ECE 474a/574a Assignment 2

Assignment Points: 100

DUE DATE: Sunday, November 8, 11:59PM

Overview

ECE474a/574a: Create a program named ***dpgen*** (short for datapath generator) that will convert a behavioral netlist specification into a synthesizable Verilog datapath implementation.

ECE 574a: After creating the Verilog implementation, the ***dpgen*** program should report the critical path of the datapath, reporting either the longest register to register delay or the longest input to register delay.

Command-line Arguments

Your program must be capable of utilizing a command-line argument to specify the input and output files.

```
dpgen netlistFile verilogFile
```

Your program must ensure the user has correctly provided the required command-line arguments and display a usage statement if the provided arguments are incorrect.

Verilog implementation of Datapath Component Library

The Verilog implementation should utilize the parameterized datapath component library created in the previous assignment.

Modification to the library may be needed to support signed and unsigned arithmetic. The following summarizes the datapath components required. For signed versions of datapath components, precede the component name listed below with S. Ex: A signed version of the COMP component should be named SCOMP.

Name	Data Inputs	Control Inputs	Data Outputs	Control Outputs	Description
REG	d	Clk, Rst	q		Register
ADD	a, b		sum		Adder
SUB	a, b		diff		Subtractor
MUL	a, b		prod		Multiplier
COMP	a, b			gt, lt, eq	Determines if a > b, a < b, and a == b
MUX2x1	a, b	sel	d		Multiplexor
SHR	a	sh_amt	d		Arithmetically shifts input sh_amt positions to the right

SHL	a	sh_amt	d		Arithmetically shifts input sh_amt positions to the left

- ***ECE 574a:***

Name	Data Inputs	Control Inputs	Data Outputs	Control Outputs	Description
DIV	a, b		quot		Returns quotient of a / d.
MOD	a, b		rem		Returns remainder of a / d.
INC	a		d		Returns a + 1
DEC	a		d		Returns a - 1

Behavioral Netlist Specification

A behavioral netlist file provides an acyclic connection of components, where:

- All empty lines should be ignored
- All lines beginning with "//" are considered comments and should be ignored
- The netlist file can be assumed to be fully space/tab delimited, i.e. one or more space or tab characters should appear between each token that needs to be parsed, including colons.
- Circuit inputs and outputs can be declared on a line using the formats:

```
input dataType inputName1, inputName2
output dataType outputName1, outputName2
```

- Valid data types include:
  - Signed Integers Types: `Int1`, `Int2`, `Int8`, `Int16`, `Int32`, and `Int64`
  - Unsigned Integer Types: `UInt1`, `UInt2`, `UInt8`, `UInt16`, `UInt32`, and `UInt64`
  - The number after `Int` and `UInt` specifies the width of the data input, output, register

- All outputs are implicitly associated with a register (`REG`) component
- Internal registers (which should be implemented as a REG component) must be explicitly declared using the format:

```
register dataType regName1, regName2
```

- Wires (internal connections between components) must be explicitly declared using the format:

```
wire dataType wireName1, wireName2
```

- Component instantiations are declared on a single line using the following formats for specific components. (Note: comments are provided to indicate the format for specific components defined within the technology library and will not appear within the sample behavioral netlist, nor will you see comments at the end of a line.)

```
o = b                // REG
```

```
o = a + b           // ADD
o = a - b           // SUB
o = a * b           // MUL
o = a > b           // COMP (gt output)
o = a < b           // COMP (lt output)
o = a == b          // COMP (eq output)
o = sel ? i1 : i0   // MUX2x1
o = a >> sh         // SHR
o = a << sh         // SHL
o = a / b           // DIV (ECE 574a only)
o = a % b           // MOD (ECE 574a only)
o = a + 1           // INC (ECE 574a only)
o = a - 1           // DEC (ECE 574a only)
```

- The width of a datapath component (except comparators) should be determined by the size of the output, register, or wire to which the output of the component connects. Signed integer inputs with fewer bits should be sign extended, and unsigned integer inputs should be padded with 0's. Inputs with greater bits should connect the least significant bits to the input. The size of comparators should be determined by the size of the largest input.
- The width of comparators should be determined by the size of the largest input.
- The result of an operation may be assigned to a register declared within the behavioral netlist. For example, assume the netlist includes a register named A. The statement A = B + C would be a valid operation. The declaration of A as a register would specify that the result of the operation should be stored in a register (REG component). If A is used in another operation (e.g., F = A + T), then the output of the register should be used as the input to that operation.
- All names for components, inputs, outputs, wires, registers, and instances should be unique.
- All names for inputs, outputs, wires, registers, and instances are case sensitive and can consist of any number of letters or digits
- Input, output, wires, and reg declarations should come before component instantiations.

ECE574a: Critical Path Calculation

The ***dpgen*** program should report the critical path of the datapath, reporting either the longest register to register delay or the longest input to register delay. The critical path calculation should utilize the following latency estimation for the components within the datapath component library:

Name	1-bit	2-bit	8-bit	16-bit	32-bit	64-bit
REG	2.616	2.644	2.879	3.061	3.602	3.966
ADD	2.704	3.713	4.924	5.638	7.270	9.566
SUB	3.024	3.412	4.890	5.569	7.253	9.566
MUL	2.438	3.651	7.453	7.811	12.395	15.354
COMP	3.031	3.934	5.949	6.256	7.264	8.416
MUX2x1	4.083	4.115	4.815	5.623	8.079	8.766
SHR	3.644	4.007	5.178	6.460	8.819	11.095

SHL	3.614	3.980	5.152	6.549	8.565	11.220
DIV	0.619	2.144	15.439	33.093	86.312	243.233
MOD	0.758	2.149	16.078	35.563	88.142	250.583
INC	1.792	2.218	3.111	3.471	4.347	6.200
DEC	1.792	2.218	3.108	3.701	4.685	6.503

Assuming the input files do not contain any errors, the critical path should be output as:

```
Critical Path : 5.124 ns
```

Submission Requirements

- All programming assignments must be implemented with C or C++ using the CMake cross platform make tools. CMake is a set of tools that provides an easy way to utilize different platforms and development environments (e.g. Mac OSX with XCode, Linux using Makefiles, Windows using Visual Studio, etc.). Using CMake, you are encouraged to utilize whatever development tools and platform you are most comfortable with.
- All assignments will be compiled and tested using the ECE department server *ece3 (ece3.ece.arizona.edu)*, on which the CMake tools are already available. You can access the ece3 server using any secure shell (SSH) client (e.g., PuTTY). You are strongly encouraged to test your programs using the ece3 server before submission.

Note: If you use your own computer for development, you will have to download and install the CMake tools.

- All assignments must be submitted as a single TAR/GZIP (.tgz) or ZIP (.zip) using the naming convention `NetID1_NetID2.tgz` or `NetID1_NetID2.zip`, where *`NetID1`* and *`NetID1`* are the UA NetIDs for each group member. The submitted archive should use the following directory and file structure:

```
NetID1_NetID2_dpgen
   |
   ---> CMakeLists.txt
   |
   ---> README.txt
   |
   ---> src
         |
         ---> CMakeLists.txt
         |
         ---> .c/.h/.cpp files
```

- All submitted assignments will be compiled using the following commands:

```
cd NetID1_NetID2_dpgen
mkdir build
cd build
cmake ..
make
```

```
./src/dpgen
```

- Do NOT include build files, executables, temporary files, etc. with your assignment submission.
- Include a README.txt in your submission that specifies: 1) name and NETID of group members, 2) course you are enrolled in, 3) brief description of your program, and 4) contribution of each group member to the assignment.