FYP Oral & Final Report Contents Plan

Final Report

Preamble:

* Title page
* Abstract
* Declaration of Originality
* Project Specification
* Acknowledgements
* Table of Contents
* List of Symbols
  + Mentioned in guidelines, not applicable here
* List of Abbreviations
  + Continue to expand abbreviations on first use
* List of Figures
  + Maybe but not necessary
  + Generally for theses
* List of Tables?
  + Maybe but not necessary
  + Generally for theses

Content:

* Introduction
  + General Overview & Background
    - DSbD
    - CHERI
    - CheriBSD
    - Morello
    - CHERI-LLVM
    - CHERI GDB
    - DPDK
    - Digital Catapult
    - Pytilia
      * Include context for industry
  + Project Objectives
  + Literature Review / Project Motivation & use case
    - Networking example
      * Firewall
    - Previous work by Pytilia
* Project Plan
  + Gantt Chart
    - Split into 2 Gantt charts? Project work and meetings/deadlines
      * Helps with the page height limit
      * Maybe combine some subtasks instead
    - Freeze rows for 2 screenshots
    - Only include the actual timeline or predicted vs actual?
      * Actual with some sentences describing changes from plan
* Project Requirements
  + Part of project plan, before the Gantt charts
  + Deeper dive into project objectives from introduction showing the required steps
* Setup
  + Physical Morello board
  + Install CheriBSD
  + Software setup
    - Install packages
    - SSH & DHCP reservation
    - Set bash shell
    - Set up vimrc
* C Applications
  + NOTE: Add these sample applications to git!
  + Created Applications (may not need to include all of these):
    - HelloWorld
    - Multiplier
    - Student Info
    - Student Calculator
    - Student Grades
* DPDK Port
  + Overview of the build process
  + Port of DPDK 20.11.1
    - Successful native compilation
      * Meson atomic issues
      * Ninja emulated TLS
      * Ninja atomic operations
      * Disabling drivers
      * Successful build
      * Limitation of one process
      * DPDK HelloWorld
    - Unsuccessful cross-compilation
      * Building the Morello FVP
      * Use of PureCap docker container
      * File path changes
      * Cross-file
      * Drivers disable correctly
      * Runs on Morello FVP
      * Fails on Morello board
      * Cannot compile TLS
    - Multiple processes
      * Kernel modules
      * Cannot build and load modules natively
      * Custom CheriBSD image
        + Kernel modules work
      * PureCap capability errors in drivers
        + Out of scope
* Packet Generator Application
  + Python script
  + Generates pcap files for input to the packet processing application
  + Except for initial packet streams, pcap file name is used to parse details
  + Raw ethernet payload defined
  + Tables of packet details
    - Initial packets
    - Variable packet counts
    - Variable packet sizes
    - Variable consumer counts
      * Should this only be included if the corresponding feature is developed?
        + Not here
        + In discussion
  + Packets not generated on the Morello board
    - The SCP command is used to copy them across to the corresponding folder
      * Likely unnecessary, unless relevant
* Packet Processing Application (CHERI Networking)
  + Overview
    - Built using PureCap mode
    - Read in packets from pcap file
    - Store in buffers
      * 1,000 buffers
      * Read in to fill buffers
      * Process packets
        + 3 modes of operation

CHERI-based, single-process operation

Consumer plugin in same process and address space

UDP socket-based, IPC operation

Consumer plugin in separate process and address space

No processing operation mode

Read packets into buffers and clear afterwards

* + - * + Clear buffer
        + Next packet
      * When emptied all buffers, read the next set and repeat
      * When no packets remain, exit the loop
  + Usage
    - Build with DPDK
      * Add to examples
    - Run in terminal
    - Settings
      * How to toggle settings
        + Environment variables
        + CLI arguments

Takes precedence

* + - * Operation modes
      * Quiet and verbose
      * Capability validation
    - Run script
      * Specify build directory, packet stream, CLI arguments
      * Validation of above
  + CHERI-based Security
    - Consumer plugin in same process and address space
    - Specify CLI argument and environment variable
    - When finished processing packets, get the counter for each consumer from plugin
    - Print total packet processing latency to terminal
    - Print consumer counters to terminal
    - Capability validation
      * Pytilia usage
        + Secured and validated buffer
        + Could not reuse buffers reliably
      * CHERI Networking usage
        + Reads into buffer
        + Copies buffer capability
        + Restricts the copy bounds to packet length
        + Removes write permissions
        + Sends this capability to the consumer
        + The consumer processes the packet

Increments a counter

* + - * Bounds
        + Screenshot
        + Attempts to read one character beyond the range of capability bounds

Buffer overrun

* + - * + Running raises an error
        + GDB shows the correct, specific error
      * Permissions
        + Screenshot
        + Attempts to overwrite a character within the capability bounds

Unauthorised access

* + - * + Running raises an error
        + GDB shows the correct, specific error
  + UDP socket-based IPC
    - Consumer plugin in the same process and address space as the packet receiver
    - Specify CLI argument and environment variable
    - Capability validation is not required
    - Print total packet processing latency to terminal
    - When finished processing packets, send a “FINISHED” packet to each packet receiver instance
    - Wait for consumer counts to be sent back from packet receivers
    - Print consumer counts to terminal
  + No processing operation mode
    - No consumers required
    - Specify CLI argument and environment variable
    - Read in packets to buffer
    - Clear buffers without processing packets
    - Print total packet processing latency to terminal
    - Print consumer counters for consistency with other execution methods
      * Always a count of 0
* Packet Receiver Application
  + Overview
    - UDP socket-based IPC
    - Runs in its own process and address space
    - Can be run in the background
    - Begin execution before CHERI Networking
      * Waits indefinitely until packets are received
    - Receives packets in sequence
    - Increments consumer count
    - Upon receiving a "FINISHED" packet, sends back the final consumer count
  + Usage
    - Build with the make command
    - Run in terminal
    - Consumer number
      * Generates ports
    - Quiet or verbose flag or environment variable
      * Not required
* Scripts
  + Appendices or GitLab
  + Less detail, note work done
  + Run
    - Bash
    - Runs CHERI Networking alone
    - Specify build directory
      * Validated
    - Specify packet stream
      * Validated
    - Specify CLI arguments
      * Validated
  + Time
    - Bash
    - Same functionality as run script
    - Uses time terminal command to get execution time
  + Record performance metrics
    - Python
    - Measures total CPU utilisation during execution of CHERI Networking
    - Parses output packet processing latencies
    - Runs time script and parses execution time
    - Outputs these 3 metrics to file
    - File structured in results directory
      * Variable being tested
      * Processing type
* Testing metrics / requirements
* Results
  + Bash alias to run all performance metrics
  + Averaged over X iterations
  + Python script to graph results
  + The first set of results?
    - Graphs averaged from 5 readings
    - Comparison of graphs
      * Packet processing latency
      * Total CPU utilisation
      * Per core CPU utilisation?
      * Execution time?
    - Printed packets to the terminal
      * Heavily affected output
      * Reason for quiet and verbose flags
  + The second set of results
    - Graphs averaged from 7 readings
    - Comparison of graphs
      * Packet processing latency
      * Total CPU utilisation
      * Per core CPU utilisation?
      * Execution time?
    - Quiet flag applied
    - CHERI mode had very high increase in packet processing latency
      * Didn’t make sense
      * Caused no processing mode to be added
  + The third set of results
    - Graphs averaged from 10 readings
    - Comparison of graphs
      * Packet processing latency
      * Total CPU utilisation
    - Quiet flag applied
    - Also ran no processing mode
    - Unaltered results and adjusted results
    - Matched expected output
  + The fourth set of results
    - After code refactor
    - Graphs averaged from 10 readings
    - Comparison of graphs
      * Packet processing latency
      * Total CPU utilisation
    - Quiet flag applied
    - Also ran no processing mode
    - Unaltered results and adjusted results
    - Matched expected output
* Discussion
  + DPDK Port
  + Packet Generation
  + CHERI Networking/Packet Receiver
  + Results
  + Issues
  + Future Work
    - Compartmentalisation
    - DPDK (Updated)
* Conclusions
  + Project objectives met

Postscript:

* Appendices
  + Tables of complete results
  + Bash alias
  + vimrc?
* References
  + IET style

Format:

* A4
* 1 in (2.54 cm) left-hand margin
* 1.5 spacing
* Times New Roman size 12
* Page numbers in the top-right corner
* Page numbers count in Roman numerals before the main content and Hindu-Arabic numerals for the main content
* Number figures & tables by section, e.g. 3.1, 3.2, 4.1, etc
* IET style references

General Questions:

* Is it OK to copy/paste sections from the interim report?