

Simple ESRAM for DirectX 12 Sample

*\* This sample is compatible with the August 2016 Xbox One XDK.*

# Description

This sample demonstrates the basics of utilizing ESRAM with DirectX 12. It leverages the XG & XGMemory APIs to reserve virtual resource memory and subsequently map it to DRAM & ESRAM. A few different page mapping schemes are showcased to provide examples of how the XGMemory library can be used to customize resource layout between DRAM & ESRAM.

The XG library provides a simple interface for calculating the number of pages for the different resource types available in DirectX. Since the actual size, layout, alignment, and metadata structures of DirectX resources are calculated internally this is a required step to accurately determine the resource size, which can then be used to calculate the page count. This page count can then be used to perform ESRAM/DRAM mapping at the per-page granularity.

The XGMemory library is the recommended method for mapping resource pages to ESRAM memory. This API hides the lower level details of allocating a virtual address space and committing its pages to physical memory while still giving the developer full control over whether each page of the address space is mapped to DRAM or ESRAM. It also internally manages mapped page coherence by utilizing a safe, free-list page allocation scheme and forcing potentially hazardous resource aliasing to be done explicitly.

A quick outline of the sample’s memory mapping schemes:

*Simple Mapping –* Maps a specified number of resource pages to ESRAM and the remaining pages to DRAM. The benefit of this method is it requires minimal work by the developer, a single function call, but provides no custom behavior.

*Split Mapping –* Splits the resource into a beginning DRAM section, a middle ESRAM section, and an ending DRAM section. Split mapping can be beneficial for render targets that sport both low- and high-utilized regions. For instance, during outdoor scene rendering the skybox commonly populates the top portion of the frame. This results in only one draw call for those pixels, which benefits less from ESRAM residency than the rest of the image which can incur significantly more overdraw.

*Metadata Mapping –* Maps only the color and depth metadata textures to ESRAM. This is beneficial as it uses minimal ESRAM while optimizing metadata accelerated operations such as clears and resolves.

*Random Mapping –* Performs a randomized per-page choice of DRAM or ESRAM according to a specified probability. There’s not a specific benefit to this mapping scheme – it simply shows off how to utilize the provided API in a unique fashion.

Further information and full API specifications of the XG & XGMemory libraries are available in the XDK Documentation.

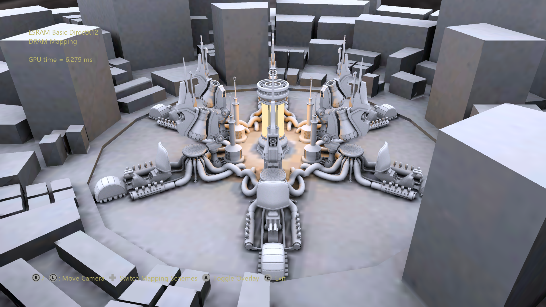
Note: The Xbox One X doesn’t have ESRAM, instead being built with faster DRAM. On this console the sample will simply render the scene with all ESRAM options and visualizations disabled.

# Using the sample

The sample has controls which are common to all states of the program which includes the ability to dynamically switch between mapping schemes. Each mapping scheme also has its own mapping-specific parameters that are modifiable through the gamepad bumpers and triggers. These controls are also visible on-screen when applicable.

## Common Controls

|  |  |
| --- | --- |
| Action | Gamepad |
| Orbit Camera | Left- & Right-Stick |
| Switch Mapping Scheme | D-Pad Left/Right |
| Toggle Overlay | A Button |
| Exit | View Button |



## Simple Mapping Controls

|  |  |
| --- | --- |
| Action | Gamepad |
| Increase color ESRAM page count | LB Button |
| Decrease color ESRAM page count | LT Button |
| Increase depth ESRAM page count | RB Button |
| Decrease depth ESRAM page count | RT Button |



## Split Mapping Controls

|  |  |
| --- | --- |
| Action | Gamepad |
| Increase ESRAM begin address | LB Button |
| Decrease ESRAM begin address | LT Button |
| Increase ESRAM end address | RB Button |
| Decrease ESRAM end address | RT Button |

## 

## Metadata Mapping Controls

|  |  |
| --- | --- |
| Action | Gamepad |
| Toggle On/Off | B Button |

## 

## Random Mapping Controls

|  |  |
| --- | --- |
| Action | Gamepad |
| Decrease ESRAM page probability | LB Button |
| Increase ESRAM page probability | RB Button |

## 

# Update history

3/12/2018 - The original version of the sample was written using the XSF-based framework. It was rewritten to use the ATG sample templates in March 2018 with support for Xbox One X.

# Privacy Statement

When compiling and running a sample, the file name of the sample executable will be sent to Microsoft to help track sample usage. To opt-out of this data collection, you can remove the block of code in Main.cpp labeled “Sample Usage Telemetry”.

For more information about Microsoft’s privacy policies in general, see the [Microsoft Privacy Statement](https://privacy.microsoft.com/en-us/privacystatement/).