FUNDAMENDTAL OF DIGITAL SYSTEM FINAL PROJECT REPORT
DEPARTMENT OF ELECTRICAL ENGINEERING
UNIVERSITAS INDONESIA

THE "GAME OF LIFE" PHYSICS ENGINE

KELOMPOK 6

Djukallita Tafiana Djoewarsa	**2406416573**
Muhamad Rifqi Fadil Itsnain	**2406355306**
Naziehan Labieb	**2406487102**
Soraya Azzizah Pahlevi	**2406487001**

PREFACE

Puji syukur kami panjatkan ke hadirat Tuhan Yang Maha Esa atas segala rahmat dan karunia-Nya sehingga kami dapat menyelesaikan laporan proyek akhir ini. Laporan ini disusun sebagai salah satu bentuk pemenuhan tugas proyek akhir pada mata kuliah Perancangan Sistem Digital Tahun Ajaran 2025, dengan judul "The 'Game of Life' Physics Engine". Proyek ini menggunakan bahasa pemrograman VHDL (VHSIC Hardware Description Language) untuk mengimplementasikan hardware accelerator dalam simulasi cellular automata (Conway's Game of Life).

Kami menyampaikan terima kasih kepada para asisten laboratorium, khususnya asisten pendamping kelompok 6, atas bimbingan, bantuan, dan kesabaran mereka selama proses implementasi serta pengujian sistem. Melalui kerja sama tim yang solid, kami dapat memperdalam pemahaman mengenai konsep dan penerapan teori dalam perancangan sistem digital komputer.

Kami menyadari bahwa laporan dan proyek ini masih memiliki berbagai keterbatasan. Oleh karena itu, kritik dan saran yang membangun sangat kami harapkan untuk perbaikan di masa mendatang.

Depok, December 6, 2025

Group PSD 6

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

Seiring dengan pesatnya perkembangan teknologi digital, kebutuhan akan sistem komputasi yang mampu bekerja secara cepat dan efisien semakin meningkat. Salah satu konsep yang banyak dimanfaatkan dalam bidang komputasi modern adalah paralelisme, yaitu kemampuan untuk menjalankan banyak proses secara bersamaan. Salah satu model sederhana namun sangat efektif untuk mempelajari perilaku sistem paralel adalah cellular automata, dengan Conway's Game of Life sebagai contoh yang paling populer dan banyak digunakan dalam pembelajaran serta penelitian.

Proyek "Game of Life" Physics Engine ini dirancang untuk merealisasikan simulasi Game of Life secara langsung di dalam perangkat keras. Sistem menggunakan grid 8×8 yang terdiri dari sel-sel yang bekerja secara paralel, sehingga setiap sel dapat memperbarui status hidup atau matinya pada setiap siklus clock secara simultan. Pendekatan ini memberikan gambaran nyata mengenai bagaimana prinsip paralelisme dapat diterapkan dalam desain rangkaian digital berbasis VHDL.

Struktur utama sistem meliputi modul sel yang menentukan perkembangan suatu sel berdasarkan kondisi sel-sel tetangganya, serta sebuah pengendali sederhana yang mengatur jalannya simulasi. Pengendali ini memiliki mode untuk memuat pola awal serta menjalankan evolusi otomatis sesuai aturan Game of Life. Selain itu, proses verifikasi dilakukan menggunakan testbench sehingga pola awal maupun hasil simulasi dapat diuji dan dievaluasi secara terstruktur.

Melalui proyek ini, diharapkan mahasiswa dapat memahami keterkaitan antara konsep teoritis cellular automata, prinsip paralelisme, dan implementasi sistem digital. Proyek ini juga memberikan pengalaman langsung dalam merancang, menguji, dan menganalisis sebuah sistem komputasi berbasis hardware yang bekerja secara paralel.

1.2 PROJECT DESCRIPTION

Proyek “Game of Life” Physics Engine bertujuan untuk membangun sebuah simulasi Conway’s Game of Life yang berjalan sepenuhnya di hardware. Setiap sel pada grid 8×8 diimplementasikan sebagai modul logika mandiri yang bekerja secara paralel, sehingga pada setiap siklus clock terdapat 64 proses yang berjalan bersamaan. Pendekatan ini menunjukkan bagaimana konsep paralelisme dapat diterapkan secara nyata menggunakan desain digital berbasis VHDL.

Komponen utama dari sistem ini adalah Cell_Core, yaitu modul yang menentukan status hidup atau mati sebuah sel berdasarkan kondisi delapan tetangganya. Logika aturan Game of Life seperti underpopulation, overpopulation, dan reproduction yang diimplementasikan menggunakan kombinasi fungsi dan aritmetika digital, sedangkan pembaruan status dilakukan secara sinkron pada rising edge. Grid 8×8 disusun menggunakan *generate loop*, lengkap dengan topologi toroidal agar sel di tepi tetap memiliki tetangga yang konsisten.

Untuk mengatur jalannya simulasi, sistem dilengkapi dengan controller berbasis FSM yang memiliki mode IDLE, EDIT_MODE, dan RUN_SIMULATION. Pada EDIT_MODE, pola awal seperti glider dimuat dari ROM sederhana, lalu pada RUN_SIMULATION sistem menjalankan evolusi otomatis sesuai aturan Game of Life. Proses verifikasi dilakukan menggunakan testbench dengan file I/O, sehingga pola awal dapat dibaca dari file dan setiap frame hasil simulasi dapat diekspor untuk dianalisis atau divisualisasikan lebih lanjut.

1.3 OBJECTIVES

Tujuan dari proyek yaitu sebagai berikut:

1. Merancang dan mengimplementasikan simulasi Conway’s Game of Life yang berjalan sepenuhnya pada perangkat keras digital menggunakan bahasa VHDL.
2. Menerapkan konsep Massively Parallel Processing melalui grid 8×8, sehingga seluruh 64 sel dapat memperbarui statusnya secara simultan dalam satu siklus clock.

3. Mengembangkan modul Cell_Core yang mampu menentukan status hidup atau mati sebuah sel berdasarkan aturan Game of Life dengan memanfaatkan kombinasi logika dan aritmetika digital.
4. Mewujudkan struktur grid dengan topologi toroidal, sehingga sel pada tepi tetap memiliki tetangga yang konsisten dan simulasi dapat berjalan tanpa batasan tepi.
5. Merancang dan mengimplementasikan controller berbasis FSM dan microcode yang bertugas memuat pola awal serta mengatur alur eksekusi simulasi, termasuk mode IDLE, EDIT_MODE, dan RUN_SIMULATION.
6. Melakukan verifikasi sistem melalui testbench dan file I/O, sehingga pola awal dapat diimpor dan hasil simulasi dapat diekspor untuk kebutuhan analisis maupun visualisasi frame-by-frame.
7. Memberikan demonstrasi nyata mengenai penerapan konsep parallelism, cellular automata, dan hardware acceleration dalam sebuah sistem digital yang terstruktur dan efisien.

1.4 ROLES AND RESPONSIBILITIES

Peran dan tanggung jawab yang diberikan kepada anggota kelompok adalah sebagai berikut:

Roles	Responsibilities	Person
Ketua Kelompok	- Memastikan pembagian tugas dan berkas sesuai - membuat code, laporan, dan ppt sesuai bagian	Soraya Azzizah Pahlevi
Anggota kelompok dan Pencetus ide	- Mengidekan proyek dan overall logika codenya itu sendiri	Naziehan Labieb

	- membuat code, laporan, dan ppt sesuai bagian	
Anggota kelompok	- membuat code, laporan, dan ppt sesuai bagian	Djukallita Tafiana Djoewarsa
Anggota kelompok	- membuat code, laporan, dan ppt sesuai bagian	Muhamad Rifqi Fadil Itsnain

Table 1. Roles and Responsibilities

CHAPTER 2

IMPLEMENTATION

2.1 EQUIPMENT

Alat-alat yang akan digunakan dalam proyek ini adalah sebagai berikut:

- Software:
    - **o** Xilinx Vivado (2022.2 atau yang lebih baru): IDE (Integrated Development Environment) utama yang digunakan untuk menulis kode VHDL, melakukan sintesis desain, dan menjalankan simulasi behavioral.
    - **o** Visual Studio Code: Digunakan sebagai editor eksternal untuk melihat dan menganalisis file output berbasis teks (output_game.txt) yang dihasilkan oleh testbench.
    - ○ Github: Digunakan sebagai platform repository untuk mempermudah kolaborasi tim, melacak perubahan kode, serta memastikan seluruh anggota dapat bekerja pada proyek secara terstruktur dan sinkron.

2.2 IMPLEMENTATION

Dalam pengerjaan Proyek Akhir Perancangan Sistem Digital ini, kami menerapkan konsep-konsep dasar VHDL sebagai berikut:

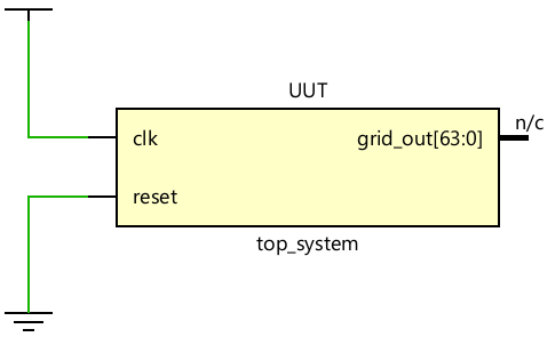


Fig 1. Schematic

2.2.1 Dataflow Style Programming in VHDL

Pendekatan Dataflow digunakan dalam entitas **cell_core** untuk melakukan operasi aritmetika dan logika secara konkuren. Hal ini diterapkan pada proses penjumlahan jumlah tetangga yang hidup (logic '1') dari vektor input neighbors. Selain itu, logika komparator untuk menentukan aturan survival (misalnya: jika tetangga < 2 atau > 3 maka mati) juga diimplementasikan menggunakan aliran data sinyal tanpa memerlukan state machine yang kompleks di dalam fungsi logika sel.

2.2.1 Behavioral Style Programming in VHDL

Behavioral Style digunakan secara ekstensif untuk mendeskripsikan perilaku sekuensial sistem yang bergantung pada sinyal clock. Penerapan utamanya terdapat pada:

- Register Sel (**cell_core**): Proses process(clk) digunakan untuk memperbarui status sel (**state_reg**) hanya pada saat rising edge clock, memastikan sinkronisasi seluruh grid.
- Logika Prioritas: Di dalam proses behavioral, kami menerapkan logika if-else untuk memberikan prioritas pada sinyal tulis dari CPU (we) dibandingkan aturan permainan, sehingga inisialisasi pola dapat dilakukan.

2.2.3 Testbench and File I/O

Berbeda dengan testbench standar yang hanya mengandalkan waveform, proyek ini menggunakan fitur TextIO (std.textio dan ieee.std_logic_textio) pada **tb_top_system**. Testbench dirancang untuk tidak hanya memberikan stimulus clock dan reset, tetapi juga menangkap (capture) output grid 64-bit pada setiap siklus clock dan menuliskannya ke dalam file eksternal **output_game.txt**. Hal ini memungkinkan verifikasi visual terhadap animasi pergerakan sel yang sulit diamati hanya melalui gelombang sinyal.

2.2.4 Structural Style Programming in VHDL

Gaya pemrograman struktural digunakan pada Top-Level Entity (**top_system**) dan Grid (**game_grid_8x8**) untuk menyusun sistem secara hierarkis.

- Pada top_system, kami menghubungkan instansi game_controller (CPU) dan **game_grid_8x8** menggunakan sinyal internal (**w_we**, **w_row**, **w_col**) melalui teknik Port Mapping.

- Pendekatan ini memisahkan logika kontrol dan logika komputasi, sehingga memudahkan proses debugging dan integrasi sistem.

2.2.5 Procedure and Function

Untuk menjaga kebersihan dan modularitas kode, logika aturan permainan Conway's Game of Life dienkapsulasi ke dalam sebuah fungsi murni (pure function) bernama **get_next_state** di dalam **cell_core**. Fungsi ini menerima parameter jumlah tetangga dan status saat ini, lalu mengembalikan status berikutnya (Hidup/Mati). Penggunaan fungsi ini menyederhanakan blok proses utama dan menghindari pengulangan kode logika if-else yang rumit.

2.2.6 Looping Construct (For Generate)

Looping Construct adalah inti dari arsitektur paralel proyek ini. Kami menggunakan pernyataan FOR...GENERATE bersarang (nested loop) di dalam game_grid_8x8 untuk menginstansiasi 64 unit **cell_core** secara otomatis. Selain itu, looping juga digunakan untuk menangani pengkabelan Toroidal (dunia tanpa ujung), di mana indeks tetangga dihitung menggunakan aritmatika modulo di dalam loop generasi untuk menghubungkan sel tepi kiri dengan kanan dan atas dengan bawah.

2.2.7 Finite State Machine (FSM)

FSM diimplementasikan di dalam **game_controller** untuk mengatur alur operasi sistem. Terdapat tiga state utama:

- RESET_STATE: Inisialisasi register dan sinyal.
- WRITE_STATE: CPU aktif mengambil data dari ROM dan menulis pola awal ke Grid.
- RUN_STATE: CPU non-aktif dan memberikan sinyal enable ke Grid untuk memulai simulasi mandiri.

Current State	Input / Condition	Next State	Output Control Signals	Deskripsi Aktivitas
RESET_STATE	Reset = 1	RESET_STATE	grid_we = 0 sim_running = 0	Inisialisasi sistem. Menunggu reset

			pc = 0	dilepas.
RESET_STATE	Reset = 0	WRITE_STATE	grid_we = 0 sim_running = 0	Transisi otomatis untuk memulai proses penulisan pola.
WRITE_STATE	is_last = 0	WRITE_STATE	grid_we = 1 sim_running = 0	Sedang Menggambar. CPU mengambil instruksi dari ROM, menulis ke Grid, dan lanjut ke instruksi berikutnya.
WRITE_STATE	is_last = 1	RUN_STATE	grid_we = 1 sim_running = 0	Instruksi Terakhir. Menulis data terakhir ke Grid dan bersiap pindah mode.
RUN_STATE	- (Always)	RUN_STATE	grid_we = 0 sim_running = 1	Simulasi Berjalan. CPU berhenti (idle), Grid diaktifkan (sim_running=1), simulasi Game of Life berjalan mandiri.

Table 1. FSM

2.2.8 Microprogramming

Teknik microprogramming diterapkan pada **game_controller** untuk memberikan fleksibilitas dalam menentukan pola awal. Kami membuat sebuah ROM (Read-Only

Memory) sederhana yang berisi instruksi mikro (koordinat baris, kolom, dan data). Controller bertindak sebagai sequencer yang membaca instruksi ini baris demi baris untuk "menggambar" pola (seperti Glider) ke dalam memori Grid sebelum simulasi dijalankan.

CHAPTER 3

TESTING AND ANALYSIS

3.1 TESTING

Untuk mengetahui kinerja dan memastikan bahwa sistem Game of Life ini bekerja sesuai dengan tujuan perancangan, perlu adanya pengujian menggunakan testbench pada setiap tahap implementasi. Pengujian ini dirancang untuk mensimulasikan berbagai kondisi yang mungkin muncul selama proses evolusi sel, mulai dari pembaruan state pada level sel tunggal hingga respons sistem ketika menjalankan pola awal secara penuh. Melalui tahapan ini, setiap komponen, baik yang kombinasional maupun sekuensial, dapat diamati perilakunya terhadap sinyal clock, reset, serta input yang diberikan. Pendekatan bertahap ini memastikan bahwa seluruh modul telah berfungsi dengan benar sebelum digabungkan ke dalam sistem lengkap.

Tahap pertama adalah pengujian cell_core dimana ttestbench memberikan berbagai kombinasi status tetangga untuk melihat apakah sel menentukan keadaan berikutnya sesuai aturan Game of Life. Pengujian ini mencakup kondisi sel mati atau hidup dengan jumlah tetangga berbeda sehingga logika dasar dapat divalidasi dahulu. Lalu ada pengujian modul game_grid_8x8 yang berfokus pada koneksi antar sel dan mekanisme tetangga toroidal. Testbench akan memeriksa apakah setiap sel menerima delapan tetangga yang benar serta apakah seluruh grid memperbarui state secara serempak pada setiap clock cycle. Selanjutnya ada pengujian game_controller yang bertugas mengatur mode edit dan simulasi. Instruksi seperti SET, CLEAR, STEP, dan RUN diuji untuk memastikan controller dapat memodifikasi isi grid dan menjalankan evolusi pola dengan benar. Terakhir, pengujian integrasi dilakukan melalui modul top_system yang dimana pola awal dibaca dari file eksternal dan hasil perubahan grid dicatat kembali ke file output. Proses ini memastikan bahwa seluruh modul dapat bekerja bersama sebagai satu sistem yang utuh.

3.2 RESULT

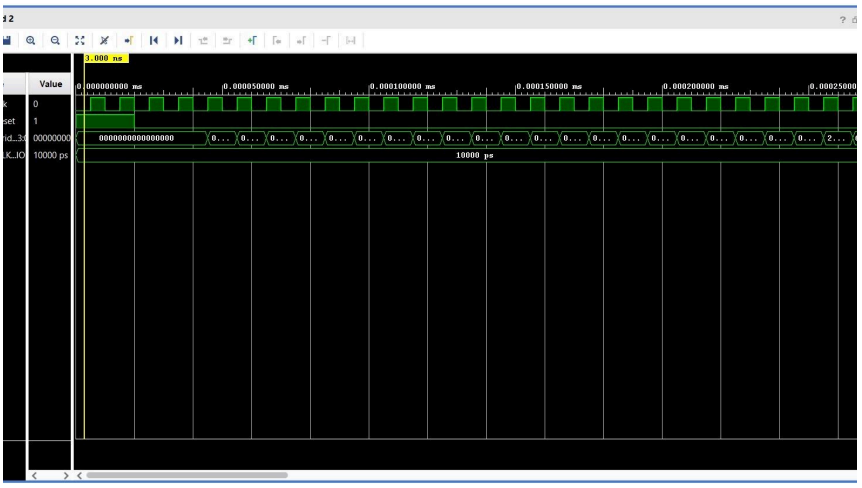


Fig 2. Testbench Result

```
FRAME: 1
00000000
00000010
00001100
00000110
00000000
00000000
00000000
00000000
==================
FRAME: 2
00000000
00000100
00001000
00001110
00000000
00000000
00000000
00000000
==================
FRAME: 3
00000000
00000000
00001010
00001100
00000100
00000000
00000000
00000000
==================
FRAME: 4
00000000
00000000
00001000
00001010
00001100
00000000
00000000
```

Fig 3. Output Game Txt Result

3.3 ANALYSIS

Hasil pengujian menunjukkan bahwa sistem Game of Life 8x8 dapat berjalan sesuai dengan fungsi yang dirancang. Dari waveform simulasi, terlihat bahwa sinyal clock bekerja stabil dan konsisten sepanjang proses, sementara sinyal reset berhasil membawa seluruh modul ke kondisi awal tanpa menimbulkan nilai yang tidak terdefinisi. Kondisi ini memastikan bahwa pembaruan state pada setiap sel hanya terjadi pada rising edge clock dan seluruh proses simulasi berjalan secara sinkron.

Pada tahap awal setelah reset, keluaran grid_out berada dalam keadaan nol yang menunjukkan bahwa seluruh sel berada pada kondisi mati. Ketika controller mulai memberikan instruksi dan clock berjalan, nilai grid_out mulai berubah secara bertahap. Perubahan nilai bit pada vektor grid_out menandakan bahwa logika pembaruan state pada modul cell_core telah bekerja, dan setiap sel menentukan state berikutnya berdasarkan jumlah tetangganya. Perubahan tersebut terjadi secara simultan pada seluruh sel, yang memperlihatkan bahwa mekanisme generate dan koneksi tetangga pada modul game_grid_8x8 telah berfungsi dengan benar.

Selain melalui waveform, analisis juga diperkuat dengan output file yang menampilkan evolusi grid pada setiap frame. Pada FRAME 1, pola awal yang diberikan melalui testbench dapat terlihat dengan jelas, membentuk konfigurasi tertentu di area tengah grid. Ketika simulasi berlanjut ke FRAME 2, pola mulai berubah mengikuti aturan Game of Life: beberapa sel mati karena kurang atau lebih tetangga, dan sel baru muncul akibat kondisi reproduksi. Pada FRAME 3 dan FRAME 4, pola mengalami transformasi lanjutan yang konsisten, menunjukkan pergeseran dan stabilisasi pada beberapa bagian grid. Perubahan antar frame tersebut menunjukkan bahwa seluruh sel telah menerima input tetangga yang benar dan logika aturan bekerja sebagaimana mestinya.

Konsistensi antara hasil waveform dan output frame menegaskan bahwa seluruh modul, mulai dari cell_core, game_grid_8x8, hingga game_controller telah bekerja secara terkoordinasi. Tidak ditemukan perubahan nilai yang terjadi di luar siklus clock atau perilaku yang tidak sesuai aturan Game of Life. Maka dari itu, sistem dapat dinyatakan berjalan stabil, akurat, dan memenuhi spesifikasi desain yang telah ditetapkan.

CHAPTER 4

CONCLUSION

Proyek THE “GAME OF LIFE” PHYSICS ENGINE berhasil menunjukkan bagaimana konsep cellular automata dapat dijalankan secara penuh di hardware melalui pendekatan desain digital yang terstruktur dan paralel. Dengan mengimplementasikan setiap sel sebagai modul logika independen, sistem mampu memperbarui seluruh 64 sel dalam satu siklus clock. Hal ini memperlihatkan kekuatan pendekatan paralelisme dalam VHDL dan bagaimana logika yang biasanya dijalankan secara software dapat diwujudkan dalam bentuk rangkaian digital yang nyata.

Melalui proses perancangan, berbagai konsep penting dalam VHDL berhasil diterapkan, seperti logika kombinasi, sequential logic, looping construct (generate), structural hierarchy, fungsi dan procedure, hingga finite state machine sebagai pengendali utama. Topologi toroidal yang diterapkan pada grid juga terbukti efektif dalam menjaga konsistensi tetangga setiap sel, sehingga simulasi dapat berjalan tanpa isu batas tepi. Pengujian melalui testbench serta penggunaan file I/O memperkuat validasi hasil, karena setiap perubahan state grid dapat diamati secara detail melalui file output.

REFERENCES

[1] GeeksforGeeks, “Conway’s Game of Life —- Introduction and Rules,” *GeeksforGeeks.org*, 2023.
Available: https://www.geeksforgeeks.org/conways-game-of-life-introduction/
[Accessed: 06-Dec-2025]

[2] FPGA4Student, “VHDL Tutorial: Basic Logic, Sequential Logic, and Testbench Examples,” *FPGA4Student.com*, 2020.
Available: https://www.fpga4student.com/
[Accessed: 06-Dec-2025]

[3] Nandland, “What is VHDL? Beginner-Friendly Explanation and Examples,” *Nandland.com*, 2023.
Available: https://nandland.com/what-is-vhdl/
[Accessed: 06-Dec-2025]

[4] Logicly, “Understanding Cellular Automata and Conway’s Game of Life,” *Logicly.com*, 2022.
Available: https://logicly.com/educational-resources/cellular-automata/
[Accessed: 06-Dec-2025]

[5] VHDLwhiz, “How the VHDL generate Statement Works,” *VHDLwhiz.com*, 2021.
Available: https://vhdlwhiz.com/generate/
[Accessed: 06-Dec-2025]

[6] VHDL Online, “Finite State Machine in VHDL —- Complete Guide,” *VHDLOnline.com*, 2022.
Available: https://vhdlonline.com/fsm-vhdl/
[Accessed: 06-Dec-2025]

[7] Project F, “FPGA Basics: Simulation, Testbench, and Verification,” *projectf.io*, 2023.
Available: https://projectf.io/posts/fpga-simulation/
[Accessed: 06-Dec-2025]

[8] EEWorldOnline, “Introduction to Microprogramming and Control Units,” *EEWorldOnline.com*, 2021.
Available: https://www.eeworldonline.com/microprogramming-basics/
[Accessed: 06-Dec-2025]

[9] FPGA Tutorial, “TextIO in VHDL — Reading & Writing Files for Simulation,” *FPGATutorial.com*, 2022.
Available: https://fpgatutorial.com/vhdl-textio/
[Accessed: 06-Dec-2025]

APPENDICES

Appendix A: Project Schematic

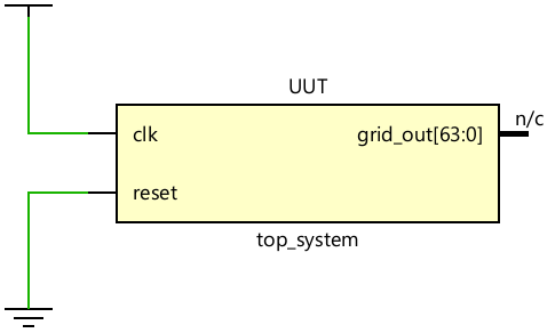


Fig 1. Schematic

Fig 2. Testbench Result

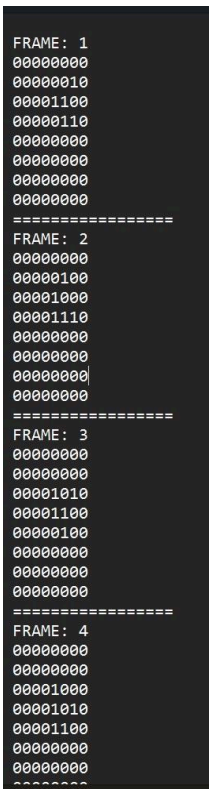
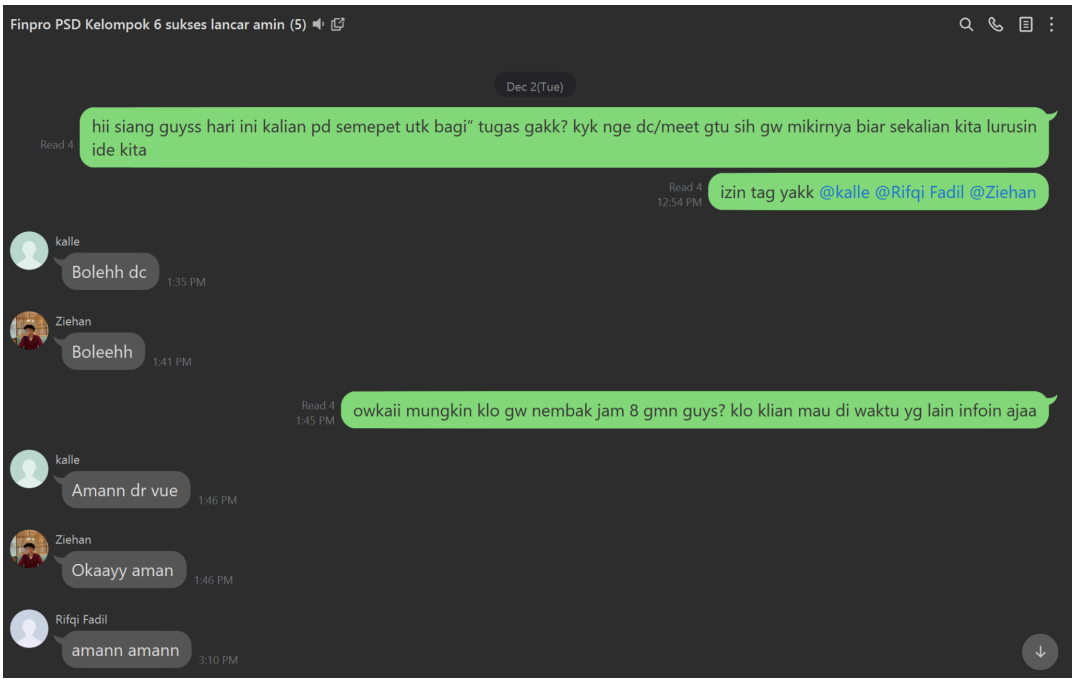


Fig 3. Output Game Txt Result

Appendix B: Documentation



Janjian DC, dokumentai saat DC lupa dilakukan