# Keypad Scan Registration FSM

**Exported:** 11/3/2025 0:24:30
**Link:** https://gemini.google.com/u/1/app/157a314fccd6ccae?pageId=none

## Prompt:

Target device: Lattice iCE40 UP5K FPGA.

Overall Goal: Write SystemVerilog to scan a 4x4 matrix keypad and display the last two hex keys pressed on a dual 7 segment display.

Current Goal: Write a synthesizable SystemVerilog module that produces a one-shot registration signal for the keypad system. Behavior:

When a key press is first detected, capture the current key code and assert a single-cycle "new key" pulse.

While any key remains pressed, do not accept additional keys.

Only after keys are released should a subsequent press be recognized.

This should handle debouncing of the keys.

Implement as a small synchronous FSM with enumerated states and glitch-free outputs. Keep names and interfaces reasonable; do not assume any hidden modules beyond what you define here.

## Response:

Here is a synthesizable SystemVerilog module for the one-shot keypad registration FSM.

This design implements the state machine you described, including debounce logic for both key presses and releases. The outputs are registered or derived combinationally from registered state, ensuring they are glitch-free.

I'll provide the file for you.

> **key_registrar.sv** - Nov 3, 12:19 AM

This module should integrate cleanly with your keypad scanner and the 7-segment display driver. The `new_key_pulse` signal acts as the "register enable" for your logic that stores the last two key presses.

## Prompt:

Target device: Lattice iCE40 UP5K FPGA.

Write a synthesizable SystemVerilog module that cycles through keypad columns (active-low, one at a time) and samples rows (active-low) to detect a single key at a time. Behavior:

Iterate columns at a suitable scan rate derived from the divided clock and sample rows.

When a key is detected, report a stable key code consistent with a standard 4x4 keypad layout and maintain it while the key remains pressed.

Provide a boolean signal indicating whether any key is currently pressed.

Use clean state encoding and synchronous logic; avoid combinational feedback and latches.

## Response:

Here is a synthesizable SystemVerilog module for the 4x4 keypad scanner.

This design creates a clock divider to set a scan rate, then uses that to drive a small state machine. The state machine iterates through the columns one by one. For each column, it samples the row inputs (after synchronizing them) to check for a key press.

The module maintains the state of the *first* key it detects during a full scan cycle. The outputs ( `key_code` and `key_is_pressed` ) are registered and update once per full scan (~1ms), providing a stable, bounce-free signal to the `key_registrar` module you already have.

I'll create the new file `keypad_scanner.sv` for this logic.

> **keypad_scanner.sv** - Nov 3, 12:16 AM

## Prompt:

Target device: Lattice iCE40 UP5K FPGA with internal oscillator as the root clock source.

Write a top-level SystemVerilog module that instantiates the scanner and one-shot modules, shifts the last two keys (older ← most recent; most recent ← new), and drives a multiplexed two-digit seven-segment display. Requirements:

Update the displayed digits only when a new key is registered.

Ensure equal apparent brightness and no visible flicker.

Keep all logic synthesizable and synchronous; use idiomatic SystemVerilog constructs.

Provide any small clock-enable or divider logic you need. You can also assume that a sevenSegment module exists that takes a 4 bit input and outputs the 7 segments.

## Response:

I've created the top-level module to connect your scanner and registrar, and I'm also providing the `key_registrar.sv` file from our earlier conversation, as it's needed for the top-level module to work.

The new file, `keypad_top.sv`, instantiates the iCE40's internal 12MHz oscillator, the scanner, and the registrar. It then adds the logic for a 2-digit shift register that updates *only* when you press a new key.

Finally, it includes a high-frequency multiplexer to drive the two 7-segment digits, ensuring a bright, flicker-free display.

Here are the files:

> **keypad_top.sv** - Nov 3, 12:19 AM

---