Interface Gráfica para Minix

Relatório Final

Mestrado Integrado em Engenharia Informática e Computação

Laboratório de Computadores

Turma 1 Grupo 2
Filipe Gama - ei12068
Guilherme Routar - ei12042

Faculdade de Engenharia da Universidade do Porto
Rua Roberto Frias, sn, 4200-465 Porto, Portugal

3 de Janeiro de 2015

Conteúdo

1 Instruções de utilização

1.1 Ecrã Inicial

Ao iniciar o programa desenvolvido, é apresentado um ecrã de introdução ao mesmo, que simula o *boot* de um Sistema Operativo, o Minix 3.1.8. Tal pode ser verificado na imagem seguinte.

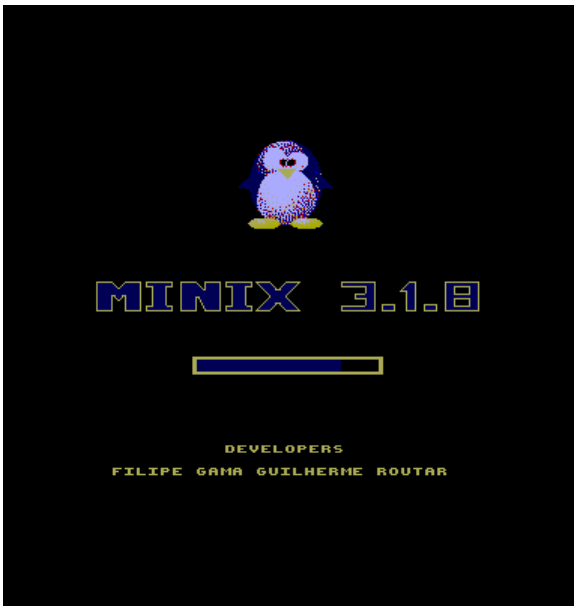

Figura 1: Animação inicial

1.2 Interface Gráfica

Posteriormente, o utilizador é redirecionado para a interface principal do programa, onde poderá navegar usando o rato ou as *arrow keys*. A simples manipulação do programa nesta fase deve-se à sua semelhança com os Sistemas Operativos atuais. Tal é especificado na secção 1.2.1 - Navegação. A interface apresenta um conjunto de pastas e ficheiros pertencentes a um determinado diretório. Tal pode ser verificado utilizando o comando *ls* no modo de texto do Minix.

As 2 primeiras pastas (”.”e ”..”) permitem retornar à pasta origem e à pasta anterior, respetivamente.

1.3 Navegação

Rato

Ao clicar (1 clique) sobre um ficheiro ou pasta pode ser verificada a seleção do(a) mesmo(a), através de um simples *highlight*.

O duplo clique permite abrir a pasta em questão, revelando o seu conteúdo.

Arrow Keys

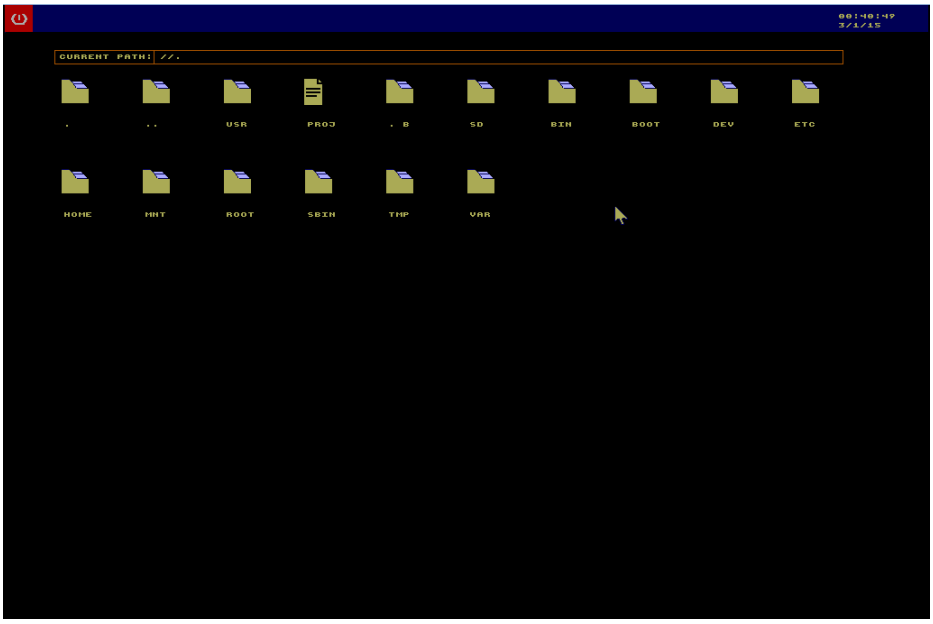

Figura 2: Ecrã inicial

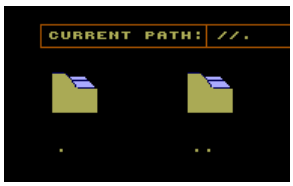

Figura 3: Pastas de voltar atrás ou à origem

A navegação utilizando *arrow keys* é igualmente intuitiva. No caso de nada estar selecionado (*highlighted*), é selecionado o ficheiro ou pasta que se apresentar primeiramente na interface (mais à esquerda) quando premida qualquer uma das 4 teclas. Caso contrário, a navegação para as 4 diferentes direções é efetuada premindo a tecla respetiva. A abertura das pastas é efetuada através da tecla *enter* e o *backspace* permite ao utilizador regressar à pasta origem (a que é apresentada inicialmente). A tecla *delete* permite apagar um determinado ficheiro ou pasta, aparecendo uma mensagem *pop up* de confirmação.

Por fim, o utilizador pode terminar o programa clicando duplamente na tecla *ESC* ou clicando no símbolo apresentado no canto superior esquerdo do ecrã. De seguida, aparece uma mensagem *pop up*, que questiona o utilizador relativamente à sua intenção.d

Como funcionalidade adicional, foi implementado um mini menu acionado pelo *right click* do rato com as opções *cut*, *copy* e *paste*. Todavia, não houve possibilidade de implementar ações associadas.

Figura 4: Pasta selecionada

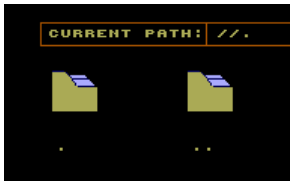

Figura 5: Voltar à origem e à pasta anterior, respectivamente

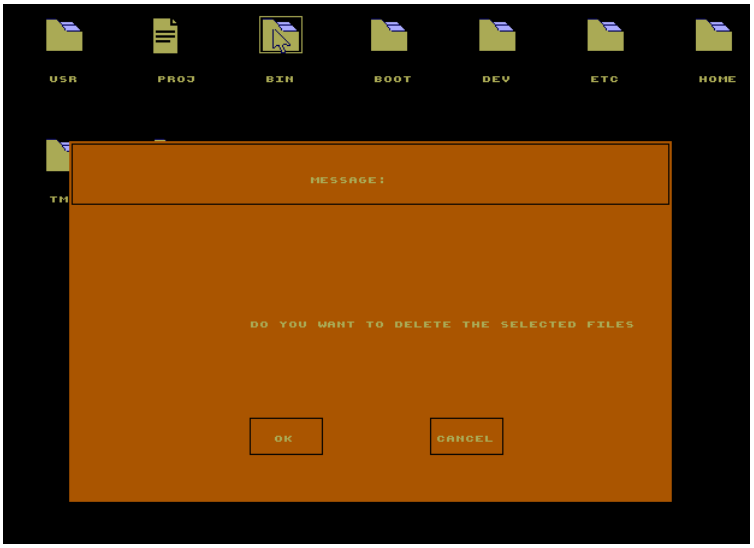

Figura 6: Pop-up confirmação delete

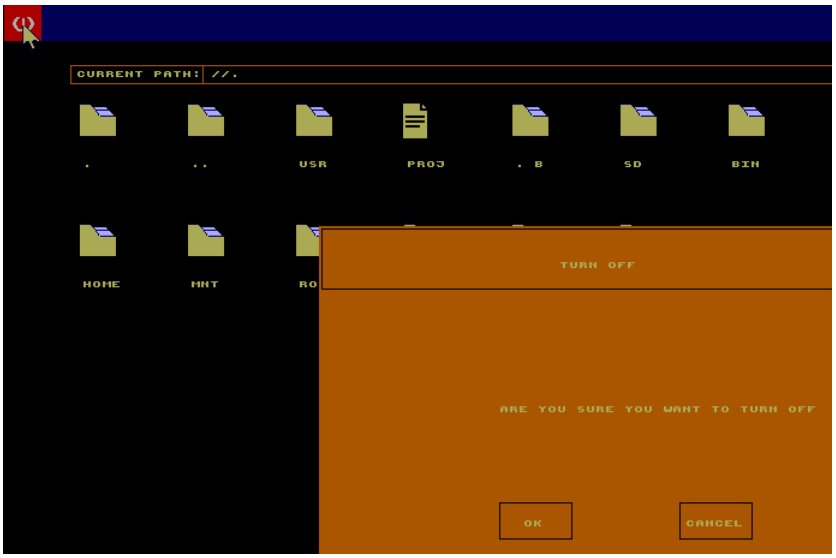

Figura 7: Botão de shutdown

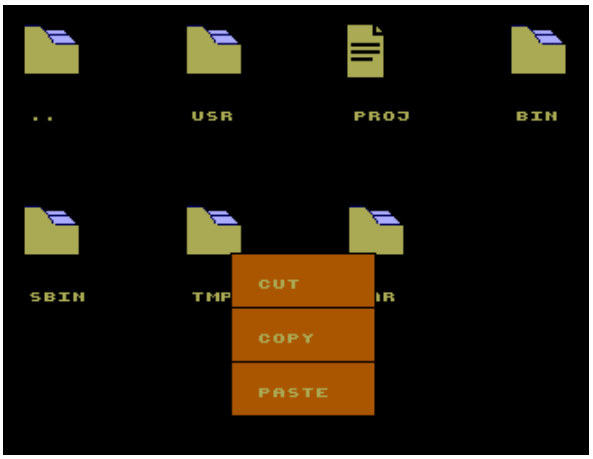

Figura 8: Menu de clique direito do rato

2 Funcionalidades

Implementadas:

- "Display"gráfico dos directórios e pastas (reais) do minix, bem como de o "path"actual em que o utilizador se encontra
- A navegação entre directórios ficou implementada, tanto utilizando o cursor (duplo clique para abrir pastas) como o teclado (enter para abrir e setas para navegar).
- Funcionalidade de apagar um directorio (inteiro e recursivamente) ou um ficheiro, através do teclado (botão delete)
- Possibilidade de renomear uma pasta.
- Botão de shutdown que pode ser utilizado com o rato, sendo mostrado um pop-up de confirmação. Também é possível sair do programa através de duplo clique do botão "escape".

Por implementar:

Das funcionalidades que tentamos implementar, ficou por fazer copy/paste de pastas ou ficheiros, visto não ser possível realizar aquilo que esperávamos - fork e exec - devido às características do minix. Apesar disso, ficou implementado o que seria a interface gráfica desta funcionalidade, usando o botão direito do rato em cima de uma pasta/ficheiro para aceder a um menu com a tal funcionalidade.

Apesar de não nos termos comprometido a tal na especificação do projecto, tentamos também implementar a visualização de ficheiros graficamente, no entanto não conseguimos realizar na totalidade esta "feature"tendo ficado só parcialmente funcional.

Dispositivos utilizados e respectivas funções:

Dispositivo	Funcionalidade	Int.
Timer	Controlo do frame rate e também para duplos cliques (tanto do teclado como do rato)	Y
Placa gráfica	Menus e toda a interface gráfica	N
Teclado	Para executar operações como delete, ou fazer a navegação entre pastas com as "arrow keys"	Y
Rato	Usado como cursor, para poder abrir pastas, selecionar ou aceder a menus	Y
RTC	Para mostrar a hora e data do sistema operativo	N

3 Estrutura e organização do código

Nota: os pesos relativos aos ficheiros dos labs foram atribuídos tendo em conta que estes já se encontravam implementados, ou seja apenas foi contada a dificuldade da sua integração no projecto. Para os mesmos também não apresentamos a contribuição por já terem sido auto-avaliados anteriormente

3.1 main.c

Neste ficheiro são apenas chamadas as funções necessárias ao início do programa.

Peso do módulo no projecto: desprezável

3.2 logic.c

É neste ficheiro que se encontra toda a "lógica"associada ao programa, nomeadamente todo o tipo de funções relacionadas com directórios, bem como relativas ao tratamento dos inputs recebidos pelos dispositivos, isto é, cálculo das consequências das interrupções fazendo por exemplo verificações de colisões.

Peso do módulo no projecto: 20%
Contribuição dos membros do grupo:
Filipe Gama - 80%
Guilherme Routar - 20%

3.3 state.c

É neste ficheiro que se encontra o loop que lida com as interrupções, e faz update ao estado do programa atualizando estruturas como do rtc e do rato. É aqui também onde se encontram guardados estados relativos a menus abertos, popups, etc.

Peso do módulo no projecto: 20%
Contribuição dos membros do grupo:
Filipe Gama - 80%
Guilherme Routar - 20%

3.4 interface.c

Todas as funções relativas ao output no ecrã se encontram neste ficheiro. A saber, desenho de menus, do relógio, do cursor, incluindo também a lógica associada ao desenho de strings graficamente,e a lógica associada ao desenho de pastas e ficheiros (i.e. verificar quando desenha com imagem de pasta ou de ficheiro, quando estão selecionados ou não, etc).

Peso do módulo no projecto: 20%
Contribuição dos membros do grupo:
Filipe Gama - 65%
Guilherme Routar - 35%

3.5 pixmap.c

Ficheiro fornecido onde foram adicionados alguns xpm necessários ao projecto.

Peso do módulo no projecto: 10%
Contribuição dos membros do grupo:
Filipe Gama - 20%
Guilherme Routar - 80%

3.6 keyboard.c

Ficheiro onde se encontram as funções associadas ao keyboard, relativas ao subscribe e ao handling das interrupções. O handler foi alterado para retornar valores em vez de os imprimir

Peso do módulo no projecto: 10%
Contribuição dos membros do grupo:
Filipe Gama - 50%
Guilherme Routar - 50%

3.7 mouse.c

Ficheiro onde se encontram as funções associadas ao mouse, relativas ao subscribe e ao handling das interrupções. Foi implementada uma estrutura de dados (mouse_state) para guardar as coordenadas do cursor e o estado dos botões durante a execução do programa. É actualizado cada vez que um pacote é inteiro é recebido.

Peso do módulo no projecto: 10%
Contribuição dos membros do grupo:
Filipe Gama - 80%
Guilherme Routar - 20%

3.8 rtc.c

Ficheiro onde se encontram as funções associadas ao keyboard, relativas ao subscribe e ao handling das interrupções. Foi implementada uma estrutura de dados (rtc_state) para guardar as variáveis relativas às horas, minutos, segundos, dia, mês e ano. É actualizada a cada interrupt (do timer), ou seja 60 vezes por segundo.

Peso do módulo no projecto: 10%

3.9 read_xpm.c

Ficheiro fornecido e apenas adicionada uma função que desenha um xpm transparente, isto é, em vez de ser desenhado um fundo preto não ser desenhado fundo simplesmente. (Usado para o cursor, entre outros)

Peso do módulo no projecto: 0%

3.10 vbe.c

Ficheiro utilizado nos labs para permitir o init do modo de vídeo, não tendo sido alterado.

Peso do módulo no projecto: 0%

3.11 video_gr.c

Ficheiro utilizado nos labs, tendo sido adicionadas uma função também relativamente ao desenho de xpm sem fundo (background) e outra para desenhar retangulos sólidos, isto é, com fundo. É onde se encontram definidas praticamente todas as funções utilizadas no ficheiro de interface, num nível mais baixo, desenho de retângulos e de xpm's, que já tinha sido criadas para o lab5.

Peso do módulo no projecto: 0%

3.12 timer.c

Ficheiro onde se encontram as funções associadas ao timer, relativas ao subscribe e ao handling das interrupções. Não houve quaisquer alterações a esta classe.

Peso do módulo no projecto: 0%

4 Grafo de chamadas de funções

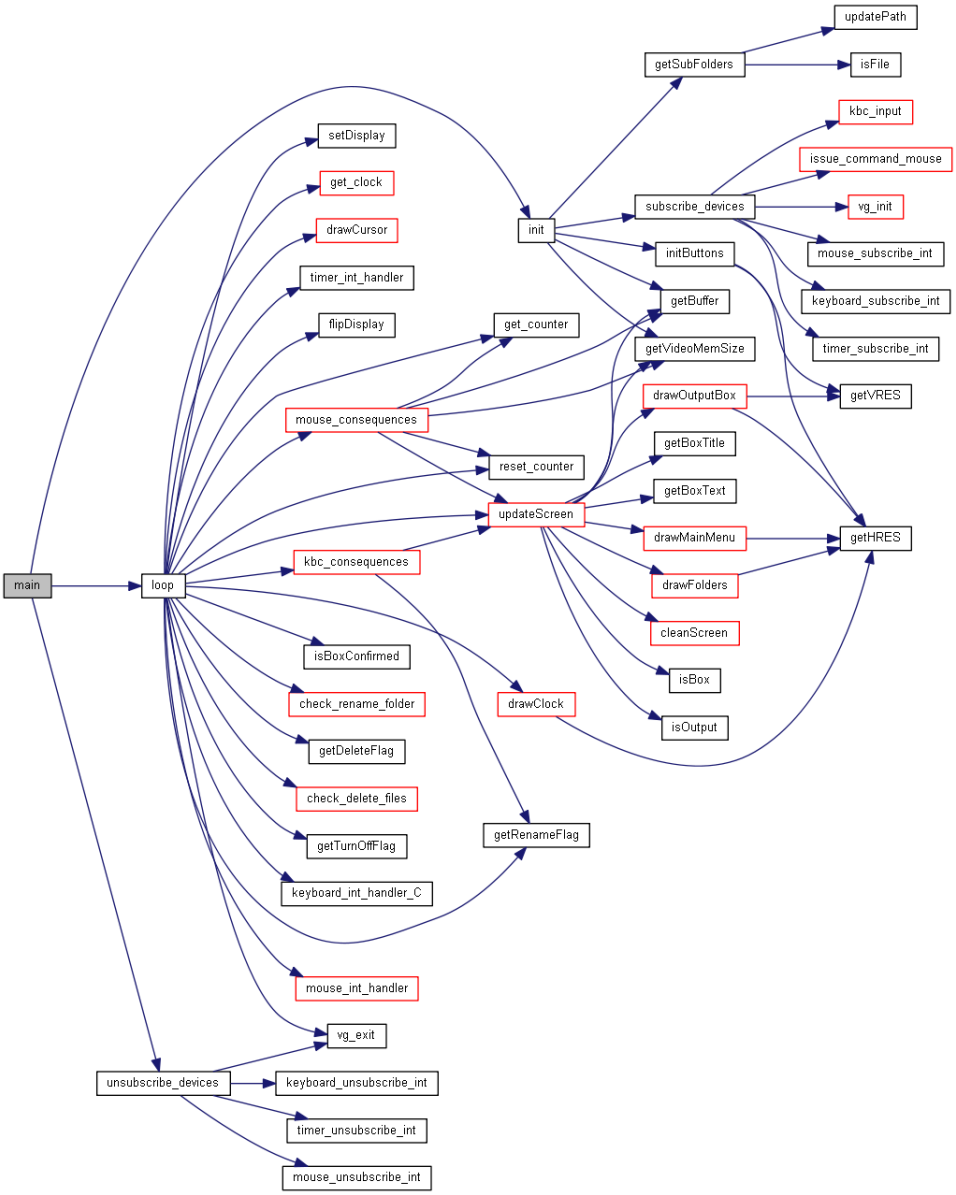

Figura 9: Diagrama de chamada de funções

5 Detalhes de implementação

Para a realização do nosso projecto foi necessário aprender alguns conceitos novos, nomeadamente funções de C para abrir, apagar, renomear directórios em unix.

6 Avaliação

Auto-avaliação relativa ao código:

Filipe Gama - 70%
Guilherme Routar - 30%