FEUP

EIC0020 – Laboratório de Computadores

2009/2010 – 2S

Light Cycles

31.05.2010

Turma 3, Grupo 6

Autores:

Carlos Tiago Rocha Babo, 080509118, ei08118@fe.up.pt

Felipe de Souza Schmitt, 080509160, ei08160@fe.up.pt

1. Resumo

O objectivo principal deste projecto final passou por entender o funcionamento dos principais periféricos do computador e tentar programa-los, dentro dos seus próprios limites, para criar um jogo. Como consequência, a familiarização com as linguagens de programação Assembly e C, a utilização da makefile e de um repositório de código (SVN).

Em relação aos periféricos, foi usada a placa gráfica, o teclado, o RTC, o porto de série, o altifalante e o timer. A placa gráfica foi programada no seu modo gráfico para apresentar toda a informação relativa à interface do jogo, controlada pelas interrupções do teclado. O RTC permite contabilizar o tempo de jogo, tanto para mover os jogadores, como para calcular a pontuação final. O altifalante, aliado ao timer, permite tocar a música de introdução e enviar um som de explosão quando o jogador perde. Por fim, o porto de série possibilita o modo *multiplayer* a dois jogadores.

Para o desenvolvimento deste projecto, foi utilizado o compilador DJGPP[1], o assemblador NASM[2], a makefile[3], o SVN[4] para a gestão das versões do projecto, o Doxygen[5] para gerar a documentação e o GDB[6] para *debug* O projecto foi desenvolvido totalmente em Microsoft Windows 98[6].

2. Descrição do Programa

Light Cycles é um jogo arcade, baseado no famosos Tron[7]. O objectivo passa por guiar a nossa Light Cycle na arena, enquanto tentamos evitar as paredes e os rastos de luz deixados pelo jogador adversário. Deste modo, o jogador deve movimentar-se de forma rápida para forçar o oponente a bater contra as paredes ou contra si mesmo.

Em cada jogada, cada um dispõe de três vidas (indicadas do lado direito da janela de jogo) e ganha aquele que conseguir manter-se por mais tempo sem quebrar as regras. O jogador pode seguir quatro direcções possíveis: direita, esquerda, cima ou baixo. Para isso, o jogador deve usar as teclas padrão (D, A, W, S) ou configura-las ao seu gosto menu principal.

Existem dois modos diferentes de se jogar: *multiplayer* com porto de série, ou contra o CPU. O primeiro permite dois jogadores, um em cada computador, defrontarem-se na arena. Já o segundo, em modo singleplayer, coloca na arena o jogador e o CPU (programado com três algoritmos de inteligência artificial).

No final de cada jogo, é verificado se a pontuação do vencedor é suficiente para entrar no quadro de melhores pontuações. Em caso afirmativo, é pedido para inserir três letras que o identifiquem. O quadro de melhores pontuações pode ser consultado através do menu principal.

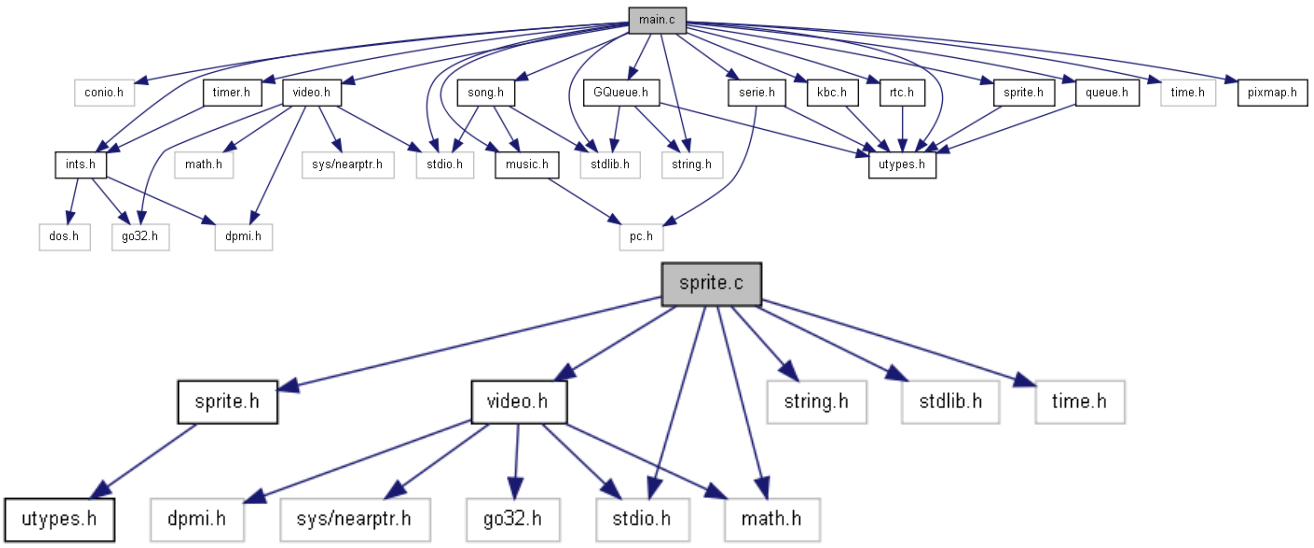
3. Implementação

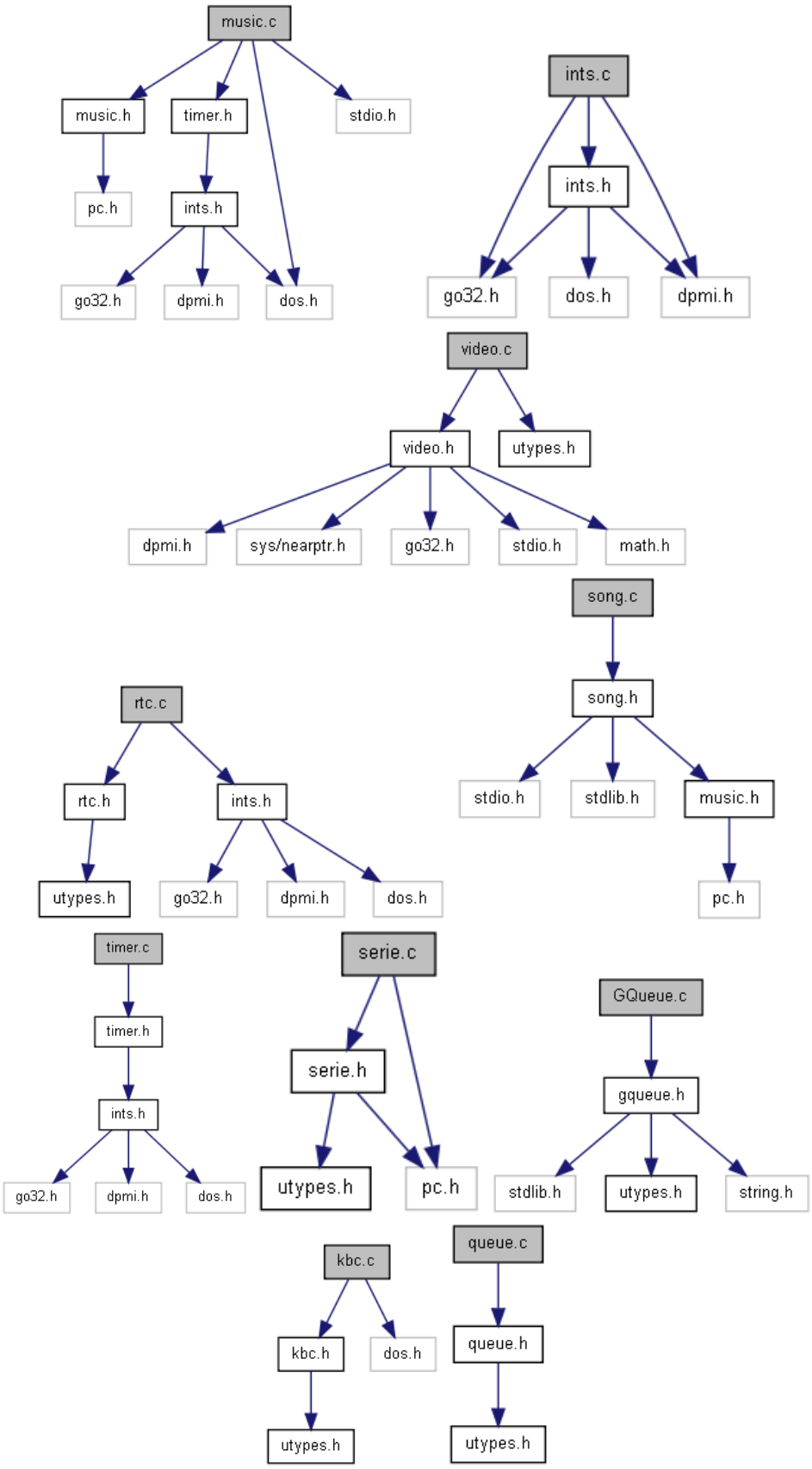
3.1 Arquitectura do Programa

Módulos e funcionalidades:

- Main: responsável pela instalação/desinstalação das interrupções e de todo o mecanismo de jogo;
- Sprite: funções relativas à apresentação de sprites no ecrã;
- Queue: contém funções que permitem simular o conceito de pilha em C;
- GQueue: semelhante à queue, mas genérica (pode ser utilizada com qualquer tipo de dados);
- Music: funções que permitem controlar o altifalante do computador;
- Timer: funções que activam/desactivam o TIMER2;
- Video: funções necessárias para programar a placa gráfica em modo gráfico;
- KBC: funções necessárias para programar o teclado e obter o scancode de cada tecla;
- RTC: funções para controlar o Real Time Clock do computador;
- Serie: funções que programam e permitem a comunicação com porto de série;
- Song: contém as funções que possibilitam a criação de estruturas do tipo Song.

Diagrama de dependências:




music.c
music.h
timer.h
stdio.h
pc.h
ints.h
go32.h
dpmi.h
dos.h
ints.c
ints.h
go32.h
dos.h
dpmi.h
video.c
video.h
utypes.h
dpmi.h
sys/nearptr.h
go32.h
stdio.h
math.h
song.c
song.h
stdio.h
stdlib.h
music.h
pc.h
rtc.c
rtc.h
ints.h
utypes.h
go32.h
dpmi.h
dos.h
timer.c
timer.h
ints.h
go32.h
dpmi.h
dos.h
serie.c
serie.h
utypes.h
pc.h
GQueue.c
gqueue.h
stdlib.h
utypes.h
string.h
kbc.c
kbc.h
dos.h
utypes.h
queue.c
queue.h
utypes.h

3.2 Funcionalidades

Funcionalidades implementadas:

- Escrita de texto em modo gráfico;
- Ver melhores pontuações;
- Ler e escrever de ficheiro as pontuações;
- Alterar as teclas de jogo;
- Controlar a Light Cycle com o teclado;
- Apresentar as vidas e tempo de jogo actual;
- Música quando se inicia o jogo e quando o jogador perde;
- Três algoritmos de Inteligência artificial;
- Modo para dois jogadores com porto de série;
- Menu de jogo.

Funcionalidades não especificadas mas implementadas:

- Utilização de sprites no título e quando o jogador bate contra si ou contra a parede;

3.3 Detalhes *Relevantes* da Implementação

Como seria de esperar, os módulos realizados nas aulas foram utilizados na concretização do projecto. Apesar disso, decidimos acrescentar mais dois para simplificar o projecto. Assim, criamos o módulo Song, que permite ler de um ficheiro uma música e criar um objecto com toda a informação necessária para a tocar e o módulo Sprite, para desenhar sprites no ecrã.

O movimento do jogador é controlado através do RTC, simulando a função delay e a pontuação é também calculada recorrendo ao tempo de jogo lido no RTC e às vidas actuais do vencedor.

No modo multiplayer, para garantir que o jogo começa sincronizado (sem considerar a natural lag entre os dois portos de série), o jogador que serve de "servidor" tem de esperar pelo segundo, que deve escolher a opção de se juntar a um jogo existente, para começar.

Em relação ao modo single-player, as ideias para o desenvolvimento da Inteligência Artificial foram concebidas ao longo do trabalho realizado nas aulas, onde foram pensados três algoritmos diferentes, de complexidade simples, possíveis de implementar no tempo disponível. Assim, os algoritmos concebidos foram os seguintes:

- "Around + Random": a ideia principal deste algoritmo é verificar em todas as jogadas se o raio de luz irá bater em algum objecto na jogada seguinte e

caso isto se verifique, mudar a direcção consoante uma ordem de prioridade: Cima, Baixo, Direita, Esquerda. Para além disso, a cada 300 passos, é escolhido um novo movimento aleatório para que não se entre num ciclo rotineiro.

- "Stay Away": este algoritmo verifica em todas as jogadas qual das direcções (Cima, Baixo, Direita, Esquerda) possibilita um maior caminho antes de bater em alguma parede ou num rasto de luz, modificando a direcção sempre que a actual já não seja a ideal.
- "Wall Hugger": como o próprio nome indica, este algoritmo é o mais básico de todos, sendo idêntico ao primeiro (Around + Random), mas sem a componente aleatória. O raio de luz percorre a direcção inicial e quando verifica que na próxima jogada irá bater num objecto, escolhe uma direcção possível, segundo a ordem de prioridade pré-definida. Assim, o raio de luz percorre sempre os caminhos junto à parede ou ao raio de luz do outro jogador.

A pontuação é calculada com base no tempo de jogo e no número de vidas do vencedor.

3.4 Instruções de compilação e utilização

Para compilar o programa executar o comando "make". Se for necessário limpar os ficheiros anteriores com "make clean". Existe ainda a opção de fazer rebuild ao projecto através do comando "make rebuild". Para correr o jogo basta chamar o executável "lightcycles", não existindo opções suportadas na linha de comandos. É necessária a existência de um ficheiro "pont.txt" para ler e guardar as pontuações (caso não exista, o sistema cria-o).

Funcionamento do programa: é apresentado um menu inicial com quatro opções: Jogar, Ver melhores pontuações, Configurar Teclas e Sair. Para escolher cada uma das opções, basta premir a tecla do número respectivo. A opção Jogar leva-nos até outro menu, onde se deve escolher: criar um jogo multiplayer, juntar a um jogo existente, ou jogar contra o CPU. Dentro deste último, pode-se escolher até três algoritmos diferentes de comportamento do adversário. A opção Melhores Pontuações mostra as melhores pontuações do ficheiro de texto "pont.txt". A opção Configurar Teclas permite definir as teclas de jogo. Por fim, a opção "Sair" fecha a aplicação.

As teclas pré-definidas para jogar são:

- A – esquerda;
- S – baixo;
- D – direita;

- W – cima;
- ESC – sair.

3.5 Conclusões

No geral, as ideias idealizadas inicialmente foram concretizadas com sucesso. O processo inicial do projecto passou por testar os módulos desenvolvidos nas aulas práticas, para tentar diminuir problemas relacionados com os mesmos e focarmo-nos na sua utilização. Apesar disso, tivemos alguns problemas com a instalação/desinstalação das interrupções, facilmente resolvidos com a ajuda do docente, assistente e leitura dos slides das aulas teóricas.

A utilização do SVN ajudou bastante no desenvolvimento, pois quando era necessário testar em dois computadores (porto de série) não era necessário passar o código para dispositivos físicos ou para *hosts* online (feupload, por exemplo).

O resultado final obtido aproxima-se bastante do conceito inicial, até porque, dada a especificação original, não existia grande margem para mudanças. Contudo, a inteligência artificial poderia ter sido alvo de um estudo mais profundo, e em caso ideal, os vários *bots* poderiam ter dificuldades diferentes, adicionando o conceito de níveis de progressão.

Em jeito de conclusão, o projecto enriqueceu o conhecimento dos periféricos do computador e da linguagem C e Assembly, de uma forma divertida e auto-didáctica.

3.6 Referências

[1] DJGPP - http://www.delorie.com/djgpp/

[2] NASM - http://www.nasm.us/

[3] Makefile - http://www.gnu.org/software/make/

[4] SVN - http://subversion.tigris.org/

[5] Doxygen - http://www.stack.nl/~dimitri/doxygen/

[6] Microsoft Windows 98 - http://www.microsoft.com/en/us/default.aspx

[7] GDB- http://www.gnu.org/software/gdb/

[8] Tron- http://en.wikipedia.org/wiki/Tron_(video_game)

Anexos

Anexo A - Documentação do Código

Main (main.c) – módulo responsável pela instalação e desinstalação das interrupções e por toda a mecânica de jogo.

- void rtc_irq_handler(void): handler do real time clock.
- void serial_isr(void): handler do porto de série.
- void init_serie(void): função que programa o porto de série e instala o seu handler.
- void finalize_serie(void): repõe o handler original do porto de série.
- void escreve_ficheiro_pontuacoes(void): escreve as pontuações no ficheiro de texto “pont.txt”.
- void le_ficheiro_pontuacoes(void): lê o ficheiro “pont.txt” e guarda os seus valores numa estrutura.
- void mostra_pontuacoes(void): mostra no ecrã as melhores pontuações.
- void actualiza_pontuacao(int p): actualiza a estrutura de pontuações.
- void change_key(Byte *key): altera as teclas pré-definidas para jogar.
- void config_keys(void): imprime a interface de escolha das novas teclas de jogo.
- void song_pre_load_queue(int n_notes): carrega as notas da queue “musica” o número de notas passadas como argumento.
- void init(void): inicializa alguns aspectos relacionados com interrupções e com a música quando o jogo é lançado.
- void finalize(void): finaliza alguns aspectos relacionados com interrupções quando o jogador sai da aplicação.
- void desenha_ecra(void): desenha toda a área de jogo.
- void jogar_multiplayer(void): função responsável pela jogabilidade entre os dois jogadores em porto de série.
- int (int x, int y): verifica qual é o melhor caminho a seguir, tendo em conta a posição do jogador adversário.
- void jogar_singleplayer(void): função responsável pela jogabilidade entre m jogador e o CPU.
- void jogar_menu_singleplayer(void): desenha o menu do modo single-player.
- void menu_jogar(void): desenha o menu de jogo.
- void draw_menu(void): desenha o menu inicial.
- int main(int a, char *argv[]): função main(), reponsável por lançar todas as outras funções.

Video (video.c) – modulo que contém todas as funções responsáveis pela comunicação com a placa gráfica e a sua programação.

- unsigned long getMemAddr(int mode): obtém o endereço da memória gráfica.

- void set_pixel(int x, int y, int color, char *base): desenha um pixel nas coordenadas (x,y).
- int get_pixel(int x, int y, char *base): devolve a cor do pixel de coordenadas (x,y).
- void clear_screen(char color, char *base): pinta o ecrã da cor passada como argumento.
- void draw_line(int xi, int yi, int xf, int yf, int color, char *base): desenha uma linha entre (xi,yi) e (xf, yf) de cor "color".

Sprite (sprite.c) – modulo com todas as funções necessárias para criar/destruir sprites.

- char *read_xpm(char *map[], int *wd, int *ht): lê uma sprite com largura e comprimento passado como argumento.
- void drawCharAt(char c, int x_ori, int y_ori, int fore_color, int back_color, int char_scale, char *video_base): desenha uma letra numa determinada posição, cor de fundo e cor passada como argumento.
- void draw_string(char *s, int x_ori, int y_ori, int fore_color, int back_color, int char_scale, char *video_base): desenha uma frase numa determinada posição, cor de fundo e cor passada como argumento.
- Sprite *create_sprite(char *pic[], char *base, int horizontal_x, int vertical_y): cria uma sprite numa determinada posição passada por argumento.
- void destroy_sprite(Sprite *spr, char *base): destrói uma sprite.
- void draw_sprite(Sprite *spr, char *base): desenha uma sprite.
- void erase_sprite(Sprite *spr, char *base): apaga uma sprite.
- void drawIntAt(int num, int x_ori, int y_ori, int fore_color, int back_color, int char_scale, char *video_base): desenha um número numa determinada posição, cor de fundo e cor passada como argumento.

Keyboard (kbc.c) – modulo responsável pela leitura e escrita do teclado.

- int write_kbc(unsigned adr, unsigned data) : escreve no endereço 'adr' a 'data' passada como argumento. Retorna -1 quando acontece timeout.
- int read_kbc(void): retorna o 'data read' do registo data.
- void blink_leds(void): piscam os três LEDs do teclado.
- void blink_led(uchar led): pisca o LED passado como argumento.

Timer (timer.c) – modulo com as funções que activam/desactivam as interrupções do timer.

- void timer_enable(void): instala a inturrpção do TIMER0.
- void timer_disable(void): desinstala a interrupção do TIMER0.
- void timer_init(int timer, int mode): programa o timer (0,1 ou 2) com o modo passado como argumento.
- void timer_load(int timer, int value): carrega o timer (0,1 ou 2) com o modo passado como argumento.
- void mili_sleep(int mili): espera de mili milisegundos.

- int get_divider(float freq): devolve a frequência correcta para programar o timer.
- void sec_sleep(int secs): espera de secs segundos.

Music (music.c) – funções responsáveis pelo uso do altifalante.

- void speaker_on(void): active o speaker.
- void speaker_off(void): desactiva o speaker.
- void play_note(Note *note): toca uma nota.
- void play_song(Song *s): toca uma música.
- void beep(void): emite um beep sonoro.

Interrupções (ints.c) – contém as funções de instalação/desinstalação de interrupções escritas na linguagem Assembly ou C e função de habilitar/desabilitar de PIC's.

- void disable_irq(int irq): desabilita a interrupção irq.
- void enable_irq(int irq): habilita a interrupção irq.
- void mask_pic(int irq): mascara o bit correspondente a 'irq', desabilitando as suas interrupções.
- void unmask_pic(int irq): desmascara o bit correspondente a 'irq', habilitando as suas interrupções.

Queue (queue.c) – funções para manipular objectos do tipo queue, que simulam o conceito de pilha.

- void queueInit(Queue *q): inicia uma queue do tipo char.
- Bool queuePut(Queue *q, Byte elem): coloca elem no topo da pilha.
- int queueGet(Queue *q): devolve o próximo elemento da pilha.
- Bool queueEmpty(Queue *q): verifica que a pilha está vazia.
- Bool queueFull(Queue *q): verifica se a pilha atingiu o tamanho máximo.
- void queueClear(Queue *q): apaga todos os elementos da queue.

Real Time Clock (rtc.c) – funções reponsáveis por ler/escrever o RTC do computador.

- Byte bcd2dec(Byte i): converte BCD para decimal.
- Byte dec2bcd(Byte i): converte decimal para BCD.
- void rtc_valid(void): espera que os dados do rtc sejam válidos.
- Byte read_rtc(Byte add): lê os dados actuais do RTC; não valida bit.
- Byte read_rtcv(Byte add): lê os dados actuais do RTC; valida o bit.
- void write_rtc(Byte add, Byte value): escreve o valor 'add' no RTC; não valida o bit.
- void write_rtcv(Byte add, Byte value): escreve o valor 'add' no RTC; valida o bit
- void rtc_read_time(RTC_TIME *time): lê o tempo do RTC.
- void rtc_read_date(RTC_DATE *date): lê a data do RTC.
- void rtc_read_alarm(RTC_TIME *time): lê o alarme do RTC.
- void rtc_write_alarm(RTC_TIME *time): escreve o alarme do RTC.

Song (song.c) – funções que definem o objecto do tipo Song.

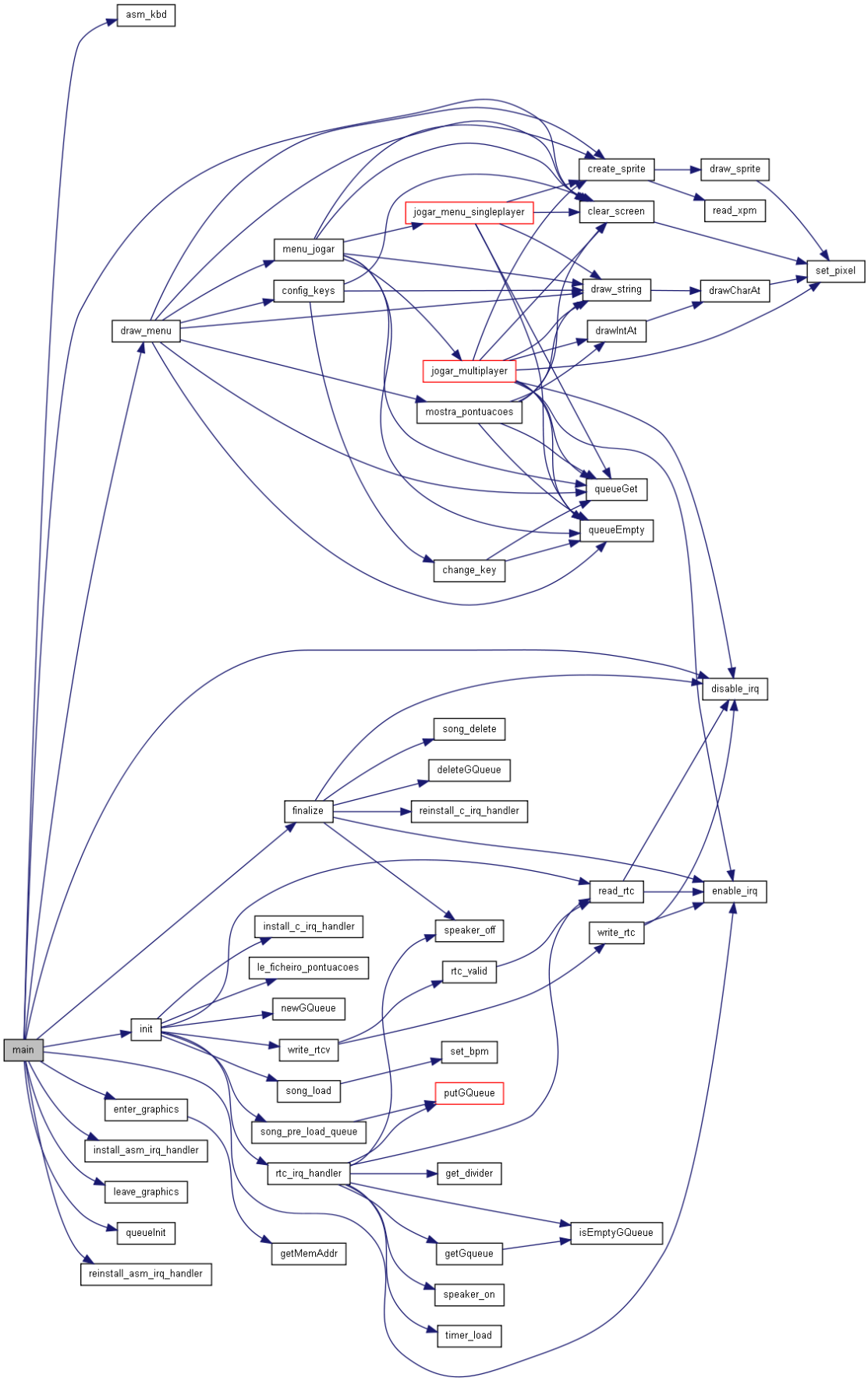
- Song *song_load(char *filename): lê do ficheiro uma música.
- void set_bpm(Song *s, int beats_measure, float bpm): define o bmp da música.

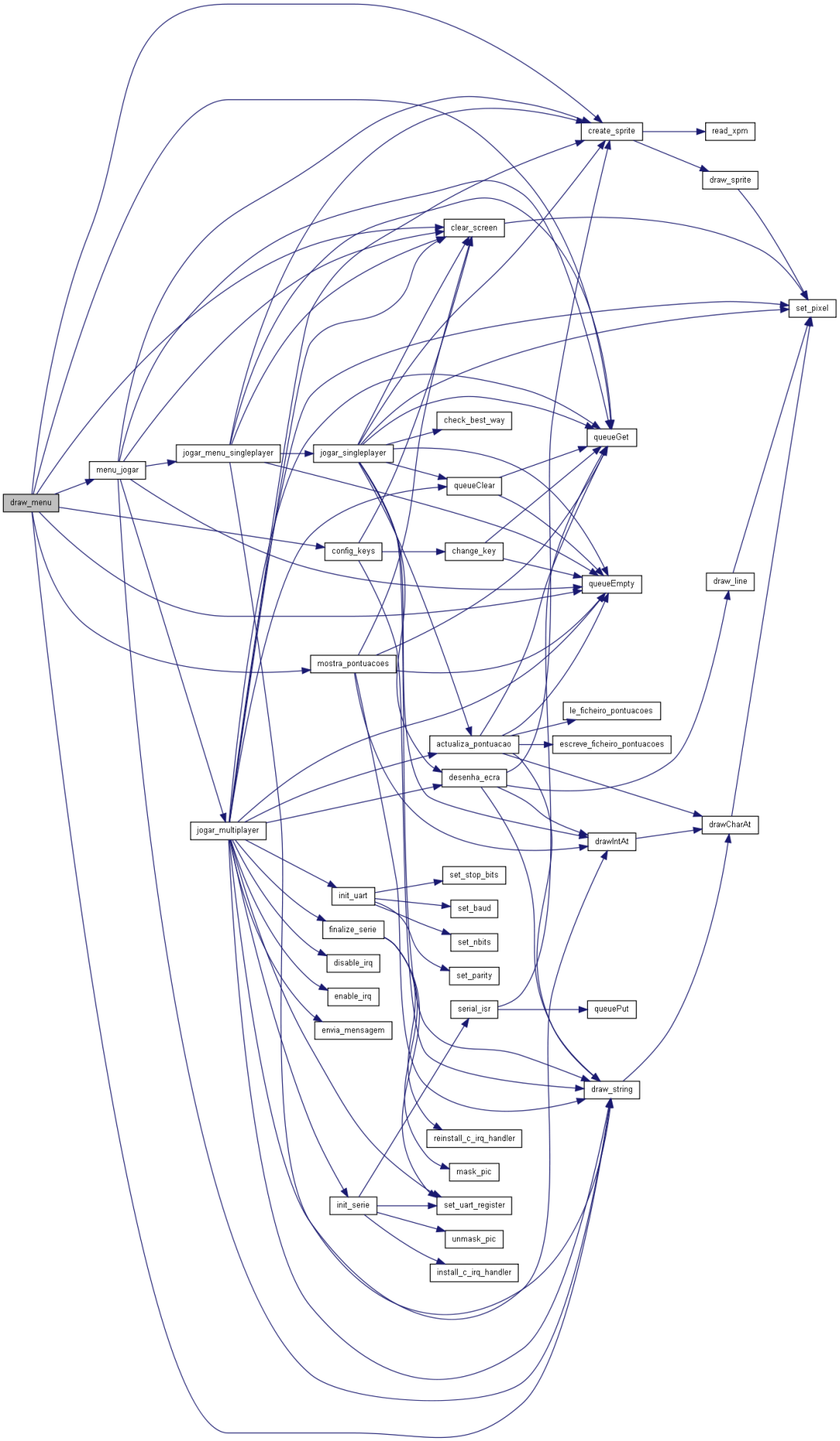
- void modify_bpm(Song *s, float old_bpm, float new_bpm): modifica o bmp da música.
- void modify_pitch(Song *s, int halfsteps_up): modifica o pitch da música.
- void song_save(Song *s, char *filename): guarda a música no ficheiro.
- void song_delete(Song *s): apaga a música.

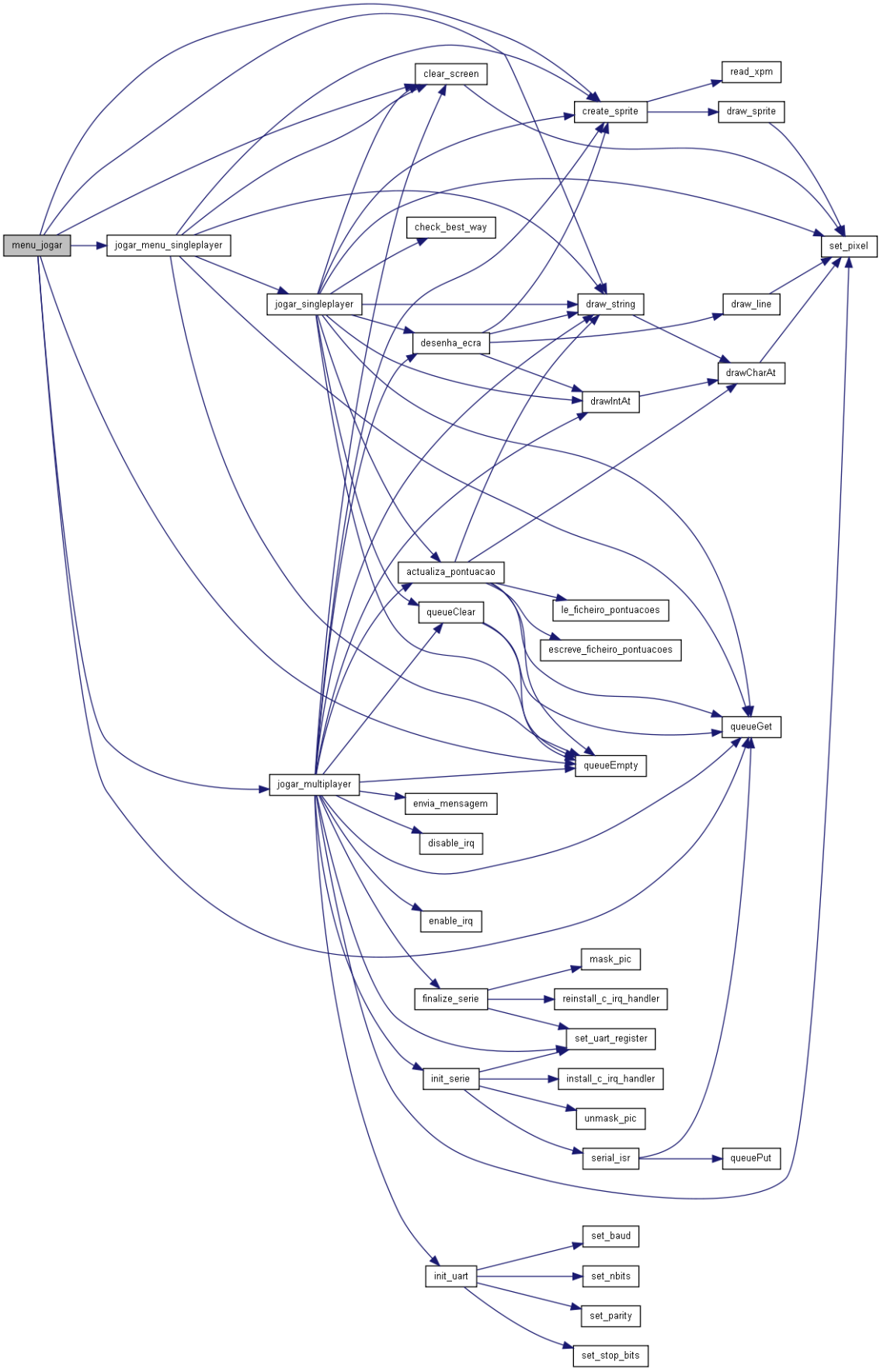
Serie (serie.c) – modulo com as funções responsáveis pela leitura, escrita e comunicação com porto de série.

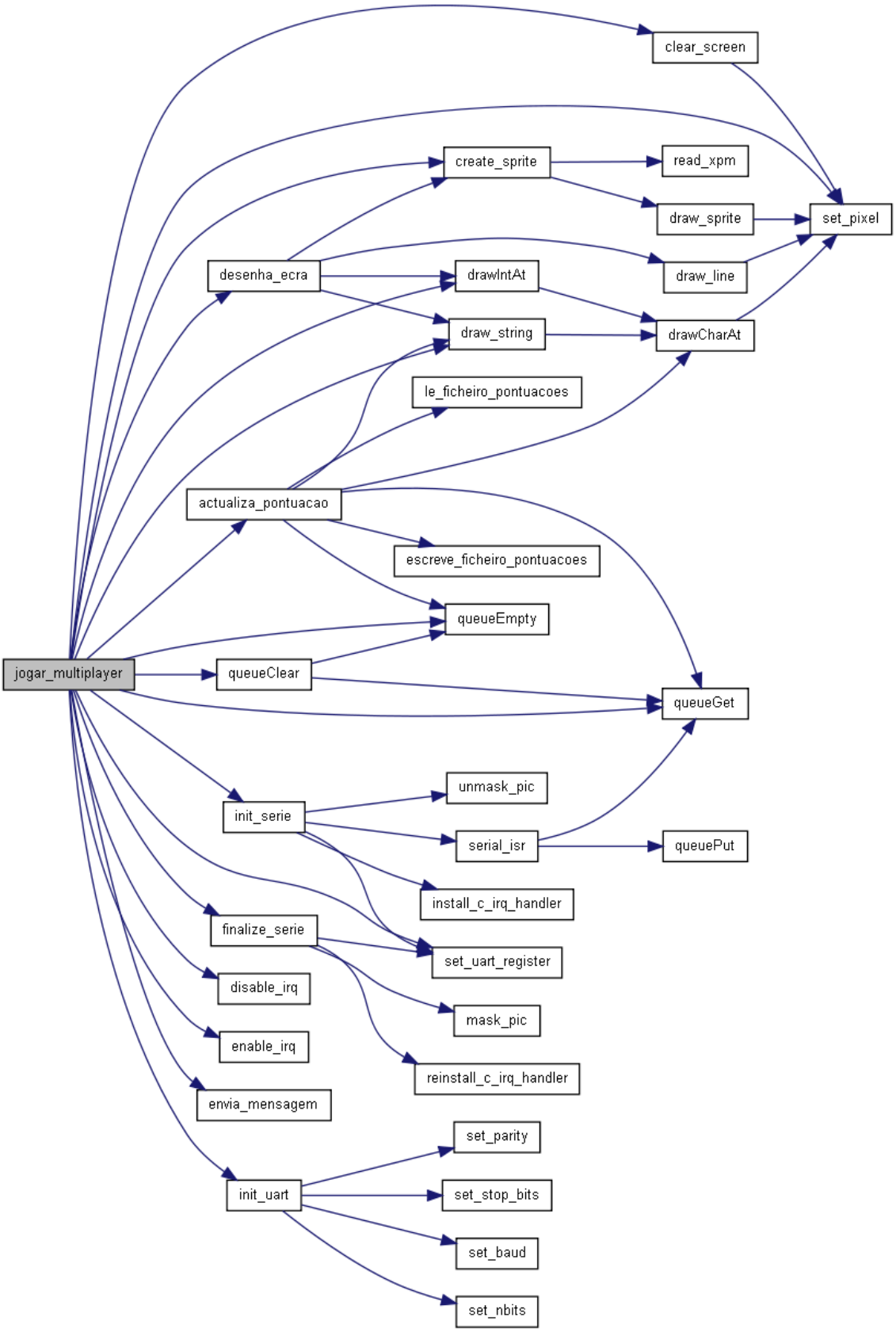
- int get_baud(Word base): retorna o baud do porto de série.
- void set_baud(Word base, int rate): define o baud do porto de série.
- Byte get_parity(Word base): devolve o valor do bit parity.
- void set_parity(Word base, Byte par): define o bit parity.
- int get_nbits(Word base): devolve o valor de nbits do porto de série.
- void set_nbits(Word base, int nbits): define o valor de nbits.
- int get_stop_bits(Word base): devolve o valor de stop bits.
- void set_stop_bits(Word base, int stop): define o valor de stop bits.
- void init_uart(Word base, int rate, int nbits, int stop_bits, Byte parity, Bool rx_int, Bool tx_int, Bool fifo): inicializa o porto de série.
- void envia_mensagem(Word base, char c): envia uma mensagem pelo porto de série.
- char recebe_mensagem(Word base): recebe uma mensagem pelo porto de série.
- Bool mensagem_espera(Word base): verifica se há alguma mensagem em espera.
- Byte get_uart_register(Word base, Word reg): devolve o registo da uart.
- void set_uart_register(Word base, Word reg, Byte b): define o valor da uart.

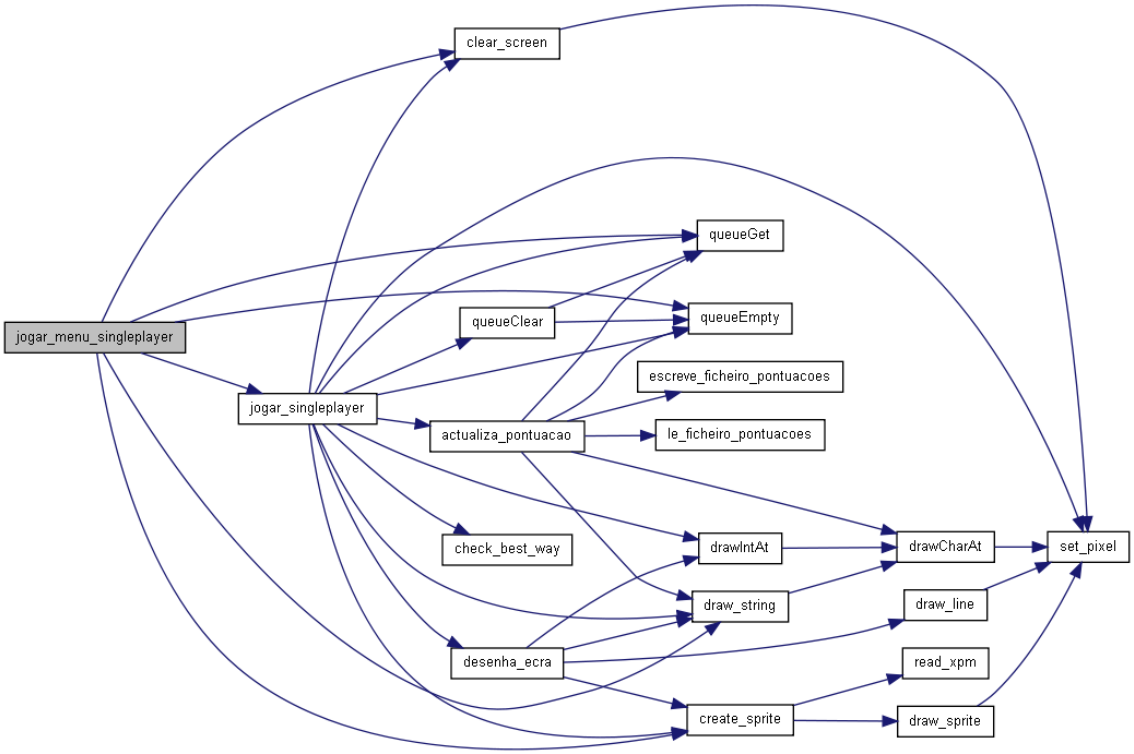
Anexo B - Diagrama de Chamada de Funções

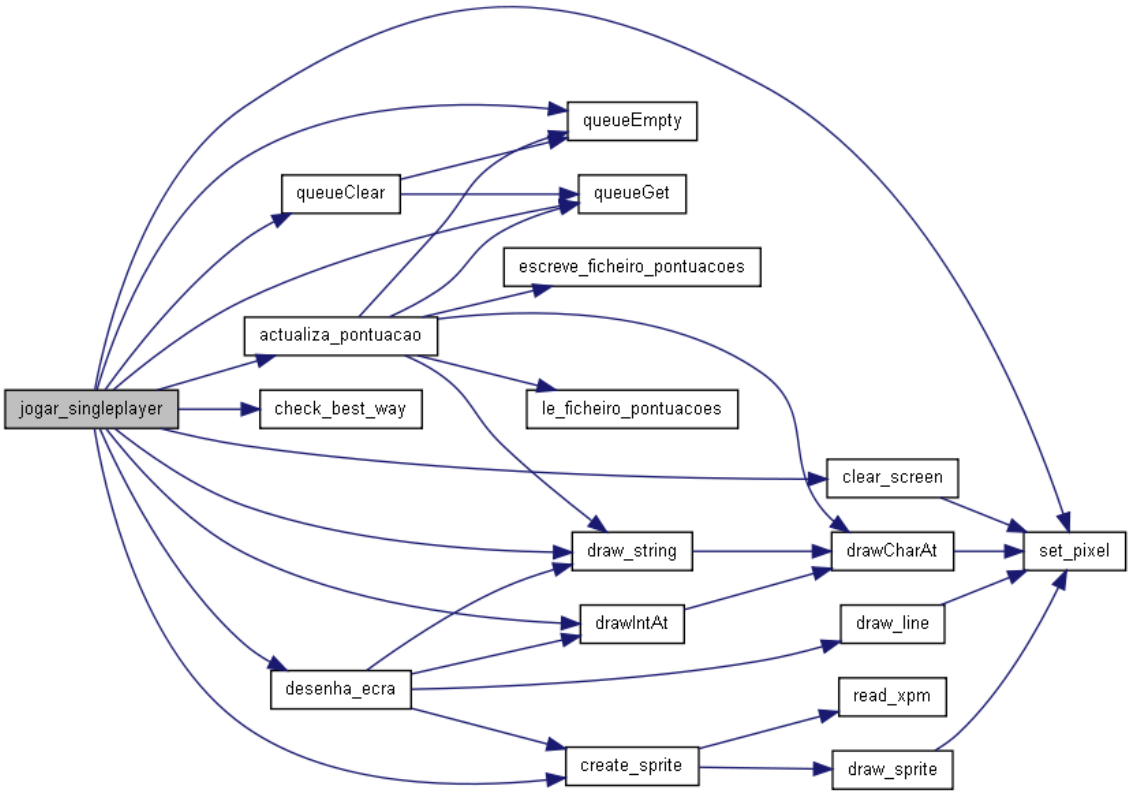


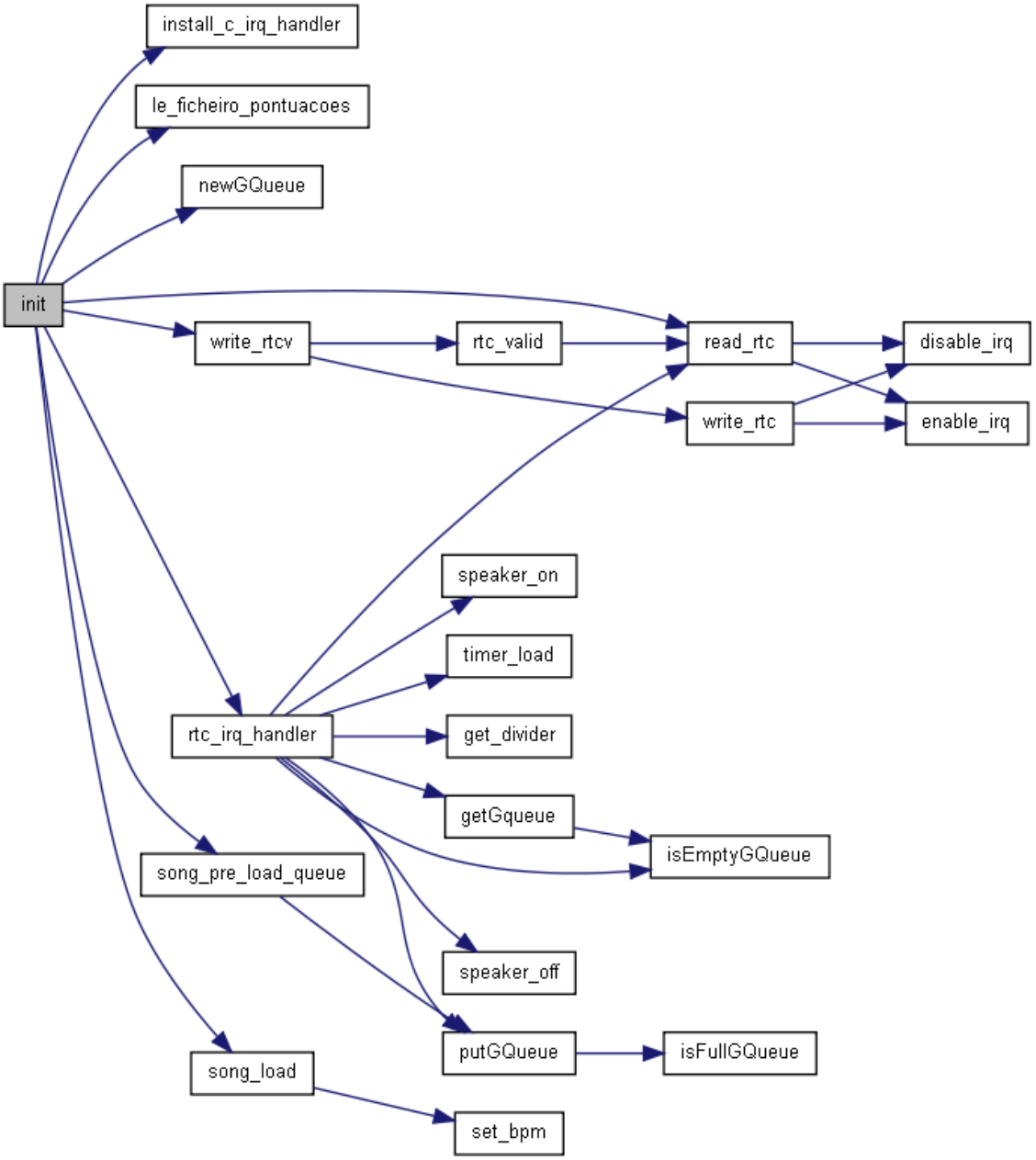

draw_menu
menu_jogar
jogar_menu_singleplayer
jogar_singleplayer
jogar_multiplayer
config_keys
mostra_pontuacoes
create_sprite
read_xpm
draw_sprite
clear_screen
set_pixel
check_best_way
queueGet
queueClear
change_key
queueEmpty
draw_line
le_ficheiro_pontuacoes
actualiza_pontuacao
escreve_ficheiro_pontuacoes
desenha_ecra
drawCharAt
drawIntAt
set_stop_bits
init_uart
set_baud
finalize_serie
set_nbits
disable_irq
set_parity
enable_irq
serial_isr
queuePut
envia_mensagem
draw_string
reinstall_c_irq_handler
mask_pic
init_serie
set_uart_register
unmask_pic
install_c_irq_handler

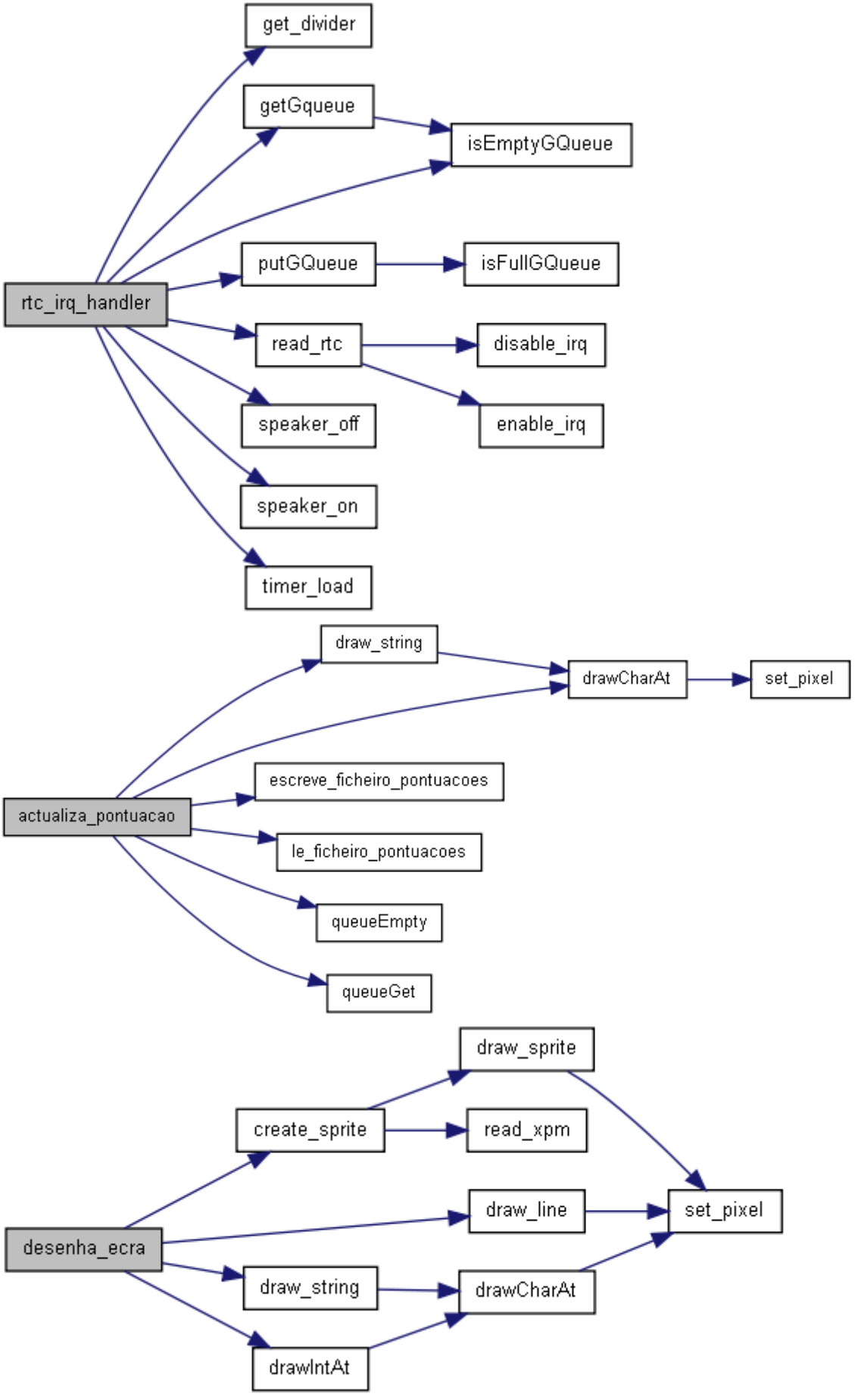

menu_jogar
jogar_menu_singleplayer
clear_screen
create_sprite
read_xpm
draw_sprite
check_best_way
set_pixel
jogar_singleplayer
draw_string
draw_line
desenha_ecra
drawCharAt
drawIntAt
actualiza_pontuacao
queueClear
le_ficheiro_pontuacoes
escreve_ficheiro_pontuacoes
queueGet
queueEmpty
jogar_multiplayer
envia_mensagem
disable_irq
enable_irq
mask_pic
finalize_serie
reinstall_c_irq_handler
set_uart_register
init_serie
install_c_irq_handler
unmask_pic
serial_isr
queuePut
set_baud
init_uart
set_nbits
set_parity
set_stop_bits

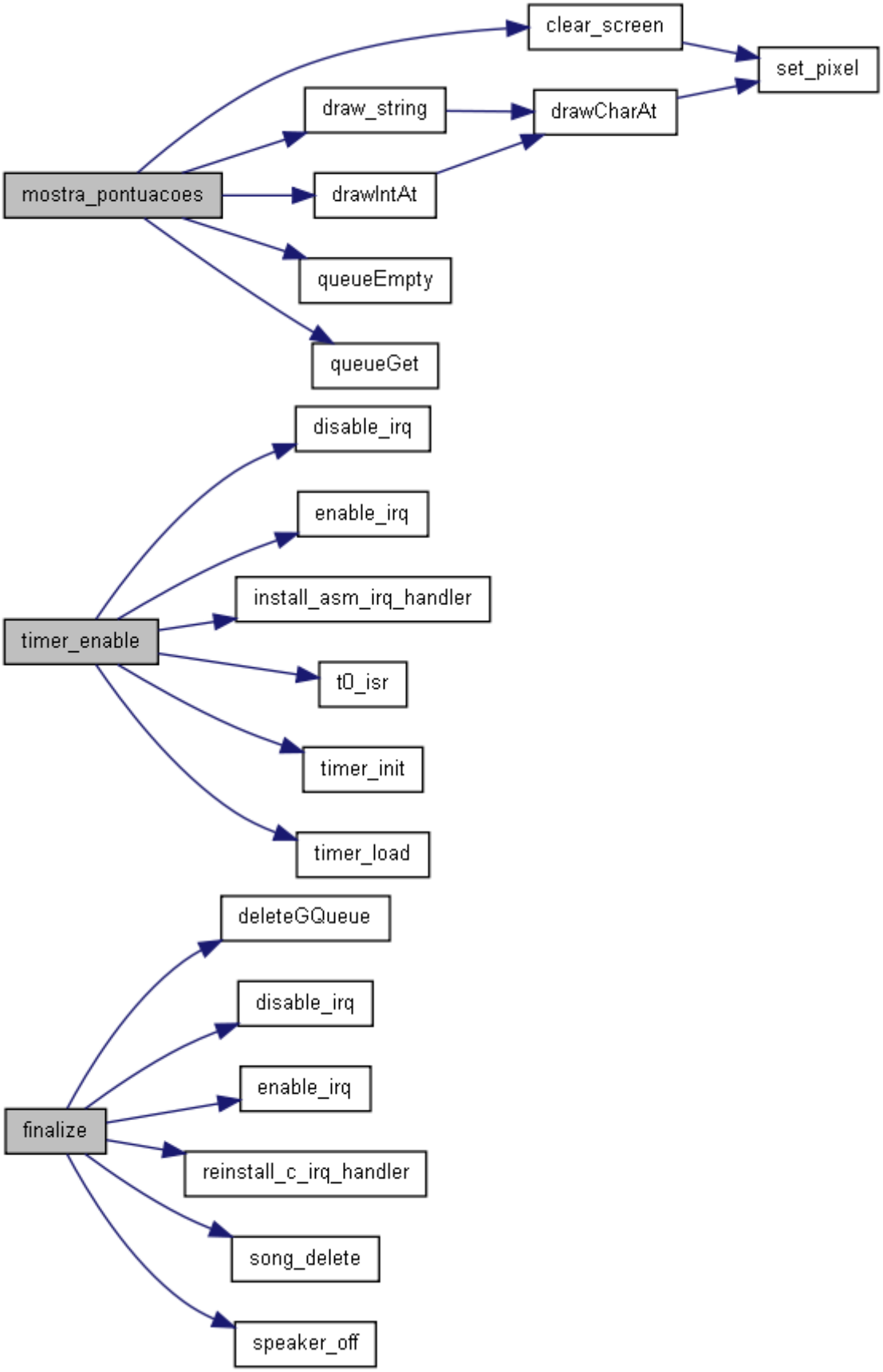

clear_screen
create_sprite
read_xpm
draw_sprite
set_pixel
desenha_ecra
drawIntAt
draw_line
draw_string
drawCharAt
le_ficheiro_pontuacoes
actualiza_pontuacao
escreve_ficheiro_pontuacoes
queueEmpty
jogar_multiplayer
queueClear
queueGet
unmask_pic
init_serie
serial_isr
queuePut
install_c_irq_handler
finalize_serie
set_uart_register
disable_irq
mask_pic
enable_irq
reinstall_c_irq_handler
envia_mensagem
set_parity
init_uart
set_stop_bits
set_baud
set_nbits

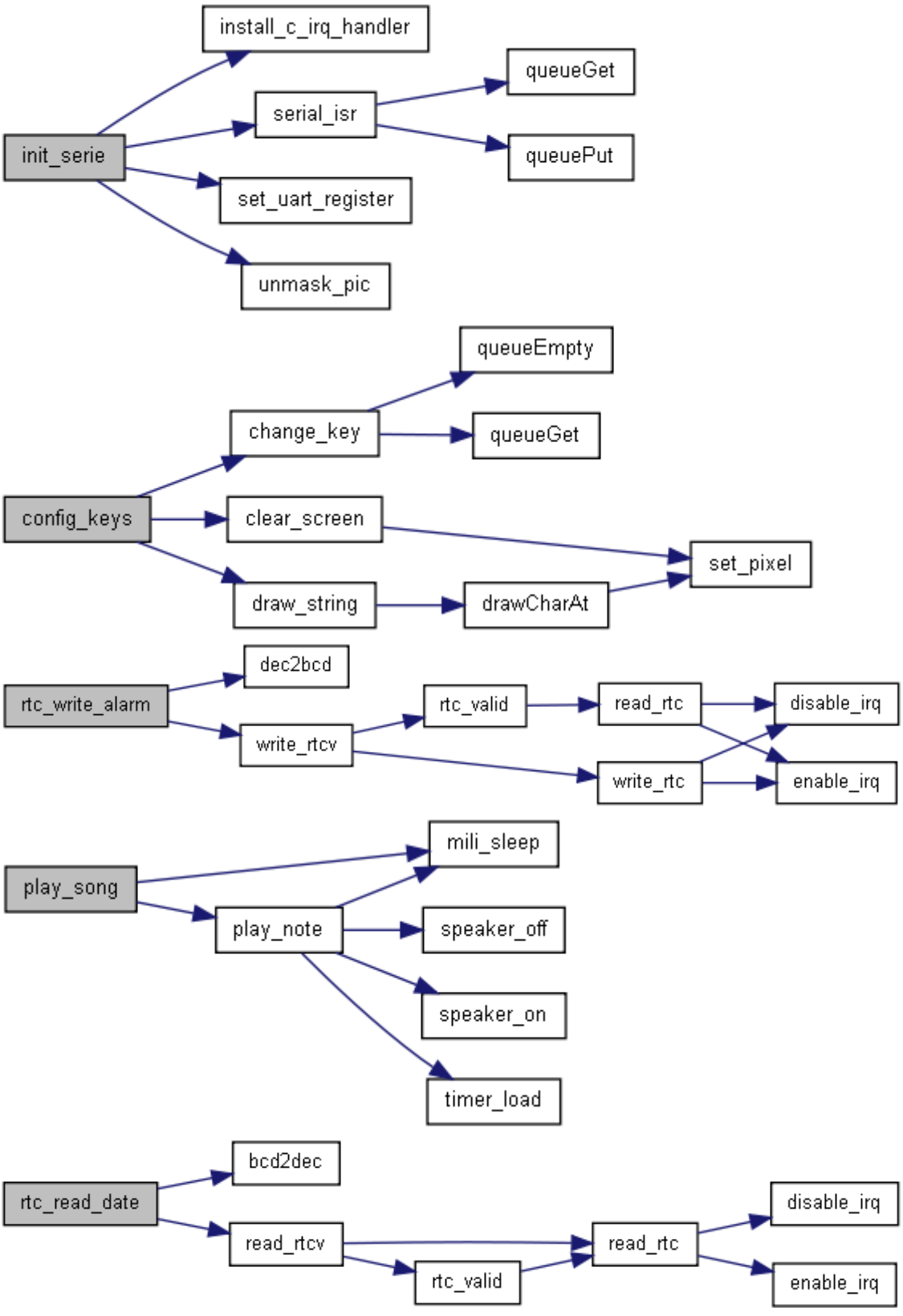

clear_screen
queueGet
queueClear
queueEmpty
jogar_menu_singleplayer
escreve_ficheiro_pontuacoes
jogar_singleplayer
actualiza_pontuacao
le_ficheiro_pontuacoes
check_best_way
drawIntAt
drawCharAt
set_pixel
draw_string
draw_line
desenha_ecra
read_xpm
create_sprite
draw_sprite

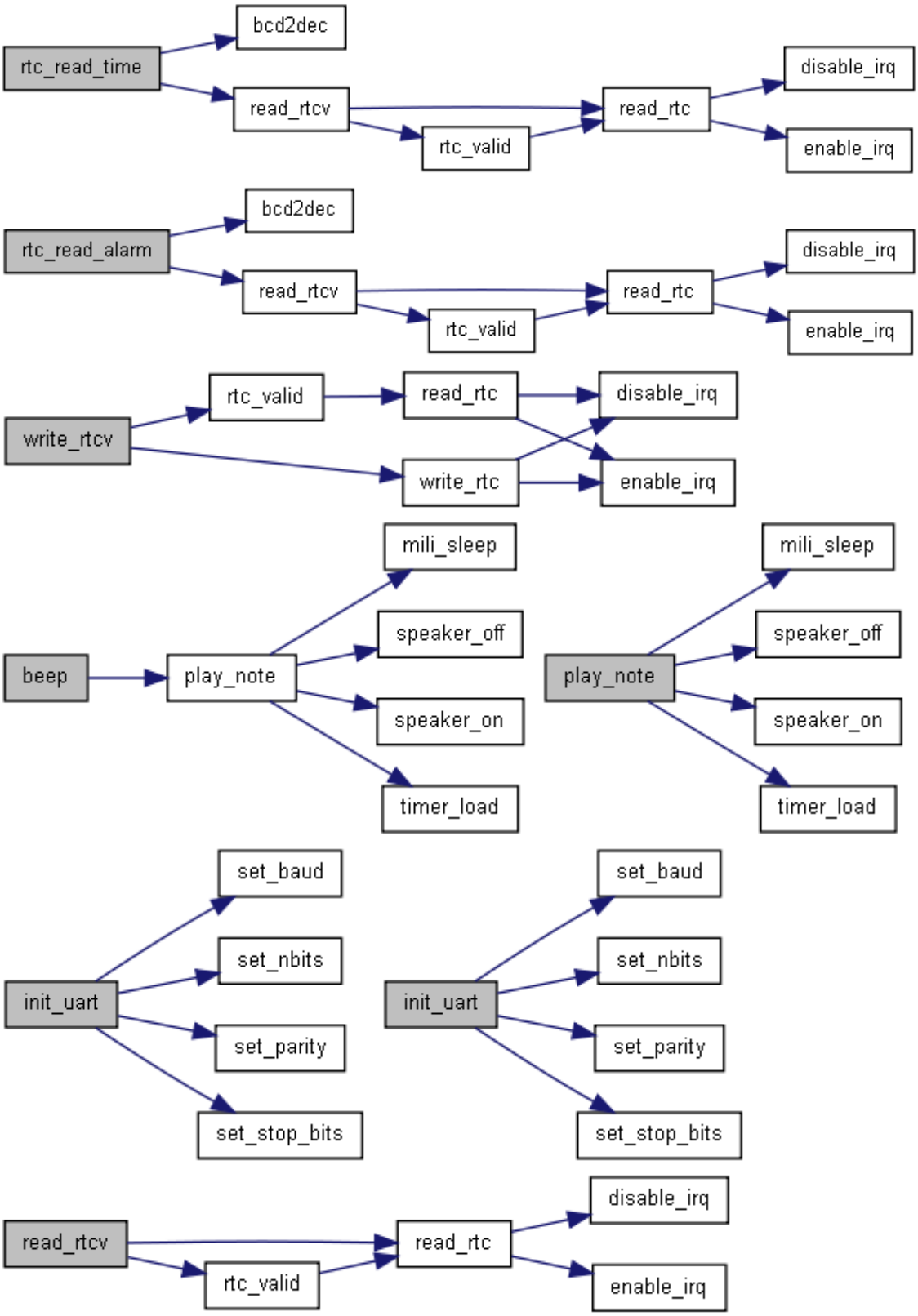

queueEmpty
queueClear
queueGet
escreve_ficheiro_pontuacoes
actualiza_pontuacao
jogar_singleplayer
check_best_way
le_ficheiro_pontuacoes
clear_screen
draw_string
drawCharAt
set_pixel
drawIntAt
draw_line
desenha_ecra
read_xpm
create_sprite
draw_sprite

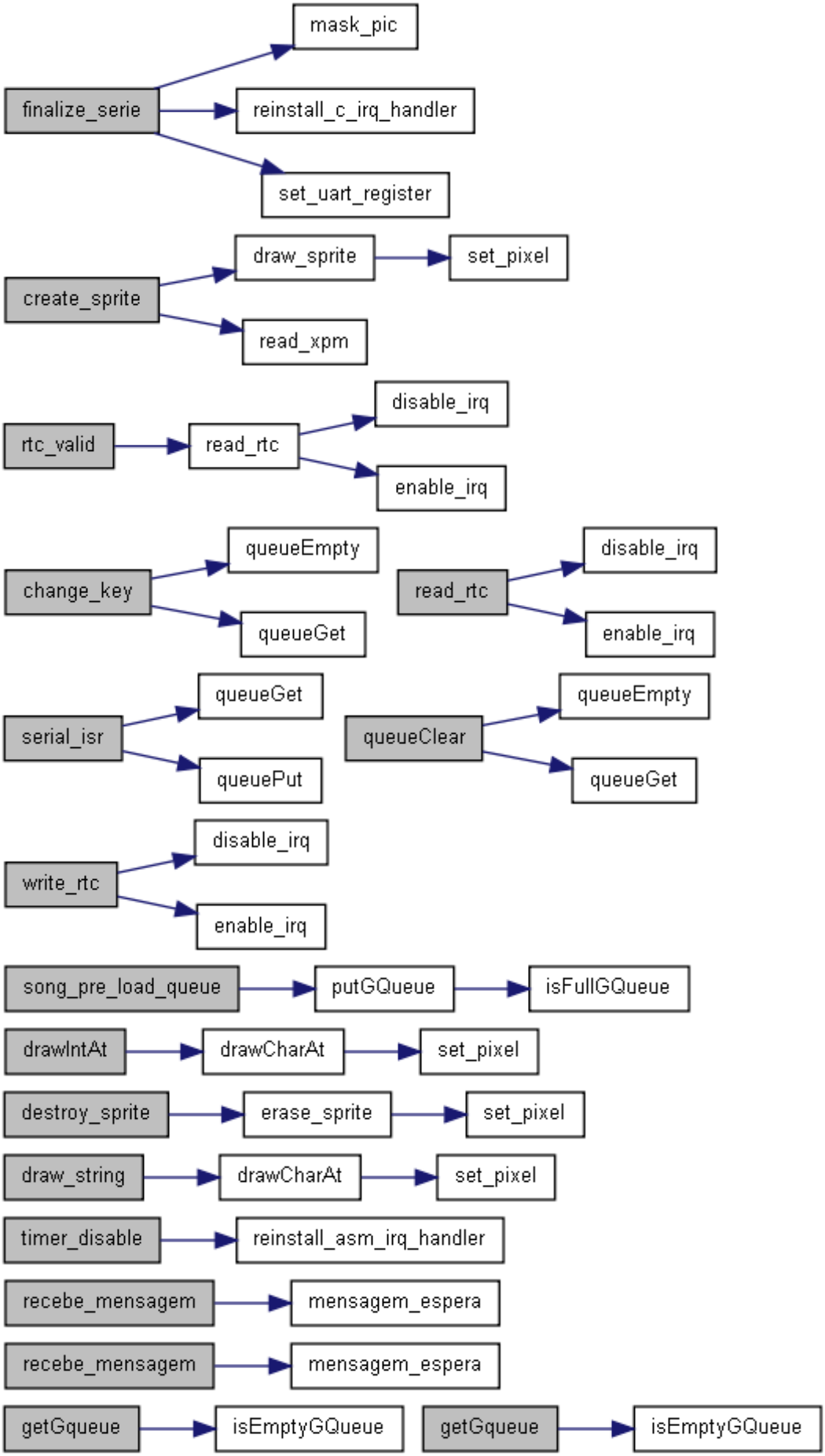

install_c_irq_handler
le_ficheiro_pontuacoes
newGQueue
init
write_rtcv
rtc_valid
read_rtc
disable_irq
write_rtc
enable_irq
speaker_on
timer_load
rtc_irq_handler
get_divider
getGqueue
isEmptyGQueue
song_pre_load_queue
speaker_off
putGQueue
isFullGQueue
song_load
set_bpm

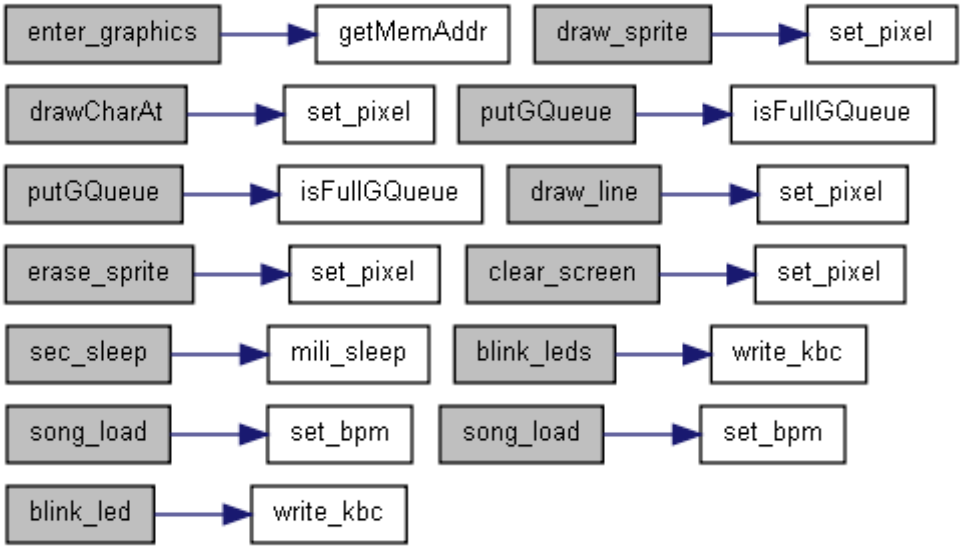

get_divider
getGqueue
isEmptyGQueue
putGQueue
isFullGQueue
rtc_irq_handler
read_rtc
disable_irq
enable_irq
speaker_off
speaker_on
timer_load
draw_string
drawCharAt
set_pixel
escreve_ficheiro_pontuacoes
actualiza_pontuacao
le_ficheiro_pontuacoes
queueEmpty
queueGet
draw_sprite
create_sprite
read_xpm
draw_line
set_pixel
desenha_ecra
draw_string
drawCharAt
drawIntAt


clear_screen
set_pixel
draw_string
drawCharAt
mostra_pontuacoes
drawIntAt
queueEmpty
queueGet
disable_irq
enable_irq
install_asm_irq_handler
timer_enable
t0_isr
timer_init
timer_load
deleteGQueue
disable_irq
enable_irq
finalize
reinstall_c_irq_handler
song_delete
speaker_off


install_c_irq_handler
queueGet
serial_isr
init_serie
queuePut
set_uart_register
unmask_pic
queueEmpty
change_key
queueGet
config_keys
clear_screen
set_pixel
draw_string
drawCharAt
dec2bcd
rtc_write_alarm
rtc_valid
read_rtc
disable_irq
write_rtcv
write_rtc
enable_irq
mili_sleep
play_song
play_note
speaker_off
speaker_on
timer_load
bcd2dec
rtc_read_date
disable_irq
read_rtcv
read_rtc
rtc_valid
enable_irq


rtc_read_time
bcd2dec
read_rtcv
rtc_valid
read_rtc
disable_irq
enable_irq
rtc_read_alarm
bcd2dec
read_rtcv
rtc_valid
read_rtc
disable_irq
enable_irq
write_rtcv
rtc_valid
read_rtc
disable_irq
write_rtc
enable_irq
beep
play_note
mili_sleep
speaker_off
speaker_on
timer_load
play_note
mili_sleep
speaker_off
speaker_on
timer_load
init_uart
set_baud
set_nbits
set_parity
set_stop_bits
init_uart
set_baud
set_nbits
set_parity
set_stop_bits
read_rtcv
rtc_valid
read_rtc
disable_irq
enable_irq


mask_pic
finalize_serie
reinstall_c_irq_handler
set_uart_register
draw_sprite
set_pixel
create_sprite
read_xpm
disable_irq
rtc_valid
read_rtc
enable_irq
queueEmpty
change_key
disable_irq
read_rtc
queueGet
enable_irq
queueGet
queueEmpty
serial_isr
queueClear
queuePut
queueGet
disable_irq
write_rtc
enable_irq
song_pre_load_queue
putGQueue
isFullGQueue
drawIntAt
drawCharAt
set_pixel
destroy_sprite
erase_sprite
set_pixel
draw_string
drawCharAt
set_pixel
timer_disable
reinstall_asm_irq_handler
recebe_mensagem
mensagem_espera
recebe_mensagem
mensagem_espera
getGqueue
isEmptyGQueue
getGqueue
isEmptyGQueue


enter_graphics
getMemAddr
draw_sprite
set_pixel
drawCharAt
set_pixel
putGQueue
isFullGQueue
putGQueue
isFullGQueue
draw_line
set_pixel
erase_sprite
set_pixel
clear_screen
set_pixel
sec_sleep
mili_sleep
blink_leds
write_kbc
song_load
set_bpm
song_load
set_bpm
blink_led
write_kbc

Anexo C – Histórico do desenvolvimento

Revision: 68
Author: fsschmitt
Date: 15:21:36, domingo, 30 de Maio de 2010
Message:
Makefile modificado.

Modified : /trunk/makefile

Revision: 67
Author: fsschmitt
Date: 02:27:09, domingo, 30 de Maio de 2010
Message:
Versao 1.3
- Acrescentado 3ª Inteligencia Artificial (Wall Hugger) (Ideia inicial, sem random)
- Acrescentado às outras IA's nomes:
->(Around+Random) : Wall hugger + random (300..300 steps)
->(StayAway) : Direcciona-se para o lado onde poderia andar mais.
->(WallHugger) : Como o proprio nome indica, anda sempre junto às paredes, com uma preferencia (up, down, right, left).

Modified : /trunk/main.c

Revision: 66
Author: fsschmitt
Date: 02:09:25, domingo, 30 de Maio de 2010
Message:
Versao 1.2 (Implementada 2ª Inteligencia Artificial)
-Adicionado jogar_menu_singleplayer.
-Alterado tabela int para char.
-Alterado y-100 para y-150
-Acrescentado Variavel global IA (inteligencia artificial) 1 ou 2.
-tabela passado a variavel global.

Testes vs Funcionamento: 90% (IA2, pode conter alguns erros ainda a verificar)

Modified : /trunk/main.c
Modified : /trunk/pont.txt

Revision: 65
Author: fsschmitt
Date: 01:57:15, domingo, 30 de Maio de 2010
Message:
Makefile corrigido. Main.c Alteracoes nos valores a verificar da tabela em y (antes: -100, depois -150).

Resultado: Melhoramento da IA.

Modified : /trunk/main.c
Modified : /trunk/makefile

Revision: 64
Author: fsschmitt
Date: 01:07:49, domingo, 30 de Maio de 2010
Message:
Codigo desnecessario apagado. Versao1.1

Modified : /trunk/main.c

Revision: 63
Author: fsschmitt
Date: 00:48:39, domingo, 30 de Maio de 2010
Message:
Apaguei ficheiros desnecessarios.

Deleted : /trunk/PROJ.EXE
Deleted : /trunk/crazyf.TXT
Deleted : /trunk/isr.asmBACK
Deleted : /trunk/lambada.txt
Deleted : /trunk/new 2
Deleted : /trunk/rtc_asm.asmBACK

Revision: 62
Author: fsschmitt
Date: 00:33:32, domingo, 30 de Maio de 2010
Message:
Versao 1.0 (Pontuacao a funcionar 100%), Funcoes optimizadas.

Modified : /trunk/main.c
Modified : /trunk/makefile

Modified : /trunk/pont.txt
Modified : /trunk/sprite.c
Modified : /trunk/sprite.h
Modified : /trunk/video.c
Modified : /trunk/video.h

Revision: 61
Author: fsschmitt
Date: 18:28:53, sábado, 29 de Maio de 2010
Message:
Melhoramentos da Interface grafica,

Verificacao da pontuacao

Modified : /trunk/PROJ.EXE
Modified : /trunk/main.c
Modified : /trunk/pixmap.h
Modified : /trunk/pont.txt
Modified : /trunk/sprite.c

Revision: 60
Author: inpanzinator
Date: 12:47:33, sábado, 29 de Maio de 2010
Message:
Pontuacões a funcionar.

Added : /trunk/PROJ.EXE
Modified : /trunk/main.c
Modified : /trunk/makefile
Added : /trunk/pont.txt
Deleted : /trunk/pontuacoes.txt
Deleted : /trunk/proj.exe
Deleted : /trunk/video-text.c
Deleted : /trunk/video-text.h

Revision: 59
Author: inpanzinator
Date: 17:08:53, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/pontuacoes.txt
Modified : /trunk/proj.exe

Revision: 58
Author: inpanzinator
Date: 16:31:10, segunda-feira, 24 de Maio de 2010
Message:
>IntelegÃªncia artificial
>PontuaÃ§Ã£o
>Sprites

Modified : /trunk/main.c
Modified : /trunk/pixmap.h
Modified : /trunk/pontuacoes.txt
Modified : /trunk/proj.exe
Modified : /trunk/sprite.c
Modified : /trunk/sprite.h
Modified : /trunk/video.c
Modified : /trunk/video.h

Revision: 57
Author: inpanzinator
Date: 12:16:07, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 56
Author: inpanzinator
Date: 12:10:23, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 55
Author: inpanzinator

Date: 12:02:53, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 54
Author: inpanzinator
Date: 11:33:45, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe
Modified : /trunk/queue.c

Revision: 53
Author: inpanzinator
Date: 11:27:18, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 52
Author: inpanzinator
Date: 11:23:46, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 51
Author: inpanzinator
Date: 11:22:03, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c

Revision: 50
Author: inpanzinator
Date: 11:19:59, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 49
Author: inpanzinator
Date: 11:18:16, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 48
Author: inpanzinator
Date: 11:15:14, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 47
Author: inpanzinator
Date: 11:08:41, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 46
Author: inpanzinator
Date: 11:05:56, segunda-feira, 24 de Maio de 2010

Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 45
Author: inpanzinator
Date: 11:02:21, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 44
Author: inpanzinator
Date: 11:01:44, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 43
Author: inpanzinator
Date: 10:48:09, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 42
Author: inpanzinator
Date: 10:44:26, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 41
Author: inpanzinator
Date: 10:30:39, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 40
Author: inpanzinator
Date: 10:27:01, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 39
Author: inpanzinator
Date: 10:23:36, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 38
Author: inpanzinator
Date: 10:10:42, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 37
Author: inpanzinator
Date: 10:02:56, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 36
Author: inpanzinator
Date: 09:57:42, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 35
Author: inpanzinator
Date: 09:54:02, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 34
Author: inpanzinator
Date: 09:49:47, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 33
Author: inpanzinator
Date: 09:38:54, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 32
Author: inpanzinator
Date: 09:27:33, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 31
Author: inpanzinator
Date: 09:22:03, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 30
Author: inpanzinator
Date: 09:10:38, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 29
Author: inpanzinator
Date: 09:06:09, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 28
Author: inpanzinator
Date: 09:03:46, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/ints.c
Modified : /trunk/ints.h
Modified : /trunk/proj.exe

Revision: 27
Author: inpanzinator
Date: 08:59:52, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 26
Author: inpanzinator
Date: 08:54:28, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/ints.h
Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 25
Author: inpanzinator
Date: 08:42:58, segunda-feira, 24 de Maio de 2010
Message:

Added : /trunk/main.c

Revision: 24
Author: inpanzinator
Date: 08:42:21, segunda-feira, 24 de Maio de 2010
Message:

Deleted : /trunk/main.c
Deleted : /trunk/main.c.mine

Revision: 23
Author: inpanzinator
Date: 08:38:28, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/ints.c
Modified : /trunk/ints.h
Added : /trunk/main.c.mine
Modified : /trunk/proj.exe

Revision: 22
Author: fsschmitt
Date: 08:23:35, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 21
Author: inpanzinator
Date: 08:17:23, segunda-feira, 24 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 20
Author: inpanzinator
Date: 08:12:43, segunda-feira, 24 de Maio de 2010
Message:

Deleted : /trunk/A.OUT
Deleted : /trunk/BLA
Deleted : /trunk/PROJ.7z
Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 19
Author: inpanzinator
Date: 17:49:25, sexta-feira, 21 de Maio de 2010
Message:
A comunicaÃ§Ã£o com porto de sÃ©rie estÃ¡ muito lenta, fazendo com que os computadores nÃ£o joguem em modo sincronizado. Alguma sugestÃ£o?

Modified : /trunk/main.c

Revision: 18
Author: inpanzinator
Date: 17:44:52, sexta-feira, 21 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 17
Author: inpanzinator
Date: 17:38:53, sexta-feira, 21 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 16
Author: inpanzinator
Date: 17:36:25, sexta-feira, 21 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 15
Author: inpanzinator
Date: 17:30:34, sexta-feira, 21 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 14
Author: inpanzinator
Date: 17:29:43, sexta-feira, 21 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 13
Author: inpanzinator
Date: 17:25:47, sexta-feira, 21 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 12
Author: inpanzinator
Date: 17:17:04, sexta-feira, 21 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 11
Author: inpanzinator
Date: 14:55:45, sexta-feira, 21 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 10

Author: inpanzinator
Date: 14:52:56, sexta-feira, 21 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 9
Author: inpanzinator
Date: 14:50:49, sexta-feira, 21 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 8
Author: inpanzinator
Date: 14:47:57, sexta-feira, 21 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 7
Author: inpanzinator
Date: 14:42:55, sexta-feira, 21 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe
Modified : /trunk/queue.c
Modified : /trunk/queue.h

Revision: 6
Author: inpanzinator
Date: 14:28:19, sexta-feira, 21 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 5
Author: inpanzinator
Date: 14:19:33, sexta-feira, 21 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 4
Author: inpanzinator
Date: 14:16:05, sexta-feira, 21 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/proj.exe

Revision: 3
Author: inpanzinator
Date: 14:06:04, sexta-feira, 21 de Maio de 2010
Message:

Modified : /trunk/main.c
Modified : /trunk/makefile
Modified : /trunk/proj.exe
Deleted : /trunk/serial.c
Deleted : /trunk/serial.h
Added : /trunk/serie.c
Added : /trunk/serie.h

Revision: 2
Author: inpanzinator
Date: 14:42:11, quinta-feira, 20 de Maio de 2010
Message:

LCOM

Added : /trunk/A.OUT
Added : /trunk/BLA
Added : /trunk/GQueue.c
Added : /trunk/GQueue.h
Added : /trunk/PROJ.7z
Added : /trunk/asm_kbd.asm
Added : /trunk/asprite.h
Added : /trunk/crazyf.TXT
Added : /trunk/ints.c
Added : /trunk/ints.h
Added : /trunk/isr.asm
Added : /trunk/isr.asmBACK
Added : /trunk/kbc.c
Added : /trunk/kbc.h
Added : /trunk/lambada.txt
Added : /trunk/main.c
Added : /trunk/makefile
Added : /trunk/music.c
Added : /trunk/music.h
Added : /trunk/new 2
Added : /trunk/pixmap.h
Added : /trunk/pontuacoes.txt
Added : /trunk/proj.exe
Added : /trunk/queue.c
Added : /trunk/queue.h
Added : /trunk/rtc.c
Added : /trunk/rtc.h
Added : /trunk/rtc.inc
Added : /trunk/rtc_asm.asm
Added : /trunk/rtc_asm.asmBACK
Added : /trunk/serial.c
Added : /trunk/serial.h
Added : /trunk/song.c
Added : /trunk/song.h
Added : /trunk/song_cat.txt
Added : /trunk/sprite.c
Added : /trunk/sprite.h
Added : /trunk/timer.c
Added : /trunk/timer.h

Added : /trunk/util.c
Added : /trunk/util.h
Added : /trunk/utypes.h
Added : /trunk/video-text.c
Added : /trunk/video-text.h
Added : /trunk/video.c
Added : /trunk/video.h

Revision: 1
Author:
Date: 14:28:24, quinta-feira, 20 de Maio de 2010
Message:
Initial directory structure.

Added : /branches
Added : /tags
Added : /trunk
Added : /wiki

Anexo D – Makefile

```
# Makefile

#Compiler C flags
CFLAGS=-Wall
#Compiler NASM flags
NFLAGS=-t -f coff
#Compiler C
CC=gcc
#Compiler NASM
NASM= nasm
#Lib
LIBS= -llcom
#Objectos
OBJECTS= video.o main.o sprite.o kbc.o asm_kbd.o timer.o music.o ints.o isr.o queue.o
rtc_asm.o rtc.o gqueue.o song.o serie.o
#NOME DO PROJECTO
LIGHT_CYCLES= lightcycles.exe

all: $(LIGHT_CYCLES)

$(LIGHT_CYCLES): $(OBJECTS) liblcom.a
	$(CC) -Wall $(OBJECTS) -L. $(LIBS) -o $(LIGHT_CYCLES)

%.o: %.c %.h
	@echo A compilar o ficheiro: $<
	@$(CC) $(CFLAGS) -c $<

%.o: %.asm
	@echo A compilar o ficheiro: $<
	@$(NASM) $(NFLAGS) $<

liblcom.a: $(OBJECTS)
	@ar -cr liblcom.a $(OBJECTS)

clean:
	-rm -rf *.o *.exe

rebuild: clean all
```