BÁO CÁO THỰC TẬP – gNodeB 5G VHT

Linux Programming Assignment

Sinh viên: Luyện Huy Tín – K64-K2

1. Create 3 thread SAMPLE, LOGGING, INPUT.

1.1. Thread SAMPLE

Thread SAMPLE thực hiện vô hạn lần nhiệm vụ sau với chu kì X ns. Nhiệm vụ là đọc thời gian hệ thống hiện tại (chính xác đến đơn vị ns) vào biến T.

- Ý tưởng: Đối với thread sample, ta tạo 1 định danh sau đó viết hàm main cho thread sample có tên là void *currently_time(Hàm này in ra giá trị thời gian thực chính xác đến từng s và ns). Sau đó, ta sử dụng clock_nanosleep() thay cho nanosleep() để có được kết quả chính xác hơn.

1.2. Thread INPUT

Thread INPUT kiểm tra file "freq.txt" để xác định chu kỳ X (của thread SAMPLE) có bị thay đổi không?, nếu có thay đổi thì cập nhật lại chu kỳ X. Người dùng có thể echo giá trị chu kỳ X mong muốn vào file "freq.txt" để thread INPUT cập nhật lại X.

- Ý tưởng: Đối với hàm input, ta tạo 1 hàm long get_freq để đọc file freq.txt. Ta tạo 1 giá trị long x có giá trị bằng giá trị bên trong file freq.txt bên trong hàm để đối chiếu với tham số truyền vào. Nếu tham số bằng x ta trả về NULL, nếu không ta thực hiện cập nhật lại giá trị bên trong file

1.3. Thread LOGGING

Thread LOGGING chờ khi biến T được cập nhật mới, thì ghi giá trị biến T và giá trị interval (offset giữa biến T hiện tại và biến T của lần ghi trước) ra file có tên "time_and_interval.txt".

- Ý tưởng: Đối với hàm loggin, ta tạo 2 biến (1 biến có độ chính xác là s và 1 biến là ns) sử dụng hàm strtol để convert chuỗi trong file time_and_interval sang kiểu long, sau đó thực hiện cập nhật thời gian thực rồi trừ đi 2 biến đó để ra được offset, sau đó lưu vào file time_and_interval.

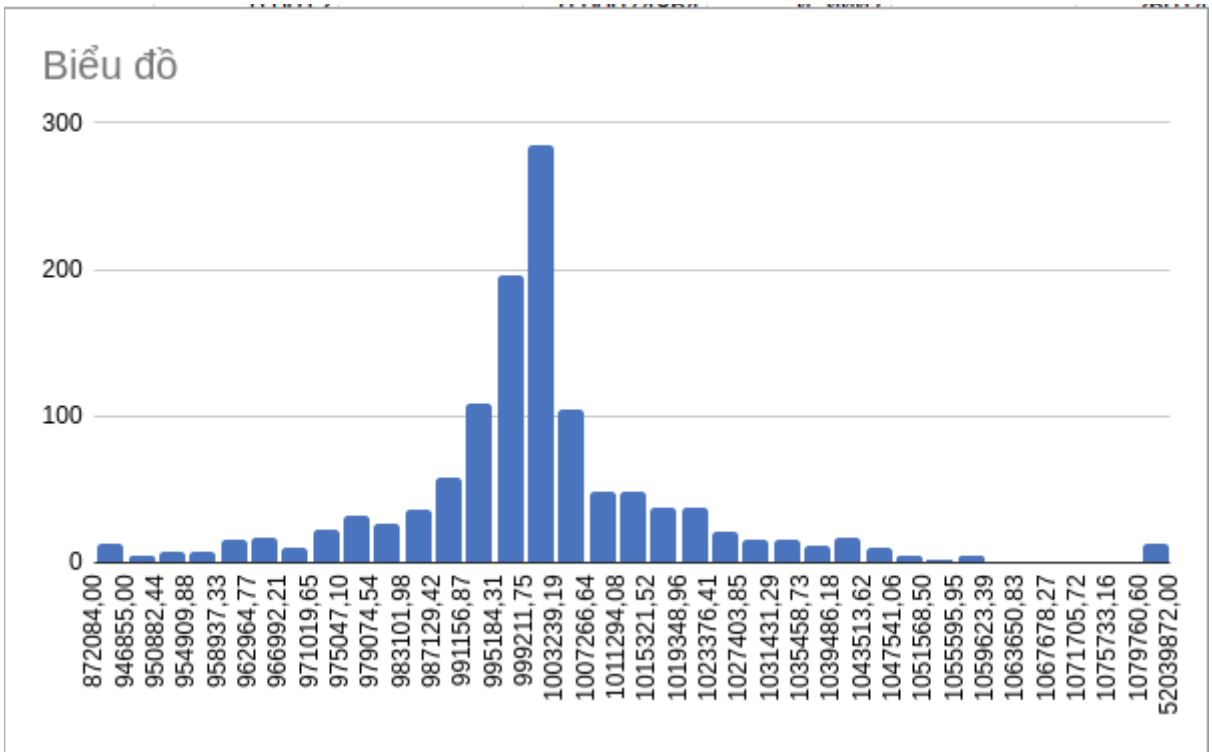
2. Shell script để thay đổi giá trị chu kỳ X

- Ý tưởng: dùng lệnh echo để thực hiện ghi đè vào file freq.txt các giá trị lần lượt là 1000000ns, 100000ns, 10000ns, 1000ns, 100ns. Với mỗi giá trị ta thực hiện chạy timeout trong vòng 60s. Tổng cộng hết 5 phút

3. Khảo sát giá trị

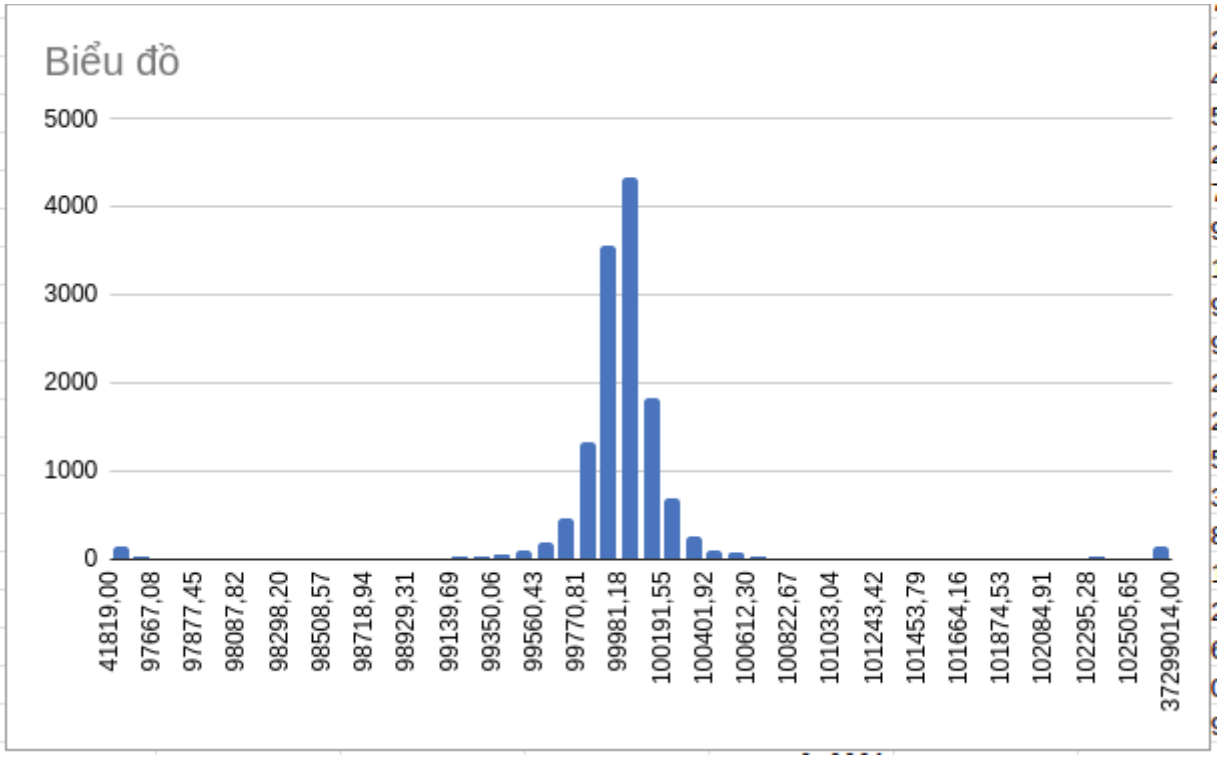
Thực hiện vẽ đồ thị histogram với mỗi giá trị interval lưu trong file. Đánh giá và đưa ra kết luận. Cụ thể:

Với freq = 1000000ns



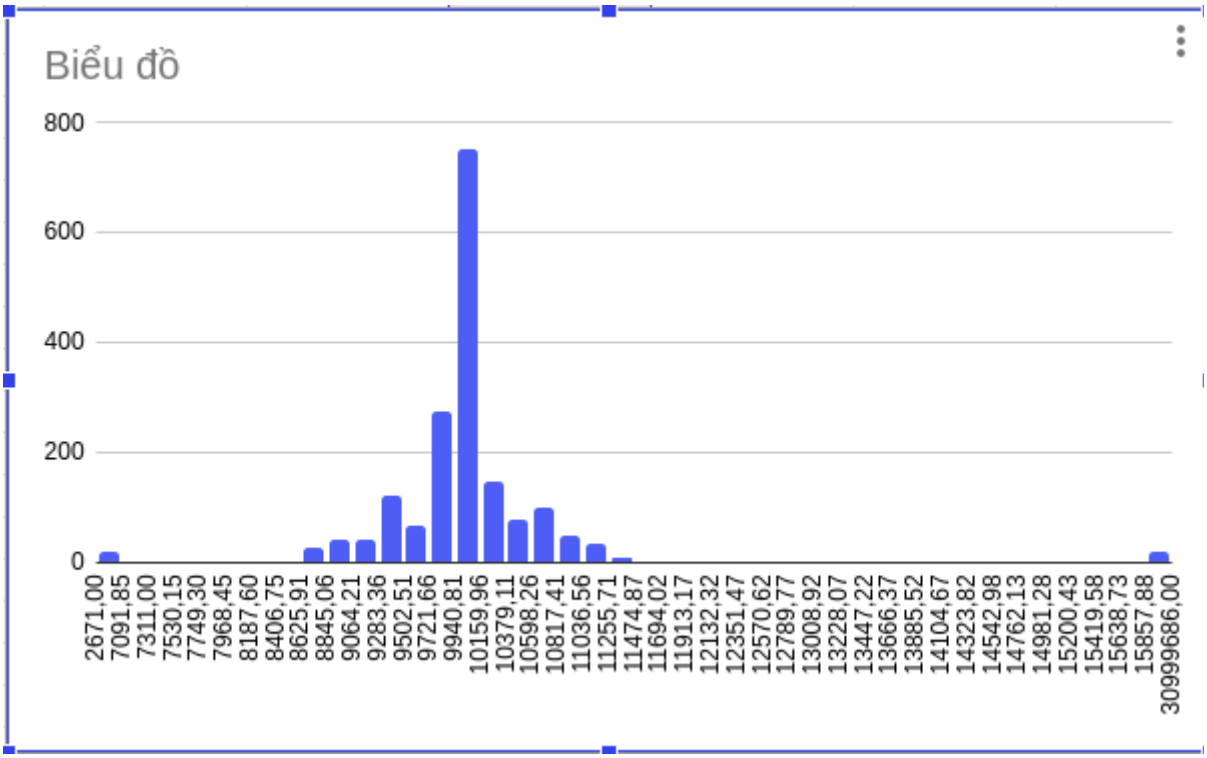
ta có giá trị offset giao động quanh mức 995184ns – 1003239ns

Với freq = 100000ns



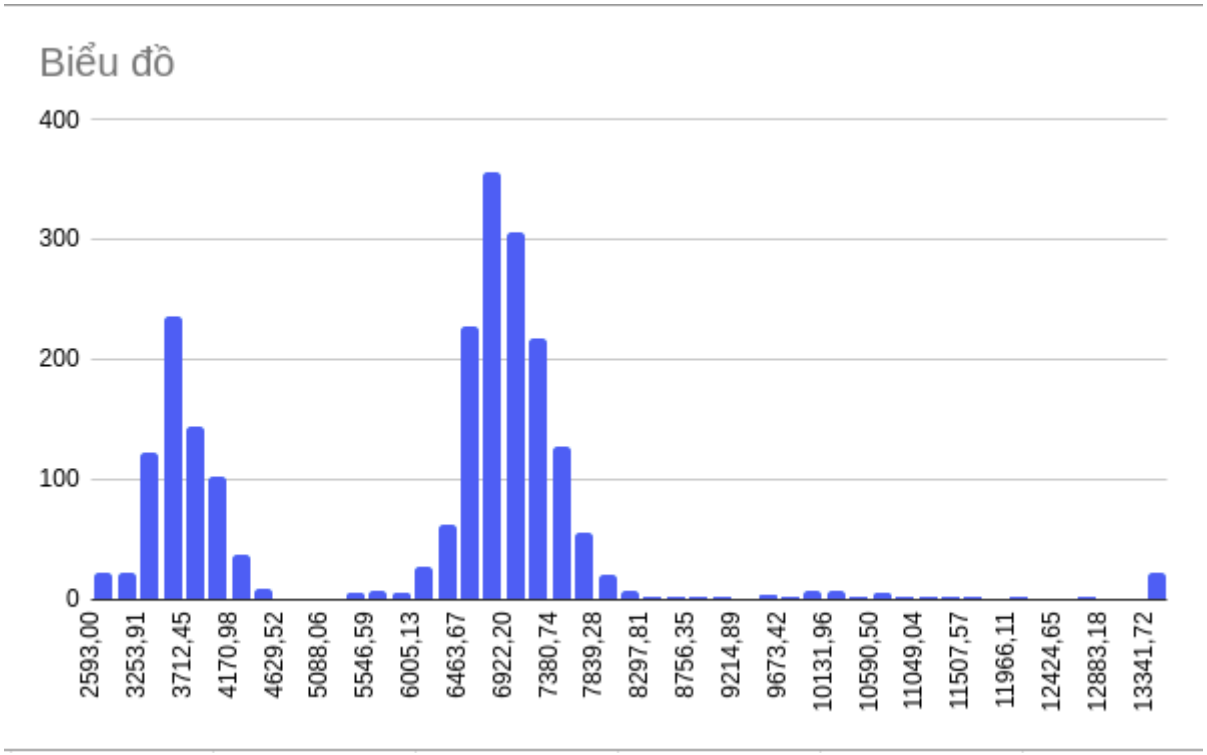
Ta có giá trị offset giao động quanh mức 99981ns – 100191ns

Với giá trị freq = 10000ns



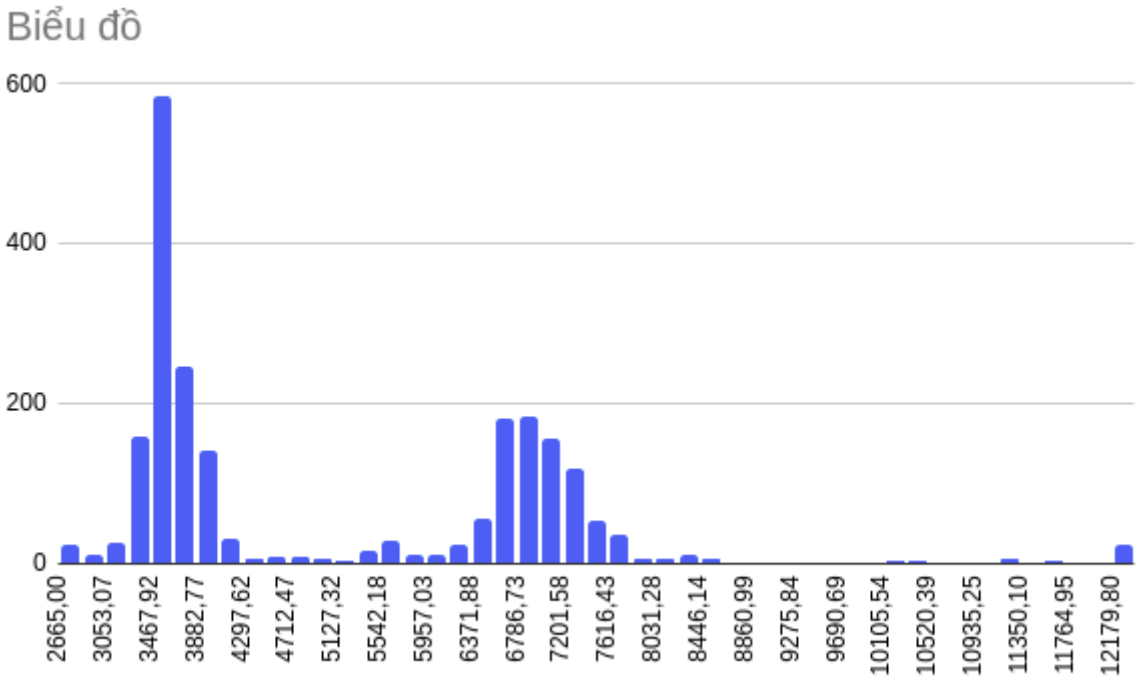
giá trị offset giao động quanh mức 9940ns – 10159ns

Với giá trị Freq = 1000ns



giá trị offset giao động quanh mức 3253ns-3712ns và 6463ns-7380ns

Với giá trị offset = 100ns



gía trị offset giao động quanh mức 3467ns – 3882ns và 6786ns - 7201ns

4. Kết luận

Với giá trị freq 1000000 thì giá trị offset lớn tuy nhiên khá ổn định, với freq 100000, 10000 thì offset bé và độ ổn định vẫn tốt, với các giá trị còn lại thì offset không còn tính ổn định

5. Những vấn đề gặp phải và tiến độ hoàn thành công việc

- Đã hoàn thành

- Môt số khó khăn gặp phải:

+ Chưa hiểu rõ về multithread, các câu lệnh trong Linux => Mất khá nhiều thời gian để tìm hiểu và áp dụng

+ Độ chính xác từ kết quả chưa được cao

+ Xử lý vấn đề vẫn còn khá chậm và chưa linh hoạt với những yêu cầu mới