Motion-Aware Study Timer Project Overview (SystemVerilog, DDR4 Board):

Project Overview:

This project implements a focused time tracker using an onboard accelerometer or gyroscope. The FPGA continuously reads motion data and filters it to classify the board as still or moving. When motion stays below a defined threshold, the system considers the board stationary and the timer counts focused time. If motion exceeds the threshold, the timer pauses. Focused minutes display on the seven-segment display. The system also sends periodic focus summaries over UART.

Required Hardware:

- DDR4 SoC FPGA development board (for example, Xilinx or Intel board used in class)
- On-board accelerometer/gyro or IMU using I2C or SPI
- USB connection (via a USB cable to the PC) for programming and UART
- Seven-segment display on the board
- Optional pushbutton for sensitivity mode switching or reset

Required Software:

- Vivado or Intel Quartus
- SystemVerilog toolchain
- UART terminal program on the PC

Project Goal:
Teach sensor interfacing, fixed point processing, timing logic, display control, and UART communication. The scope fits a single course project.

SystemVerilog Modules Needed:

- accel_reader (config and sample)
- filter_unit (simple moving average)

IMU Interface Module (i2c_master or spi_master)

Reads accelerometer or gyro values. Converts raw SPI or I2C data into signed values.

- Implements I2C or SPI master.

- Configures accelerometer or gyro registers.
- Samples X, Y, Z data at a fixed rate.
- Outputs signed fixed point values.

Motion Filter Module (motion_classifier (thresholding)

Calculates magnitude or RMS motion. Compares against a threshold. Outputs still or moving.

- Computes a motion metric such as magnitude or RMS.
- Uses fixed point arithmetic.
- Applies simple filtering like moving average or low pass.
- Compares result to a programmable threshold.
- Outputs a 1-bit still or moving signal.
- Optionally tracks average or recent motion.

Focus Timer Module (focus_timer)

- Counts time when still is true.
- Uses system clock.
- Stores seconds and minutes.
- Generates a one second tick from the system clock.
- Always increments total elapsed time.
- Increments focused time only when still is true.
- Optionally requires still to be true for N consecutive samples.
- Computes focus percentage as focused over total.

SSEG Display Module (sseg_driver)

Convert minutes into segment patterns. Drives all digits.

- Converts focused time to seconds or minutes.
- Splits value into digits.
- Drives all segment and enable signals.
- Refreshes digits using multiplexing.

UART Logger Module (uart_tx)

Sends messages with:

- Focused time
- Total time
- Motion statistics
- Transmits ASCII text messages.
- Sends summaries at a fixed interval, for example every 1 or 5 seconds.
- Includes elapsed time, focused time, percent focus, and motion stats.

Top-Level Module (top_level integration module)

Connects all modules:

- IMU to filter
- Filter to timer
- Timer to SSEG
- Timer and filter to UART
- Connects IMU to motion filter.
- Connects filter to focus timer.
- Connects timer to seven-segment display.
- Connects timer and motion data to UART.
- Includes clocking, reset, and pin mappings.

Implementation Steps

Step 1: Verify IMU Interface

- Read X, Y, Z values at a fixed rate.
- Display/stream raw values over UART for debugging.

Step 2: Build Motion Filter

- Compute |accel| (the magnitude of the acceleration vector) or RMS.
- Compare against a threshold
- Output a 1-bit stable still signal.
- Compute magnitude or RMS.

Step 3: Build Focus Timer

- Create a one-second clock divider.
- When still is true, increment focused time (seconds).
- Track total time (seconds) independently.

Step 4: Drive Seven-Segment Display

- Convert focused minutes or seconds to digits
- Output correct segment patterns

Step 5: Add UART Summaries

- Send text messages at fixed intervals (like every 5 seconds)
- Include focus percentage and motion metrics.

Step 6: Integrate and Test Everything

- Connect all modules in the top-level SystemVerilog module.
- Add constraints for IMU, SSEG, and UART pins.
- Add constraints for the push buttons.
- Synthesize, program, and test/validate on hardware.

Optional Features

- Button to switch sensitivity modes
- Reset button
- Log data to a computer file
- Pushbutton toggles sensitivity profiles.

- Profile stored in a small register or state machine.

Expected Outputs

- SSEG shows focused minutes
- UART prints: total time, focused time, percent focus, average motion

Scope Confirmation

The design is small and fits the course. It covers sensor input, logic processing, display output, and UART communication. This design is compact and realistic. You handle real sensors, digital filtering, timing logic, display driving, and serial communication. It fits cleanly within a digital design or FPGA systems course.

System Components
1. Sensor Interface
- Implement I2C or SPI controller in SystemVerilog.
- Configure accelerometer registers.
- Continuously sample X, Y, Z acceleration.

2. Motion Processing
- Compute magnitude: sqrt(x^2 + y^2 + z^2). Use fixed-point arithmetic.
- Apply low-pass filtering to smooth noise.
- Compare to threshold defining “still” vs “moving.”
- Maintain a rolling still-time counter.

3. Focus Timer Logic
- If still for N consecutive cycles, increment focused_time.
- Always increment elapsed_time.
- Track percent focus: focused_time / elapsed_time.

4. Output Modules
- Seven-segment display driver showing focused_time in seconds or minutes.
- UART transmitter sending periodic logs:
  - Elapsed time
  - Focused time
  - Percent focus
  - Recent RMS acceleration

Development Tips
- Start with sensor communication.
- Verify acceleration readings in UART before adding processing.
- Integrate slowly, test each module.

- Use timing constraints to meet the board's clocking needs.

High-Level Workflow:

- Initialize the IMU after reset. Initialize sensor through I2C/SPI.
- Continuously sample acceleration or rotation data.
- Filter samples and compute motion level.
- Classify still versus moving.
- Update elapsed and focused timers.
- Display focused time on the seven-segment display.
- Send/Transmit UART logs periodically (once per second).
- Acquire raw acceleration samples continuously.
- Filter data and compute motion metric.
- Update focus and elapsed timers.
- Drive SSEG with real-time focused time.

Expected Outputs:

- Seven-segment display shows focused time.
- UART prints total time, focused time, percent focus, and average motion.