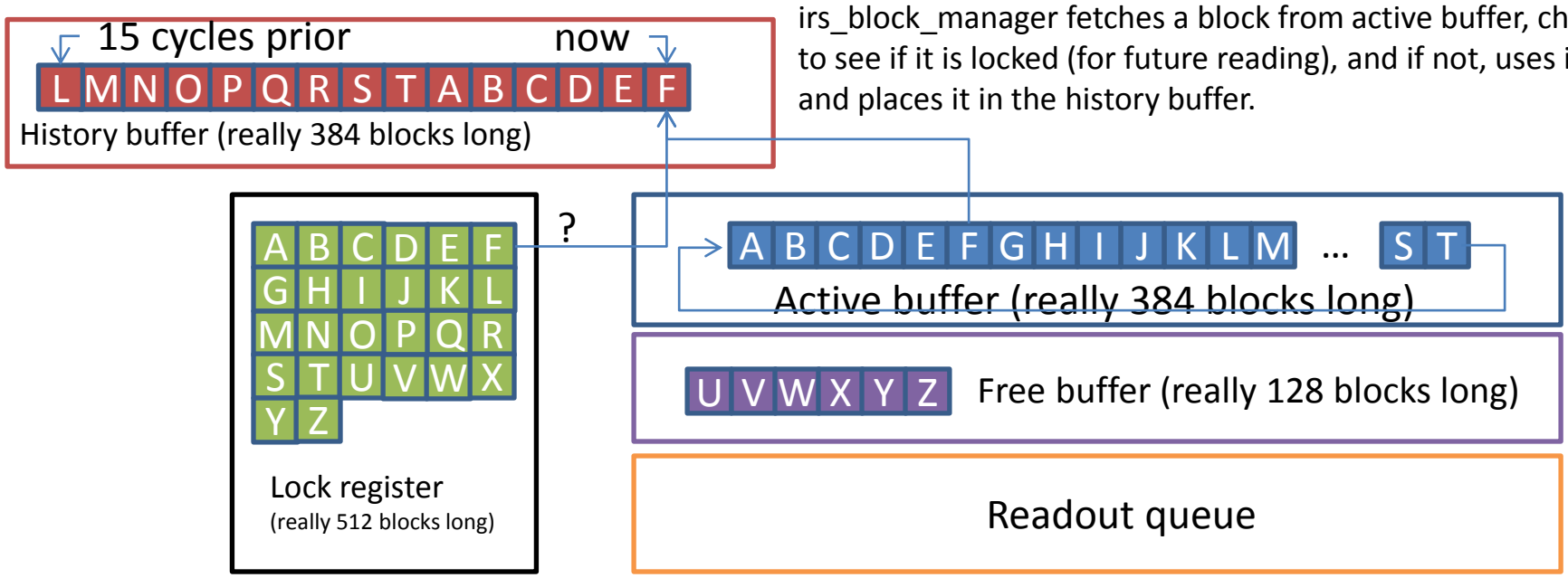
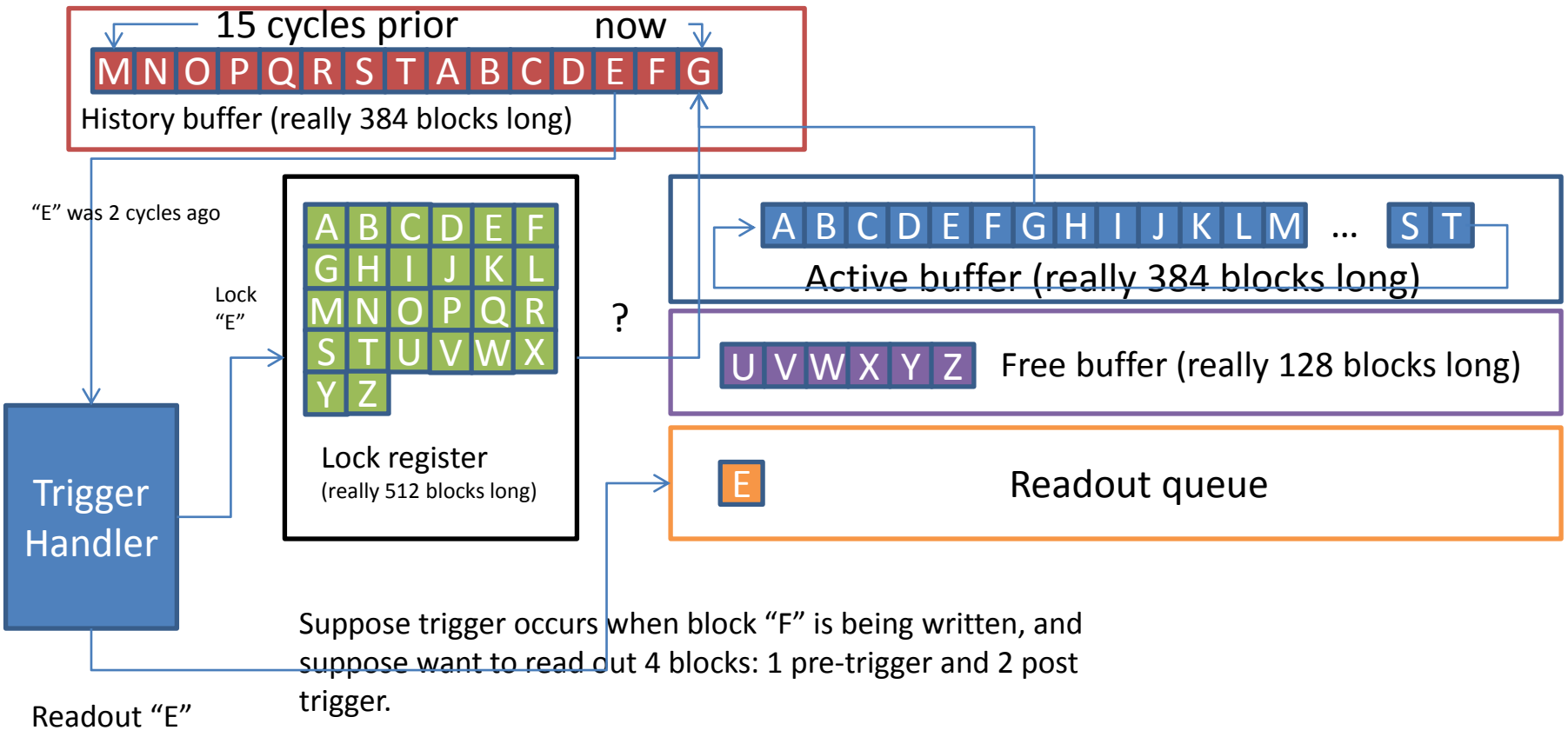
THE IRS_BLOCK_MANAGER ARCHITECTURE: BEFORE A TRIGGER



THE IRS_BLOCK_MANAGER ARCHITECTURE: WHEN TRIGGER OCCURS



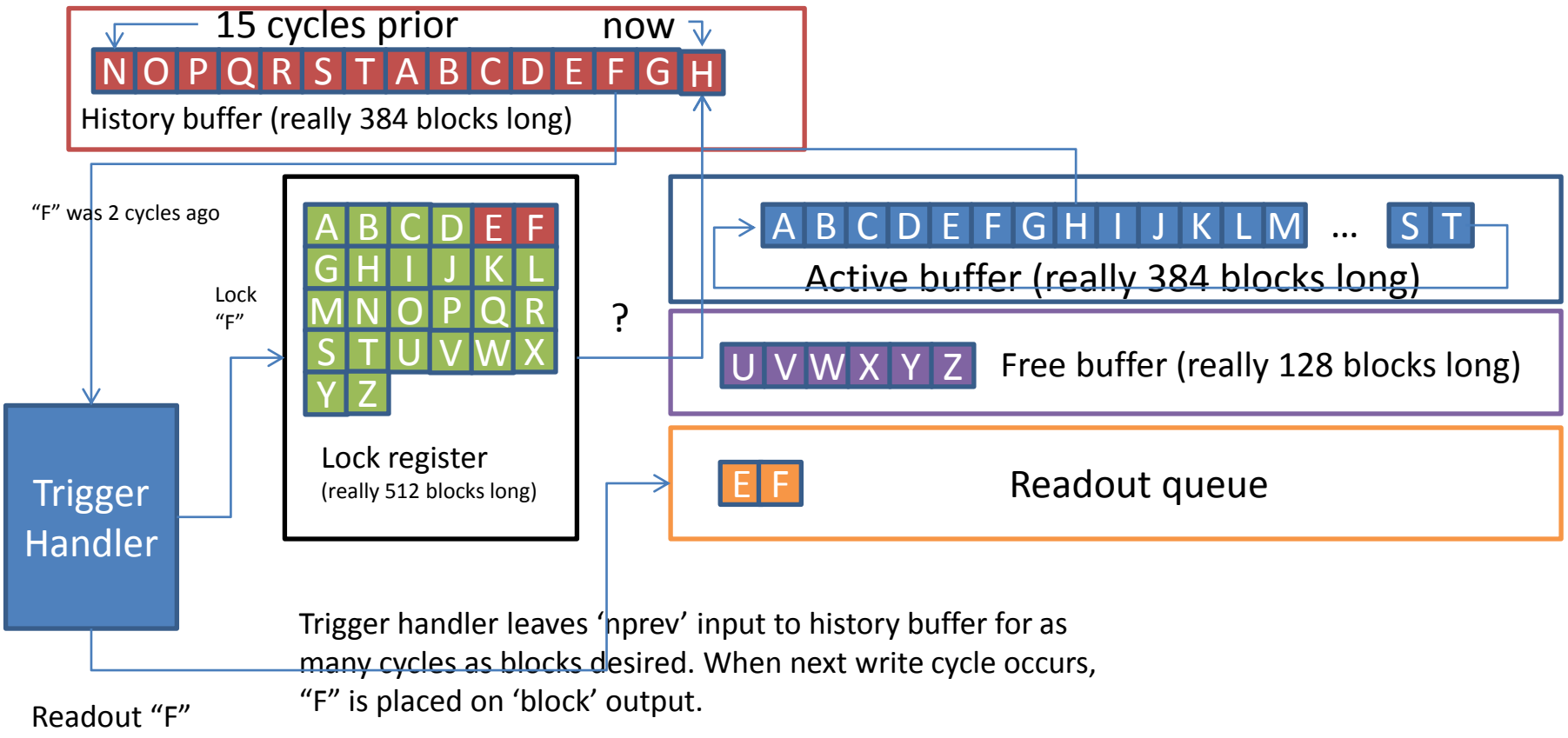
Suppose trigger occurs when block "F" is being written, and suppose want to read out 4 blocks: 1 pre-trigger and 2 post trigger.

Trigger places "2" on 'nprev' input to history buffer. At next write cycle (when "G" is written to) history buffer puts "E" on 'block' output.

Trigger handler then places "E" on lock_address input of block manager, and asserts "lock" and "lock_strobe".

Trigger handler places "E" in readout queue.

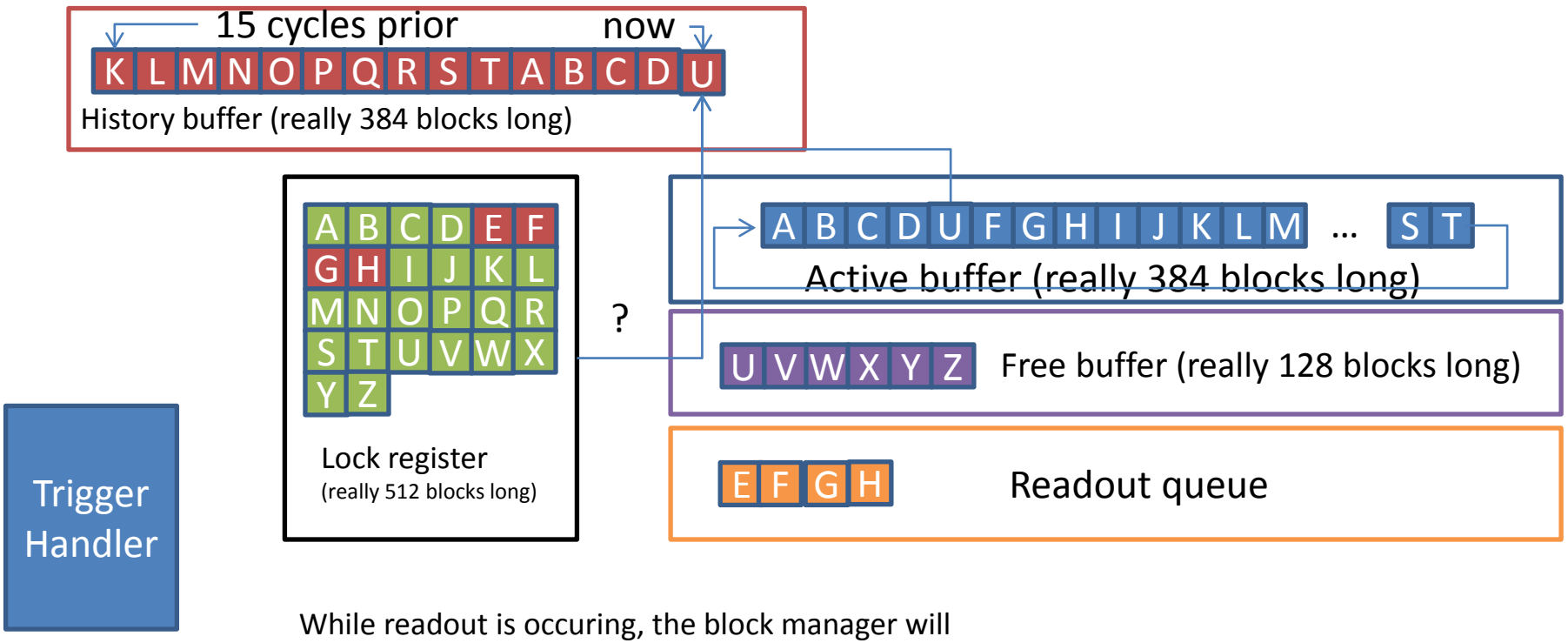
THE IRS_BLOCK_MANAGER ARCHITECTURE: AFTER TRIGGER OCCURS



Trigger handler leaves 'nprev' input to history buffer for as many cycles as blocks desired. When next write cycle occurs, "F" is placed on 'block' output.

Process repeats until E,F,G,H are placed in readout queue and locked in the lock register.

THE IRS_BLOCK_MANAGER ARCHITECTURE: WHILE READOUT IS OCCURRING

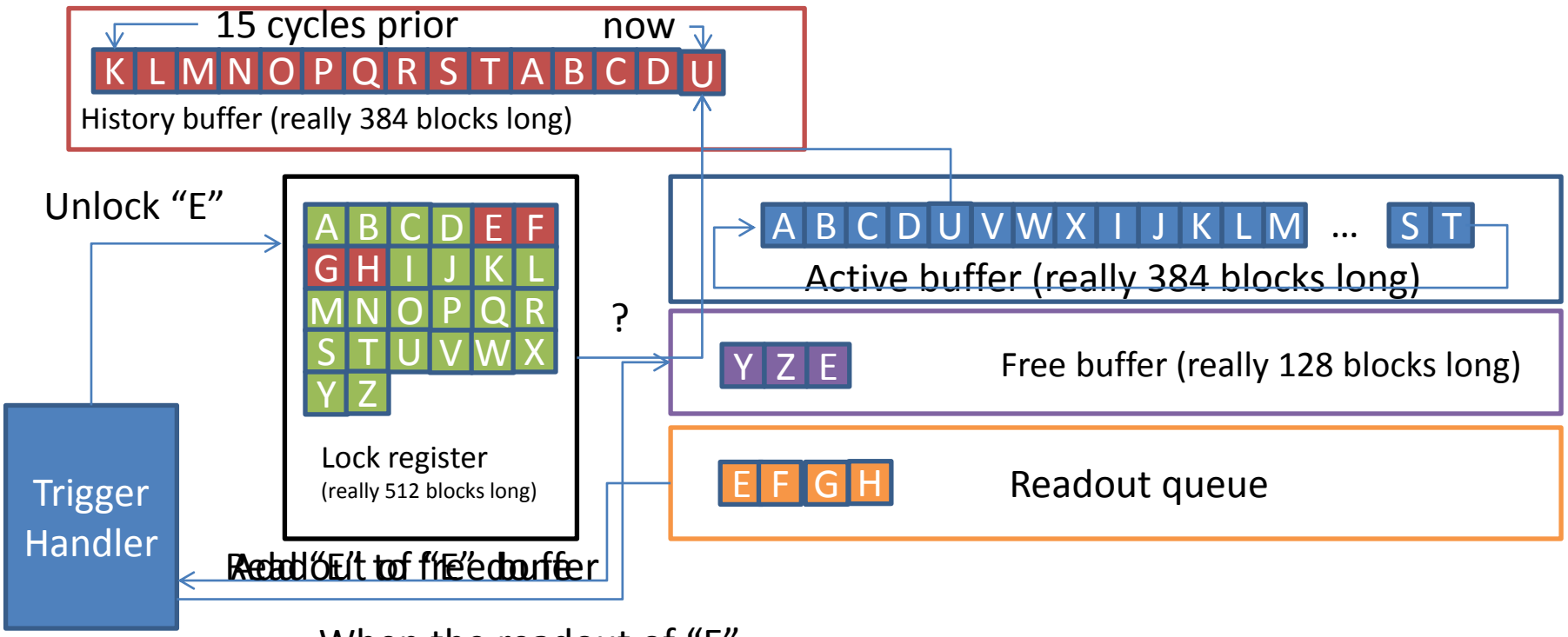


While readout is occuring, the block manager will reencounter “E” in the active buffer.

It will discover “E” is locked, and use the first block from the free buffer instead, deleting it from the free buffer.

It then *overwrites* “E” in the active buffer with “U”.

THE IRS_BLOCK_MANAGER ARCHITECTURE: WHEN READOUT COMPLETES



When the readout of "E" completes…

Trigger handler places "E" on lock_address input, "0" on 'lock', and asserts 'lock_strobe'

Trigger handler then puts "E" on 'free_address' and asserts 'free_strobe'