

EE3220 System on Chip Design

Course Review

Final Exam

- Date: 30 April 2025
- Time: 14:00 – 16:00
- Venue: LT-6
- Format: Closed-Book Exam (2 double-sided cheat sheets, calculator)
- Questions: Short and Long Essay Questions
- Coverage: Lectures, Tutorials, Labs, Assignments, TBL, Guest Lectures

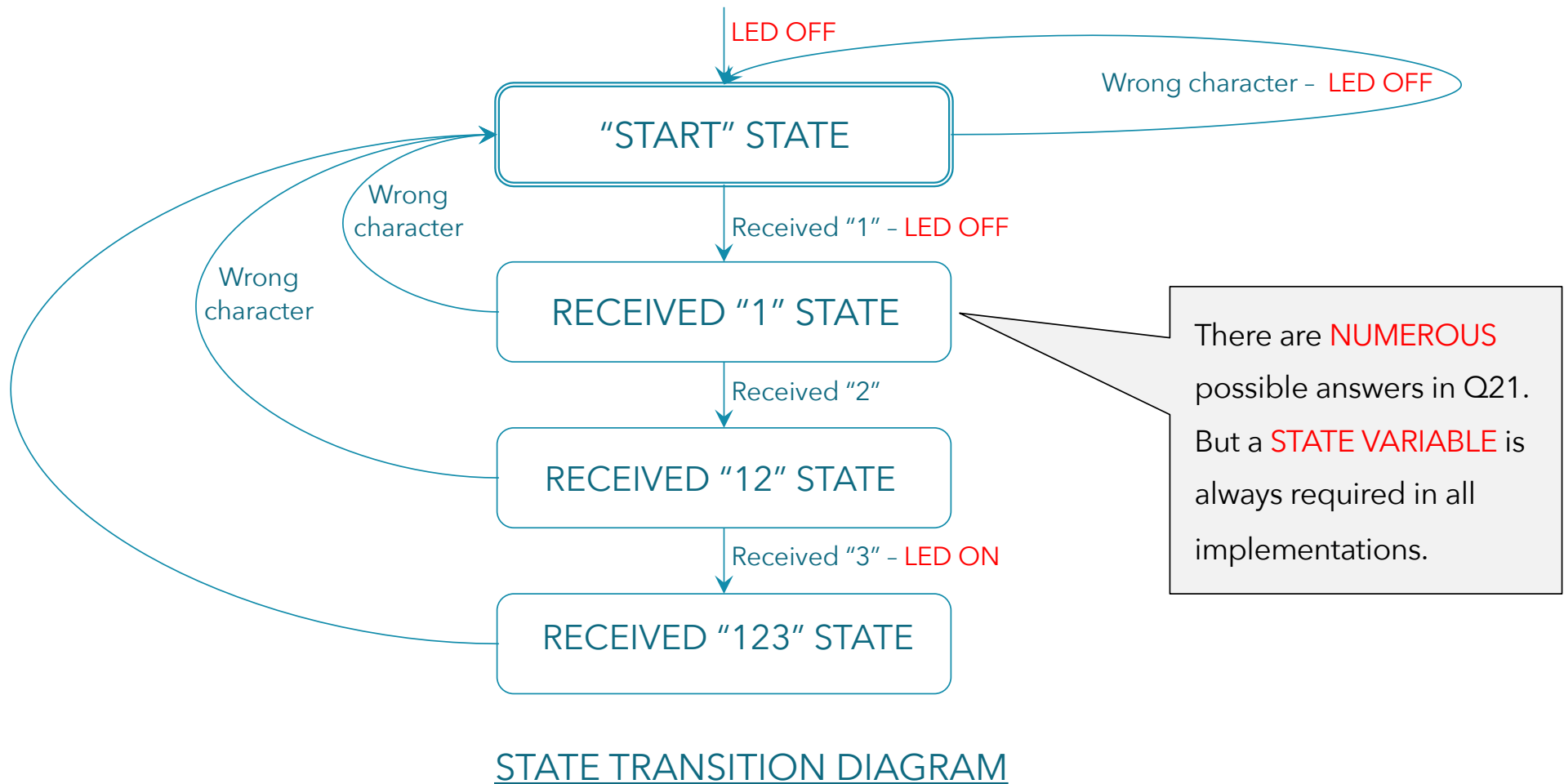
Question Types

- Programming Questions:
 - C, Assembly, Verilog
- Conceptual Questions:
 - Lectures 1, 2 (H/W and S/W Architecture)
 - Lectures 3, 4 (ARM Hardware)
 - Lectures 5, 7, 8 (ARM Firmware)
 - Lectures 9, 10, 11 (HDL, FPGA)
 - Lectures 6, 12 (TBL: USB Firmware, Wi-Fi SoC Security)
 - Lectures 13 (RISC-V)
 - Labs 1, 2, 3, 4 (ARM Firmware, Xilinx FPGA)

C Programming

- Language Features:
 - Data Types (Integer, floating-point, unsigned, custom)
 - Control flow statements (Conditional, Looping, Jump)
 - Memory Segments (Text, Data, BSS, Heap, Stack)
 - Variable Types (Auto, Static, Global, Volatile)
 - Functions
 - Pointers and Arrays
- How to build an FSM?

Quiz 2: Question 21



Quiz 2: Question 21

```
enum State {S0, S1, S2, S3}; // FSM

int main(){
    DigitalOut led(LED1);
    char command;
    State current_state = S0; // STATE VARIABLE
    led = 0;

    while (true){
        pc.read(&command, 1);
        switch (current_state) {
            case S0: if (command == '1') {
                        current_state = S1;
                    } else {
                        current_state = S0;
                        led = 0;
                    }
                    break;
            case S1: if (command == '2') {
                        current_state = S2;
                    } else {
                        current_state = S0;
                    }
                    break;
            case S2: if (command == '3') {
                        current_state = S3;
                    } else {
                        current_state = S0;
                    }
                    break;
            case S3: current_state = S0;
                    led = 1;
                    break;
        }
    }
}
```

```
int main(){
    DigitalOut led(LED1);
    char command;
    char passphrase[3] = {'1','2','3'};
    int index = 0; // STATE VARIABLE
    led = 0;

    while (true){
        pc.read(&command, 1);
        if (command == passphrase[index]) {
            index++;
            if (index == 3) {
                index = 0;
                led = 1;
            } else {
                led = 0;
            }
        } else {
            index = 0;
            led = 0;
        }
    }
}
```

Verilog Programming

- Language Features
 - Module Declaration
 - Data Types
 - Procedural Blocks
 - Operators
 - Control Structures
 - Blocking vs Non-blocking Assignments
- How to build an FSM?

Embedded System & SoC

- Embedded System
 - Specialized computing system integrated into a device
- SoC
 - Single chip integrates CPU, memory, I/O, peripherals
 - Enables compact, energy-efficient design
- Examples:
 - Bike computer
 - Motor control
 - Zynq SoC

ARM Cortex-M4

- ARMv7-M (Thumb-2) ISA
- Features:
 - 3-stage pipeline
 - Atomic operations
 - Exception model: NVIC, xPSR, EXC_RETURN
 - Program image: MSP/PSP, Vector, etc.
- Applications:
 - IoT applications
 - Embedded applications

ARM Interrupts & Exceptions

- Types: Reset, NMI, HardFault, IRQs
- Flow:
 - Finish current instruction
 - Save context
 - Load handler address
 - Execute handler
 - Use EXC_RETURN to restore context
- Critical for:
 - Responsive and safe embedded systems

Embedded C Programming

- C over Java / Python
 - Deterministic, lightweight
- Concepts:
 - Volatile variables
 - Critical sections and race conditions
 - Stack usage in recurse functions
- Examples:
 - Recursive factorial function
 - ISR handlers

UART, Timer, PWM

- UART:
 - Asynchronous serial communication
 - Circular queues, RX/TX ISRs
- Timer:
 - Generates periodic interrupts
 - Used for scheduling, delays
 - Programmable output control
- PWM:
 - Duty cycle control

Verilog HDL Fundamentals

- Levels:
 - Behavioral, RTL, Gate-level
- Key Concepts
 - reg, wire, always, assign
 - Blocking versus non-blocking
 - Synchronous versus asynchronous reset
 - FSMs, D flip-flops, counters, parity generators
- Usage:
 - Design and simulate digital hardware

FPGA + Zynq SoC

- Zynq Architecture
 - Dual ARM Cortex-A9 + FPGA fabric
- Vivado Flow:
 - IP integrator -> Block Diagram
 - Top HDL -> Bitstream
 - SDK/Vitis for S/W + Debug
- Applications:
 - Embedded Linux
 - Real-time acceleration
 - Custom peripherals