Benchmarking

Bitcoin Core has an internal benchmarking framework, with benchmarks for cryptographic algorithms (e.g. SHA1, SHA256, SHA512, RIPEMD160, Poly1305, ChaCha20), rolling bloom filter, coins selection, thread queue, wallet balance.

Running

For benchmarking, you only need to compile `bitcoin_bench`. The bench runner warns if you configure with `--enable-debug`, but consider if building without it will impact the benchmark(s) you are interested in by unlatching log printers and lock analysis.

```
make -C src bitcoin_bench
```

After compiling bitcoin-core, the benchmarks can be run with:

```
src/bench/bench_bitcoin
```

The output will look similar to:

```
|               ns/op |                op/s |    err% |     total | benchmark
|--------------------:|--------------------:|--------:|----------:|:----------
|       57,927,463.00 |               17.26 |    3.6% |      0.66 | `AddrManAdd`
|          677,816.00 |            1,475.33 |    4.9% |      0.01 | `AddrManGetAddr`

...

|              ns/byte |              byte/s |    err% |     total | benchmark
|--------------------:|--------------------:|--------:|----------:|:----------
|               127.32 |        7,854,302.69 |    0.3% |      0.00 | `Base58CheckEncode`
|                31.95 |       31,303,226.99 |    0.2% |      0.00 | `Base58Decode`

...
```

Help

```
src/bench/bench_bitcoin -?
```

To print the various options, like listing the benchmarks without running them or using a regex filter to only run certain benchmarks.

Notes

More benchmarks are needed for, in no particular order: - Script Validation - Coins database - Memory pool - Cuckoo Cache - P2P throughput

Going Further

To monitor Bitcoin Core performance more in depth (like reindex or IBD): https://github.com/chaincodelabs/bitcoinperf

To generate Flame Graphs for Bitcoin Core: https://github.com/eklitzke/bitcoin/blob/flamegraphs/doc/flamegra