

# Baylor University

Department of  
Electrical and Computer Engineering

ELC 5311 (graduates), 4396(undergrad)  
Advanced Digital Logic Laboratories

Keith Schubert  
Associate Professor  
Department of Electrical and Computer Engineering  
Baylor University



# Contents

<b>1</b>	<b>Basys 2 Programming</b>	<b>3</b>
1.1	Project Setup . . . . .	3
1.2	Passthrough . . . . .	3
1.3	Making a Counter for the Seven Segment Display . . . . .	9
<b>2</b>	<b>Barrel Shifter</b>	<b>11</b>
<b>3</b>	<b>Stack</b>	<b>13</b>
3.1	Introduction . . . . .	13
3.2	Assignment . . . . .	13
<b>4</b>	<b>PWM Dimmer</b>	<b>15</b>
4.1	Pulse Width Modulation . . . . .	15
4.2	Assignment for all . . . . .	15
4.3	Graduate Student Additional Work . . . . .	15
<b>5</b>	<b>Parking Lot</b>	<b>17</b>
5.1	Is the Lot Full . . . . .	17
5.2	Task . . . . .	17
<b>6</b>	<b>Babbage Difference Engine Emulation Circuit</b>	<b>19</b>
6.1	Newton's Differences . . . . .	19
6.2	Babbage's Difference Engine . . . . .	19
6.3	Graduate Students . . . . .	20
<b>A</b>	<b>Github</b>	<b>21</b>



# Lab 1

## Basys 2 Programming

### 1.1 Project Setup

```
Family      :Artix-7
Sub-Family  :Artix-7
Package     :csg324
Speed Grade: -1
Select "xc7a100tcsg324-1"
```

### 1.2 Passthrough

We are going to begin with a simple project to turn on LEDs when the switch under them is on. There are eight switches (called sw<7> ... sw<0>), and eight LEDs (called Led<7> ... Led<0>). Our first easy part will be to assign the LEDs to be identical to the switches, see Code 1.1.

Listing 1.1: Verilog code for pass-through

```
'timescale 1ns / 1ps
module pass_through_simple(
    input  [7:0] sw,
    output [7:0] Led
);

assign Led=sw;

endmodule
```

This code simply passes the switches through to the LEDs, but it demonstrates the **assign** statement, which is one of our basic ways of designing combinational circuits. The other important thing is to have a Xilinx Design Constraints file (XDC) file. Note before vivado (Xilinx products before version 7 of their FPGAs, i.e. Spartan-6 and below) they used a different file type called a user constraints file (UCF). The big difference is in how timing is handled, which we will get into later in the course, for now I just want you to be aware that there are two standards and XDC should be used on FPGA families with a 7 or later. The book will use a UCF because it is working on a Spartan-3 board. An example of an XDC file is below.

Listing 1.2: Xilinx Design Constraints file (XDC)

```
## This file is a general .xdc for the Nexys4 DDR Rev. C
```

```

## To use it in a project:
## - uncomment the lines corresponding to used pins
## - rename the used ports (in each line, after get_ports) according to the top level sign

## Clock signal
#set_property -dict { PACKAGE_PIN E3      IOSTANDARD LVCMOS33 } [get_ports { CLK100MHZ }]; #
#create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports {CLK100MHZ}]

##Switches

#set_property -dict { PACKAGE_PIN J15      IOSTANDARD LVCMOS33 } [get_ports { SW[0] }]; #IO-L
#set_property -dict { PACKAGE_PIN L16      IOSTANDARD LVCMOS33 } [get_ports { SW[1] }]; #IO-L3
#set_property -dict { PACKAGE_PIN M13      IOSTANDARD LVCMOS33 } [get_ports { SW[2] }]; #IO-L
#set_property -dict { PACKAGE_PIN R15      IOSTANDARD LVCMOS33 } [get_ports { SW[3] }]; #IO-L1
#set_property -dict { PACKAGE_PIN R17      IOSTANDARD LVCMOS33 } [get_ports { SW[4] }]; #IO-L1
#set_property -dict { PACKAGE_PIN T18      IOSTANDARD LVCMOS33 } [get_ports { SW[5] }]; #IO-L
#set_property -dict { PACKAGE_PIN U18      IOSTANDARD LVCMOS33 } [get_ports { SW[6] }]; #IO-L
#set_property -dict { PACKAGE_PIN R13      IOSTANDARD LVCMOS33 } [get_ports { SW[7] }]; #IO-L
#set_property -dict { PACKAGE_PIN T8       IOSTANDARD LVCMOS18 } [get_ports { SW[8] }]; #IO-L
#set_property -dict { PACKAGE_PIN U8       IOSTANDARD LVCMOS18 } [get_ports { SW[9] }]; #IO-2
#set_property -dict { PACKAGE_PIN R16      IOSTANDARD LVCMOS33 } [get_ports { SW[10] }]; #IO-L
#set_property -dict { PACKAGE_PIN T13      IOSTANDARD LVCMOS33 } [get_ports { SW[11] }]; #IO-L
#set_property -dict { PACKAGE_PIN H6       IOSTANDARD LVCMOS33 } [get_ports { SW[12] }]; #IO-L
#set_property -dict { PACKAGE_PIN U12      IOSTANDARD LVCMOS33 } [get_ports { SW[13] }]; #IO-L
#set_property -dict { PACKAGE_PIN U11      IOSTANDARD LVCMOS33 } [get_ports { SW[14] }]; #IO-L
#set_property -dict { PACKAGE_PIN V10      IOSTANDARD LVCMOS33 } [get_ports { SW[15] }]; #IO-L

## LEDs

#set_property -dict { PACKAGE_PIN H17      IOSTANDARD LVCMOS33 } [get_ports { LED[0] }]; #IO-L
#set_property -dict { PACKAGE_PIN K15      IOSTANDARD LVCMOS33 } [get_ports { LED[1] }]; #IO-L
#set_property -dict { PACKAGE_PIN J13      IOSTANDARD LVCMOS33 } [get_ports { LED[2] }]; #IO-L
#set_property -dict { PACKAGE_PIN N14      IOSTANDARD LVCMOS33 } [get_ports { LED[3] }]; #IO-L
#set_property -dict { PACKAGE_PIN R18      IOSTANDARD LVCMOS33 } [get_ports { LED[4] }]; #IO-L
#set_property -dict { PACKAGE_PIN V17      IOSTANDARD LVCMOS33 } [get_ports { LED[5] }]; #IO-L
#set_property -dict { PACKAGE_PIN U17      IOSTANDARD LVCMOS33 } [get_ports { LED[6] }]; #IO-L
#set_property -dict { PACKAGE_PIN U16      IOSTANDARD LVCMOS33 } [get_ports { LED[7] }]; #IO-L
#set_property -dict { PACKAGE_PIN V16      IOSTANDARD LVCMOS33 } [get_ports { LED[8] }]; #IO-L
#set_property -dict { PACKAGE_PIN T15      IOSTANDARD LVCMOS33 } [get_ports { LED[9] }]; #IO-L
#set_property -dict { PACKAGE_PIN U14      IOSTANDARD LVCMOS33 } [get_ports { LED[10] }]; #IO-L
#set_property -dict { PACKAGE_PIN T16      IOSTANDARD LVCMOS33 } [get_ports { LED[11] }]; #IO-L
#set_property -dict { PACKAGE_PIN V15      IOSTANDARD LVCMOS33 } [get_ports { LED[12] }]; #IO-L
#set_property -dict { PACKAGE_PIN V14      IOSTANDARD LVCMOS33 } [get_ports { LED[13] }]; #IO-L
#set_property -dict { PACKAGE_PIN V12      IOSTANDARD LVCMOS33 } [get_ports { LED[14] }]; #IO-L
#set_property -dict { PACKAGE_PIN V11      IOSTANDARD LVCMOS33 } [get_ports { LED[15] }]; #IO-L

#set_property -dict { PACKAGE_PIN R12      IOSTANDARD LVCMOS33 } [get_ports { LED16_B }]; #IO-L
#set_property -dict { PACKAGE_PIN M16      IOSTANDARD LVCMOS33 } [get_ports { LED16_G }]; #IO-L
#set_property -dict { PACKAGE_PIN N15      IOSTANDARD LVCMOS33 } [get_ports { LED16_R }]; #IO-L

```

```
#set_property -dict { PACKAGE_PIN G14 IOSTANDARD LVCMOS33 } [get_ports { LED17_B }]; #IO-
#set_property -dict { PACKAGE_PIN R11 IOSTANDARD LVCMOS33 } [get_ports { LED17_G }]; #IO-
#set_property -dict { PACKAGE_PIN N16 IOSTANDARD LVCMOS33 } [get_ports { LED17_R }]; #IO-
```

### ##7 segment display

```
#set_property -dict { PACKAGE_PIN T10 IOSTANDARD LVCMOS33 } [get_ports { CA }]; #IO-L24N
#set_property -dict { PACKAGE_PIN R10 IOSTANDARD LVCMOS33 } [get_ports { CB }]; #IO-25-1
#set_property -dict { PACKAGE_PIN K16 IOSTANDARD LVCMOS33 } [get_ports { CC }]; #IO-25-1
#set_property -dict { PACKAGE_PIN K13 IOSTANDARD LVCMOS33 } [get_ports { CD }]; #IO-L17P
#set_property -dict { PACKAGE_PIN P15 IOSTANDARD LVCMOS33 } [get_ports { CE }]; #IO-L13P
#set_property -dict { PACKAGE_PIN T11 IOSTANDARD LVCMOS33 } [get_ports { CF }]; #IO-L19P
#set_property -dict { PACKAGE_PIN L18 IOSTANDARD LVCMOS33 } [get_ports { CG }]; #IO-L4P-
```

```
#set_property -dict { PACKAGE_PIN H15 IOSTANDARD LVCMOS33 } [get_ports { DP }]; #IO-L19N
```

```
#set_property -dict { PACKAGE_PIN J17 IOSTANDARD LVCMOS33 } [get_ports { AN[0] }]; #IO-L
#set_property -dict { PACKAGE_PIN J18 IOSTANDARD LVCMOS33 } [get_ports { AN[1] }]; #IO-L
#set_property -dict { PACKAGE_PIN T9 IOSTANDARD LVCMOS33 } [get_ports { AN[2] }]; #IO-L
#set_property -dict { PACKAGE_PIN J14 IOSTANDARD LVCMOS33 } [get_ports { AN[3] }]; #IO-L
#set_property -dict { PACKAGE_PIN P14 IOSTANDARD LVCMOS33 } [get_ports { AN[4] }]; #IO-L
#set_property -dict { PACKAGE_PIN T14 IOSTANDARD LVCMOS33 } [get_ports { AN[5] }]; #IO-L
#set_property -dict { PACKAGE_PIN K2 IOSTANDARD LVCMOS33 } [get_ports { AN[6] }]; #IO-L
#set_property -dict { PACKAGE_PIN U13 IOSTANDARD LVCMOS33 } [get_ports { AN[7] }]; #IO-L
```

### ##Buttons

```
#set_property -dict { PACKAGE_PIN C12 IOSTANDARD LVCMOS33 } [get_ports { CPU_RESETN }]; #
#set_property -dict { PACKAGE_PIN N17 IOSTANDARD LVCMOS33 } [get_ports { BTNC }]; #IO-L9
#set_property -dict { PACKAGE_PIN M18 IOSTANDARD LVCMOS33 } [get_ports { BTNU }]; #IO-L4
#set_property -dict { PACKAGE_PIN P17 IOSTANDARD LVCMOS33 } [get_ports { BTNL }]; #IO-L12
#set_property -dict { PACKAGE_PIN M17 IOSTANDARD LVCMOS33 } [get_ports { BTNR }]; #IO-L1
#set_property -dict { PACKAGE_PIN P18 IOSTANDARD LVCMOS33 } [get_ports { BTND }]; #IO-L9
```

### ##Pmod Headers

#### ##Pmod Header JA

```
#set_property -dict { PACKAGE_PIN C17 IOSTANDARD LVCMOS33 } [get_ports { JA[1] }]; #IO-L
#set_property -dict { PACKAGE_PIN D18 IOSTANDARD LVCMOS33 } [get_ports { JA[2] }]; #IO-L
#set_property -dict { PACKAGE_PIN E18 IOSTANDARD LVCMOS33 } [get_ports { JA[3] }]; #IO-L
#set_property -dict { PACKAGE_PIN G17 IOSTANDARD LVCMOS33 } [get_ports { JA[4] }]; #IO-L
#set_property -dict { PACKAGE_PIN D17 IOSTANDARD LVCMOS33 } [get_ports { JA[7] }]; #IO-L
#set_property -dict { PACKAGE_PIN E17 IOSTANDARD LVCMOS33 } [get_ports { JA[8] }]; #IO-L
#set_property -dict { PACKAGE_PIN F18 IOSTANDARD LVCMOS33 } [get_ports { JA[9] }]; #IO-L
#set_property -dict { PACKAGE_PIN G18 IOSTANDARD LVCMOS33 } [get_ports { JA[10] }]; #IO-L
```

### ##Pmod Header JB

```
#set_property -dict { PACKAGE_PIN D14 IOSTANDARD LVCMOS33 } [get_ports { JB[1] }]; #IO-L
#set_property -dict { PACKAGE_PIN F16 IOSTANDARD LVCMOS33 } [get_ports { JB[2] }]; #IO-L
#set_property -dict { PACKAGE_PIN G16 IOSTANDARD LVCMOS33 } [get_ports { JB[3] }]; #IO-L
#set_property -dict { PACKAGE_PIN H14 IOSTANDARD LVCMOS33 } [get_ports { JB[4] }]; #IO-L
#set_property -dict { PACKAGE_PIN E16 IOSTANDARD LVCMOS33 } [get_ports { JB[7] }]; #IO-L
#set_property -dict { PACKAGE_PIN F13 IOSTANDARD LVCMOS33 } [get_ports { JB[8] }]; #IO-L
#set_property -dict { PACKAGE_PIN G13 IOSTANDARD LVCMOS33 } [get_ports { JB[9] }]; #IO-L
#set_property -dict { PACKAGE_PIN H16 IOSTANDARD LVCMOS33 } [get_ports { JB[10] }]; #IO-L
```

### ##Pmod Header JC

```
#set_property -dict { PACKAGE_PIN K1 IOSTANDARD LVCMOS33 } [get_ports { JC[1] }]; #IO-L
#set_property -dict { PACKAGE_PIN F6 IOSTANDARD LVCMOS33 } [get_ports { JC[2] }]; #IO-L
#set_property -dict { PACKAGE_PIN J2 IOSTANDARD LVCMOS33 } [get_ports { JC[3] }]; #IO-L
#set_property -dict { PACKAGE_PIN G6 IOSTANDARD LVCMOS33 } [get_ports { JC[4] }]; #IO-L
#set_property -dict { PACKAGE_PIN E7 IOSTANDARD LVCMOS33 } [get_ports { JC[7] }]; #IO-L
#set_property -dict { PACKAGE_PIN J3 IOSTANDARD LVCMOS33 } [get_ports { JC[8] }]; #IO-L
#set_property -dict { PACKAGE_PIN J4 IOSTANDARD LVCMOS33 } [get_ports { JC[9] }]; #IO-L
#set_property -dict { PACKAGE_PIN E6 IOSTANDARD LVCMOS33 } [get_ports { JC[10] }]; #IO-L
```

### ##Pmod Header JD

```
#set_property -dict { PACKAGE_PIN H4 IOSTANDARD LVCMOS33 } [get_ports { JD[1] }]; #IO-L
#set_property -dict { PACKAGE_PIN H1 IOSTANDARD LVCMOS33 } [get_ports { JD[2] }]; #IO-L
#set_property -dict { PACKAGE_PIN G1 IOSTANDARD LVCMOS33 } [get_ports { JD[3] }]; #IO-L
#set_property -dict { PACKAGE_PIN G3 IOSTANDARD LVCMOS33 } [get_ports { JD[4] }]; #IO-L
#set_property -dict { PACKAGE_PIN H2 IOSTANDARD LVCMOS33 } [get_ports { JD[7] }]; #IO-L
#set_property -dict { PACKAGE_PIN G4 IOSTANDARD LVCMOS33 } [get_ports { JD[8] }]; #IO-L
#set_property -dict { PACKAGE_PIN G2 IOSTANDARD LVCMOS33 } [get_ports { JD[9] }]; #IO-L
#set_property -dict { PACKAGE_PIN F3 IOSTANDARD LVCMOS33 } [get_ports { JD[10] }]; #IO-L
```

### ##Pmod Header JXADC

```
#set_property -dict { PACKAGE_PIN A14 IOSTANDARD LVDS } [get_ports { XA_N[1] }]; #IO-L
#set_property -dict { PACKAGE_PIN A13 IOSTANDARD LVDS } [get_ports { XA_P[1] }]; #IO-L
#set_property -dict { PACKAGE_PIN A16 IOSTANDARD LVDS } [get_ports { XA_N[2] }]; #IO-L
#set_property -dict { PACKAGE_PIN A15 IOSTANDARD LVDS } [get_ports { XA_P[2] }]; #IO-L
#set_property -dict { PACKAGE_PIN B17 IOSTANDARD LVDS } [get_ports { XA_N[3] }]; #IO-L
#set_property -dict { PACKAGE_PIN B16 IOSTANDARD LVDS } [get_ports { XA_P[3] }]; #IO-L
#set_property -dict { PACKAGE_PIN A18 IOSTANDARD LVDS } [get_ports { XA_N[4] }]; #IO-L
#set_property -dict { PACKAGE_PIN B18 IOSTANDARD LVDS } [get_ports { XA_P[4] }]; #IO-L
```

### ##VGA Connector



```

#set_property -dict { PACKAGE_PIN A3      IOSTANDARD LVCMOS33 } [get_ports { VGA_R[0] }]; #IO
#set_property -dict { PACKAGE_PIN B4      IOSTANDARD LVCMOS33 } [get_ports { VGA_R[1] }]; #IO
#set_property -dict { PACKAGE_PIN C5      IOSTANDARD LVCMOS33 } [get_ports { VGA_R[2] }]; #IO
#set_property -dict { PACKAGE_PIN A4      IOSTANDARD LVCMOS33 } [get_ports { VGA_R[3] }]; #IO

#set_property -dict { PACKAGE_PIN C6      IOSTANDARD LVCMOS33 } [get_ports { VGA_G[0] }]; #IO
#set_property -dict { PACKAGE_PIN A5      IOSTANDARD LVCMOS33 } [get_ports { VGA_G[1] }]; #IO
#set_property -dict { PACKAGE_PIN B6      IOSTANDARD LVCMOS33 } [get_ports { VGA_G[2] }]; #IO
#set_property -dict { PACKAGE_PIN A6      IOSTANDARD LVCMOS33 } [get_ports { VGA_G[3] }]; #IO

#set_property -dict { PACKAGE_PIN B7      IOSTANDARD LVCMOS33 } [get_ports { VGA_B[0] }]; #IO
#set_property -dict { PACKAGE_PIN C7      IOSTANDARD LVCMOS33 } [get_ports { VGA_B[1] }]; #IO
#set_property -dict { PACKAGE_PIN D7      IOSTANDARD LVCMOS33 } [get_ports { VGA_B[2] }]; #IO
#set_property -dict { PACKAGE_PIN D8      IOSTANDARD LVCMOS33 } [get_ports { VGA_B[3] }]; #IO

#set_property -dict { PACKAGE_PIN B11     IOSTANDARD LVCMOS33 } [get_ports { VGA_HS }]; #IO
#set_property -dict { PACKAGE_PIN B12     IOSTANDARD LVCMOS33 } [get_ports { VGA_VS }]; #IO

```

### ##Micro SD Connector

```

#set_property -dict { PACKAGE_PIN E2      IOSTANDARD LVCMOS33 } [get_ports { SD_RESET }]; #IO
#set_property -dict { PACKAGE_PIN A1      IOSTANDARD LVCMOS33 } [get_ports { SD_CD }]; #IO
#set_property -dict { PACKAGE_PIN B1      IOSTANDARD LVCMOS33 } [get_ports { SD_SCK }]; #IO
#set_property -dict { PACKAGE_PIN C1      IOSTANDARD LVCMOS33 } [get_ports { SD_CMD }]; #IO
#set_property -dict { PACKAGE_PIN C2      IOSTANDARD LVCMOS33 } [get_ports { SD_DAT[0] }]; #
#set_property -dict { PACKAGE_PIN E1      IOSTANDARD LVCMOS33 } [get_ports { SD_DAT[1] }]; #
#set_property -dict { PACKAGE_PIN F1      IOSTANDARD LVCMOS33 } [get_ports { SD_DAT[2] }]; #
#set_property -dict { PACKAGE_PIN D2      IOSTANDARD LVCMOS33 } [get_ports { SD_DAT[3] }]; #

```

### ##Accelerometer

```

#set_property -dict { PACKAGE_PIN E15     IOSTANDARD LVCMOS33 } [get_ports { ACL_MISO }]; #IO
#set_property -dict { PACKAGE_PIN F14     IOSTANDARD LVCMOS33 } [get_ports { ACL_MOSI }]; #IO
#set_property -dict { PACKAGE_PIN F15     IOSTANDARD LVCMOS33 } [get_ports { ACL_SCLK }]; #IO
#set_property -dict { PACKAGE_PIN D15     IOSTANDARD LVCMOS33 } [get_ports { ACL_CSN }]; #IO
#set_property -dict { PACKAGE_PIN B13     IOSTANDARD LVCMOS33 } [get_ports { ACL_INT[1] }]; #
#set_property -dict { PACKAGE_PIN C16     IOSTANDARD LVCMOS33 } [get_ports { ACL_INT[2] }]; #

```

### ##Temperature Sensor

```

#set_property -dict { PACKAGE_PIN C14     IOSTANDARD LVCMOS33 } [get_ports { TMP_SCL }]; #IO
#set_property -dict { PACKAGE_PIN C15     IOSTANDARD LVCMOS33 } [get_ports { TMP_SDA }]; #IO
#set_property -dict { PACKAGE_PIN D13     IOSTANDARD LVCMOS33 } [get_ports { TMP_INT }]; #IO
#set_property -dict { PACKAGE_PIN B14     IOSTANDARD LVCMOS33 } [get_ports { TMP_CT }]; #IO

```

### ##Omnidirectional Microphone

```
#set_property -dict { PACKAGE_PIN J5      IOSTANDARD LVCMOS33 } [get_ports { M_CLK }]; #IO-2
#set_property -dict { PACKAGE_PIN H5      IOSTANDARD LVCMOS33 } [get_ports { M_DATA }]; #IO-1
#set_property -dict { PACKAGE_PIN F5      IOSTANDARD LVCMOS33 } [get_ports { M_LRSEL }]; #IO-0
```

### ##PWM Audio Amplifier

```
#set_property -dict { PACKAGE_PIN A11     IOSTANDARD LVCMOS33 } [get_ports { AUD_PWM }]; #IO-0
#set_property -dict { PACKAGE_PIN D12     IOSTANDARD LVCMOS33 } [get_ports { AUD_SD }]; #IO-1
```

### ##USB-RS232 Interface

```
#set_property -dict { PACKAGE_PIN C4      IOSTANDARD LVCMOS33 } [get_ports { UART_TXD_IN }];
#set_property -dict { PACKAGE_PIN D4      IOSTANDARD LVCMOS33 } [get_ports { UART_RXD_OUT }];
#set_property -dict { PACKAGE_PIN D3      IOSTANDARD LVCMOS33 } [get_ports { UART_CTS }]; #IO-0
#set_property -dict { PACKAGE_PIN E5      IOSTANDARD LVCMOS33 } [get_ports { UART_RTS }]; #IO-1
```

### ##USB HID (PS/2)

```
#set_property -dict { PACKAGE_PIN F4      IOSTANDARD LVCMOS33 } [get_ports { PS2_CLK }]; #IO-0
#set_property -dict { PACKAGE_PIN B2      IOSTANDARD LVCMOS33 } [get_ports { PS2_DATA }]; #IO-1
```

### ##SMSC Ethernet PHY

```
#set_property -dict { PACKAGE_PIN C9      IOSTANDARD LVCMOS33 } [get_ports { ETH_MDC }]; #IO-0
#set_property -dict { PACKAGE_PIN A9      IOSTANDARD LVCMOS33 } [get_ports { ETH_MDIO }]; #IO-1
#set_property -dict { PACKAGE_PIN B3      IOSTANDARD LVCMOS33 } [get_ports { ETH_RSTN }]; #IO-2
#set_property -dict { PACKAGE_PIN D9      IOSTANDARD LVCMOS33 } [get_ports { ETH_CRSDV }]; #IO-3
#set_property -dict { PACKAGE_PIN C10     IOSTANDARD LVCMOS33 } [get_ports { ETH_RXERR }]; #IO-4
#set_property -dict { PACKAGE_PIN C11     IOSTANDARD LVCMOS33 } [get_ports { ETH_RXD[0] }]; #IO-5
#set_property -dict { PACKAGE_PIN D10     IOSTANDARD LVCMOS33 } [get_ports { ETH_RXD[1] }]; #IO-6
#set_property -dict { PACKAGE_PIN B9      IOSTANDARD LVCMOS33 } [get_ports { ETH_TXEN }]; #IO-7
#set_property -dict { PACKAGE_PIN A10     IOSTANDARD LVCMOS33 } [get_ports { ETH_TXD[0] }]; #IO-8
#set_property -dict { PACKAGE_PIN A8      IOSTANDARD LVCMOS33 } [get_ports { ETH_TXD[1] }]; #IO-9
#set_property -dict { PACKAGE_PIN D5      IOSTANDARD LVCMOS33 } [get_ports { ETH_REFCLK }]; #IO-10
#set_property -dict { PACKAGE_PIN B8      IOSTANDARD LVCMOS33 } [get_ports { ETH_INTN }]; #IO-11
```

### ##Quad SPI Flash

```
#set_property -dict { PACKAGE_PIN K17     IOSTANDARD LVCMOS33 } [get_ports { QSPI_LDQ[0] }]; #IO-12
#set_property -dict { PACKAGE_PIN K18     IOSTANDARD LVCMOS33 } [get_ports { QSPI_LDQ[1] }]; #IO-13
#set_property -dict { PACKAGE_PIN L14     IOSTANDARD LVCMOS33 } [get_ports { QSPI_LDQ[2] }]; #IO-14
#set_property -dict { PACKAGE_PIN M14     IOSTANDARD LVCMOS33 } [get_ports { QSPI_LDQ[3] }]; #IO-15
#set_property -dict { PACKAGE_PIN L13     IOSTANDARD LVCMOS33 } [get_ports { QSPI_CSN }]; #IO-16
```

## 1.3 Making a Counter for the Seven Segment Display

Now we want to drive the seven segment display. All the cathodes of the LEDs in the same position are hooked together. So for example, all four of the top LEDs' cathodes are hooked together, as are all the cathodes of the upper right, and so on. The top LED is called A or seg<0>, the letters and numbers proceed clockwise, and the middle segment is G or seg<6>. Each digit also has a decimal point, which is called dp, and their cathodes are also connected. The FPGA does sink the current in this case (*Why can it sink what it couldn't source?*), so a zero turns the LED on if the anode was also sourced.

To drive the seven segment display we will then need to convert from a 4 bit binary number to the seven segment display cathode pattern. We will leave the decimal point off. This can be seen in Code 1.3.

Listing 1.3: Verilog code for converting 4 bit binary to sseg.

```
'timescale 1ns / 1ps
module sseg_driver(
    input [3:0] num,
    output reg [0:6] sseg,
    output dp
);
    assign dp = 1;

    always@* begin
        case(num)
            0: sseg <= 7'b0000001;
            1: sseg <= 7'b1001111;
            2: sseg <= 7'b0010010;
            3: sseg <= 7'b0000110;
            4: sseg <= 7'b1001100;
            5: sseg <= 7'b0100100;
            6: sseg <= 7'b0000010;
            7: sseg <= 7'b0001111;
            8: sseg <= 7'b0000000;
            9: sseg <= 7'b0001100;
            10: sseg <= 7'b0001000;
            11: sseg <= 7'b1100000;
            12: sseg <= 7'b0110001;
            13: sseg <= 7'b1000010;
            14: sseg <= 7'b0110000;
            default: sseg <= 7'b0111000;
        endcase
    end
endmodule
```

All the anodes for a digit are hooked together, so since there are four digits there are four anode lines (called an<3> ... an<0>). The output of the fpga is not strong enough to supply all the current for the LEDs, so a pnp transistor is used to switch power (*Why does a pnp switch power better, and an npn switch ground better?*). This means a zero, is needed to turn the pnp on and a 1 will turn it off.

This slicing allows control of 32 LEDs with 12 wires and one fourth can be on simultaneously. Since each LED will be off three fourths of the time, we need to switch them faster than the eye can see, and since the diode's pn junction holds charge for a bit (discharges like a capacitor, why?) and thus continues to glow, and the eye continues to see light for a bit. The Basys 2 board has a 50MHz clock, so we need to count to

about a million, which is 20 bits. We thus need a counter that can count to whatever number of bits we want.

Listing 1.4: Verilog code for counting a parameterized number of bits.

```
'timescale 1ns / 1ps
module counter#(
    parameter BITS=20
)(
    input clk ,
    input reset ,
    output reg [BITS-1:0] count
);

    initial
        count <=0;

    always @(posedge(clk), posedge(reset))
        if(reset)
            count <= 0;
        else
            count <= count+1;

endmodule
```

To show this off, we will have a slow count displaying, with say a count around a second or so. This is about 50, which is close enough to 6 bits for our use. We will thus first make a 26 bit counter, triggering the rotation of the anodes off bits 18 and 19, and the next count off bit 25. We will thus make one more counter, this time with 16 bits so we can drive all four segments. We just need to change our top level module and ucf to reflect this.

## Lab 2

# Barrel Shifter

Do Experiment 3.11.1 Multifunction Barrel Shifter from the book.



## Lab 3

# Stack

### 3.1 Introduction

Stacks are a fundamental data type in computers, forming the basis of push down automata, one of the three basic categories of automata (finite automata, push down automata, Turing equivalent automata).

### 3.2 Assignment

Do Experiment 4.7.7 Stack from the Book.





## Lab 4

# PWM Dimmer

### 4.1 Pulse Width Modulation

Pulse width modulation is a direct way to drive analog devices with a digital output. Since it does not require any additional equipment, it is very popular in a number of applications. The basic idea is similar to a bang-bang control<sup>1</sup>, but unlike typical bang-bang controls the cycle time is set very short so the output is effectively a weighted averaged, since most systems function like a low pass filter. The weighting is due to the duty cycle, which is the fraction of the cycle time that the output is on/high. If the output power of the digital line is  $P$  and the duty cycle is  $d$ , then the effective power is  $P_e = dP$ .

Pulse width modulation is very effective in simple servos, LEDs, and other simple devices that need to be driven by a digital controller. They are particularly common in hobby applications, but also find use in telecommunications, monitors, fans, and pumps. You can time them off the leading edge, trailing edge, or center of the edges, which results in three slightly different versions, but all are identified as PWM versions.

### 4.2 Assignment for all

Do Experiment 4.7.2 PWM and LED Dimmer from the Book.

### 4.3 Graduate Student Additional Work

Incorporate your dimmer in the LED time-multiplexing circuit of 4.5.1. In order to do this, you need more than 8 switches. Since this is only for testing, use switches 0-7 for the original circuit and also use 0-3 for the dimmer. This has duplication but does not produce any issues for demonstration purposes. Note: basic code for the system is available in the starter code from the book available from the class GitHub.

---

<sup>1</sup>Bang-bang controls are best known in thermostats, where the system is on or off, but not in any state in between.



## Lab 5

# Parking Lot

### 5.1 Is the Lot Full

Knowing if a parking lot is full, allows us to advise motorists as well as store usage data to plan for future additions. Since many parking lots were not designed around controlling entrance and exit of vehicles or people, retrofitting a parking lot for a counter becomes an issue. Additionally, many entrances are two-way, creating further complications. You are part of a team that will be building a sensor for retrofitting small parking lots ( $< 100$  cars). As such the expense of cutting through the road to put car sensors is prohibitive, so a light based sensor (see Figure 5.11 in the book) has been selected. In 5.5.3, the method of determining if a car is entering or exiting is outlined, but there are many other exceptions, such as a car starting to enter and then backing up. You may assume the sensors are spaced to be able to identify a car from smaller objects like people or bikes since the latter will only trip one sensor at a time, while cars will trip both. Such a system could be tricked by two people, however, so you may assume for simplicity that the lanes to the parking lot are not high foot traffic areas.

### 5.2 Task

Design, implement, test, and document your solution to 5.5.3 in the textbook. For graduate students: assume there are two entry/exit points so your counter must handle multiple sensor units sending inc and dec signals.



## Lab 6

# Babbage Difference Engine Emulation Circuit

### 6.1 Newton's Differences

Newton's method of differences is foundational to interpolation, where it comes in as divided differences, and as a simplification of calculus. Essential each difference is approximating a derivative. You calculate the differences between successive steps, say  $f(n)$  and  $f(n-1)$  for the values of  $n$  that are in a region of interest. If the differences are constant you are done, if not take the differences of these. Repeat until they become constant. Since differences approximate derivatives, repeatedly performing differences on finite polynomials will eventually result in a constant. For example consider performing Newton's differences on  $f(n) = n^2$ ,

$n$	0	1	2	3	4
$f(n) = n^2$	0	1	4	9	16
$f_1 = f(n) - f(n-1) = n^2 - (n-1)^2 = n^2 - (n^2 - 2n + 1) = 2n - 1$	-	1	3	5	7
$f_2 = f_1(n) - f_1(n-1) = 2n - 1 - (2(n-1) - 1) = 2n - 1 - (2n - 3) = 2$	-	-	2	2	2

Notice after two differences you end up with a constant difference of 2, which is the same as the second derivative of the function. Note this is not an accident, since it is following the leading coefficient. It works for both the points and algebraically due to the leading coefficient being the important part, but note the intermediate terms are not exactly the derivative<sup>1</sup>, but are needed to keep the scales right given the jump between each integer.

You can also try this on  $f(n) = n^3$  (or any other finite polynomial),

$n$	0	1	2	3	4
$f(n) = n^3$	0	1	8	27	64
$f_1 = f(n) - f(n-1) = n^3 - (n-1)^3 = 3n^2 - 3n + 1$	-	1	7	19	37
$f_2 = f_1(n) - f_1(n-1) = 3n^2 - 3n + 1 - (3(n-1)^2 - 3(n-1) + 1) = 6n - 6$	-	-	6	12	18
$f_3 = f_2(n) - f_2(n-1) = (6n - 6) - (6(n-1) - 6) = 6$	-	-	-	6	6

For non-polynomials or infinite polynomials it will keep going but the intermediates are meaningful due to Taylor Series approximation. If you are interested in going further with this take a course on numerics (a much beloved field of mine...), though this is as much as we need.

### 6.2 Babbage's Difference Engine

This is what Babbage implemented in his famous difference engine (and expanded on in the analytic engine), and which Lady Ada Lovelace became the first programmer due to her love of and skill with mathematics.

<sup>1</sup>This leads to the need for Newton to develop infinitesimals to fix it, that in turn led to Calculus

Together this idea brings together three of the great events of computer history, which is why we will build it (something Babbage was not able to due to the cost and time. The process is nicely described in Experiment 6.5.7 Babbage Difference Engine Emulation Circuit from the book. Steps 1-4 are the methodology, step 5 just asks you to redo the design for a second system. Design and implement for both polynomials ( $f(n) = 2n^2 + 3n + 5$  and  $h(n) = n^3 + 2n^2 + 2n + 1$ ). Calculate the algebraic form

### 6.3 Graduate Students

Babbage's difference engine can be generalized to where you can input the parameters for each of the sub-functions. Design a system that would work for any second or third order polynomial by inputting coefficients. Include the design with explanation (you don't have to build it) in your report in addition to the work the undergrads must do.

# Appendix A

## Github

Git is a version control system originally developed by Linus Torvalds to do the Linux OS development. Github is a git web service that I will be using to develop the labs. You can get the manual and code from:  
`https://github.com/KeithEvanSchubert/Advanced\_Digital\_Logic.git`