \|\|<br>(clink != oldalink) | incusecount(alink) |

; If we're only setting the CLINK,
and we're setting it to be the same as the ALINK,
 don't bump count

caller->clink = clink
decusecount(oldclink)

| (oldalink !=<br>oldclink) && alink | decusecount(oldalink) |

; must be careful to increment any
 use counts before decrementing any

| !alink | alink=oldalink |

| !clink | clink=oldclink |

| (alink==clink) &&<br>(clink->usecount==0) &&<br>((blink=caller->blink)==caller-2) &&<br>(blink->ivar ==clink->nextblock) &&<br>(blink->usecnt==0) &&<br> (!clink->nopush) &&<br>(!clink->incall) | caller->slowp=NIL |

;; We have made CALLER fast again
: its alink and clink are same,
usecnt of blink and caller are normal,
bf is contiguous with CALLER and
 CALLER's caller