



**California Polytechnic State University Pomona**

**DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING**

**Dgtl Circuit Dsgn Verilog Lab**

**ECE 3300L Section E01**

**Report # 5**

**Prepared by**

**Arvin Ghaloosian (017153422)**

**Vittorio Huizar (016826784)**

**Group H**

**Presented to**

**Mohamed Aly (Mohamed El-Hadedy)**

**Due - July 21, 2025**

## DESIGN OVERVIEW:

We built a tiny digital odometer that goes from 00 to 99, can run forwards or backwards, and lets us choose the speed with five slide-switches. The board gives us a 100 MHz crystal clock, way too fast to watch. So we built a clock divider which turns that into a 32-bit binary counter that's basically counting nanoseconds. Then we built a 32x1 mux which grabs one bit from that counter to control speed. In the top module we detect the rising edge of that bit and call it a tick. Every tick means "advance the ones digit by one." We have two counters do the math and the seg7\_scan.v flicks the right hand two digits of the seven-segment display on and off really fast so our eyes think both are lit at the same time.

---

## MODULE HIERARCHY:

- bcd\_updown\_top.v
    - tick\_generator
    - bcd\_digit
    - Dual\_bcd\_counter
    - sevenseg\_driver
- 

## Top Module:

### Explanation:

This is the master coordinator that stitches everything together and connects to the board pins named in the .xdc file.

- BTN0 (active-low) resets the whole thing to 00.
  - A tiny edge detector flips the dir flag each time you press BTN1.
  - Instantiates tick\_generator so SW[4:0] pick the speed.
  - Feeds the tick and dir into dual\_bcd\_counter to get units and tens.
  - Pipes those digits into sevenseg\_driver; only the two right-most 7-seg digits are used, the others stay dark.
  - Mirrors the units nibble on LED3-0, tens on LED7-4, and shows the raw switch setting on LED12-8 so you can see the speed tap you chose.
- 

## Tick\_generator Module:

**Explanation:**

This module is our speed control. It keeps a 32-bit counter that races along at the board's 100 MHz clock. Because each bit in a binary counter flips at half the rate of the bit to its right, we can pick any bit to get a slower rhythm. The five slide-switches (SW[4:0]) decide which bit we tap: switch = 0 chooses the slowest bit (bit 31), switch = 31 grabs the fastest bit (bit 0). We then watch that chosen bit and spit out a one clock cycle "tick" pulse every time it rises, that tick drives the rest of the design.

---

**bcd\_digit Module:****Explanation:**

It stores a single decimal digit (0-9). When the enable input goes high for one clock, the wheel moves exactly one step. If dir is 1 we count up; if dir is 0 we count down. When we wrap from 9→0 (going up) or 0→9 (going down), the module raises rollout for one cycle. Reset (rst\_n) instantly forces the number to 0. Internally the code just checks the state and decides what the next value and rollout should be.

---

**dual\_bcd\_counter Module:****Explanation:**

This is simply two bcd\_digit's put together for direction purposes.

---

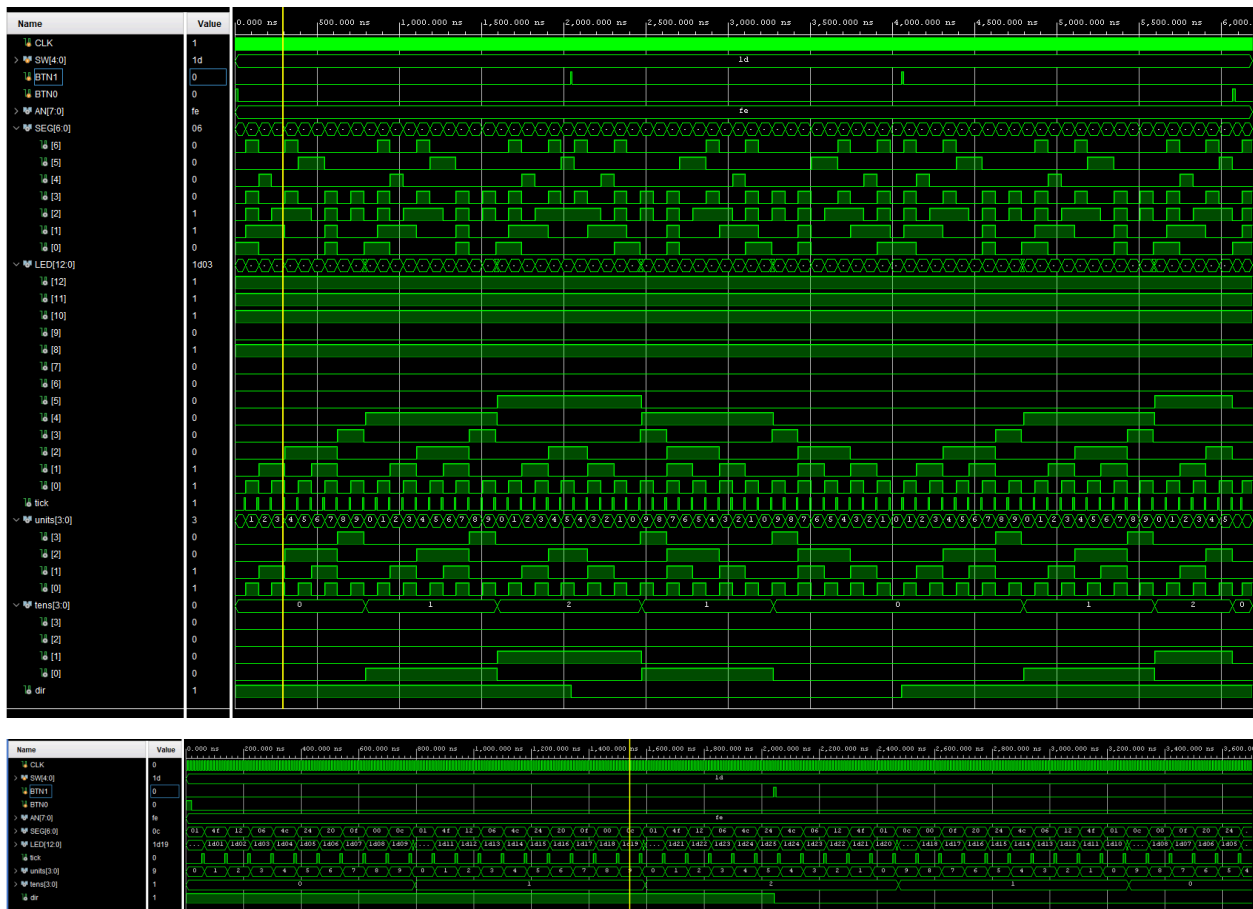
**sevensseg\_driver Module:****Explanation:**

FPGAs save pins by time-sharing the 7-segment display. This driver flicks between the units and tens digits so quickly your eyes blend them together. A tiny 16-bit prescaler inside produces a toggle around 1.5 kHz per digit. On one phase we light the right-hand digit with the units pattern; on the other phase we light the left-hand digit with the tens pattern. A case lookup (the seg\_lut function) converts each 4-bit BCD nibble into the seven LED segments, and we remember the Nexys A7 segments are active-low (0 means "on").

---

**XDC Snippet:**

TEST BENCH:





Changing direction to count down...

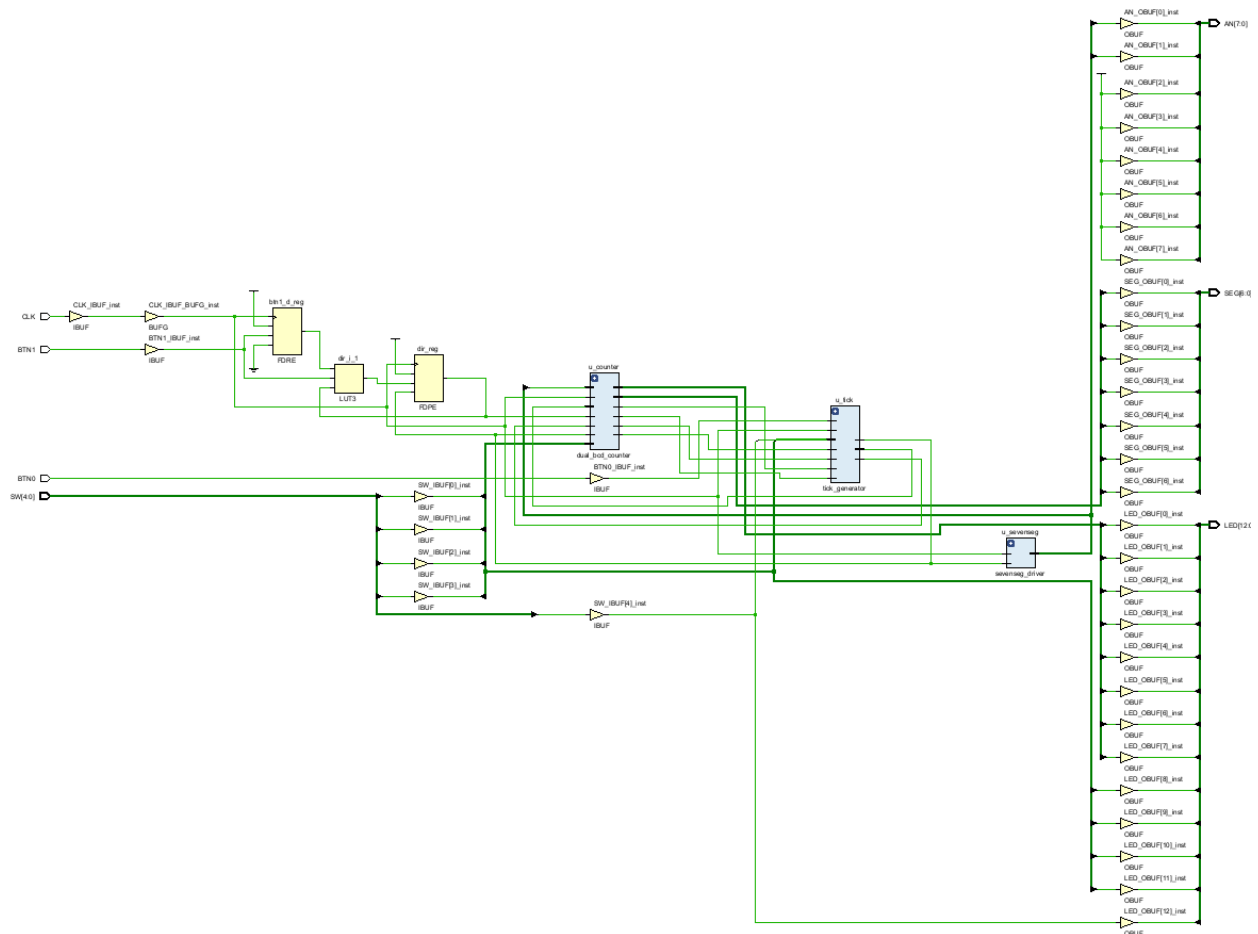
Time=2040000		BTN0=0, BTN1=1, SW=29		dir=1, tick=0		tens= 2, units= 5		LED=ld25
Time=2045000		BTN0=0, BTN1=1, SW=29		dir=0, tick=0		tens= 2, units= 5		LED=ld25
Time=2050000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 2, units= 5		LED=ld25
Time=2055000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 2, units= 5		LED=ld25
Time=2065000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 2, units= 4		LED=ld24
Time=2135000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 2, units= 4		LED=ld24
Time=2145000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 2, units= 3		LED=ld23
Time=2215000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 2, units= 3		LED=ld23
Time=2225000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 2, units= 2		LED=ld22
Time=2295000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 2, units= 2		LED=ld22
Time=2305000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 2, units= 1		LED=ld21
Time=2375000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 2, units= 1		LED=ld21
Time=2385000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 2, units= 0		LED=ld20
Time=2455000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 2, units= 0		LED=ld20
Time=2465000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 2, units= 9		LED=ld29
Time=2475000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 1, units= 9		LED=ld19
Time=2535000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 1, units= 9		LED=ld19
Time=2545000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 1, units= 8		LED=ld18
Time=2615000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 1, units= 8		LED=ld18
Time=2625000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 1, units= 7		LED=ld17
Time=2695000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 1, units= 7		LED=ld17
Time=2705000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 1, units= 6		LED=ld16
Time=2775000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 1, units= 6		LED=ld16
Time=2785000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 1, units= 5		LED=ld15
Time=2855000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 1, units= 5		LED=ld15
Time=2865000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 1, units= 4		LED=ld14
Time=2935000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 1, units= 4		LED=ld14
Time=2945000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 1, units= 3		LED=ld13
Time=3015000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 1, units= 3		LED=ld13
Time=3025000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 1, units= 2		LED=ld12
Time=3095000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 1, units= 2		LED=ld12
Time=3105000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 1, units= 1		LED=ld11
Time=3175000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 1, units= 1		LED=ld11
Time=3185000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 1, units= 0		LED=ld10
Time=3255000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 1, units= 0		LED=ld10
Time=3265000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 1, units= 9		LED=ld19
Time=3275000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 0, units= 9		LED=ld09
Time=3335000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 0, units= 9		LED=ld09
Time=3345000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 0, units= 8		LED=ld08
Time=3415000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 0, units= 8		LED=ld08
Time=3425000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 0, units= 7		LED=ld07
Time=3495000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 0, units= 7		LED=ld07
Time=3505000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 0, units= 6		LED=ld06
Time=3575000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 0, units= 6		LED=ld06
Time=3585000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 0, units= 5		LED=ld05
Time=3655000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 0, units= 5		LED=ld05
Time=3665000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 0, units= 4		LED=ld04
Time=3735000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 0, units= 4		LED=ld04
Time=3745000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 0, units= 3		LED=ld03
Time=3815000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 0, units= 3		LED=ld03
Time=3825000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 0, units= 2		LED=ld02
Time=3895000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 0, units= 2		LED=ld02
Time=3905000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 0, units= 1		LED=ld01
Time=3975000		BTN0=0, BTN1=0, SW=29		dir=0, tick=1		tens= 0, units= 1		LED=ld01
Time=3985000		BTN0=0, BTN1=0, SW=29		dir=0, tick=0		tens= 0, units= 0		LED=ld00

## Explanation:

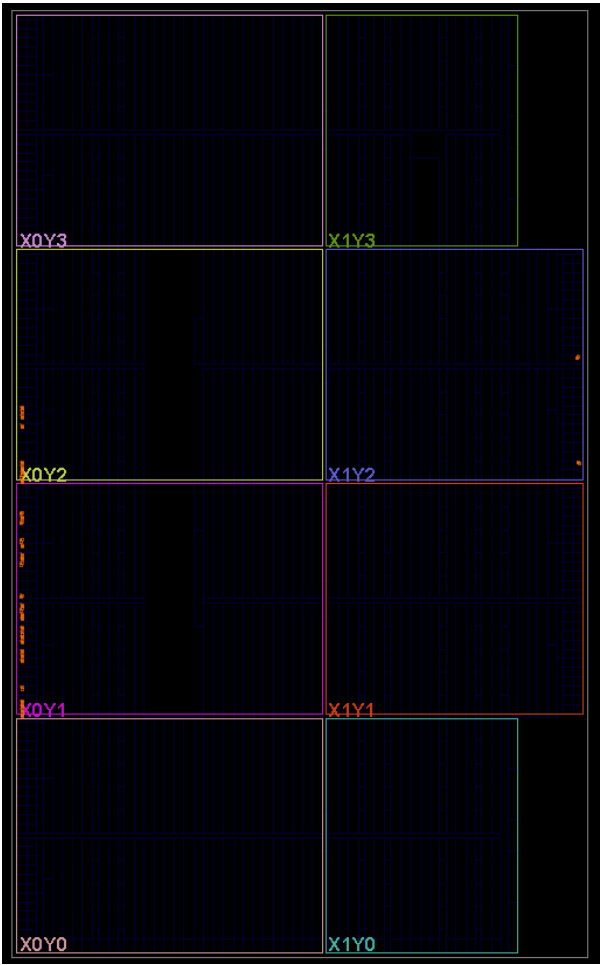
The bcd\_updown\_top\_tb testbench is designed to simulate a user interacting with the BCD

counter. It begins by generating a 100 MHz clock and then starts a sequence of timed events to test the module's functionality. First, it triggers a system wide reset by activating BTN0, ensuring the counter starts in a known state (zero, set to count up). After releasing the reset, the simulation allows the counter to increment for a period. It then tests the direction-change logic by simulating a press of BTN1, which causes the dir signal to flip and the counter to begin decrementing. After another delay, it simulates a second press of BTN1 to verify that the counter can switch back to counting up. Finally, it asserts the reset again to confirm that the circuit can be returned to its initial state at any time.

## SCHEMATIC:



SYNTHESIS SCHEMATIC:



RESOURCE UTILIZATION:

Site Type	Used	Fixed	Prohibited	Available	Util%
Slice LUTs*	35	0	0	63400	0.06
LUT as Logic	35	0	0	63400	0.06
LUT as Memory	0	0	0	19000	0.00
Slice Registers	60	0	0	126800	0.05
Register as Flip Flop	60	0	0	126800	0.05
Register as Latch	0	0	0	126800	0.00
F7 Muxes	1	0	0	31700	<0.01
F8 Muxes	0	0	0	15850	0.00



---

## CONTRIBUTIONS:

### Arvin Ghaloosian (50%)

- Wrote and verified simulation code
- Handled on-board testing and debugging
- Implemented LED mirroring and switch logic

### Vittorio Huizar (50%)

- Designed and verified tick\_generator
- Edited constraint file for correct pin mapping
- Verified and edited up/down logic

---

## REFLECTION:

Splitting the job into tiny files (divider, mux, digit, scanner) made each one easy to write and unit-test before wiring them together. We also never realised you can make a clean 1 cycle pulse just by remembering the previous state of a signal which was super handy for tick generation. Picking the prescaler bit felt like guessing at first, now we understand that every left shift doubles the period. One thing we could have implemented to make the design better is button debouncing to ensure the direction change doesn't press multiple times.

---

## Link to demo video:

[https://youtube.com/watch?v=\\_J6T0dxOkh8](https://youtube.com/watch?v=_J6T0dxOkh8)

---

