CmpE Internship Summer 2018
Design Doc Template

Dial Controlled Flashlight

Authors & Editors:

Zackery K. Plovanic

Revision History:

Revision	Date	Description
A.2	**7/13/2018**	Clarified Design, added pseudocode for functionality.
	6/30/2018	Approved by Vincent.
	6/26/2018	Vincent set as reviewer. Approval pending.
A.1	**6/16/2018**	Initial Commit

Table of Contents

Objective

In this project I aim to create a dial-controlled flashlight. The system is composed of a series of LEDs that are controlled/activated depending on the current position of a rotary encoder. As this rotary encoder is turned, the LEDs turn on/off as the encoded signal changes.

Background

Input Device: Rotary Encoder - https://reference.digilentinc.com/reference/pmod/pmodenc/reference-manual

Output Device: Eight High Brightness LEDs - https://reference.digilentinc.com/reference/pmod/pmod8ld/reference-manual

Overview

For this project, the architecture for the system as a whole needs to be designed. This includes integrating a MIPS processor with memory devices, ROM and RAM, as well as the I/O devices that we have selected, a rotary encoder and high-brightness LEDs. These devices need to be designed to prevent any possibility of bus contention that can damage the system. In addition to this all, a clock generator is needed to slow the native clock of the Nexys 4 to allow the MIPS processor sufficient time to execute its instructions.

To allow the system architecture to interact with my rotary encoder, a quadrature encoder is needed to track the position of the PMOD device. Aside from that, no additional architecture is needed to interface with the PMOD devices since they communicated via GPIO, rather than separate protocols like the SPI devices.

In addition to the hardware design, I will also need to write the assembly code to constantly read the rotary encoder and write the corresponding value to the LEDs. This needs to be done using the MIPS instruction set. Once that program is written, I will take that hex code and place it into the ROM to be executed.

Detailed Design

My system uses memory mapped IO to interface with the PMOD devices. The system's addressing is as follows:

Device	31-16	15	14	13-12	11	10	9-0	Equation
RAM	0	X	0	X	X	X	X	NOR(31-16)*~1)
Input 0	0	X	1	X	0	0	X	NOR(31-16)*14 *~11*~10
Input 1	0	X	1	X	0	1	X	NOR(31-16)*14 *~11*10
Output 0	0	X	1	X	1	0	X	NOR(31-16)*14 *11 *~10
Output1	0	X	1	X	1	1	X	NOR(31-16)*14 *11 *10

Table 1:

The system is designed in such a way that ensures mutual exclusivity. Address lines 31-16 need to be low in order to interface with the system. Line 14 selects whether the processor is accessing its RAM or an IO device. If an IO device is in fact being accessed, lines 10&11 determine which specific IO device is in use. The entirety of the system can be seen here: https://drive.google.com/file/d/1uP8uKzEfXDxVvkTnBYWsJRX8j2orhl_X/view?usp=sharing.

The pseudocode of the program to be stored in side the ROM would be have the following logic:

```
Int temp = 0;
While(1)
{
Temp = getEncoderValue();
setLED(temp);
```

```
}
```

Caveats

-

Testing Plan

Unit testing scheme

- Simulate the processor accessing memory locations to ensure that each device is mutually exclusive.
- Step through the program code via simulation to ensure that functionality is as intended.

Integration Testing

-

Demonstration Project

N/A