*** GameBoy Advance Top Secret ***
O Guia dos Fudebas
No Mundo do GameBoy
Versão 0.4b

0) *** O Processador do GameBoy Advance ***

O GameBoy Advance usa um processador ARM, rodando a provavelmente 16.7Mhz, o que da' 16.7 MIPS, por ser um RISC.

Ele roda aproximadamente 1232 ciclos enquanto se desenha uma linha, 280869 ciclos enquanto se desenha um frame, 83776 cliclos durante o VBlank e 272 ciclos no HBlank.

1) *** Organizacao dos periféricos no GameBoy Advance ***

No GameBoy Advance, todos os recursos sao acessiveis por mapeamento em RAM. Desta forma, desde escrever um ponto na tela ate' ler os controles so' pode ser feito atraves de posicoes de memoria especificas `a cada funcao, assemelhando-se a chamar algumas funcoes do BASIC no MSX.

Chama-se este tipo de organizacao de "Perifericos mapeados em Memoria".

Desta forma, o Mapa da Memória e' parte chave e fundamental ao desenvolvimento no GameBoy Advance.

O Processador ARM do GBA e' capaz de enderecar ate' 4Gb de RAM, e os enderecos sao sempre, portanto, de 32 bits.

No entanto, os cartuchos podem ter ate' 16Mb apenas, pois possuem apenas 24 linhas de endereco.

```
   ******************************
2) *** Mapa da Memoria do GBA ***
   ******************************

0x00000000 (16Kbytes)
0x00003FFF
       BIOS - Nao legivel, apenas executavel.

0x02000000 (256Kb)
0x0203FFFF
       Parte da RAM onde o seu jogo vai residir. Aqui ficam
     dados e codigo. RAM Externa.

0x03000000 (32Kb)
0x03007FFF
       Esta parte e' RAM interna.

0x04000000 (1Kb)
0x04000400
       Area de mapeamento dos dispositivos. Todos os
     dispositivos (controles, som, DMA, etc) sao mapeados
     nesta area.

0x05000000 (1Kb)
0x050003FF
       Especificacao das cores de palete, em modo 15 bits.
     Sao 512 cores definidas, As primeiras 256 para os BGs
     e as ultimas 256 para os sprites.

0x06000000 (96Kb)
0x06017FFF
       Esta e' a area da VRAM. Aqui ficam guardados os
     desenhos bitmap. Esses bitmaps sao a imagem
     propriamente dita nos modos bitmap, mas nos modos tile
     sao as definicoes dos tiles (similar `a tabela de
     padroes no MSX). Tambem e' aqui que ficam os padroes
     dos sprites.

0x07000000 (1Kb)
0x070003FF
       Esta e' a regiao de controle dos Sprites do GBA.
     Sempre que for preciso mexer com um sprite, e' nessa
     area que a coisa acontece.

0x08000000 (32Mb)
0x09FFFFFF
       Aqui fica a ROM original do cartucho.
     Ha' mirror dela em
          0x0A000000-0x0BFFFFFF
          0x0C000000-0x0DFFFFFF

0xE0000000 (64Kb por banco)
0x........
       Aqui e' onde fica a SRAM.
```

3) *** O Processador de Video ***

O processador de video do GBA tem alguns modos especificos, cada um util para algum tipo de funcao. Voce muda os modos de video escrevendo no reistro de modo do LCD. Existem 6 modos no GBA, de 0 a 5.

Em absolutamente qquer modo de video voce deve escrever na VRAM sempre em multiplos de 2 bytes. Se tentar escrever 1 byte na VRAM vai ganhar um efeito quadriculado, pois o byte escrito vai ser replicado. Nao e' possivel escrever em quantidades de 8 bits na VRAM.

MODO 0

Neste modo existem:
- 240x160 pontos (30x20 tiles)
- 128 Sprites
- 4 planos de BG (1 tabela de nomes para cada um)
- Modo tiled (8x8)
- Modo Paletado
- Scroll disponivel ate' 64x64 tiles.

MODO 1

Neste modo existem:
- 240x160 pontos (30x20 tiles)
- 128 Sprites
- 3 planos de BG. BG0 e BG1 em tiled normal e BG2 em modo de rotacao/escala
- Modo tiled (8x8)
- Modo Paletado
- Scroll disponivel ate' 64x64 tiles.

MODO 2

Neste modo existem:
- 240x160 pontos (30x20 tiles)
- 128 Sprites
- 2 planos de BG. BG2 e BG3 em modo de rotacao/escala
- Modo tiled (8x8)
- Modo Paletado
- Scroll disponivel ate' 64x64 tiles.

```
+--------+
| MODO 3 |
+--------+
    Neste modo existem:
    - 240x160 pontos
    - 128 Sprites
    - 1 plano de BG - BG2 (1 pagina de video)
    - Modo bitmap (2 bytes - 1 ponto)
    - Modo full-color
    - Scroll disponivel ate' ?

    Mapa de Bits
    ------------
    O mapa de bits comeca em 0x06000000 e tem 75Kb, mas
    apenas metade e' disponivel para apresentacao. Voce
    pode escolher seu inicio em 0x06000000 ou em
    0x06012C00. No mapa de bits, cada ponto esta' no
    formato:

     F E D C B A 9 8 7 6 5 4 3 2 1 0
     X B B B B B G G G G G R R R R R

    R = Vermelho (0 a 31)
    G = Verde (0 a 31)
    B = Azul (0 a 31)

    Planos Ativos
    -------------
    Desligue todos os planos menos o BG2.


+--------+
| MODO 4 |
+--------+
    Neste modo existem:
    - 240x160 pontos
    - 128 Sprites
    - 1 plano de BG - BG2 (2 paginas de video)
    - Modo bitmap (2 bytes, 2 pontos)
    - Modo Paletado (so' 256 cores)
    - Scroll disponivel ate' ?

    Mapa de Bits
    ------------
    A pagina 0 do mapa de bits fica de 0x06000000 a
    0x06009600 e a pagina 1 fica de 0x0600A000 a
    0x06013600.
    No mapa de bits, byte representa um ponto (cor da
    palete) mas eles so' podem ser acessados em dupla (e'
    obrigatorio escrever 2 por vez).

    Planos Ativos
    -------------
    Desligue todos os planos menos o BG2.
```

```
+--------+
| MODO 5 |
+--------+
   Neste modo existem:
   - 160x128 pontos
   - 128 Sprites
   - 1 plano de BG - BG2 (2 paginas de video)
   - Modo bitmap (2 bytes, 1 ponto)
   - Modo Full-Color
   - Scroll disponivel ate' ?

   Mapa de Bits
   ------------
   A pagina 0 do mapa de bits fica de 0x06000000 a
   0x06009FFF e a pagina 1 fica de 0x0600A000 a
   0x06013FFF. No mapa de bits, cada ponto esta' no
   formato:

    F E D C B A 9 8 7 6 5 4 3 2 1 0
    X B B B B B G G G G G R R R R R

   R = Vermelho (0 a 31)
   G = Verde (0 a 31)
   B = Azul (0 a 31)

 ------------------------
 * Controlando o Video *
 ------------------------

- Registrador de MODO (0x04000000) - W
  --------------------------------
   REG_DISPCNT

   O registrador de modo serve para definir o display,
   ou seja, o que e' ou nao mostrado na tela, e o modo
   de operacao do display.

   F E D C B A 9 8 7 6 5 4 3 2 1 0
   O 1 0 S L K J I B D H F A M M M

     1 = Liga janela 1
     0 = Liga janela 0
     D = Mapeamento dos sprites (1=1D, 0=2D)
         (mais informacoes sobre isso na secao de sprites)
     O = Liga display de Sprites
     H = Processar sprites no HBlank (1 desliga)
        (permite mais sprites por linha)
             1: 308 ptos/128 8x8 = 2.40625 ptos por spr
             0: 40 ptos/2.40625 ptos por spr = 99.7 sprs
     M = Modo de video. (0 a 5)
     B = Desliga video
     F = Pagina de Video Ativa (modos Bitmap)
     A = GameBoy Color Mode (deixar em Zero)
     I = Liga BG0
     J = Liga BG1
     K = Liga BG2
     L = Liga BG3
     S = Liga sprites
```

- Registrador de Status (0x04000004) - R/W

  REG_STAT

  Este serve para verificar o status do video.
  F E D C B A 9 8 7 6 5 4 3 2 1 0
  T T T T T T T T X X Y H V Z G W

  W = Status do refresh vertical. Sera' 0 durante
      o VDraw e 1 durante o VBlank.
  G = Status do refresh horizontal. Sera' 0 durante
      o HDraw e 1 durante o HBlank.
  Z = Status da interrupcao do VCount. Fica em 1 quando
      a interrupcao ocorre.
  V = Liga a interrupcao de VBlank. Ela e' desligada
      a cada inicio de VBlank.
  H = Liga a interrupcao de HBlank. Ela e' desligada
      a cada inicio de HBlank.
  Y = Liga a interrupcao de linha por VCount. E'
      desligada quando a linha do VCount e' alcansada
  T = Linha para a interrupcao de VCount disparar.

- Registrador de Linha (0x04000006) - R

  LCY / REG_VCOUNT

  Este registrador e' bastante util, pq mostra qual
linha o processador de video esta' desenhando atualmente.
Ele conta de 0 a 159 enquanto esta' desenhando a tela (160
linhas) e depois conta mais 68 (ate' 227) no tempo de VBlank.
  Espera ele chegar ate' 160 e' um jeito de sincronizar
em 160Hz.

- Registradores de Modo de BG (0x04000008 a 0x040000F) - W

  Estes registradores controlam o modo de cada BG,
Para cada BG temos um endereco diferente:

  BG0 (fundo) - 0x0400008 (REG_BG0)
  BG1   - 0x040000A (REG_BG1)
  BG2   - 0x040000C (REG_BG2)
  BG3 (frente) - 0x040000E (REG_BG3)

  O formato e':

  F E D C B A 9 8 7 6 5 4 3 2 1 0
  Z Z R N N N N N C M 0 0 S S P P

  Z = Tamanho do mapa de tiles
      00 : 256x256 (32x32 tiles)
      01 : 512x256 (64x32 tiles)
      10 : 256x512 (32x64 tiles)
      11 : 512x512 (64x64 tiles)
  R = Para BG2/3, Overflow Rot/Scale. 1=Wrap, 0=Transp.

```
     C = Modo de Cor do palete (0=16*16, 1=256)
     M = Mosaico do BG (1=ligado)
     N = Endereco inicial da tabela de nomes
          0x6000000 + N * 0x800
     S = Endereco inicial da tabela de padroes de tile
          0x6000000 + S * 0x4000
     P = Prioridade (00-primeira, 11-ultima) Quando mesma
         prioridade pra tudo, BG0 e' o primeiro, e BG3 o
         ultimo.
```

- Registradores de Scroll de BG (0x04000010 a 0x040001F) - W

BG	Addr Horz.	Addr Vert.
0	0x04000010 (REG_BG0SCX)	0x04000012 (REG_BG0SCY)
1	0x04000014 (REG_BG1SCX)	0x04000016 (REG_BG1SCY)
2	0x04000018 (REG_BG2SCX)	0x0400001A (REG_BG2SCY)
3	0x0400001C (REG_BG3SCX)	0x0400001E (REG_BG3SCY)

O formato e':

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
X X X X X X S S S S S S S S S S
```

S = Valor do scroll (em pixels)

O registrador controla o pixel do topo superior esquerdo da tela. O BG e' tratado como uma lista sem fim, cujo fim se liga ao comeco. Assim, um BG pode "girar" infinitamente.

- Zoom e Rotacao de Backgrounds

Todos os registros desta secao tem o mesmo formato:

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
S I I I I I I F F F F F F F F F
```

S = Sinal
I = Inteiro
F = Fracao

* 0x04000020 - REG_BG2DX/PA
* 0x04000030 - REG_BG3DX/PA

Escala a imagem na direcao X de uma quantia igual a 1/(valor do registro). O valor 1 da' o tamanho original, 2 e' a metade, .5 e' o dobro. Relativo ao canto superior esquerdo.

* 0x04000022 - REG_BG2DMX/PB
* 0x04000032 - REG_BG2DMX/PB

Rotaciona as coordenadas X do BG com as Y. Um valor 0 vai resultar em nenhuma rotacao. Um valor 1.0 vai fazer a imagem aparecer rotacionada da esquerda (anti-horario), e -1 para a direita (horario). Relativo ao canto superior esquerdo

* 0x04000024 - REG_BG2DY/PC
* 0x04000034 - REG_BG2DY/PC

Rotaciona as coordenadas Y do BG com X. 0 nao rotaciona, 1 rotaciona para baixo (horario) e -1 para cima (anti-horario) Relativo ao canto superior esquerdo

* 0x04000026 - REG_BG2DMY/PD
* 0x04000036 - REG_BG2DMY/PD

Escala o BG na direcao Y de uma quantia igual a 1/(valor do registro). O valor 1 da' o tamanho original, 2 e' a metade, .5 e' o dobro. Relativo ao canto superior esquerdo.

Coordenadas X e Y no plano de rotacao

Os registros seguintes definem a posicao do pixel que aparece como 0,0. Sao similares aos registros de scroll, REG_SCX e REG_SCY, pois tomam seus lugares. (REG_SCX e REG_SCY se desligam quando ligado escala/rotacao do BG).

0x04000028 - REG_BG2X (X para BG2 Rotacional)
0x0400002C - REG_BG2Y (Y para BG2 Rotacional)
0x04000038 - REG_BG3X (X para BG3 Rotacional)
0x0400003C - REG_BG3Y (Y para BG3 Rotacional)

O formato aqui e':

```
 31  29  27  25  23  21  19  17  15  13  11  9   7   5   3   1
 V U T S R Q P O N M L K J I H G F E D C B A 9 8 7 6 5 4 3 2 1
 X X X X S I I I I I I I I I I I I I I I I I I F F F F F F F F
```

S - Sinal
I - Integero
F - Fracao

* As Janelas *

*** Revisar Isso ***

As janelas sao areas especificas que voce seleciona para apresentar apenas alguns tipos de dados ou entao para aplicar alguns efeitos especificos a essas regioes.

Elas sao duas, e a 0 esta' sempre sobre a 1.
Os backgrounds em janelas sempre aparecem em cima dos sprintes nas janelas, nao importando as prioridades.
Se objetos forem desligados no DISPCNT, eles desaparecerao das janelas, nao importando a configuracao das mesmas.

- Regist. de Posição de Janela (0x04000040 a 0x0400047) - W

Janela	Xse/Xid	Yse/Yid
0	0x04000040 (REG_WIN0_H)	0x04000044 (REG_WIN0_V)
1	0x04000042 (REG_WIN1_H)	0x04000046 (REG_WIN1_V)

O Xse e o Yse sao sempre os valores superiores (bits 8 a 15), e o Xid e Yid os bits inferiores (0 a 7).

- Regist. de Diplay Interno da Janela (0x0400048) - W

REG_WIN_IN

Controla a apresentacao dentro das janelas comuns.

O formato e':

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
X X Z J I H G F X X Y E D C B A
```

Y - Liga efeitos de cores especiais para janela 0
A - Mostra sprites dentro da Janela 0
B - Mostra BG3 dentro da Janela 0
C - Mostra BG2 dentro da Janela 0
D - Mostra BG1 dentro da Janela 0
E - Mostra BG0 dentro da Janela 0
Z - Liga efeitos de cores especiais para janela 1
F - Mostra sprites dentro da Janela 1
G - Mostra BG3 dentro da Janela 1
H - Mostra BG2 dentro da Janela 1
I - Mostra BG1 dentro da Janela 1
J - Mostra BG0 dentro da Janela 1

- Regist. de Diplay Externo da Janela (0x040004A) - W

REG_WIN_OUT

Controla a apresentacao fora das janelas e dentro das janelas dos sprites.

O formato e':

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
X X Z J I H G F X X Y E D C B A
```

Y - Liga efeitos de cores especiais fora das janelas
A - Mostra BG0 fora das janela 0 e 1
B - Mostra BG1 fora das janela 0 e 1

```
     C - Mostra BG2 fora das janela 0 e 1
     D - Mostra BG3 fora das janela 0 e 1
     E - Mostra sprites fora das janelas 0 e 1
     Z - Liga efeitos de cores especiais para sprites
     F - Mostra BG0 dentro das janela dos sprites
     G - Mostra BG1 dentro das janela dos sprites
     H - Mostra BG2 dentro das janela dos sprites
     I - Mostra BG3 dentro das janela dos sprites
     J - Mostra sprites dentro das janela dos sprites

  ------------------
  * O Modo Mosaico *
  ------------------

     O processador de video pode "quadricular" tudo pra
voce, no chamado modo mosaico. Com o modo mosaico ligado,
voce pode ajustar o tamano do bloco de mosaico. Pra isso
voce usa o registrador 0x0400004C (REG_MOSAIC). O formato e':

     F E D C B A 9 8 7 6 5 4 3 2 1 0
     V V V V U U U U J J J J I I I I

     I = Tamanho X do mosaico de BG
     J = Tamanho Y do mosaico de BG
     U = Tamanho X do mosaico de Sprite
     V = Tamanho Y do mosaico de Sprite

  ----------------------
  * Os Efeitos de Blend *
  ----------------------

     O Alpha Blend e' um efeito que permite a existencia
de transparencias e brilho nas imagens. Voce precisa setar
alguns registradores para conseguir o efeito desejado. Eles
estao listados abaixo:

     * 0x04000050 (REG_BLDCNT)

     F E D C B A 9 8 7 6 5 4 3 2 1 0
     X X T S R Q P O M M L K J I H G

     G = Blend BG0 (1o. pixel fonte)
     H = Blend BG1 (1o. pixel fonte)
     I = Blend BG2 (1o. pixel fonte)
     J = Blend BG3 (1o. pixel fonte)
     K = Blend sprites (1o. pixel fonte)
     L = Blend backdrop (1o. pixel fonte)
     M = Modo de Blend
         00: Efeitos desligados
         01: Alpha Blend
         10: Fade in
         11: Fade out
     O = Blend BG0 (2o. pixel fonte)
     P = Blend BG1 (2o. pixel fonte)
     Q = Blend BG2 (2o. pixel fonte)
     R = Blend BG3 (2o. pixel fonte)
     S = Blend sprites em alpha
     T = Blend backdrop em alpha
```

Para alpha blend, por exemplo, pode-se usar:

BLDCNT = 0 | BIT06 | A1 | A2

Onde A1 representa os layers que voce quer primeiro,
e A2 os que voce quer depois.

Para Fade,
Escuro:

BLDCNT = 0 | BIT06 | BIT07 | A1;

Claro:

BLDCNT = 0 | BIT07 | A1;

* 0x04000052 - REG_BLDALPHA

Controla o coeficiente Alpha dos pixels. Sao possiveis 31 niveis.

F E D C B A 9 8 7 6 5 4 3 2 1 0
X X X B B B B B X X X A A A A A

A = Coeficiente A, do primeiro pixel fonte
B = Coeficiente B, do segundo pixel fonte

Determina como a mistura de cores vai ocorrer.
Aparentemente se voce usar A=12 e B=4, a cor resultante sera' 12/16 de A e 4/16 de B.

* 0x04000054 - REG_BLDY

Controla a intensidade de fade in ou out.

F E D C B A 9 8 7 6 5 4 3 2 1 0
X X X X X X X X X X F F F F F

F = Valor de escurecimento/iluminamento
Quanto maior o valor, maior o fade.

---
* Os Tiles de Background *

Cada tile tera' sempre 64 bytes para modos de 256 cores, um byte para cada ponto, especificando sua cor (um numero do palete).
No modo 16 cores, temos 2 pontos por byte, sendo que o nibble inferior fica para a cor da esquerda, e os nibble superior para a cor da direita.

Os tiles ficam sempre de 0x06000000 a 0x0600FFFF.

Na Tabela de Nomes (mapa de tiles), cujo endereco base e' 0x06010000, cada tile e' representado por um numero de 16 bits, podendo ter ate' 64x64 tiles no seguinte formato:

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
P P P P V H T T T T T T T T T T
```

P = Numero da palete atual (ignorado se usando palete de 256 cores)
T = Numero do tile (de acordo com tabela de padroes atual)
V = Espelho vertical ativado
H = Espelho horizontal ativado

Sao possiveis ate' 32 tabelas de nomes, das quais apenas 4 estao ativas simultaneamente (uma para cada BG). Obviamente e' possive fazer mais que isso, trocando-se as tabelas de nomes durante a interrupcao de linha. As tabelas de nomes so' podem comecar em intervalos de 0x800 sobre o endereco base.

* As Palettes *

As paletes sao duas: uma para os BGs e outra para os sprites. Ambas podem ser usadas em 2 modos: 16 paletes de 16 cores e 1 palete de 256 cores.

A palete dos BGs fica no endereco 0x05000000 e vai ate' 0x050001FF.

A palete dos Sprites fica no endereco 0x05000200 e vai ate' 0x050003FF.

O formato das cores e' de 16 bits, 0BBBBBGGGGGRRRRR. O Bit 15 nada faz.

A cor zero, tanto dos sprites quanto dos BGs e' semrpe transparente.

* Controlando os Sprites *

O GBA pode ter ate' 128 sprites simultaneos, de 8x8 a 64x64 pixels cada um. Eles podem ser de 256 cores OU de 16 cores. Nao e' possivel ter sprites de 16 cores e sprites de 256 corea so mesmo tempo.

Voce pode ter ate' 128 sprites na mesma linha, se eles forem de 8 pixels de largura, mas apenas 16 se eles tiverem 64 pixels de largura. Em outras palavras, voce pode ter ate' 1024 pixels de sprites em uma linha, nada mal para uma tela de 240 pontos de largura.

Os sprites podem ser "cortados" (clipping) quando rotacionados ou em ZOOM, mas isso pode diminuir o numero maximo de sprites por linha.

Para desligar um sprite, ha' dois jeitos: move-lo para a posicao 160 da tela ou selecionar rotacao de tamanho duplo, com rotacao e escala desligados.

Os sprites podem usar Alpha-Blending com os BGs, mas nao entre si. Nem perca tempo tentando isso. (^=

A tabela de Atributos dos sprites (OAM) comeca em 0x07000000 e tem 1Kb, sendo 8 bytes para cada um dos 128 sprites. Esses 8 bytes formam 4 atributos de 16 bits cada um, o que pode ser facilmente definido como:

```
typedef struct tagSprite
{
unsigned int attribute0;
unsigned int attribute1;
unsigned int attribute2;
unsigned int attribute3;
} Sprite;

Sprite OAM_Copy[128];
```

Isso lhe da' uma matriz de 128 estuturas, uma para cada sprite, de forma a acessar facilmente os dados dos sprites. Ela fica funcionando como uma copia da tabela de sprites real e isso e' util, pois a tabela real fica travada a maior parte do tempo, sendo que voce podera' mexer nela apenas durante o VBlank e o Hblank (e neste ultimo apenas em alguns casos). Assim, com uma copia da OAM, voce pode mexer nela, e durante o VBlank copia-la integralmente sobre a OAM real. Isso tambem evita mudar sprites durante seu desenho, o que certamente causaria efeitos pouco desejaveis.

Os bytes de atributos tem a seguinte configuracao:

Bytes 0 e 1 (Atributo 0)

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
H H C M S S T R J J J J J J J J
```

H = High da selecao de tamanho (mais detalhes abaixo)
- 00 indica sprite quadrado
- 01 indica sprite horizontal
- 10 indica sprite vertical

C = Cores. 1 = 256 cores. 0 = 16 cores
M = Mosaico (1=ligado)
S = Modo de apresentacao:
- 00- normal
- 01- Semi-transparente
- 10- Janela
- 11- Proibido (?)

T = Tamanho da janela em zoom/rot. 1- Normal. 0- Duplo
R = Rotacao/Escala: 1- ligado, 0- Desligado
J = Coordenada Y do sprite (em pixels)

Bytes 2 e 3 (Atributo 1)

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
L L V H R R R I I I I I I I I I
```

R, V e H = Se rot/zoom ligado, selecao de rotacao/zoom.
V = Espelho vertical (se ROT/ZOOM desligado)

```
      H = Espelho horizontal (se ROT/ZOOM desligado)
      I = Coordenada X do sprite.
      L = Low da selecao de tamanho.
          00 indica dimensao 8x8, 16x8 ou 8x16
          01 indica dimensao 16x16, 32x8 ou 8x32
          10 indica dimensao 32x32, 32x16 ou 16x32
          11 indica dimensao 64x64, 64x32 ou 32x64

      Os tamanhos disponiveis sao:

      HHLL  Dim. HHLL  Dim.
      0000: 8  x 8    1000: 8  x 16
      0001: 16 x 16   1001: 8  x 32
      0010: 32 x 32   1010: 16 x 32
      0011: 64 x 64   1011: 32 x 64
      0100: 16 x 8    1100: Not used
      0101: 32 x 8    1101: Not used
      0110: 32 x 16   1110: Not used
      0111: 64 x 32   1111: Not used


      Bytes 4 e 5 (Atributo 2)
      ------------------------

      F E D C B A 9 8 7 6 5 4 3 2 1 0
      C C C C P P T T T T T T T T X

      C = Numero da palete do sprite (desligado em 256 cores)
      P = Prioridade 00- Primeira, 11- Ultima
      T = Numero do tile, de acordo com a tabela de
          indexacao dos tiles. 0 a 1023 nos modos tiled.
          para modos fullcolor apenas os tiles 512 a 1023
          podem ser usados.


      Bytes 6 e 7 (Atributo 3)
      ------------------------

      Controlam o ZOOM e a Rotacao do sprite. O formato e':

      F E D C B A 9 8 7 6 5 4 3 2 1 0
      S I I I I I I I F F F F F F F F

      S = Sinal
      I = Inteiro
      F = Fracao

        Note que essa caracteristica nao e' definida "Sprite a
      Sprite". Sua definicao e' espalhada de 4 em 4 sprites.
         Lembra um pouco o esquema do brilho no YJK. Isso e'
indexado pelos bits H, V e R nos atributos dos bytes 2 e 3
(atributo 1).
      Note que e' tudo relativo ao centro do sprite. (no caso
de backgrounds, e' com relacao ao canto superior esquerdo)

        Fica da seguinte forma, comecando com os bytes 7 e 8
      do sprite 0:
```

```
*** Revisar Isso ***

Sprite 0, bytes 7 e 8 - DX/PA:
Escala o sprite na direcao X de uma quantia igual a
1/(valor do registro). O valor 1 da' o tamanho
original, 2 e' a metade, .5 e' o dobro.

Sprite 1, bytes 7 e 8 - DMX/PB:
Rotaciona as coordenadas X do sprite com as Y. Um valor
0 vai resultar em nenhuma rotacao. Um valor 1.0 vai
fazer a imagem aparecer rotacionada da esquerda,
para baixo (anti-horario) e -1 da direita para baixo
(horario).

Sprite 2, bytes 7 e 8 - DY/PC:
Rotaciona as coordenadas Y do sprite com X. 0 nao
rotaciona, 1 rotaciona de cima para direita (horario)
e -1 de baixo para direita (anti-horario)

Sprite 3, bytes 7 e 8 - DMY/PD:
Escala o sprite na direcao Y de uma quantia igual a
1/(valor do registro). O valor 1 da' o tamanho
original, 2 e' a metade, .5 e' o dobro.

Para fazer o Sprite rodar e escala:
pa=x_escala * cos(angulo)
pb=x_escala * sen(angulo)
pc=x_escala * -sen(angulo)
pd=x_escala * cos(angulo)


- Tabela de Padroes de Sprites
  ----------------------------

  A tabela de padroes de sprites comeca sempre em
  0x06010000 para os modos 0, 1 e 2, e voce tem
  disponivel 1024 tiles de sprites nestes modos.
  Nos modos 3, 4 e 5, A tabela de padroes de sprites
  dve ser usada apenas a partir de 0x06001400, e
  so' se deve referenciar dos tiles de 512 a 1023.
  Todos os tiles de sprites sao sempre de 8x8.
  Cada sprite tem 64 bytes, casa byte representando
  uma cor do palete de sprites.

- Palete de Sprites
  -----------------
  A palete usada e' a que fica no endereco 0x05000200
  (palete de 15 bits, 0BBBBBGGGGGRRRRR). Sao 256 cores.

- Sprites maiores que 8x8
  ----------------------
  Quando voce selecionar sprites maiores que 8x8, voce
  indica apenas 1 numero de padrao. Os padroes
  subsequentes serao usados para fazer o desenho
  completo. Dependendo de escolha de modelo 1D ou 2D o
  resultado sera' diferente.

  Para 1D eles serao pegos sequencialmente.Por exemplo,
```

para um sprite 32x32, comecando no padrao 5, temos:

```
+--------------------
| 05 | 06 | 07 | 08 |
+--------------------+
| 09 | 10 | 11 | 12 |
+--------------------+
| 13 | 14 | 15 | 16 |
+--------------------+
| 17 | 18 | 19 | 20 |
+--------------------+
```

Para 2D eles serao pegos sequencialmente numa linha,
e somado 32 para a segunda linha. Por exemplo,
para um sprite 32x32, comecando no padrao 5, temos:

```
+--------------------
| 05 | 06 | 07 | 08 |
+--------------------+
| 37 | 38 | 39 | 40 |
+--------------------+
| 69 | 70 | 71 | 72 |
+--------------------+
| 101| 102| 103| 104|
+--------------------+
```

```
   ******************************
4) *** Os Processadores de Som ***
   ******************************
```

Note: I've obtained this info (much of it verbatim) from BeLogic's unofficial GBA sound info site, which gives a more thorough explanation as
well as some sample source code and demos.

Address: 0x04000060 - REG_SOUND1CNT_L (Sound 1 Sweep control.)
Address: 0x04000062 - REG_SOUND1CNT_H (Sound 1 Length, wave duty and envelope control)
Address: 0x04000064 - REG_SOUND1CNT_X (Sound 1 Frequency, reset and loop control)
Address: 0x04000068 - REG_SOUND2CNT_L (Sound 2 Length, wave duty and envelope control)
Address: 0x0400006C- REG_SOUND2CNT_H (Sound 2 Frequency, reset and loop control)

I still don't know the functionality of these. Most likely I will have to wait for BeLogic to update their site with info before I can complete this section.

Address: 0x04000070 - REG_SOUND3CNT_L (Sound 3 Enable and wave ram bank control)

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
X X X X X X X X N S M X X X X X
```

M = Bank Mode (0 - 2 32 sample banks, 1 - 1 64 sample bank)

S = Bank Select. Controls which bank is active for playing/reloading. If set to 0, samples are played from bank 0 and writing to the Wave Ram will store the data in Bank 1, and vice-versa.
N = Sound Channel 3 output enable. When this is set and bit 15 from REG_SOUND3CNT_X is set, the sound starts to play.

Sound channel 3 is a circuit that can produce an arbitrary wave pattern.
Samples are 4 bit, 8 samples per word, and are located in Wave Ram registers from 0x400090 to 0x40009F. The Wave Ram is banked, providing the ability to play a 64 samples pattern or to select between two 32 samples
patterns (Bit 5). Sound channel 3 always produces some audio artifacts (distortion) when sound is initialized. Fortunately, switching banks does not require re-initialisation during playback, thus allowing for dynamic reloading of the Wave Ram without generating any distortion.

Both banks of Wave Ram are filled with zero upon initialization of the Gameboy, Bank 0 being selected. So writing to bank 0 implies setting bit 6 to 1 before loading Wave Ram then set it back to 0 to play it.

Address: 0x04000072 - REG_SOUND3CNT_H (Sound 3 Sound length and output level control)

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
R R R X X X X L L L L L L L L
```

L = Sound length. The sound length is an 8 bit value obtained from the following formula:

Register=Note length (in seconds)*256

Hence a 1 second maximum and a 3.9 millisecond minimum sound duration. After the sound length has be changed, the sound channel must be resetted via bit F of REG_SOUND3CNT_X.

R = Output volume ratio:

000 - Mute
001 - 100%
100 - 75%
010 - 50%
011 - 25%

Address: 0x04000074 - REG_SOUND3CNT_X (Sound 3 Frequency, reset and loop control)

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
R T X X X F F F F F F F F F F F
```

F = Sound frequency. The minimum frequency is 64Hz and the maximum is 131Khz. Can be calculated from the following formula:

F(hz)=4194304/(32*(2048-register value)).

T = Timed sound. When set to 0, sound 3 is played continuously regardless of the length data in REG_SOUND3CNT_H. When set to 1, sound is played for that specified length and after that, bit 2 of REG_SOUNDCNT_X is reset.

R = Sound reset. When set, sound resets and restarts at the specified frequency. Frequency and sound reset must be perfomed in a single write since both are write only. In continuous mode, frequency can be changed without resetting the sound channel.

Address: 0x04000078 - REG_SOUND4CNT_L (Sound 4 Length, output level and envelope control)
Address: 0x0400007C - REG_SOUND4CNT_H (Sound 4 Noise parameters, reset and loop control)

As with Sound channels 1 and 2, I still don't know the functionality of these. Most likely I will have to wait for BeLogic to update their site with info before I can complete this section.

Address: 0x04000080 - REG_SOUNDCNT_L (Sound 1-4 Output level and Stereo control)

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
R Q P O N M L K J I I I H G G G
```

G = DMG Left Volume
H = Vin Left on/off (?) - According to BeLogic, Vin on/off allowed the original GameBoy paks to provide their own sound source. It is unkown whether they still work on a GBA.
I = DMG Right Volume
J = Vin Right on/off (?)
K = DMG Sound 1 to left output
L = DMG Sound 2 to left output
M = DMG Sound 3 to left output
N = DMG Sound 4 to left output
O = DMG Sound 1 to right output
P = DMG Sound 2 to right output
Q = DMG Sound 3 to right output
R = DMG Sound 4 to right output

This register controls only the DMG output amplifiers and have no effects on the individual sound channels' processing, or Direct Sound channels' volume.

Address: 0x04000082 - REG_SOUNDCNT_H (Direct Sound control and Sound 1-4 output ratio)

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
Q P O N M L K J X X X X I H G G
```

G = Output Sound Ratio for channels 1-4.
- 00 - 25%
- 01 - 50%
- 10 - 100%
- 11 - ??

H = Direct sound A output ratio (0 - 50%, 1 - 100%)
I = Direct sound B output ratio (0 - 50%, 1 - 100%)

```
J = Direct Sound A to right output
K = Direct sound A to left output
L = Direct sound A Sampling rate timer (timer 0 or 1). Use this to
set which timer contorls the playback frequency.
M = Direct sound A FIFO reset
N = Direct sound B to right output
O = Direct sound B to left output
P = Direct sound B Sampling rate timer (timer 0 or 1). Use this to
set which timer contorls the playback frequency.
Q = Direct sound B FIFO reset
```

This register is used in controlling Direct Sound on the GBA. Output ratios control the volume, in percentage, that gets output to the speakers.

Address: 0x04000084 - REG_SOUNDCNT_X (Master sound enable and Sound 1-4 play status)

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
X X X X X X X N X X X J I H G
```

```
G = DMG Sound 1 Status (Read only). 0 - Stopped, 1 - Playing
H = DMG Sound 2 Status (Read only). 0 - Stopped, 1 - Playing
I = DMG Sound 3 Status (Read only). 0 - Stopped, 1 - Playing
J = DMG Sound 4 Status (Read only). 0 - Stopped, 1 - Playing
N = All Sound circuit enable
```

This register is used to monitor the play status of sounds and to turn on or off all sound circuits. Turning the sound circuits off saves battery power, allowing them to last up to 10% longer.

Address: 0x04000088 - REG_SOUNDBIAS (Sound bias and Amplitude resolution control)

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
R R X X X X X B B B B B B B B B
```

```
B = PWM bias value, controlled by the BIOS.
R = Amplitude resolutions

     00 - 9 bit at 32768 hz
     01 - 8 bit at 65536 hz
     10 - 7 bit at 131072 hz
     11 - 6 bit at 262144 hz
```

The BIAS setting is used to offset the sound output and bring it back into a signed range. When the BIOS starts up, it runs a timing loop where it slowly raises the BIAS voltage from 0 to 512. This setting should not be changed. At best, the sound will become distorted. At worst the amplifier inside the GBA could be damaged.

When accessing bits F-E, a read-modify-write is required. The default value for bits F-E is 00. Most if not all games use 01 for this setting.

Address: 0x04000090 - REG_WAVE_RAM0_L (Sound 3 samples 0-3)
Address: 0x04000092 - REG_WAVE_RAM0_H (Sound 3 samples 4-7)
Address: 0x04000094 - REG_WAVE_RAM1_L (Sound 3 samples 8-11)

```
Address: 0x04000096 - REG_WAVE_RAM1_H (Sound 3 samples 12-15)
Address: 0x04000098 - REG_WAVE_RAM2_L (Sound 3 samples 16-19)
Address: 0x0400009A - REG_WAVE_RAM2_H (Sound 3 samples 20-23)
Address: 0x0400009C - REG_WAVE_RAM3_L (Sound 3 samples 23-27)
Address: 0x0400009E - REG_WAVE_RAM3_H (Sound 3 samples 28-31)

These registers together contain four (4 bytes each) 4-bit wave RAM
samples for Sound channel 3.

Address: 0x040000A0 - REG_FIFO_A_L (Direct Sound channel A samples
0-1)
Address: 0x040000A2 - REG_FIFO_A_H (Direct Sound channel A samples
2-3)
Address: 0x040000A4 - REG_FIFO_B_L (Direct Sound channel B samples
0-1)
Address: 0x040000A6 - REG_FIFO_B_H (Direct Sound channel B samples
2-3)

These are the locations of the Direct Sound 8-bit FIFO samples, from
which Direct Sound pulls the music data to be played on the
speakers. Note that there are only 8 bytes total for all your
samples. You repeatedly fill these from a buffer of your own using
DMA0 or DMA1, or by using timer interrupts.

To fill them using DMA, first set Timer 0 or Timer 1 to refresh at
the appropriate sample rate (for example, 16khz). Next, set the DMA
source address to a sound sample in memory, and the destination
address to one of these FIFO registers. Use REG_SOUNTCNT_H to reset
FIFO and tell Direct Sound to get its sampling rate from Timer 0 or
Timer 1. Finally, set the
DMA control register to start on FIFO empty (start mode 11) and to
repeat, then enable the timers. All of this will cause the hardware
to play sound samples in FIFO at the rate specified in your timer,
and automatically refill them using DMA.

To fill these using interrupts, follow a similar process, but
instead of using DMA, set the clock to interrupt on overflow. When
using interrupts instead of DMA, BeLogic recommends setting the
timer divider to 1024 and start the timer at 0x0000 order to get a
sampling rate of 16.384 khz. This apparently causes less distortion
than if you simply set the start time of the clock to 0xFFFF -
(2^24/16000).

Note that reading from these registers can yield unpredictable
results. It might be interesting to see just how unpredictable...
```

```
   *************************
5) *** A entrada de Dados ***
   *************************
```

A entrada de dados tambem e' feita por posicoes de memoria. Tudo e' feito em uma posicao de memoria (16 bits), e essa posicao e': 0x04000098 - W

A estrutura e':

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
X X X X X J I D U L R S E B A
```

A = Botao A
B = Botao B
E = Botao Select
S = Botao Start
R = Direita
L = Esquerda
U = Cima
D = Baixo
I = Botao R
J = Botao L

Vale lembrar que 0 indica uma tecla pressionada, e 1 indica tecla NAO pressionada.

- Setando Interrupcoes para as Teclas

Voce pode indicar as treclas para acinoar interrupcoes tambem. Para tanto, voce precisa definir adequadamente o REG_IE e o REG_IME, alem de definir este outro registro, que e' o Keyboard Control:

0x04000132 - REG_KEYCNT

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
T K X X X X J I D U L R S E B A
```

A = Botao A
B = Botao B
E = Botao Select
S = Botao Start
R = Direita
L = Esquerda
U = Cima
D = Baixo
I = R
J = L
K = Gerar interrupcoes ao apertar teclas
G = Tipo de interrupcao
  0: Interrupcao gerada para qquer uma das teclas indicadas.
  1: Interrupcao gerada se TODAS as teclas indicadas estiverem pressionadas.

```
   ************************************
6) *** Transferencia de Dados Via DMA ***
   ************************************
```

O GBA tem 4 canais de DMA, de 0 a 3, sendo a prioridade decrescente (0>1>2>3). Apenas um DMA pode ocorrer de cada vez, e ele paralisa a CPU (o ARM), mas isso nao impede de um outro DMA ser disparado, pois voce podera' "pendurar" um DMA em um evento nao dependente do processador (como um VBlank). Neste caso, se um DMA de maior prioridade for disparado, o de menor prioridade sera' interrompido, o de maior prioridade sera' processado e entao o de menor prioridade sera' continuado.

Se ocorrer alguma interrupcao durante o DMA, ela ficara "pending" e sera' processada apenas ao fim do DMA.

```
   ---------
   * DMA 0 *
   ---------
```

O DMA0 e' controlado atraves de 4 registradores. Um indica a fonte dos dados, outro indica o destino dos dados, outro indica o numero de blocos a transferir e, finalmente, um controla o DMA0.

O DMA0 e' o DMA de maxima prioridade.

O DMA0 serve apenas para transferir dados da RAM interna para a RAM interna, nao sendo util, portanto, para copiar dados da ROM para a RAM. Util, porem, para atualizar telas no MODO3, por exemplo, quando grandes quantidades de dados precisam ser enviadas, possivelmente da RAM para a VRAM. Um outro uso e' usa-lo no HBlank para mexer nos dados de Zoom/Rotacao dos planos de video, mas deve ser usado com cuidado, ou fara' com que linhas erradas aparecem nos lugares errados.

Os registros sao:

```
Fonte:          0x040000B0 (REG_DMA0SAD - 27 bits)
Destino:   0x040000B4 (REG_DMA0DST - 27 bits)
Contador:  0x040000B8 - W  (REG_DMA0SIZE - 14bits)
Controle:  0x040000BA - W  (REG_DMA0CNT - 2 bytes)
```

Este controle de DMA permite que ate' 256Kb seja transferido de uma area para outra da memoria, desde que seja sempre memória interna.

Ele pode ser utilizado de forma a gerar uma interrupcao ao seu final, e tambem ter uma hora especifica para ser disparado. Ele tambem pode ser usado para preencher a memoria com um valor constante. O endereco pode ou nao ser incrementado com cada copia, assim, ao fim da transferencia, ele pode continuar o mesmo... ou nao. Voce escolhe.

Para realizar uma transferencia, primeiro defina os enderecos fonte e destino, o contador e depois defina o de controle. O registrador de controle tem o seguinte formato:

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
N I T T X W R B B A A X X X X X
```

```
      N = Ao ser setado, inicia o DMA
      I = Indica para interrupcao ser gerada ao fim do DMA
      T = Indica hora de comecar o DMA.
          00- Inicia imediatamente
          01- Inicia no proximo VBlank
          10- Inicia no proximo HBlank
          11- ? Transferencia inicia no inicio da
              renderizacao da linha?
      W - Tamanho do bloco. Se ligado, transfere dados de
          32 bits (blocos de 4 bytes). Se deligado, transfere
          dados de 16 bits (blocos de 2 bytes).
      R - Indica pro DMA se repetir ao fim da transferencia,
          se T for 1 ou 2.
      B = Incremento aplicado no endereco fonte:
          00 - Incrementa depois de cada copia
          01 - Decrementa apos cada copia
          10 - Nao muda
          11 - Proibido
      A = Forma de incremento do destino:
          00 - Incrementa depois de cada copia
          01 - Decrementa apos cada copia
          10 - Nao muda
          11 - Incrementa depois de cada copia e, ao fim,
               endereco original e' restaurado.
```

Se ligado o auto-incremento, o endereco sera' atualizado de 2 ou 4 bytes, dependendo do tamanho escolhido.

```
   -----------------
   * DMA 1 e DMA 2 *
   -----------------
```

O DMA1 e DMA2 sao controlados atraves de 4 registradores cada. Um indica a fonte dos dados, outro indica o destino dos dados, outro indica o numero de blocos a transferir e, finalmente, um para controlar o referido canal.

Esses DMAs servem tanto para transferir dados da RAM interna para a RAM interna, como da RAM/ROM externa para a interna. Em geral utilizados para tocar musica com o Direct Sound.

Os registros sao (DMA1):

```
Fonte:          0x040000BC (REG_DMA1SAD - 28 bits)
Destino:   0x040000C0 (REG_DMA1DST - 27 bits)
Contador:  0x040000C4 - W  (REG_DMA1SIZE - 14bits)
Controle:  0x040000C6 - W  (REG_DMA1CNT - 2 bytes)
```

Quando em modo FIFO, o contador fica desabilitado.

Os registros sao (DMA2):

```
Fonte:          0x040000C8 (REG_DMA2SAD - 28 bits)
Destino:   0x040000CC (REG_DMA2DST - 27 bits)
Contador:  0x040000D0 - W  (REG_DMA2SIZE - 14bits)
Controle:  0x040000D2 - W  (REG_DMA2CNT - 2 bytes)
```

Quando em modo FIFO, o contador fica desabilitado.

Este controle de DMA permite que ate' 256Kb seja transferido de uma area para outra da memoria, desde que seja sempre memória interna.
Ele pode ser utilizado de forma a gerar uma interrupcao ao seu final, e tambem ter uma hora especifica para ser disparado. Ele tambem pode ser usado para preencher a memoria com um valor constante. O endereco pode ou nao ser incrementado com cada copia, assim, ao fim da transferencia, ele pode continuar o mesmo... ou nao. Voce escolhe.

Para realizar uma transferencia, primeiro defina os enderecos fonte e destino, o contador e depois defina o de controle. O registrador de controle tem o seguinte formato:

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
N I T T X W R B B A A X X X X X
```

N = Ao ser setado, inicia o DMA
I = Indica para interrupcao ser gerada ao fim do DMA
T = Indica hora de comecar o DMA.
    00- Inicia imediatamente
    01- Inicia no proximo VBlank
    10- Inicia no proximo HBlank
    11- Instrui o DMA a se repetir quando a fila FIFO
      se esvaziar. Uasado para Direct Sound.
W - Tamanho do bloco. Se ligado, transfere dados de
    32 bits (blocos de 4 bytes). Se deligado, transfere
    dados de 16 bits (blocos de 2 bytes).
R - Indica pro DMA se repetir ao fim da transferencia,
    se T for 1 ou 2.
B = Incremento aplicado no endereco fonte:
    00 - Incrementa depois de cada copia
    01 - Decrementa apos cada copia
    10 - Nao muda
    11 - Proibido
A = Forma de incremento do destino:
    00 - Incrementa depois de cada copia
    01 - Decrementa apos cada copia
    10 - Nao muda
    11 - Incrementa depois de cada copia e, ao fim,
         endereco original e' restaurado.

Se ligado o auto-incremento, o endereco sera' atualizado de 2 ou 4 bytes, dependendo do tamanho escolhido.

---
* DMA 3 *

O DMA3 e' controlado atraves de 4 registradores. Um indica a fonte dos dados, outro indica o destino dos dados, outro indica o numero de blocos a transferir e, finalmente, um controla o DMA3.
O DMA3 e' o DMA de menor prioridade.
O DMA3 em geral e' usado para transferir dados da ROM para a RAM.

Os registros sao:

```
Fonte:     0x040000D4 (REG_DMA3SAD - 28 bits)
Destino:   0x040000D8 (REG_DMA3DST - 28 bits)
Contador:  0x040000DC - W  (REG_DMA3SIZE - 14bits)
Controle:  0x040000DE - W  (REG_DMA3CNT - 2 bytes)
```

Este controle de DMA permite que ate' 256Kb seja transferido de qquer lugar para qquer lugar. Ele tambem pode ser usado para preencher a memoria com um valor constante. O endereco pode ou nao ser incrementado com cada copia, assim, ao fim da transferencia, ele pode continuar o mesmo... ou nao. Voce escolhe.

Para realizar uma transferencia, primeiro defina os enderecos fonte e destino e depois defina o de controle. O registrador de controle tem o seguinte formato:

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
N I T T G W R B B A A X X X X X
```

N = Ao ser setado, inicia o DMA
I = Indica para interrupcao ser gerada ao fim do DMA
T = Indica hora de comecar o DMA.
    00- Inicia imediatamente
    01- Inicia no proximo VBlank
    10- Inicia no proximo HBlank
    11- ? Transferencia inicia no inicio da renderizacao da linha?
G = Indica requisicao de dados ao GamePak
W = Tamanho do bloco. Se ligado, transfere dados de 32 bits (blocos de 4 bytes). Se deligado, transfere dados de 16 bits (blocos de 2 bytes).
R - Indica pro DMA se repetir ao fim da transferencia, se T for 1 ou 2.
B = Incremento aplicado no endereco fonte:
    00 - Incrementa depois de cada copia
    01 - Decrementa apos cada copia
    10 - Nao muda
    11 - Proibido
A = Forma de incremento do destino:
    00 - Incrementa depois de cada copia
    01 - Decrementa apos cada copia
    10 - Nao muda
    11 - Incrementa depois de cada copia e, ao fim, endereco original e' restaurado.

Se ligado o auto-incremento, o endereco sera' atualizado de 2 ou 4 bytes, dependendo do tamanho escolhido.

```
   *********************
7) *** A Interrupcao ***
   *********************
```

O GBA tem interrupcoes para quase qualquer atividade. Elas sao controladas por 3 registros basicos, um que permite ligar/desligar uma por uma, outro que permite saber o estado delas, e um terceiro que permite ligar/desligar todas as interrupcoes.

O efeito normal de interrupcao do ARM e' pular para o endereco 0x03007FFC. O codigo ai' deve ser 100% ARM, nada de thumb. Para retornar, a instrucao e' BX LR.

Quando desejar desligar uma interrupcao especifica, voce deve desligar o IME antes de mexer no IE, para evitar que uma interrupcao ocorra DURANTE a sua alteracao. Religue o IME logo em seguida.

- (REG_IE) Registro de Habilitacao da Inter. (0x04000200) - W

Cada bit desde registro habilita uma interrupcao:

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
X X G T D C B A S I G F E J H V

G = DREQ/IREQ (GamePak)
T = Teclas
A = DMA0
B = DMA1
C = DMA2
D = DMA3
S = Serial/Geral/Joystick/UART (Resumindo, comunicacao)
E = Timer 0
F = Timer 1
G = Timer 2
I = Timer 3
J = Interrupcao do contador do 0x04000004
H = Interrupcao do HBlank
V = Interrupcao do VBlank
```

- (REG_IF) Registro de Flags da Inter. (0x04000202) - R

Este registro serve para verificar quais interrupcoes estao habilitadas. Ele e' read-only, e e' o complemento do registro de habilitacao da interrupcao (entao, interrupcoes ligadas estarao representadas como 0)

A configuracao e' a mesma do habilitador de interrupcao:

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
X X G T D C B A S I G F E J H V

G = DREQ/IREQ (GamePak)
T = Teclas
A = DMA0
B = DMA1
C = DMA2
D = DMA3
S = Serial/Geral/Joystick/UART (Resumindo, comunicacao)
E = Timer 0
F = Timer 1
G = Timer 2
I = Timer 3
J = Interrupcao do contador VCOUNT
H = Interrupcao do HBlank
V = Interrupcao do VBlank
```

- (REG_IME) Registro Meste da Interrupcao (0x04000208) - W

Este registro serve para ligar/desligar todas as interrupcoes de uma unica vez. Nao ha' NMI. O unico bit que existe e' o bit 0. Colocando ele em 1, liga-se as interrupcoes, colocando em 0, desliga-se.

8) *** Os Timers ***

O GBA tem 4 timers. Cada um deles tem um registro de dados onde voce pode ler o valor atual e outro de controle. Todos os 4 sao absolutamente identicos. Todos sao de 16 bits. No entanto, os Timers 0 e 1 sao para controle da taxa de amostragem do FIFO do Direct Sound.

Os registros de dados sao:

```
0x04000100 - REG_TM0DAT (Timer 0)
0x04000104 - REG_TM1DAT (Timer 1)
0x04000108 - REG_TM2DAT (Timer 2)
0x0400010C - REG_TM3DAT (Timer 3)
```

E seguem o formato:

```
F E D C B A 9 8 7 6 5 4 3 2 1 0
D D D D D D D D D D D D D D D D
```

D = Valor atual do timer

Estes registros sao R/W. O padrao e' comecar contando a partir de zero, mas voce pode mudar isso mexendo nos valores destes contadores.

Quando usados os Timers 0 e 1 em conjunto com DMA 1 e 2 em modo de operacao com inicio quando o FIFO esvaziar, eles podem causar o re-enchimento do buffer com dados. Para selecionar a frequencia de playback em hz, escreva o valor

0xFFFF - (2^24/Freq de Plaback em hz)

neste registro. Isso indica o valor no qual o contador vai estourar precisamente quando o proximo sample for necessario, e causar o DMA a ativar.
Quando usar interrupcoes, acerte o valor inicial para 0, mas use o REG_TMxCNT para mudar a frequencia de atualizacao para 1024, causando uma interrupcao de 16.384Khz.

- Mudando a Frequencia de Atualizacao do Timer

Para este fim, voce usa os seguintes registros de controle:

0x04000102 - REG_TM0CNT (Timer 0)
0x04000106 - REG_TM1CNT (Timer 1)
0x0400010A - REG_TM2CNT (Timer 2)
0x0400010E - REG_TM3CNT (Timer 3)

O Formato deles e':

F E D C B A 9 8 7 6 5 4 3 2 1 0
X X X X X X X E I X X C F F

F = Frequencia de atualizacao do Timer
    00: Normal - 16.78MHz (~17mlns ticks/s)
    01: A cada 64 clocks - ~262187.5Hz
    10: A cada 256 clocks - ~65546.875Hz
    11: A cada 1024 clocks - ~16386.71875Hz
C = Cascade. Neste modo, a frequencia e' ignorada.
    O timer sera' atualizado quando o anterior estourar.
I = Gera interrupcao no overflow
E = Liga o timer

9) *** A BIOS ***

A BIOS do GBA e' chamada atraves de Interrupts de SoftWare (instrucao SWI do Assembly). O valor passado `a instrucao define qual vai ser a funcao chamada. As funcoes existentes na BIOS sao:

0x00 - SoftReset
Reset por software
0x01 - RegisterRamReset
Limpa registradores em RAM(?)
0x02 - Halt
Espera proxima interrupcao
0x03 - Stop
?

```
0x04 - IntrWait
      Espera interrupcao?
0x05 - VBlankIntrWait
      Espera Int de VBlank
0x06 - Div
      Divisao.
      Entrada:
      - R0=Numerador
      - R1=Denominador
      Saída:
      - R0 = Resultado da operacao
      - R1 = Resto da divisao
      - R3 = Valor absoluto da divisao (sem sinal)
0x07 - DivArm
      ?
0x08 - Sqrt
      Raiz Quadrada.
      Entrada:
      - R0 = Numero
      Saída:
      - R0 = Raiz quadrada do numero
0x09 - ArcTan
      Arco tangente
      Entrada:
      - R0 = Angulo (em radianos?)
      Saida:
      - R0 = Arcotangente do angulo
0x0A - ArcTan2
      Arco tangente
      Entrada:
      - R0 = Angulo (em graus?)
      Saida:
      - R0 = Arcotangente do angulo
0x0B - CPUSet
      Especie de DMA
      Entrada:
      - R0 = Endereco fonte
      - R1 = Endereco destino
      - R2 = Controle (Bit26 1=32bit, 0=16 bit, 0 a 15 e'
           o numero de transferencias?)
0x0C - CPUFastSet
      Especie de DMA
      Entrada:
      - R0 = Endereco fonte
      - R1 = Endereco destino
      - R2 = Controle (Bit26 1=32bit, 0=16 bit, 0 a 15 e'
           o numero de transferencias?)
0x0D - ?
0x0E - BgAffineSet
0x0F - ObjAffineSet
0x10 - BitUnPack
      Descompressor de bits?
0x11 - LZ77UnCompWRAM
      Descompressor LZ77 para a RAM
      Entrada:
      - R0 = Endereco fonte
      - R1 = Endereco destino
```

```
0x12 - LZ77UnCompVRAM
      Descompressor LZ77 para a VRAM
      Entrada:
      - R0 = Endereco fonte
      - R1 = Endereco destino
0x13 - HuffUnComp
      Descompressor Huffman,
      Entrada:
      - R0 = Endereco fonte
      - R1 = Endereco destino
0x14 - RLUnCompWRAM
      Descompressor RLE para RAM
      Entrada:
      - R0 = Endereco fonte
      - R1 = Endereco destino
0x15 - RLUnCompVRAM
      Descompressor RLE para VRAM
      Entrada:
      - R0 = Endereco fonte
      - R1 = Endereco destino
0x16 - Diff8bitUnFilterWRAM
0x17 - Diff8bitUnFilterVRAM
0x18 - Diff16UnFilter
0x19 - SoundBiasChange
0x1A - SoundDriverInit
0x1B - SoundDriverMode
0x1C - SoundDriverMain
0x1D - SoundDriverVsync
      Sincroniza audio com video
0x1E - SoundChannelClear
      Limpa canal de som
0x1F - MIDIKey2Freq
      Converte informacao MIDI para frequencia
0x20 - MusicPlayerOpen
0x21 - MusicPlayerStart
0x22 - MusicPlayerStop
0x24 - MusicPlayerFadeOut
0x25 - MultiBoot
0x26 - ?
0x27 - ?
0x28 - SoundDriverVsyncOff
      Desliga sincronia de audio com video
0x29 - SoundDriverVsyncOn
      Liga sincronia de audio com video
?    - FIQMasterEnable
```