Manual Técnico do ZX Spectrum para Programadores MSX

INTRODUÇÃO

Há alguns bons anos os programadores dos computadores da linha MSX receberam um presente: o **MSX Top Secret**, escrito por Edison Pires. O MSX Top Secret é um guia que condensa grande parte do conhecimento existente sobre a arquitetura MSX e seus periféricos. Alguns anos depois, foi lançada uma edição revisada e ampliada, com o nome de MSX Top Secret II. Como desenvolvedor para a linha MSX, não posso deixar de relatar que a maioria dos projetos que desenvolvi não teriam visto a luz do dia sem a ajuda de primorosa obra.

Quando decidi portar um jogo de ZX Spectrum para MSX (Ghosts'n Goblins, no ano de 2007), precisei conhecer melhor o Spectrum; mais do que seria suficiente para programar algo em BASIC. Na realidade, era necessário um conhecimento tão profundo do Spectrum quanto o que eu já tinha de MSX para realizar a conversão com sucesso, pois era necessário fazer um mapeamento de funções do Spectrum para as funções do MSX.

O primeiro passo foi buscar bibliografia pela Internet e, qual não foi minha surpresa ao descobrir que as informações sobre o Spectrum eram ainda mais difíceis de serem encontradas que as do MSX. Pior que isso: além de difíceis de encontrar, elas ainda eram desencontradas. Como o MSX, o Spectrum teve muitos modelos. Vários modelos introduziram mudanças significativas e, em alguns casos, quebraram a compatibilidade com os modelos anteriores (em especial com relação aos timings de processamento e vídeo). O problema é que a grande maioria das fontes de informação, com suas parcas informações, considerava que o modelo de Spectrum descrito era o único existente e, pior, não informava qual era.

Assim, com uma pausa na idéia do desenvolvimento, foi iniciada a pesquisa e a criação deste livro, cuja primeira versão digital foi usada inicialmente para o desenvolvimento do porte do Ghosts'n Goblins e foi revisada e ampliada diversas vezes durante a conversão deste mesmo jogo e, posteriormente, em 2008, na conversão do SimCity para MSX.

Obviamente que este é um trabalho contínuo, já que a plataforma Spectrum é bastante complexa e tem muitos periféricos a serem descritos. Como as informações são difusas e contraditórias, muitas informações só podem ser corrigidas e preenchidas durante adaptações de jogos e programas que usem os periféricos (quando então seu funcionamento pode ser univocamente determinado pelo código do mesmo).

Apresento a vocês, então, a versão ZX Spectrum do Top Secret. Aproveitem!

Daniel Jorge Caetano
daniel@caetano.eng.br

1. ARQUITETURA GERAL

Cada modelo de Spectrum tem suas especificidades e, mesmo nos modelos "oficiais", há diferenças significativas entre eles. Entretanto, o modelo básico de desenvolvimento é o Spectrum 48K e o Spectrum 128. Houve modelos de menos de 48K, mas pouquíssima coisa foi desenvolvida para estes modelos. A maioria dos softwares de Spectrum são duais, ou seja, podem ser executados tanto no modelo 48 quanto no modelo 128.

A arquitetura do modelo 128 difere muito pouco do 48: basicamente foi adicionada uma memória mapeada; não confunda com o conceito de memória mapeada do MSX, entretanto. A memória mapeada do Spectrum 128 é bastante simples e pouco flexível.

Os outros modelos, como Spectrum +2, +2A e +3, introduzem mudanças maiores e mais complexas, mas não necessariamente vale a pena preocupar-se com elas, pois raros são os programas que usam estes recursos. Mesmo a memória mapeada do Spectrum 128 é usada, em geral, apenas para colocar música nos jogos, isto é, o jogo inteiro roda num Spectrum 48, mas só com efeitos sonoros pelo beeper, e as rotinas de música (replayer e dados) são colocados na memória extendida e usados quando o carregador do jogo detecta que está sendo executado no modelo 128.

A seguir, nos próximos tópicos deste capítulo, serão descritas as generalidades da arquitetura de cada modelo de Spectrum. É importante lembrar que a seção sobre o Spectrum 48 apresenta o básico da arquitetura e cada outra seção apresenta os detalhes que foram sendo acrescentados em cada evolução.

1.1. O Spectrum 48K

O ZX Spectrum possui uma arquitetura muito simples. Na arquitetura 48K há, basicamente, uma CPU Z80 a 3.5MHz, um circuito de vídeo, chamado "ULA", e um gerador de som simples.

A memória do ZX Spectrum é dividida em duas partes: RAM e VRAM, como a do MSX. A diferença é que a RAM é acessada exclusivamente pela CPU e a VRAM pode ser acessada tanto pela CPU quanto pela ULA. Em outras palavras, o ZX Spectrum tem uma espécie de ADVRAM desde sua criação no início da década de 1980.

VRAM	16KB
RAM	32KB

É importante observar, entretanto, que o acesso direto à VRAM por parte da CPU tem um impacto sobre o tempo de processamento. Isso ocorre porque a ULA tem preferência de acesso à memória de vídeo (VRAM) e, caso a CPU precise acessar a VRAM enquanto a ULA está desenhando a tela no monitor, ela será

obrigada a esperar, para que a ULA termine seu uso da memória, e só então poderá realizar o acesso. Essa espera é provocada pela ULA através do pino HALT do Z80. É por esta razão que, no ambiente Spectrum, a VRAM é costumeiramente chamada de "**contended RAM**" enquanto a RAM é chamada de "**uncontended RAM**".

> Essa questão da temporização é importante porque a temporização de código rodando na região da VRAM não é precisa; por esta razão, não é possível executar código que necessite temporização precisa, como leitura de cassete e de tocadores de música, nesta região. Isso tem uma conseqüência no carregamento de jogos grandes (48KB ou mais) a partir de fita: via de regra eles precisam estar comprimidos em menos de 32KB, para serem lidos diretamente na região da RAM "uncontended". Após isso, os programas são descompactados de trás pra frente, num processo bastante confuso para quem está tentando entender o que o carregador faz. É importante ter isso em mente, em especial quando se carrega jogos de Spectrum 128: os binários lidos da fita estarão comprimidos e o carregador irá descomprimí-los.

Existe ainda uma ROM de sistema, de 16KB, contendo o Sinclair BASIC, mas a grande maioria dos programas em linguagem de máquina ignora solemente sua existência. A organização básica da memória desta arquitetura é:

ROM (BASIC)	0000h~3FFFh
VRAM (Screen)	4000h~5AFFh
VRAM	5B00h~7FFF
RAM	8000h~BFFFh
RAM	C000h~FFFFh

Como a ROM está sempre na posição inicial da ROM, um jogo tem duas opções para ser executado:

- Ser executado em IM1, e usar os contadores da rotina de interrupção da BIOS e o leitor de teclado da BIOS para operar (exemplo: Ghosts'n Goblins)
- *** Ser executado em IM2, apontar o registrador I para uma região da RAM (EF, por exemplo) e de EF00 a EFFF conter um endereço (EEEEh, por exemplo), que será o endereço da rotina de interrupção (lembrando que, no Z80, em IM2 o endereço da rotina de interrupção será lido no endereço formado pelo registrador I e o valor lido do barramento (que pode ser qualquer um) no momento em que a interrupção ocorre.

1.2. O Spectrum 128 / +2

Na arquitetura 128 e superiores foram adicionados outros recursos como uma MMU (Memory Management Unit, gerenciador de memória mapeada, acessado pelo Bank Register) simples e um PSG (gerador de som programável - exatamente o mesmo AY-3-8912 do MSX). O clock do Z80 também é ligeiraqmente maior: 3.54MHz (mais próximos aos 3.58MHz do MSX). As portas do PSG são

usadas como uma espécie de serial (capaz de operar um sintetizador compatível com MIDI).

> Este uso diferenciado das portas do PSG no Spectrum conflita com o uso dado ao MSX para estas portas. No MSX elas são usadas como entrada, para joystick, cassete, etc. Assim, quando das conversões, é preciso tomar cuidado especial com o valor do registrador 7 do PSG: no Spectrum os bits 6 e 7 podem receber quaisquer valores, mas no MSX *** o bit 6 deve sempre valer 0 e o bit 7 deve sempre valer 1. Caso essa configuração seja alterada e o joystick seja usado, o resultado pode ser um PSG queimado.

Tanto a RAM como a VRAM foram expandidas para 64KB cada, intercalados. Como o Z80 não é capaz de endereçar linearmente todos os 128KB, o acesso é feito em forma paginada (controlada pela MMU simples). Mais detalhes podem ser vistos na seção sobre Mapeamento de Memória. Vale lembrar que os problemas do "delay" de acesso da CPU à VRAM continuam os mesmos.

A ROM também foi expandida para 32KB nesta arquitetura, também sendo acessada de forma mapeada. Entretanto, somente uma parte dela tem uso de cada vez. Isso ocorre porque 16KB dela é o Sinclair BASIC do Spectrum 48K, enquanto os outros 16KB são o novo Sinclair BASIC do Spectrum 128. Vale lembrar que a ROM do 48K no Spectrum 128 é ligeiramente modificada, por questões de compatibilidade.

BASIC 128	ROM (Banco 0)
BASIC 48	ROM (Banco 1)

Tirando o fato que é possível "trocar" a parte da ROM, RAM e VRAM ativas, o mapeamento de memória do Spectrum 128 é, essencialmente, o mesmo do Spectrum 48K.

1.3. O Spectrum +2A / +3

Na arquitetura +2A e superiores foram adicionados mais recursos para a MMU (acessados por um segundo registrador de controle) e uma ROM maior, agora de 64KB que, na verdade, são 4 blocos de 16KB (apenas um deles ativo por vez). Adicionalmente, o Spectrum +3 possui um drive de 3" e uma porta de impressora. O +2A foi um +3 que ao invés de drive tinha um leitor de cassete (como a versão +2 normal).

Nesta arquitetura, a RAM e a VRAM não ficam mais intercaladas: os primeiros 64KB são RAM e os 64KB superiores são VRAM. O acesso a estas páginas continua o mesmo, através da MMU. Esta mudança fez com que alguns demos de Spectrum 128 (que dependem de timing fixo) deixassem de funcionar nas versões +2A e +3, criando alguns problemas para os usuários.

A ROM expandida contém, em seus 4 blocos, os seguintes elementos: o banco 0 contém o editor e menus do 128, o banco 1 contém o BASIC do 48K, o banco 2 contém o verificador de sintaxe do 128 e o banco 3 contém o +3DOS.

BASIC 128	ROM (Banco 0)
BASIC 48	ROM (Banco 1)
SYNCHK 128	ROM (Banco 2)
+3DOS	ROM (Banco 3)

Tirando o fato que é possível "trocar" a parte da ROM, RAM e VRAM ativas, como no Spectrum 128, o mapeamento de memória dos Spectrum +2A e +3 é, essencialmente, o mesmo do Spectrum 48K.

2. VARIÁVEIS DA BIOS EM MEMÓRIA RAM

Assim como a BIOS do MSX, a BIOS do Spectrum também utiliza uma área da RAM, de 5B00h a 5CB5h, armazenando valores importantes; alguns destes valores são utilizados pelas rotinas de interrupção padrão - como o buffer de teclado, o que os tornam ainda mais importantes, já que alguns jogos se utilizam destes valores para sua operação.

A seguir será feita uma exposição detalhada destas posições de memória e seus valores.

SWAP (5B00h,16)
Valor inicial: ??
Conteúdo: subrotina de paginamento (ZX128).

STOO (5B10h,17)
Valor inicial: ??
Conteúdo: subrotina de paginamento. Chamada em DI e com AF e BC na pilha (ZX128).

YOUNGER (5B21h,9)
Valor inicial: ??
Conteúdo: subrotina de paginamento.

REGNUOY (5B2Ah,16)
Valor inicial: ??
Conteúdo: subrotina de paginamento.

ONERR (5B3Ah,24)
Valor inicial: ??
Conteúdo: subrotina de paginamento.

OLDHL (5B52h,2)
Valor inicial: ??
Conteúdo: amazenamento temporário do registrador HL durante a troca de ROMs.

OLDBC (5B54h,2)
Valor inicial: ??
Conteúdo: amazenamento temporário do registrador BC durante a troca de ROMs.

OLDAF (5B56h,2)
Valor inicial: ??
Conteúdo: amazenamento temporário do registrador AF durante a troca de ROMs.

TARGET (5B58h,2)
Valor inicial: ??
Conteúdo: Endereço da subrotina a executar na ROM 3.

RETADDR (5B5Ah,2)
Valor inicial: ??
Conteúdo: Endereço de retorno na ROM 1.

BANKM (5B5Ch,1)
Valor inicial: 00h
Conteúdo: Cópia do último valor escrito na porta 7FFDh (gerenciamento de memória).

RAMRST (5B5Dh,1)
Valor inicial: RST 8
Conteúdo: Instrução RST 8, usado para a ROM 1 indicar erros antigos à ROM 3.

RAMERR (5B5Eh,1)
Valor inicial: ??
Conteúdo: Número de erro passado da ROM 1 para a ROM 3. Também usado pelo SAVE/LOAD como armazenamento temporário do drive.

BAUD (5B5Fh,2)
Valor inicial: ??
Conteúdo: Período do bit da RS-232, em T States/26. Definido pelo comando FORMAT LINE do BASIC.

SERFL (5B61h,2)
Valor inicial: ??
Conteúdo: Flag do "segundo carctere recebido" e o dado.

COL (5B63h,1)
Valor inicial: ??
Conteúdo: Coluna atual, podendo variar de 1 até a largura atual da tela.

WIDTH (5B64h,1)
Valor inicial: 80
Conteúdo: Largura em colunas do fundo (paper).

TVPARS (5B65h,1)
Valor inicial: ??
Conteúdo: Número de parâmetros "inline" esperados pela RS-232.

FLAGS3 (5B66h,1)

Valor inicial: ??

Conteúdo: Contém vários flags:

Descrição do endereço 5B66h

b7	b6	b5	b4	b3	b2	b1	b0
?	?	DBP	DSK	RP	TE	?	?

TE - *Token Expansion*: indica (1) que os tokens do BASIC devem ser expandidos durante a impressão.

RP - *RS-232 Printing*: indica (1) que a saída da impressão deve ser a RS-232. O valor default é 0, indicando a impressão pela porta Centronics (+2A e +3)

DSK - *DiSK interface present*: indica (1) se a interface de disco está presente (+2A/+3).

DBP - *Drive B Present*: indica (1) que há um drive B presente.

BANK678 (5B67h,1)

Valor inicial: 00h

Conteúdo: Cópia do último valor escrito na porta 1FFDh (gerenciamento de memória do +2A/+3).

XLOC (5B68h,1)

Valor inicial: ??

Conteúdo: Armazena a posição X quando for usado o comando COPY sem expansão.

YLOC (5B69h,1)

Valor inicial: ??

Conteúdo: Armazena a posição Y quando for usado o comando COPY sem expansão.

OLDSP (5B69h,2)

Valor inicial: ??

Conteúdo: Armazena o ponteiro da pilha (SP) quando a TSTACK estiver em uso.

SYNRET (5B6Ch,2)

Valor inicial: ??

Conteúdo: Endereço de retorno para ONERR.

LASTV (5B6Eh,5)

Valor inicial: ??

Conteúdo: Último valor impresso pela calculadora.

RCLINE (5B73h,2)

Valor inicial: ??

Conteúdo: Linha atual sendo renumerada.

RCSTART (5B75h,2)
Valor inicial: 10
Conteúdo: Número de linha inicial para renumeração.

RCSTEP (5B77h,2)
Valor inicial: 10
Conteúdo: Valor do incremento da renumeração de linhas.

LODDRV (5B79h,1)
Valor inicial: ??
Conteúdo: Armazena 'T' se foi dado um comando LOAD, VERIFY ou MERGE para uma fita cassete; caso contrário, armazena 'A', 'B' (drives) ou 'M' (microdrive).

SAVDRV (5B7Ah,1)
Valor inicial: ??
Conteúdo: Armazena 'T' se foi dado um comando SAVE para uma fita cassete; caso contrário, armazena 'A', 'B' (drives) ou 'M' (microdrive).

DUMPLF (5B7Bh,1)
Valor inicial: 9
Conteúdo: Armazena um número que contrla os line feeds no comando 'COPY EXP', em 1/216 avos. Se um dump não estiver sendo apresentado corretamente em uma folha de A4, mude este valor para 8.

STRIP1 (5B7Ch,8)
Valor inicial: ??
Conteúdo: Armazena um bitmap.

STRIP2 (5B84h,8 / 5B8Bh,8)
Valor inicial: ??
Conteúdo: Armazena dois bitmaps.

TSTACK (5BFFh,115)
Valor inicial: ??
Conteúdo: Pilha temporária, cresce para baixo a partir deste entederço. Usada quando a página 7 da RAM é colocada no topo da memória (uso do editor ou chamada do +3DOS). Pode crescer sobreponto STRIP1 e 2 se necssário, mas os 115 bytes padrão são suficientes para as chamadas ao +3DOS pelo BASIC.

KSTATE (5C00h,8)
Valor inicial: ??
Conteúdo: Usado na leitura do teclado.

LASTK (5C08h,1)
Valor inicial: ??
Conteúdo: Armazena a tecla mais recente apertada.

REPDEL (5C09h,1)
Valor inicial: 35
Conteúdo: Número de interrupções que uma tecla precisa ficar apertada antes que comece a repetir.

REPPER (5C0Ah,1)
Valor inicial: 5
Conteúdo: Número de interrupções entre repetições sucessivas de uma tecla.

DEFADD (5C0Bh,2)
Valor inicial: 0
Conteúdo: Endereço dos argumentos da função definida pelo usuário, quando uma está sendo executada. Caso contrário, contém 0.

KDATA (5C0Dh,1)
Valor inicial: ??
Conteúdo: Armazena o segundo byte dos controles de cor digitados pelo teclado.

TVDATA (5C0Eh,2)
Valor inicial: ??
Conteúdo: Armazena os bytes de controle de cor, posição e tabs que são enviados à TV.

STRMS (5C10h,38)
Valor inicial: ??
Conteúdo: Endereços dos canais associados aos streams.

CHARS (5C36h,2)
Valor inicial: ??
Conteúdo: Endereço da tabela de caracteres menos 256. Normalmente aponta para a ROM.

RASP (5C38h,1)
Valor inicial: ??
Conteúdo: Comprimento do sinal de aviso sonoro.

PIP (5C39h,1)
Valor inicial: ??
Conteúdo: Comprimento do som do click do teclado.

ERRNR (5C3Ah,1)
Valor inicial: ??
Conteúdo: 1 menos o código de erro informado. Inicia em 255 (para -1). Em geral, IY aponta para este endereço.

FLAGS (5C3Bh,1)
Valor inicial: ??
Conteúdo: Flags variados usados pelo BASIC.

Descrição do endereço 5C3Bh

b7	b6	b5	b4	b3	b2	b1	b0
?	?	NK	?	?	?	?	?

NK - *New Key*: indica (1) quando uma nova tecla foi adicionada ao buffer (atualizado pela rotina de interrupção).

TVFLAG (5C3Ch,1)
Valor inicial: ??
Conteúdo: Flags relacionados com a TV.

ERRSP (5C3Dh,2)
Valor inicial: ??
Conteúdo: Endereço do item na pilha da máquina a ser usado como retorno da rotina de erro.

LISTSP (5C3Fh,2)
Valor inicial: ??
Conteúdo: Endereço do retorno da listagem automática.

MODES (5C41h,1)
Valor inicial: ??
Conteúdo: Especifica cursor 'K', 'L', 'C', 'E' ou 'G'.

NEWPC (5C42h,2)
Valor inicial: ??
Conteúdo: Endereço da linha para a qual a execução pulará.

NSPPC (5C44h,1)
Valor inicial: ??
Conteúdo: Número da instrução na linha para a qual a execução pulará.

PPC (5C45h,2)
Valor inicial: ??
Conteúdo: Número da linha da instrução em execução.

SUBPPC (5C47h,1)
Valor inicial: ??
Conteúdo: Número dentro da linha da instrução em execução.

BORDCR (5C48h,1)
Valor inicial: ??
Conteúdo: Cor da borda multiplicada por 8 (deslocada 3 bits). Também contém os atributos normalmente usados para a metade inferior da tela.

EPPC (5C49h,2)
Valor inicial: ??
Conteúdo: Número da linha atual (com o cursor de programa).

VARS (5C4Bh,2)
Valor inicial: ??
Conteúdo: Endereço das variáveis.

DEST (5C4Dh,2)
Valor inicial: ??
Conteúdo: Endereço da variável que está recebendo um dado.

CHANS (5C4Fh,2)
Valor inicial: ??
Conteúdo: Endereço do canal de dados.

CURCHL (5C51h,2)
Valor inicial: ??
Conteúdo: Endereço da informação sendo usada para entrada ou saída.

PROG (5C53h,2)
Valor inicial: ??
Conteúdo: Endereço do programa BASIC.

NXTLIN (5C55h,2)
Valor inicial: ??
Conteúdo: Endereço da próxima linha do programa.

DATAADD (5C57h,2)
Valor inicial: ??
Conteúdo: Endereço do terminador do último item DATA.

ELINE (5C59h,2)
Valor inicial: ??
Conteúdo: Endereço do comando sendo digitado.

KCUR (5C5Bh,2)
Valor inicial: ??
Conteúdo: Endereço do cursor.

CHADD (5C5Dh,2)
Valor inicial: ??
Conteúdo: Endereço do próximo caractere a ser interpretado - o caractere depois do argumento do PEEK, ou um NEWLINE ao fim de um POKE.

XPTR (5C5Fh,2)
Valor inicial: ??
Conteúdo: Endereço do caractere após o marcador [].

WORKSP (5C61h,2)
Valor inicial: ??
Conteúdo: Endereço da área de trabalho temporária.

WORKSP (5C63h,2)
Valor inicial: ??
Conteúdo: Endereço da posição mais baixa da pilha (da calculadora?).

STKEND (5C65h,2)
Valor inicial: ??
Conteúdo: Endereço do início do espaço livre.

BREG (5C67h,1)
Valor inicial: ??
Conteúdo: Registrador B da calculadora.

MEM (5C68h,2)

Valor inicial: ??

Conteúdo: Endereço da área usada pela memória da calculadora (normalmente é a área MEMBOT, mas nem sempre).

FLAGS2 (5C6Ah,1)

Valor inicial: ??

Conteúdo: Mais flags de sistema.

Descrição do endereço 5C6Ah

b7	b6	b5	b4	b3	b2	b1	b0
?	?	?	?	CS	?	?	?

CS - *Caps Shift*: indica (1) quando a tecla Caps Shift estiver apertada ou o Caps Lock estiver acionado.

DFSZ (5C6Bh,1)

Valor inicial: ??

Conteúdo: Número de linhas (incluindo uma linha em branco) na parte inferior da tela.

STOP (5C6Ch,2)

Valor inicial: ??

Conteúdo: O número da linha de programa do topo em listagens automáticas.

OLDPPC (5C6Eh,2)

Valor inicial: ??

Conteúdo: Número da linha para o qual o comando CONTINUE deverá pular.

OSPPC (5C70h,1)

Valor inicial: ??

Conteúdo: Numero da instrução dentro da linha para a qual o comando CONTINUE deverá pular.

FLAGX (5C71h,1)

Valor inicial: ??

Conteúdo: Mais flags.

STRLEN (5C72h,2)

Valor inicial: ??

Conteúdo: Comprimento da string destino em uma atribuição.

TADDR (5C74h,2)

Valor inicial: ??

Conteúdo: Endereço do próximo item na tabela de sintaxe.

SEED (5C76h,2)

Valor inicial: ??

Conteúdo: Indica a semente do comando RND. Definida pela instrução RANDOMIZE.

FRAMES (5C78h,3)
Valor inicial: ??
Conteúdo: Contador de frames, atualizado a cada 20ns (50Hz). Similar ao JIFFY do MSX, mas com 3 bytes. No formato LSB.

UDG (5C7Bh,2)
Valor inicial: ??
Conteúdo: Endereço do primeiro caractere gráfico definido pelo usuário. Normalmente é usado para liberar espaço na RAM, diminuindo o número de caracteres definidos pelo usuário.

COORDS (5C7Dh,1)
Valor inicial: ??
Conteúdo: Coordenada X do último ponto plotado.

COORDS+1 (5C7Eh,1)
Valor inicial: ??
Conteúdo: Coordenada Y do último ponto plotado.

PPOSN (5C7Fh,1)
Valor inicial: ??
Conteúdo: Número da coluna da impressora de 33 colunas.

PRCC (5C80h,2)
Valor inicial: ??
Conteúdo: Endereço da próxima posição do buffer da impressora ZX em que LPRINT irá escrever. Pode ser um valor entre 5B00~5B1Fh. Não usado no modo ZX128 (a áera do buffer é usada com outros propósitos) ou quando alguns periféricos específicos são ligados.

ECHOE (5C82h,2)
Valor inicial: ??
Conteúdo: Número de uma das 33 colunas e número de uma das 24 linhas (na parte baixa) do fim do buffer de entrada.

DFCC (5C84h,2)
Valor inicial: ??
Conteúdo: Endereço no arquivo de vídeo (display file) da posição para o comando PRINT.

DFCCL (5C86h,2)
Valor inicial: ??
Conteúdo: Endereço no arquivo de vídeo (display file) da posição para o comando PRINT, para a parte inferior da tela.

SPOSN (5C88h,1)
Valor inicial: ??
Conteúdo: Número de uma das 33 colunas como posição para o comando PRINT.

SPOSN+1 (5C89h,1)
Valor inicial: ??
Conteúdo: Número de uma das 24 linhas como posição para o comando PRINT.

SPOSNL (5C8Ah,1)
Valor inicial: ??
Conteúdo: Número de uma das 33 colunas como posição para o comando PRINT, para a parte inferior da tela.

SPOSNL+1 (5C8Bh,1)
Valor inicial: ??
Conteúdo: Número de uma das 24 linhas como posição para o comando PRINT, para a parte inferior da tela.

SCRCT (5C8Ch,1)
Valor inicial: ??
Conteúdo: Contador de rolagem (scroll). Sempre 1 ou mais. Indica quantas rolagens devem ocorrer antes que a pergunta "scroll?" seja apresentada.

ATTRP (5C8Dh,1)
Valor inicial: ??
Conteúdo: Cores atuais permanentes (como definido pelos comandos de cores).

MASKP (5C8Eh,1)
Valor inicial: ??
Conteúdo: Máscara usada para cores permanentes transparentes. Qualquer bit 1 indica que o atributo correspondente não deve ser pego de ATTRP, mas deve ser mantido o que já estiver na tela.

ATTRT (5C8Fh,1)
Valor inicial: ??
Conteúdo: Cores atuais temporárias (como definido por itens de cores).

MASKT (5C90h,1)
Valor inicial: ??
Conteúdo: Máscara usada para cores temporárias transparentes. Qualquer bit 1 indica que o atributo correspondente não deve ser pego de ATTRP, mas deve ser mantido o que já estiver na tela.

PFLAG (5C91h,1)
Valor inicial: ??
Conteúdo: Mais flags.

MEMBOT (5C92h,30)
Valor inicial: ??
Conteúdo: Região de memória da calculadora. Usado para armazenar números que não podem ser colocadas convenientemente na pilha da calculadora.

NMIADD (5CB0h,2)

Valor inicial: ??

Conteúdo: Endereço para rotina NMI do usuário. NOTA: nas máquinas originais, este endereço não funcionava e deve ser considerado como 'não usado'. Programas que usam estes bytes para passagem de parâmetros devem ser modificados.

RAMTOP (5CB2h,2)

Valor inicial: ??

Conteúdo: Endereço da última posição de memória da área de sistema do BASIC.

PRAMT (5CB4h,2)

Valor inicial: ??

Conteúdo: Endereço do último byte da RAM física.

3. MAPEAMENTO DE MEMÓRIA

Diferentemente do MSX, o Spectrum teve uma arquitetura inicial pouco flexível, como a ausência do sistema de slots. Se por um lado isso torna o sistema menos adaptável, por outro lado facilitar muito a programação, nunca sendo necessário fazer as aborrecidas (e propensas a erros) buscas de periféricos em slots. Adicionalmente, a ausência deste mecanismo de slots facilita a adaptação de programas, sendo este um detalhe a menos com que se preocupar.

A arquitetura de memória do Spectrum, entretanto, não permaneceu estática. Assim como o MSX2 ganhou a memória mapeada, também o Spectrum teve sua versão de expansão de memória. Mais uma vez, o sistema de memória expandida do Spectrum é muito mais simples que o do MSX, facilitando a adaptação de softwares de Spectum 128 para MSX com o uso de Memory Mapper. Obviamente, a adaptação de softwares de Spectrum 128 para MSX 1.0 sem expansão de memória é algo significativamente complexo, dado o fato que o MSX1.0 possui, de fábrica, apenas 64KB de RAM.

O fato da expansão de memória do Spectrum 128 ser muito simples e pouco flexível causa um aborrecimento, entretanto. Nas versões subsequente foram sendo realizadas mudanças na arquitetura, para flexibilizar a manipulação de memória extendida. Isso faz com que softwares de Spectrum 48, 128, +2A, +3 e posteriores (os modelos não oficiais russos, com mais de 128K) todos gerenciem memória de maneira distinta e, em geral, tenham múltiplos sistemas de gerenciamento, para permitir o funcionamento em várias versões de Spectrum. Isso deixa o código cheio de desvios, comparações e complexidades que obrigam o programador a conhecer não só para que versão específica um programa foi feito, mas também como funciona a arquitetura de memória de cada modelo.

3.1. Mapeamento do Spectrum 48K

No Spectrum 48K existe apenas 48KB de memória RAM e 16KB de memória ROM (BASIC), memória esta alocada estaticamente de acordo com o seguinte mapa:

ROM (BASIC)	0000h~3FFFh
VRAM (Screen)	4000h~5AFFh
VRAM	5B00h~7FFF
RAM	8000h~BFFFh
RAM	C000h~FFFFh

3.2. Mapeamento do Spectrum 128, +2A e +3

A partir do Spectrum 128, a memória do Spectrum passou a ser paginada e mapeada; isto é, o conteúdo de cada posição da RAM depende de uma configuração, similar ao que ocorre com os registros de mapper do MSX (no caso do MSX, selecionados pelas portas FCh a FFh). A memória é dividida em 4 regiões de 16KB, denominadas frames, como os frames (ou páginas) do MSX:

0000h ~3FFFh	=> Frame 0
4000h ~7FFFh	=> Frame 1
8000h ~BFFFh	=> Frame 2
C000h ~FFFFh	=> Frame 3

A memória de 128KB do Spectrum 128 é dividida em 8 páginas de 16KB cada, que podem ser alocadas dos frames 1 a 3. Entretanto, esta alocação não é livre como no MSX, como será visto mais adiante.

Vale lembrar que os 128KB são, na realidade, uma mistura de 64KB de RAM com 64KB de VRAM. No Spectrum 128 e +2, a RAM é constituída pelos blocos pares (páginas 0, 2, 4 e 6) e a VRAM é constituída pelos blocos ímpares (páginas 1, 3, 5 e 7). Grosso modo, pode-se estabelecer a divisão dos 128KB no Spectrum 128/+2 da seguinte forma:

Shadow Screen	Página 7 - VRAM
Livre	Página 6 - RAM
Screen	Página 5 - VRAM
Livre	Página 4 - RAM
Livre	Página 3 - VRAM
Livre	Página 2 - RAM
Livre	Página 1 - VRAM
Livre	Página 0 - RAM

No Spectrum +2A e +3 houve uma reorganização na divisão entre RAM e VRAM: os 128KB passaram a ser, na realidade, dois blocos de 64KB de RAM e VRAM, sendo o bloco inferior (páginas 0 a 3) a RAM e o bloco superior (páginas 4 a 7) a VRAM. Grosso modo, pode-se estabelecer a divisão dos 128KB no Spectrum +2A/+3 da seguinte forma:

Shadow Screen	Página 7 - VRAM
Livre	Página 6 - VRAM
Screen	Página 5 - VRAM
Livre	Página 4 - VRAM
Livre	Página 3 - RAM
Livre	Página 2 - RAM
Livre	Página 1 - RAM
Livre	Página 0 - RAM

Esta mudança é importante porque a RAM é ligada apenas à CPU, mas e VRAM é ligada tanto à ULA quanto à CPU. Isso faz com que o processador enxergue tanto a RAM quanto a VRAM normalmente; a arquitetura de vídeo do Spectrum é uma espécie de ADVRAM.

Entretanto, ainda que a CPU enxergue a VRAM como RAM normal, há uma razão para diferenciar os bancos que são de RAM ou VRAM: a velocidade de acesso da CPU à VRAM pode ser bastante prejudicada, ou seja, mais lenta do que a velocidade de acesso à RAM normal. Isso ocorre porque que a ULA tem prioridade de acesso à VRAM, o que sempre obriga a CPU a "esperar" até que a ULA termine seu trabalho para então poder acessar a VRAM. Se a ULA estiver usando a VRAM e a CPU tentar acessá-la, o sinal HALT da CPU será acionado, fazendo com que a CPU tenha de esperar até o barramento da VRAM estar livre. Mais detalhes sobre este mecanismo na seção que trata do vídeo do Spectrum e da ULA.

3.3. Registradores de Configuração da Memória (128 ~+3)

A partir do Spectrum 128, a configuração principal de qual página estará em qual frame pode ser feita pela porta de I/O **7FFDh**, chamada de **Bank Register**, que é responsável pelo mapeamento da memória. Esta porta é, entretanto, apenas de escrita. Sua leitura pode retornar valores aleatórios ou FFh. A BIOS mantém uma cópia do valor deste registrador no endereço **5B5Ch**. Os bits deste registrador são definidos da seguinte forma:

Escrita em 7FFDh

b7	b6	b5	b4	b3	b2	b1	b0
-	-	LC	RS	US	MB2	MB1	MB0

MB0 ~MB1 - *Memory Bank selector*: determina qual a página da RAM (0~7) está no frame 3.

US - *ULA frame Selector*: determina qual página será usada pela ULA para gerar a imagem (US=0 - página 5; US=1 - página 7). Note que ele apenas muda qual página é usada para gerar a imagem do vídeo, mas não altera a página ativa no frame 1 (que é sempre a 5).

RS - *ROM Selector*: determina qual ROM está sendo usada (RS=0 - ZX128K ROM; RS=1 - ZX48K ROM).

LC - *Lock mapper Control*: quando setado (1) desliga o controle de mapeamento, fixando o mapeamento configurado. Existente para compatibilidade com programas antigos, que poderiam escrever lixo nesta porta. Sendo assim, esse bit é ativado ao entrar no modo 48K. Depois de ativado, só pode ser desativado através de um reset do equipamento.

Considerando os frames como apresentados abaixo, qualquer uma das 8 páginas de RAM (8x16KB = 128KB) podem ser alocadas no frame 3 (C000h a FFFFh); entretanto, o frame 2 (8000h a BFFFh) sempre conterá a página 2 e o

frame 1 (4000h a 7FFFh) sempre conterá a página 5 (VRAM). O frame 0 (0000h a 3FFFh) pode conter qualquer uma das páginas da ROM.

0000h ~3FFFh	=> Frame 0
4000h ~7FFFh	=> Frame 1
8000h ~BFFFh	=> Frame 2
C000h ~FFFFh	=> Frame 3

Assim, as configurações possíveis são, no Spectrum 128:

FFFFh

C000h	Pág 0	Pág 1	Pág 2 *(junto c/ 8000h)*	Pág 3	Pág 4	Pág 5 *(junto c/ 4000h)*	Pág 6	Pág 7 *(shadow screen)*
8000h	Pág 2							
4000h	Pág 5 screen							
0000h	ROM 0 *(BASIC 128)*	ROM 1 *(BASIC 48)*						

Assim que o Spectrum 128 é ligado no modo 128K, a memória é configurada da seguinte forma:

Escrita em 7FFDh

b7	b6	b5	b4	b3	b2	b1	b0
-	-	0	0	0	0	0	0

Ou seja, uma configuração similar à do Spectrum 48K:

ROM 0 (Sp128)	0000h~3FFFh
Página 5 (vídeo)	4000h~7FFFh
Página 2	8000h~BFFFh
Página 0	C000h~FFFFh

3.3.1. Registrador de Configuração Secundário (+2A ~ +3)

No Spectrum +2A e +3 foi criado um sistema de mapeamento adicional, controlado pela porta **1FFDh**. Este sistema foi criado para permitir o funcionameno do +3DOS (compatibilidade com CP/M) e também para que o BASIC fosse expandido, tornando-se mais amigável. Obviamente, os modos adicionais também podem ser usados pelo programador, embora seja incomum que algum programa faça uso dos mesmos.

Os novos modos basicamente permitem que algumas configurações pré-determinadas de RAM sejam alocadas, além de permitir que os novos dois bancos de ROM dos Spectrum +2A/+3 possam ser alocados. Adicionalmente, o novo registro permite o controle do motor do drive e o strobe da porta de impressora adicionada. A arquitetura de memória é, ainda assim, muito pouco flexível, mas de configuração um tanto confusa.

A porta de I/O **1FFDh** é a responsável pelo controle dos novos mecanismos de memória. Esta porta é, assim como a porta 7FFDh, apenas de escrita. Sua leitura pode retornar valores aleatórios ou FFh. A BIOS mantém uma cópia do valor deste registrador no endereço **5B67h**. Os bits deste registrador são definidos da seguinte forma:

Escrita em 1FFDh

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	PST	DM	RC1	RC0	PM

PM - *Paging Mode*: determina a função dos bit RC0 e RC1. Se PM=0 o sistema está no modo *normal* e se PM=1 o sistema está no modo *special*.

RC0-RC1 - *RAM/ROM Control*: no modo normal, RC0 é ignorado e RC1 constitui o bit alto da seleção de ROM. No modo special, RC0 e RC1 determinam a configuração da RAM (como será visto adiante).

DM - *Disk Motor*: Se configurado com 1, liga o motor do drive. Se configurado com 0, desliga o motor.

PST - *Printer STrobe*: sinal usado para controle da impressora.

Os dois modos de operação (normal e special) existem para permitir total compatibilidade com os sistemas de mapeamento anteriores (modo normal) e permitir os novos modos de configuração (modo special). As documentações oficiais não indicam um bit de "travamento" desta configuração; entretanto, é possível que o bit de travamento da porta 7FFDh também trave esta porta de configuração.

Normal Paging Mode

No sistema de paginamento normal, o controle do mapeamento é quase totalmente feito pela porta 7FFDh; a única função da porta 1FFDh é controlar o bit alto do banco de ROM selecionado. A tabela abaixo indica as possíveis seleções:

1FFDh Bit 2 (RC1)	**7FFDh Bit 4 (RS)**	**Banco de ROM Selecionado**
0	0	ROM 0: Spectrum 128 BASIC Editor
0	1	ROM 1: Spectrum 48K BASIC
1	0	ROM 2: Spectrum 128 Syntax Checker
1	1	ROM 3: +3DOS

Special Paging Mode

No modo de paginamento special, a configuração de páginas feita pela porta 7FFDh é ignorada, sendo a configuração definida então pelos bits RC0 e RC1 da porta 1FFD, de acordo com a tabela a seguir (RAM em todos os frames):

FFFFh	RC0 = 0 RC1 = 0	RC0 = 0 RC1 = 1	RC0 = 1 RC1 = 0	RC0 = 1 RC1 = 1
C000h	Pág 3	Pág 7 (Shadow Screen)	Pág 3	Pág 3
8000h	Pág 2	Pág 6	Pág 6	Pág 6
4000h	Pág 1	Pág 5 (Screen)	Pág 5 (Screen)	Pág 7 (Shadow Screen)
0000h	Pág 0	Pág 4	Pág 4	Pág 4

Assim que o Spectrum +2A/+3 é ligado no modo 48K, a memória é configurada da seguinte forma:

Porta 7FFDh

b7	b6	b5	b4	b3	b2	b1	b0
-	-	0	0	0	0	0	0

Porta 1FFDh

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	0	0	0	0	0

Ou seja, uma configuração similar à do Spectrum 48K, idêntica à do Spectrum 128:

ROM 0 (Sp128)	0000h~3FFFh
Página 5 (vídeo)	4000h~7FFFh
Página 2	8000h~BFFFh
Página 0	C000h~FFFFh

3.4. Mudança de Páginas de Memória

A mudança de páginas de memória pode ser feita simplesmente escrevendo no registrador 7FFDh (e no 1FFDh, se for o caso). Entretanto, alguns cuidados precisam ser tomados.

O primeiro cuidado básico é desabilitar as interrupções e o segundo é colocar a pilha (SP) em uma área que não será paginada. Caso o sistema de interrupção padrão (da ROM) vá ser usado, deve-se atualizar o valor do endereço 5B5Ch da RAM com a cópia do valor escrito no registrador 7FFDh (já que a rotina de interrupção da BIOS usa este valor), além de atualizar o valor 5B67h da RAM com a cópia do valor escrito no 1FFDh, se for o caso.

Um código comum para mudar bancos no Spectrum 128 seria o seguinte:

```
LD      A, (05B5Ch)         ; Pega o valor anterior do controlador de memória
AND     11111000b           ; Apaga banco anterior
OR      4                   ; Seleciona banco 4
LD      BC,07FFDh           ; Aponta porta do controlador de memória
DI                          ; Desliga IRQ
LD      (05B5Ch),A          ; Atualiza registro em memória do ctrl. de memória
OUT     (C), A              ; Comanda o controlador de memória
EI                          ; Religa as interrupções.
```

O princípio com esse registrador será sempre "modifique só o que precisa modificar".

4. O VÍDEO DO SPECRUM (ULA)

A ULA é capaz de gerar, basicamente, um display de caracgteres de 24 linhas por 32 colunas, sendo que cada caractere de 8x8 pode ter uma cor de frente (INK) e uma cor de fundo (PAPER). A ULA permite ainda que se defina a cor da borda (BORDER), que é a cor que será usada para pintar a região da tela não coberta pela imagem definida na VRAM.

A ULA é simplesmente um dispositivo capaz de transformar os dados da memória em uma saída de vídeo adequada, então seu funcionamento não é muito importante. Entretanto, é interessante saber que, enquanto a ULA desenha na tela, o acesso ao banco da VRAM é mais lento, fazendo com que o desempenho do equipamento caia.

A ULA tem um registrador de controle, que pode ser usado para definir a cor da borda. Este controle é feito escrevendo nos 3 bits mais baixos da porta xxFEh ou FFFEh (qualquer porta par, na verdade):

Escrita em xxFEh

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	G	R	B

As cores são formadas pela ativação ou desativação dos bits, como pode ser visto na tabela da seção 4.1.

NOTA: A leitura de xxFEh não produz um resultado claro, visto que dependendo do byte alto da porta (o xx) um periférico diferente será acionado. Leituras da porta 0FEh aparentemente diretas, com IN A,(0FEh), em geral são leituras de alguma outra porta: observe o valor em A antes. Exemplo:

```
LD    A,0FDh          ; Lê dado da porta 0FDFEh em A
IN    A,(0FEh)
```

A ULA tem apenas um modo de operação de vídeo, que será descrito a seguir.

4.1. Modo de Operação de Vídeo

A ULA basicamente transforma um bitmap (mapa de bits) de 256 bits por 192 bits em uma imagem de 256 colunas por 192 linhas. Este mapa de bits fornece quais pontos estão acesos (cor INK) e quais estão apagados (cor PAPER). É bastante similar com o conceito usado para construir cada padrão da pattern table do MSX... Mas no spectrum **não há padrões**, o bitmap é **linear**. Isso significa que ao contrário do MSX, onde 32 bytes seqüenciais determinam 32/8 = 4 padrões (blocos de 8x8), no Spectrum 32 bytes seqüenciais determinam uma linha inteira da imagem (32*8 = 256 pixels em linha).

Para saber qual é a cor que deve ser impressa em cada um deles, existe uma tabela que indica a cor INK e a cor PAPER. Entretanto, a resolução desta

tabela é de apenas 32x24, significando que cada grupo de 8x8 pixels compartilha a mesma cor de frente e fundo. Mais uma vez, é um conceito similar ao da color table da Screen 2 do MSX, com a diferença que a color table da Screen 2 no MSX tem resolução de 32x192 (ou seja, cada linha de um bloco de 8x8 pode ter seu próprio conjunto de cores INK e PAPER).

A cor de cada bloco 8x8 do spectrum é definida por um byte. Este byte é composto por 3 bits da INK, 3 bits da PAPER e dois bits de controle, um deles indicando que aq cor INK deve ser mais clara e o outro indica que o blink está ativado naquele bloco. Estas informações são organizadas da seguinte forma em um byte:

b7	b6	b5	b4	b3	b2	b1	b0
BLK	I	G	R	B	G	R	B
		PAPER			INK		

R,G,B - *Red, Green, Blue*: Bits usados para construir as cores de frente e fundo, combinando níveis de intensidade de vermelho (R), verde (G) e azul (B). Esta definição, é claro, é distinta para o INK e o PAPER.

I - *Intensidade*: determina qual a intensidade da cor INK **e** PAPER. Se for 0, as cores são mais escuras; se for 1, as cores aparecerão mais claras.

BLK - *Blink*: Se estiver ativado (1) indica que o blink está ativado; ou seja, a cada 16 frames a ULA irá inverter os valores de INK e PAPER, fazendo com que o texto fique piscando normal e em vídeo reverso.

As cores possíveis no Spectrum são:

Cor	G	R	B	Nome
0	0	0	0	Preto
1	0	0	1	Azul
2	0	1	0	Vermelho
3	0	1	1	Magenta
4	1	0	0	Verde
5	1	0	1	Ciano
6	1	1	0	Amarelo
7	1	1	1	Cinza

Com o bit de intensidade ligado, as cores ficam mais claras, com excessão do preto (continua preto). O Cinza, por sua vez, com o bit de intensidade ligado se torna branco.

Vale lembrar que a cor da borda é definida nos 3 bits mais baixos (b0~b2) da porta FEh (escrita).

4.1.1. Organização da VRAM

A VRAM fica no Frame 1 da memória, ou seja, de 4000h a 7FFFh; nem toda esta região é usada, entretanto, para a construção da tela. Exatamente 6912 bytes são usados, sendo 6144 bytes para o bitmap (256 bits/linha * 192 linhas = 32 bytes/linha * 192 linhas = 6144 bytes) e 768 bytes para a tabela de cores (32 blocos/linha * 24 linhas = 768 blocos, sendo 1 byte para cada bloco).

Assim, o mapa de bits da imagem começa na posição 4000h e vai até a posição 57FFh (6144 = 1800h). Entretanto, a distribuição dos pixels não é exatamente linear. A tela é dividida em três blocos verticais: um da linha 0 a 63 (iniciando em 4000h), um da linha 64 a 127 (iniciando em 4800h) e outro da linha 128 a 191 (iniciando em 5000h).

Dentro de cada um destes blocos, temos conjuntos de 32 bytes seqüenciais, formando as linhas de 256 pixels (8 pixels por byte); entretanto, as linhas **não** estão em ordem. A ordem das mesmas na memória é: **0, 8, 16, 24, 32, 40, 48, 56,** 1, 9, 17, 25, 33, 41, 49, 57, **2, 10, 18, 26, 34, 42, 50, 58,** 3, 11, 19, 27, 35, 43, 51, 59, **4, 12, 20, 28, 36, 44, 52, 60,** 5, 13, 21, 29, 37, 45, 53, 61, **6, 14, 22, 30, 38, 46, 54, 62**, 7, 15, 23, 31, 39, 47, 55, 63. A representação é análoga para o segundo e o terceiro blocos.

Esquematicamente (em pixels):

Lin\Col	0	1	2	...	254	255
0	4000h (b7)	4000h (b6)	4000h (b5)	...	401Fh (b1)	401Fh (b0)
1	4100h (b7)	4100h (b6)	4100h (b5)	...	411Fh (b1)	411Fh (b0)
2	4200h (b7)	4200h (b6)	4200h (b5)	...	421Fh (b1)	421Fh (b0)
3	4300h (b7)	4300h (b6)	4300h (b5)	...	431Fh (b1)	431Fh (b0)
4	4400h (b7)	4400h (b6)	4400h (b5)	...	441Fh (b1)	441Fh (b0)
5	4500h (b7)	4500h (b6)	4500h (b5)	...	451Fh (b1)	451Fh (b0)
6	4600h (b7)	4600h (b6)	4600h (b5)	...	461Fh (b1)	461Fh (b0)
7	4700h (b7)	4700h (b6)	4700h (b5)	...	471Fh (b1)	471Fh (b0)
8	4020h (b7)	4020h (b6)	4020h (b5)	...	403Fh (b1)	403Fh (b0)
9	4120h (b7)	4120h (b6)	4120h (b5)	...	413Fh (b1)	413Fh (b0)
10	4220h (b7)	4220h (b6)	4220h (b5)	...	423Fh (b1)	423Fh (b0)
11	4320h (b7)	4320h (b6)	4320h (b5)	...	433Fh (b1)	433Fh (b0)

...	...	...	...	...	...	...
190	56E0h (b7)	56E0h (b6)	56E0h (b5)	...	56FFh (b1)	56FFh (b0)
191	57E0h (b7)	57E0h (b6)	57E0h (b5)	...	57FFh (b1)	57FFh (b0)

O mapa de cores vai de 5800h até 5AFFh (768 = 300h) e é bem mais simples que mapa de bits da imagem. Esquematicamente (em blocos de 8x8 pixels):

Lin\Col	0	1	2	...	30	31
0	5800h	5801h	5802h	...	581Eh	581Fh
1	5820h	5821h	5822h	...	583Eh	583Fh
...	...	...	...	...	...	...
22	5AC0h	5AC1h	5AC2h	...	5ADEh	5ADFh
23	5AE0h	5AE1h	5AE2h	...	5AFEh	5AFFh

4.2. Geração de Interrupções

As interrupções (IRQ, mascarável, em que o Z80 pula sua execução para 0038h) são geradas pela ULA, sendo disparada uma vez a cada frame (50Hz). Muitos programas usam a IRQ para sincronizar o jogo com o display de quadros, mantendo a cadência do jogo constante. Outros usam para sincronizar efeitos visuais; para isso é importante saber o tempo que se passa para a impressão de cada linha.

É importante lembrar que a temporização não será precisa, pois as escritas (assim como as leituras) à porta xxFEh dependem da resposta da ULA e, portanto, podem causar um "halt" do Z80, Assim, os acessos a esta porta podem ser, ocasionalmente, mais lentos que o normal. Além disso, ao "contar ciclos" é importante não esquecer da rotina de "setup" da interrupção na contagem do tempo, já que ela consome alguns (vários) ciclos. Outra consideração é que uma interrupção não pode ocorrer no meio de uma instrução, o que pode fazer com que o tempo de início da rotina de interrupção varie um pouco. Finalmente, os delays que as intruções sofrem dependem do ciclo T em que a operação de I/O (memória ou porta) ocorre, e este momento não é fixo, variando de acordo com a instrução do Z80.

Spectrum 48K

No Spectrum 48K, cada linha da tela demora 224 estados T para ser impressa. É sabido que do momento em que a IRQ ocorre até a impressão do primeiro byte da tela (4000h) existe um intervalo de pelo menos 64 linhas de borda, ou seja, 14336 estados T. É neste ponto em que começam a ocorrer os delays aos acessos do Z80 por conta da ULA.

Depois disso tem-se a impressão de 192 linhas + bordas, seguidas pelas 56 linhas de borda finais da tela. Com isso, um frame gasta em torno de

(64+192+56)*224 = 69888 estados T, o que significa que a interrupção de 50Hz ocorre, na verdade, a cada 3,5MHz/69888 = 50,08Hz no Spectrum 48K.

É importante lembrar que, no MSX, quem gera a interrupção é o VDP. Considerando que o MSX, no Brasil e Japão, roda a 60Hz, a temporização do Spectrum pode ser bastante difícil de simular, dificultando adaptações de softwares que sejam muito amarrados à temporização.

Como complementação à informação do Spectrum 48K, os 224 estados T da linha são gastos da seguinte forma: 128 estados T para a impressão dos 256 pixels da linha em si, 24 estados T da borda, 48 estados T do retraço horizontal, e mais 24 estados T para a borda da linha seguinte. Em outras palavras isso significa que são gastos 4 estados T para cada 8 pixels, valor usado por alguns jogos para calcular o momento em que a cor da borda deve ser mudada para conseguir alguns efeitos.

Spectrum 128

No Spectrum 128, cada linha da tela demora 228 estados T para ser impressa. É sabido que do momento em que a IRQ ocorre até a impressão do primeiro byte da tela (4000h) existe um intervalo de pelo menos 63 linhas de borda, ou seja, 14364 estados T. É neste ponto em que começam a ocorrer os delays aos acessos do Z80 por conta da ULA.

Depois disso tem-se a impressão de 192 linhas + bordas, seguidas pelas 56 linhas de borda finais da tela. Com isso, um frame gasta em torno de (63+192+56)*228 = 70908 estados T, o que significa que a interrupção de 50Hz ocorre, na verdade, a cada 3,54MHz/70908 = 50,01Hz no Spectrum 128 (comparados aos 50.08 do Specturm 48K).

Pelas mesmas razões já citadas, a temporização do Spectrum 128 também pode ser bastante difícil de simular, dificultando adaptações de softwares que sejam muito amarrados à temporização.

4.3 Delays na CPU Causados pela ULA

Como já foi dito diversas vezes, quando a CPU tentar ler/escrever na porta xxFEh ou ler/escrever na região da VRAM (4000h a 7FFFh no spectrum 48K e páginas 4 a 7 no Spectrum 128, **mesmo quando alocadas no frame 3**, de C000h a FFFFh) poderá haver um delay: se a ULA estiver usando a VRAM, ela irá causar um "halt" na CPU, fazendo-o esperar até que, momentaneamente, a ULA deixe de usar o barramento da VRAM.

Este delay não é fixo, em geral é até a ULA terminar de imprimir um conjunto de bytes. Para exemplificar, segue um exemplo de acesso à memória. Se, em um Spectrum 48K, a CPU tentar acessar a VRAM no ciclo 14335 após a interrupção (um antes do primeiro byte da tela ser desenhado), o Z80 será "congelado" por 6 ciclos T inteiros. Se tentar acessar no ciclo 14336 após a interrupção, será "congelado" por 5 ciclos T inteiros... e assim por diante, como pode ser visualizado abaixo:

Ciclos T 48K	Ciclos T 128/+2	Atraso	Ciclos T +2A/+3	Atraso
14335	14361	6	14365	1
14336	14362	5	14366	0
14337	14363	4	14367	7
14338	14364	3	14368	6
14339	14365	2	14369	5
14340	14376	1	14370	4
14341	14377	0	14371	3
14342	14378	0	14372	2
14343	14379	6	14373	1
14344	14380	5	14374	0
14345	14381	4	14375	7
14346	14382	3	14376	6
14347	14383	2	14377	5
14348	14384	1	14378	4
14349	14385	0	14379	3
14350	14386	0	14380	2

No ciclo 14463, quando começa o desenho da borda final da primeira linha, há 96 ciclos em que não ocorrem atrasos. No ciclo 14559, onde se inicia o desenho da próxima linha, repete-se o esquema de "halt" do Z80.

Os efeitos para o Spectrum 128 são similares, mas os valores em ciclos T mudam (como indicado na tabel), já que o Z80 do 128 roda a uma velocidade ligeiramente maior.

No Spectrum +2A/+3 os efeitos são bem diferentes, por causa da mudança no hardware, não ficando a diferença por conta apenas do início do delay.

4.3.1 Delays no I/O do Z80 Causados pela ULA

No caso de acessos às portas de I/O, a ULA irá responder sempre que o bit mais baixo for zerado (portas pares), o que faz com que possa existir delay nestes acessos. Em outras palavras, o acesso à porta FEh (xxFEh) sofre delay, **menos** nos Spectrum +2A/+3, em que o hardware foi modificado e a escrita nesta porta não causa mais delays.

Além disso, se as portas estiverem na faixa de números entre 4000h e 7FFFh, a ULA confundirá este valor (colocado no barramento de endereços) como uma tentativa de acesso à VRAM, e introduzirá o delay da mesma forma que ocorre com acessos a VRAM.

É importante notar que o delay **não** ocorre quando o byte alto do endereço da porta estiver entre C0h e FFh, **mesmo que** exista VRAM alocada para o Frame 3 (de C000h a FFFFh). A tabela abaixo resume as situações em que ocorrem delay:

Byte alto da Porta entre 40h ~7Fh?	Bit 0 da Porta	Há Delay ?
Não	0	Sim
Não	1	Não
Sim	0	Sim
Sim	1	Sim

4.4. Efeitos Colaterais Interessantes da ULA

4.4.1. Identificando se a ULA está atualizando a tela

Um dado interessante é que qualquer leitura a uma porta não definida (usualmente FFh ou qualquer uma terminada em FFh mas fora da faixa 40FFh ~ 7FFFh), em alguns modelos de Spectrum, irá retornar FFh sempre que a ULA estiver desenhando bordas (ou seja, não está lendo a VRAM).

Por outro lado, se ela estiver usando a VRAM, a leitura desta porta retornará o valor usado pela ULA naquele instante (valor da tabela de atributos), que pode ser qualquer valor - incluindo o FFh.

Este é um bom mecanismo para detectar se a ULA está usando a VRAM, mas só funciona nos Spectrum 48K, 128 e +2, já que era um efeito colateral do circuiro. No Spectrum +2A e +3 o circuito foi aprimorado e não mais é possível saber se a ULA está ou não usando a VRAM.

4.4.2. Efeito de Neve (Snow Effect)

O fato do bus de memória ser compartilhado entre ULA e CPU causa alguns efeitos interessantes. Bem, com a tarefa de facilitar o circuito de refresh, a cada vez que ocorre um ciclo de fetch de instruções, o Z80 coloca no barramento de endereços um valor pelo registrador I (de interrupções) e R (contador de instruções).

Isso não teria nenhuma importância se não fosse o barramento compartilhado. Ocorre que se o registrador I contiver um valor entre 040h e 07Fh, a ULA pensa que a CPU está querendo ler a VRAM. Como isso acontece muito rapidamente (a cada instrução lida), a ULA não consegue lidar com isso, e acaba perdendo um byte, repetindo o byte anteriormente lido para imprimir na tela. Isso gera o chamado "snow effect", que alguns jogos usam para produzir efeito de neve.

5. ENTRADA E SAÍDA - COMUNICAÇÃO COM O USUÁRIO

O Spectrum possui uma grande quantidade de dispositivos de comunicação com o usuário, desde o teclado até joysticks e pistolas. Esta seção descreve alguns destes dispositivos.

5.1. Teclado

O teclado do ZX Spectrum é muito similar ao do MSX, funcionando como uma espécie de matriz de 8 linhas por 5 colunas. A idéia é ler uma porta determinada pela linha que se deseja ler, e o valor retornado será composto por 5 bits, indicando quais teclas daquela linha estão pressionadas.

É definida uma porta para cada linha que se deseja ler (os valores Bn indicam qual bit do endereço lido precisa estar em ZERO para selecionar aquela linha. É possível selecionar mais de uma linha por vez):

FEFEh	Linha 1 (B8)
FDFEh	Linha 2 (B9)
FBFEh	Linha 3 (B10)
F7FEh	Linha 4 (B11)
EFFEh	Linha 5 (B12)
DFFEh	Linha 6 (B13)
BFFEh	Linha 7 (B14)
7FFEh	Linha 8 (B15)

Como se tratam de portas 16 bits, elas podem ser lidas com um comando:

```
LD    BC, 0FDFEh      ; Indica linha 2
IN    A,(C)
```

Entretanto, o mais usual é que os programas trabalhem de uma outra forma:

```
LD    A,0FDh          ; Lê a porta FDFEh em A!
IN    A,(0FEh)
```

O valor lido indica as teclas pressionadas como bits 0. Bits 1 indicam teclas não pressionadas. A tabela a seguir indica quais teclas correspondem a qual linha e coluna.

ATENÇÃO: As portas EFFEh e F7FEh são usadas também para ler os Joysticks do tipo Interface-II! Neste caso, a tabela abaixo não serve!

Porta	b4	b3	b2	b1	b0
FEFEh	V	C	X	Z	SHIFT
FDFEh	G	F	D	S	A
FBFEh	T	R	E	W	Q
F7FEh	5	4	3	2	1
EFFEh	6	7	8	9	0
DFFEh	Y	U	I	O	P
BFFEh	H	J	K	L	ENTER
7FFEh	B	N	M	SYMBOL	SPACE

Quando a tecla SHIFT (ou SYMBOL SHIFT?) estiver ativada, o significado de algumas teclas mudam:

Porta	b4	b3	b2	b1	b0
FEFEh					
FDFEh					
FBFEh					
F7FEh	ESQUERDA	INV VIDEO	TRUE VIDEO	CAPS LOCK	EDIT
EFFEh	BAIXO	CIMA	DIREITA	GRAPH	DELETE
DFFEh					
BFFEh					
7FFEh		,	.	EXTEND MODE	BREAK

Como foi observado anteriormente, é possível selecionar mais de uma linha por vez. Isso significa que é possível ler mais de uma linha por vez. O resultado desta operação é um AND lógico entre os resultados das linhas, se fossem lidas independentemente.

Como um efeito colateral do sistema de matriz, é possível identificar univocamente até duas teclas. Caso mais que duas teclas estejam pressionadas, surge o efeito de "teclas fantasmas", fazendo algumas teclas serem detectadas mesmo sem que estejam ativadas.

5.2. Saída de Som da ULA

A ULA é capaz de emitir áudio. Esta emissão é feita escrevendo no bit 4 (Speaker Out) da porta xxFEh (qualquer porta par), de maneira muito similar à escrita de dados por cassete.

Escrita em xxFEh

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	SO	-	-	-	-

5.3. O Gerador de Sons Programáveis (PSG)

Disponível como um *upgrade* para o Spectrum 48K, este dispositivo de áudio tornou-se padrão no Spectrum 128. O PSG do Spectrum é exatamente o mesmo do MSX, mas suas conexões com a CPU diferem ligeiramente.

O PSG, no MSX, funciona com duas portas: uma para seleção de registrador e outra para leitura/escrita no registrador do PSG. No Spectrum também há duas portas, mas seu comportamento é diferente daquele encontrado no MSX, como é descrito na tabela abaixo:

Porta	Escrita	Leitura
FFFDh	Register Select	Data Read
BFFDh	Data Write	?

5.3.1. A interface RS-232 / MIDI

O PSG possui 3 portas. O ZX Spectrum usa a porta A para comunicação serial. Uma escrita neste registro permite enviar dados; uma leitura neste endereço permite receber dados.

A configuração destas portas (fixa no MSX) é feita pelo registrador 7 do PSG, pelo bit 6. Se o bit 6 estiver igual a 0, indica que a porta está configurada para entrada e quando o bit 6 for 1, indica que a porta está configurada como saída. Assim, dependendo do que se deseja fazer com a porta, é preciso lembrar de configurar a direção da mesma corretamente pelo registrador 7 do PSG.

No caso da comunicação RS-232, os 4 bits mais baixos do registrador 14 serão usados **apenas para saída** e os 4 bits mais altos serão usados **apenas para entrada**, como mostra a figura a seguir:

Escrita no Regristrador 14 do PSG

b7	b6	b5	b4	b3	b2	b1	b0
TXD	DTR	CTS	X	RXD	CTS	X	TXD
IN				OUT			

5.4. ZX Interface I

A ZX Interface I é uma espécie de dispositivo multipropósito, que contém a interface com o MicroDrive, uma interface RS-232 e um sistema de comunicação em rede.

Esta interface usa três portas diferentes e possui um circuito para chavear uma ROM de 8KB de algum dispositivo de comunicação que seja ligado à ela. Esta ROM será ativada sempre que a CPU executar a instrução no endereço 0008h da

ROM-BIOS (rotina de erro) ou 1708h (rotina da função CLOSE #). Ela fica ativa até que a instrução RET no endereço 0700h seja executada.

As 3 portas de controle são:

•**xxE7h**: Esta porta é usada para enviar ou receber dados do microdrive. Quando esta porta é lida ou escrita, a CPU será congelada até que o microdrive tenha terminado de realizar a operação de escrita ou leitura de 8 bits. Caso o cartucho microdrive não esteja formatado ou o motor do microdrive não esteja rodando, o micro irá travar.
•**xxEFh**: Esta é uma porta de controle e tem funções diferentes na escrita e leitura:

Leitura de xxEFh

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	BSY	DTR	GAP	SYN	WRP

BSY - *BuSY*: usado para indicar que o microdrive está ocupado (?).
DTR - *Data Terminal Ready*: Sinal usado pela RS-232 (?).
GAP - *GAP*: Sinal usado pelo microdrive (?).
SYN - *SYNchrony*: Sinal usado pelo microdrive (?).
WRP - *WRite Protect*: Sinal usado pelo microdrive (?).

Escrita em xxEFh

b7	b6	b5	b4	b3	b2	b1	b0
-	-	WAI	CTS	ERS	R/W	CCL	CDT

WAI - *WAIt*: usado para sincronização da rede (?).
CTS - *Clear To Send*: Sinal usado pela RS-232 (?).
ERS - *ERaSe*: Sinal usado pelo microdrive (?).
R/W - *Read/Write*: Sinal usado pelo microdrive (?).
CCL - *Communication CLock*: Sinal usado pelo microdrive (?).
CDT - *Communication DaTa*: Sinal usado pelo microdrive (?).

•**xxF7h**: Esta é uma porta para envio e recepção de dados pela rede ou pela RS-232, e também tem funções diferentes na leitura e escrita:

Leitura de xxF7h

b7	b6	b5	b4	b3	b2	b1	b0
TXD	-	-	-	-	-	-	NIN

TXD - *Transmitted Data*: Sinal de recepção da RS-232.
NIN - *Network INput*: Recepção de dados pela rede.

Escrita em xxF7h

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	DOT

DOT - *Data OuT*: usado para enviar dados pela RS-232 ou pela rede. Se CDT (bit 0) da porta xxEFh for 0, o dado será enviado para a RS-232; se CDT for 1, o dado será enviado para a rede. Vale lembrar que esta comunicação só funciona se o microdrive não estiver sendo usado.

5.5. ZX Interface II

A Interface II é a interface de joystick padrão do Spectrum. Ela usa 4 portas para sua comunicação: P_SINCLAIR1 (EFFEh), B_SINCLAIR1 (1000h), P_SINCLAIR2 (F7FEh) e B_SINCLAIR2 (0800h).

P_SINCLAIRx se referem à posição do joystick, dada no seguinte formato:

Leitura de EFFEh (P_SINCLAIR1)

b7	b6	b5	b4	b3	b2	b1	b0
1	1	1	LFT	RGT	DN	UP	FIR

LFT - *LeFT*: 0 indica botão "para esquerda" pressionado.
RGT - *RiGhT*: 0 indica botão "para direita" pressionado.
DN - *DowN*: 0 indica botão"para baixo" pressionado.
UP - *UP*: 0 indica botão "para cima" pressionado.
FIR - *FIRe*: 0 indica botão de "tiro" pressionado.

Leitura de F7FEh (P_SINCLAIR2)

b7	b6	b5	b4	b3	b2	b1	b0
1	1	1	FIR	UP	DN	RGT	LFT

FIR - *FIRe*: 0 indica botão de "tiro" pressionado.
UP - *UP*: 0 indica botão "para cima" pressionado.
DN - *DowN*: 0 indica botão"para baixo" pressionado.
RGT - *RiGhT*: 0 indica botão "para direita" pressionado.
LFT - *LeFT*: 0 indica botão "para esquerda" pressionado.

5.6. Kempston Interface

A Interface Kempston era uma interface de joysticks muito comum para Spectrum. Ela usa 2 portas para sua comunicação: P_KEMPSTON (xx1Fh), B_KEMPSTON (xx20h).

5.7. Cursor / Protek / AGF

O cursor é mapeado direto em teclas e, sendo assim, sua leitura é feita usando a matriz do teclado, para obter os valores das teclas 5, 6, 7, 8 e 0

Porta	b4	b3	b2	b1	b0
F7FEh	5	4	3	2	1
EFFEh	6	7	8	9	0

Estes valores são interpretados como:

5- Esquerda
6- Baixo
7- Cima
8- Direita
0- Tiro

5.8. Interface Centronics do +3

A comunicação com a impressora do Spectrum +3 ocorre basicamente por duas portas. A primeira é a porta 1FFDh, que contem o sinal strobe:

Escrita em 1FFDh

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	PST	-	-	-	-

PST - *Printer STrobe*: sinal usado para controle da impressora.

A segunda é a porta 0FFDh, usada para comunicação (8 bits de dados?)

Escrita em 0FFDh

b7	b6	b5	b4	b3	b2	b1	b0
?	?	?	?	?	?	?	?

6. ENTRADA E SAÍDA - ARMAZENAMENTO

O Spectrum possui diversos sistemas de armazenamento: da fita cassete ao disquete, passando pelo microdrive. Esta seção cobre o uso de alguns destes dispositivos.

6.1. Leitura e Escrita de Fita Cassete

Como a maioria dos computadores pessoais de sua época, o Spectrum vinha com uma interface de cassete integrada.

A leitura do cassete é feita pela porta xxFEh, recebendo o dado pelo bit 6 (Ear In). A escrita é feita pela mesma porta xxFEh, porém neste caso o bit da escrita é o bit 3 (Mic Out).

Escrita em xxFEh

b7	b6	b5	b4	b3	b2	b1	b0
-	EI	-	-	MO	-	-	-

EI - *Ear In*: Bit para entrada (leitura) da fita cassete.
MO - *Mic Out*: Bit para saída (escrita) para fita cassete.

Note que para a leitura do bit 6 é importante zerar os bits 3 (Mic Out) e 4 (Speaker Out). Isso ocorre porque há ligações elétricas entre estes bits em alguns modelos de Spectrum, fazendo com que haja interferência nos valores lidos. Veja o exemplo na tabela abaixo:

Valor Bit 3	Valor Bit 4	Lido no Bit 6 (Spectrum Edição 2)	Lido no Bit 6 (Spectrum Edição 3)
1	1	1	1
1	0	1	1
0	1	1	0
0	0	0	0

Isso pode ser usado para determinar a versão do equipamento ou mesmo um emulador; basta testar as combinações desta tabela e ler o valor do bit 6. Caso o valor do bit 6 seja sempre 1 ou sempre 0 (em qualquer combinação de valores dos bits 3 e 4), o software está rodando em um emulador.

Nas máquinas Spectrum +2A e +3 o bit 6 não sofre esta influência dos bits 3 e 4, possuindo sempre o valor 0 na ausência de sinal de entrada.

Um problema associado à leitura da porta xxFEh é que o resultado que ela retorna depende da ULA, então o Z80 será congelado (halt) durante o processo. Isso significa que o acesso a esta porta é mais lento que os 11 estados T usuais de um IN em uma porta qualquer. Além disso, se os bits altos da porta estiverem entre 40h e 7Fh (40FEh ~7FFEh), haverá o efeito colateral do Snow Effect, explicado na seção 4, que trata sobre o vídeo do Spectrum.

6.1.1. Estrutura de Dados do Cassete

Um arquivo em cassete, gravado com o comando SAVE, possui dois blocos: um header de 19 bytes e o bloco de dados, de tamanho variado.

Bloco de Header:

•Uma instrodução de 8063 pulsos, cada um com uma duração de 2168 estados T.
•Um primeiro pulso de sincronia, com duração de 667 estados T.
•Um segundo pulso de sincronia, com duração de 735 estados T.
•Dados: um bit 0 codificado como 2 pulsos de 855 estados T cada e um bit 1 como dois pulsos de 1710 estados T cada.

A região de dados contém as seguintes informações:

Byte de Idenficação:	00h
Tipo de Arquivo:	1 byte (0, 1, 2 ou 3)
Nome do Arquivo:	10 bytes (preenchido com espaços)
Comprimento do Bloco de Dados:	2 bytes
Parâmetro 1:	2 bytes
Parâmetro 2:	2 bytes
CheckSUM:	1 byte (XOR de **todos** os bytes do header)

Os tipos de arquivo são: 0 para programa, 1 para matriz numérica, 2 para matriz de caracteres e 3 para arquivo de código.

No caso de um arquivo de programa (tipo 0), o parâmetro 1 indica o número da linha de auto-execução (ou um número maior que 8000h se nenhum foi fornecido, indicando "não executar") e o parâmetro 2 indica o início da área de variáveis do programa, relativamente ao endereço inicial de carregamento.

No caso de um arquivo de código (tipo 3), o parâmetro 1 conterá o endereço inicial daquele bloco de código, a o parâmetro 2 será sempre 8000h. Por exemplo, um arquivo do tipo SCREEN$ será gravado como tipo "código", iniciando no endereço 4000h e com comprimento 1B00h.

Finalmente, no caso de um arquivo de dados (tipos 1 e 2), o primeiro byte do parâmetro 1 (LSB) contém o nome da variável.

Bloco de Dados:

•Uma instrodução de 3223 pulsos, cada um com uma duração de 2168 estados T.
•Um primeiro pulso de sincronia, com duração de 667 estados T.
•Um segundo pulso de sincronia, com duração de 735 estados T.
•Dados: um bit 0 codificado como 2 pulsos de 855 estados T cada e um bit 1 como dois pulsos de 1710 estados T cada.

A região de dados contém as seguintes informações:

Byte de Idenficação: FFh
Dados: Número de bytes indicado no header
CheckSUM: 1 byte (XOR de **todos** os bytes do arquivo)

6.2. Leitura e Escrita do +3 Disk Drive

O Spectrum teve uma versão com disk drive, na versão Spectrum +3. O controlador de disco usado é o uPD765A, a mesma FDC do Amstrad CPC (provavelmente porque o fabricante do Spectrum +3 era a Amstrad... :)

As portas de controle do disk drive no Spectrum +3 são: 1FFDh, 2FFDh e 3FFDh.

Escrita em 1FFDh

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	DM	-	-	-

DM - *Disk Motor*: Se configurado com 1, liga o motor do drive. Se configurado com 0, desliga o motor.

A porta 2FFDh, quando lida, retorna o valor do registrador de status do FDC.

Leitura de 2FFDh

b7	b6	b5	b4	b3	b2	b1	b0
?	?	?	?	?	?	?	?

A porta 3FFDh serve para leitura e escrita. Bytes escritos nesta porta serão enviados ao FDC e bytes lidos desta porta serão lidos do FDC.

Leitura/Escrita em 3FFDh

b7	b6	b5	b4	b3	b2	b1	b0
?	?	?	?	?	?	?	?

7. ARQUIVOS DE EMULADORES

A maioria dos programas de Spectrum encontram-se na internet em algum formato "encapsulado", ou seja, não estão na forma de arquivos simples. Para qualquer edição e/ou conversão destes programas, faz-se necessário extrair as informações destes arquivos de encapsulamento.

Existem três tipos básicos de arquivos:

- **Snapshots**: imagens da memória do Spectrum (uma espécie de Save-State)
- **Imagens de Cassete**: são arquivos que tentam reproduzir com alguma fidelidade o conteúdo de uma fita cassete de ZX Spectrum.
- **Imagens de Disco**: são arquivos de uma imagem exata (raw) de um disco de spectrum.

As seções seguintes trata de descrição de alguns dos mais comuns tipos de arquivos destas categorias.

7.1. Arquivos do Tipo Snapshot

Um snapshot é um arquivo que representa o conteúdo da memória do Spectrum em um dado instante. São usados para armazenar jogos e programas simples, que não possuem multi-load (carregamento durante o jogo) e não é possível extrair os arquivos originais deste tipo de imagem. Os formatos mais comuns deste tipo são SNA, SLT e Z80.

7.1.1. Arquivos do tipo SNA

Talvez o tipo de snapshot mais comum, o arquivo no formato SNA tem dois tipos: o SNA de Spectrum 48K e o SNA de Spectrum 128.

SNA de Spectrum 48K

O SNA de Spectrum 48K possui 49179 bytes, que podem ser detalhados da seguinte forma:

Byte	Tamanho	Descrição
0	1 byte	Valor do registrador I
1	4 words	Valor de HL', DE', BC' e AF'
9	5 words	Valor de HL, DE, BC, IY e IX
19	1 byte	Interrupção (b2 = 1 => EI; b2 = 0 => DI)
20	1 byte	Valor do registrador R
21	2 words	AF, SP
25	1 byte	Interrupt Mode (0=IM0; 1=IM1; 2=IM2)
26	1 byte	Cor da Borda (0 a 7)
27	49152 bytes	Dump da RAM (4000h ~FFFFh)

Este tipo de arquivo tem um problema sério; como pode ser observado, ele não registra o valor do PC (Program Counter). Para ter o PC registrado em algum lugar, antes de salvar o arquivo .SNA é sempre executada a instrução fictícia "PUSH PC" (CALL qquer coisa), que coloca o PC na pilha (SP). Assim, após carregado o SNA em um emulador e os valores dos registradores definidos, bastaria indicar um RETN para que a execução continuasse onde havia sido tirado o snapshot.

O problema é que, em alguns casos, o SNA pode ser gerado em pontos críticos onde não havia mais espaço na pilha, e por conseqüência o "PUSH PC" irá sobrescrever dados ou código do programa, corrompendo a imagem.

SNA de Spectrum 128

O SNA de Spectrum 128/+2 possui 131103 ou 147487 bytes, que podem ser detalhados da seguinte forma:

Byte	Tamanho	Descrição	
0	1 byte	Valor do registrador I	S
1	4 words	Valor de HL', DE', BC' e AF'	N
9	5 words	Valor de HL, DE, BC, IY e IX	A
19	1 byte	Interrupção (b2 = 1 => EI; b2 = 0 => DI)	
20	1 byte	Valor do registrador R	D
21	2 words	AF, SP	E
25	1 byte	Interrupt Mode (0=IM0; 1=IM1; 2=IM2)	
26	1 byte	Cor da Borda (0 a 7)	4
27	16384	Dump da página 5 (4000h ~7FFFh)	8
16.411	16384	Dump da página 2 (8000h ~BFFFh)	K
32.795	16384	Dump da página no frame 3 (C000h ~	
49.179	1 word	Valor do registrador PC	A
49.181	1 byte	Valor da porta 7FFDh	D
49.182	1 byte	Indicativo se a ROM do TR-DOS está	I
49.183	16384	Dump da 1a. página de memória não	C
65.567	16384	Dump da 2a. página de memória não	I
81.951	16384	Dump da 3a. página de memória não	O
98.335	16384	Dump da 4a. página de memória não	N
114.71	16384	Dump da 5a. página de memória não	A
131.10	16384	Dump da 6a. página de memória não	L

É importante notar que a página alocada no terceiro frame irá sempre ser salva, mesmo que seja a página 2 ou 5 (neste caso, fica na imagem repetida). As páginas armazenadas a partir do byte 49183 serão todas as não paginadas em frame algum, na ordem (de 0 a 7). Por exemplo: se a página 4 está alocada, os bancos 5, 2 e 4 estarão no primeiro bloco (igual ao de Spectrum 48K) e as páginas 0, 1, 3, 6 e 7 serão armazenadas a partir do byte 49183. Neste caso, o último bloco (a partir do byte 131103) não existirá.

Caso a página alocada no frame 3 seja repetida, por exemplo, se a página 5 está alocada no frame 3, os bancos 5, 2 e 5 estarão no primeiro bloco (igual ao de Spectrum 48K) e as páginas 0, 1, 3, 4, 6 e 7 serão armazenadas a partir do byte 49183. Neste caso, o último bloco (a partir do byte 131103) será necessário.

É importante notar que na versão 128K do arquivo tipo SNA o valor de PC **é** armazenado, dispensando o truque do "PUSH PC", evitando os problemas citados anteriormente.

7.1.2. Arquivos do tipo SLT

Os arquivos do tipo Z80 e SNA possuem um problema: se o jogo é multi-load (cada fase precisa ser lida do cassete, por exemplo) é necessário criar um SNA para cada fase para poder jogá-las. Para resolver este problema, foi criado o padrão SLT, contendo os arquivos todos lidos em cada estágio. O formato foi desenvolvido com a possibilidade de expansão em mente.

O SLT possui tamanho variável de bytes, mas eles podem ser detalhados da seguinte forma:

Tamanho	Descrição
n bytes	Texto "Z80 Snapshot (version 2+)"
3 bytes	Três bytes nulos (00h 00h 00h)
3 bytes	Texto "SLT"

1 word	Tipo de Dado (0 = fim da tabela, 1 = dado de nível)
1 word	Identificador de dados. Se tipo é 1, identifica o # do nível
1 dword	Comprimento dos dados
1 word	Tipo de Dado (0 = fim da tabela, 1 = dado de nível)
1 word	Identificador de dados. Se tipo é 1, identifica o # do nível
1 dword	Comprimento dos dados
...	...

x bytes	Bytes de dados
y bytes	Bytes de dados
...	...

O primeiro trecho é o header do arquivo. O segundo trecho é uma espécie de "table of contents", ou seja, o índice do arquivo. O terceiro trecho são os dados em si.

Quando for solicitada a leitura de um novo nível/estágio (instrução ED/FB), o registrador A conterá o número do nível a ser lido. A idéia é escolher pela tabela de conteúdos o bloco correto, calcular a posição inicial dos dados e então eles devem ser carregados.

Os dados de um nível estarão armazenados na forma com que o tipo Z80 armazena as páginas de memória.

7.1.3. Arquivos do tipo Z80

O formato .Z80 é, certamente, o mais usado de todos. Assim como os formatos anteriormente citados, o formato .Z80 é um snapshot da memória, além dos valores dos registradores. Existem basicamente 3 versões do arquivo tipo Z80, além de algumas extensões feitas por alguns emuladores.

Z80 Versão 1

Os arquivos tipo Z80 versão 1 foram criados para armazenar um Snapshot de um Spectrum 48K. O conteúdo do arquivo pode ser descrito da seguinte forma:

Byte	Tamanho	Descrição
0	1 byte	Registrador A
1	1 byte	Registrador F
2	1 word	Registrador BC (LSB / MSB)
4	1 word	Registrador HL
6	1 word	Registrador PC
8	1 word	Registrador SP
10	1 byte	Registrador I
11	1 byte	Registrador R (bit 7 sem significado)
12	1 byte	b0 - Bit 7 do registrador R b1-3 - Cor da borda b4 - 1= BASIC SamROM ativada b5 - 1= bloco de dados comprimido b6-7 - sem significado
13	1 word	Registrador DE
15	1 word	Registrador BC'
17	1 word	Registrador DE'
19	1 word	Registrador HL'
21	1 byte	Registrador A'
22	1 byte	Registrador F'
23	1 word	Registrador IY
25	1 word	Registrador IX
27	1 byte	Interrupção (0=DI; outro valor = EI)
28	1 byte	IFF2
29	1 byte	bit 0-1 - Modo de IRQ (IM0 a IM2) bit 2 - 1 = Emulação da Edição 2 bit 3 - 1 = Freqüência de IRQ dupla bit 4-5 - 0 ou 2 = vídeo normal; 1 = sincronização alta; 3 = sincronização baixa. bit 6-7 - 0 = cursor, 1 = Kempston, 2 = Sinclair esquerdo, 3 = Sinclair direito.

Por questões de compatibilidade, se o byte 12 valer FFh, ele deve ser considerado como valendo 1.

Depois deste bloco de 30 bytes, seguem os 48KB da memória do Spectrum 48KB, podendo estar comprimido, de acordo com o bit do byte de offset 12 do cabeçalho. A compressão é muito simples:

a) Sempre que uma seqüência de 5 ou mais bytes iguais for encontrada, ela é substituída por um código de 4 bytes: ED ED xx yy, onde xx é a quantidade de repetições e yy é o valor a ser repetido.

b) Se ED ED for encontrado na memória, ele será codificado como ED ED 02 ED.

c) Um byte precedido por ED nunca será considerado para a compressão. Por exemplo, se for encontrad na memória: ED 00 00 00 00 00 00, isso será codificado como ED 00 ED ED 05 00.

Z80 Versão 2 e 3

Byte	**Tamanho**	**Descrição**
0	1 byte	Registrador A
1	1 byte	Registrador F
2	1 word	Registrador BC (LSB / MSB)
4	1 word	Registrador HL
6	1 word	00h (indica versão 2 e 3)
8	1 word	Registrador SP
10	1 byte	Registrador I
11	1 byte	Registrador R (bit 7 sem significado)
12	1 byte	b0 - Bit 7 do registrador R b1-3 - Cor da borda b4 - 1= BASIC SamROM ativada b5 - 1= bloco de dados comprimido b6-7 - sem significado
13	1 word	Registrador DE
15	1 word	Registrador BC'
17	1 word	Registrador DE'
19	1 word	Registrador HL'
21	1 byte	Registrador A'
22	1 byte	Registrador F'
23	1 word	Registrador IY
25	1 word	Registrador IX
27	1 byte	Interrupção (0=DI; outro valor = EI)
28	1 byte	IFF2
29	1 byte	bit 0-1 - Modo de IRQ (IM0 a IM2) bit 2 - 1 = Emulação da Edição 2 bit 3 - 1 = Freqüência de IRQ dupla bit 4-5 - sem significado bit 6-7 - 0 = cursor, 1 = Kempston, 2 = Sinclair esquerdo, 3 = Sinclair direito.

30	1 word	Comprimento do block header adicional: 30 para versão 2 de Z80 e 54 ou 55 para versão 3 de Z80.
32	1 word	Valor do Registrador Program Counter
34	1 byte	Modo de Hardware (explicado adiante)
35	1 byte	No modo SamRAM, estado do 74LS259 No modo 128, contém o último OUT para a porta 7FFDh No modo Timex contém o último OUT para a porta F4h
36	1 byte	Contem FFh se a ROM da Interface I está paginada. No modo Timex, contém o último OUT para a porta FFh
37	1 byte	B0: 1 se a emulação do registrador R está ativada. B1: 1 se a emulação do LDIR está ativada. B2: 1 se PSG (AY) está em uso (mesmo em uma máquina 48K) B3-5: Não usado B6: 1 se emulação da Fuller Audio Box está ligada (só se B2=1) B7: Modifica hardware (explicado adiante)
38	1 byte	Último OUT para a porta FFFDh (# do reg. do processador de som)
39	16 bytes	Conteúdo dos registradores do processador de som
55	1 word	Contador de estados T (low)
57	1 byte	Contador de estados T (high)
58	1 byte	Byte flag usado pelo Spectator (emulador de Sinclair QL) Ignorado na leitura, sempre zero na escrita.
59	1 byte	FFh, se a ROM do MGT estiver paginada.
60	1 byte	FFh, se a ROM da Multiface estiver paginada. Deveria ser sempre 0.
61	1 byte	FFh se 0000h~1FFFh é ROM, 0 se é RAM.
62	1 byte	FFh se 2000h~3FFFh é ROM, 0 se é RAM.
63	5 words	5 valores de mapeamento de teclado, para joystick do usuário
73	5 words	5 ASCII words, teclas correspondentes ao mapeamento acima
83	1 byte	Tipo de MGT: 0= Disciple+Epson, 1=Disciple+HP, 16=Plus D
84	1 byte	Estado do botão de inibiçãodo Disciple: 0=out; FFh=in
85	1 byte	Flag de inibição do Disciple: 0=Rom pageável; FFh=Não pageável
86	1 byte	Último OUT para a porta 1FFDh

A parte acinzentada só existe na versão 3 do arquivo tipo Z80 e a última linha (offset 86) só existe se o byte 30 for 55.

Os únicos pontos que os valores possuem significados distintos entre a versão 2 e 3 são no byte 34 (modo de hardware) e ...

Modos de Hardware

Valor	Significado na Versão 2	Significado na Versão 3	Significado na Versão 3.02	Extesões
0	48K	48K	48K	
1	48K + Interface 1	48K + Interface1	48K + Interface1	
2	SamRAM	SamRAM	48K + MGT	
3	128K	48K + MGT	SamRAM	
4	128K + Interface 1	128K	128K	
5	-	128K + Interface 1	128K + Interface 1	
6	-	128K + MGT	128K + MGT	
7	-	-	-	Spectrum +3
8	-	-	-	+3 (não usar)
9	-	-	-	Pentagon (128)
10	-	-	-	Scorpion (256)
11	-	-	-	Didaktik
12	-	-	-	Spectrum +2
13	-	-	-	Spectrum +2A
14	-	-	-	TC2048
15	-	-	-	TC2068
128	-	-	-	TS2068

As extensões aparecem tanto em arquivos da versão 2 quanto da versão 3.

Modificações de Hardware

Se o bit 7 do byte 37 estiver definido, o efeito dependerá do tipo de hardware definido no modo de hardware. Ativar o bit 7 significa o seguinte em cada uma das situações.

Modo de Hardware Definido	Modo de Hardware a Ser Emulado
Spectrum 48K	Spectrum 16K
Spectrum 128	Spectrum +2
Spectrum +3	Spectrum +2A

O Contador de Estados T

O contador de estados T alto conta sempre em MOD 4 (resto de divisão por 4). Logo após a interrupção da ULA, a parte high do contador de estados é colocada em 3 e é incrementado a cada 5 milisegundos emulados. Nestes

intervalos de 20ms, o low do contador de estdos T decresce de 17471 até 0 (17726 no Spectrum 128), produzindo um total de 69888 (70908) estados T por frame.

As 5 ASCII words

As 5 ASCII words (com byte alto sempre 0) nas posições 73 a 82 indicam as teclas correspondentes às direções do joystick (esquerda, direita, baixo, cima e tiro). Shift, Symbol Shift, Enter e Espaço são denotados por [,], / e \, respectivamente. Os valores ASCII são usados apenas para apresentar as teclas do joystick.

A informação usada na emulação real serão as 5 words do mapeamento do teclado (que devem corresponder aos valores ASCII), sendo o byte baixo (low) na faixa de 0 a 7, determinando a linha do teclado e o byte alto (high) é um byte de máscara que determina a coluna. O Enter, por exemplo, é armazenado como 0106h, ou seja, linha 6, coluna 1 e a tecla 'g' seria 1001h, ou seja, linha 1 coluna 4.

Definição da Multiface no offset 60

O byte do offset 60 deve sempre ser 0 porque o conteúdo da RAM da Multiface não é gravada no arquivo Snapshot. Se a Multiface estava paginada quando o Snapshot foi gerado, o programa emulado provavelmente irá travar.

RAM na região da ROM

Os valors nos offsets 61 e 62 devem ser definidos em função de outras flags, como as dos offsets 34, 59, 60 e 83.

Armazenamento dos Dados

A partir do header, seguem os blocos de dados da memória, cada um contendo os dados comprimidos de um bloco de 16K. A compressão é exatamente a mesma da versão 1, com exceção do marcador de fim, que não existe mais. A estrutura de um bloco de memória é:

Byte	Tamanho	Descrição
0	1 word	Comprimento dos dados comprimidos (sem contar os 3 bytes do header). Se for FFFFh, os dados não estão comprimidos.
2	1 byte	Tipo do bloco de dados
3	n bytes	Bytes de dados
...	...	...

O tipo do bloco depende do modo de hardware, e é definido conforme a tabela:

Tipo	48K	128	SamRAM
0	48K ROM	ROM (BASIC)	48K ROM
1	Interface I, Disciple	ou Plus D, de acordo	com a configuração.
2	-	ROM (RESET)	SamRAM ROM (BASIC)
3	-	Página 0	SamRAM ROM (Monitor)
4	Frame 2	Página 1	Frame 2 normal
5	Frame 3	Página 2	Frame 3 normal
6	-	Página 3	Frame 2 shadow
7	-	Página 4	Frame 3 shadow
8	Frame 1	Página 5	Frame 1
9	-	Página 6	-
10	-	Página 7	-
11	Multiface ROM	Multiface ROM	-

No modo 48K, os blocos dos tipos 4, 5 e 8 são normalmente salvas. No modo SamRAM, os blocos do tipo 4 a 8 são salvos. No modo 128, todos os blocos dos tipos 3 a 10 são salvos. No modo Pentagon são gravados os mesmos blocos do 128. No Scorpion, as 16 páginas de RAM são gravadas nos blocos de 3 a 16 (qual tipo?)

7.2. Arquivos do Tipo Cassete

Um arquivo Tipo Cassete representa o conteúdo de uma fita cassete padrão de Spectrum, sendo dois o tipos principais de codificação: TAP e TZX.

Os arquivos no formato TAP possuem codificação bastante simples e não são capazes de reproduzir qualquer fita, sendo que alguns softwares precisam ser "adaptados" para poderem ser lidos no formato TAP. Para corrigir este problema, foi criado o formato TZX, que é capaz de armazenar praticamente qualquer tipo de dado de fita (incluindo proteções contra cópia e arquivos com headers arbitrários), mas sua codificação é bastante complexa.

7.2.1. Arquivos do tipo TAP

Os arquivos do tipo TAP são os mais comuns e, basicamente, sua estrutura é praticamente "RAW", isto é, os dados estão lá exatamente como estariam em uma fita cassete normal (só que já decodificados, claro). Antes de cada **bloco de informações,** existe a sequinte seqüência de dados (compondo o **header**): 13h 00h. Após o bloco de informações, vem o bloco de dados.

Bloco de Informações:
Header do Bloco: 13h 00h
Byte de Idenficação: 00h
Tipo de Arquivo: 1 byte (0, 1, 2 ou 3) (veja leitura de cassete)
Nome do Arquivo: 10 bytes (preenchido com espaços)
Comprimento do Bloco de Dados: 2 bytes
Parâmetro 1: 2 bytes (veja leitura de cassete)
Parâmetro 2: 2 bytes (veja leitura de cassete)
CheckSUM: 1 byte (XOR de **todos** os bytes do arquivo)
Comp. Real do Bloco de Dados: 2 bytes (inclui identificador e checksum)

Bloco de Dados:
Byte de Idenficação: FFh
Dados: Número de bytes indicado no header
CheckSUM: 1 byte (XOR de **todos** os bytes do arquivo)

APÊNDICE A. PORTAS USADAS NO ZX SPECTRUM

xx1Fh - Kempston Interface (P)
xx20h - Kempston Interface (B)
xxE7h - ZX Interface I - MicroDrive Data
xxEFh - ZX Interface I - Control
xxF7h - Interface I - RS-232/Network Data
xxFEh - Configuração da ULA
xxFFh - Leitura do barramento
0800h - ZX Interface II - Joystick 2 (B)
0FFDh - +2A/+3 - Porta Centronics - Porta Paralela do +3 (Bidirecional?)
1000h - ZX Interface II - Joystick 1 (B)
1FFDh - +2A/+3 - Configuração de Hardware
2FFDh - +3 FDC Status
3FFDh - +3 FDC Data
7FFEh - Leitura da linha 8 da Matriz do teclado.
BFFDh - Comunicação com o AY-3-8192 (PSG).
BFFEh - Leitura da linha 7 da Matriz do teclado.
DFFEh - Leitura da linha 6 da Matriz do teclado.
EFFEh - Leitura da linha 5 da Matriz do teclado / Interface II (Leitura do Joy 1 (P)).
F7FEh - Leitura da linha 4 da Matriz do teclado / Interface II (Leitura do Joy 2 (P)).
FBFEh - Leitura da linha 3 da Matriz do teclado.
FDFEh - Leitura da linha 2 da Matriz do teclado.
FEFEh - Leitura da linha 1 da Matriz do teclado.
FFFDh - Comunicação com o AY-3-8192 (PSG).
7FFDh - 128 - Configuração de Memória

APÊNDICE B. RESUMO DAS PORTAS USADAS NO ZX SPECTRUM

Todos os modelos de ZX Spectrum respondem às seguintes portas:

xx1Fh - Kempston Interface (P)
Escrita: Não
Leitura: Sim
Contended: Não.

xx20h - Kempston Interface (B)
Escrita: Não
Leitura: Sim
Contended: Sim.

xxE7h - ZX Interface I - MicroDrive Data
Escrita: Envia 8 bits para o MicroDrive
Leitura: Recebe 8 bits do MicroDrive
Contended: Não.

xxEFh - ZX Interface I - Control
Escrita: Sim
Leitura: Sim
Contended: Não.

Leitura em xxEFh

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	BSY	DTR	GAP	SYN	WRP

BSY - *BuSY*: usado para indicar que o microdrive está ocupado (?).
DTR - *Data Terminal Ready*: Sinal usado pela RS-232.
GAP - *GAP*: Sinal usado pelo microdrive.
SYN - *SYNchrony*: Sinal usado pelo microdrive.
WRP - *WRite Protect*: Sinal usado pelo microdrive.

Escrita em xxEFh

b7	b6	b5	b4	b3	b2	b1	b0
-	-	WAI	CTS	ERS	R/W	CCL	CDT

WAI - *WAIt*: usado para sincronização da rede.
CTS - *Clear To Send*: Sinal usado pela RS-232.
ERS - *ERaSe*: Sinal usado pelo microdrive.
R/W - *Read/Write*: Sinal usado pelo microdrive.
CCL - *Communication CLock*: Sinal usado pelo microdrive.
CDT - *Communication DaTa*: Sinal usado pelo microdrive.

xxF7h - Interface I - RS-232/Network Data
Escrita: Sim
Leitura: Sim
Contended: Não.

Leitura em xxF7h

b7	b6	b5	b4	b3	b2	b1	b0
TXD	-	-	-	-	-	-	NIN

TXD - *Transmitted Data*: Sinal de recepção da RS-232.
NIN - *Network INput*: Recepção de dados pela rede.

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	-	-	-	-	DOT

DOT - *Data OuT*: usado para enviar dados pela RS-232 ou pela rede. Se CDT (bit 0) da porta xxEFh for 0, o dado será enviado para a RS-232; se CDT for 1, o dado será enviado para a rede. Vale lembrar que esta comunicação só funciona se o microdrive **não** estiver sendo usado.

xxFEh - Configuração da ULA
Escrita: Sim
Leitura: Depende dos bytes altos
Contended: Sim (menos nos modelos +2A e +3).

Escrita em FFFEh

b7	b6	b5	b4	b3	b2	b1	b0
-	EI	-	SO	MO	G	R	B

R, G, B - *Border Red, Green and Blue*: determinam a cor da borda da tela.
SO - *Speaker Out*: Emite ruído pela saída de áudio padrão do Spectrum.
EI - *Ear In*: Bit para entrada (leitura) da fita cassete.
MO - *Mic Out*: Bit para saída (escrita) para fita cassete.

xxFFh - Leitura do barramento
Escrita: Não
Leitura: retorna FF, se a ULA não estiver usando a VRAM, ou outro valor, se a ULA estiver usando a VRAM. Nos modelos +2A e +3 retorna sempre FFh.
Contended: Não.

0800h - ZX Interface II - Joystick 2 (B)
Escrita: Não
Leitura: Sim
Contended: Sim.

1000h - ZX Interface II - Joystick 1 (B)
Escrita: Não
Leitura: Sim
Contended: Sim.

7FFEh - Leitura da linha 8 da Matriz do teclado.
Escrita: Não (na verdade, modifica a cor da borda - escrita em 0FEh)
Leitura: retorna 0 nos bits referentes às teclas pressionadas.
Contended: Sim.

BFFDh - Comunicação com o AY-3-8192 (PSG).
Escrita: Data Write (escrita no registrador selecionado atual)
Leitura: ?
Contended: Não.

BFFEh - Leitura da linha 7 da Matriz do teclado.
Escrita: Não (na verdade, modifica a cor da borda - escrita em 0FEh)
Leitura: retorna 0 nos bits referentes às teclas pressionadas.
Contended: Sim.

DFFEh - Leitura da linha 6 da Matriz do teclado.
Escrita: Não (na verdade, modifica a cor da borda - escrita em 0FEh)
Leitura: retorna 0 nos bits referentes às teclas pressionadas.
Contended: Sim.

EFFEh - Leitura da linha 5 da Matriz do teclado / Interface II (Joy 1 (P)).
Escrita: Não (na verdade, modifica a cor da borda - escrita em 0FEh)
Leitura: retorna 0 nos bits referentes às teclas pressionadas.
Contended: Sim.

Leitura de EFFEh (Interface-II)

b7	b6	b5	b4	b3	b2	b1	b0
1	1	1	LFT	RGT	DN	UP	FIR

LFT - *LeFT*: 0 indica botão "para esquerda" pressionado.
RGT - *RiGhT*: 0 indica botão "para direita" pressionado.
DN - *DowN*: 0 indica botão"para baixo" pressionado.
UP - *UP*: 0 indica botão "para cima" pressionado.
FIR - *FIRe*: 0 indica botão de "tiro" pressionado.

F7FEh - Leitura da linha 4 da Matriz do teclado / Interface II (Joy 2 (P)).
Escrita: Não (na verdade, modifica a cor da borda - escrita em 0FEh)
Leitura: retorna 0 nos bits referentes às teclas pressionadas.
Contended: Sim.

Leitura de F7FEh (Interface-II)

b7	b6	b5	b4	b3	b2	b1	b0
1	1	1	LFT	RGT	DN	UP	FIR

LFT - *LeFT*: 0 indica botão "para esquerda" pressionado.
RGT - *RiGhT*: 0 indica botão "para direita" pressionado.
DN - *DowN*: 0 indica botão"para baixo" pressionado.
UP - *UP*: 0 indica botão "para cima" pressionado.
FIR - *FIRe*: 0 indica botão de "tiro" pressionado.

FBFEh - Leitura da linha 3 da Matriz do teclado.
Escrita: Não (na verdade, modifica a cor da borda - escrita em 0FEh)
Leitura: retorna 0 nos bits referentes às teclas pressionadas.
Contended: Sim.

FDFEh - Leitura da linha 2 da Matriz do teclado.
Escrita: Não (na verdade, modifica a cor da borda - escrita em 0FEh)
Leitura: retorna 0 nos bits referentes às teclas pressionadas.
Contended: Sim.

FEFEh - Leitura da linha 1 da Matriz do teclado.
Escrita: Não (na verdade, modifica a cor da borda - escrita em 0FEh)
Leitura: retorna 0 nos bits referentes às teclas pressionadas.
Contended: Sim.

FFFDh - Comunicação com o AY-3-8192 (PSG).
Escrita: Register Select (0 a 14)
Leitura: Data Read (leitura da porta de I/O ativa).
Contended: Não.

A.1. Portas Adicionadas no Spectrum 128

7FFDh - Configuração de Memória
Escrita: Sim
Leitura: Cópia no endereço 5B5Ch
Contented: Sim.

Escrita em 7FFDh

b7	b6	b5	b4	b3	b2	b1	b0
-	-	LC	RS	US	MB2	MB1	MB0

MB0 ~ MB1 - *Memory Bank selector*: determina qual a página da RAM (0~7) está no frame 3.

US - *ULA frame Selector*: determina qual página será usada pela ULA para gerar a imagem (US=0 - página 5; US=1 - página 7). Note que ele apenas muda qual página é usada para gerar a imagem do vídeo, mas **não** altera a página ativa no frame 1 (que é sempre a 5).

RS - *ROM Selector*: determina qual ROM está sendo usada (RS=0 - ZX128K ROM; RS=1 - ZX48K ROM).

LC - *Lock mapper Control*: quando setado (1) desliga o controle de mapeamento, fixando o mapeamento configurado. Existente para compatibilidade com programas antigos, que poderiam escrever lixo nesta porta. Sendo assim, esse bit é ativado ao entrar no modo 48K. Depois de ativado, só pode ser desativado através de um reset do equipamento.

A.2. Portas Adicionadas no Spectrum +2A/+3

0FFDh - Porta Centronics - Porta Paralela do +3 (Bidirecional?)

Escrita: Sim
Leitura: Sim (?)
Contended: Não.

Escrita/Leitura em 0FFDh

b7	b6	b5	b4	b3	b2	b1	b0
?	?	?	?	?	?	?	?

1FFDh - Configuração de Hardware

Escrita: Sim
Leitura: Cópia no endereço 5B67h
Contended: Não.

Escrita em 1FFDh

b7	b6	b5	b4	b3	b2	b1	b0
-	-	-	PST	DM	RC1	RC0	PM

PM - *Paging Mode*: determina a função dos bit RC0 e RC1. Se PM=0 o sistema está no modo *normal* e se PM=1 o sistema está no modo *special*.

RC0-RC1 - *RAM/ROM Control*: no modo normal, RC0 é ignorado e RC1 constitui o bit alto da seleção de ROM. No modo special, RC0 e RC1 determinam a configuração da RAM (como será visto adiante).

DM - *Disk Motor*: Se configurado com 1, liga o motor do drive. Se configurado com 0, desliga o motor.

PST - *Printer STrobe*: sinal usado para controle da impressora.

2FFDh - +3 FDC Status

Escrita: Não?
Leitura: Sim.
Contended: Não.

Leitura de 2FFDh

?b7	b6	b5	b4	b3	b2	b1	b0
?	?	?	?	?	?	?	?

3FFDh - +3 FDC Data

Escrita: Sim.
Leitura: Sim.
Contended: Não.

Leitura/Escrita em 3FFDh

b7	b6	b5	b4	b3	b2	b1	b0
?	?	?	?	?	?	?	?