VRAM Transfer Manager

Version 1.0.0

NTR-06-0141-001-A2
Released: November 15, 2005

Table of Contents

Code

Tables

Figures

Revision History

Version	Revision Date	Description
1.0.0	01/05/2005	Changed an instance of “NITRO” to “Nintendo DS.”
0.1.0	07/16/2004	Initial version.

1 Introduction

The Nintendo DS does not allow the contents of VRAM to be rewritten during a screen display period. To rewrite VRAM contents while displaying a screen with an application, the application should normally be programmed to write data to the VRAM during V-Blank periods. The VRAM transfer manager in the NITRO-System library provides a feature that holds VRAM rewrite requests from applications in queue so the requested data can be written to VRAM during a V-Blank period.

2 VRAM Transfer Manager

2.1 Overview

The VRAM transfer manager provides features that register user requests to rewrite VRAM to the queue as VRAM transfer tasks, and at a later time write the data to the VRAM according to the registered tasks. These features are used to write data to VRAM during a V-Blank period according to VRAM transfer tasks registered during a screen display period.

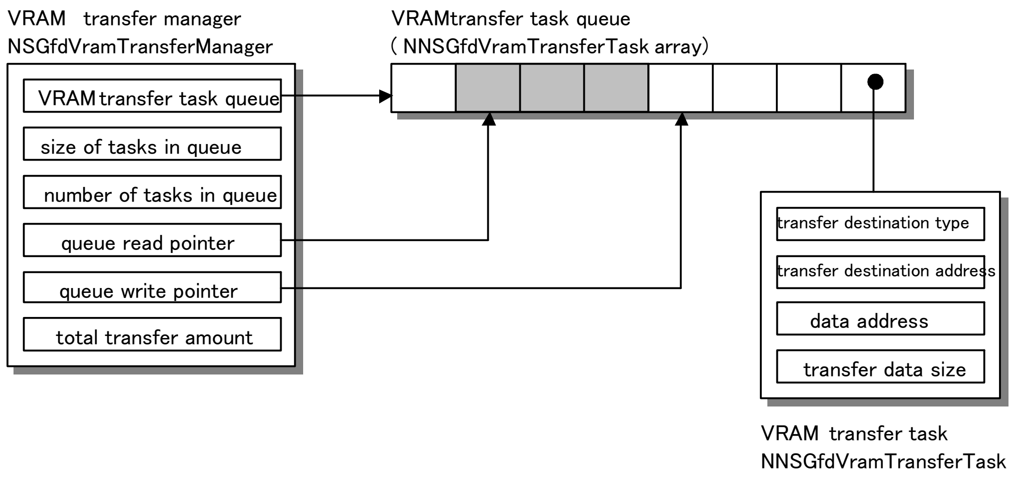

Figure 2-1 Diagram of the VRAM Manager

2.2 Initializing the VRAM Transfer Manager

Before using the VRAM transfer manager, it must be initialized. Call the following function to initialize the VRAM transfer manager.

```
void NNS_GfdInitVramTransferManager(
      NNSGfdVramTransferTask* pTaskArray, u32 lengthOfArray);
```

The `NNS_GfdInitVramTransferManager()` function initializes the VRAM transfer manager and sets it to an available state.

The VRAM transfer manager saves VRAM transfer tasks in an array of `NNSGfdVramTransferTask` structure. When initializing the VRAM transfer manager, the user needs to prepare the `NNSGfdVramTransferTask` structure array. The user needs to set a pointer to the array to the first argument `pTaskArray` of the `NNS_GfdInitVramTransferManager()` function and the size of the array to the second argument `lengthOfArray`.

16 bytes are required to register one VRAM transfer task. Prepare the size of the `NNSGfdVramTransferTask` structure array to match the maximum number of VRAM transfer tasks to register to the VRAM transfer manager at once.

Code 2-1 Initializing the VRAM Manager

```
#define NUM_TASKS    8

NNSGfdVramTransferTask    taskArray[NUM_TASKS];

NNS_GfdInitVramTransferManager(taskArray, NUM_TASKS);
....
```

2.3 Registering a VRAM Transfer Task

Call the following function to register a VRAM transfer task to the VRAM transfer manager.

```
BOOL NNS_GfdRegisterNewVramTransferTask(
     NNS_GFD_DST_TYPE type, u32 dstAddr, void* pSrc, u32 szByte);
```

The `NNS_GfdRegisterNewVramTransferTask()` function sequentially registers specified VRAM rewrite data to the VRAM transfer manager queue as a VRAM transfer task. The number of VRAM transfer tasks that can be registered to the VRAM transfer manager at once is the number of elements in the `NNSGfdVramTransferTask` structure array specified when initializing the VRAM transfer manager.

For the `type` argument of the data transfer destination type, specify what type of data to transfer according to the VRAM transfer manager. The following types can be specified for `type`.

Table 2-1 Data Transfer Destination Types

Category	Constants Indicating the Data Transfer Destination Type	Meaning
3D	NNS_GFD_DST_3D_TEX_VRAM	3D texture image rewrite
	NNS_GFD_DST_3D_TEX_PLTT	3D texture palette rewrite
	NNS_GFD_DST_3D_CLRIMG_COLOR	3D clear image color rewrite
	NNS_GFD_DST_3D_CLRIMG_DEPTH	3D clear image depth rewrite
2D main	NNS_GFD_DST_2D_background0_CHAR_MAIN	2D main BG0 character rewrite
	NNS_GFD_DST_2D_background1_CHAR_MAIN	2D main BG1 character rewrite
	NNS_GFD_DST_2D_background2_CHAR_MAIN	2D main BG2 character rewrite
	NNS_GFD_DST_2D_background3_CHAR_MAIN	2D main BG3 character rewrite
	NNS_GFD_DST_2D_background0_SCR_MAIN	2D main BG0 screen rewrite
	NNS_GFD_DST_2D_background1_SCR_MAIN	2D main BG1 screen rewrite
	NNS_GFD_DST_2D_background2_SCR_MAIN	2D main BG2 screen rewrite
	NNS_GFD_DST_2D_background3_SCR_MAIN	2D main BG3 screen rewrite
	NNS_GFD_DST_2D_background2_BMP_MAIN	2D main BG2 bitmap rewrite
	NNS_GFD_DST_2D_background3_BMP_MAIN	2D main BG3 bitmap rewrite
	NNS_GFD_DST_2D_OBJECT_PLTT_MAIN	2D main OBJ palette rewrite
	NNS_GFD_DST_2D_background_PLTT_MAIN	2D main BG palette rewrite
	NNS_GFD_DST_2D_OBJECT_EXTPLTT_MAIN	2D main OBJ extended palette rewrite
	NNS_GFD_DST_2D_background_EXTPLTT_MAIN	2D main BG extended palette rewrite
	NNS_GFD_DST_2D_OBJECT_OAM_MAIN	2D main OAM rewrite
	NNS_GFD_DST_2D_OBJECT_CHAR_MAIN	2D main OBJ character rewrite

Category	Constants Indicating the Data Transfer Destination Type	Meaning
2D sub	`NNS_GFD_DST_2D_background0_CHAR_SUB`	2d sub BG0 character rewrite
	`NNS_GFD_DST_2D_background1_CHAR_SUB`	2d sub BG1 character rewrite
	`NNS_GFD_DST_2D_background2_CHAR_SUB`	2d sub BG2 character rewrite
	`NNS_GFD_DST_2D_background3_CHAR_SUB`	2d sub BG3 character rewrite
	`NNS_GFD_DST_2D_background0_SCR_SUB`	2d sub BG0 screen rewrite
	`NNS_GFD_DST_2D_background1_SCR_SUB`	2d sub BG1 screen rewrite
	`NNS_GFD_DST_2D_background2_SCR_SUB`	2d sub BG2 screen rewrite
	`NNS_GFD_DST_2D_background3_SCR_SUB`	2d sub BG3 screen rewrite
	`NNS_GFD_DST_2D_background2_BMP_SUB`	2d sub BG2 bitmap rewrite
	`NNS_GFD_DST_2D_background3_BMP_SUB`	2d sub BG3 bitmap rewrite
	`NNS_GFD_DST_2D_OBJECT_PLTT_SUB`	2d sub OBJ palette rewrite
	`NNS_GFD_DST_2D_background_PLTT_SUB`	2d sub BG palette rewrite
	`NNS_GFD_DST_2D_OBJECT_EXTPLTT_SUB`	2d sub OBJ extended palette rewrite
	`NNS_GFD_DST_2D_background_EXTPLTT_SUB`	2d sub BG extended palette rewrite
	`NNS_GFD_DST_2D_OBJECT_OAM_SUB`	2d sub OAM rewrite
	`NNS_GFD_DST_2D_OBJECT_CHAR_SUB`	2d sub OBJ character rewrite

In a VRAM transfer task, the data transfer destination type shown above, the transfer destination address, the data address, and the transfer data size are stored. The actual data to be transferred is not copied, so the data should be kept in memory until the VRAM transfer manager writes it to the VRAM. The data is written to the VRAM when the user calls the `NNS_GfdDoVramTransfer()` function.

2.4 Executing a VRAM Transfer Task

Call the following function to write data registered to a VRAM transfer task to the VRAM.

```
void NNS_GfdDoVramTransfer(void);
```

Ordinarily, the `NNS_GfdDoVramTransfer()` function needs to be called during a V-Blank period, as that is when data can be written to VRAM. The `NNS_GfdDoVramTransfer()` function executes all VRAM transfer tasks in the order registered to the VRAM transfer manager queue. The VRAM transfer manager does not control whether all of the registered VRAM transfer tasks can be executed during a V-Blank period. The user should manage the total amount of VRAM rewrite data and register to the VRAM transfer manager only the amount that can be rewritten during a V-Blank period.

2.5 Obtaining the Size of the VRAM Transfer Tasks

With the VRAM transfer manager, the total size of the VRAM transfer tasks registered to the VRAM transfer manager can be obtained. Call the following function to obtain the total amount of VRAM transfer tasks.

```
u32 NNS_GfdGetVramTransferTaskTotalSize(void);
```

By using this function to obtain the total transfer amount, a determination can be made of whether a data transfer can be completed during a V-Blank.