

Using COTSon to analyze the performance of CJPEG on different architectures (CJPEG: compress an image from a bitmap format into a JPEG format)

Configuration

1 - download and install COTSon (only if it is not already installed): in the following the base directory of COTSon (where the 'configure' file is) will be indicated with <cotson_dir>

2 - download benchmark_cjpeg-v2.tgz

```
$ wget http://www.dii.unisi.it/~giorgi/teaching/tools2/cjpeg_benchmark-v2.tgz --user=cael  
--password=XXXX
```

3 - change to the cotson examples directory

```
$ cd <cotson_dir>/src/examples
```

4 - unpack benchmark_cjpeg-v2.tgz

```
$ tar xzf benchmark_cjpeg-v2.tgz
```

Simulation

	Cache Configuration A	Cache Configuration B
L1cache size	1 kB	32 kB
Main Memory latency	24	100

5 - Run the four test cases at once:

```
$ make
```

Verification

6 - You MUST check that the simulation went OK by comparing the generated JPEG files against the expected output:

```
$ diff output-large-memoryA.jpeg expected_output/output-large-memoryA.jpeg
```

```
$ diff output-large-memoryB.jpeg expected_output/output-large-memoryB.jpeg
```

```
$ diff output-small-memoryA.jpeg expected_output/output-small-memoryA.jpeg
```

```
$ diff output-small-memoryB.jpeg expected_output/output-small-memoryB.jpeg
```

Performance Analysis

7 - For each simulation, you can analyze statistics file:

- node.1.heartbeat_small-memoryA.log : cache configuration A and input small

- node.1.heartbeat_small-memoryB.log : cache configuration B and input small

- node.1.heartbeat_large-memoryA.log : cache configuration A and input large

- node.1.heartbeat_large-memoryB.log : cache configuration B and input large

For example to read the cycles spent in the Region of Interest (ROI):

```
$ grep timer.cycles *
```