

Static 1-hazards can also be detected from the expression for \mathcal{F} .

Suppose we have two assignments to (x_1, \dots, x_n) , denoted u and v , which differ only in one variable. We will assume without loss of generality that this variable is x_1 . The expression for \mathcal{F} will have :

1. products, S , in which the literals x_1 and x'_1 do not appear,
2. terms that are of the form Px_1 , (P is independent of x_1 and x'_1)
3. terms that are of the form Qx'_1 , (Q is independent of x_1 and x'_1),
4. terms that are of the form $Rx_1x'_1$, (R is independent of x_1 and x'_1)

Without loss generality assume that the transition from u to v involves changing $x_1 = 1 \rightarrow 0$.

- $\mathcal{F} = 1$ at both u and v (when they are extended to $2n$ variables with the true steady state values of the complements).
- if a term of type (1) above evaluates to 1 at u it must also evaluate to 1 at v . So there can be no transient effect seen at the output of the network.
- if no such term exists then all type (2) terms must go to 0 and all type (3) terms must go to 1. The timing of the network could be set so that no type (3) term goes to 1 before all of the type (2) terms go to 0. This would imply that during the transient $x'_1 = 0 = x_1$. Note that the terms of type (4) are irrelevant to this transient so there is a static 1-hazard.

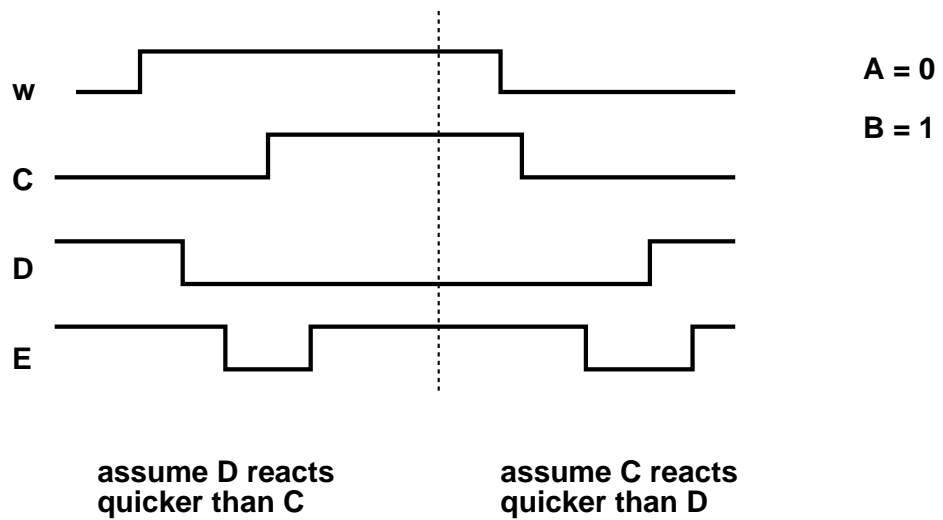
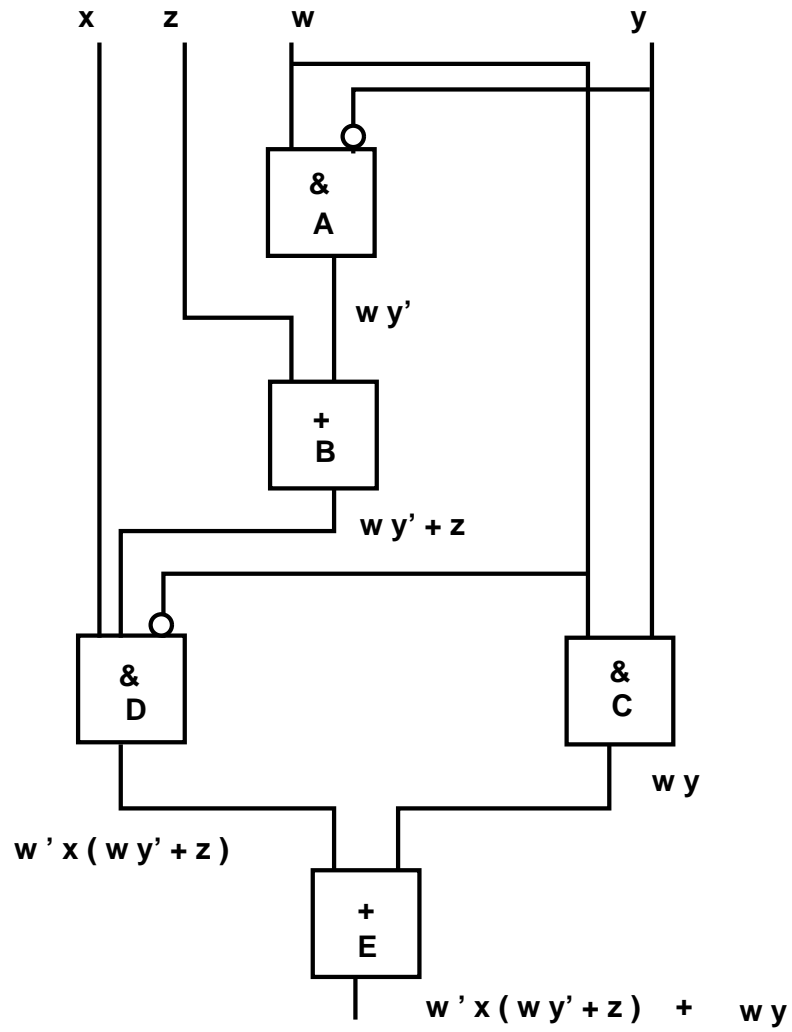
This can be summarized as the following theorem:

Theorem: A static logic 1-hazard exists if and only if

- (a) There is a pair of inputs for such that for both inputs the network outputs 1.
- (b) There is no stable 1-set that is active for both inputs.

In our example, this is true for the transition

$$(0, 1, 1, 1) \rightarrow (1, 1, 1, 1)$$



Static 1-hazards can occur in two-level AND/OR networks. But we have a dual to our previous result:

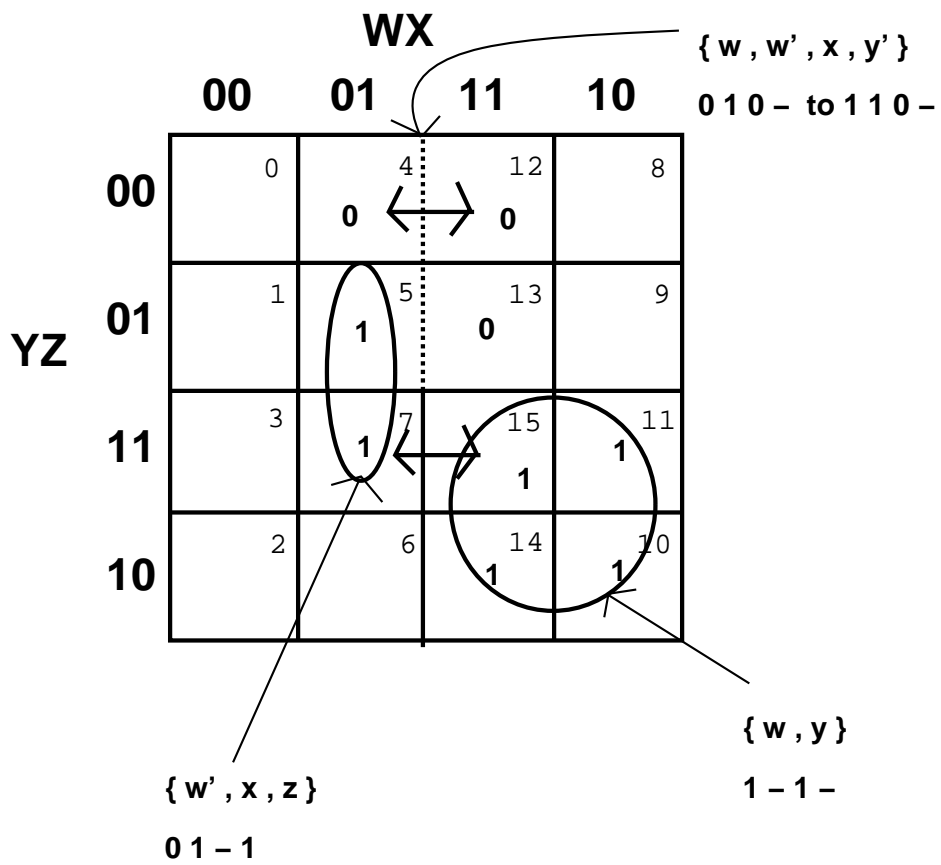
Theorem: No static 1-hazards exist in a two-level OR/AND network.

K-maps can also be useful in detecting these hazards.

- Plot the stable 1-sets like i-cubes.
- Plot the unstable 1-sets as boundaries that cannot be crossed.

The two examples are shown as cells connected by the double-headed arrow.

Note how the static logic 1-hazard can be fixed by adding xyz to the network and therefore creating a stable 1-set for both input patterns.



0-sets can be defined and used for the same procedure.

They are based on a product-of-sums representation of the function \mathcal{F} and can be produced easily by working with the dual of \mathcal{F} .

So we can summarize the static hazard situation in the following way assuming that an appropriate transition between minterms or maxterms exists:

Multilevel networks

	static 0	static 1
0-sets	active stable set missing	active unstable set present
1-sets	active unstable set present	active stable set missing

Two-level networks

	static 0	static 1
AND/OR	none	possible
OR/AND	possible	none

Logic Simulation approaches tend to be used
in practice for large complex circuits

Two input patterns:

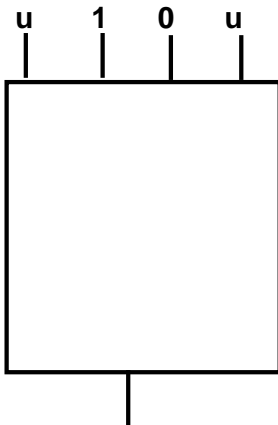
(x_1, x_2, \dots, x_n)

(y_1, y_2, \dots, y_n)

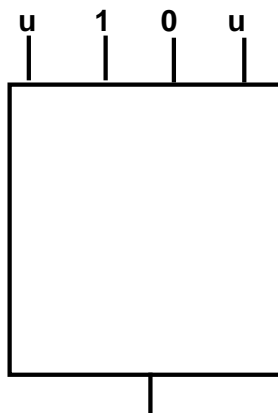
suppose they differ in
a set of bits. set those
positions to u (undetermined)
Set the rest to the common
constant value of 0 or 1

We therefore have some pattern
of 1's 0's and u's. suppose $n = 4$
then we might have (u 1 0 u)

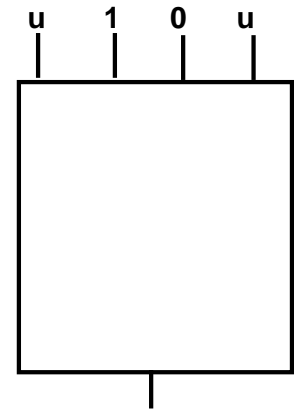
Simulate to determine the output given this
input pattern.



1
no static hazard
possible



0
no static hazard
possible



u
static hazard possible
not necessary though

Can be modified for dynamic hazards
and in general, very complex simulation
techniques can be developed to handle the
effects of timing on a combinational circuits