Final Project

Multicore Architectures

Antonios Providakis

Agenda

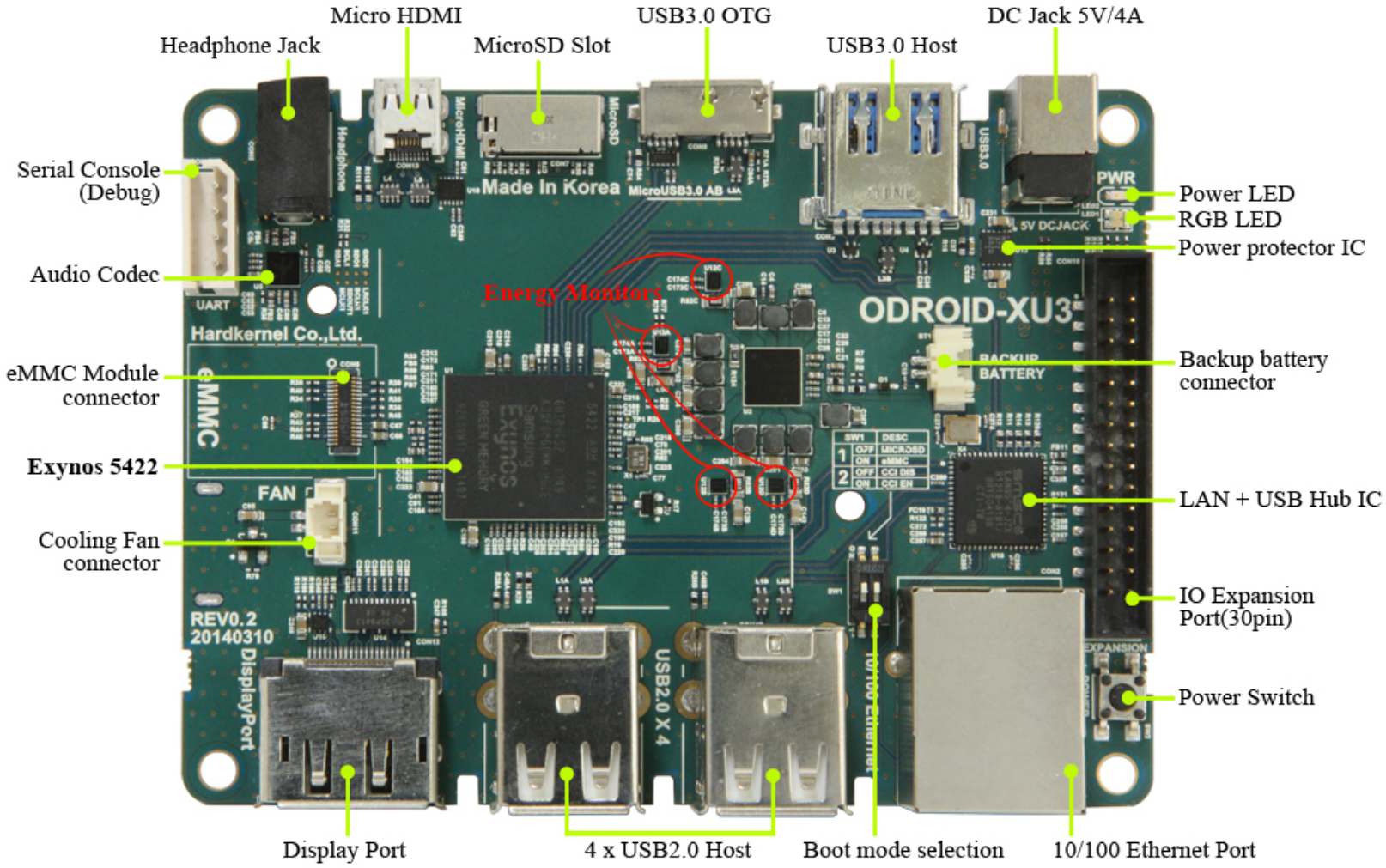
- Introduction – Project Goal
- Hardware
- Software
- Implementation Details
- Experimental Setup
- Results

Introduction – Project Goal

- Performance of Odroid XU3 board while performing parallel AES encryption

Hardware

- ODROID-XU3
- Samsung Exynos5422
  - A15 2.0Ghz quad core
  - A7 1.4Ghz quad core
- RAM: 2GB
- Energy Monitors
  - Big CPU
  - Little CPU
  - GPU
  - DRAM



Software

- AES library
- EnergyMon
  - A Portable Interface for Runtime Energy Monitoring
  - Supports several energy/power monitoring sources
    - TI INA-231 (odroid)
    - MSR
    - etc.

Nickname	Data Source	Data Type	Access	Units (Dec. Precision)	Overflow	Refresh Interval	Interface
msr	Intel MSR via *msr* kernel module [32]	Energy	Shared	*platform-specific*	32 bits	1 ms	dev file(s)
rapl	Intel MSR via *intel_rapl* kernel module [8]	Energy	Shared	microjoules	*platform-specific*	1 ms	sysfs files
odroid	TI INA-231 sensors [20]	Power	Shared	watts (6)	-	264 ms	sysfs files
odroid-ioctl	TI INA-231 sensors [20]	Power	Shared	microwatts	-	264 ms	dev files
osp	ODROID Smart Power via *hidapi* [13, 29]	Energy	Exclusive	watt-hours (3)	1000 Wh	200 ms	USB
osp-polling	ODROID Smart Power via *hidapi* [13, 29]	Power	Exclusive	watts (3)	-	200 ms	USB
wattsup	WattsUp? meter [38]	Power	Exclusive	deciwatts	-	1 sec	USB
shmem	Shared memory provider	Energy	Shared	microjoules	*provider-managed*	*provider-dependent*	memory
dummy	None	-	Shared	-	-	-	memory

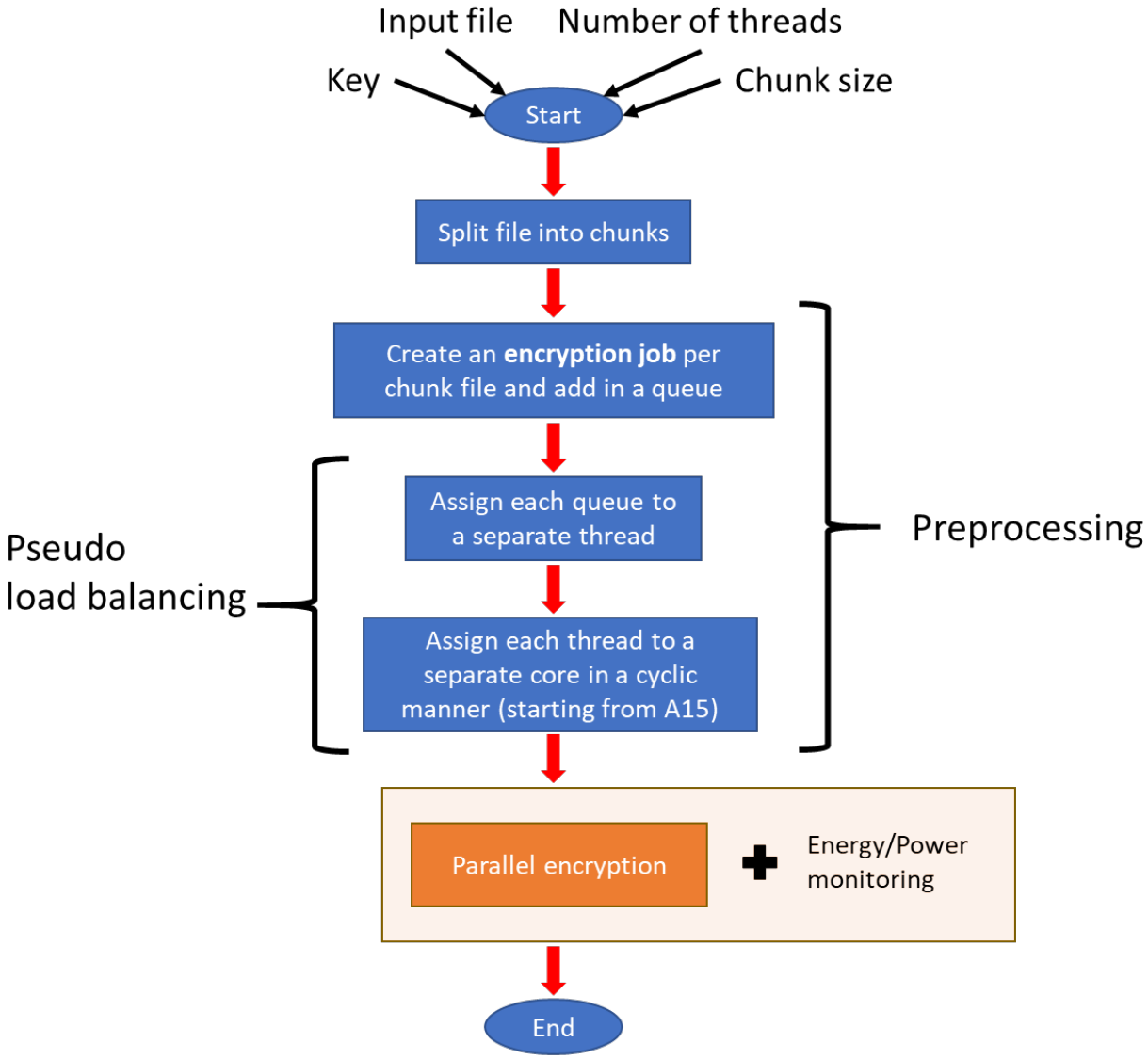
* Connor Imes, Lars Bergstrom, and Henry Hoffmann. 2016. A portable interface for runtime energy monitoring. In *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering* (FSE 2016). ACM, New York, NY, USA, 968-974. DOI: https://doi.org/10.1145/2950290.2983956

Implementation Details

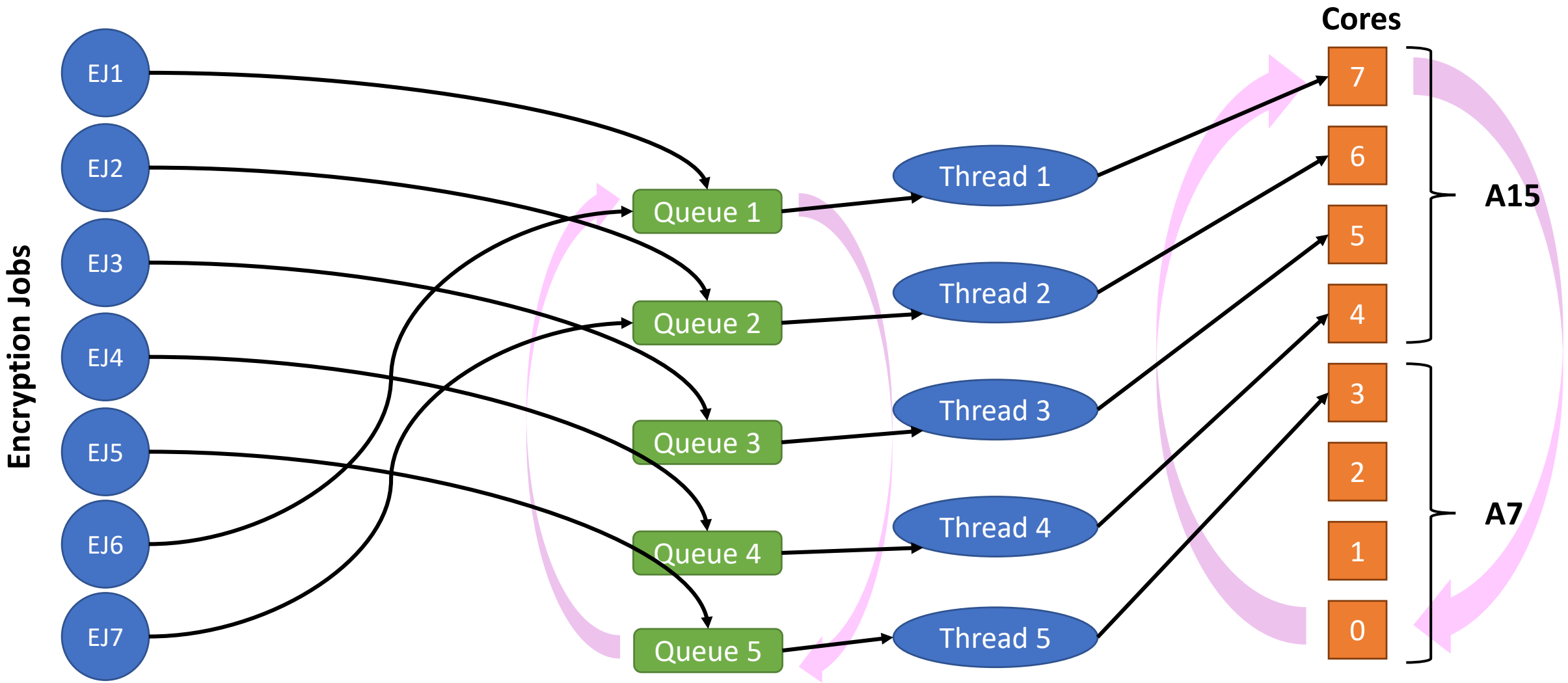
- Input file is split into smaller **chunk files**
- Chunk file <$^{1\text{-}1}$---> **Encryption Job**
- Thread <$^{1\text{-}1}$---> **Encryption Jobs Queue**

```
struct job
{
  unsigned int id;
  unsigned char *iv;
  char in_file_name[256];
  char out_file_name[256];
  unsigned char *key;
  unsigned int *key_length;
  struct aes_ctx ctx;
};
```

Implementation Details – Execution Flow



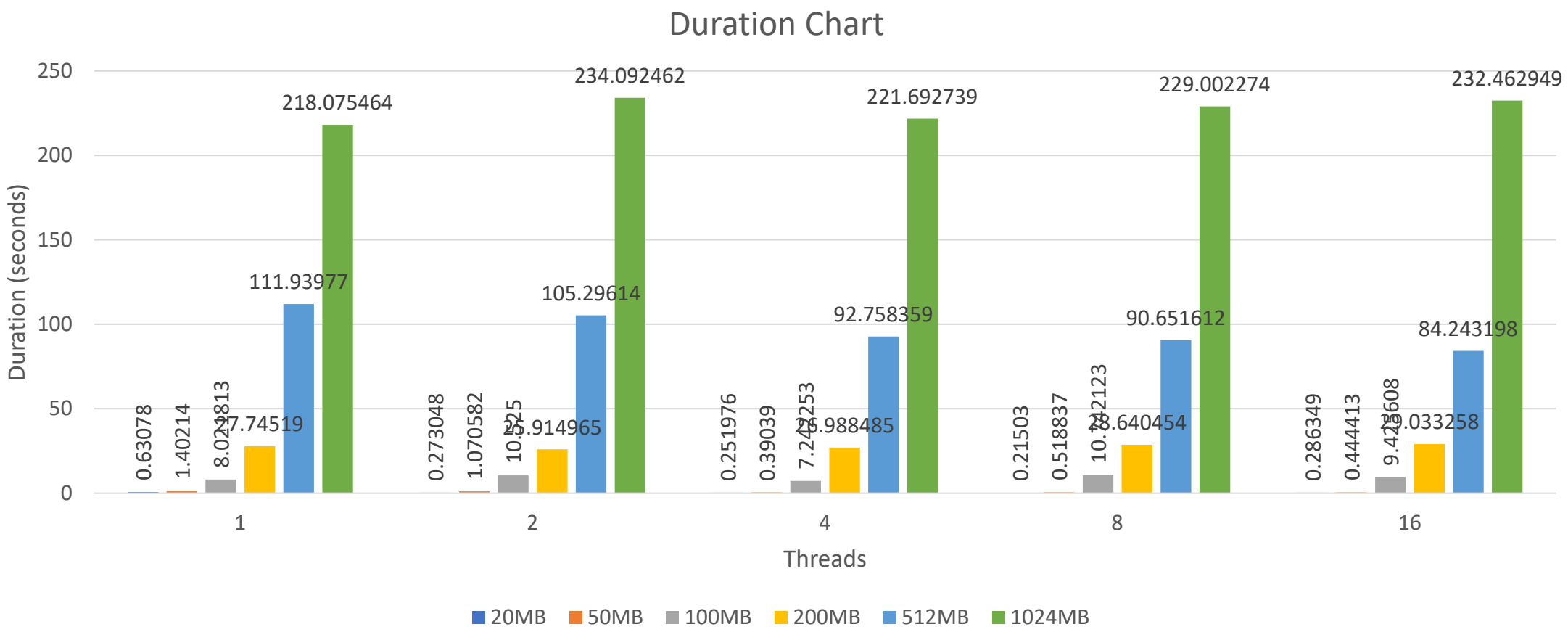
Implementation Details – Pseudo Load Balancing



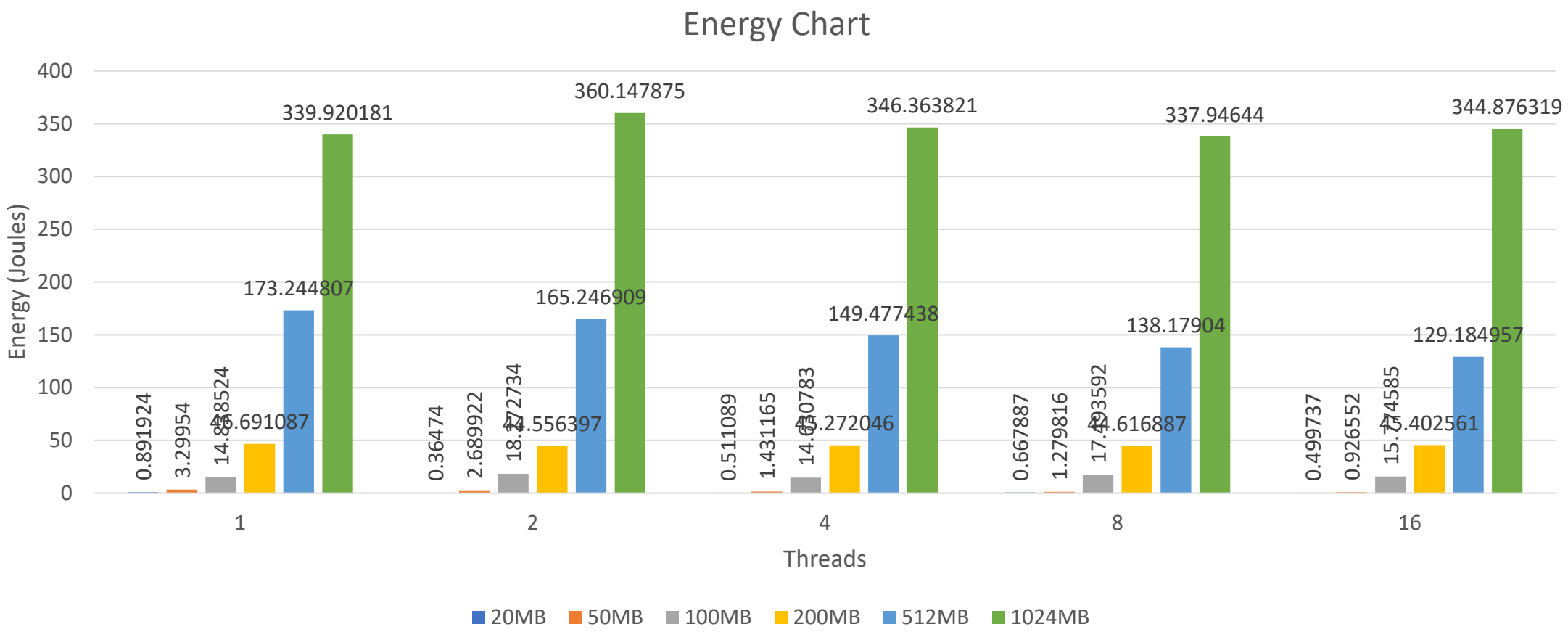
Experimental Setup

- Text-only mode (no GUI)
- AES Key size: 128 bits
- Input files are split into 2 MB chunk files
- Variety of runs:
  - 20, 50, 100, 200, 512, 1024 MB files
  - 1, 2, 4, 8, 16 threads
- **Average of 3 runs** of all above combinations

Results – Duration of parallel part



Results – Energy Consumption



Results – Power Consumption