Watch Dogs 2 - PC Adaptation Success Story with NVIDIA

Jaakko Haapasalo, Producer, GeForce Titles (NVIDIA)
Marius Tudorache, Eastern European PC Director (Ubisoft)
Farid Rzaev, Developer Technology Engineer (NVIDIA)
Oleh Kuznetsov, Senior Programmer (Ubisoft)
November 2016...

The Watch Dogs 2 PC port is great

[...] it runs well, has a ton of graphical options, and comes with a complete set of quality-of-life adjustments for mouse and keyboard players.

There is a wide range of options that will allow you to cater the experience to your style, which is just what a PC port should offer.

The PC version is superb, and the game is one of the bright spots in this fall’s slate of games.
14 out of 14 PC-specific mentions in critic reviews on Metacritic are positive

“Very Positive” - 82% on Steam
Success
PC Platform
Agenda

Game Ready Quality Program
Partnering with Watch Dogs 2 PC - Through The Numbers
Ubisoft PC Quality, and working with NVIDIA
Technical Case Study: Eliminating Stutter
Learnings
Game Ready Quality Program

Jaakko Haapasalo (NVIDIA)
Program Overview

1. PC Technical Requirements Checklist

2. Stability, Performance & Stutter Analysis

3. Minimum & Recommended Specs

developer.nvidia.com
(see also)

Game Ready Drivers

Optimal Performance Settings

GTM
Title Engagement

T - 12mo  T - 6mo  T - 3mo  T - 2mo  T - 1mo  Launch  Live

Testing & Optimization  GTM

Review
1. **TRC** (Technical Requirements Checklist)

PC platform requirements to validate your game against

Focused on technical quality and readiness, such as resolution support, UI scaling, frame rate, smoothness, settings, and correctness of various effects.

Review involves several play-throughs at different settings, significant investment in QA time

3 review milestones

40 requirements

16 titles tested in 2016 pilot
Requirement Categories

• Required (P0)
  • Core
• Recommended (P1)
  • Quality of Life
• Advised (P2)
  • Forward-looking
Requirements and Rating

• Description
• Additional information (context, intent and failure modes)
• Criteria for Full, Partial and Failed compliance

• Overall TRC rating as a weighted average (0..100)
2. Stability, Performance and Stutter

Get
Build

Improve

Test

Analyze

developer.nvidia.com
3. Minimum & Recommended Specs

Benchmark Search (Playthrough!)

"Challenging" Benchmark

MinRec Search GPU x CPU x SYSMEM

60FPS Target

1K, 2K, 4K

Low / High Presets

High: IQ > Any Other Platform

MinSpec

RecSpec
Timeline

Title Engagement

- T - 12mo
- T - 6mo
- T - 3mo
- T - 2mo
- T - 1mo

Launch

GTM

Review

- TRC Preview
- TRC Update
- TRC Final Results

Min/Rec Spec Prelims
Min/Rec Spec Results
Min/Rec Check
Min/Rec Check

Performance & Stutter Analysis

developer.nvidia.com
WD2 Game Ready: The Numbers

Jaakko Haapasalo (NVIDIA)
PC Testing

TWIMTBP Labs:
- 7 months
- 26 builds
- 10 Stutter & performance tests
- 5 Min & Rec Spec tests
- 3 TRC reviews
Min & Recommended Specs: 1K & 2K

- Series 1
- Series 2
- Series 3

Date Range: 1-Jan to 5-Jan
Partnering with NVIDIA

Marius Tudorache (Ubisoft)
New NVIDIA-Ubisoft collaboration process

• Transparent communication
• Smoke-tested builds
• Delivery of builds and performance reports
• TRC simulations
• Post-Mortem and learnings across projects
• Dedicated JIRA for each project - tracking bugs outside of e-mails

! Think co-dev instead of tech support !
PC Testing - UBI PC Requirements

• 13 months (1st check - October 2015)
• 320+ builds (WD2-GAME-PC2->PC15x + RAW)
• ~28 performance/TRC tests
• ~28 Min & Rec Spec tests
• 7 TRC official reviews (Alpha/Beta/Master/D1P)
Ubisoft - Winning PC gamers

After 20 years of PC releases:

• Knowledge & Information Sharing Management for PC
  • Internal PC Summits
• Improving PC Technical Requirements
• Developing PC Communities - internally, externally, E-Sports
• Improving communication with PC gamers
Current UBI TRC Structure (WD2 proto.)

101 Tech. requirements

- Short title - for easy referencing
- Requirement description
- Terminology
- Remarks
- Intent
- Exemption
- TRC Rating per requirement - different for each milestone
- Overall Rating - work in progress
Using target TRCs as tech design

• Clarity
• Transparency
• Conformity

• Predictability
• Continuity

Major Challenge: Public Tech Requirements (open-sourced to communities)
WD2 communicated Min & Reco

We wanted to express:

• Performance-based specs split per GPU series
• Coverage (old to new gen)
• Clarity for casual gamers
<table>
<thead>
<tr>
<th>KEPLER</th>
<th>KEPLER refresh</th>
<th>MAXWELL</th>
<th>PASCAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>12500</td>
<td></td>
<td></td>
<td>GeForce GTX 1080</td>
</tr>
<tr>
<td>12000</td>
<td></td>
<td></td>
<td>GeForce GTX 980 Ti</td>
</tr>
<tr>
<td>11500</td>
<td></td>
<td></td>
<td>GeForce GTX 1070</td>
</tr>
<tr>
<td>11000</td>
<td></td>
<td></td>
<td>GeForce GTX 1060</td>
</tr>
<tr>
<td>10500</td>
<td></td>
<td></td>
<td>GeForce GTX 780 Ti</td>
</tr>
<tr>
<td>10000</td>
<td></td>
<td></td>
<td>GeForce GTX 970</td>
</tr>
<tr>
<td>9500</td>
<td></td>
<td></td>
<td>GeForce GTX 780</td>
</tr>
<tr>
<td>9000</td>
<td></td>
<td></td>
<td>GeForce GTX 780 Ti</td>
</tr>
<tr>
<td>8500</td>
<td></td>
<td></td>
<td>GeForce GTX 770</td>
</tr>
<tr>
<td>8000</td>
<td></td>
<td></td>
<td>GeForce GTX 960</td>
</tr>
<tr>
<td>7500</td>
<td></td>
<td></td>
<td>GeForce GTX 770</td>
</tr>
<tr>
<td>7000</td>
<td></td>
<td></td>
<td>GeForce GTX 760</td>
</tr>
<tr>
<td>6500</td>
<td></td>
<td></td>
<td>GeForce GTX 950</td>
</tr>
<tr>
<td>6000</td>
<td></td>
<td></td>
<td>GeForce GTX 760</td>
</tr>
<tr>
<td>5500</td>
<td></td>
<td></td>
<td>GeForce GTX 960</td>
</tr>
<tr>
<td>5000</td>
<td></td>
<td></td>
<td>GeForce GTX 680</td>
</tr>
<tr>
<td>4500</td>
<td></td>
<td></td>
<td>GeForce GTX 770</td>
</tr>
<tr>
<td>4000</td>
<td></td>
<td></td>
<td>GeForce GTX 660 Ti</td>
</tr>
<tr>
<td>3500</td>
<td></td>
<td></td>
<td>GeForce GTX 950</td>
</tr>
<tr>
<td>3000</td>
<td></td>
<td></td>
<td>GeForce GTX 660</td>
</tr>
</tbody>
</table>

PASSMARK VALUES
Detecting and Eliminating Stutter

Farid Rzaev (NVIDIA) and Oleh Kuznetsov (Ubisoft)
Agenda

- Stutter overview
- Stutter detection
- Stutter analysis
- Fixing stutter
- Results
Stutter? Hitches?

Everybody knows what it is....
Let's Define

What stutter/hitches are?

- Inconsistency in performance between frames
Player’s Point of View

Occasional freezes

Uncomfortable feeling under **recommended specs**
Stutter is a Huge Thing

Focus on our experience with Watch Dogs 2

Catching up issues
- Tools

Fixing issues
- Methods / Approaches
Finding Stutter Cases

Watch Dogs 2 is an open world action-adventure game

- Each stutter case should be covered by a benchmark
In-game Benchmark

We had an internal in-game benchmark, but it didn’t cover all of the cases.
Two Benchmarks

In-game

Fast Car

www.gameworks.nvidia.com
Frametime graph (ms)
Tools

- GPUView
- Concurrency visualizer (CV)
Stutter issues

1. GPU Memory consumption  \rightarrow  GPU View / Afterburner
2. CPU Workload Inconsistency  \rightarrow  GPU View / CV
3. Bursts of creates / destroys  \rightarrow  GPU View
4. Page faults  \rightarrow  CV / GPU View
GPU Memory consumption

- Running out of the video memory
- Didn’t account some video memory for the driver (~10%)

Tools:
- GPUView eviction events
- GPUView reference charts
UBI - Memory Consumption

Approaches

• Check VRAM usage (lighting, models, textures, post effects, …)
• Give artists the possibility to tune MIP maps skipping on a per texture basis
• Look for the allocated but not used resources.
• WD2: For the low settings we skipped Global Illumination probes for Far Away objects

Challenges

• Visual quality on the low settings
• Textures in RAM if not enough VRAM
CPU Utilization Inconsistency

100 ms
WD2: We use all CPU threads, but leave some idle time for a driver and other processes

Frame structure:

<table>
<thead>
<tr>
<th>Main Render</th>
<th>Driver</th>
<th>Render</th>
<th>Render</th>
<th>Render</th>
<th>Present</th>
</tr>
</thead>
<tbody>
<tr>
<td>Worker</td>
<td>Engine</td>
<td>Render</td>
<td>Windows</td>
<td>Idle</td>
<td>Idle</td>
</tr>
<tr>
<td>Worker</td>
<td>Render</td>
<td>Driver</td>
<td>Engine</td>
<td>Idle</td>
<td>Idle</td>
</tr>
<tr>
<td>Main Engine</td>
<td>Engine</td>
<td>Engine</td>
<td>Engine</td>
<td>Idle</td>
<td>Idle</td>
</tr>
</tbody>
</table>

UBI - CPU Utilization Inconsistency
Multithread rendering (Using deferred context)
Use Present time to do engine jobs

<table>
<thead>
<tr>
<th>Main Render</th>
<th>Present</th>
<th>Render</th>
<th>DC:Execute</th>
<th>DC:Execute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Worker</td>
<td>Engine</td>
<td>Driver</td>
<td>Render</td>
<td>Driver</td>
</tr>
<tr>
<td>Worker</td>
<td>Engine</td>
<td>DC:FillList</td>
<td>Render</td>
<td>Windows</td>
</tr>
<tr>
<td>Main Engine</td>
<td>Engine</td>
<td>Engine</td>
<td>Engine</td>
<td>DC:FillList</td>
</tr>
</tbody>
</table>
Bursts of creates / destroys

Too many of creates / destroys of resources on the render thread
## UBI - Resource Management

Creating resources on the main render thread:

<table>
<thead>
<tr>
<th>Main Render</th>
<th>Resource</th>
<th>Resource</th>
<th>Resource</th>
<th>Render</th>
<th>Present</th>
</tr>
</thead>
<tbody>
<tr>
<td>Worker</td>
<td>Engine</td>
<td>Render</td>
<td>Engine</td>
<td>Idle</td>
<td>Idle</td>
</tr>
<tr>
<td>Worker</td>
<td>Render</td>
<td>Engine</td>
<td>Idle</td>
<td>Idle</td>
<td>Idle</td>
</tr>
<tr>
<td>Main Engine</td>
<td>Engine</td>
<td>Engine</td>
<td>Engine</td>
<td>Idle</td>
<td>Idle</td>
</tr>
</tbody>
</table>
Creating resources on the worker threads:

<table>
<thead>
<tr>
<th>Main Render</th>
<th>Render</th>
<th>Render</th>
<th>Render</th>
<th>Render</th>
<th>Present</th>
</tr>
</thead>
<tbody>
<tr>
<td>Worker</td>
<td>Engine</td>
<td>Resource</td>
<td>Resource</td>
<td>Idle</td>
<td>Idle</td>
</tr>
<tr>
<td>Worker</td>
<td>Engine</td>
<td>Engine</td>
<td>Resource</td>
<td>Render</td>
<td>Idle</td>
</tr>
<tr>
<td>Main Engine</td>
<td>Engine</td>
<td>Engine</td>
<td>Engine</td>
<td>Engine</td>
<td>Engine</td>
</tr>
</tbody>
</table>
UBI - Resource Management

Resource jobs between the frames:

Frame | Frame
--- | ---
Render | Render | Render | Render | Present | Render | Render
Engine | Resource | Engine | Idle | Engine | Engine | Idle
Resource | Engine | Resource | Engine | Engine | Engine | Engine
Engine | Engine | Engine | Idle | Engine | Engine | Engine
UBI - Resource Management

Approaches

• Avoid resource creation on the main render thread
Bursts of creates / destroys

Too many of creates / destroys of resources on multiple threads
UBI - Bursts of creates / destroys

Approaches

• Limit resource creation per frame
• The lower minimum limits you set - the less spikes you have
• Use texture pool to decrease Release calls
• Use typeless formats in texture pool

Challenges

• Resource Release can take significant amount of time but is hard to be measured
Page Faults - HDD specific

- HDD specific Page Faults that lead to stutter
- Windows cache’s HDD reads
  - Page Faults were only seen on the FIRST run of the game
Page Faults - CV
UBI - Using the System File Cache

Approaches

• Any free memory on Windows PC may be used as the file cache
• Fill windows file cache by game data files before they are requested
• Use logos/intro videos and idle menu time to fill the cache
• Cache the most important files first

Challenges

• Not enough time to cache the entire game
• Not enough free memory to cache the entire game
Result

• It took us 4-5 months: Start 08/2016 - End 12/2016

• Weekly meetings

• Weekly builds with updates and summaries

• During the week: local skype chats and DAILY calls

• Super fast responses on both sides:
  1. NVIDIA: catching and reporting issues, suggestions
  2. UBI: fixing and sending new builds in a short time
Result

Feel comfortable under **recommended specs**
GPU Memory eviction
GPU Memory Reference Charts
CPU Workload Inconsistency

100 ms

www.gameworks.nvidia.com
References

• *Stuttering in Game Graphics: Detection and Solutions* - Cem Cebenoyan

• *Analyzing Stutter - Mining More from Percentiles* - Iain Cantlay

• *SLI and Stutter Avoidance* - Iain Cantlay, Lars Nordskog
Learnings
Take Home

• Apply technical validation criteria before designing
• Adapt code as well as design for best results
• Playtest your game with casual AND enthusiast hardware
• UBI PC Requirements and NVIDIA TRCs are aligned
Take Home

• NVIDIA Technical Requirements Checklist is a great way to get started focusing on the PC as a Platform

• Low performance and Stutter still pose major risks for games today; isolate them early, and iterate aggressively (come talk to us!)

• Partnering with the Game Ready Quality Program will help us support your vision for superior PC Quality
Thank you!

Email jhaapasalo@nvidia.com for a sneak peek

Look out for more information on http://developer.nvidia.com soon!