TRADUÇÃO DO LIVRO DESIGN E IMPLEMENTAÇÃO DO SISTEMA OPERACIONAL MTX

PREFÁCIO

O propósito desse livro é prover uma plataforma suitável para ensinar e aprender a teoria e prática de desenvolvimento de sistemas operacionais. Abrange os conceitos e princípios de sistemas operacionais, e demonstra como aplicar os mesmos no designer e implementação de um sistema operacional completo gerado e produzido passo-a-passo. Esse livro usa o desenvolvimento do sistema operacional MTX para transmitir a real intenção desse livro, o qual tem prentenção de apresentar os princípios de design e técnicas de implementação de sistemas operacionais em geral.

Devido seu conteúdo técnico, esse livro não é indicado para cursos que ensinam somente, em nível introdutório, os conceitos básicos e princípios de sistemas operacionais sem a pratica da programação. É indicado para tecnicamente para cursos de sistemas operacionais que têm foco, em ambos, na prática e teoria. O estilo envolucional do livro, acoplado de código fonte detalhado e completo sistemas simples de trabalho, torna-lo especial suíte para estudo próprio. Esse livro contém vários novos e originais materiais especialmente no designe e uso de algoritimos paralelos em SMP – Symetric Multiprocessing, processamento simétrico e paralelo de processos (programas, aplicativos etc). Também é um guia para estudo avançado de sistemas operacionais.

Empreender este projeto de livro demonstrou ser um grande desafio. Enquanto este livro estava sendo preparado para publicação, eu tive bênçãos de encorajamentos e ajudas de varias pessoas. Eu gostaria de dedicar essa oportunidade para agradecer a todos. Eu também agradeço a Publicadora Internacional Springer AG por permitir me de dispor o código fonte de esse livre gratuito e publicamente. Espero, que o livro ajude os desenvolvimentos do sistema MTX e que possa-se aprimorar esse livro no futuro.

O código fonte incluído no livro pode ser baixado diretamente do website do autor em <http://www.eecs.wsu.edu/~cs460/mtxhome>.

CAPITULO 1

INTRODUÇÃO

* 1. SOBRE ESSE LIVRO

Esse livro é sobre o design e implementação de sistemas operacionais. Um sistema operacional (SO) é um série de programas e arquivos de suporte, os quais rodam em um sistema de computador para que esse sistema possa ser conveniente e de fácil uso. Sem um sistema operacional, um sistema de computador é essencialmente sem utilidade. O estudo de sistemas operacionais envolvem uma ampla variedades de matérias subjetivas em ciência da computação e engenharia da computação. Isso inclui arquitetura de computador, programação de computador e, o mais importante, interface entre o software e hardware do computador. Prover uma completa serie de conhecimento sobre sistemas computacionais. Tal conhecimento não é somente necessário para profissionais da computação mas também beneficia todos os tipos de esforços na careira relacionada a computação. Esse livro abrange os conceitos básicos e princípios de sistemas operacionais. Mostra-se como aplicar esses conceitos e princípios para fazer design e implementação do sistema operacional MTX em detalhes. MTX é um pequeno sistema baseado em Unix, com intenção principalmente ao ensinamento e aprendizagem da teoria e pratica de sistemas operacionais. Designado para PC´s Intel x86, e roda em PCs reais e maquinas virtuais que emulam o hardware do PC. Escrito em grande parte na linguagem de alto nível C, com pouco de 2% em código assembly. O núcleo do MTX inclui gerenciamento de processo e memoria, I/O drivers de dispositivos e um compatível arquivo de sistema Linux EXT2. Permite que usuários façam login via console do PC e via portas seriais, e suporta execuções de comandos simples com redirecionamento I/O, bem como mutiplos comandos conectados por pipes. Ao invés de apresentar um sistema completo em única etapa com tentativa de dissecar o sistema, esse livro mostra o desenvolvimento de um sistema operacional em etapas que evoluem-se progressivamente, iniciando a partir de um simples programa com múltipla tarefa, ate um completo sistema operacional. Em cada etapa, mostra-se como aplicar os conceitos e princípios de sistemas operacionais para que seja realizado o design e implementação de vários componentes de sistemas. Cada etapa consiste em uma pequena do sistema, o qual permite ao leitor que teste e observer as operações de um sistema operacional a partir de seu interior. Em cada estapa sucessiva, adiciona-se novas características ao sistema e expande-se a capacidade do sistema enquanto se caminha ao resultade de um sistema operacional completo. O design e implementação do MTX abrange o aspecto quase que absoluto do sistema de arquiteturas baseadas no Intel x86, partindo do modo real de 16-bit ate o modo protegido de 32-bit e eventualmente ate o multiprocessamento simétrico (SMP). Desconsiderando as diferenças de arquitetura, demonstra-se que os mesmos princípios de design e técnicas de implementações ode ser aplicado em todos os casos.

* 1. OBJETIVO E PUBLICO ALVO

O objetivo desse livro é prover uma plataforma suitavel para ensinamento e aprendizagem da teoria e pratica de sistemas operacionais. Use-se o desenvolvimento do sistema MTX visando demonstrar os princípios de design e técnicas de implementações de sistemas operacionais em geral. Devido ao conteúdo técnico, esse livro não é indicado para cursos de SO que ensinam somente os conceitos básicos e princípios de SO sem qualquer pratica de programação. É indicado aos estudantes de ciência da computação e profissionais da computação, que desejam estudar os detalhe internos de sistema operacionais. Esse livro abrange ambos aspectos práticos e teóricos de sistemas operacionais. Descreve o design e implementação de um sistema operacional real em detalhe. É um guia como os manuais tecnicamente orientados a cursos de sistemas operacionais que tem fim de servir de balança entre a teoria e a pratica. O estilo evolucional do livro, acoplado com código fonte detalhado e sistemas simples de trabalho completos, torna esse livro essencialmente um guia para próprio estudo de entusiastas da computação. O livro contem vários novos e originais materiais, especialmente no design e uso algoritimos paralelos em SMP. É também um guia para estudo avançado de sistemas operacionais.

* 1. CARACTERISTICAS ÚNICA DESSE LIVRO

Esse livro tem varias características únicas que diferenciam-lo de outros livros.

1. O livro é auto-organizado. Capitulo 2 abrange todas informações fundamentais e de segundo-plano que são necessárias ao estudo de sistemas operacionais. Isso inclui a CPU e operações I/O , endereços virtuais e reais, etapas de desenvolvimento de programas, montadores de programas em C com código assembly, imagem de execução, convenções de chamadas de sistema e tempo-real no uso da pilho. Quando se fala driveres de dispositivos explana-se as operações de cada dispositivo antes de mostrar o driver atual quanto ao seu design e implementação.Quando se fala dos sistemas de arquivo descreve-se operações de arquivo em detalhe antes de mostrar a implementações de um sistema de arquivo completo. Quando se trabalho com PCs em modo protegido de 32-bit também explana-se as operações do modo protegido em detalhes.
2. Capitulo 3 prover uma completa abragencia acerca da inicialização de sistemas operacionais, o qual não está presente em outros livros de SO. Complementado ao principio de inicialização, mostra-se como escrever carregadores de sistemas operacionais para carregar-se sistemas operacionais reais, tal como Linux, a partir de uma variedade de dispositivos carregáveis. Todos os carregadores usados no MTX são desenvolvidos no capitulo. Eles são comparáveis ao GRUB (GNU GRUB 2010) e isolinux (Syslinux 2013) em performance.
3. Esse livro mostra o design e implementação de um sistema operacional completo com etapas evolutivas e sucessivas.No capitulo 4, usa-se um programa simples para introduzir o conceito de processo e demonstrar o principio da mudança de contexto – salvar o estado do processo atual e preparar para que a cpu passe a dedicar ao processo conseguinte ao atual. Usa-se o modelo de ciclo de vida de processo para ilustrar o principio de multiplatarefa. O mesmo principio é usado no decorrer do livro para criar e gerenciar imagens de processo, independente do hardware gerenciador de memoria do sistema. Essa abordagem única ajuda ao leitor melhor entender gerenciamento de processo em um sistema operacional.
4. Esse livro inicia o desenvolvimento do MTX em modo real de 16-bit. Então extende o modo real do MTX ate o modo protegido de 32-bit e eventualmente ate SMP – Multiprocessamento Simetrico. Despido das diferences de arquitetura, demonstra que o mesmo principio de design e técnica de implementação pode ser aplicada a todos os casos.
5. Em vez de abranger somente conceitos e princípios de threads, esse livro mostra como extender-se o modelo de processo para que se possa implementar suporte de threads e demonstrar aplicações de threads por programas concorrentes.
6. MTX usa uma mistura de ferramentas para sicronização de processo, os quais são escolhidos para melhor ajudar as necessidades, e justificar o uso deles. No modo real de 16-bit, usa-se sleep/wakeup para gerenciamento de processo mas usa-se semaphores nos drivers de dispositivos e para melhor eficiência do sistema de arquivo. Também permite uma transição suave do ambiente uniprocessador para SMP. Por complementação, mostra-se como aplicar alguns dos clássicos problemas de sicronização de processo para o design do SO. Especificamente, usa-se o problema produtor de consumo como modelo para pipes e aplica ao problema leitor-escritor para gerenciamento de I/O buffer em SMP para improvisar concorrência.
7. Capitulo 8 abrange processamento de interrupção e agendamento de processo. Abrange interrupção de hardware do PC e processamento de interrupções em detalhes, e apresenta um geral framework o design de drivers de dispositivos como interrupção-driver. Explana por que manipuladores de interrupção não devem dormir ou ficar bloqueados. Discute os objetivos, politicas e algoritimos do agendamento de processo, e demonstra as diferenças dos efeitos dos algoritimos dos agendamentos de processo através de sistemas simples.
8. Capitulo 9 apresenta um tratamento unificado de interrupções e sinal de processamento, que ajuda clarear os papeis intencionais e uso adequado de sinais. Complementamente, usa exemplos para ilustrar os conceitos e técnicas de coletores de sinal.
9. Capitulo 12 abrange gerenciamento de I/O buffer. Discute o algoritimo de gerenciamento de buffer Unix (Bach, 1990) e aponta suas deficiências. Então mostra como fazer design de novos algoritimos para melhorar a performance do cache de buffer. Varios materiais apresentados nesse capitulo são novos e originais.
10. Explana a relação da hierarquia de operações de arquivo em um SO, a partir do espaço de usuário ate baixo espaço de núcleo (kernel) ate nível de driver de dispositivo. Então mostra a implementação de um sistema de arquivo Linux compatível em detalhes, sistema de arquivo EXT2.
11. Na ordem mostra a diferença de capacidades de segmentação e paginação, apresenta três versões diferentes do MTX no modelo protegido; mtx32.1 usa segmentação, mtx32.2 usa paginação estática e pmtx usa paginação dinâmica.
12. Capitulo 15 no SMP é uma variedade única. Mostra como configurar sistemas SMP-compativel e explana a sequencia inicial de sistemas SMP-compativel. Extende os mecanismos de sicronização de processo convencional, tal como sleep/wakeup e semaphores, ate ambiente de SMP suitavel. Apresenta uma metodologia geral para design do kernel SMP, e mostra como aplicar os princípios para adaptar o kernel pmtx para SMP. Então foca no design e usa algoritimos paralelos para SMP. Especificamente, usa algoritimos paralelo em gerenciamento de processo, gerenciamento de recurso, pipes e I/O buffers para melhorar ambos a concorrência e eficiência. Esse nova abordagem do SMP é novo e original. Compara kernel monolítico e microkernel baseado no SO design e demonstra um sistema hibrido, que incorpora a força de ambas abordagens. No decorrer do livro, usa-se sistemas simples para demonstrar os princípios de design e técnicas de implementações.
    1. CONTEÚDO DO LIVRO

Esse livro está organizado como asseguir, capitulo 2 abrange as fundamentações e segundo-plano que são necessários para estudar sistemas operacionais. Esses incluem operações da CPU, endereços real e virtual, operações de I/O, etapas de desenvolvimento de programa, motadores C com código assembly, imagem de execução de programa, funções de chamadas e pilha em uso em tempo-real. Tambem abrange alguma informação básica do sistema de arquivo EXT2.

Capitulo 3 abrange carregamento do sistema operacional. Discute o carregamento do SO em geral e descreve o processo de carregamento a partir de dispositivos real e virtual em detalhes. Esses incluem carregamento a partir do floppy disco, partições de hard disco, CD/DVD-ROM e USB drivers. Em vez de escrever um carregador customizado para carregar somente o MTX, mostra como desenvolver programas de carregadores para carregar outros sistemas operacionais, tal como Linux, a partir de uma variedade de dispositivos carregáveis.

Capitulo 4 inicia o desenvolvimento do kernel do MTX. Primeiro, usa um programa simples para introduzir o conceito de processo e demonstrar a técnica da mudança de contexto de processo, Então implementa um ambiente multitarefa para dar suporte a múltiplos processos. Extende o sistema multitarefa para suportar a criação do processo dinâmico, gerenciamento de processo e terminação de processo. Mostra como parar e continuar um processo, e extende stop/continue para operações sleep/wakeup para sicronização e explana o uso deles em um núcleo de SO. Então implementa a operações de wait para permitir processos a aguardar para terminação de processo filho. Complementarmente, também mostra como ajustar prioridades de processo para agendamento de processo baseado em prioridade. Esses levão a um simples kernel de SO para gerenciamento de processo. Em cada etapa, demonstra-se os princípios de design e técnica de implementação através de um completo simples sistema, que permite ao leitor testar e observar as operações internas de um núcleo de SO.

Capitulo 5 abrange modo usuário e chamadas de sistemas. Explana a discussão entre o espaço de endereço do kernel e modo usuário, o mecanismo de transmissão entre usuário e modo kernel e como implementar a transição. Mostra como criar processos com images de modo usuário e leva-las a retornar para executar as images no modo usuário. Baseado nisso, implementa chamadas de sistema simples, que permite processos no modo usuário entrar no kernel, executa funções de kernel e retornar para modo usuário. Então expande as chamadas de sistemas para suportar outras funções de gerenciamento de processo. Isso inclui fork, exec e a avançada técnica de vfork. Com fork e exec, mostra como se iniciar o MTX com um simples processo init, que faz fork de um rudimentar processo sh para rodar programas em modo usuário. Então faz com que o kernel do MTX tenha semelhante capacidade ao do kernel Unix em se tratando de gerenciamento de processo. Então o modelo de processo para suportar threads e demonstrar aplicações de threads por programas concorrentes.

Capitulo 6 abrange sicronização de processo. Explana os conceitos de processos concorrentes, o principio básico de sicronização de processo e a relações hierárquicas entre os vários tipos de sincronização de ferramentas. Mostra como implementar regiões criticas em ambos ambientes, uniprocessador e multiprocessador. Entãomostra como implementar os vários tipos de mecanismos de sicronização de processo. Isso inclui sleep/wakeup para ambiente de uniprocessador, spinlock e semaphores para ambiente de multiprocessador. Mostra como aplicar semaphores para ambos design de sistemas operacionais e programação concorrente. Discute deadlocks e como lidar com deadlocks. Como uma aplicação de sicronização de processo, explana os conceitos de pipes no kernel do MTX. Então discute comunicação de processo via mensagens e mostra como fazer design e implementação de mecanismos de passagem de mensagem, usando ambos protocolos, sicronomos e assicronomos. Complementarmente, também apresenta um modelo para cliente-servidor baseado em comunicação de processo.

Capitulo 7 abrange gerenciamento de memoria. Discute os objetivos e princípios de gerenciamento de memoria, que inclui segmentação, paginação, paginação-demanda, memoria virtual e regras de substituição de paginas. Discute gerenciamento de memoria em modo real do MTX em detalhes. Isso inclui gerenciamento de images de processo no kernel e gerenciamento de memoria dinâmica no espaço de usuário durante o tempo de execução.

Capitulo 8 abrange precessamento de interrupção e agendamento de processo. Explana as interrupções de hardware do PC e processamento de interrupção em detalhes. Incorpora o temporizador do PC no kernel do MTX e implementa as funções de serviço de temporizador. Discute os princípios de agendamento de processo em sistemas de compartilhamento de tempo. Explana os objetivos, politicas e algoritimos de agendamento de processos em Unix e Linux. Implementa algoritimos variados de agendamento de diferente processo no MTX e compara os efeitos deles na performance do sistema.

Capitulo 9 abrange sinais e processamento de sinal. Apresenta um unificado tratamento de interrupções e sinais, que ajudam clariar os papeis de sinais e processamento de sinal. Explana a fonte de sinais e os sinais usados apropriados. Mostra como implementar sinais e recurso de sinal no MTX e demonstra pegadores de sinal por exemplo.

Capitulo 10 abrange drivers de dispositivos. Isso inclui o display do console do PC, keyboard, printer, portas seriais, driver de floppy, disco rígido IDE e ATAPI driver para CDROM. Exceto para o display do console, que na usa interrupções, todos os drivers de dispositivos são drivers-em-interrupção e usa semaphores para sicronização. Enfatiza nos princípios de design de driver e apresenta um framework geral que é aplicado a todos interrupção-em-driver drivers, e explana em detalhes por que recursos de interrupções não deve sleep ou ser blocked. Tambem mostra como implementar processos de primeiro e segundo plano. Demonstra cada driver de dispositivo por um simples sistema, que permite ao leitor testar e observar operações de drivers de dispositivos.

Capitulo 11 abrange sistema de arquivo. Explana os vários níveis de operações de arquivo. Isso inclui operações de baixo nível em dispositivos de armazenamento, suporte de sistema de arquivo no kernel do SO, chamadas de sistemas para operações com arquivo, biblioteca de funções de I/O, comandos de usuário e scripts sh. Descreve o fluxo de controle de operações de arquivo, a partir do espaço de usuário ate baixo espaço do kernel para o nível de driver de dispositivo. Em vez de abranger somente princípios, mostra a implementação de um sistema de arquivo compatível com EXT2 (Card et al. 1995) em detalhe.

Capitulo 12 abrange gerenciamento de buffer I/O. Discute o algoritimo Unix de gerenciamento de buffer I/O (Batch 1990) e aponta as deficiências dele. Então mostra como fazer algoritimos de design de novo buffer para melhorar a performance do cache de buffer I/O. Varios materiais presentes nesse capitulo são novos e originais.

Capitulo 13 abrange interface de usuário. Apresenta listagem completa das chamadas de sistema do MTX e mostra como desenvolver programas em modo usuário. Explana os papeis e algoritimos do init e programas de login, que são essencial a inicialização do sistema. Mostra como escrever interpretador de comando sh para executar comandos de usuário. O simples sh suporta execuções de comandos singulares com redirecionamentos I/O bem como múltiplos comandos conectados por pipes. Tambem lista outros programas em modo usuário, que são usados para demonstrar a capacidade do MTX.

Capitulo 14 extende do modo-real do MTX para o modo-protegido. Explana as operações do modo protegido em detalhes e mostra como configurar o gerenciamento de hardware da memoria do PC para ambos, segmentação e paginação. Mostra como aplicar valor ao IDT para precessamento de interrupção e exceção. Então extende do modo-real do MTX para o modo-protegido 32-bit. Na ordem de inlustração de diferenças nas capacidades de segmentação e paginação, apresenta três versões diferentes do MTX no modo protegido;mtx32.1 usa segmentação, mtx32.2 usa paginação estática e pmtx usa paginação dinâmica. Tambem mostra como fazer recurso de pagina de falha e discute os princípios de demanda-paginação e memoria virtual.

Capitulo 15 abrange SMP – Multiprocessamento simétrico. Explana multiprocessador e sistemas SMP-compilant (Intel 1997), configurando IOAPIC e local APICs para SMP e inicialização de sistemas de sequencia de SMP. Abramge os princípios de design de SMP dos kernels. Então extende PMTX de cima ate SMP. Em vez de apenas usar bloqueios para proteger estruturas de dados do kernel, que tende a registringir concorrência, enfatiza-se no design e uso paralelo de algoritimos para gerenciamento de processo, gerenciamento de recurso, pipes e gerenciamento de buffer I/O. Essa nova abordagem para SMP é nova e original. Complementarmente, também mostra um sistema SMP em modo real de 16-bit.

Capitulo 16 discute outras abordagens de design de SO. Compara kernel beseado em monolítico com microkernel (Accetta et al 1986), e apresenta um sistema hibrido que inclui ambas abordagens. Demonstra a capacidade do tal sistema hibrido, implementa um servidor de sistema de arquivo iso9660 no espaço de usuário, que comunica com processos de cliente via mensagens.

A secção de Apendice contem instruções em como instalar e rodar o MTX em PC real e maquinas virtuais. Tambem inclui uma lista de todos o código fonte para os sistemas simples desse livro.

* 1. USO DESSE LIVRO COMO LIVRO DE SO

Esse livro é uma suíte bem como livro para cursos tecnicamente orientados a sistemas operacionais em um curriculum de Ciencia/Engenharia da computação que esforça-se por um equilíbrio entre a teoria e a pratica. Prerequesitos para tal do curso são muito poucos.

- Estruturas elementares de dados e programação em C.

- Um curso introdutório de SO ajuda mas não essencialmente.

Um único semestre de curso sobre SO baseado nesse livre pode incluir os tópicos seguintes.

1. Principais funções de SO, desenvolvimento de programa, execução de imagem e tempo real da pilha em uso (Capitulo 2).
2. Principio de carregamento, desenvolvimento de um carregador para que carregue o MTX a partir de um disco virtual (Capitulo 3).
3. Desenvolver um kernel de SO para gerenciamento de processo (Capitulo 4).
4. Modo de kernel e modo usuário de transições, simples chamadas de sistemas, fork e exec (Capitulo 5).
5. Sicronização de processo, semaphores, deadlock e deadlock hadling, pipes e passagem de mensagem (partes do capitulo 6).
6. Gerenciamento de memoria, princípios de paginação e memoria virtual (Capitulo 7).
7. Processamento de interrupção, temporizador, serviço de temporizador e agendamento dinâmico de processo (Capitulo 8).
8. Sinais e processamento de sinal, intalar pegadores de sina para handle exeções (Capitulo 9).
9. Drivers de dispositivos: display do console, teclado, porta serial e IDE drivers (Capitulo 10).
10. Sistema de arquivo: organização de sistema de arquivo, operações com arquivo no kernel (partes do capitulo 11).
11. Interface com usuário: init,login,sh e programas de comando de usuário (Capitulo 13).
12. Introdução as operações do modo protegido, segmentação e paginação (Capitulo 14).
13. Introdução ao SMP e tempo real SO (partes do capitulo 15).

As secções de problemas de cada capitulo contem questões designadas a rever os conceitos e princípios apresentados no capitulo. Varios dos problemas são suitaveis bem como projetos de programação para levar os estudantes a experimentar alternativas de design e implementações. Um avançado curso de SO pode abranger capitulo 14 no modo protegido em mais detalhes. Deveria também incluir capitulo 15 no SMp, com uma ênfase no design dos algoritimos paralelo que melhore a concorrência. Tal como um curso pode também incluir e expandir no capitulo 16 para que se estude diferentes abordagens para realizar design do SO.

Varios materiais desse livro tem sido usado em um nível sênior de curso sobre SO, CS460, na escola de engenharia eletrônica e ciência da computação na universidade do estado de Washington por vários anos. O objetivo do curso é fazer com que os estudantes desenvolveam um pequeno mas completo sistema operacional que trabalhe atualmente. O presente curso syllabus, notas lecture e conceitos de programação, estão disponíveis em <http://www.eecs.wsu.edu/~cs460>. Varias partes desse livro, gerenciamento de processo (capitulo 4 e 5), sicronização de processo (capitulo 6), processamento de interrupção (capitulo 8) e drivers de dispositivos (capitulo 10) estão também suitavel para cursos de sistemas embarcados.

* 1. SISTEMAS MTX PARA TESTE E EVOLUÇÃO

Sistemas simples MTX estão disponíveis para teste e evolução em <http://www.eecs.wsu.edu/~cs460/mtxhome/MTXinstallCD.iso>

MTXinstallCD.iso é uma image de CDROM carregável. Contem uma imagem virtual de disco IDE, vdisk, com 4 partições. Cada partição contem um sistema rodável de MTX. Baixe o arquivo MTXinstallCD.iso. Monte ele sob Linux e copie o arquivo vdisk para o diretório, como segue em

Mount –o loop MTXinstallCD.iso /mnt; cp /mnt/vdisk ./; umount /mnt

O caminho simples para se testar o MTX é rodar no QEMU vdisk diretamente, como segue em

qemu –hda vdisk –m 512m 0smp 8 –serial mon:stdio

Entao carregue e rode o MTX a partir de um numero de partição, onde

1. RMTX em model real de 16-bit
2. PMTX em model protegido de 32-bit usando paginação dinâmica
3. SMP\_MTX em modo protegido de 32-bit usando algoritimos paralelo para SMP
4. MTX32.1 em modo protegido de 32-bit usando segmentação.

Em todos os casos, faz-se login como root, e senha é 12345. Todos os comandos executáveis estão no diretório /bin. O leitor pode também rodar qemu no vdisk com opções adicionais, tal como –serial /dev/pts/2 –parallel /dev/pts/3 –fda FDimage

-cdrom CDimage

Complementarmente, MTXinstallCD.iso também contem todos os códigos fontes e simples sistemas desse livro, bem como instruções sobre como instalar as imagens do sistema MTX. O leitor pode consultar a secção de apêndice desse livro para detalhes.

Finalmente, o MTX tem comportamento de executável de Multitarefa. Começou no final dos anos 80 como um simples sistema multitarefa usado pelo autor para demonstrar processos, mudança de contexto e gerenciamento de processo em um kernel de SO. Ao passar do tempo, tem-se graduativamente evoluído até ser o que é a forma de hoje. MTX é um sistema derivado do Unix. Seu design é fortemente influenciado pelo Unix (Ritchie e Thompson, 1978; Bach, 1990), e empresta varias técnicas de implementação do Unix, mas não é um simples clone do Unix. Por instancia, a organização interna e a sequencia de inicialização do MTx são muito diferentes ao do Unix. Em vez de usar um mecanismo singular para sicronização de processo, MTX usa uma mixtura de ferramentas de sicronização de processo, que são escolhidas para melhorar as necessidades de suíte, e justificar seus usos. Nas áreas de gerenciamento de buffer I/O e SMP, todos algoritimos usados no MTX são novos e originais. Um vez que todos sistema derivado de Unix mereça nome de derivado de Unix, exemplo Minix, Linix, etc. Parace ser apropriado para mim também dedicar ao MTX mais um nome glamoroso, Wanix, em homenagem aos estudantes da universidade do estado de washigton, que tiveram curso de SO e motivaram me a desenvolver o sistema e escrever esse livro.