ENEL594-FinalProject

This GitHub repository contains all files relevant for the ENEL 594 Final Project. The files for Chapter 4 of the thesis are in verilator_codeql. The files for Chapter 5 of the thesis are in cfg. All project files were developed and used on a machine with the Ubuntu 20.04.5 LTS OS; all installation instructions assume a Ubuntu system.

Directory Structure

```
.
├── cfg             # cfg project files
│   ├── bin         # location of executable
│   ├── build       # build temp files
│   ├── include     # include files
│   ├── Makefile
│   ├── src         # source files
│   └── tests       # test files, including files used for experiments
├── README.md
└── verilator_codeql            # verilator + codeQL project files
    ├── conditionals            # experiments for conditional stmt
variations
    ├── FSM                     # experiments for different FSMs
    ├── hierarchy               # experiments for single file vs multiple
files
    ├── RTD012                  # RTD012 experiments
    ├── RTG011                  # RTG011 experiments
    ├── secure_FSM              # secure FSM experiments
    ├── secure_reg              # Register w/ acct ctrl experiments
    └── vscode-codeql-starter   # VS Code CodeQL utility, contains queries
used for experiments
```

Dependencies

```
sudo apt update
# Verilator
sudo apt install git perl python3 make autoconf g++ flex bison ccache
sudo apt install libgoogle-perftools-dev numactl perl-doc
sudo apt install libfl2  # ignore if gives error
sudo apt install libfl-dev  # ignore if gives error
sudo apt install zlibc zlib1g zlib1g-dev  # ignore if gives error
# Slang
sudo apt install python3 cmake build-essential
```

Slang

```
git clone -b v2.0 https://github.com/MikePopoloski/slang.git
cmake -B build
cmake --build build
sudo cmake --install build --strip
```

CodeQL

Download latest release from GitHub

```
unzip <download_path>/codeql-linux64.zip
export PATH=$PATH:<download_path>/codeql    # run every time a new terminal
is used or add to bashrc
```

Verilator

Verilator can be installed multiple ways, but we reccommend installing from source:

```
git clone -b v4.210 https://github.com/verilator/verilator  # clone v4.210
(version used)
cd verilator
autoconf         # Create ./configure script
./configure      # Configure and create Makefile
make -j `nproc`  # Build Verilator itself (if error, try just 'make')
sudo make install
```

Usage Instructions

cfg

To build cfg for a SystemVerilog file:

```
cd cfg
make
./bin/cdfg <path_to_sv_file>
```

This will:

- output the AST and CFG stats to stdout
- output the assignment clusters pre/post cfg analysis
- output a visualization of the CFG in ./cfg.svg

Verilator + CodeQL

These instructions are used to run the experimental work shown in the thesis

1. "verilate" design

```
verilator --cc --Mdir <path_of_dir_to_create> <path_to_sv_file>
```

2. build codeql database:

```
codeql database create <database_name_to_create> --language=cpp --
command='make -C <path_of_created_dir> -f V<mod_name>.mk'
```

3. Use CodeQL VS Code extension: follow guide

4. OR Use CodeQL CLI:

```
codeql database analyze <path_of_database_created> --format=csv --
output=codeql-res.csv <path_to_codeql_query>
```