2 Project Plan

2.1 Task Decomposition

- Part 1: Software Testing Implementation in Python
    - Research some well established algorithms for video compression and decompression
        - Choose 3-5 (or more) Algorithms to thoroughly benchmark the following metrics:
            - Compression Ratio
            - Latency
            - Complexity
    - Find a dataset for testing the compression and decompression
        - Considering that the project demonstration will use HDMI and many development boards have HDMI controllers that convert the data to RGB24, find some input data we can use to test our algorithm that matches the RGB24 protocol.
    - Write the Python implementation that benchmarks the algorithms chosen beforehand
        - Based on the performance of the algorithms we test, we will choose an algorithm that we can implement in hardware
        - Explore parameterizing algorithms so that a knob could be used in the demonstration to control the latency vs compression ratio relationship.
- Part 2: FPGA Implementation + Demonstration
    - Choose a development board to implement the algorithm on for the demonstration
        - Note: HDMI Sink + Source and FPGA large enough to support the algorithm are the requirements for this developmental board
    - Create a plan/schematic for a pipelined FPGA solution for the selected algorithm
    - Implement the compression and decompression Algorithm on the FPGA and benchmark the compression ratio and latency
    - Gather the necessary hardware for the demonstration
        - 2 HDMI Monitors
        - 1 Development Board
        - 1 HDMI Splitter
        - 2 HDMI Cables
    - Final debugging and testing of the entire system working together

2.2 Project Management/Tracking Procedures

We will be using the agile management style for software development. We still intend on having a couple meetings a week and can use these meetings as SPRINTs for planning project goals/outcomes for the next meeting.

Gitlab will track our issues when we work on the python elements of the project. We can also use the issues board to track progress on FPGA pipelining.

2.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

- Having a list of applicable video compression algorithms.
- Creating a framework to test video compression algorithms in python, and using that to pick our final compression algorithm.
- Having a design laid out of the pipeline
- Having the FPGA finalized
- In an agile development process, these milestones can be refined with successive iterations/sprints (perhaps a subset of your requirements applicable to those sprint).

2.4 Project Timeline/Schedule

2.5 Risks And Risk Management/Mitigation

Python implementation will have low risks (0.2) because python compression libraries exist and are pretty well established.

Having a design of the FPGS will have high risks because putting a compression algorithm to hardware can be a complex, tedious, and abstract (0.5)

- choose an algorithm with less complexity and with plenty of documentation to make the implementation easier

Creating a list of applicable algorithms will be pretty low risk (0.05) because our requirements for picking algorithms are relatively simple: must be not too complex, must have low latency, must be implementable in hardware.

An agile project can associate risks and risk mitigation with each sprint.

2.6 Personnel Effort Requirements

Task	Man-Hours	Explanation
Algorithm Research	4	Spend enough but not too much time finding the most optimal algorithms to run on FPGA.
Python Benchmarking Implementation	10	Creating a Python program that can benchmark the different algorithm prototypes.
Python compression algorithms	20	Implementing various video compression/decompression algorithms in Python that take in a video as an input.
Research and Choose a Dev Board For The Demonstration	1-2	Choose a dev board that has HDMI sink & source, preferably one made for video processing
Create a Plan/Schematic for the FPGA based on the software algorithm benchmark	10	Based on the software implementation of the algorithm, make a plan for the architecture of the FPGA
Implement the Compression part of the FPGA	20	Write the VHDL code to compress an input stream of raw RGB24 values.
Testing/Debugging of Compression Algorithm on FPGA	10	Test some expected values including edge cases, debug accordingly until fully functional
Implement the Decompression part of the FPGA	20	Write the VHDL code to decompress an input stream of compressed RGB24 values.
Testing of Decompression Algorithm on FPGA	10	Test some expected values including edge cases, debug accordingly until fully functional
Gather Hardware Needed for the Demo	1-2	Make sure we have all required hardware for the demo whether that be from Deere or the TLA

Final Debugging of the Entire System	10	Having all the hardware for the demo set up and making sure everything works together.

2.7 Other Resource Requirements

Potentially need a camera for live video.