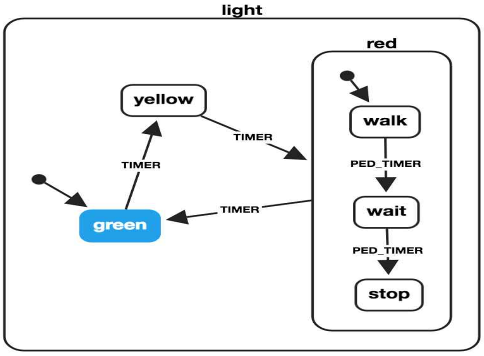
Traffic Light Control System

2021050300
Department of Computer Science
Kim Jaemin

1. Project Overview

- The goal of this project is to implement a traffic light control system simulator using MIPS assembly language.
- The simulation models a real-world intersection, where traffic lights for east-west and north-south directions alternate in operation.
- Key MIPS implementation points:
  1) Delay simulation using loops (timers)
  2) Register usage for storing state values (e.g., $s0, $s1)
  3) Output display through syscall or an I/O simulator
- The program will include the following features:
  1) State transition logic: The lights will transition in the order Green → Yellow → Red, alternating between north-south and east-west directions.
  2) Delay logic for each state: Each light state will maintain its color for a fixed duration, implemented using loops (e.g., NOP, LOOP) in assembly.
  3) Console output display: The current light status will be visually represented in the terminal for easier understanding. Example → [NS: GREEN | EW: RED], [NS: YELLOW | EW: RED]



- Example configuration of traffic light states:

State	Vehicle Signal	Pedestrian Signal	Duration
s0	Green	Red	5seconds
s1	Yellow	Red	2seconds
s2	Red	Green	5seconds
s3	Red	Flashing Green	2seconds

● Through this simulation, the objective is to practice core assembly concepts such as conditional branching, loops, function structures, and output control in a hands-on system context.

2. Motivation

- Unlike high-level languages, assembly language directly interacts with hardware, requiring a deep understanding of control flow and state-based logic.
- To gain practical experience with these characteristics, I selected the traffic light control system as a familiar yet logically structured real-world application.
- Although traffic lights may appear simple—just changing colors over time—they actually involve state transitions, timed delays, and output control, making them an ideal example of a state-based system.
- This complexity makes them well-suited for applying fundamental assembly concepts in practice.
- Through this project, I aim to design a state-based control logic, implement loops, conditionals, and delay mechanisms in assembly, and enhance my understanding of low-level programming through hands-on experience.

3. Weekly Development Plan (5 Weeks)

- Week 1 (Apr 29 – May 3) - Set up the assembly development environment. Design the control structure of the traffic light system and write the initial pseudocode.
- Week 2 (May 3 – May 10) - Implement basic traffic light operations
  - Green → Yellow → Red transitions and delay logic using loops.
- Week 3 (May 10 – May 17) - Design and implement detailed console output formatting.
  Example: NS: GREEN | EW: RED
- Week 4 (May 17 – May 24) - Add extended functionality such as nighttime blinking mode,
  which activates orange blinking lights based on time-based conditions.
- Week 5 (May 24 – May 31) - Final code review, add comments and documentation, capture test results,
  and submit the final report via LMS.