# Phase 3 Project Report: Pipeline with Forwarding and Hazard Units Processor

**Authors:** Matthew Collins & Lewis Bates  
**Emails:** mcollins42@tntech.edu & lfbates42@tntech.edu  
**Course:** ECE 4120/5120  
**Date:** April 28, 2025

## Table of Contents

1. [Introduction](#introduction)
2. [Part 3: Bonus - Forwarding and Hazard Units](#X169607f3038d61d6e2504fb3dc2729ac9b6d7af)
   * [Modified Block Diagram](#part-3-modified-block-diagram)
   * [Elaboration on VHDL Implementation](#Xaa98953de0f770df581ed54939c8d67b742c0e1)
   * [Flow Summary, RTL View, and Technology Map View](#X81b65cda256ae9a1e538f57718fd5f320370d9a)
   * [Testbench Elaboration](#part-3-testbench-elaboration)
   * [Waveform Elaboration](#part-3-waveform-elaboration)
   * [Performance Penalty](#part-3-performance-penalty)
   * [Hardware Overhead](#part-3-hardware-overhead)
3. [Conclusion](#conclusion)

## Introduction

This report documents the design, implementation, and testing of a MIPS processor for Phase 3 of the ECE 4120/5120 project.

* **Part 3 (Bonus)**: An enhanced pipelined processor with forwarding and hazard units.

Each part includes modified block diagrams, VHDL implementation details, synthesis views, testbench descriptions, and waveform analyses, as required by the project deliverables. Additionally, pipelining improvements, hardware overhead, and bonus-related performance penalties are analyzed.

## Part 3: Bonus - Forwarding and Hazard Units

### Part 3: Modified Block Diagram

### Part 3: Top-Level Entity

A diagram of a computer

AI-generated content may be incorrect.

### Part 3: Instruction Fetch and Instruction Decode Stages

A diagram of a computer system

AI-generated content may be incorrect.

### Part 3: Execute, Memory, and Write Back

A diagram of a computer program

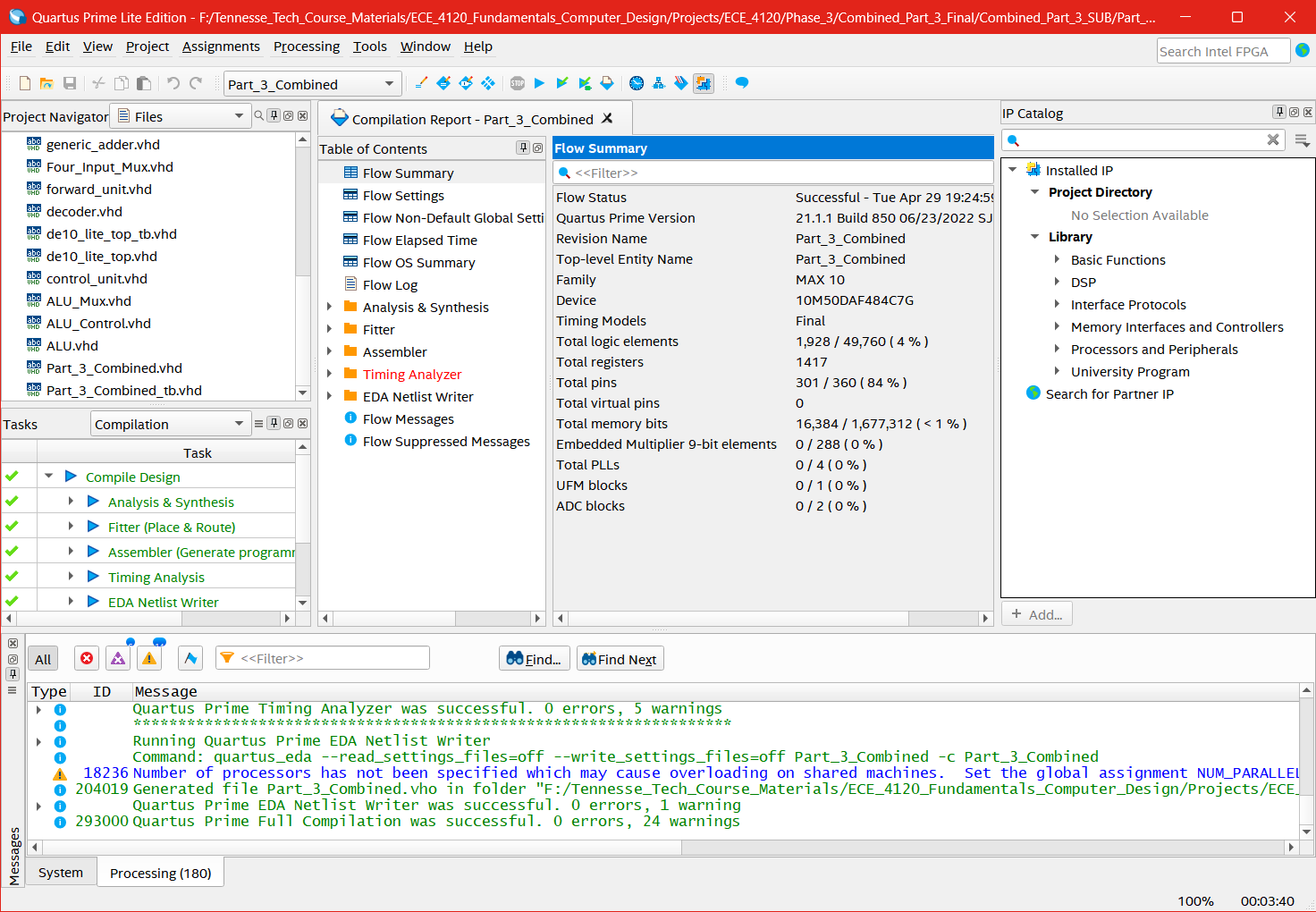
AI-generated content may be incorrect.

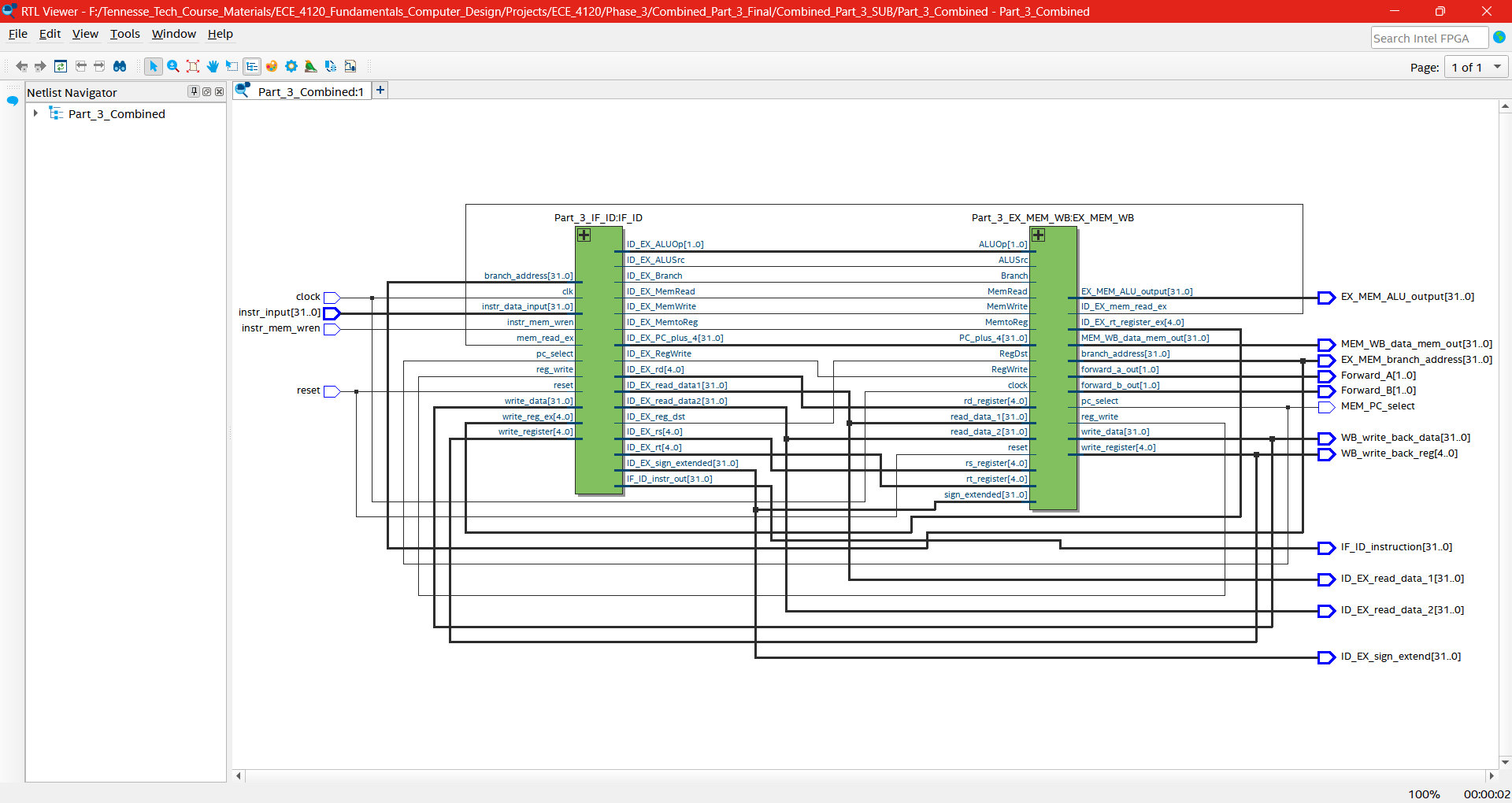
* **Enhancements**: Adds forwarding paths from EX/MEM and MEM/WB to EX, and a hazard unit for stalling.
* **Clock Signals**: Same as Part 2.
* **Control Signals**: Extended with forwarding and stall signals.

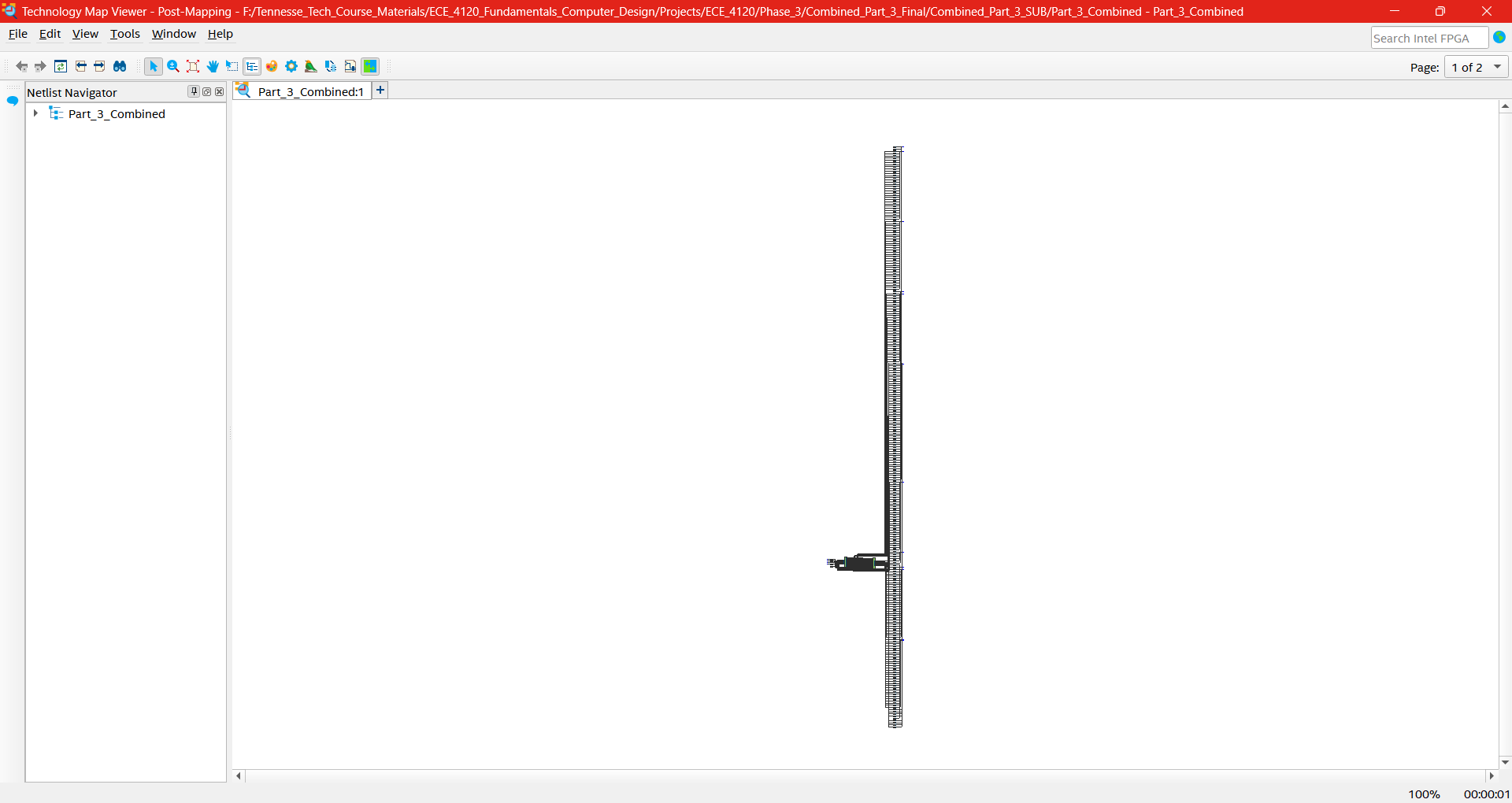
### Part 3: Elaboration on VHDL Implementation

* **Forwarding Unit**:
  + Detects dependencies (e.g., EX\_MEM\_rd = ID\_EX\_rs) and forwards data from EX\_MEM\_ALU\_out or MEM\_WB\_write\_data to ALU inputs.
  + VHDL: Adds muxes before ALU inputs, controlled by comparison logic.
* **Hazard Unit**:
  + Detects load-use hazards (e.g., lw $t1 followed by add $t2, $t1, $t2) and stalls by disabling IF/ID and ID/EX writes.
  + VHDL: Logic to set stall signals based on ID\_EX\_MemRead and register matches.
* **Modifications**:
  + Part\_3\_EX\_MEM\_WB.vhd: Adds forwarding muxes, comparison logic and outputs for forwarded data.
  + Part\_3\_IF\_ID.vhd: Incorporates stall logic to freeze PC and IF/ID register.

### Part 3: Flow Summary, RTL View, and Technology Map View



****

****

* **Flow Summary: Further resource increase due to forwarding muxes and hazard logic.**
* **RTL View**: Shows added forwarding paths and stall controls.
* **Technology Map View**: Reflects additional FPGA logic.

### Part 3: Testbench Elaboration

The testbench would test:

1. beq $t1, $t2, Equal
2. lw $t1, 8($t2)
3. add $t2, $t1, $t2
4. sw $t3, 100($t2)
5. add $t4, $t1, $t2

* **Forwarding**: Between add $t2, $t1, $t2 and subsequent add.
* **Stalling**: Between lw $t1 and add $t2, $t1, $t2.
* **Verification**: Checks forwarded values and stall cycles.

### Part 3: Waveform Elaboration

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

* **Forwarding**: Shows ALU\_input failed to update properly from MEM\_WB\_write\_data .
* **Stalling**: Displays pipeline bubble (e.g., IF/ID unchanged) for load-use hazard.
* **Labels**: Highlight forwarding paths and stall insertion.

### Part 3: Performance Penalty

* **Forwarding Only**: Minimal penalty; eliminates most stalls, slight delay from muxes.
* **Forwarding + Hazard**: One-cycle stall per load-use hazard (e.g., lw to add), increasing latency for dependent instructions.

### Part 3: Hardware Overhead

* **Forwarding Only**: Additional muxes (e.g., two 32-bit muxes per ALU input) and comparison logic.
* **Forwarding + Hazard**: Adds stall control logic and register disable signals, increasing logic elements further in addition to the hardware overhead for forwarding.

## Conclusion

This report presents a complete implementation of a proposed design for forwarding and hazard units (Part 3). The bonus features enhance performance by addressing data hazards, though at the cost of increased hardware complexity. Our deliverables have errors that prevent the final implememtation from aligning with all of the project requirements. This will be shown in the synthesis and simulation results in our final submission.