Department of
Computer Science and Engineering

University of Dhaka

Assignment 2: ARM-Based Employee Payroll & Salary Processing System

CSE 2106: Microprocessor and Assembly Language Lab

Batch: 30/2nd Year 1st Semester 2025

1 Assignment 2: PayrollSys-32: ARM-Based Employee Payroll & Salary Processing System

Background Story

You are assigned to a fictional company **TechNova Solutions**, a mid-sized software firm transitioning its payroll system into an embedded edge device for secure, offline salary computation. The company acquired a custom ARM-based device (running on a Cortex-M microcontroller), which will perform:

- Employee record processing
- Attendance analytics
- Payroll generation
- Allowance and deduction computation
- Tax calculation
- Overtime handling
- Leave management
- Pay-slip preparation

Your task is to implement the full **PayrollSys-32** system in **pure ARM Assembly**, using memory-mapped RAM to store structures, tables, computations, flags, and logs. All modules must operate together as an integrated system, with each implemented as a callable subroutine via `BL` instructions.

Module 1 — Employee Record Initialization

Story Context: Every new employee onboarding requires storing personal and salary-structure details in memory.

Technical Requirements:

- Design a structure containing:
  - Employee ID (32-bit)
  - Name pointer (32-bit)
  - Base Salary (32-bit)
  - Job Grade (8-bit)
  - Department Code (8-bit)
  - Bank Account Number (32-bit)
  - Attendance Pointer (32-bit)
  - Allowance Table Pointer (32-bit)
- Initialize 5 employee records with test values.
- Store structures from memory address `0x20000000`.
- Ensure proper alignment (byte/halfword/word).

Module 2 — Attendance Data Loader

Story Context: Monthly attendance data is uploaded to the embedded payroll device.

Technical Requirements:

- Create a 31-day attendance log (1 byte/day): 1 = present, 0 = absent.
- Load attendance from memory locations `0x20001000`, `0x20001100`, etc.
- Count monthly present days and store in the employee structure.
- Use loops and `LDRB`/`STRB` instructions.

Module 3 — Leave Management & Deduction Logic

Story Context: Employees with too many absences must receive salary deductions.
Requirements:

- If present days < 22, apply:

$$\text{leave\_deficit} = 22 - \text{present\_days}$$

$$\text{deduction} = \text{leave\_deficit} \times 300$$

- Store deduction in payroll structure.
- Set a flag `LDEFICIT` if leave deficit > 5.

Module 4 — Overtime Calculation Module

Story Context: Employees may work overtime, stored in memory.
Requirements:

- OT hours stored from address `0x20002000`.
- OT rate by job grade:
  - Grade A: 250/hour
  - Grade B: 200/hour
  - Grade C: 150/hour
- Implement a grade-based lookup table.
- Compute:

$$\text{ot\_pay} = \text{ot\_hours} \times \text{rate}$$

Module 5 — Allowance Processor

Story Context: Allowances vary by department and salary structure.
Allowances

- HRA: 20% of base salary
- Medical: 3000
- Transport:
  - IT: 5000
  - HR: 4000
  - Admin: 3500

Requirements:

- Compute HRA using fixed-point arithmetic.
- Total:

$$\text{total\_allowance} = \text{HRA} + 3000 + \text{transport}$$

Module 6 — Tax Computation Using Slab System

Story Context: PayrollSys-32 uses a slab-based progressive tax model.
Slabs:

- $\leq$ 30,000: 0%
- 30,001–60,000: 5%
- 60,001–120,000: 10%
- $> 120,000$: 15%

Requirements:

- Implement using `CMP`, `BHI`, `BLE`, `BGE`.
- Compute:

$$\text{tax} = \text{gross\_salary} \times \text{tax\_rate}$$

Module 7 — Net Salary Calculator

Story Context: All payroll components integrate into one final salary.
Requirements:

$$\text{net\_salary} = \text{base\_salary} + \text{allowance} + \text{ot\_pay} - \text{tax} - \text{leave\_deduction}$$

- Detect overflow and set `OVERFLOW_FLAG`.
- Store in the employee record.

Module 8 — Sorting Employees by Net Salary

Story Context: Management needs a sorted list of employees by salary.
Requirements:

- Use bubble sort or selection sort.
- Sort by `net_salary` (descending).
- Swap entire employee structures.
- Store sorted list at `0x20005000`.

Module 9 — Departmental Salary Summary

Story Context: Finance requires department-wise expenditure reports.
Requirements:

- Compute:
  - total_salary_IT
  - total_salary_HR
  - total_salary_Admin
- Store them in a summary table.

Module 10 — Bonus & Performance Rating Engine

Story Context: End-of-year bonuses depend on performance scores.
Data: Scores stored at `0x20006000`.
Bonus Rules:

- ≥ 90: 25% of base salary
- 75–89: 15%
- 60–74: 8%
- < 60: 0%

Requirement:

- Add computed bonus to `employee->bonus`.

Module 11 — UART Pay-Slip Generator

Story Context: Employees must receive a salary summary via UART.
Requirements:

- Implement UART transmit routine:
  - Write to UART_DR
  - Wait for TX ready
- Convert numeric values to ASCII.
- Print:
  - Employee ID
  - Net Salary
  - Tax
  - Allowance
  - Bonus
  - Final Pay

Expected Outcome

A fully integrated ARM Assembly program that:

- Initializes and manages employee records in memory
- Tracks attendance, working hours, and overtime
- Computes gross salary, allowances, deductions, and net pay
- Generates payroll alerts (missing attendance, overtime limit, deduction anomalies)
- Schedules monthly salary processing using timer counters
- Implements sorting of employees by salary or performance metrics
- Outputs a structured payroll summary through UART

Students must demonstrate:

- Modular system design using ARM Assembly
- Mastery of memory management, pointer operations, and structured data handling
- Implementation of algorithms such as sorting, scheduling, and multi-stage computation
- Use of UART communication for reporting and debugging
- Proper handling of overflow, error flags, and exception conditions

2 Policy

Copying from the Internet, classmates, seniors, or from any other source is strongly prohibited. 100% marks will be *deducted* if any such copying is detected.