**SIMATS SCHOOL OF ENGINEERING**

**SAVEETHAINSTITUTEOFMEDICALANDTECHNICALSCIENCES CHENNAI-602105**

**CAPSTONE PROJECT REPORT PROJECT TITLE**

# Automata-Based Hardware Scheduler for Task Management in MultiCore Processor

***Submitted in the partial fulfillment for the award of the degree of***

**BACHELOR OF ENGINEERING**

**IN COMPUTER SCIENCE AND ENGINEERING**

**CSA1377|Theory of Computation with Algorithms**

**Submitted by**

Junaithul Harsha. J (192211690)

Kathiravan (192211129)

**UNDER THE GUIDANCE OF**

E.MONIKA

OCTOBER 2024

**DECLARATION**

We, **Kathiravan ,Junaithul Harsha** students of **Bachelor of Engineering in CSE**, Department of Computer Science and Engineering, Saveetha Institute of Medical and Technical Sciences, Saveetha University, Chennai, hereby declare that the work presented in this Capstone Project Work entitled **“Automata based Hardware Scheduler for task Management."**is the outcome of our bonafide work and is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics.

Junaithul Harsha (192211690)

Kathiravan(192211129)

Date: 24-10-2024

Place: Saveetha School of Engineering, Thandalam.

**CERTIFICATE**

This is to certify that the project entitled **“Design and Optimization of DFA for Pattern Recognition in Hardware."** submitted by **MANO C,MADHAVAN C** has been carried out under our supervision. The project has been submitted as per the requirements in the current semester of B.E

Computer science and engineering

Teacher-in-charge

E.Monika

**Automata-Based Hardware Scheduler for Task Management in Multicore Processors**

**Project Title:**

**Automata-Based Hardware Scheduler for Task Management in Multicore Processors**

**Objective:**

**The primary objective of this project is to design and implement a hardware-based task scheduler for multicore processors using automata theory. The focus will be on optimizing resource allocation and task prioritization in real-time systems, ensuring efficient and deterministic scheduling under varying workloads.**

**Background and Motivation:**

**Multicore processors are essential in modern computing for applications ranging from mobile devices to high-performance servers. Effective task management and scheduling are critical for ensuring these systems run efficiently, especially in real-time applications like autonomous vehicles, industrial control systems, and multimedia streaming. Traditional software-based schedulers can struggle with predictability and response times. Thus, there is a need for a dedicated hardware scheduler that leverages the efficiency and simplicity of automata theory to manage tasks with low latency.**

**Using automata in hardware scheduling provides:**

**Predictable Execution: Automata offer a finite state structure, enabling the design of deterministic and efficient state transitions.**

**Reduced Overheads: Hardware scheduling minimizes the computational load on the CPU, freeing up resources for other critical tasks.**

**Real-Time Guarantees: With optimized resource allocation and task prioritization, real-time constraints can be met more consistently.**

**Key Objectives:**

**Design of Automata-Based Scheduler:**

**Develop a finite state machine (FSM) model to represent the various states and transitions for task management.**

**Define states and transitions to account for task arrival, execution, suspension, and completion.**

**Implement priority levels and constraints to ensure higher-priority tasks are executed first.**

**Resource Allocation and Task Prioritization:**

**Incorporate a mechanism to dynamically adjust task priorities and allocate resources based on task urgency and processor availability.**

**Design algorithms to maximize CPU utilization across cores while meeting the real-time requirements of individual tasks.**

**Hardware Implementation:**

**Develop a hardware description of the scheduler using a hardware description language (HDL) such as VHDL or Verilog.**

**Simulate the scheduler on an FPGA or another hardware prototyping platform to validate its functionality.**

**Compare performance metrics like task latency, throughput, and power consumption against a software-based scheduler.**

**Evaluation and Testing:**

**Measure the effectiveness of the automata-based scheduler in real-time task management by running benchmark tests.**

**Evaluate the system's performance under different load scenarios, with both real-time and non-real-time tasks.**

**Analyze the results and provide insights into the strengths and limitations of automata-based scheduling.**

**Methodology:**

**Literature Review:**

**Study automata theory concepts applicable to task scheduling and resource management.**

**Review existing multicore processor scheduling techniques, especially those implemented in hardware.**

**Design and Simulation:**

**Design the scheduler FSM and implement it in an HDL.**

**Simulate the system using tools like ModelSim or Xilinx Vivado to validate correctness.**

**Hardware Prototyping:**

**Implement the design on an FPGA to test the scheduler in a real hardware environment.**

**Use tools to measure power consumption, latency, and resource usage on the FPGA.**

**Performance Comparison:**

**Test the automata-based scheduler against software-based scheduling approaches to measure improvements in task management efficiency.**

**Expected Outcomes:**

**A functional hardware-based scheduler optimized for multicore processors, using automata theory.**

**A detailed comparison of the automata-based hardware scheduler with traditional software-based schedulers.**

**Performance improvements in terms of reduced scheduling latency, efficient task prioritization, and improved resource utilization in multicore environments.**

**Documentation and insights on the applicability of automata-based schedulers in real-time and multicore processing.**

**Tools and Resources:**

**Hardware Description Language (HDL): VHDL or Verilog for hardware design.**

**Simulation Tools: ModelSim, Xilinx Vivado.**

**FPGA Prototyping Board: A suitable FPGA board (e.g., Xilinx or Altera) to test and validate the scheduler.**

**Benchmarking Software: Tools for evaluating real-time performance metrics (e.g., latency, throughput).**

**Timeline:**

**Phase 1 (Weeks 1–4): Research and literature review on automata-based scheduling and multicore processing.**

**Phase 2 (Weeks 5–8): Design of FSM-based scheduler model and simulation of individual components.**

**Phase 3 (Weeks 9–12): Implementation of the scheduler in HDL and FPGA prototyping.**

**Phase 4 (Weeks 13–16): Testing and evaluation of scheduler performance, comparison with software-based schedulers.**

**Phase 5 (Weeks 17–20): Final documentation, project presentation, and submission.**

**Potential Challenges:**

**Complexity in FSM Design: Ensuring that the FSM handles all possible states and transitions in real-time can be complex.**

**Hardware Limitations: FPGA resource constraints may limit the complexity of the scheduler design.**

**Real-Time Testing: Finding realistic test cases and benchmarks that simulate real-time workloads accurately.**

**Conclusion:**

**The proposed automata-based hardware scheduler project has the potential to offer an innovative and efficient solution for task management in multicore processors. With real-time task prioritization, reduced overheads, and improved resource utilization, this scheduler could be highly beneficial for systems with stringent real-time requirements.**

**References**

* + - * Hopcroft, J. E., & Ullman, J. D. (1979). *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley.
      * Patterson, D. A., & Hennessy, J. L. (2013). *Computer Organization and Design: The Hardware/Software Interface*. Elsevier.
      * Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to Algorithms*. MIT Press.
      * Lysecky, R., Vahid, F., & Givargis, T. (2004). "Hardware Description

Language (HDL) Analysis and Synthesis Using FPGA-based DFA." *Journal of Parallel and Distributed Computing*, 64(4), 442-452.

* + - * Xilinx, Inc. (2017). *Vivado Design Suite User Guide*. Xilinx.