Store → Load Byte Level Forwarding

SQ_head

addr_fwd_reg

hd > H

SQ_retired
SQ_valid_addr
SQ_addr
SQ_word_in_block
SQ_store_mask
SQ_data

generate
hd_down

ld_tl

generate
ld_tl_up

SQ_addr, SQ_word_in_block

hd_above_H

CAM
C=ld_addr
M=SQ_addr
gives addr matches

ld_addr,
word_in_block

SQ_addr_matches

ld_mask

SQ_store_mask

hd_d

ld_in

if (hd_above_tail)
[2·SQ_SZ] = {0', hd_down & H_up}
else
[2·SQ_SZ] = {hd_down, ld_tl_up}

[2·SQ_SZ][4] bytes_to_fwd
for (i=0; i < SQ_SIZE; i++)
    for(j=0; j<4; j++)
        bytes_to_fwd[i][j]= SQ_store_mask[i][j]
        bytes_to_fwd[i+SQ_SZ][j] = SQ_store_mask[i][j]

4 PSELS

P
S
E
L

4n
2·SQ_SZ

PSEL to
select which
byte to take
data from

[2*SQ_SIZE] width
selects 1

for i in SQ_size
byte_to_fwd[i:]
psel_out[i:] |
psel_out[i+SQSZ];

4n
SQ_SZ

for i in 4
for j in SQ_SIZE
for k in 8

data[i·8+k] = psel_out[i:][j] &
SQ_data[j][i·8+k];