

Co-design

Group Project C

“The Game of Mancala”

by Dmitry Vashurin, Tulegenova Karina & Klim Bagryantsev

17/05/2022

INTRODUCTION

The Mancala game is a two-player turn-based strategy board game played with small objects and rows of pits. The objective is usually to capture all or some amount of the opponent's pieces. Mancala has many variations, which differ from region to region. Our implementation refers to a version invented by William Julius Champion, USA.



\*example of a Mancala game board

RULES OF THE GAME

Our variation considers six small pits on each side and a big pit, called mancala, at each end. Each player controls the pits and their seeds on his side of the board. The object of the game is to get more seeds to your mancala than your opponent. At the beginning of the game, allpits have four seeds and players’ mancalas are empty.

**Turn**

* During the turn, the player removes all seeds from one of his pits. Moving counter-clockwise, the player drops one seed in each pit next to initial, including the player's mancala but not opponent's one

**Capture**

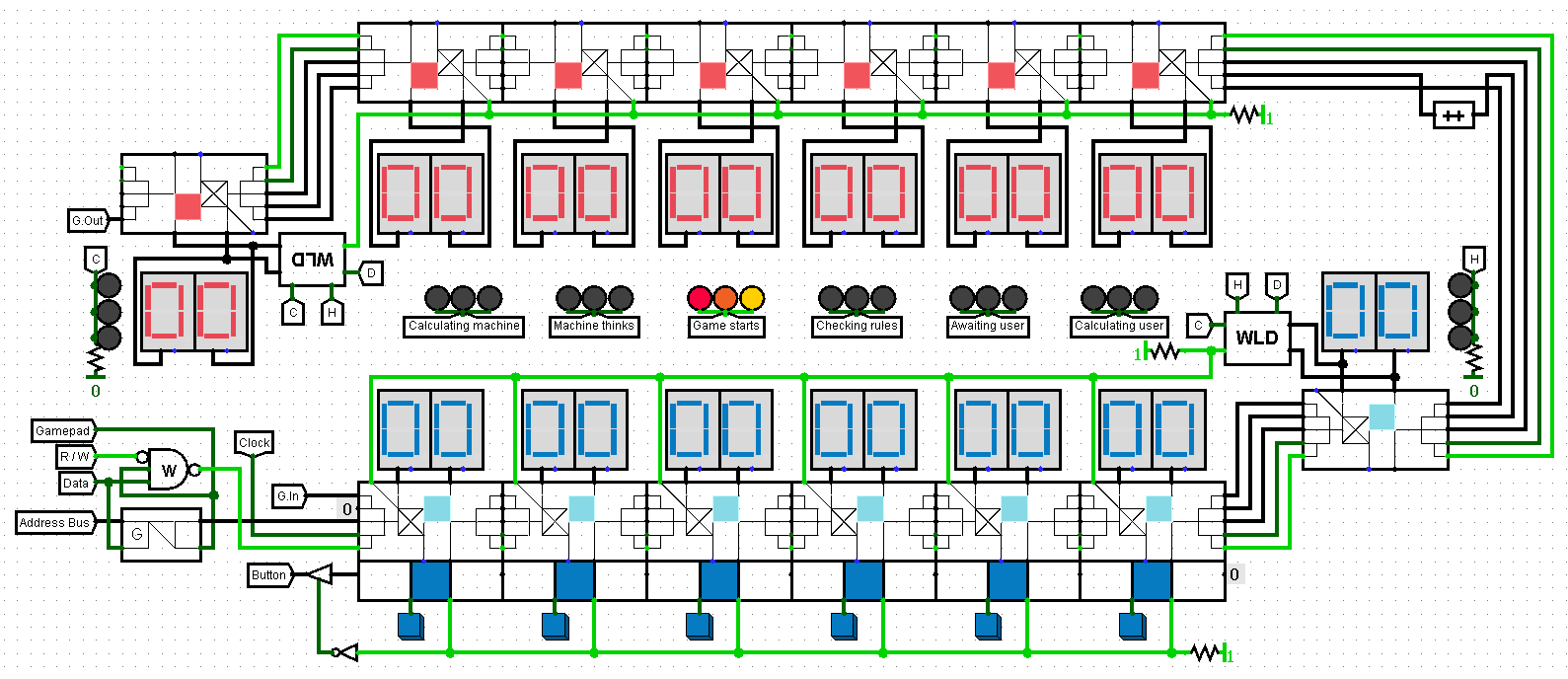
* If the last seed lands in an empty player’s pit and the opposite pit contains seeds, both the last seed and the opposite seeds goes to player’s mancala

**Extra move**

* If the last seed lands in the player's mancala, the player gets an extra move. There is no limit on the moves a player can make in their turn.

When one player no longer has any seeds in any of their houses, the game ends. The other player moves all remaining seeds to their store, and the player with the most seeds in their store wins. The game may result in a draw.

GAME BOARD

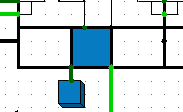
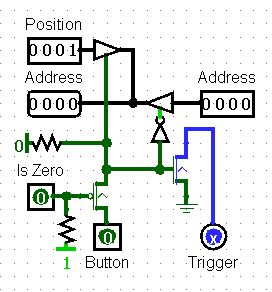


The above circuit represents the structure of a game board implemented in Logisim. All other circuits, described further, are its sub circuits by default.

Six LED lights in in the center of the board denote current game state

|  |  |
| --- | --- |
| Calculating machine | AI player’s move processing |
| Machine thinks | AI player calculates the best move (see Computer ROM) |
| Game starts | Filling pits with seeds (see Initialization ROM) |
| Checking rules | see Referee ROM |
| Awaiting user | The game pauses till the player makes his move |
| Calculating user | Player’s move processing |

BUTTON DRIVER



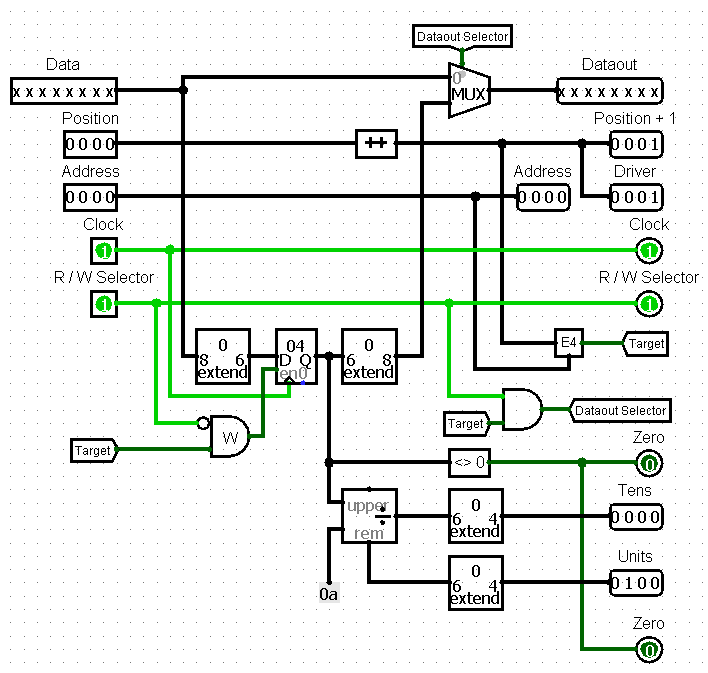
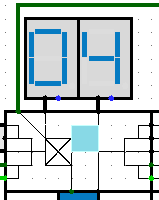
**BUTTON DRIVER ON GAME PAD**

**BUTTON DRIVER CIRCUIT**

|  |  |
| --- | --- |
| TRIGGER | Checks that only one button is pushed |
| IS ZERO | Checks whether the pit has seeds or not |
| BUTTON | Signal from button trigger |
| ADDRESS (INPUT) | Gets an address from the previous button (0000 by default) |
| POSITION | Gets the position of the pit |
| ADDRESS (OUTPUT) | Passes the address of the pit to the next button driver |

If the button is pushed, it makes all necessary checks and then decides whether to pass the address input from the previous button driver or to pass an address of pit, where the button was pushed

BUTTON REGISTER & DISPLAY DRIVER



**BUTTON REGISTER ON GAME BOARD**

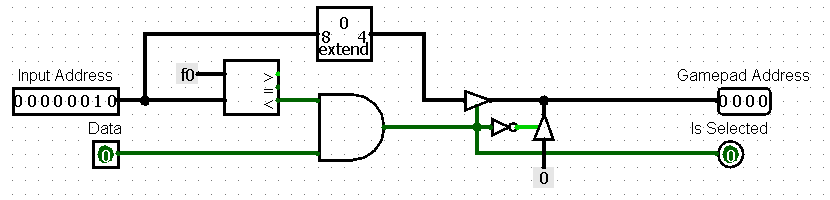
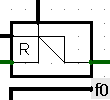
**BUTTON REGISTER CIRCUIT**

|  |  |
| --- | --- |
| DATA | Gets the number of seeds passed from the previous button register |
| DATAOUT | Passes the number of seeds to the next button register |
| POSITION (INPUT/OUTPUT) | Gets the position of the current pit and passes its increased value further |
| ADDRESS (INPUT/OUTPUT) | Gets and passes the address of the pit from the Gamepad address manager |
| R/W SELECTOR | Read/Write mode |
| DRIVER | Passes the position of the pit to the button driver |
| TENS/UNITS | Amount of seeds in current register separated into digits for displays |

When the position is equal to address, while the selector is in Write mode, given data gets stored in Tens/Units of the current register. Otherwise, if the selector is in Read mode register data from Tens/Units passes to the next button register ending up in G.out

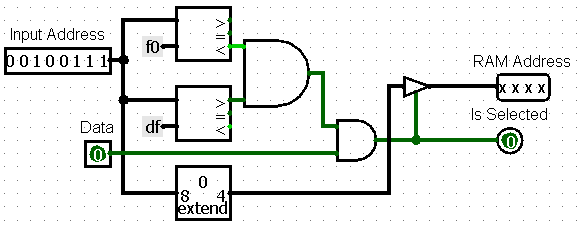
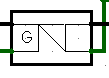
In other cases data transfers through current register onto the next one

RAM & GAMEPAD ADDRESS MANAGER



**GAM ON GAME BOARD**

**GAMEPAD ADDRESS MANAGER CIRCUIT**

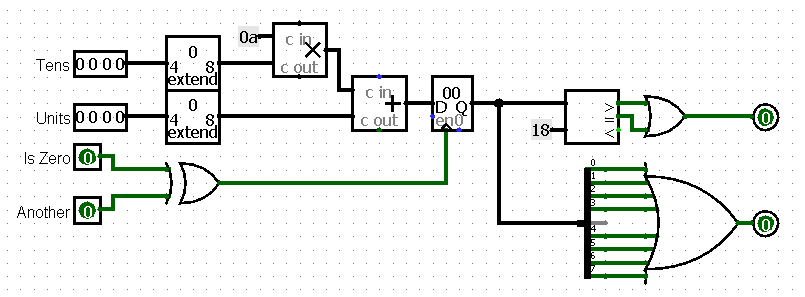
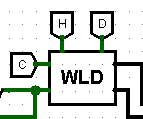


**RAM ON GAME BOARD**

**RAM ADDRESS MANAGER CIRCUIT**

Converts 8-digit input address into a 4-digit output address for Gamepad/ROM

WIN-LOSE-DRAW CHIP

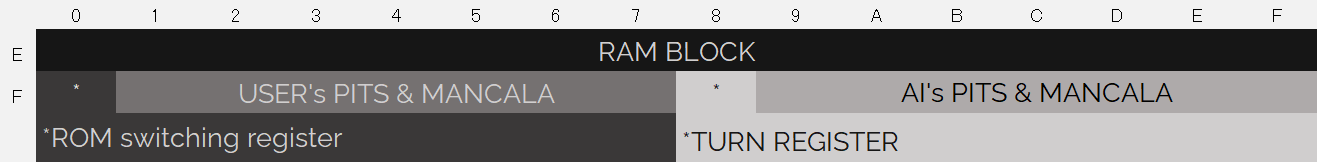


**WLD CHIP ON GAME BOARD**

**WLD CHIP CIRCUIT**

Gets the amount of seeds in player’s mancala and judges on the amount of seeds who is the winner. It is obvious that more than a half of stones in one’s mancala guarantees that the player has won. It also checks whether the player’s pits are empty or not.

DATA STRUCTURE



The game reads and writes the pits’ data to these memory blocks, so that the Computers and Player’s ROMs can make proper decisions.

RAM block contains data about the last move. I.e. the pit where the move ended up, whom of the players has made the last move and other miscellaneous information.

ROM switching register is suited for “jumper” template, which switches different ROM blocks.

Turn register’s object is to store information about player, who must make next move.

ROM BANKS STRUCTURE

INITIALIZATION BANK

Initializes the values in the players’ pits and switches to a Human ROM bank to let the player make his first move.

REFEREE BANK

Executes the capture rule and the extra move rule. Judges who will be the next to make a turn and switches to a proper player’s bank. Checks whether all pits are empty or not.

HUMAN BANK

Processes player’s move: executes sowing, stores the data to RAM block and switches to the Referee Bank

COMPUTER BANK

One of the computer bank’s tasks is to calculate the best move and because it is hard to implement some extraordinary AI in Assembly code -> our realization just checks whether it is possible to get an extra move or to capture opponent’s seeds. Otherwise, it just makes any possible move.

The other part of the bank simply executes AI’s turn according to the rules of the game the same way Human bank does.

PITS OPERATION SYSTEM

All necessary calculations are executed in ROM blocks, so that the game board is in fact consists of a bus for reading and writing necessary data into button registers and 7-segment displays to represent the pit’s values. It allows us to simplify the circuit’s structure.