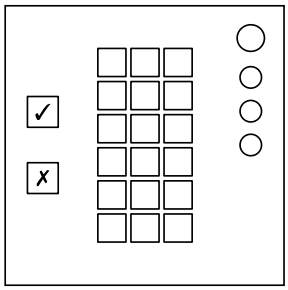
On the Subject of Game Changer



Wait... How many stages?! I thought you said there were only two to four!

This module consists of a 6 row, 3 column grid, a binary counter of 3 LEDs, a green submit button, and a red reset button.

To solve the module, solve the 6 row, 3 column Game of Life grid for each iteration. The rules for this game are different to the normal Game of Life.

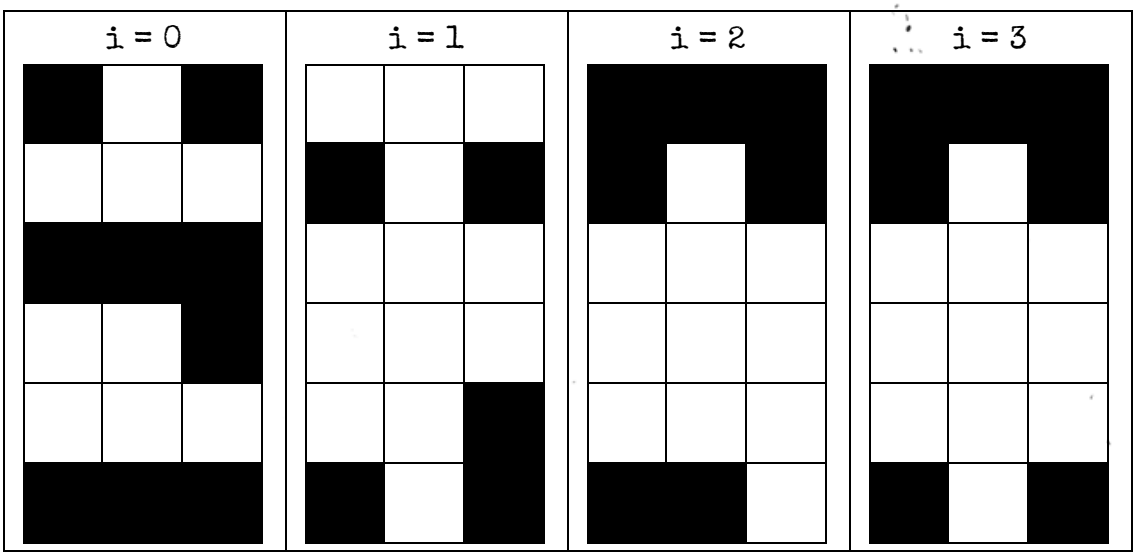
Using the current board and the determined conditions, iterate the displayed board using those conditions to obtain the next iteration to submit. Upon submitting the correct iteration, the 3 LEDs will start counting up in binary how many iterations were correctly submitted, and you then must the use correct iteration to submit the next iteration from the current iteration. For clarity, the initial board display is iteration 0, iteration 1 is the first board to submit after determining the conditions from the initial board.

For each iteration to submit:

1. Take the first 9 tiles in reading order from the current iteration. Ordered from 0 - 8 inclusive, tiles that are white will correspond to how many white tiles must surround a tile to turn a black tile into a white tile.
2. Then take the first 9 tiles in **reverse** reading order from the current iteration. Ordered from 0 - 8 inclusive, tiles that are white will correspond to how many white tiles must surround a tile to keep a tile white.
3. Apply these rules to the current board.

The module will solve when 8 iterations have been submitted correctly or when the next correctly submitted iteration is exactly the same as any previous iteration, including the initial state. Alternatively, the user may interact with the status light to make the module solve on the next correct submitted iteration. The status light may only be safely interacted if the user submitted at least 3 iterations correctly. Attempting to interact with the status light before correctly submitting 3 iterations may cause a strike.

Iterating Game Changer: Example



In this example, the filled in tile represents a dead tile, and the unfilled tile represent an alive tile for each 6 row, 3 column grid provided.

For each displayed iteration (abbreviated with "i=#"), to obtain the next iteration, the displayed iterations have the following rules to apply on the current iteration for each alive tile:

0. Birth on 1, 3, 4, 5 surrounding alive tiles, Survival on 3, 4, 5, 7, 8 surrounding alive tiles.
1. Birth on 0, 1, 2, 4, 6, 7, 8 surrounding alive tiles, Survival on 1, 4, 5, 6, 7, 8 surrounding alive tiles.
2. Birth on 4, 6, 7, 8 surrounding alive tiles, Survival on 0, 3, 4, 5, 6, 7, 8 surrounding alive tiles.
3. Birth on 4, 6, 7, 8 surrounding alive tiles, Survival on 1, 3, 4, 5, 6, 7, 8 surrounding alive tiles. (The next iteration is stable.)