Basic Calculator/ALU

Austin Barber, Ahmed Saudi, Chris Mooradian, Nadeen Dakermange
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
ajbarber@oakland.edu, ahmedsaudi@oakland.edu, chmoorad@oakland.edu, dakermange@oakland.edu

***Abstract*—** The goal of this project was to design a simple calculator (Basic ALU) using an FPGA. An Arduino and a 4x4 keypad receives inputs from the user and feed those values for processing to the FPGA. The calculator can perform addition, subtraction, multiplication, or division, and the outputs are displayed on a custom 7-segment display.

Introduction

The goal of this project was to create a basic calculator (ALU) capable of handling parallel operations such as addition, subtraction, multiplication, and division of two 2-digit operands. Inputs would be provided by the user through a 4x4 matrix keypad, and a 4-digit output would be displayed on a custom 7-Segment display. Design of the ALU required application of several concepts used in class, namely: combinational (addition, subtraction, multiplication) and synchronous (division) arithmetic circuits, coordination of synchronous and combinational circuits using counters and enables, and multiplexing data bus signals.

In a broader sense, the intention was to design the beginnings of a basic microprocessor. With the addition of a small number of registers and control lines, an ALU readily becomes a useful, if not a basic, programmable device. The VDHL developed for this project will be used for that purpose.

In order to develop a high performance ALU, the project will be divided into two parts. The Arduino is used for handling the input data while we use the FPGA for processing. The capability of the FPGA is to perform parallel operations which increase the overall performance of the ALU.

Methodology

A. Arduino Programming

The first step was to program an Arduino board to read inputs from the keypad and convert them into usable data for a Basys 3 FPGA. The group initially planned to use an Arduino UNO board, but discovered that the larger and more capable Arduino MEGA would be needed to handle the 8 inputs from the keypad and 19 outputs to the FPGA. The Arduino code was written to capture input from the keypad and convert it to a number in BCD or a 2-bit operator signal, which would then be output to the pins of the microcontroller. Digital pins from the Arduino board output 5V and the Basys-3 FPGA allows for input voltages up to 3.3V, as such a logic converter was implemented to prevent damaging the FPGA.

B. VHDL programming

After wiring and programming the Arduino to handle the keypad inputs and activate the appropriate FPGA switches, we needed to implement VHDL code to handle all of the ALU's logic and output a result to a custom made LED 7-segment display. The Arduino MEGA outputs 4 sets of 4 bit BCD, one set for each operand. The VHDL code then concatenates the first 2 sets into one 8 bit binary number, which serves as the first operand, then the second 2 sets into what will become the second operand. The group then implemented a multiplexer that will take the 2 bit operator and send an enable signal to the proper logic unit in the ALU. The addition and subtraction unit is a circuit of 5 full adders used differently to achieve their respective mathematical operation, and the division circuit uses a combination of registers, full adders, and state machines to divide two sets of 2 digit numbers. The appropriate result, selected by the operator, is held in a register until the division circuit is done, and the division FSM outputs a done signal, which prevents the signal from going to the binary to BCD convertor before the division is done, as division takes more steps than any other operator. The outputs of the mentioned Binary to BCD converter, along with the original calculator inputs and the operator, go into a 9 input multiplexor, which multiplexes through what is to be displayed on the 7-segment displays using the outputs of a modulo-5

counter. Figure 5 provides a look at the VHDL logic implemented in this project. Table 1 shows the resource utilization of the proposed design.

Name	Slice LUTs	Slice Registers	Bonded IOB	Block RAM Tile
Utilization	39	31	19	0

Table 1. Resource Utilization of the Design

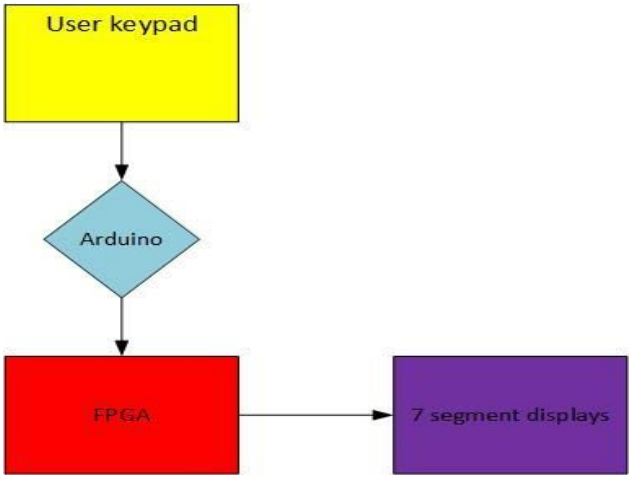


Fig 1. Block Diagram

C. Wiring and Implementation

A crucial step that was taken to ensure the full functionality of the calculator was the necessary wiring of the Arduino Mega, Basys 3 Board, and custom 7-segment display. In wiring the Arduino Mega, PWM pins were assigned for all inputs of the 4x4 matrix keypad. In addition to the keypad, BCD output pins for both operands and operator were connected from the Arduino, to logic converters on a breadboard, and then to input pins on the Basys 3 Board. After the data from the Arduino has been interpreted by the FPGA for a given operation, the inputs and outputs for said operation will need to be displayed on the custom 7-segment display.

The custom 7-segment display contains five 7-segments in full, where each 7-segment has its own common ground. In order to display individual numbers on each 7-segment display, a clock signal from the FPGA will cycle through each ground line, activating them one at a time by completing their circuit. To do this, the base of an NPN BJT for each ground line will be connected to an output pin from the FPGA, where when it outputs a '1', the transistor is activated and thus completes a path to ground for a given 7-segment display. In addition to these outputs from the FPGA, outputs for each individual segment of each 7-segment display will also need to be connected. Each segment of every 7-segment display contains 2 LEDs connected in parallel, as well as a current limiting resistor. After all of these connections have been made, the code will then be responsible for fulfilling the intended functionality of a basic calculator, or ALU.

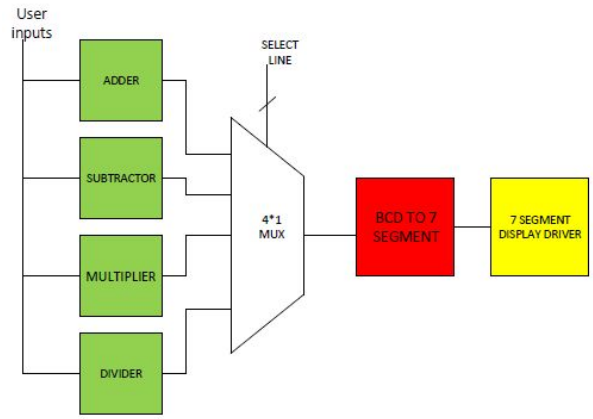


Fig 2. Logical view of implemented design

Experimental Setup

To verify that the Arduino Mega was correctly producing BCD values, each output pin was connected to a single LED on a breadboard. Data was input using the keypad, and BCD data was visually confirmed. Next, the Arduino outputs were wired to a 5V to 3V logic converter, and a multimeter was used to ensure that logic highs were being stepped down to approximately 3.3V. Fig 4 shows the experimental setup of the project

Initially, Vivado's behavioral simulation tool was used to verify that the ALU was functioning properly. Next, the program was implemented on a Basys3 board, and the on-board switches and 7-segment display were used in place of the keypad and custom LED display. Once proper operation was confirmed, the Arduino Mega was connected to the P-mod pins of the Basys3 and the outputs were wired to the custom 7-segment display.

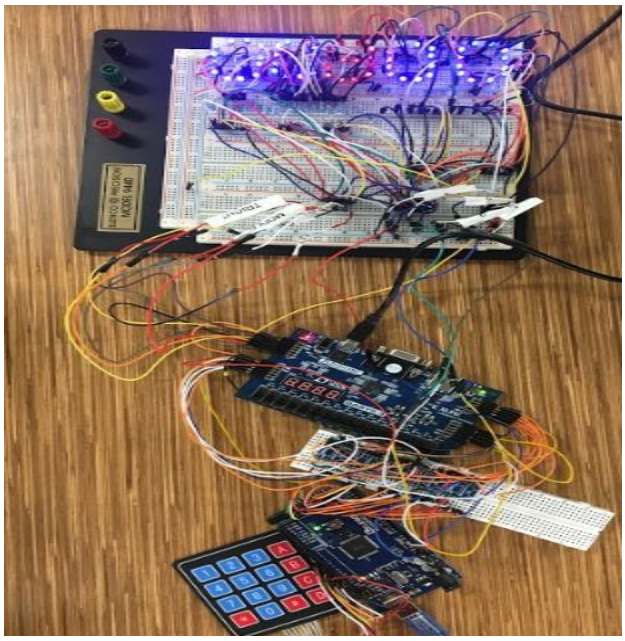
Results

The calculator performed arithmetic as expected, and the custom LED board displayed the inputs and outputs correctly (see video linked below) :

https://www.youtube.com/watch?v=GyH208-_Zk4&feature=youtu.be

The ALU calculates both the quotient and remainder for the division operation, however only the quotient is displayed.

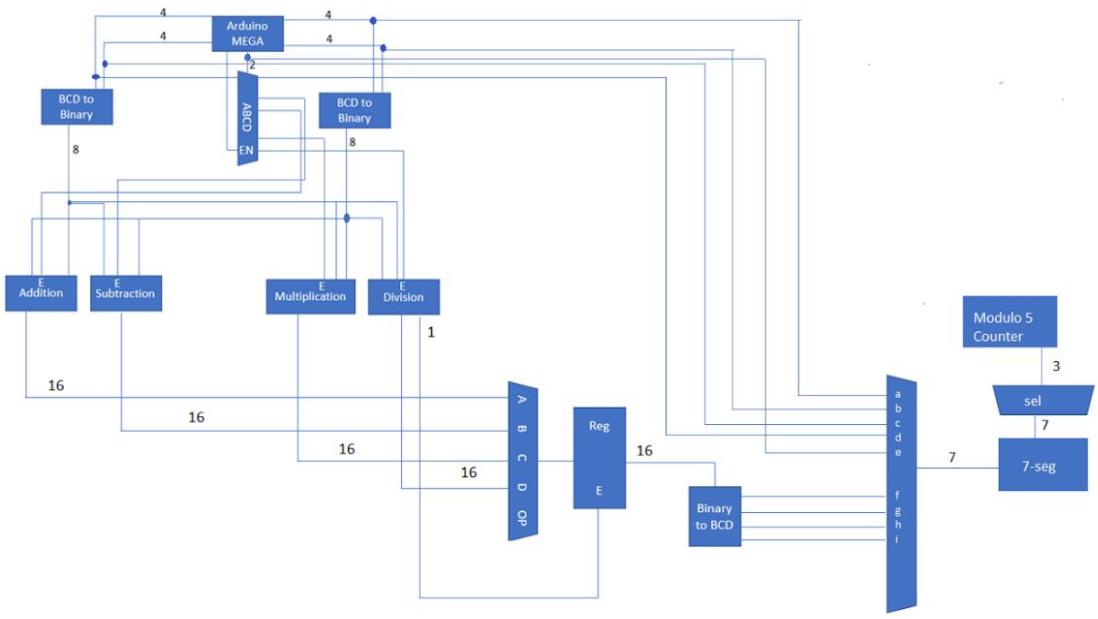
Fig 4: Real Hardware implementation



Conclusions

Simple arithmetic such as addition, subtraction, multiplication, or division are a given in modern day computers, and technology has evolved to be able to run these operations millions of times a second. Developing our own calculator has not only strengthened our understanding of VHDL and microprocessors, but has also given the group a broader appreciation for modern technology and its intricacies.

Fig 5: VHDL Logic Chart



References:

Binary to BCD circuit was taken from Dr. Llamocca's website:
http://www.secs.oakland.edu/~llamocca/VHDLforFPGAs.html