



Multi-Head Support In DirectX® 9

Guennadi Riguer

Developer Relations

`griguer@ati.com`

Outline

- Multi-Head Introduction
- Multi-Head Problems and Solutions
- Multi-Head CAPS
- Identifying Multi-Head Devices
- Creating Multi-Head Devices
- Presentation Parameters
- Resetting Multi-Head Devices
- Multi-Head Swap Chains
- Swap Chain Shortcuts
- Rendering and Presentation



DirectX® 9 Multi-Head Intro

- A Multi-Head card
 - Common frame buffer
 - Common graphics accelerator
 - Multiple DAC's and monitor outputs
- All heads form a “Group”
 - One head is “Master”
 - Other are “Slaves”
- Adapters referenced by ordinals



Former Multi-Head Problems

- No hardware available
- Poor API support
 - Multiple devices in DirectX® 8
 - Duplication of resources



New Multi-Head Picture

- All new ATI hardware supports Dual-Head configurations
 - 20M+ ATI cards on the market
- Other IHV also support Multi-Head
 - Matrox even has Triple-Head
- DX 9 API is much better
 - Single device with multiple swap chains



Multi-Head CAPS

- New CAPS in DX 9
 - NumberOfAdaptersInGroup
 - MasterAdapterOrdinal
 - AdapterOrdinalInGroup



Multi-Head CAPS Examples

- One single-head card scenario:

	Adapter ordinal	Adapters in group	Master ordinal	Ordinal in group
Card 1	0	1	0	0

- One dual-head card scenario:

	Adapter ordinal	Adapters in group	Master ordinal	Ordinal in group
Card 1	0	2	0	0
Card 1	1	0	0	1



Multi-Head CAPS Examples

- Two dual-head and one single-head card scenario:

	Adapter ordinal	Adapters in group	Master ordinal	Ordinal in group
Card 1	0	2	0	0
Card 1	1	0	0	1
Card 2	2	2	2	0
Card 2	3	0	2	1
Card 3	4	1	4	0

Identifying Multi-Head Devices

- Use caps to identify multi-head devices

```
// Get number of adapters
UINT cAdapters = m_pD3D->GetAdapterCount();
for ( UINT i = 0; i < cAdapters; i++ ) {
    // Get CAPS
    m_pD3D->GetDeviceCaps( i, D3DDEVTYPE_HAL, &caps );
    // Multi-head if more than 1 adapter in group
    if ( caps.NumberOfAdaptersInGroup > 1 ) {
        // Found multi-head adapter
        // . . .
    }
}
```



Creating Multi-Head Devices 1/2

- Create windows for each adapter
 - Create top-most windows with `WS_POPUP` and `WS_VISIBLE` style flags
 - Designate first, master adapter window as focus window
 - Create slave adapter windows using master adapter window as a parent



Creating Multi-Head Devices 1/2

- Create a single D3D device
 - Create with behaviour flag `D3DCREATE_ADAPTERGROUP_DEVICE`
 - Use multiple presentation parameters, one per adapter in the group
 - Use focus (master adapter) window handle with `CreateDevice`
- Don't create swap chains with `CreateAdditionalSwapChain` for multi-head device



Presentation Parameters

- Supply unique device window handles to each set of presentation parameters
- Set Windowed to FALSE
- All EnableAutoDepthStencil have to be the same
- If EnableAutoDepthStencil is TRUE, AutoDepthStencilFormat, BackBufferWidth, BackBufferHeight and BackBufferFormat should be the same



Resetting Multi-Head Device

- Reset also takes array of presentation parameters
- All adapters have to be full screen on reset
- Destroy and recreate non multi-head device to switch to windowed mode



Multi-Head Swap Chains

- Use swap chains to get access to multiple adapters
- Use `GetNumberOfSwapChains` to get number of swap chains
- Use `GetSwapChain` to get swap chain
- Swap chains updated with extra functions
 - `GetDisplayMode`
 - `GetFrontBuffer`
 - `GetRasterStatus`



Swap Chain Shortcuts

- Some device function have swap chain reference
 - SetGammaRamp(UINT iSwapChain, ...)
 - GetGammaRamp(UINT iSwapChain, ...)
 - GetBackBuffer(UINT iSwapChain, ...)
 - GetFrontBuffer(UINT iSwapChain, ...)
- Use adapter ordinal in group to identify swap chain



Presents on Multi-Head Device

- Supports per-head or atomic present
- Per-head swap using swap chain
 - `m_pSwapChain->Present(...);`
- Atomic present happens at the device level
 - `m_pd3dDevice->Present(...);`



Rendering Example

```
// Get number of swap chains
UINT cSwapChains = m_pD3D->GetAdapterCount();
// Get current render target
IDirect3DSurface9 *pOldRenderTarget = NULL;
m_pDevice->GetRenderTarget( 0, &pOldRenderTarget );
// Get current depth/stencil
IDirect3DSurface9 *pDepthStencil = NULL;
m_pDevice->GetDepthStencilSurface( &pDepthStencil );
// Render for each swap chain
for ( i = 0; i < cSwapChains; i++ ) {
    // Get swap chain
    IDirect3DSwapChain9 *pSwapChain = NULL;
    m_pDevice->GetSwapChain( i, &pSwapChain );
    // Get swap chain back buffer
    IDirect3DSurface9 *pBackBuffer = NULL;
    pSwapChain->GetBackBuffer( 0, D3DBACKBUFFER_TYPE_MONO, &pBackBuffer );
    // Set swap chain buffer as render target and attach back buffer
    m_pDevice->SetRenderTarget( 0, pBackBuffer );
    m_pDevice->SetDepthStencilSurface( m_pDepthStencil[ i ] );
    // Clear and render
    m_pDevice->Clear( 0, NULL, D3DCLEAR_TARGET|D3DCLEAR_ZBUFFER, 0x00000000, 1.0f, 0 );
    if ( SUCCEEDED( m_pDevice->BeginScene() ) ) {
        // Render scene here...
        m_pDevice->EndScene();
    }
}
```



Conclusion

- Current Multi-Head Status
- Multi-Head CAPS
- Identifying Multi-Head Devices
- Creating Multi-Head Devices
- Presentation Parameters
- Resetting Multi-Head Devices
- Multi-Head Swap Chains
- Rendering and Presentation

