Write Wise

Projeto de Laboratório de Computadores

Autores:

Henrique Caridade (up202108817)

Igor Cherstnev (up202105300)

Luís Diogo (up201806340)

Pedro Pinto (up202108826)

28 de maio de 2023

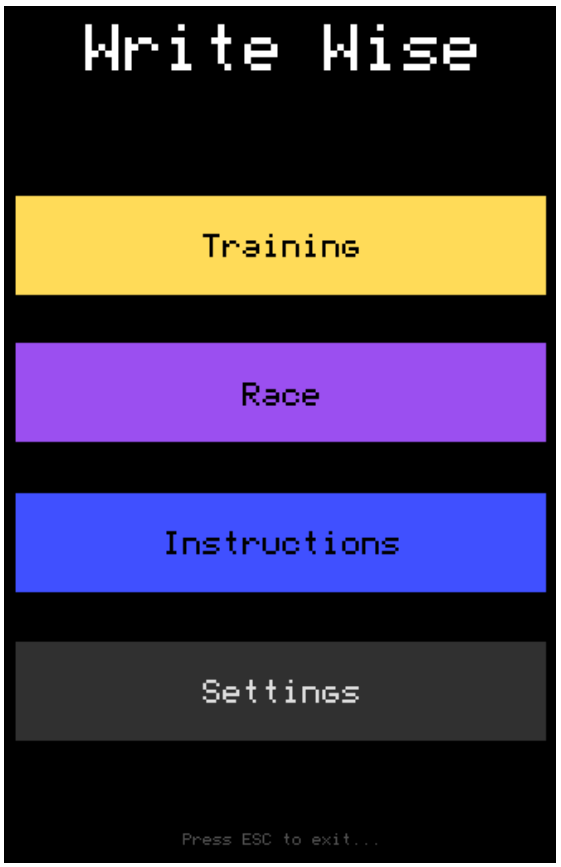
Índice

1. Instruções de utilização do programa

1.1. Ecrã inicial

Ao iniciar o programa, é apresentado um ecrã de introdução com o nome do projeto, onde o utilizador deve premir qualquer tecla do teclado para avançar para o ecrã seguinte.

1.2. Menu



Training: Inicia o modo de treino, onde o utilizador pode treinar a sua digitação no teclado.

Race: Inicia o modo de corrida, onde o utilizador pode competir com outro utilizador, ligado noutro computador.

Instructions: O utilizador pode consultar as instruções de utilização.

Settings: Abre o menu das definições, onde o utilizador pode mudar o tema ou a resolução do ecrã.

1.3. Definições

Neste ecrã, o utilizador pode alterar a resolução pretendida, assim como o tema.

1.4. Instruções

Instruções de utilização do modo de treino e do modo de corrida.

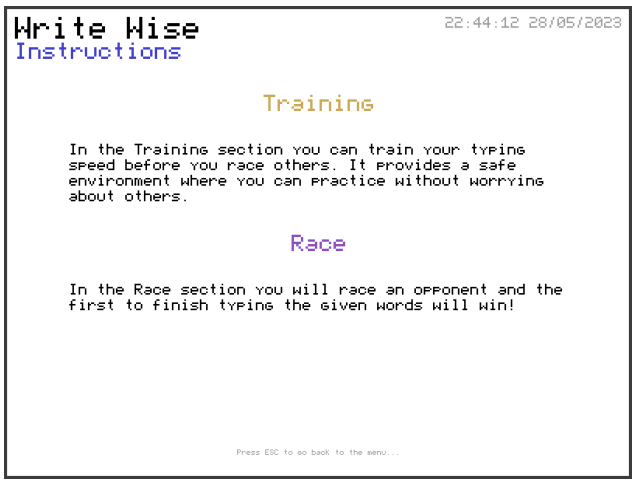
1.5. Modo de treino

Neste modo, o utilizador pode treinar a sua rapidez a digitar texto no teclado e analisar a sua eficiência, o seu número de palavras escritas por minuto e consultar o seu progresso a nível de caracteres já escritos.

1.6. Modo de corrida

Neste modo, o utilizador pode competir com outro utilizador, ligado noutro computador, e realizar uma corrida em que ganha quem digitar mais rapidamente o texto indicado no ecrã, podendo, na mesma, consultar a sua eficiência, o seu número de palavras por minuto e o progresso realizado. Para a corrida começar, ambos os utilizadores têm que estar prontos.

1.7. Tema claro



Páginas da aplicação no tema claro.

2. Project Status

2.1. Dispositivos I/O utilizados

Dispositivo	Utilização	Int
Timer	Controlar a frame rate e contar o tempo dos testes.	Sim
Keyboard	Escrita de texto.	Sim
Mouse	Navegar as opções do menu.	Sim
Graphics Card	Mostrar a aplicação no ecrã e suportar múltiplos modos gráficos.	Não
RTC	Dizer a data atual no canto superior direito.	Não
Serial Port	Suportar as corridas entre dois jogadores.	Sim

2.2. Timer

O Timer é usado no nosso programa, principalmente, para controlar a frame rate(taxa de atualização) do ecrã, mas também é usado para atualizar o RTC todos os segundos, para contar o tempo de timeout de um "Acknowledgement" da Serial Port e também para contar o tempo de um teste usado depois (e durante o teste) para calcular as wpm (palavras por minuto).

A função principal do Timer é timer_get_elapsed_count() que retorna o contador interno do timer (contador é inicializado a 0 no início do programa e incrementa a cada interrupt gerado pelo Timer).

2.3. Keyboard

O Keyboard é uma das partes fundamentais do nosso programa pois é o Keyboard que é utilizado para a digitação de texto nos testes, mas também é usado para navegação (com a tecla ESC) e no ecrã Inicial para ir para o menu.

A função principal do teclado (após o Interrupt Handler guardar o scancode) é o keyboardScancodeHandler() que trata da informação recebida (scancode).

2.4. Mouse

O Mouse é usado no nosso programa para a navegação dos menus e para a escolha de opções no ecrã das Definições (tudo isto com botões).

A função principal do mouse (após o Interrupt Handler dizer que o Mouse Packet está pronto) é o mouseUpdate() (definida no ficheiro ui.c) que é que usa a informação do Mouse Packet para atualizar a posição do cursor, detectar "cliques" e chama as callbacks dos botões definidos.

2.5. Graphics Card

A Graphics Card é usada para mostrar a aplicação no ecrã.

A nossa aplicação usa double buffering e também um buffer que guarda a parte estática do ecrã por isso todos os frames a parte estática do ecrã é desenhada para o buffer, depois os elementos não estáticos são desenhados para o buffer e só depois é copiado o buffer para a memória da Graphics Card.

A nossa fonte para desenhar texto para o ecrã é nada mais do que uma array 2D de booleanos para cada caractere, quando é para desenhar basta percorrer a array e desenhar os pixels no sítio certo.

A nossa aplicação suporta 4 modos gráficos:

Mode	Screen Resolution	Color Model	Bits per pixel (R:G:B)
0x110	640x480	Direct color	15((1:)5:5:5)
0x115	800x600	Direct color	24 (8:8:8)
0x11A	1280x1024	Direct color	16 (5:6:5)
0x14C	1152x864	Direct color	32 ((8:)8:8:8)

As funções mais importantes onde a Graphics Card é usada são todas as funções que começam com draw...(), como por exemplo, drawTextColor(...) ou drawRectColor(...).

2.6. RTC

O RTC (Real Time Clock) é usado para mostrar o tempo no canto superior direito para o utilizador saber as horas mesmo dentro da aplicação.

A função principal onde o RTC é usado é na drawRealTime() que usa a informação do RTC para mostrar o tempo no ecrã.

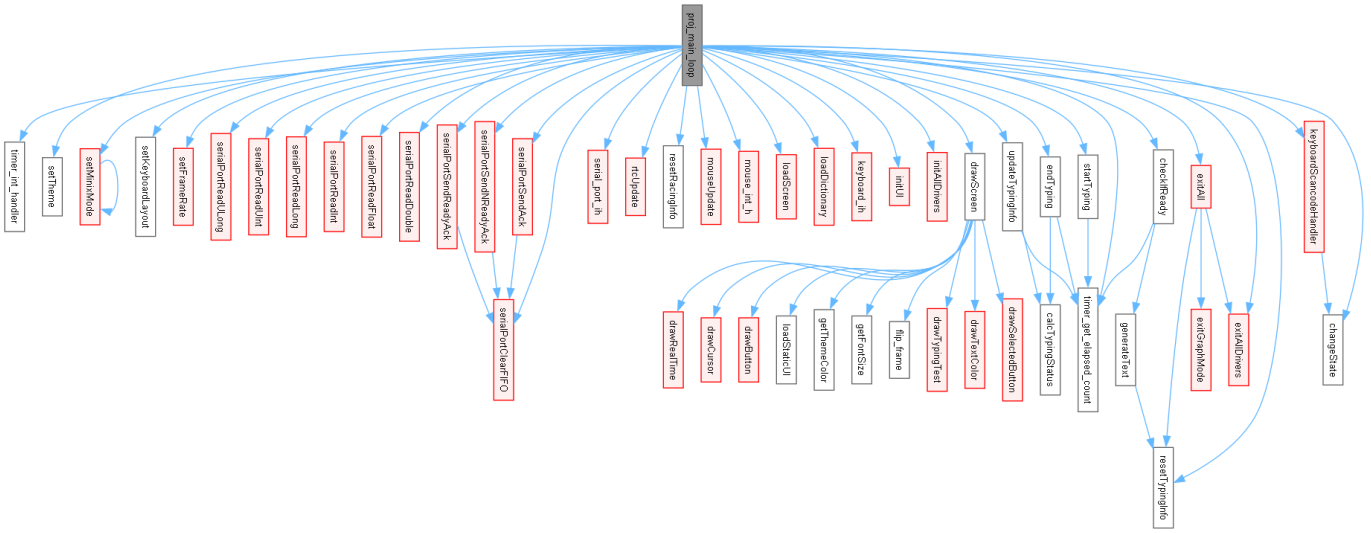
2.7. Serial Port

A Serial Port é usada crucialmente para o ecrã de corrida, é por ela que é feita a troca de informações como a “seed” para a geração de texto de forma a que fiquem com um texto igual, para verificar se ambos os jogadores estão prontos para começar a corrida e também para a sincronização do início da corrida. Tudo isto é feito usando FIFOs e “interrupts”.

As funções mais importantes onde a Serial Port é usada são todas as funções que começam com serialPort...(), como por exemplo, serialPortSendAck() ou serialPortSendByte(...).

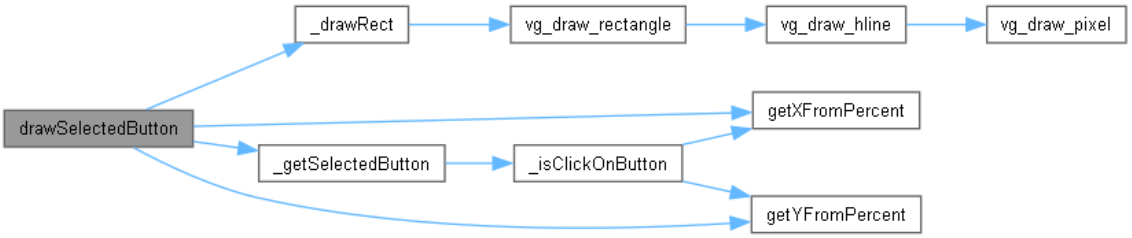
3. Code Organization/Structure

3.1. proj.c (26,6%)



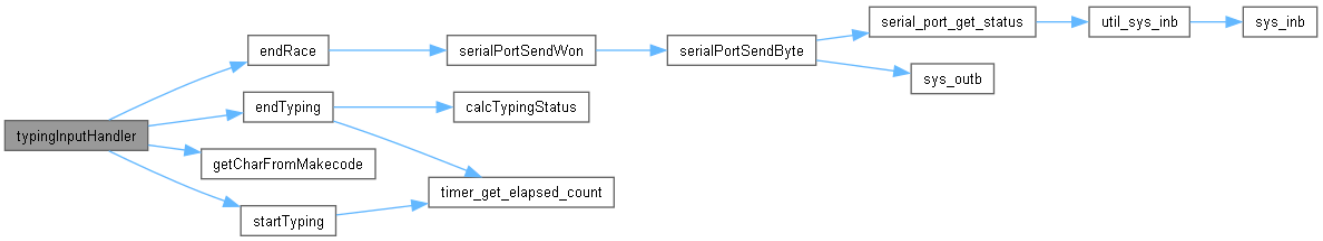
Este módulo é quem junta todos os outros módulos para fazer a aplicação funcionar, tendo como principal funcionalidade tratar dos interrupts gerados pelos drivers.

3.2. ui.c (9,3%)



Este módulo contém principalmente funções para inicializar a UI, gerir os botões, desenhar os botões e o cursor no ecrã e receber o input do rato.

3.3. typing.c (9,6%)

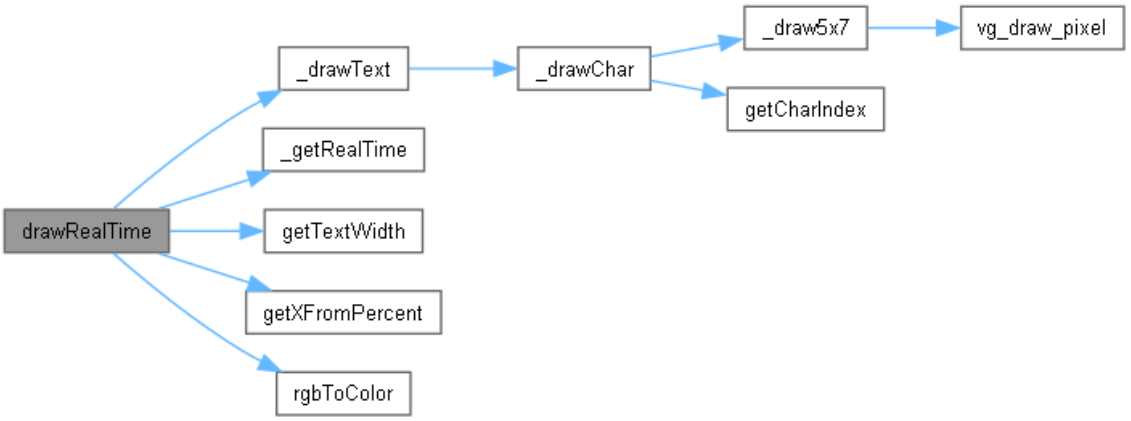


Este módulo contém as funções para escrever texto, recebendo o input do teclado, calcular as estatísticas e dar display do texto a escrever no ecrã.

3.4. data.c (1,8%)

Este módulo trata de buscar as palavras depois usadas nos testes.

3.5. lib.c (11,8% + 40,9%)



Este módulo contém funções para inicializar todos os drivers, definir o frame rate, alternar entre o text mode e o graphic mode, definir o tema e calcular a posição do rato no ecrã, através das suas coordenadas x e y, dependendo da resolução.

3.5.1. text.c (2,5%)

Este submódulo trata de guardar e gerir a fonte do nosso programa.

3.5.2. scancode.c (2,3%)

Este submódulo tem funções para transformar scancodes de um layout de teclado para caracteres que se pode escrever para o ecrã com o submódulo anterior.

3.5.3. drivers (36,1%)

Este submódulo contém todos os ficheiros que falam com os drivers diretamente.

3.5.3.1. utils.c (1,2%)

Este submódulo contém funções auxiliares usadas pelos outros submódulos dos drivers.

3.5.3.2. timer.c (3,5%)

Este submódulo contém as funções que falam com o dispositivo timer.

3.5.3.3. serialPort.c (12,8%)

Este submódulo contém as funções relacionadas com a comunicação pela Serial Port.

3.5.3.4. rtc.c (3,8%)

Este submódulo contém as funções relacionadas ao dispositivo RTC (Real Time Clock).

3.5.3.5. mouse.c (3,3%)

Este submódulo contém as funções relacionadas ao dispositivo Mouse.

3.5.3.6. keyboard.c (3,5%)

Este submódulo contém as funções relacionadas ao dispositivo Keyboard.

3.5.3.7. KBC.c (3,1%)

Este submódulo contém as funções usadas para a comunicação dos dispositivos Mouse e Keyboard.

3.5.3.8. graphics.c (4,9%)

Este submódulo contém as funções relacionadas ao dispositivo Graphics Card.

4. Implementation Details

Neste projeto decidimos em primeiro lugar fazer uma “livraria” que trouxesse o trabalho de “baixo nível” de comunicar com os drivers para um nível mais alto e mais fácil de trabalhar. Essa “livraria” é a pasta lib. Nessa pasta contém, por exemplo, funções de desenhar para o ecrã onde não é preciso preocupar com o modo em que está a VBE é só colocar um valor em color hex normal (0xRRGGBB) e desenha sempre bem e também não é preciso preocupar com que as dimensões do ecrã pois funciona por percentagem do ecrã (valores de 0 a 1).

Depois dessa “livraria” estar feita fizemos outras pastas com funcionalidades que eram necessárias: a UI é quem trata do inputs do Mouse e botões; a Data que trata de buscar as palavras para os testes de um ficheiro; e Typing que trata de guardar as informações dos testes e atualizá-las dependo dos scancodes recebidos. Esta modularidade de código facilitou muito o desenvolvimento do resto do programa.

O ficheiro proj.c é quem tem a informação da aplicação, por exemplo o estado da app, as cores dos temas, etc. usando essa informação para desenhar o ecrã certo e receber tratar os inputs de maneira correta numa espécie de Máquina de estados.

A Serial Port funciona com um “protocolo” criado por nós. Quem quiser mandar informação tem de mandar um byte de informação (bytes para além de 0xF0), que avisa qual é a informação que vai ser enviada, por exemplo o byte INFO_ULONG (0xF2) que avisa que vai ser enviado um unsigned long ou seja 8 bytes, ou às vezes só esse byte é a informação como por exemplo o byte INFO_READY (0xF8) que avisa que o outro utilizador está pronto para começar a corrida. Para garantir que se recebeu existem bytes de “Acknowledgement” INFO_ACK (0xF0), INFO_READY_ACK (0xFA) e INFO_NREADY_ACK(0xFB).

5. Conclusion

Este projeto foi concluído com sucesso, sem problemas significativos ou atrasos. Todas as funcionalidades foram implementadas conforme planeado e utilizando todos os dispositivos aprendidos ao longo do semestre. Algumas das principais conquistas do projeto incluem: implementação do modo de jogo de corrida multijogador (Race), para além do modo de jogo principal (Training); interface gráfica intuitiva e agradável, proporcionando uma experiência visualmente atrativa.

Durante o desenvolvimento do projeto, aprendemos algumas lições importantes: organização e modularidade do código são fundamentais para facilitar o desenvolvimento e a manutenção. Documentação adequada é essencial para compreender o código e facilitar a colaboração entre desenvolvedores. A abordagem iterativa e incremental no desenvolvimento de software permite ajustes e melhorias contínuas.

Em resumo, o projeto foi concluído atendendo aos requisitos e entregando uma aplicação de elevada qualidade. As lições aprendidas e as dificuldades ultrapassadas, fornecem uma base sólida para projetos futuros.