Productivity Tools for IC Design (EE2510). Oct-Nov 2025. Exercise Set 6.

Section 1. TCL

1. Write a TCL program to read a file like this:

```
clock: clk1
period: 10
inputs: {a b c}
outputs: {y z}
```

And produce an output like:

```
create_clock -name clk1 -period 10 [get_ports clk1]
set_input_delay 2 -clock clk1 [get_ports {a b c}]
set_output_delay 2 -clock clk1 [get_ports {y z}]
```

2. Make a program to extract netlist hierarchy from verilog:

```
module submodule1(...);
        leaf1 a(...);
        leaf2 b(...);
endmodule

module top(...);
        submodule1 u1( ... );
        submodule2 u2( ... );
endmodule

module submodule2(...);
        leaf2 c(...);
endmodule
```

Then the output should be:

```
Hierarchy:
top
  |---- submodule1
  |        |---- leaf1
  |        |---- leaf2
  |---- submodule2
           |---- leaf2
```

Section 2. Jenkins

As mentioned in previous ES, jenkins can be run locally by

```
$ java -jar jenkins.war --httpPort=8080
```

Which will launch a webserver which can be accessed from your browser at http://localhost:8080. Go there and create a new administrator by providing username and password. Once you log in, you will get a screen which has menu on the left hand side. These menu items are referred to below.

1. A basic jenkins pipeline.

Use this zipped folder as a base for this exercise. The folder contains a Makefile, a c-source (main.c) and a script (test.sh). The recipe “all” compiles and places the output file in “build” folder. Then the recipe “test” can see if that worked fine. In my case I unzipped this into ”/home/sid/workspace/ee2510/tmp/” folder and that is used below in steps.

We are building a jenkins recipe which runs both the recipes and saves the generated output for that particular run.

We will automate this in two ways.
A. Manually: Go to Jenkins -> New Item. Provide a name for this project (e.g “local-test”) and then select “Freestyle project” from options below. Say OK.
In the next screen select “Source code management” as none (since we are not involving git in this recipe) and then add a build step with following code after selecting “Execute Shell” option.

```
cp -r /home/sid/workspace/ee2510/tmp/jenkins-example1/* .
make all
make test
```

Here we are telling jenkins that for the “build” stage, it needs to copy the whole example to its own workspace (the “.” at the end of copy command). This folder is ~/.jenkins/workspace in my installation. Then we are telling it to run the two make recipes.

After this, in “Post-build actions” select “Archive the artefacts” and enter in the space provided:

```
build/*
```

Which will save the files in ./build/ folder of the jenkins workspace.

Now you can save the recipe, click it, and use “build now” to execute it. It should pass with a green tick. Clicking on the link with green tick will take you to the information about the build (e.g. console logs, build artefacts).

B. We can also use the recipe as a pipeline: again make a new item, give project name and choose the pipeline. In script box on the upcoming page fill:

```
pipeline {
        agent any
        stages {
        stage('Build') {
        steps {
                sh '''
                cp -r /home/sid/workspace/ee2510/tmp/jenkins-example1/* .
                make all
                '''
        }
        }
        stage('Test') {
        steps {
                sh '''
                make test
                '''
        }
        }
        }
        post {
        always {
        archiveArtifacts artifacts: 'build/*'
        }
        }
}
```

Save and then execute it with “build now” as you have done earlier.

2. How to add github triggers.

```
pipeline {
        agent { github.com/….}
        triggers {
        githubPush()            // Trigger when GitHub sends a webhook
        }

        stages {
        stage('Checkout') {
        steps {
                checkout scm
        }
        }

        stage('Build') {
        steps {
                sh '''
                make all
                '''
        }
        }

        stage('Test') {
        steps {
                sh '''
```

```
                    cp -r /home/sid/workspace/ee2510/tmp/jenkins-example1/* .
                    make test
                    '''
          }
          }

          stage('Archive') {
          steps {
                    archiveArtifacts artifacts: 'build/**'
          }
          }
          }
}
```

Then in github repository, Settings -> Webhooks -> Add webhook, with content type as application/json and “Just the push event”

```
http://<your-jenkins-server>:8080/github-webhook/
```

With this setup every push event triggers the jenkins recipe.