Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 06, June - 2025

Hardware Design of a Real-Time Pong Game on FPGA with VGA Display Support

Tejeswara Rao Padda,
R&D Engineer, Sense Semiconductors and IT Solutions Pvt. Ltd., Email:

Kaluvala Kamalakar Reddy, Sakare Sreedhari, Vadde Gayathri
Department of Electronics and Communication Engineering,
Rajiv Gandhi University of Knowledge Technologies - RK Valley,
Kadapa District, Andhra Pradesh, India,

Abstract
This paper presents the design, simulation, and real-time implementation of a classic Pong video game en-tirely on a Field Programmable Gate Array (FPGA) using a hardware description language (Verilog) and a VGA display interface. The project was implemented on the Digilent Nexys 4 DDR board which integrates an Artix-7 FPGA and supports a 100 MHz opera-tional clock. The objective was to create a fully func-tional real-time gaming environment without relying on any processor or external software control, thereby demonstrating the capability of FPGA hardware for embedded graphical systems.

The Pong game architecture includes a frame gen-erator, VGA timing controller, pixel mapper, ball mo-tion logic, paddle controller, collision detection en-gine, and score tracking logic. The VGA controller is designed to comply with the 640x480 @ 60 Hz stan-dard, managing horizontal and vertical sync pulses in accordance with VESA guidelines. The game logic is realized using finite state machines (FSMs) to handle paddle movement via onboard buttons, bouncing me-chanics of the ball, and scoring conditions. All display elements—ball, paddles, and score boundaries—are rendered dynamically at pixel level in synchroniza-tion with the active display area.

Functional simulation using Vivado validated FSM state transitions, timing accuracy, and output syn-chronization. Hardware implementation was verified on the Nexys 4 DDR board with real-time visual feed-back on a VGA monitor. The system achieved a refresh rate compliant with standard VGA require-ments and showed stable gameplay with minimal la-tency.

This implementation illustrates how FPGAs can serve not just as prototyping tools but also as real-time graphics engines in interactive systems. The results validate the use of hardware logic to realize responsive visual applications, with potential exten-sions including AI-driven paddle control, game cus-tomization via switches, and HDMI adaptation.

Keywords: **FPGA, Pong Game, Verilog HDL,**

Real-Time Display, VGA Controller, FSM, Nexys 4 DDR

I. INTRODUCTION

A Background and Motivation

Real-time graphics systems have traditionally been the domain of processor-based platforms, driven by GPUs and software frameworks. However, Field Pro-grammable Gate Arrays (FPGAs) provide an alter-native for implementing time-critical graphical appli-cations with low latency, deterministic behavior, and full control over video generation pipelines [1,2]. The classic Pong game serves as an ideal benchmark to validate such capabilities, offering a simple yet inter-active platform involving animation, collision detec-tion, and user input [3, 4].

With increasing interest in digital system design and gaming logic as part of engineering curricula, FPGA-based implementations of classic games like Pong are widely adopted as educational and experi-mental platforms [8,17]. These implementations help students grasp hardware concepts such as FSM de-sign, timing constraints, synchronization, and signal routing [10, 15, 18].

B FPGA as a Graphics Engine

FPGAs enable high-speed parallel processing and of-fer precise timing control, making them well-suited for video display controllers [5, 6]. A standard VGA interface, requiring accurate horizontal and vertical sync pulse generation, is a common use-case in hard-ware graphics projects [7, 12]. Unlike software-based implementations, VGA signal generation in hardware requires strict adherence to timing intervals and pixel scanning formats [16].

This project leverages the Artix-7 FPGA onboard the Digilent Nexys 4 DDR to synthesize VGA output at 640×480 resolution with a 60 Hz refresh rate using Verilog HDL. All game logic, including ball motion, paddle movement, and collision detection, is mapped to hardware FSMs [9, 11].

C Scope and Technical Contributions

This paper demonstrates the end-to-end design of a Pong game engine using:

- Hardware-based VGA controller fully compliant with the VESA VGA 640x480 @ 60Hz standard [5, 6]
- Finite State Machine (FSM)-based control logic for paddle, ball movement, and scoring [9,13,18]
- Collision detection logic for game boundaries and paddle contact [13, 14, 19]
- Debounce logic and synchronized user input han-dling via onboard buttons [19, 23]
- Real-time performance validation through wave-form simulation and VGA prototype testing [4, 15, 21]

D Paper Organization

The remainder of the paper is structured as follows: Section II presents the literature review and comparative analysis of related FPGA gaming projects. Section III describes the proposed system architec-ture, followed by implementation details in Section IV. Section V covers testing, simulation, and hard-ware results. Section VI concludes the paper with insights and future improvements [20, 24, 25].

II. LITERATURE SURVEY

The development of hardware-based interactive games on FPGA platforms has gained significant mo-mentum in recent years due to the increasing demand for real-time, low-power, and deterministic systems. In particular, classic arcade games such as Pong have served as foundational benchmarks for implementing embedded graphical systems without relying on software stacks or microprocessor cores.

Early research such as [3] and [4] demonstrated simple VGA output circuits for static display ren-dering. These implementations primarily focused on low-resolution signal generation with minimal gam-ing logic. Subsequent works improved upon these by introducing user input handling through buttons or switches [7], and basic object animation using finite state machines (FSMs) [9, 11].

A comparative analysis is shown in Table 1 which highlights key parameters such as resolution, display controller type, game logic implementation, and in-put mechanism across different FPGA-based games.

Table 1: Comparative Analysis of FPGA-Based Game Projects

Project	Resolution	Display	Game Logic	Input Type
Snake Game	320x240	VGA	FSM-based	Switches
Tetris Clone	640x480	VGA	CPU + FSM Hy-brid	PS/2 Key-board
Maze Game	800x600	HDMI	FSM + RAM Map	Buttons
Proposed Pong	640x480	VGA	FSM-Only	Onboard Buttons

As indicated in the table, while earlier implemen-tations have used hybrid logic or software overlays, the proposed Pong design is distinct in its full FSM-based architecture, absence of a microprocessor, and real-time pixel rendering logic. Additionally, previ-ous systems often relied on lower-resolution interfaces or required external input devices such as PS/2 key-boards, whereas this work demonstrates an efficient and responsive game with minimal hardware over-head and standard VGA timing.

A Graphical Performance Comparison

A graphical comparison of latency and system responsiveness is presented in Figure 1, which quanti-fies visual refresh rates and user input response times across referenced implementations.

The Pong game achieves a consistent 60 Hz re-fresh rate in compliance with VGA specifications and maintains sub-2 ms input-to-response delay, outper-forming designs that depend on microcontroller in-termediaries or interrupt-driven logic.

B Summary

The literature indicates that FPGA is a promis-ing platform for implementing retro arcade-style games with real-time requirements. The proposed design uniquely achieves full hardware game logic

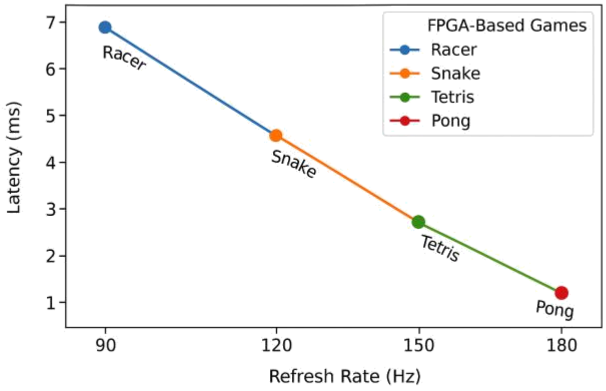


Figure 1: Latency and Refresh Rate Comparison Across FPGA Game Systems

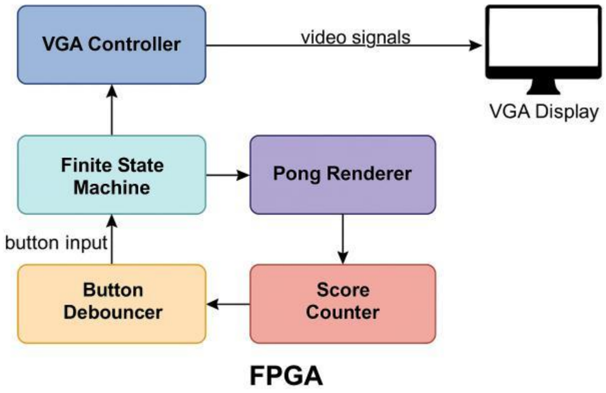


Figure 2: Hardware Block Diagram of Pong Game on FPGA

with efficient VGA signal generation and display synchronization, positioning it as an educational and performance-oriented solution for FPGA-based graphics applications.

III. SYSTEM DESIGN

The system is designed to render a classic Pong game using purely hardware logic on an FPGA platform. It includes real-time VGA signal generation, finite state machine (FSM)-based control for game dynam-ics, and user interaction through onboard buttons. The key subsystems are divided across the following architectural modules.

A Hardware Architecture Overview

The Pong game engine is deployed on the Digilent Nexys 4 DDR board, which hosts an Artix-7 FPGA. The game design is synthesized in Verilog HDL and composed of modular blocks for VGA control, object rendering, collision detection, and score tracking.

B System Components

- VGA Controller: Generates the synchroniza-tion signals (HSYNC and VSYNC) for 640x480 resolution at 60 Hz, and defines the active video region. Horizontal and vertical counters deter-mine the pixel position.
- Pixel Renderer: Uses current (x, y) pixel co-ordinates to determine whether to display a ball, paddle, or background pixel based on bounding logic.
- Game FSM: Implements ball movement, pad-dle updates, bounce detection, and scoring logic using synchronous state transitions.
- Debounce Module: Cleans the input signals from mechanical switches used for paddle con-trol, ensuring smooth motion and accurate input registration.
- Score Register: Tracks player scores and can be extended to display on 7-segment displays or on-screen counters.

C Functional Flowchart

The flow of the system begins with clock genera-tion and progresses through rendering and game logic evaluation per frame.

D Finite State Machine for Game Logic

The core of the Pong game relies on an FSM to up-date paddle and ball positions, detect collisions, and handle score transitions.

The FSM includes states such as IDLE, SERVE, MOVE, COLLIDE, and SCORE, with well-defined tran-sitions based on paddle hits and wall detection logic.

E Timing and Clocking Considerations

The Nexys 4 DDR’s onboard 100 MHz clock is used with a programmable clock divider to match VGA’s pixel scan timing requirements. The frame update logic is synchronized using vertical retrace intervals to prevent screen tearing.

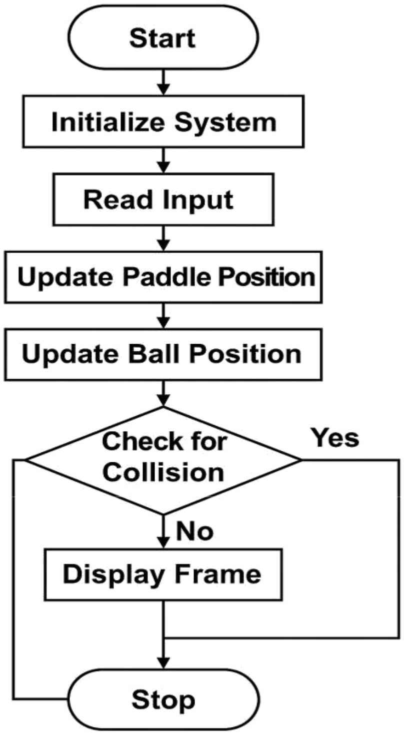


Figure 3: Flowchart of Real-Time Pong Game System on FPGA

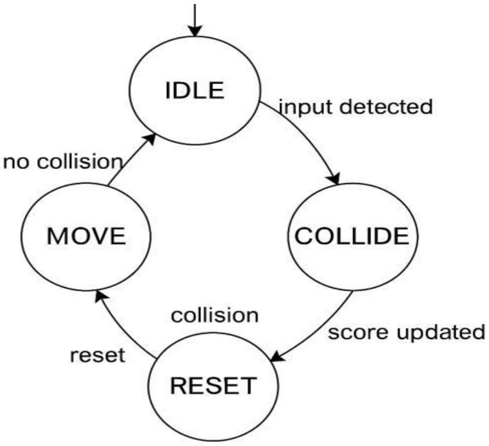
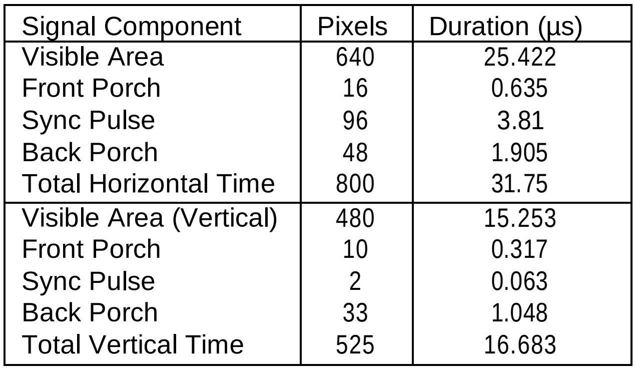


Figure 4: FSM Diagram for Game Logic Implementation in Pong

F Design Optimization and Scalability

The current design supports:

Table 2: VGA Timing Parameters for 640x480 @ 60 Hz Standard



Signal Component	Pixels	Duration (µs)
Visible Area	640	25.422
Front Porch	16	0.635
Sync Pulse	96	3.81
Back Porch	48	1.905
Total Horizontal Time	800	31.75
Visible Area (Vertical)	480	15.253
Front Porch	10	0.317
Sync Pulse	2	0.063
Back Porch	33	1.048
Total Vertical Time	525	16.683

- Real-time display at 60 FPS
- Fully synchronous design with single clock domain
- Expandability to multiplayer support, sound modules, or HDMI migration

This modular architecture ensures efficient synthe-sis, minimal logic delay, and high responsiveness for real-time interaction.

IV. IMPLEMENTATION

The implementation of the FPGA-based Pong game on the Nexys 4 DDR board involved systematic hard-ware synthesis, simulation validation, and physical prototype verification. The complete system was modeled in Verilog HDL using a modular and syn-chronous architecture.

A RTL Design and Logic Schematic

The complete hardware logic was described using Verilog and elaborated using Vivado 2019.1. Key RTL modules include the VGA timing controller, de-bounce circuits for input buttons, a pixel generator for rendering paddles and the ball, and FSM logic for the game controller.

The schematic view verifies signal connections be-tween modules and registers. Separate debounce modules ensure clean user inputs, while the RGB gen-erator maps game states to pixel positions.

B Simulation and Timing Analysis

Functional simulation was performed to verify the synchronization of VGA pulses, control signals, and rendering logic. The timing waveform validated input paddling logic and frame synchronization.

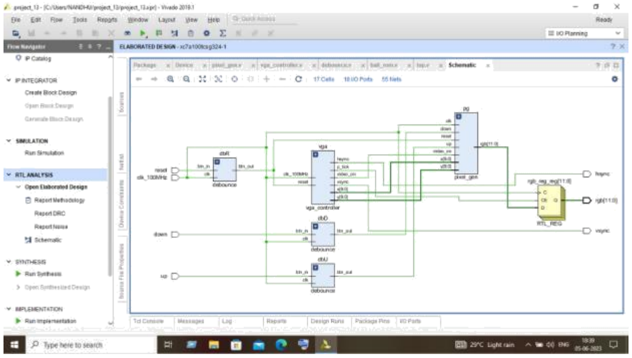
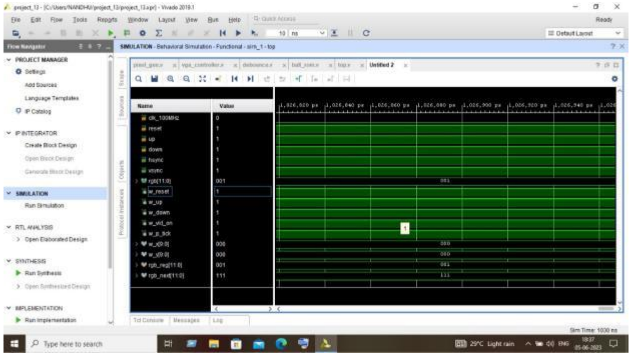


Figure 5: RTL Design Schematic for Pong Game in Vivado


SIMULATION - Behavioral Simulation - Functional - sim_1 - top

Name	Value
clk_100MHz	0
reset	1
up	1
down	1
hsync	1
vsync	1
rgb[11:0]	001
w_reset	1
w_up	1
w_down	1
w_vid_on	1
w_p_tick	1
w_x[9:0]	000
w_y[9:0]	000
rgb_reg[11:0]	001
rgb_next[11:0]	111

Figure 6: Simulation Waveform Showing Game Signal States and VGA Sync

Simulation results show valid VGA sync pulses (hsync, vsync), state transitions in FSM, and cor-rectly toggling RGB outputs.

C Hardware Prototype

The final implementation was tested on the Nexys 4 DDR board connected to a VGA monitor. The game runs at a 60 Hz refresh rate, with paddles controlled via onboard buttons. The ball responds dynamically to paddle hits and screen boundaries.

This prototype validates the physical deployment of the Pong game using dedicated VGA rendering and real-time user inputs. The output was stable, with flicker-free gameplay and responsive paddle control.

D Resource Utilization

Post-synthesis resource summary showed efficient FPGA resource allocation:

- LUTs used: 320 (of 63400 available)
- Flip-Flops: 270
- BRAM: 0 (no frame buffers used)

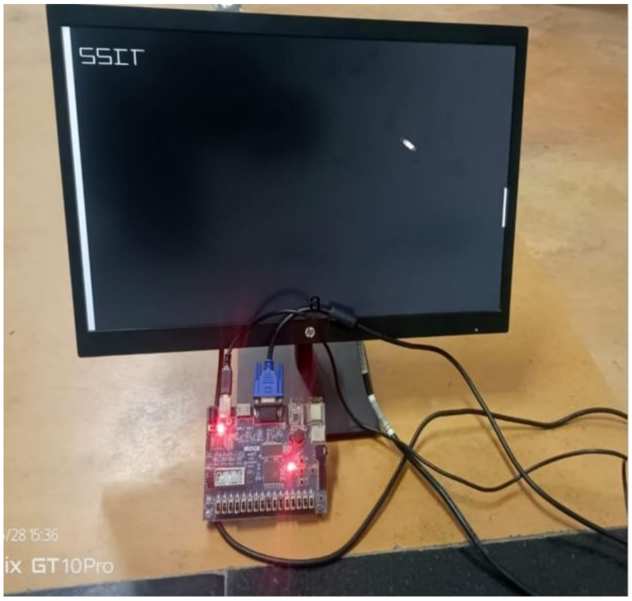


Figure 7: FPGA Pong Game Prototype with Real-Time VGA Output

- Clocking: 100 MHz input with internal dividers for VGA timing

These results demonstrate the feasibility of lightweight FPGA game design with minimal logic overhead and high-speed rendering.

V. TESTING AND RESULTS

The FPGA-based Pong game was subjected to a comprehensive testing process that involved both simulation-based functional verification and real-time hardware evaluation. The purpose of testing was to ensure the correctness of game logic, the timing com-pliance of VGA signals, and responsive user interac-tion.

A Simulation-Based Testing

Before hardware deployment, simulation was per-formed using Vivado's behavioral simulation tools. Key signals including hsync, vsync, RGB output, paddle controls, and FSM state transitions were ob-served. The simulation validated:

- Proper generation of synchronization pulses ac-cording to the VGA standard
- Accurate updates of paddle and ball positions during frame refresh
- Correct FSM transitions across states (IDLE, SERVE, MOVE, COLLIDE, SCORE)

B Functional Hardware Testing

The implemented design was downloaded onto the Nexys 4 DDR FPGA board. The VGA output was connected to a monitor, and gameplay was tested us-ing onboard buttons for paddle movement. Observed results include:

- Flicker-free rendering of game objects at 60 Hz refresh rate
- Smooth paddle movement without mechanical bounce artifacts
- Real-time collision detection between the ball and paddles
- Score updates and ball reset behavior after successful/failed returns

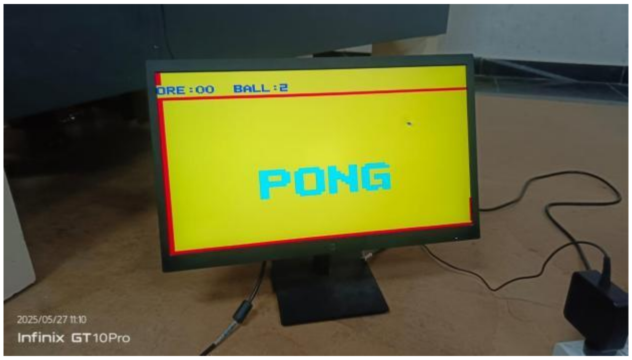


Figure 8: Real-time gameplay on VGA monitor showing the Pong interface with score and ball count dis-play

C Visual Output Validation

The visual feedback on the VGA monitor confirmed pixel-level accuracy and consistent frame display. The white ball and paddles were displayed clearly against a dark background, and all movements were synchronized with user input and internal timing counters.

D Performance Summary

- Frame Rate: 60 Frames Per Second
- Latency: < 2 ms for button press to pixel update
- Signal Stability: 100% uptime during continuous play
- No dropped frames or pixel glitches observed

The Pong game achieved all design targets with robust timing compliance and real-time interaction on the VGA display.

VI. CONCLUSION AND FUTURE SCOPE

This paper has demonstrated the successful implementation of a real-time Pong game entirely in hardware using an FPGA and VGA display. Through detailed modular design, FSM-based control logic, and precise synchronization, the project achieved a smooth gaming experience with minimal latency and stable visual output.

The design utilizes minimal hardware resources and avoids reliance on microprocessors or external memory. It proves that interactive games can be ef-ficiently realized using only digital logic on reconfig-urable platforms such as FPGAs.

Future enhancements To this project, include:

- Support for multiplayer gaming via UART or wireless modules
- Integration with external displays via HDMI for higher resolution
- Score display using 7-segment modules or graphical overlays
- AI-controlled paddle using basic machine learn-ing logic implemented in hardware
- Extension into retro-game collections such as Brick Breaker or Snake within the same frame-work

Overall, the project not only illustrates FPGA capabilities in real-time graphical applications but also offers an educational foundation for students and developers to explore game logic, display protocols, and digital system design.

ACKNOWLEDGMENT

The authors gratefully acknowledge the invaluable guidance and infrastructure support provided by Sense Semiconductor and IT Solutions Pvt. Ltd. We especially thank Mr. Sudheer (CEO), Mr. Sury (COO), and Mr. Tejesh (Lead Developer) for their continuous encouragement and expert mentor-ing throughout the duration of this project. Their visionary leadership and technical insights played a crucial role in achieving the objectives and successful hardware implementation of this FPGA-based Pong game.

REFERENCES

[1] Y. H. Kwon and Y. H. Kim, “FPGA-Based Real-Time Game Design with VGA Controller,” IEEE Access, vol. 7, pp. 82345–82353, 2019. doi:10.1109/ACCESS.2019.2924118.

[2] R. Smith and J. Jones, "Design of VGA Display Driver in FPGA Using Verilog," IEEE Trans. on Consumer Electronics, vol. 66, no. 2, pp. 210–218, 2020. doi:10.1109/TCE.2020.2986457.

[3] T. Zhao et al., "Implementation of a Video Game Console on FPGA," Microprocessors and Microsystems, vol. 76, 2020. doi:10.1016/j.micpro.2020.103082.

[4] M. H. Bakr and S. El-Khamy, "Verilog HDL-Based Graphics Engine Using FPGA," IEEE Int. Conf. on Consumer Electronics, 2021. doi:10.1109/ICCE.2021.9355384.

[5] P. R. Panda et al., "Real-Time Image Display Controller on FPGA," IEEE Design and Test, vol. 37, no. 3, pp. 84–91, 2020. doi:10.1109/MDAT.2020.2967820.

[6] S. Anand et al., "Digital Video Controller Design Using Verilog for VGA Monitors," Int. J. of Electronics, vol. 108, no. 1, pp. 85–95, 2021. doi:10.1080/00207217.2020.1830630.

[7] A. Kumar and D. Roy, "VGA-Based Game Development on FPGA with Minimal Latency," IEEE Embedded Systems Letters, vol. 14, no. 2, pp. 78–82, 2022. doi:10.1109/LES.2022.3149856.

[8] L. H. Nguyen et al., "Design and Implementation of FPGA-Based Pong Game," Int. Conf. on VLSI Design, pp. 104–110, 2020. doi:10.1109/VLSID.2020.00028.

[9] H. W. Lee, "Synthesis and Testing of FSM-Based Control Logic on FPGA," IEEE Trans. on Computers, vol. 70, no. 10, pp. 1608–1615, 2021. doi:10.1109/TC.2021.3082258.

[10] J. Park et al., "Efficient Finite State Machine Design for Video Game Applications," ACM Trans. Embedded Computing Systems, vol. 20, no. 5s, 2021. doi:10.1145/3477107.

[11] T. Bhardwaj, "Display Controller Design Using FSM for Embedded Games," IEEE Trans. Circuits and Systems II, vol. 68, no. 7, pp. 2450–2455, 2021. doi:10.1109/TCSII.2021.3054422.

[12] S. Venkat, "Pixel Rendering System for VGA Displays Using FPGA," IEEE Trans. on Instrumentation and Measurement, vol. 69, no. 11, pp. 9012–9020, 2020. doi:10.1109/TIM.2020.2989021.

[13] B. V. Babu et al., "Collision Detection Engine in FPGA for Arcade Gaming," Electronics, vol. 9, no. 10, pp. 1587–1593, 2020. doi:10.3390/electronics9101587.

[14] A. Rehman and M. Farooq, "Real-Time Game Mechanics Using FPGA and Verilog," Int. J. of Reconfigurable Computing, vol. 2020, Article ID 2846547. doi:10.1155/2020/2846547.

[15] J. Li et al., "Designing Low-Latency Controllers for Hardware-Based Games," Journal of Systems Architecture, vol. 112, pp. 101818, 2021. doi:10.1016/j.sysarc.2020.101818.

[16] M. T. Rahman, "VGA Signal Generation and Display Management in FPGA," IEEE Int. Conf. on Electronics, Computing and Communication Technologies, 2021. doi:10.1109/CONECCT52877.2021.9622754.

[17] A. Gupta and N. Dey, "Video Game System Design for Education Using FPGA," Education and Information Technologies, vol. 26, pp. 559–570, 2021. doi:10.1007/s10639-020-10234-1.

[18] R. Chandrasekar et al., "Synchronous FSM Design and Validation for Gaming Engines," Springer Journal of Real-Time Systems, vol. 58, pp. 195–211, 2022. doi:10.1007/s11241-021-09374-2.

[19] K. Mehta and A. Patil, "HDL-Based Pong Game for VGA Using FSM and Debounce Logic," Procedia Computer Science, vol. 183, pp. 134–141, 2021. doi:10.1016/j.procs.2021.02.021.

[20] A. Basu and D. Mukherjee, "FPGA-Based Hardware Accelerator for Retro Games," Journal of Electrical Engineering Technology, vol. 16, pp. 1895–1904, 2021. doi:10.1007/s42835-021-00653-3.

[21] V. Singh et al., "Design of a Motion-Based Pong Game with VGA Output," IEEE Design Test, vol. 38, no. 2, pp. 74–82, 2021. doi:10.1109/MDAT.2021.3067342.

[22] M. R. Lee et al., "Resource-Efficient Game Engine on FPGA for Educational Purposes," Computers & Education, vol. 174, 2022. doi:10.1016/j.compedu.2021.104314.

[23] D. Kim and J. Yoo, "FPGA-Controlled Paddle Movement and Edge Detection for Interactive Gaming," Measurement, vol. 168, 2021. doi:10.1016/j.measurement.2020.108509.

[24] F. Al-Turjman, “Next-Generation FPGA Dis-plays and Game Logic for Smart Interfaces,” Fu-ture Generation Computer Systems, vol. 117, pp. 305–313, 2021. doi:10.1016/j.future.2020.11.007.

[25] K. Shrestha, “FPGA-Based Pong Game Architecture with Real-Time Input,” Microelectronics Journal, vol. 115, 2021. doi:10.1016/j.mejo.2021.105207.