DCU School of Electronic Engineering Assignment Submission

Student Name(s):	Michael Lenehan
Student Number(s):	15410402
Programme:	B.Eng in Electronic and Computer Engineering
Module Code:	EE496
Lecturer:	X. Wang
Project Due Date:	30/11/2018

Assignment 2

Michael Lenehan

Purpose

The purpose of this assignment is to complete the design of a 16-bit cryptographic coprocessor. The design process is divided amongst the two sections of this assignment. Firstly, the combinational components of the design must be completed. These components include an ALU, capable of performing addition and subtraction operations, along with a number of logical operations, a Shifter, which can shift or rotate the input bits, and a Lookup module, which substitutes the input bits for a value specified in the provided lookup table.

The second section of this assignment focuses on the synchronous design elements, i.e. the memory registers used to store the values output from the combinational logic components. Initially both input and output register components must be createed and tested. Once working, A register file is used to model the memory of the completed system. The completed register file component must then be combined with the combinational logic components, giving the completed crypto coprocessor

Procedure

2.1 Combinational Logic Design

The Combinational Logic of the Crypto Coprocessor allows for the non-memory associated functionality of the coprocessor. The components required give the cryptographic functionality, such as the arithmetic and logical operations of the ALU.

2.1.1 ALU Design

The ALU design is completed according to the provided diagram and operations table found below:

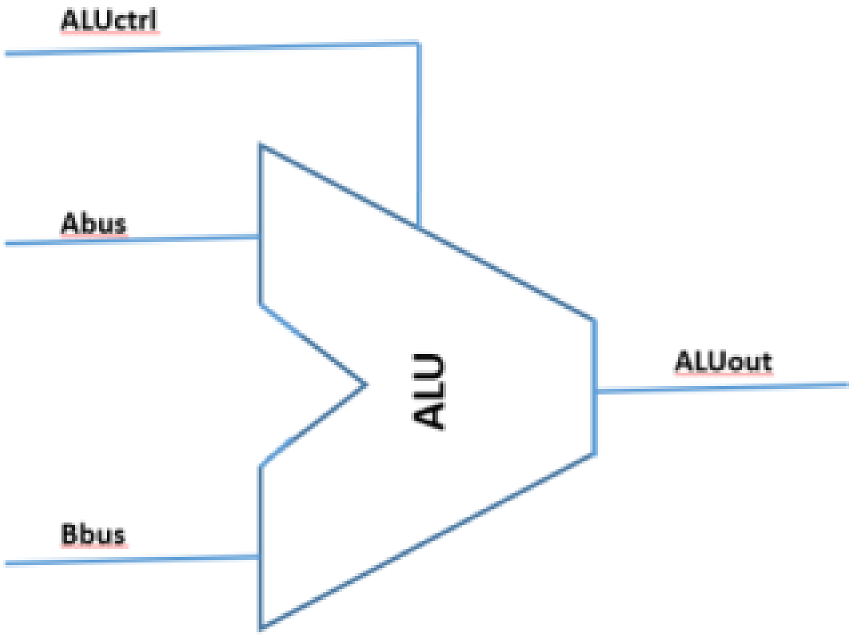

Figure 1: ALU Component

Table 1: ALU Operations

ALUctrl	Mnemonic	Function
0000	ADD	ALUout = Abus + Bbus
0001	SUB	ALUout = Abus - Bbus
0010	AND	ALUout = Abus & Bbus
0011	OR	ALUout = Abus | Bbus
0100	XOR	ALUout = Abus ^ Bbus
0101	NOT	ALUout = !Abus
0110	MOV	ALUout = Abus

The ALU takes three inputs; a control signal (ALUctrl), and two 16-bit input values (Abus, Bbus). An operation is performed on the 16-bit input values according to the operation specified by the control signal, with the output of the operation being output on the ALUout signal.

2.1.2 Shifter Design

The Shifter component is completed according to the provided diagram and operations table, found below:

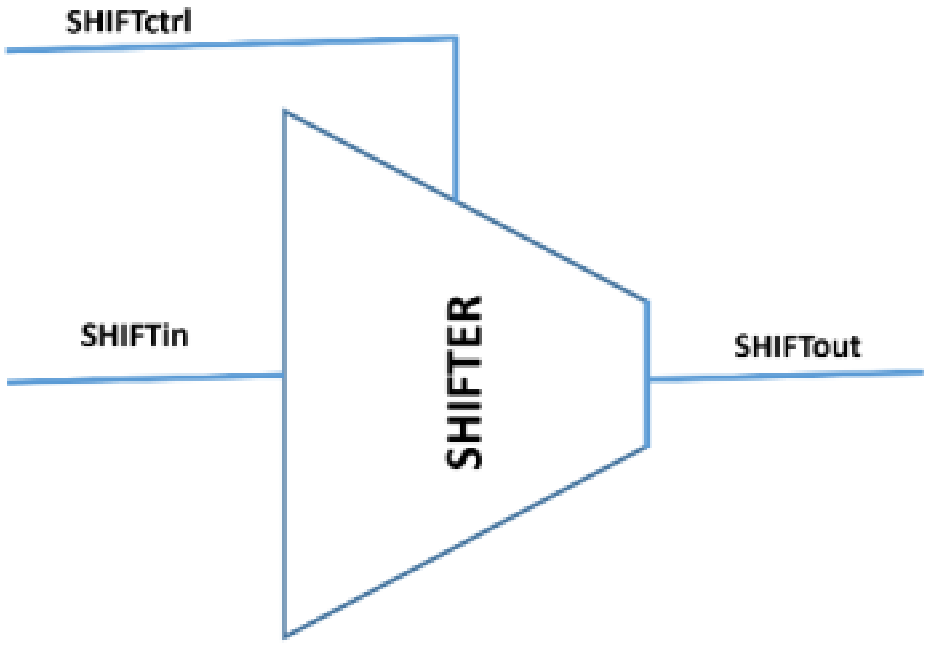

Figure 2: Shifter Component

Table 2: Shifter Operations

SHIFTctrl	Mnemonic	Function
1000	ROR4	Rotate Right 4 bits
1001	ROL4	Rotate Left 4 bits
1010	SLL4	Shift Left Logic 4 bits
1011	SRL4	Shift Right Logic 4 bits

The Shifter takes two inputs; a control signal (SHIFTctrl), and a 16-bit input value (SHIFTin). The shifter then performs an operation, as specified by the SHIFTctrl input, passing the output to the SHIFTout signal.

2.1.3 Lookup Component Design

The Lookup Table component is completed according to the diagram, and operation tables, found below:

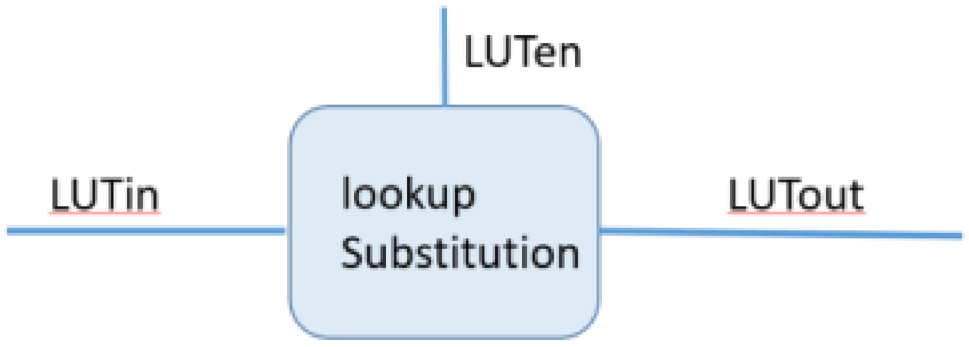

Figure 3: Lookup Table Component

The expanded diagram below displays the the index of the bits which may be passed through the component unchanged. Bits 7-4 are substituted as specified in S-Box 1, with bits 3-0 substituted as specified in S-Box 2.

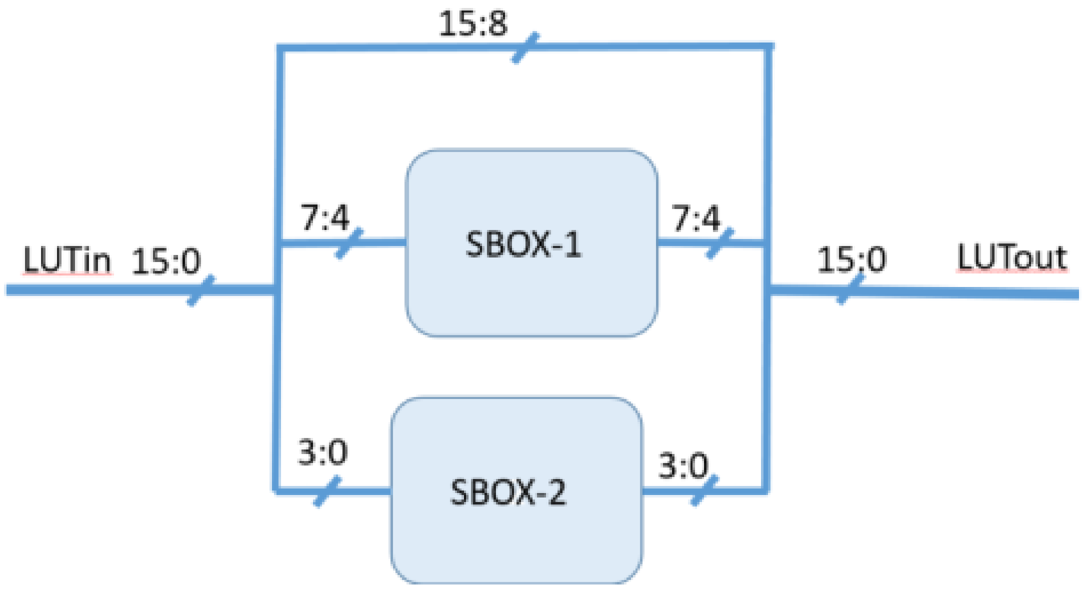

Figure 4: Lookup Table Component Expanded

Table 3: S-Box 1 Substitutions

Input	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Output	1	11	9	12	13	6	15	3	14	8	7	4	10	2	5

Table 4: S-Box 2 Substitutions

Input	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Output	15	0	13	7	11	14	5	10	9	2	12	1	3	4	8

The Lookup Table component takes two inputs; a control signal (LUTen), and a 16-bit input signal (LUTin). The substitution of the bits of the least significant byte of the input signal is performed according to the above tables. The output signal (LUTout) is then assigned the value of the unchanged most significant byte of the input, along with the substituted least significant bits.

2.1.4 Structural Model

The Structural model of the Combinational Logic is completed by combining the ALU, Shifter, and LUT components. The input control signal must be decoded to send input ABus and BBus signals to the appropriate component, and to output the appropriate result from the multiplexer.

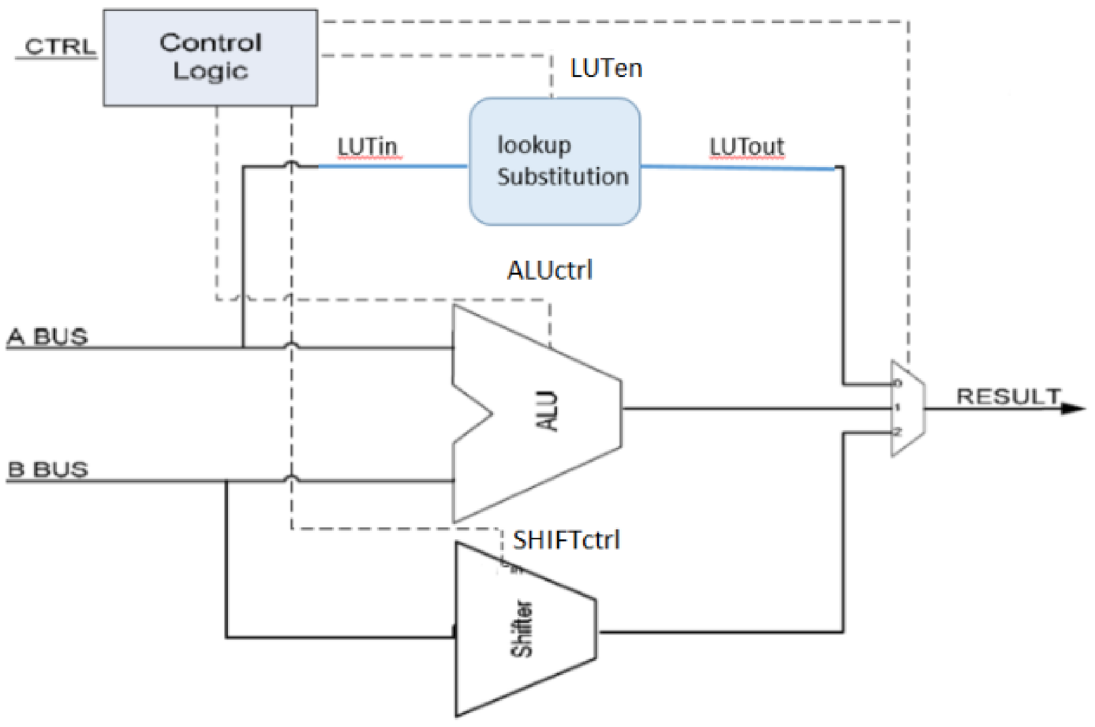

Figure 5: Structural VHDL Model

2.1.5 Testing

Testing of the Combinational Logic must be completed. Using a testbench, a series of input control signals and A and B Bus are passed into the system. The output of these operations must be compared with the expected results to ensure that the system is operating correctly.

2.2 Synchronous Logic Design

The Synchronous Logic Design of the Crypto Coprocessor implements the memory elements of the design. Initially using an input and output register, this is expanded to using a register file, which provides synchronous writes with either synchronous or asynchronous reads.

2.2.1 Registers

The Input and Output Register components are implemented as specified in the provided diagram, found below:

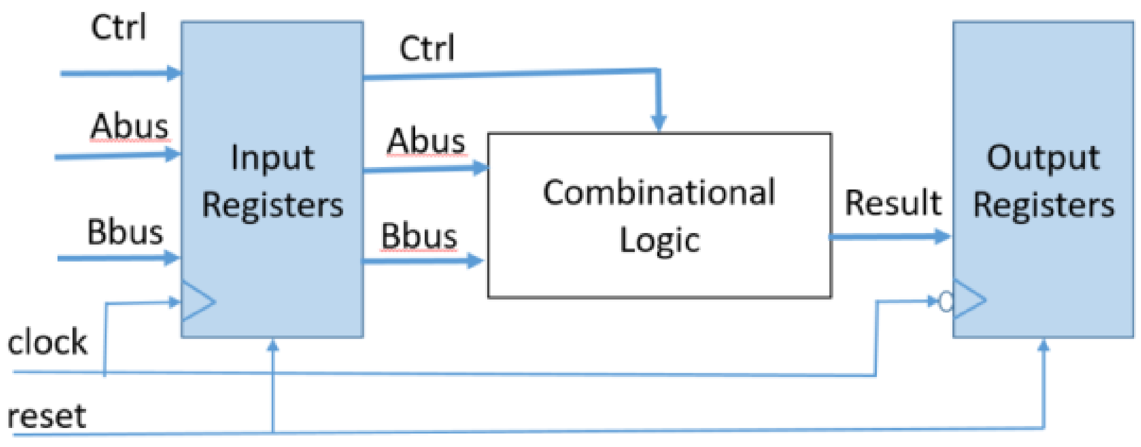

Figure 6: Register Components

The Input Register component has five inputs; a control signal (Ctrl), two 16-bit input signals (Abus, Bbus), a rising edge triggered clock input, and a reset input. The register outputs the control signal, and both the A and B Bus to the Combinational Logic.

The Output Register has three inputs; A 16-bit input (Result) which takes in the output of the Combinational Logic, a falling edge triggered clock input, and a reset input.

2.2.2 Memory

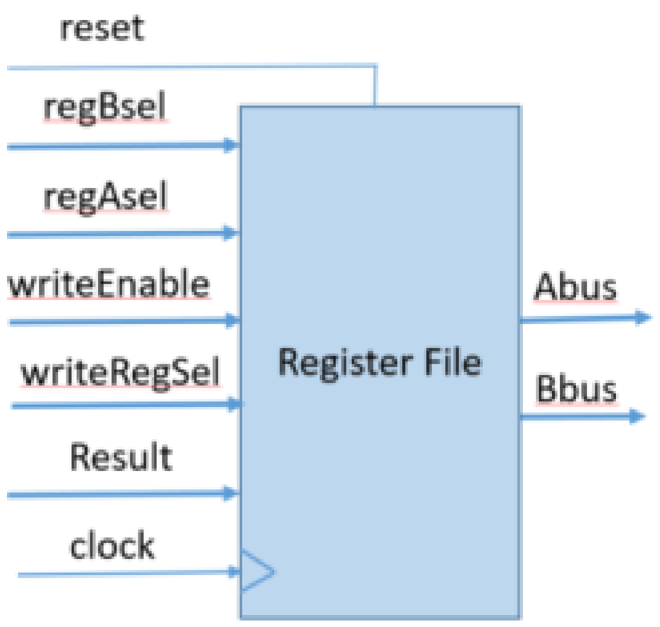

Figure 7: Register File Component

2.2.3 Complete Crypto Coprocessor

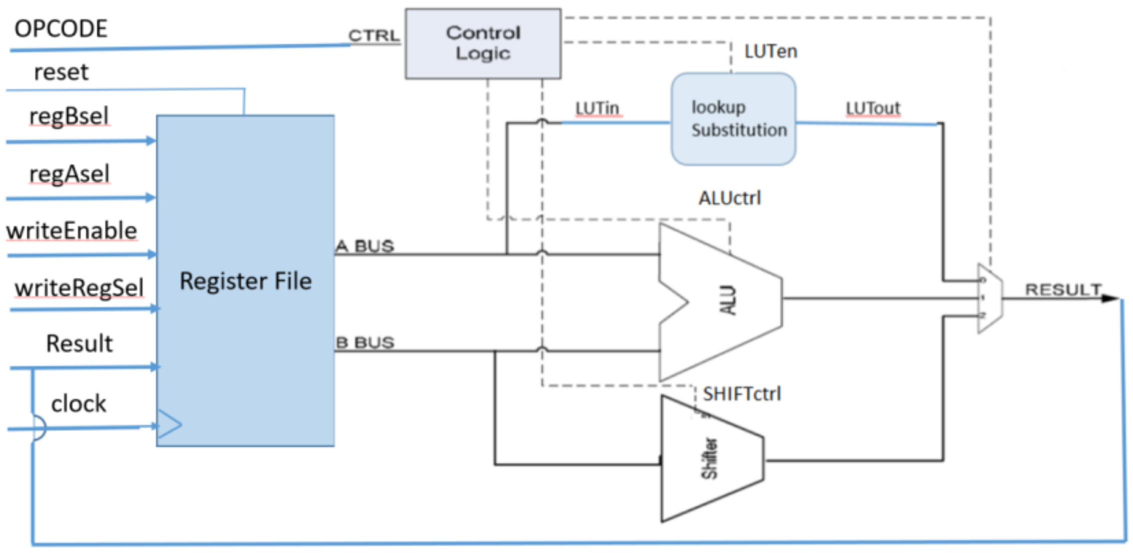

Figure 8: Complete Crypto Coprocessor VHDL Model

2.2.4 Testing

Results

The following results were obtained from the completion of the the combinational and synchronous sections of the assignment.

3.1 Combinational Logic

The combinational logic consists of the Arithmetic Logic Unit, the Shifter component, the Lookup Substitution component, and the Control Signal Decoder.

3.1.1 ALU

The ALU must perform a number of arithmetic or logical operations on the input 16-bit values. These operations include arithmetic addition and subtraction, and logical AND, OR, XOR, NOT, and Move operations.

Using a case statement, the control signal may be used to determine the operation to be performed. The code, found in the Appendix "ALU Code" section, generates the following schematic:

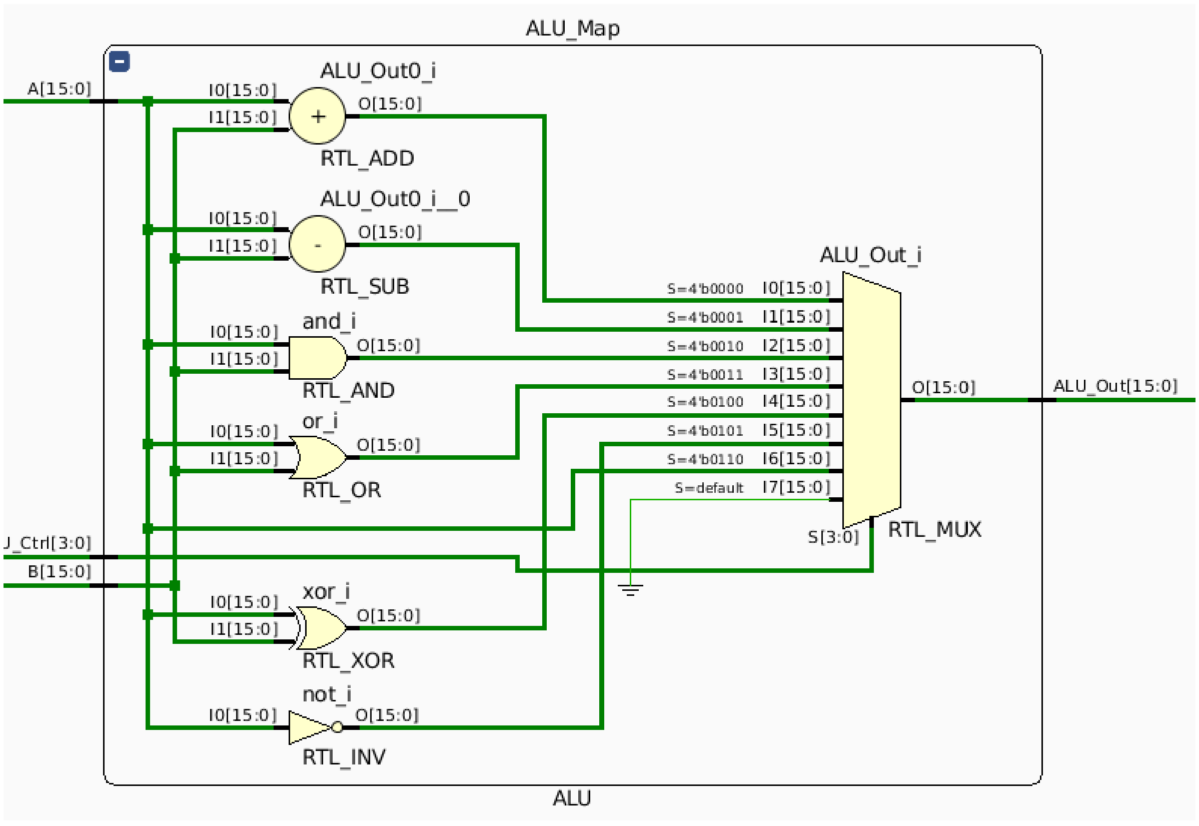

Figure 9: Generated ALU Compnent Schematic

3.1.2 Shifter

The Shifter component must perform bitwise rotation or shifting operations. As the ports used in this component are of type "unsigned" the ROR, ROL, SLL, and SRL functions may not be used, as they may have unexpected behaviour. As such the "rotate_right()", "rotate_left()", "shift_right()", and "shift_left()" functions must be used.

A case statement chooses the operation to perform on the 16-bit input based on the input control signal. The code, found in the Appendix "Shifter Code" section, generates the following schematic:

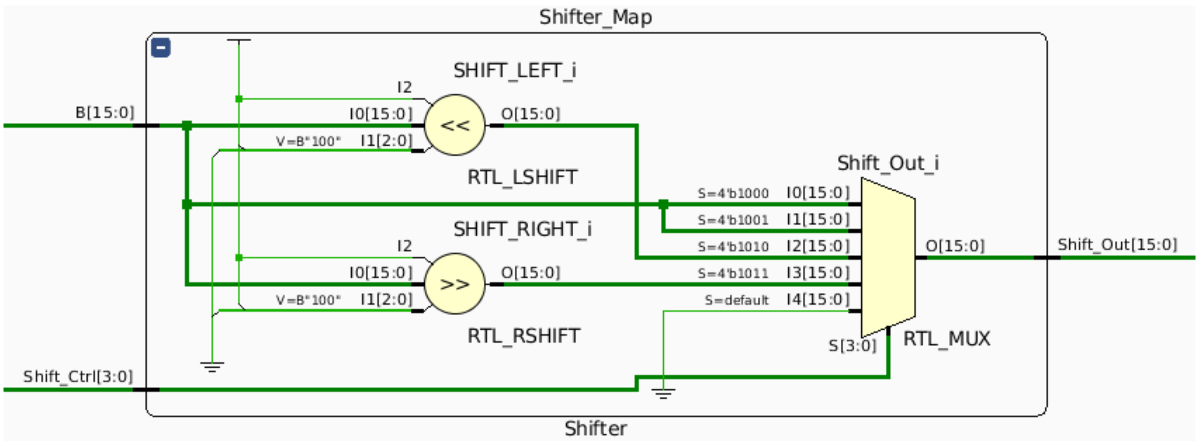

Figure 10: Generated Shifter Component Schematic

3.1.3 LUT

The Lookup Substitution component performs substitutions based in the input 16-bit values. The upper byte of the input value must pass through unnafected, with each of the lower four bits having their values substituted. Using two case statements, the upper 4-bits of the lower byte, and the lower 4-bits of the lower byte have their values substituted. These new values are then recombined with the unchanged upper byte, and output.

The code, found in the Appendix "LUT Code" section, generates the following schematic:

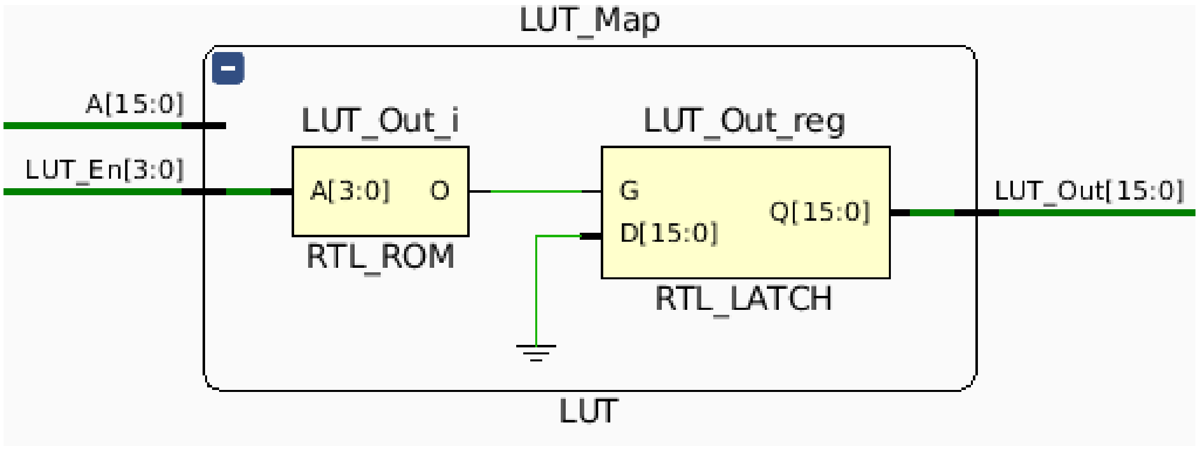

Figure 11: Generated Lookup Table Component Schematic

3.1.4 Control Decoder

The Control signal decoder component takes the input control signal, and outputs both the original control signal for each of the other components, and a value from 0 to 2, which allows an output multiplexer to output the value of the desired operation.

A case statement is used to choose the value from 0 to 3 to assign to each of the input control signal values. The value of 0 indicates a Lookup substitution operation, the value 1 indicates an ALU operation, and the value 2 indicates a Shifter operation. The code, found in the Appendix "Control Decoder Code" section, generates the following schematic:

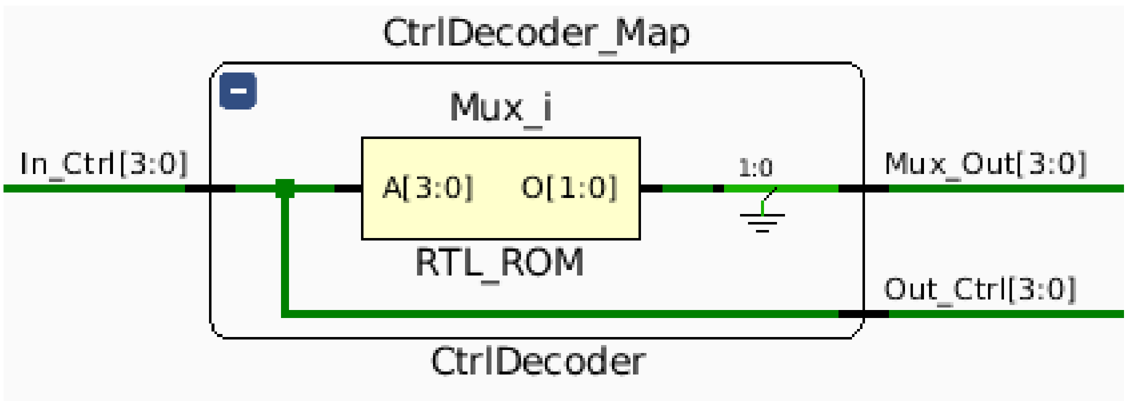

Figure 12: Generated Control Signal Decoder Schematic

3.1.5 Combinational Logic

The combinational logic combines the aforementioned components, passing in two 16-bit values (the A and B Bus), and a 4-bit control value. The appropriate operation is performed on the input values, with the appropriate result being output.

The code, found in the Appendix "Combinational Logic Code" section, generates the following schematic:

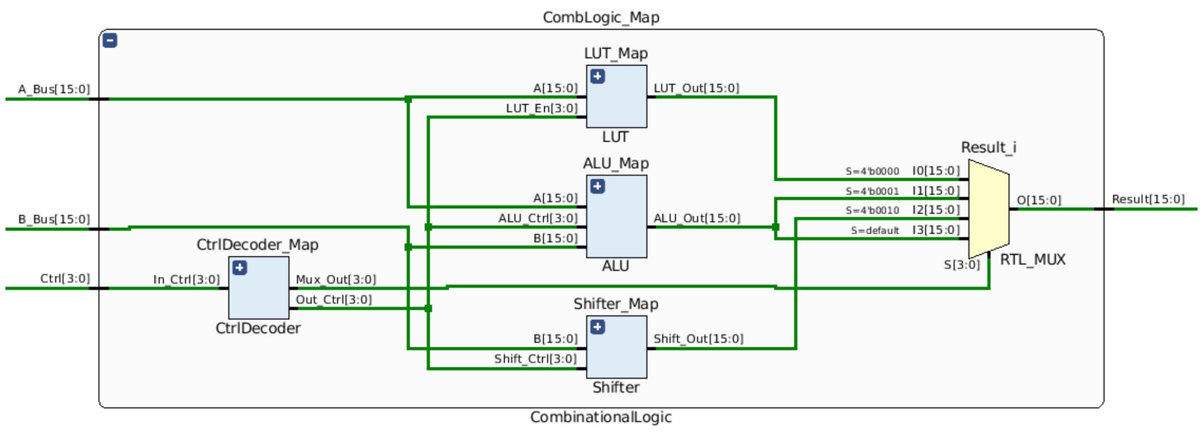

Figure 13: Generated Combinational Logic Component Schematic

3.2 Synchronous Logic

The synchronous logic consists of either a number of registers (Input and Output), or a register file, which can store a number of results, and perform operations on the specified results.

3.2.1 Registers

3.2.2 Register File

3.2.3 Complete Crypto Coprocessor

3.3 Testing

Testing was completed using a combination of a testbench and stimulus generator. A separate testbench and stimulus generator combination was used for the combinational logic, synchronous logic, and complete crypto coprocessor.

3.3.1 Combinational Logic

In order to test that the combinational logic was operating as required, a testbench, which, using a stimulus generator, passed specified inputs to the A and B buses. The

Result output was then compared with an expected output, logging a “passed” or “failed” message to the console.

The A bus, B bus, and expected results used in the test are contained in the following table (in hexidecimal representation):

Table 5: Inputs and Expected Outputs of Combinational Logic

Index	OpCode	RA	RB	Expected Result
1	0	F0F1	F0F1	E1E2
2	1	F0F2	F0F1	0001
3	2	B2A1	C3D1	8281
4	3	234A	A21D	A35F
5	4	234A	A21D	8157
6	5	F21D	–	0DE2
7	6	ABCD	–	ABCD
8	8	–	12BC	C12B
9	9	12BC	2BC1	
10	A	–	BD14	D140
11	B	–	BD14	0BD1
12	C	B2A1	–	B270

The code for the testbench and stimulus generator can be found in the Appendix

3.3.2 Synchronous Logic

The synchronous logic was tested in the same way as the combinational logic. A testbench and stimulus generator compare expected results with the output of the second register. The stimulus generator was modified to initiate the “clktmp” variable with a value of ‘0’. The table used in the combinational logic section was again used for testing purposes, giving the same outputs as above.

3.3.3 Complete Crypto Coprocessor Test Program

The completed Crypto Coprocessor was tested by passing in a number of 16-bit instruction words. These words refer to the opcode, register to assign to the A_Bus, register to assign to the B_Bus, and register to assign the result of the operation to.

A testbench and stimulus generator were again used to pass in the instruction word, and to compare the outputs to the expected output.

The following table gives the operations performed in the testing process:

Table 6: Inputs and Expected Outputs of Completed Crypto Coprocessor

Operation	A Register	B Register	Result Register
ADD	R5	R4	R12
XOR	R1	R8	R7
ROR4	–	R12	R0
SLL4	–	R9	R3
ADD	R0	R7	R10
SUB	R7	R3	R12
NOP	–	–	–
AND	R12	R10	R9
NOP	–	–	–
LUT	R9	–	R2

Conclusion

Appendix