Indian Institute of Technology Kharagpur

AUTUMN Semester, 2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Computer Organization and Architecture Laboratory

Addendum: Final Integration - Autonomous Program Execution

October 21, 2025

Instructions: *This document outlines the final FPGA demonstration for the processor design project. The objective is to demonstrate a fully integrated processor that autonomously executes a program from on-chip memory upon a single command.*

Context and Objective

This final task evolves the processor from a manually-stepped design to an autonomous system. You will implement a control scheme where a single button press initiates the entire program execution. The processor will run at the full speed of the FPGA's system clock until a 'HALT' instruction is encountered, at which point it will stop and display the final result. This mirrors the boot-up sequence of a real-world microprocessor.

FPGA Task: Autonomous Program Execution

Your processor will fetch and execute a program from a BRAM-based instruction memory. A single button press will reset and start the processor. The program will run to completion, and the final result will be displayed on the board's LEDs.

Key Implementation Requirements:

- **Synthesizable BRAM with COE File:** You must use the IDE's IP Catalog (e.g., Vivado's Block Memory Generator) to create a ROM using BRAM resources. This IP core must be configured to load its initial contents from a '.coe' file containing your hand-assembled machine code.
- **Test Program and Machine Code:** Convert the assembly program below into 32-bit machine code based on your custom instruction format.

```
; Program to calculate sum of integers from 5 down to 1 (5+4+3+2+1=15)
ADDI R1, R0, 5      ; R1 = 5 (counter)
ADDI R2, R0, 0      ; R2 = 0 (accumulator)
LOOP:
ADD  R2, R2, R1     ; R2 = R2 + R1
SUBI R1, R1, 1      ; R1 = R1 - 1
BPL  R1, LOOP       ; Branch to LOOP if R1 > 0
ST   R2, 0(R0)      ; Store result from R2 to Mem[0]
HALT                ; Halt the processor
```

- **Single-Button Control and Autonomous Execution:** The two-button system is now replaced with a single **reset button**.
  - The processor must be driven by the main 100MHz system clock of the FPGA.
  - Pressing the reset button should synchronously reset the processor: the PC must be set to 0, and a control FSM should enter a 'RUN' state.

  - In the ‘RUN‘ state, the processor executes one instruction on each clock cycle (or as per your pipeline design) until it fetches the ‘HALT‘ instruction.
  - The ‘HALT‘ instruction must cause the control FSM to transition to an ‘IDLE‘ state, where it remains until the next reset. In the ‘IDLE‘ state, the PC should no longer advance.

- **Multiplexed LED Output:** The 32-bit value in the result register (e.g., R2) must be continuously displayed on the 16 LEDs after the program halts. A slide switch will be used to select which half of the register is visible.

Hardware Pin Assignments (Nexys4 DDR / Nexys A7)

You must use the following pin assignments in your XDC constraints file. This ensures consistency and proper operation on the target hardware.

Table 1: Required Pinout for the Nexys4 DDR / Nexys A7 Board

Signal in Verilog	**FPGA Pin**	**Description**
`clk`	`E3`	100MHz System Clock Input
`reset`	`BTNC`	CPU Reset Button (Active-High)
`led[15:0]`	`H17, K15, ...`	16 Green LEDs for Output
`sw[0]`	`J15`	Display Select Switch (0=Lower, 1=Upper)

Suggested Demonstration Flow

1. Briefly show the TA your Verilog implementation of the Run/Idle control FSM and the final ‘.coe‘ file.
2. Press the designated **`reset` button** (‘BTNC‘) once.
3. Explain that the button press initiated the program, which has now run to completion at full clock speed and is in the ‘IDLE‘ state.
4. **Final Verification:** Use the display select switch (‘SW[0]‘) to show the TA that the processor has computed the correct result.
   - With the switch OFF, the LEDs should show the lower 16 bits of the result: ‘16’h000F‘.
   - With the switch ON, the LEDs should show the upper 16 bits of the result: ‘16’h0000‘.