CC3301
Programación de Software de Sistemas
Profesor: Luis Mateu

- Propósito del curso
- Resultados de aprendizaje
- ¿Por qué aprender C, Risc-V y Linux?
- Programa de curso
- Bibliografía
- Evaluación
- Trabajo para la casa

Propósito del curso

El propósito del curso CC3301 Programación de software de sistemas (PSS) es que los/las estudiantes escriban y mejoren programas en lenguaje C, que requieren hacer un uso eficiente de la plataforma, y por lo tanto necesitan conocer su arquitectura de hardware y la interfaz de programación de aplicaciones (API) del sistema operativo.

Resultados de aprendizaje

En este curso aprenderán a:

- escribir programas eficientes en el lenguaje C
- usar el debugger *ddd* para diagnosticar errores
- usar *sanitize* para detectar errores de programación como fugas de memoria o punteros locos
- identificar a qué instrucciones assembler Risc-V se compilan los programas en C
- escribir trozos de programas en assembler Risc-V
- escribir programas que usan directamente los servicios provistos por Linux (el núcleo de sistemas operativos como Debian, Ubuntu, Android, etc.)
- Crear procesos paralelos para aprovechar todos los cores del computador y así disminuir el tiempo de ejecución

Acerca del profesor Luis Mateu

- Profesor de “jornada parcial” del DCC
- Solo me dedico a los 3 cursos que dicto para la facultad
- Los otros 2 cursos son arquitectura de computadores y sistemas operativos
- Trabajé ~ 5 años en Synopsys: empresa con sede en California y centros de investigación y desarrollo en varios países, incluyendo uno en Chile
- Synopsys es líder en software para diseñar circuitos digitales, usado por Intel, AMD, nVidia, Qualcomm, etc.
- Trabajé en el mantenimiento de *Design Compiler*, un compilador de Verilog/Vhdl de unas 10 millones de líneas de código en C que corre principalmente bajo Linux

¿Por qué aprender a programar en C?

- Es el lenguaje preferido cuando se necesita eficiencia, por ejemplo, para programar un decodificador de video
- Existe una amplia base de software escrito en C y que necesita mejorarse:
  - ✓ Resolver bugs
  - ✓ Agregar funcionalidades
  - ✓ Portar a nuevas plataformas
  - ✓ Mejorar la eficiencia
- Ejemplos: los núcleos de Linux (C) y Windows (C y C++), el intérprete de Python, el compilador just-in-time de Java, administradores de bases de datos, etc.
- Desventajas:
  - No es robusto
  - Es inseguro
- Futuro ficción: será reemplazado por Rust

¿Por qué aprender assembler Risc-V?

- Porque es el assembler más fácil de aprender
- Hace más fácil aprender assemblers más complejos como x86 y Arm
- Se requiere programar en assembler para usar de la manera más eficiente las instrucciones especializadas de los procesadores como manejo de vectores
- Algunos bugs son tan complejos que requieren revisar el assembler generado por el compilador
- Risc-V se posiciona en términos de uso como la tercera plataforma después de Arm y x86
- Y no parará de crecer porque un fabricante de chips no necesita pagar una licencia por usar el set de instrucciones de Risc-V
- Hay diseños avanzados de Risc-V open source y gratuitos
- Diseñado para que los fabricantes puedan agregar instrucciones aceleradoras del cálculo de operaciones gráficas, de redes neuronales, de encriptación, etc.

¿Por qué aprender un sistema operativo basado en Linux?

- Linux es una reimplementación del núcleo del sistema operativo del legado Unix
- Android, el sistema operativo de los celulares, usa un núcleo derivado de Linux
- iOS y OS X (de Apple) son derivados de Unix
- La mayoría de los servidores en Internet usan un sistema operativo basado en Linux: CentOS, RedHat o Debian
- Los notebooks y computadores de escritorio típicamente usan el sistema operativo Windows, pero *WSL 2* de Microsoft también permite correr en Windows los sistemas operativos Debian, Ubuntu y otros basados en Linux, con todas sus aplicaciones

¿Qué es la programación de software de sistemas?

De acuerdo a Wikipedia se trata de programar software:

- Que provee servicios a otro software
- Que requiere alto desempeño
- O ambos

Ejemplos: núcleos de sistemas operativos, compiladores, aplicaciones de ciencia computacional, motores de juegos, automatización industrial, etc.

Típicamente se programan en el lenguaje C

Lo opuesto es la *programación de software de aplicaciones* como un sistema de administración para una empresa, scripts para páginas web, un juego de salón, etc.

Finalmente: ¡es subjetivo!

Programa de curso

- Programación en el lenguaje C: tipos, sintaxis de las instrucciones, direcciones de memoria, punteros, errores típicos
- Arquitectura de computadores: Risc vs. Cisc, assembler Risc-V, compilación de programas en C a instrucciones Risc-V
- Implementación de la CPU: circuitos digitales, la memoria, diseño de la cpu, ejecución secuencial, en pipeline, superescalar y fuera de orden.
- Linux: estudio de la API de Linux/Unix (*applicaction programming interface*), el shell de comandos, cómo se crean los procesos y se cargan los programas, paralelización con múltiples procesos

Bibliografía

- Se publicarán en la sección novedades de U-cursos:
  - ✓ Videos de las clases de este semestre y/o de semestres anteriores
  - ✓ Pdf de las presentaciones
  - ✓ Material para probar los ejemplos de las clases
- En página Web: https://users.dcc.uchile.cl/~lmateu/CC3301/
  - ✓ Instrucciones para instalar la distribución oficial de Linux en este curso: Debian 12
  - ✓ Controles de semestres pasados
- Material complementario:
  - ✓ Apuntes del curso (necesito actualizarlos)
  - ✓ Kernighan, B y Ritchie, D (1988) "The C Programming Language"
  - ✓ Richard Stones, Neil Matthew (2003), "Beginning Linux Programming (Programmer to Programmer)"

Evaluación

- Nota de controles: 50 %
  - ✓ 3 controles en el horario extendido de la clase auxiliar: 60 %
  - ✓ Examen: 40 %
  - ✓ Tienen un fin evaluativo
- Tareas: 50 %
  - ✓ 8 tareas cortas (no se elimina ninguna tarea)
  - ✓ Son tareas individuales
  - ✓ Tienen un fin formativo
  - ✓ Debe pasar los tests de prueba en Debian 12. Si no pasa el test de prueba en ese sistema operativo la nota es 1.0.
  - ✓ Se descuentan 5 décimas si la compilación arroja warnings o si está mal indentado
  - ✓ Se puede entregar con algunos días de atraso, pero se descuentan 5 décimas por día
  - ✓ Puede recibir ayuda para lograr el correcto funcionamiento de su solución
- Requisitos de aprobación: nota de controles igual o superior a 4 y nota de tareas igual o superior a 4

Importante

- No intente resolver las tareas sin haber estudiado la materia correspondiente primero. Cometerá errores que no será capaz de entender y terminará perdiendo más tiempo que el que se ahorró al no estudiar.
- Si sobrepasa el tiempo nominal publicado para resolver la tarea, pida ayuda.
- Pedir ayuda no es copiar cuando el código lo escribió Ud. mismo.
- Copiar un fragmento de código de la tarea de un compañero ***sí es copia***.
- Puede pedir ayuda a un compañero.
- O por correo a los 3 profesores de este curso. Alguno podría atender su consulta.
- Una tarea en C puede correr exitosamente en su computador y fallar completamente en plataformas ligeramente distintas. Su tarea será corregida bajo Debian 12. Asegúrese de que funciona en esa plataforma.
- Se recomienda formar un grupo de estudio

Trabajo para la casa

- Instale el Linux oficial del curso (Debian 12): siga las instrucciones que aparecen en esta página en la sección cómo correr Debian en su computador.
- Estudie este tutorial que le enseñará los comandos básicos de Linux. Abra un terminal y experimente con estos comandos.
- Antes de la clase del jueves: estudie en los apuntes del curso la sección principios básicos del lenguaje C
  - Compile y ejecute todos los ejemplos dados
  - Complete el ejercicio final: factorial