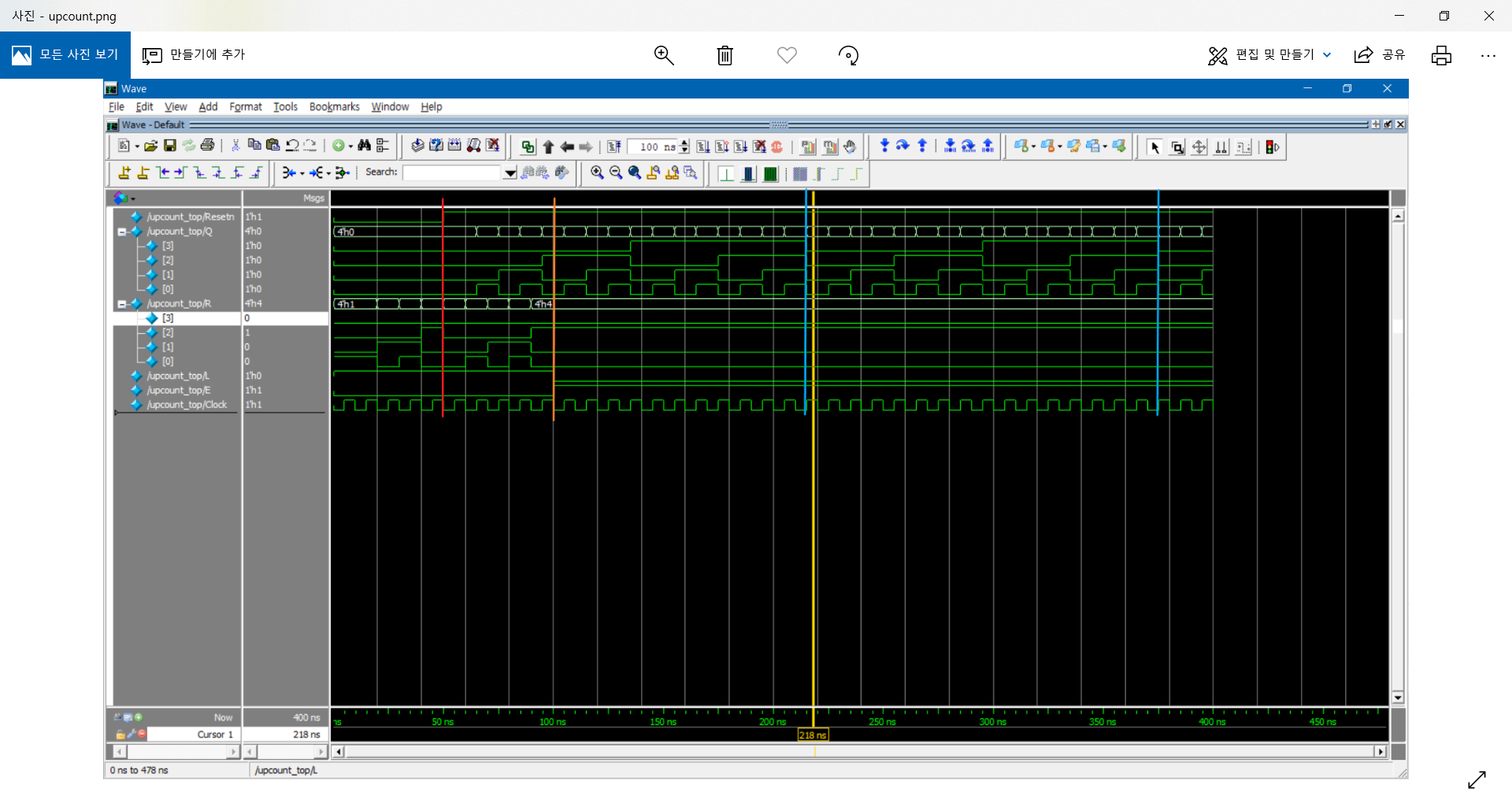
**1. upcount파형**



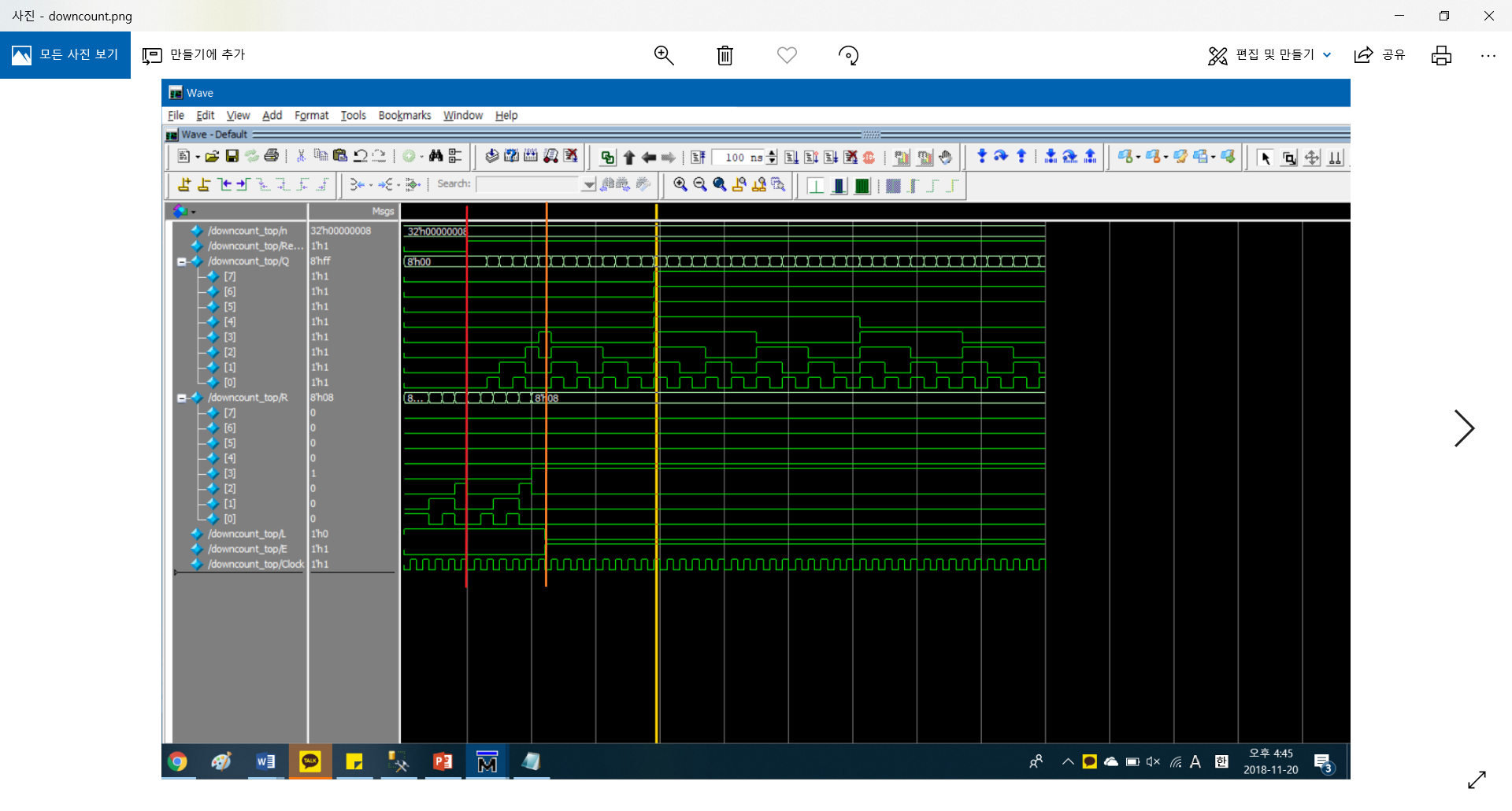
위에서 순서대로 Reset, Q, R, L, E, Clk 값입니다.

R값이 0일 때에는 , L이 1이고 R값이 존재해도 Q에 반영되지 않습니다.

R값이 1이되는 빨간선 이후로, R값이 그대로 Q값으로 출력되게 됩니다.

L이 0이되고, E이 1이된 주황선 이후로는, Q값이 UP\_count 됩니다.

**2. down count파형**



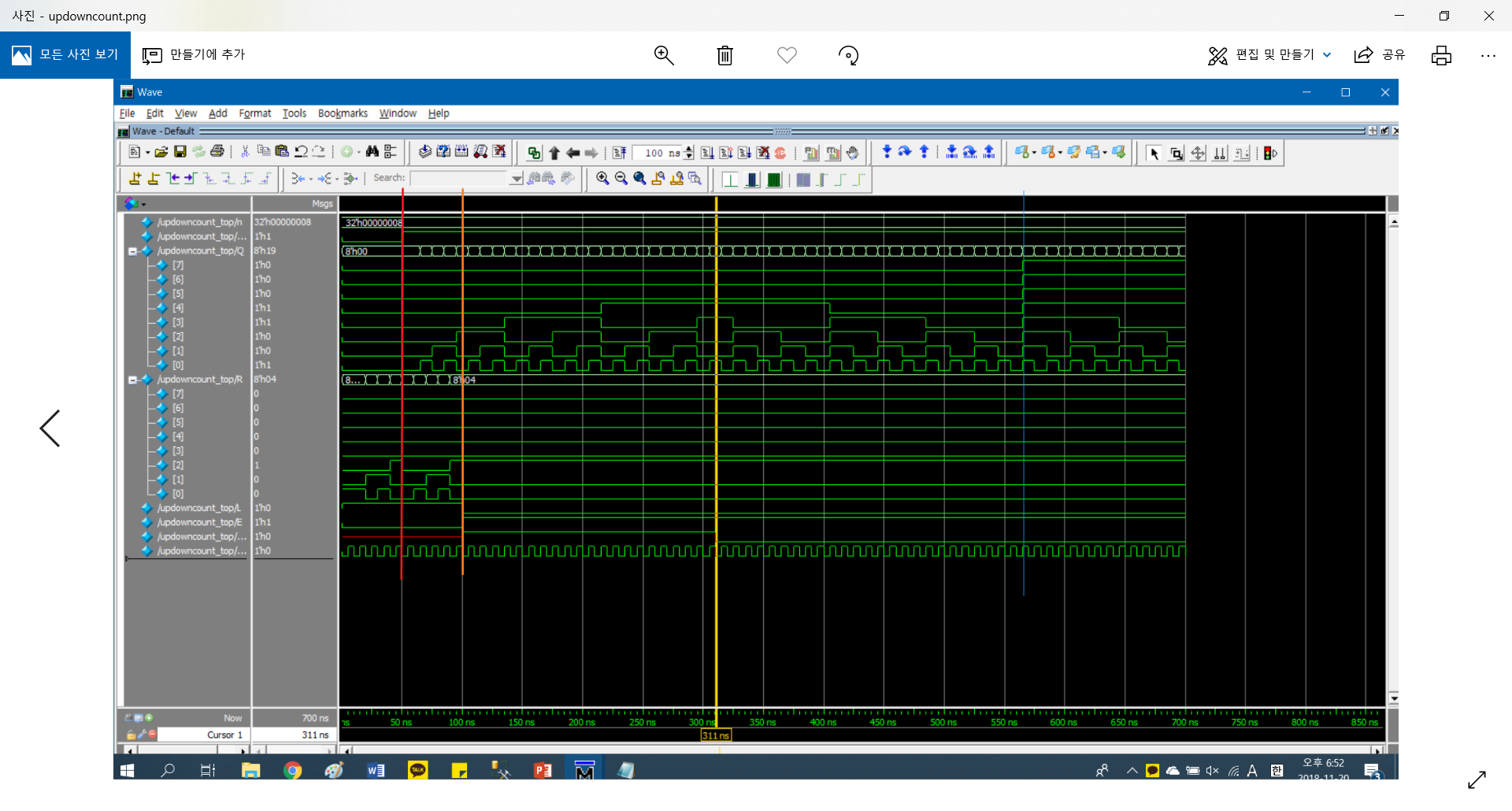
위에서 순서대로 Reset, Q, R, L, E, Clk 값입니다.

R값이 0일 때에는 , L이 1이고 R값이 존재해도 Q에 반영되지 않습니다.

R값이 1이되는 빨간선 이후로, R값이 그대로 Q값으로 출력되게 됩니다.

L이 0이되고, E이 1이된 주황선 이후로는, Q값이 DOWN\_count 됩니다.

**3.up-down counter파형**



위에서 순서대로 Reset, Q, R, L, E, up\_down, Clk 값입니다.

R값이 0일 때에는 , L이 1이고 R값이 존재해도 Q에 반영되지 않습니다.

R값이 1이되는 빨간선 이후로, R값이 그대로 Q값으로 출력되게 됩니다.

L이 0이되고, E이 1이되고, up\_down값이 1이된 주황선 이후로는, Q값이 UP\_count 됩니다.

up\_down값이 0이된 노란선 이후로는 Q값이 DOWN\_count 됩니다.

*--참고--*

parallel load는 병렬 로드로 기존 값에 상관 없이 data - R값을 그대로 출력하는 것을 말하고 count는 주어진 진수에 맞추어 Resetn=1 이고 L(load)=0 E(enable)1일 때 counter된다.

clock이 상승에지일 때 값이 변한다고 가정하면.

always @(posedge clock) begin

if(~reset) Q<=0; //여기서의 reset은 동기식 제어신호이다.(clock에 동기)

clock이 상승에지 이거나 reset이 하강에지일 때 값이 변한다고 가정하자

always @(posedge clock or negedge reset)

if(~reset) Q<=0; //여기서의 reset은 비 동기식 제어신호이다.(clock에 영향 받지 않는다.)