LED Metronome Peripheral for SCOMP

Eldridge Surianto

Submitted
April 22, 2025

Introduction

This document summarizes the functionality and implementation of the LED Metronome Peripheral for the SCOMP processor. The peripheral enables users to set LED oscillation rate on the DE10-Standard board and select between bouncing and flashing modes through a simple 16-bit accumulator output without modifying the SCOMP architecture.

We prioritized accurate timing and smooth tempo transitions to emulate a real metronome, while minimizing instruction overhead by limiting the BPM and mode inputs to 10 bits, reserving unused bits for future expansions. We also adopted a modular VHDL design to facilitate easy debugging and future expandability.

We maintained consistent progress by extending work sessions beyond scheduled lab hours, established clear weekly goals for each member, and iteratively prototyped our design. Ultimately, our final product not only met all original specifications but also incorporated additional capabilities, delivering a responsive and user-friendly LED enhancement tool.

Device Functionality

The metronome peripheral is a write-only system that enables a SCOMP programmer to set the tempo in beats per minute (BPM) and control the LED display mode through the accumulator. To use the peripheral, a programmer uses the OUT instruction to write the 16-bit accumulator word to the peripheral's designated I/O address (0x20). The word is interpreted as follows:

- Bits 0–8 store the BPM value (1-511 bpm supported).
- Bit 9 toggles between Bouncing mode (LEDs light and dim sequentially left-right and reverse) and Flashing mode (all LEDs blink together).
- Bits 10–15 are unused.

Address	Register Name	B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
0x20	Metronome Speed and Mode	-	-	-	-	-	-	AC_SPD_DATA[9:0]									

Figure 1. Peripheral register map.

For example, executing these instructions causes LEDs to flash at 100 BPM:

```
LOADI 100 ; load 100 into the accumulator
OR Bit9   ; enable "flashing" mode
OUT Metronome
Bit9: DW &B1000000000
Metronome: EQU &H20
```

Design Decisions and Implementation

The information flow of the peripheral can be defined as the sequence of 5 steps, as seen in the figure below.

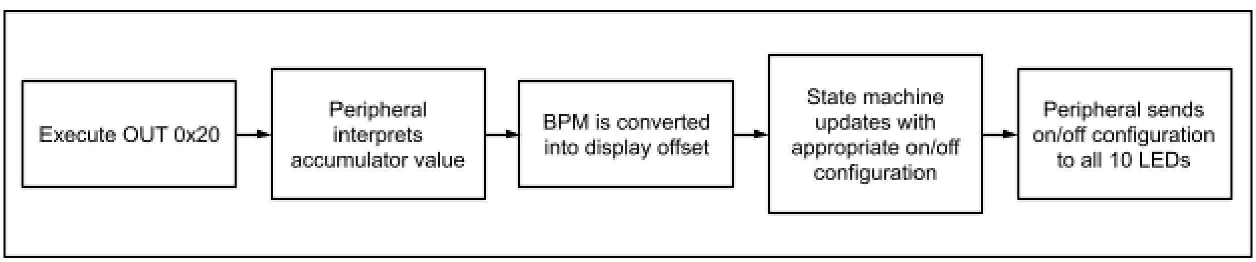

Figure 2. Peripheral information flow.

We limited the BPM input to the lower 9 bits of the accumulator; BPM values higher than 511 are not practical. This allowed a straightforward interface without burdening the user with complicated formatting. This simplified hardware decoding, since parsing only 9 bits, enabled a compact, efficient VHDL implementation.

Another major decision was offering two distinct lighting modes: Bouncing and Flashing. We used bit 9 as a switch to toggle between modes. This added flexibility with minimal complexity, opening up a broader range of potential applications using the SCOMP assembly code while maintaining a simple API.

We additionally prioritized modularity in VHDL design, which facilitated easy testing, debugging, and prevented hardware redundancy in later additions like the flashing mode. The code was separated into two processes, one calculated divisions while the other updated the LEDs, both based on the 12 MHz clock.

A final aspect that we focused on was tempo accuracy and smooth lighting. When changing the BPM, the state machine continues at the current brightness/LED level. This allows for continuous transitions into different speeds.

Conclusions

The LED Metronome Peripheral successfully fulfilled its purpose by providing a user-friendly and precise method for visualizing tempo control through LEDs in a SCOMP environment. Its intuitive and modular design serves as a foundation for future enhancements, with unused accumulator bits offering opportunities to add features such as setting metronome duration or limiting cycle counts. With the simplicity of the API, future engineers have flexibility to incorporate the peripheral into diverse applications, such as visual timers or network monitors. If I were to revisit the project, I would explore adding a BPM tracker that measures user button presses and displays it on the HEX displays—a useful feature in modern metronome apps.