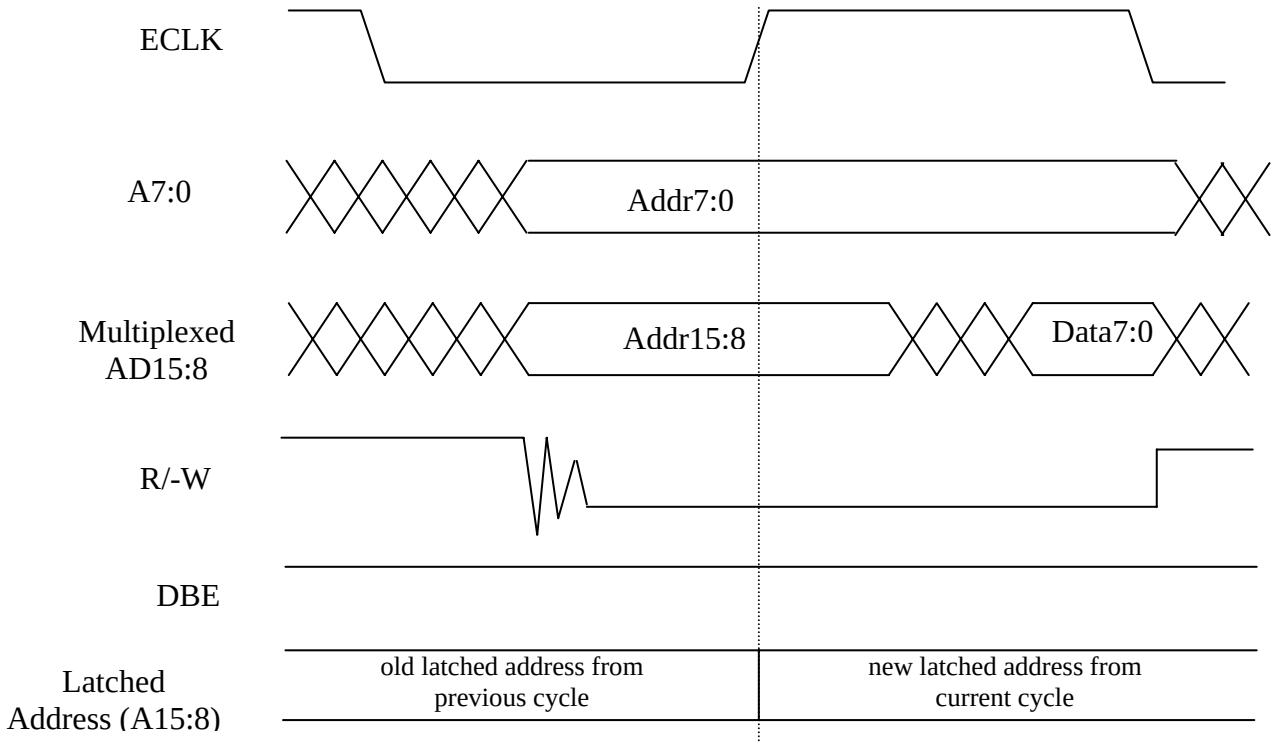
R/~W Ringing Problem on UF 68HC12 Board (if Bad CE when Writing)

Write Cycle



Note: This problem exits if the write part of a RAM chip enable has an equation like this:

RAM_CE = ~RESET * f(Addresses) * W, where R/~W=R(H)=W(L)

Note that there is **no** E-clock in the above equation.

Typical Example of the Problem (assumes RAM at $C000-CFFF):

1. Write $55 to address $C000, then write $55 to $C001, and finally one more write of $55 to $C002.
2. You will see upon reading memory, $C000 has contents $C0, $C001 has contents $C0 and $C002 has contents $55 (correct value).

Explanation of the Cause:

1. The RAM_CE is true (using the above equation) in the first half of the E-clock, every time W is true (since the previous address is also within the RAM's memory map). Since the previous cycle's address (high byte) is on the address/data bus (A15:8/D7:0) during the early part of the first half of the E-clock when W is true, the high byte of the address is what gets written to the **previous** RAM address (over writing the correct value that had been previously stored.
2. The reason the last address gets the correct value is because the assumed next instruction is **not** another write cycle. If it was, it would also get changed to the wrong value.

A Solution for the Problem:

Add the **E-clock** signal to the write portion of **all** chip enable equations. Here is a short example. Let's say we want to put an 8-bit output latch at $8000 and use partial-address decoding. The CE equation should be:

CE_for_Write = ~RESET * A15 * ~A14 * ~A13 * ~A12H * W * **E**

This will ensure that the chip enable will only go true when E is high thus avoiding the ringing area altogether.