LAB 4: TIMING HAZARDS

This lab will explore the effects that nonideal time delays on gate outputs, the nature of the "glitches" that result in the output, methods for mitigating these hazards, and how/why these methods work, revisiting the particle accelerator case study in the previous lab, Lab 3. In particular, we will look at a couple of modules that allow key functions in the accelerator to either proceed when users want them to, and prevent them otherwise. Timing hazards pose unique issues in maintaining this control, and as such must be detected and resolved.

Going forward, *assume a delay of one unit of time for every gate.*

(As in past labs, please use Logisim-evolution or CircuitVerse to implement and simulate circuits.)

Part 1: Beam Safety Module

The particle accelerator operates by firing a beam of particles into a test area. Conditions in the detector are assessed, and the detector, in turn, provides a 4-bit input C (with C_3 as the MSB, and C_0 as the LSB). Based on these conditions, the Beam Safety Module assesses whether it is safe to activate the beam, returning an output S, which will be 0 if it is unsafe to activate, and 1 if it is, immediately activating the beam.

Given the nature of the Beam Safety Module's role, it is absolutely vital to ensure that the Beam Safety Module does not return 1 (however briefly) when it should return 0, as it will activate the beam when it is unsafe.

Before the BSM is improved, a total redesign has been ordered. The original design (Fig. 1) was designed at a time of greater scarcity, when the group only had NOR gates. ***Determine the function $S(C) = S(C_3C_2C_1C_0)$ as a POS. Then, design and implement an AND-OR circuit that realizes this function.***

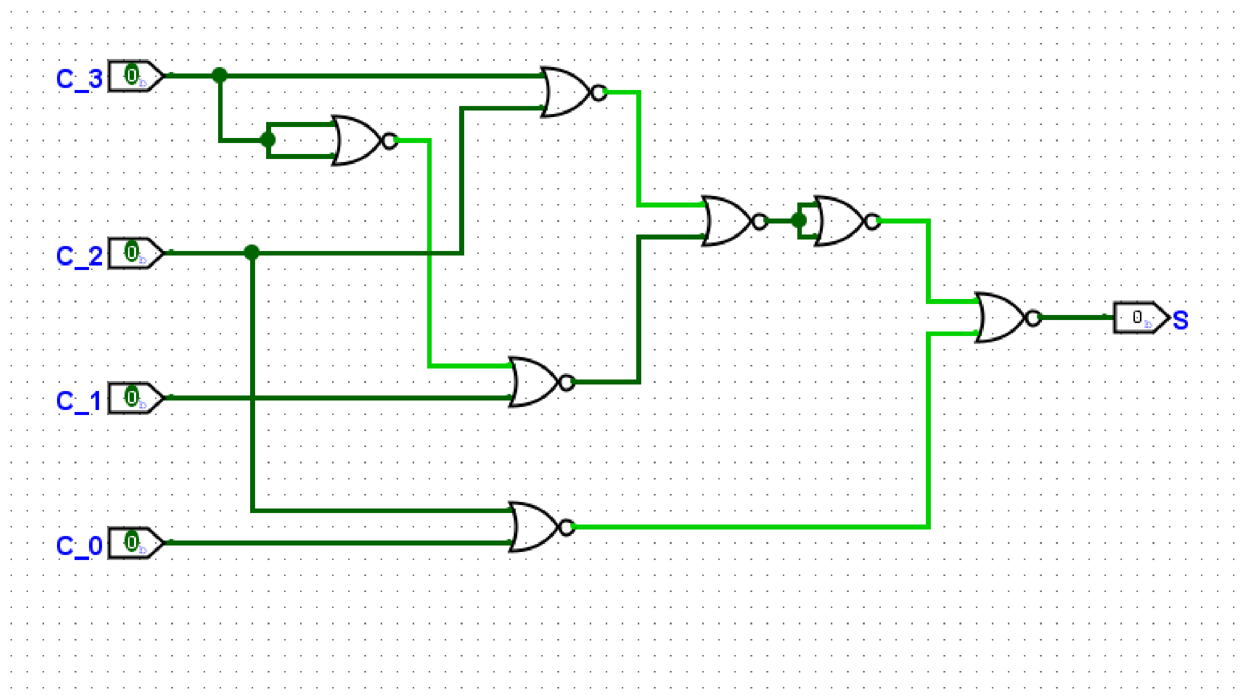

Figure 1. The original design of the BSM, implemented entirely using NOT gates.

With the Beam Safety Module redesign, you now have to contend with potential glitches. For this, you can make use of the POS version of the Consensus Theorem:

$$(A+B)(\overline{A}+C) = (A+B)(\overline{A}+C)(B+C) \quad \textbf{(1)}$$

First, test and determine any hazardous input transitions. For each hazardous transition, use the Consensus Theorem to determine a consensus term. Using these consensus terms, determine a hazard-free expression for S and modify your redesign of the BSM accordingly.

[Solution:]

For the first part of the problem, students should determine that:

$$S = (C_3 + C_2)(C_3' + C_1)(C_2 + C_0)$$

Then, you get the circuit in <ECE_Project_4_BSM_Hazard> in the GitHub repository.

The first two terms in S, $(C_3 + C_2)$ and $(C_3' + C_1)$, contribute a hazard in the C_3 transition ($C_3C_2C_1C_0$: $0001 \rightarrow 1001$), and it can be resolved using the consensus term $(C_2 + C_1)$.

Then, this fix can be easily implemented, as in <ECE_Project_4_BSM_HazardAndConsensus>.

On a K-map, one can see the consensus term S_consensus coverage overlapping with that of the other maxterms S1, S2, and S3:

C_3C_2 → C_1C_0 ↓	**00**	**01**	**11**	**10**
00	**0 (*,&,%)**	**-**	**0 (~)**	**0 (~,&,%)**
01	**0 (*,&)**	**-**	**0 (~)**	**0 (~,&)**
11	**0 (*)**	**-**	**-**	**-**
10	**0 (*,%)**	**-**	**-**	**0 (%)**

S1 (*) = C3+C2; S2 (~)= (C3'+C1); S3 (%) = C2+C0; S_consensus (&) = C2 + C1

Of course, additional hazards can certainly be introduced in the transitions of other variables; for example, if an additional sum is tacked on to the product in the POS, like:

$(C_0' + L(C_3,\ C_2,\ C_1))$ where L is a function of the other bits of the input C.

Then this term will form a hazardous pair with S3 = $(C_2 + C_0)$, creating a potential hazard in the transition of C_0, and so you may or may not have to introduce a consensus term of the form $(C_2 + L)$. A similar procedure can be used to introduce a potential hazard in the C_2 transition.

[end of solution]

Part 2: Data Writing Module

The Data Writing Module (DWM) will come in two variations: "mini" version (or a mini DWM), and the regular version (just DWM). The "mini" version takes a 4-bit input from the detector, D (with D_3 as the MSB and D_0 as the LSB), representing the condition of the data being received. Based on these conditions, the mini DWM will output a value F will be 0 if the users want to "write" the data generated, and 0 if the data should be tossed.

Just like with the BSM, the DWM must not glitch to 0 (however briefly) when the output should be 1, as this will result in the loss of data sought out by the user, and thus a lower quality analysis.

F is naturally a function of the input D:

$F(D) = F(D_0, D_1, D_2, D_3) = (D_2'D_0') + (D_3D_2D_1') + (D_3D_1D_0)$ **(2)**

Design and implement an AND-OR circuit to realize the function F.

Does this implementation of the DWM have any timing hazards? If so, determine what input transitions cause the hazard, what hazardous pairs exist in the current SOP of F, and what consensus terms can be added to F from said hazardous pairs.

Afterward, modify your original AND-OR circuit for F to remove the hazards found.

[Solution:]

Let's first define the three terms in the original F. From there, we can define the three consensus terms that would emerge from the three hazardous pairs that can be determined:

Label	Boolean expression	Comments
F_1	$(D_2'D_0')$	
F_2	$(D_3D_2D_1')$	
F_3	$(D_3D_1D_0)$	
$F_{C,1}$	$(D_3D_1'D_0')$	Will later correspond to "C_1" in the solution circuit; consensus of F_1 and F_2
$F_{C,2}$	$(D_3D_2D_0)$	Will later correspond to "C_2" in the solution circuit; consensus of F_2 and F_3
$F_{C,3}$	$(D_3D_2'D_1)$	Will later correspond to "C_3" in the solution circuit; consensus of F_3 and F_1

The overlap between the terms is more obvious on a K-map:

$D_3D_2 \rightarrow$ $D_1D_0 \downarrow$	**00**	**01**	**11**	**10**
00	1	-	1 (*)	1 (*)
01	-	-	1 (%)	-
11	-	-	1 (%)	1 (&)
10	1	-	-	1 (&)

F1 (blue) ; F2 (green) ; F3 (yellow); F_C,1 (*); F_C,2 (%); F_C,3 (&)

Then $F = F_1 + F_2 + F_3 \rightarrow F_1 + F_2 + F_3 + F_{C,1} + F_{C,2} + F_{C,3}$, a hazard-free expression, and the implementation in <ECE_Project_4_DWM> can be modified into < ECE_Project_4_DWM_Consensus> to remove the hazards caused by the input transitions:

$$D_3D_2D_1D_0\colon\ 1100 \rightarrow 1000;\ 1111 \rightarrow 1101;\ 1011 \rightarrow 1010$$

[end of solution]