Performance Lessons from Porting Source 2 to Vulkan

Dan Ginsburg



Overview

- Dota 2 Vulkan Performance Results
- Performance Lessons Learned

Overview

- Dota 2 Vulkan Performance Results
- Performance Lessons Learned

Source 2 Overview

- OpenGL, Direct3D 9, Direct3D 11, Vulkan
- Windows, Linux, Mac
- Dota 2 Reborn



Dota 2 Performance Results - Disclaimer

- Not an ideal showcase for Vulkan
- Source 2 renderer is multithreaded, but...
 - Dota 2 is only ~1500 draw calls per frame
 - Allows DX/GL a frame of latency to avoid being renderthread bound
 - Does not (yet!) take advantage of:
 - Baking descriptors
 - Command buffer resubmission

Dota 2 Performance Results - Disclaimer

- Not an ideal showcase for Vulkan
- Source 2 renderer is multithreaded, but...
 - Dota 2 is only ~1500 draw calls per frame
 - Allows DX/GL a frame of latency to avoid being renderthread bound
 - Does not (yet!) take advantage of:
 - Baking descriptors
 - Command buffer resubmission
- Still very pleased with results!

Dota 2 Vulkan Performance – DX9 Latency

— MainThrd	(Unaccounted)MainLoop {6.96ms}	(Unaccounted)MainLoop {4.35ms}	(Unaccounted)MainLoc
	CLoopTypeClientServer::Update {6.96ms}	CLoopTypeClientServer::Update {4.34ms}	CLoopTypeClientServe
— GlobPool0	ClobThread::CheckSignals {2.58ms} ClobThread::CheckS ClobThread::CheckS ClobThread::CheckSignals {2.58ms} Cl	CheckSignals (1.83ms) Job ClobThread CheckS Job: CSc Job: Threac Job: Re C	obThread::CheckSignals
	Update Three Three	Upr Thread Procedural CSc	
- GlobPool1	CjobThread CheckSignals (2.51ms) CjobThread Checks CjubThread Checks Job: Upc CjubThread Checks Job: CScene Job: CjobThread	CheckSignals (1.88ms): Job: L ClooThread-CheckS Job: CS Job: C ClobThr	ead::CheckSignals
	Updatel Thread_Rer Thr	Upda Thread Threa	
— GlobPool2	Job: Up Job: CSC Job: F Job: R	Job: CScene Job: Job: Frances	nad:Cheddignais
•	Update Thread_ Drav CSc	Upd Thread_Rer Draw	
- D3D9RenderThread	CRenderThreadBase::Dispatch() #Player 0/FXAA/BatchR CRender1	CRenderThreadBase::Dispatch() #Player 0/FXAA/BatchRer CRend	CRender
	CRenderThreadDx9::Execute {2.37ms} CRender1	CRenderThreadDx9::Execute {2.46ms} CRend	CRender

Frame Start

Frame End

Dota 2 Vulkan Performance – DX9 Latency

Jnaccounted)MainLoc LoopTypeClientServe
LoopTypeClientServe
iread::CheckSignals
I::CheckSignals
d::Chatidignais
CRender
CRender

Frame Start

Frame End

Present Issued

Dota 2 Vulkan Performance – DX9 Latency

— MainThrd	(Unaccounted)MainLoop {	6.96ms}				(Una	(Unaccounted)MainLoc									
	CLoopTypeClientServer::Upda	te {6.96ms}				CLoopTy	CLoopTypeClientServe									
— GlobPool0	CJobThread::CheckSignals {2.58ms} CJobThread::CheckS CJobThread::CheckS Job: 1	Up CjobThread::CheckSig	g job: C		CJobThrea	:CheckSignals {1.83ms}	CJobThread::CheckS	Job: CSc	Job: Threac	Job: Re	JobThread::CheckSignals					
	Upda	ste	Threa			υρι		Thread_	Procedural	CSc						
- GlobPool1	CJobThread::CheckSignals (2.51ms) CJobThread::CheckSi CJobThread::CheckS Job:	Upc CjobThread::CheckSi	job: CScene	Job:	CJobThread	CheckSignals {1.88ms} Job: (CJobThread::CheckS	Job: CS	Job: C	CJobTh	read::CheckSignals					
	Upda	itel	Thread_Rer	Thr		Upda		Thread	Threa							
- GlobPool2	and the second	Jp	Job: CSc	Job: F	Job: Rc	inner Cherriques <mark>Job</mark>	Goolfreedochards	Job: CSco	ne Jo	b:F Cjott	anad Checklopals					
	upda	ite	Thread_	Draw	CSc	Upd		Thread_P	er Di	raw						
 D3D9RenderThread 	CRenderThreadBase::Dispatch() #Player 0/FXAA/BatchR CRend	erl Chicado - Chicado		ead5ync0b)		CRenderThr			BatchRer CRe	nd	CRender					
	CRenderThreadDx9::Execute {2.37ms} CRend	er1				CRend	lerThreadDx9::Exec	ute {2.46:	ms} CRe	nd	CRender					
Frame	Start		I	⁻ ra	me	End		Pr	eser	nt Is	sued					

DX9 Latency: 3.8ms

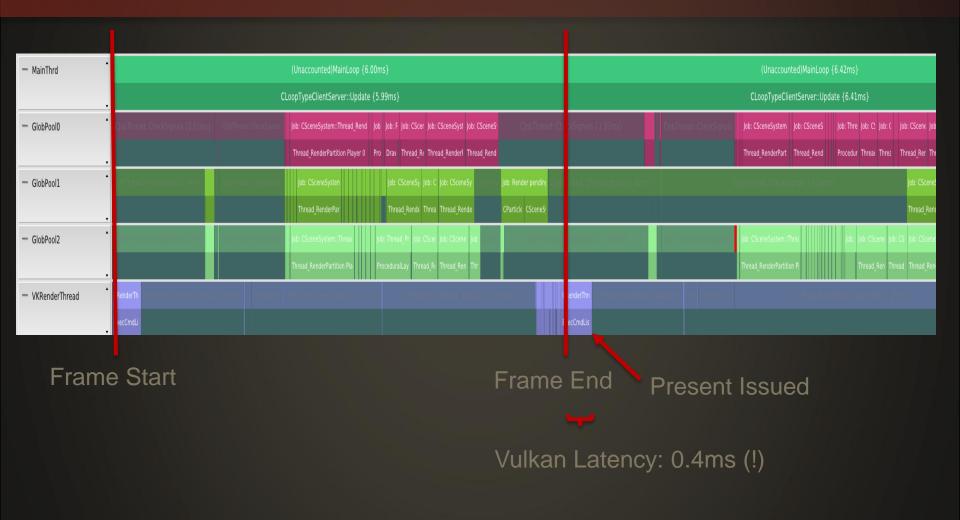
Dota 2 Vulkan Performance – Vulkan Latency

- MainThrd		(Unaccounted)MainLoop {6.00ms	s}		(Unaccounted)MainLoop {6.42ms}									
		CLoopTypeClientServer::Update {5.9	99ms}	CLoopTypeClientServer::Update {6.41ms}										
- GlobPool0	CJobThread::CheckSignals {2:62ms} CJobThread::C	CheckSignals Job: CSceneSystem::Thread_Rend Job	Job: F Job: CScer Job: CSceneSyst Job: CSceneS	:kSignals {1.93ms}	CJobThread::CheckSignals	Job: CSceneSystem Job: CSceneS	Job: Thre Job: C! Job: C	Job: CScene Job						
		Thread_RenderPartition Player 0 Pro	Drav Thread_Rt Thread_Renderf Thread_Rend				Thread_RenderPart Thread_Rend	Procedur Threat Threa	Thread_Rer Thr					
- GlobPool1	JobThread::CheckSignals {2.54ms}	CheckSignals Job: CSceneSysten	Job: CSceneSy Job: C Job: CSceneSy ClobThrea	Job: Render pending CJob	read::CheckSignals {1.42ms}	Ċ	obThread::CheckSignals {3.24ms	}	Job: CScenes					
		Thread_RenderPar	Thread_Rende Threa Thread_Rende	CParticle CSceneS					Thread_Renc					
- GlobPool2	and the second second	Job: CSceneSystem::Threac Job): Thread_Pr Job: CScer Job: CScene Job	Qui finant a	000000000	QddffrendsChed Signals	Job: CSceneSystem::Threa	Job: Job: CScene Job	: CS Job: CScene:					
		Thread_RenderPartition Pla	oceduralLay Thread_Rr Thread_Ren Thr				Thread_RenderPartition Pl	Thread_Ren Thr	read Thread_Ren					
- VKRenderThread	RenderTh (ThreadSyncObect Wall10) (135ns)	Chreadlyn Chreadlyn Obert War 10 (1 50m)	CThreadSyncObject: Wait(10) (2:03	ims)	enderThn CThreadSyncObject. V	Vait(10) CTre CThreadSync0	CThreadSynd	Object: Wait(10) (3.15	ims)					
	(ecCmdLi				ecCmdLis									
Frame	e Start		F	End										

Dota 2 Vulkan Performance – Vulkan Latency

— MainThrd			(Unaccounted)MainLoop {6.				(Unaccounted)MainLoop {6.42ms}									
		CI	LoopTypeClientServer::Update	{5.99ms}			CLoopTypeClientServer::Update {6.41ms}									
- GlobPool0	ClobThread::CheckSignals {2.62ms}	CJobThread::CheckSignals	Job: CSceneSystem::Thread_Rend	Job Job: F Job: CSce	:kSignals {1.93ms}	CJobThread::CheckSignals	Job: CSceneSystem	Job: CSceneS	Job: Thre Job: C! Job: C	Job: CScene Job						
			Thread_RenderPartition Player 0	Pro Drav Thread_R	(Thread_Renderf Thread_Ren	d			Thread_RenderPart	Thread_Rend	Procedur Threat Threa	Thread_Rer Thr				
- GlobPool1	JobThread::CheckSignals {2:54ms}	CJobThread::CheckSignals	Job: CSceneSysten	Job: CSceneSy	Job: C Job: CSceneSy Clooth	ea Job: Render pending CJob	read::CheckSignals {1.42ms}	(CJobThread::CheckS	ignals {3.24m	s}	Job: CScene!				
			Thread_RenderPar	Thread_Rende	Thread_Rende	CParticle CSceneS						Thread_Rend				
- GlobPool2		Geologica and geol	Job: CSceneSystem::Threac	Job: Thread_Pr Job:	CScer Job: CScene Job	a Qabihood 3	oʻsinik (1966)	GathranisChedSignalis	Job: CSceneSystem::	Threa	Job: Job: CScene Jo	b: CS- Job: CScene				
			Thread_RenderPartition Pla	ProceduralLay Thre	ead_R(Thread_Ren Thr				Thread_RenderPartiti	ion Pl	Thread_Ren T	hread Thread Ren				
- VKRenderThread	RenderTh ChreadSyncObject (ValG10)	(1.36ms) CThreadSync	CThreadSyncObject: War(10) (1.30m	s) CThrea	dSyncObject::Wait(10) {3	1.03ms)	enderThn CThreadSyncObject	Wait(10) CTre CTrreadSyrc0		CThreadSy	ncObject::Wait(10) (3.1	15ms)				
	(ecCmdLi						ecCmdLis									
Fram	e Start				e End Present Issued											

Dota 2 Vulkan Performance – Vulkan Latency



Dota 2 Vulkan – Latency Reduction

- Renderthread no longer a bottleneck
- Reduces "wallclock" time of frame
 - Time from end of frame to present reduced by 3.4ms
- Really important for:
 - Latency sensitive games (eSports)
 - VR

Dota 2 Vulkan - Framerate

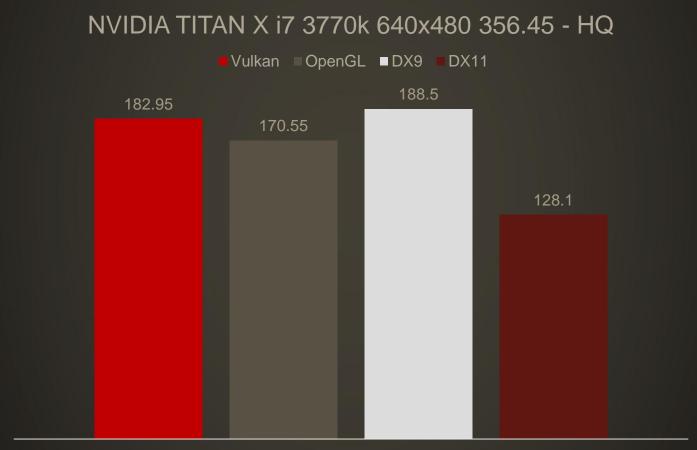
Two timedemos:

- Typical Dota 2 Match
- High Drawcall Battle Scene
- Test system:
 - NVIDIA TITAN X 356.45
 - i7-3770k @ 3.50GHz
- Test settings:
 - Resolution: 640x480 (CPU Perf)
 - Highest Rendering Quality
 - Vulkan/GL/DX9/DX11

Dota 2 Timedemo – Typical Dota 2 Match



Dota 2 Timedemo – Typical Dota 2 Match

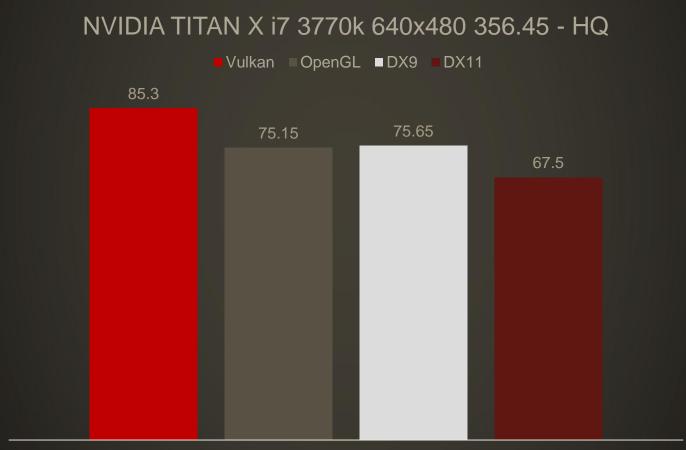


FPS

Dota 2 Timedemo – Battle Scene



Dota 2 – High Drawcall Timedemo



FPS

Dota 2 Vulkan Performance - Overall

- Significant latency reduction
- Improved framerate in heavy scenes
- Only going to get better...

Overview

- Dota 2 Vulkan Performance Results
- Performance Lessons Learned

Overview

Dota 2 Vulkan Performance Results

Performance Lessons Learned

- Command Buffer Recycling
- Command Buffer Batching
- Redundant Call Filtering
- Updating Descriptors
- Pipeline Cache Usage

Command Buffer Recycling Overview

- At least one VkCommandPool per thread
- Recycling options:
 - vkResetCommandPool resets all command buffers in pool
 - vkResetCommandBuffer reset single command buffer
- Reset can either recycle or release resources

Command Buffer Recycling

- Souce 2 recycles individual command buffers after completion
- vkBeginCommandBuffer costly
 - Using VK_COMMAND_BUFFER_RESET_RELEASE_RESOURCES_BIT
 - Driver reallocates resources
 - Done to reduce memory footprint, but came at perf cost

Fast Command Buffer Recycling

vkCreateCommandPool

- Use vk_command_pool_create_reset_command_buffer_bit
- vkResetCommandBuffer(pCmdBuffer, 0)
 - flags == 0, keeps resources for reuse
 - Downside: memory growth
- Source 2 strategy for handling memory growth:
 - Destroy command buffers no longer needed
 - Heuristic to destroy command buffers

- vkQueueSubmit implies a flush
 - Also has CPU costs memory residency
- Important to batch submits

— MainThrd	CLoopTypeClientServer::			eClientServe	er::Update {	5.65ms}		CLoopTypeClientServer::Update {6.33ms}					CLoopTypeClientServer::Update {6.37ms}								CLoopTypeClientServer::Update {6.27ms}							
•	CSceneSystem::WaitFr	CEventDispato	ł CEventDisp	CSceneSy	rstem::WaitFo	rRendering	'oComplete	CEventDispatche	r::Dis CEventDis	pa CScene	System::W	VaitForRenderin	gToComplete	CEventDispatcher::Dis CEventDispa CSceneSystem::WaltForRenderingToComplete CRend							CEventDispat	(EventDispatcher::: (EventDispat: CSceneSystem::WailForRenderingToComplete (Rende						ete CRende
- GlobPool0	Job. CjobThread C	CheckSignals	CjobThreac j o	ob: CSce	Job: Job: (. Job: Render	p GobThread:Cl	eckSignals (2.32ms)	CJobThread::	Job: CScei		Job: C Job: CSc	CjobThread::Che	(Signals (2.59ms)	CjobThread:	Job: CSr		Job: Render	r pei Job	CJobThread	::CheckSignals	CjobThread::C	C job: C	CSc	Job:	Render p	CjobT	hread::CheckSigna
•	Thr		Т	hread_R	Thre Three	2 DrawDrawL	1			Thread_R/		Threa Thread				Thread _.		DrawDrawL	.ist Thr				Threa	adji	Drai	vDrawLi:		
- GlobPool1	Job: Rende	bThread::Che	CjobThreac	Job: CS	Job: C Jot).)	Job: Render 🖓	Thread::CheckSignals	CjobThread::	Job: CScene	System::T	Job Job: Rende	rper CjobThre	d::CheckSignals	CjobThread	Job: CSce		Job: CS	Job: CSc	e CjobThread	::CheckSignals	QobThread: 0	. Job: C	SceneSyste	5W	Job:	Job: I	Ren ClobThread :
•	CP: CSce			Thread	Threa Th	re	CPart CSce			Thread_Ren	derPartitic	Thr DrawDraw	List'			Thread_R		Thread	Thread	R			Threa	d_RenderPi	art	Thre	CPa	
- GlobPool2	Job: C				stem::" Job					Job: C	job:	C Job: Job	Job: Render pen			Job: CScenes		Job: Job: J	ob: CScei	Job: Rer			Job: C	Scer	Job:	CS Job: C	IScen	
Ţ	Threa				rPartiti Thr					Threa	Thre	a Thri Thr	CParticle CSce			Thread_Rend		The The 1	Thread_R(CPa			Thread	i_R¢	Thr	ead Three	d_Re	
- VKRenderThread	Chrosopolij 🤇 🕅	e (Trodspo	lóje Chi	CThreadSyr	Chreat	SyncObject: \	(an(10)	Rer (Threadsyntoty	d:Nat Chr	Chea	۵y (T	hreadSyncObjec	t: ((a)(10)		CR	enderThreadBas			CRe	ThreadSj				CRe	CThrea	dSyncObjec	:Wait1	
	Lx	e(E	Kec	Т						Ex				Exer					Exe				

— MainThrd	CLoopTypeClientServer::		/peClientServer::L	Update {5.65ms}				ClientServer::Upda	te {6.33ms}				eClientServer:	:Update (6			CLoopTypeClientServer::Update {6.27ms}					
	CSceneSystem::WaitFr	CEventDispatch CEventDis	p CSceneSysten	n::WaitForRendering	ToComplete	CEventDispatcher::	Dis CEventDisp	8 CSceneSystem	::WaitForRenderin	gToComplete	CEventDispatcher.:Dis CEventDispa CSceneSystem::WaitForRenderingToComplete CRend							CEvenDispatcher:: CEvenDispat: CSceneSystem::WaitForRenderingToComplete C				
- GlobPool0	Job CjobThread	CheckSignals CjobThreac	Job: CSce	Job: Job: C Job: Render	p CjobThread::Cl	eckSignals (2.32ms)	CjobThread::	Job: CScei	Job: C Job: CSc	CjobThread::Cheo	KSignals (2.59ms)	CjobThread::	Job: CSi	Job: Rer	der pei Job	CJobThread:	:liheckSignals	CjobThread::C	Job: CSc	Job:	Render p	CjobThread::CheckSignal
	Thr		Thread_R	Thre Three DrawDrawl	je			Thread_R/	Threa Thread				Thread _.	DrawDr	wList Thr				Thread_I	Drai	wDrawLis	
- GlobPool1	Job: Rende	bThread::Che CjobThreac	Job: CS	Job: C. Job: (Job: Render 🖓	Thread::CheckSignals	CjobThread::-	Job: CSceneSystem::1	l job job: Rende	rper CjobThrea	d::CheckSignals	CjobThread::	Job: CSce	Job: (lS Job: CSce	CjobThread:	CheckSignals	CJobThread::C	Job: CSceneS	lystem	Job:	Job: Ren ClobThread. C
	CPi CSce		Thread	Threa Thre	CPart CSce			Thread_RenderPartiti	c Thr DrawDrawl	List'			Thread_R	Thre	ad Thread_R				Thread_Reno	lerPart	Thre	CPa
- GlobPool2	Job: C			::" Job				Job: C. Jo	b: C job: job	Job: Render pen			Job: CSceneSys	tem Job: Jol	: Job: CScei	Job: Rer			Job: CScer	Job:	: CS Job: CS	cen et el
Ţ	Threa		Thread_RenderParti	iti Thr				Threa: Ti	hrea Thri Thr	CParticle CSce			Thread_Renderl	Part Thri Th	r Thread_Ri	CPi			Thread_Rc	Thr	ead Thread	Re
- VKRenderThread	Drodycta 0	Re Chrouspicop Chi	CThreadSyr	(ThreadSyncObject:	Ka(10) C	Rer Throdynthyd	the Che	Chrassy	CThreadSync0bject	::Wat(10)		CRe	nderThreadBas		CRe CT	nadáj			CRe	CThrea	dSyncObject	Wat(10)
	b	(e(8	Kec						Exe	:CmdList(Frm f		Exer				Exe			

Batched submit: ~0.7ms / frame

— MainThrd	CLoopTypeClientServer::		.oopTypeClientServe	er::Update {5.65ms}			CLoopTypeC	lientServer::Update	e {6.33ms}				eClientServer::	Update {6.37ms}			CLoopTypeClientServer::Update {6.27ms}					
,	CSceneSystem::WaitFr	CEventDispatcł CEv	wentDisp CSceneSys	stem::WaitForRenderi	ngToComplete	CEventDispatcher::	Dis CEventDispa	CSceneSystem::	WaitForRenderingT	oComplete	CEventDispatcher	r::Dis CEventDisp	va CSceneSys	tem::WaitForRenderin	CEventDispato	CEventDispatcher:: (EventDispat: CSceneSystem::WaltForRenderingToComplete CRend						
- GlobPool0	Job CjobThread C	CheckSignals CjobTl	Threac Job: CSce	Job: Job: C Job: Ren	der p CjobThread::Ci	eckSignals (2.32ms)	CJobThread::	Job: CScei	Job: C Job: CSc C	JobThread::Chec	Signals (2:59ms)	ClobThread::	Job: CSi	Job: Render pei Jo	CJobThread	: CheckSignals	CjobThread::0	Job: CSc	Job: Render p	Cjob	Thread::CheckSigna	
	Thr		Thread_R	Thre Three DrawDra	wLis			Thread_R/	Threa Thread				Thread	DrawDrawList Th				Thread	DrawDrawLis			
- GlobPool1	Job: Rende	bThread::Che CjobTi	Threac Job: CS	Job: C. Job: (Job: Render ())	Thread::CheckSignals	CjobThread::	Job: CSceneSystem::T	Job Job: Render p	er CjobThrea	l::CheckSignals	ClobThread.:	Job: CSce	Job: CS Job:	CSce ClobThread:	CheckSignals	QobThreads	Job: CSceneSyste	ım Job:	jot	: Ren ClobThread.	
	CP: CSce		Thread	Threa Thre	CPart CSce			Thread_RenderPartitic	Thr DrawDrawList	8			Thread_R	Thread Thr	ad_R			Thread_RenderPa	art Thre	e CP	a	
- GlobPool2	job: C		Job: CSceneSys	tem::" Job				job: C. job	:C Job: Job Job): Render pen			Job: CSceneSyste	em Job: Job: Job: CSC	ı job: Rer			Job: CScer	Job: CS	ob: CScen	ilina Galag	
	Threa		Thread_Render	Partiti Thr				Threa: Thr	ea Thri Thr CP	article CSce			Thread_RenderP	art Thrı Thr Thread	(CP:			Thread_Re	Thread	Thread_Re		
 VKRenderThread 	Chrostyrchy (R e	er CThroadSyncobje	Chi Chiudiy	CThreadSyncObjec	t. Watt 10)	Rer (Threadsyncobject	ilil îli	CThreadSy C	ThreadSyncObject: W	Vait(10)		CRer	iderThreadBas	CRe	Chinads			CRe	CThreadSyncC	bject::Wait(1	0	
,	Exe	e			E	Kec						Exec	CmdList(Frm f	Exe				Exe				
	_ ⊾									Ļ												

Batched submit: ~0.7ms / frame

Unbatched submits: ~4.5ms / frame

Source 2 Command Buffer Batching

- Gather command buffers on renderthread
 - Up to a threshold, needed during load time
- Wait for present request
- Issue single submit with all batched command buffers

Redundant Call Filtering

Your job now!

- Vulkan drivers may not (should not!) filter calls
- If we don't do it, we will force IHVs to
- Hurts the good apps at the expense of the bad
- Examples from Source 2:
 - vkCmdBindIndexBuffer
 - vkCmdBindVertexBuffers
 - vkCmdBindPipeline
 - Dynamic render state
 - vkCmdSet*

Updating Descriptors

- vkUpdateDescriptorSets #1 hotspot
- vkCmdBindDescriptorSets #2 hotspot
- Source 2 approach:
 - Single pipeline layout shared across all pipelines
 - Descriptor sets will have unused entries
 - Update/bind descriptor set per draw
 - Not efficient!

Updating Descriptors – The Right Way

- In shaders, organize descriptor sets by update frequency
- Bake descriptor sets up front
- Use compatible pipeline layouts to simplify descriptor allocation

Updating Descriptors – The Right Way

- In shaders, organize descriptor sets by update frequency
- Bake descriptor sets up front
- Use compatible pipeline layouts to simplify descriptor allocation
- ...we plan to do this in the future. Will help perf a lot.

Pipeline Creation

- vkCreateShaderModule is relatively fast
 - Loads in the SPIR-V, no heavy compilation
 - ~0.01ms in Dota 2
- vkCreateGraphicsPipelines is expensive
 - Driver performs shader compile here
 - 0.2 152ms in Dota 2 before cache is warmed

Vulkan Pipeline Cache

Serialize compiled pipelines to disk

- Preload to remove first-time stutters
- Header contains VendorID/DeviceID/UUID
 - Otherwise opaque format
- Avoid unnecessary shader compiles
 - Driver de-duplicates
 - Only driver knows when recompile is needed based on state
 - Pipeline cache should contain only unique pipelines
- Allows compilation on multiple threads
 - Merge later using vkMergePipelineCaches

Summary

Dota 2 Vulkan Performance Results

- Reduced latency
- Improved framerate in expensive scenes
- Performance Lessons Learned
 - Command Buffer Recycling
 - Command Buffer Batching
 - Redundant Call Filtering
 - Updating Descriptors
 - Pipeline Cache Usage

Questions?

