FPGA Microprocessor

ECE 2700 PRESENTATION

Matthew Irvine , David Navarre, George Trupiano,
Joshua Duncan

Design Objective

Our team decided to to program a 4-bit microprocessor for the Nexys A7 FPGA Development Board. The processor can perform math, logic, and bit-wise functions using simple 4-bit instructions.

Data Transfer Functionality Ex.

- Reading data
- Storing data

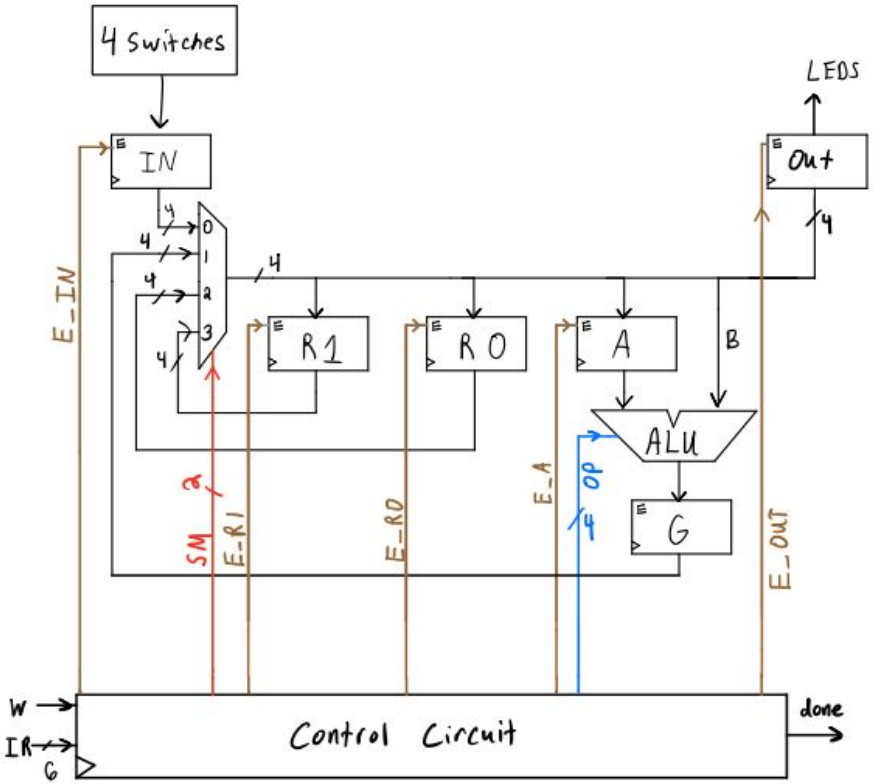
Arithmetic Functions Ex.

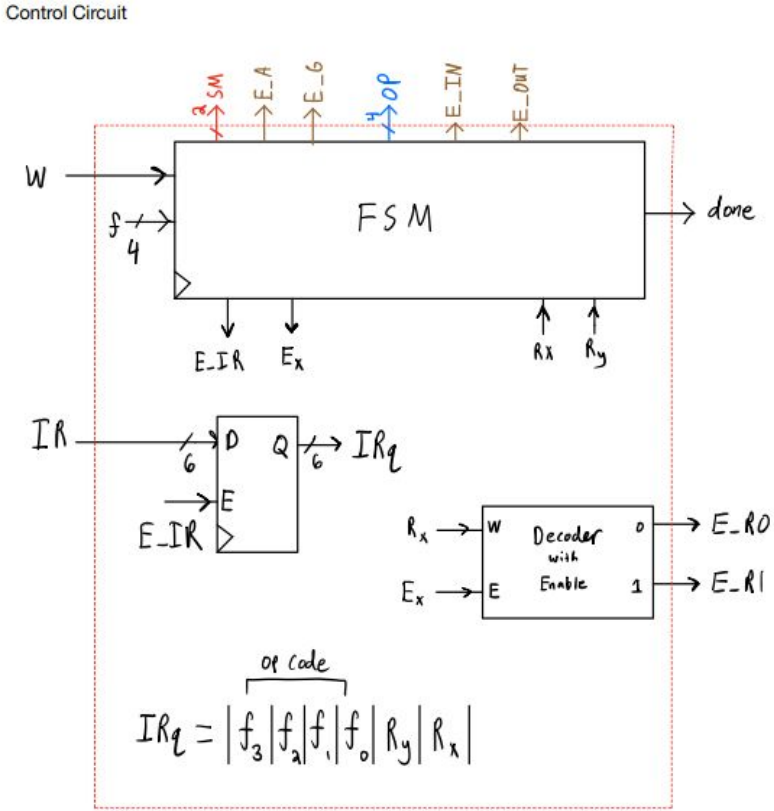
- Addition
- Subtraction
- Incrementation

Logic Functions Ex.

- AND
- XNOR
- NAND

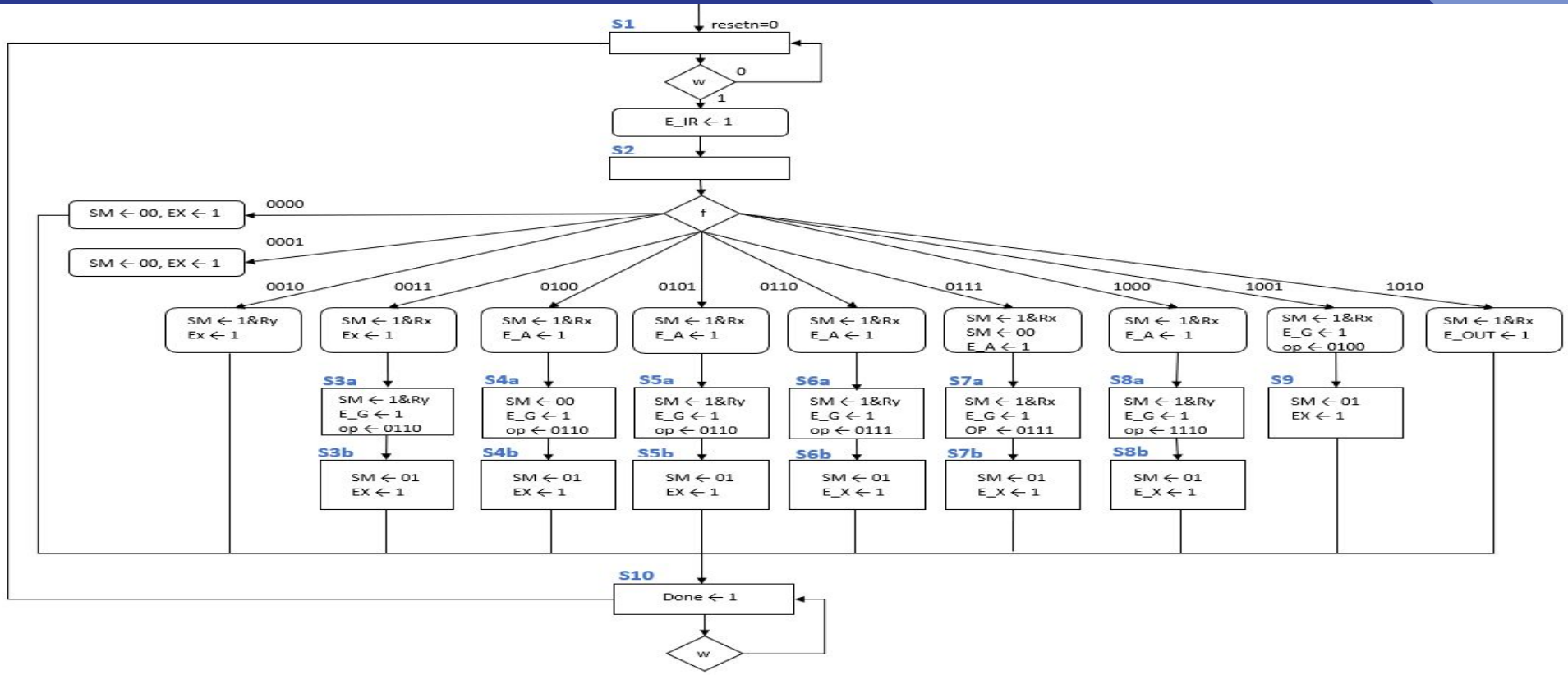
Block Diagram





Finite State Machine

There are thirteen different states for the FSM. When the code has run through the necessary states the result is output to the LED display and represented in binary.



Instruction Set

When “w” = 1, the instructions are taken from IR and executed. Instruction = |f3||f2||f1||f0||Ry||Rx|.

This is referred to as “machine language instruction” or assembly instruction.

- Opcode (operation code): IR(5…2). These bits specify the operation to be performed.
- Operands: IR(1..0). These bits specify the register indices to be used in the operation:
- Rx: Index of the register where the result of an operation is stored (and data can be read from Ry)
- Ry: index of the register where the result of an operation is stored (and data can be read from Ry)
- To simplify operation, values cannot be stored in both registers simultaneously.

f(IR[5…2])	Operation	Function
0000	Load IN	IN←Switches
0001	Load R0, IN	R0←IN
0010	Copy R0, R1	R0←R1
0011	Add R0, R1	R1←R0+R1
0100	Add R0, IN	R0←R0+IN
0101	Xnor R0, R1	R1←R0 XNOR R1
0110	Subtract R1, R0	R0←R0-R1
0111	Subtract IN, R0	R0←IN - R0
1000	Xor R0,R1	R0←R0 Xor R1
1001	Inc R0	R0←R0+1
1010	LOAD OUT, R0	OUT←R0
1011	.....any op	Any function
1100	.....any op	Any function
1101	.....any op	Any function
1110	.....any op	Any function
1111	.....any op	Any function

Arithmetic Logic Unit

The Arithmetic Logic Unit contains arithmetic and logic functions. In particular the FPGA microprocessor used addition, incrementation, XOR, and AND Logic to accomplish simple operations

OP	Operation	Function
0000	y <= A	Transfer 'A'
0001	y <= A + 1	Increment 'A'
0010	y <= A - 1	Decrement 'A'
0011	y <= B	Transfer 'B'
0100	y <= B + 1	Increment 'B'
0101	y <= B - 1	Decrement 'B'
0110	y <= A + B	Add 'A' and 'B'
0111	y <= A - B	Subtract 'B' from 'A'
1000	y <= not A	Complement 'A'
1001	y <= not B	Complement 'B'
1010	y <= A AND B	AND
1011	y <= A OR B	OR
1100	y <= A NAND B	NAND
1101	y <= A NOR B	NOR
1110	y <= A XOR B	XOR
1111	y <= A XNOR B	XNOR

References

https://www.fpga4student.com/

http://www.secs.oakland.edu/~llamocca/VHDLforFPGAS.html