|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **THE UNIVERSITY OF SHEFFIELD School of Electrical and Electronic Engineering**  **3rd Year Individual Project — Project Initialisation Document** | | | |  |
| **Student Name** | | Amaan Mujawar | | | |
| **Project Title** | | Implement an Arithmetic Unit utilising Approximate Computing into RISC-V SoC | | | |
| **Supervisor** | | Mr Neil Powell | **Second Marker** |  | |

**Project motivation:**

The motivation for this project is to design and implement an Arithmetic Unit utilising Approximate Computing methods for integration into a Reduced Instruction Set Computer (RISC-V) System on Chip (SoC). Approximate Computing allows for less precise calculations where perfect accuracy is not essential, making it beneficial for applications such as image processing, video streaming, and machine learning, where minor inaccuracies can enhance speed and reduce energy consumption.

The rationale behind this project stems from the growing demand for efficient computing in data-intensive environments. As devices become more complex and data volumes increase, traditional computing methods often struggle, leading to significant power and time consumption. Approximate Computing provides a solution by enabling faster, more energy-efficient operations without greatly compromising output quality.

The project will unfold in several stages, including planning, background research, design and development, integration, and testing. The initial phase will involve a literature review of Approximate Computing techniques to identify promising methods and applications, informing the Arithmetic Unit's design.

The project aims to systematically investigate how Approximate Computing can enhance RISC-V SoC performance. The ultimate goal is to create a functional Arithmetic Unit that exemplifies the benefits of this approach, paving the way for further advancements in efficient computing methods.

**Project Specification:**

The objective of this project is to design, implement, and integrate an Arithmetic Unit utilising Approximate Computing methods within a RISC-V SoC. Development will utilise the Artix Nexus 7 FPGA and the Xilinx Vivado Design Suite as primary tools.

Key tasks include conducting a literature review to identify relevant Approximate Computing techniques, followed by the design phase, which will focus on creating architectural designs and schematic diagrams outlining the unit's functionality. Implementation will involve coding the Arithmetic Unit using hardware description languages (HDLs) and synthesising, simulating, and programming the FPGA with Vivado. Rigorous testing and verification will include developing test cases to evaluate performance and accuracy, ensuring proper integration with the RISC-V SoC.

Throughout the project, comprehensive documentation will capture design choices, implementation processes, and testing outcomes. The expected outcome is a fully functional Arithmetic Unit that demonstrates the benefits of Approximate Computing, along with detailed documentation of the design and implementation process, providing valuable insights into the performance advantages of Approximate Computing in RISC-V SoCs.

**Project Schedule:**

This project will consist of several main tasks and work packages, each with specific objectives and deliverables. It will follow clear milestones, including the completion of the literature review, finalisation of design specifications, FPGA implementation, testing and verification, and submission of a final report.

The initial task is a literature review focused on identifying and evaluating existing Approximate Computing techniques relevant to the Arithmetic Unit, resulting in a summary report that will inform the subsequent design phase.

Following the literature review, the design development phase will create architectural designs and schematic diagrams, producing comprehensive design documentation, including specifications and performance metrics, reliant on insights from the literature review.

The implementation phase will involve coding the Arithmetic Unit using hardware description languages (HDLs) and programming the Artix Nexus 7 FPGA with Xilinx Vivado. Deliverables for this phase include functional HDL code and a programmed FPGA, heavily dependent on finalised design documentation.

Rigorous testing and verification will follow, validating the performance and accuracy of the Arithmetic Unit through developed test cases and simulations in Vivado. This phase will document results and inform any necessary design iterations, relying on successful implementation.

Throughout the project, detailed documentation will capture design decisions, implementation processes, and testing outcomes, culminating in a final report summarising findings and insights.

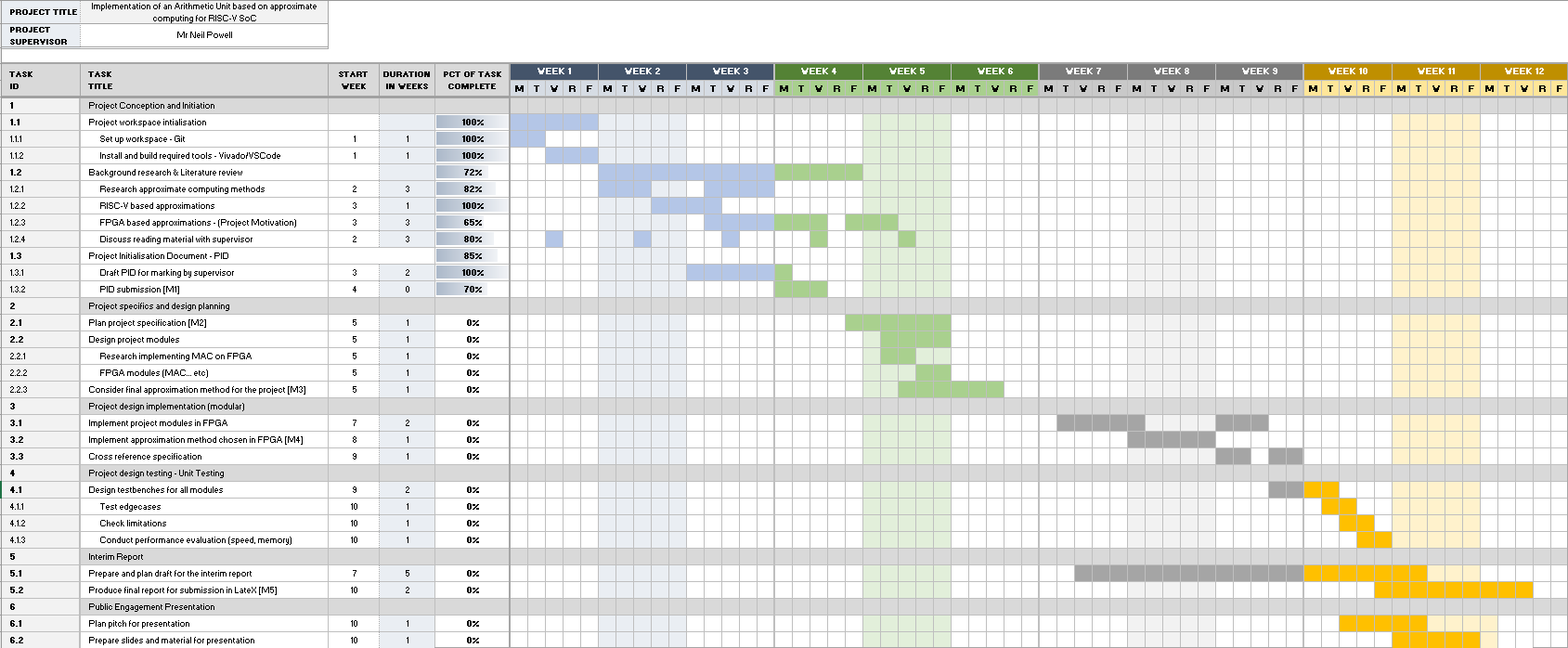
The critical path comprises the literature review, design development, implementation, and testing phases, with potential delays affecting the overall timeline. Key milestones will include the completion of the literature review, finalization of design documentation, FPGA implementation, successful testing and verification, and submission of the final project report. A Gantt chart will visualise the project timeline, highlighting tasks, deliverables, and interdependencies to ensure timely progress.

**Risk Register:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Description of Risk** | **Risk evaluation (L/M/H)** | **Chance of risk (L/M/H)** | **Mitigation of Risk** |
| **1** | Loss of data (USB key) | M | L | Multiple back-ups, GitHub. |
| **2** | Delays in literature review affecting subsequent phases | M | M | |  | | --- | |  |   Implement peer reviews and consult supervisor regularly. |
| **3** | Incomplete or inaccurate design documentation | M | M | Consult with supervisor to ensure documentation is thorough and accurate. |
| **4** | Technical difficulties with FPGA programming | H | M | Seek support from resources and experienced colleagues. |
| **5** | Issues during testing leading to inaccurate results | H | M | Develop detailed test cases and conduct iterative testing. |
| **6** | Hardware failures (e.g., FPGA malfunctions) | H | L | Keep backup hardware available and conduct regular checks. |
| **7** | Resource availability (e.g., tools, software licenses) | M | L | Secure necessary tools and licenses early in the project. |
| **8** | Changes in project scope or requirements | M | M | Establish clear specifications. |
| **9** | Staff member availability (e.g., illness, other commitments) | M | M | Have tasks that can be carried on in parallel, so workflow is not interrupted. |
| **10** | Time management issues affecting project timeline | M | M | Conduct regular status meetings with supervisor. |

**Appendix:**

**Gantt chart:**



**References:**

*Reference should be IEEE referencing style. Web references should be kept to a minimum as they are usually not peer-reviewed.*

1. W. Kempton, J. Tomic, “Vehicle-to-grid power implementation: From stabilizing the grid to supporting large-scale renewable energy”, Journal of Power Sources, vol. 144, no. 1, pp. 280-294, June, 2005.