8 Frames in 16ms

Rollback Networking in Mortal Kombat and Injustice

Michael Stallone
Lead Software Engineer – Engine
NetherRealm Studios
mstallone@netherrealm.com

What is this talk about?

The how, why, and lessons learned from switching our network model from lockstep to rollback in a patch.

Staffing

- 4-12 concurrent engineers for 9 months
- Roughly 7-8 man years for the initial release
- Ongoing support is part time work for ~6 engineers

Terminology

- RTT Round trip time. Time a packet takes to travel from Client A > Client B > Client A
- Network Latency One way packet travel time
- Netpause Game pauses due to not receiving data from remote client for too long
- QoS Quality of Service. Measurement of connection quality

Terminology

- **Input Latency** Injected delay between a button press and engine response
- **Confirm frame** Most recent frame with input from all players
- **Desync** Clients disagree about game state, leads to a disconnect
- **Dead Reckoning** Networking model. Uses projections instead of resimulation

Basics

- Hard 60hz 1v1 fighting game
- Peer to Peer
- A network packet is sent once per frame
- Standard networking tricks to hide packet loss

Determinism

The vast majority of our game loop is bit-for-bit deterministic.

We “fencepost” many values at various points in the tick, and any divergence causes a desync.

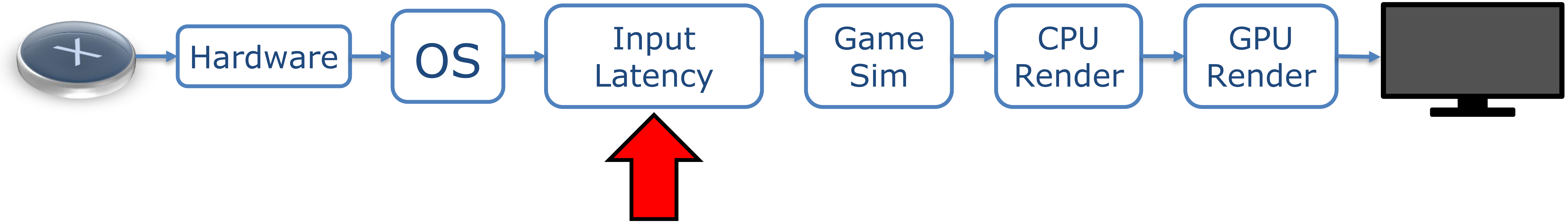
This is the foundation that everything is built on.

The Problem

Our online gameplay suffered from inconsistent (and high) input latency.

The players were not happy.

Latency Diagram



Lockstep

Only send gamepad data

The game will not proceed until it has input from the remote player for the current frame

Input is delayed by enough frames to cover the network latency

Lockstep

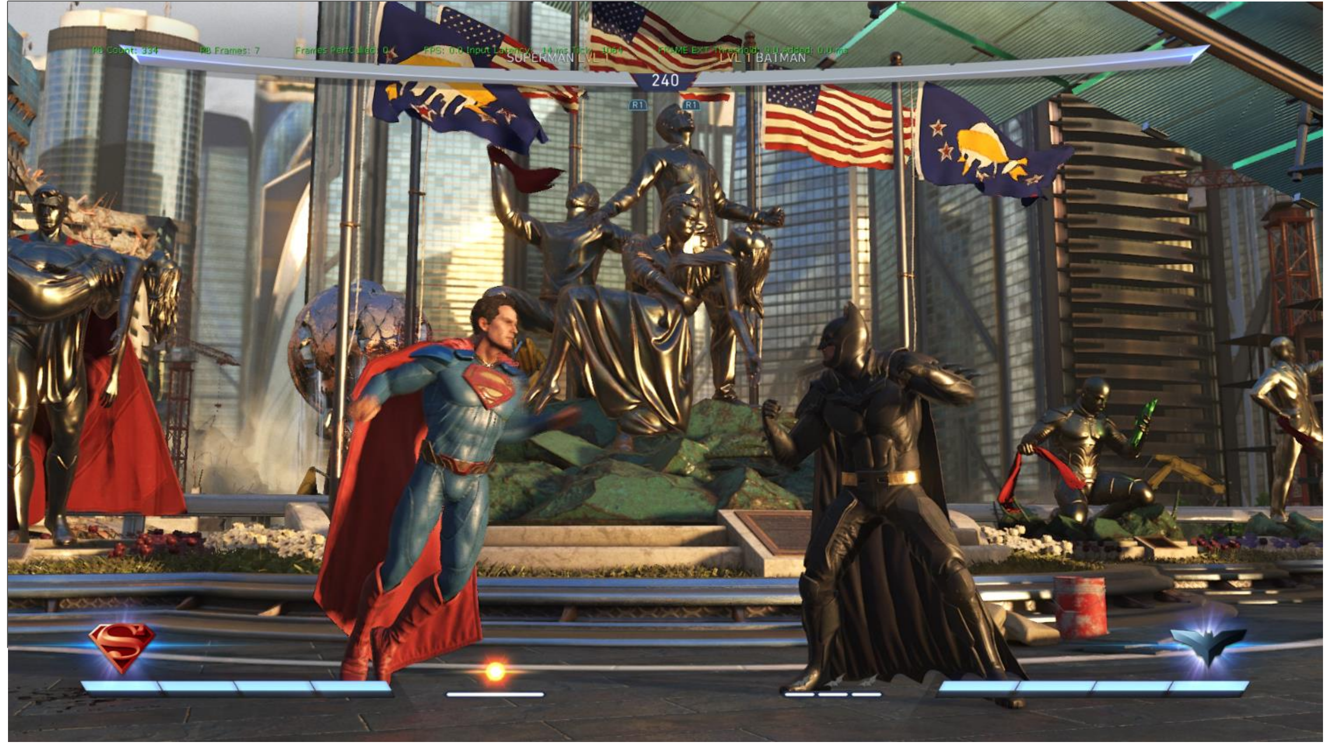
Current Frame

Future Frames

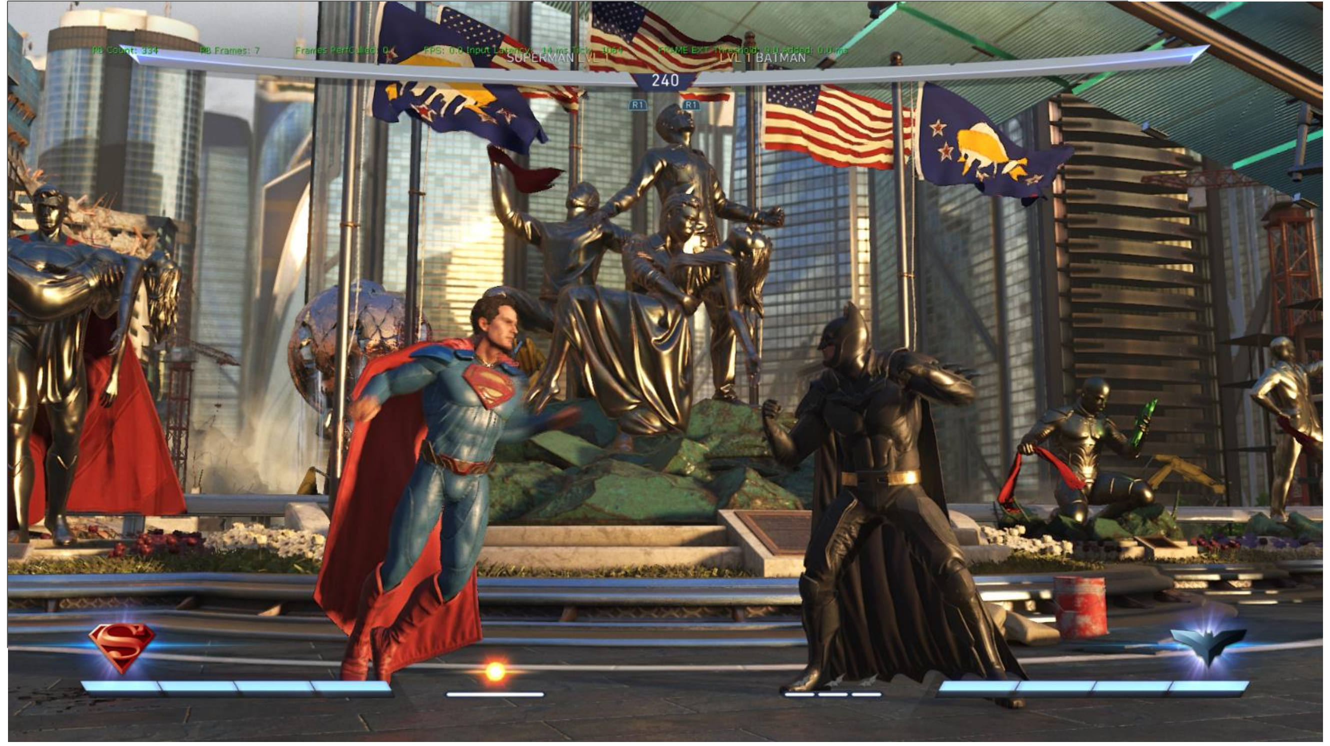
Player 1 1 2 3

Pad Input → X 8

Player 1



Player 2



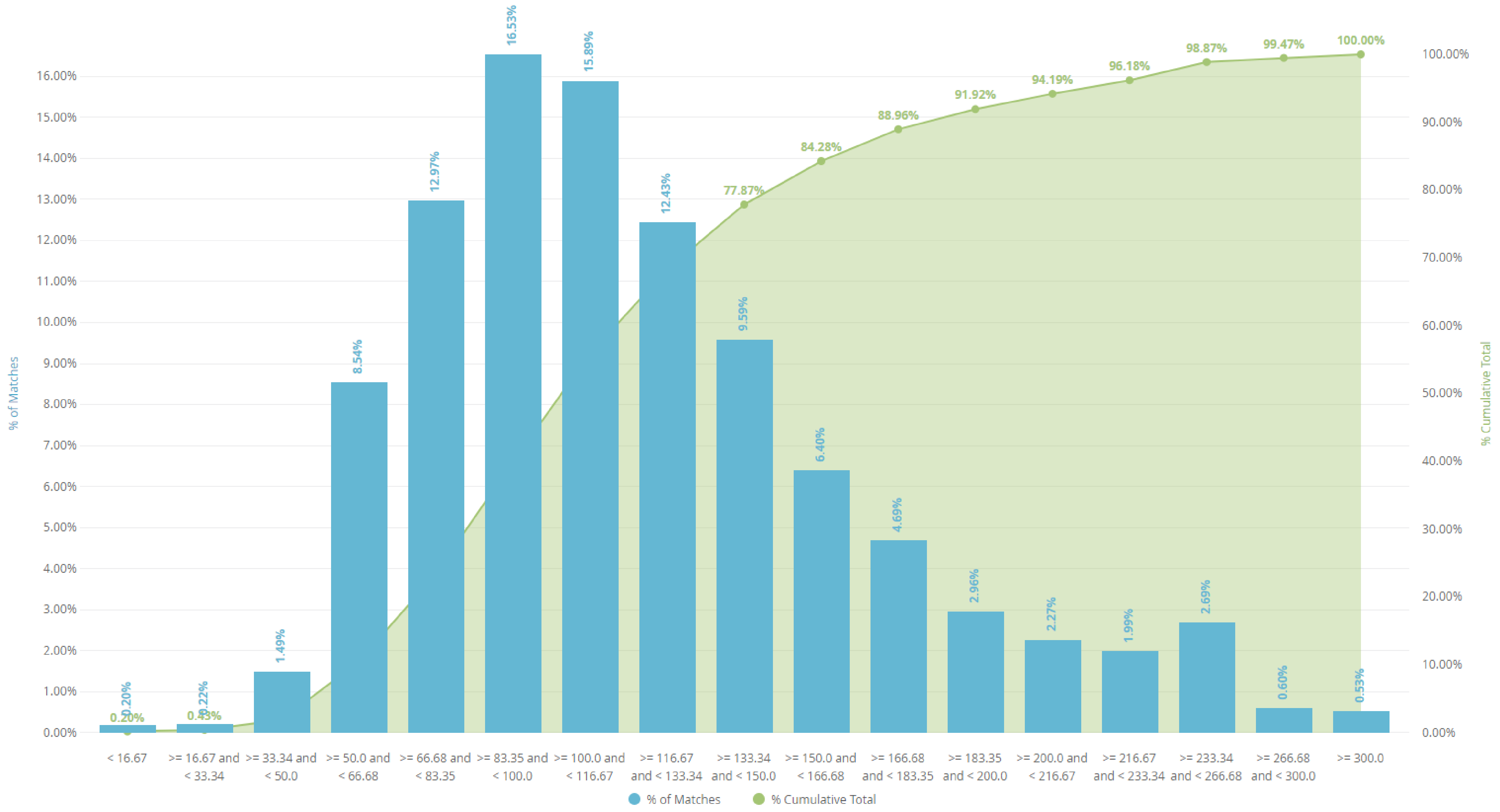
Player 2 1 2 3

The Present

Mortal Kombat X and Injustice 2 have 3 frames of input latency and support up to 10 frames (333ms) of network latency before pausing the game.

The online experience is much improved and the players are happy.

Latency Curve



Rollback

Only send gamepad data

Game proceeds without remote input

When remote input is received, rollback and simulate forward

Rollback

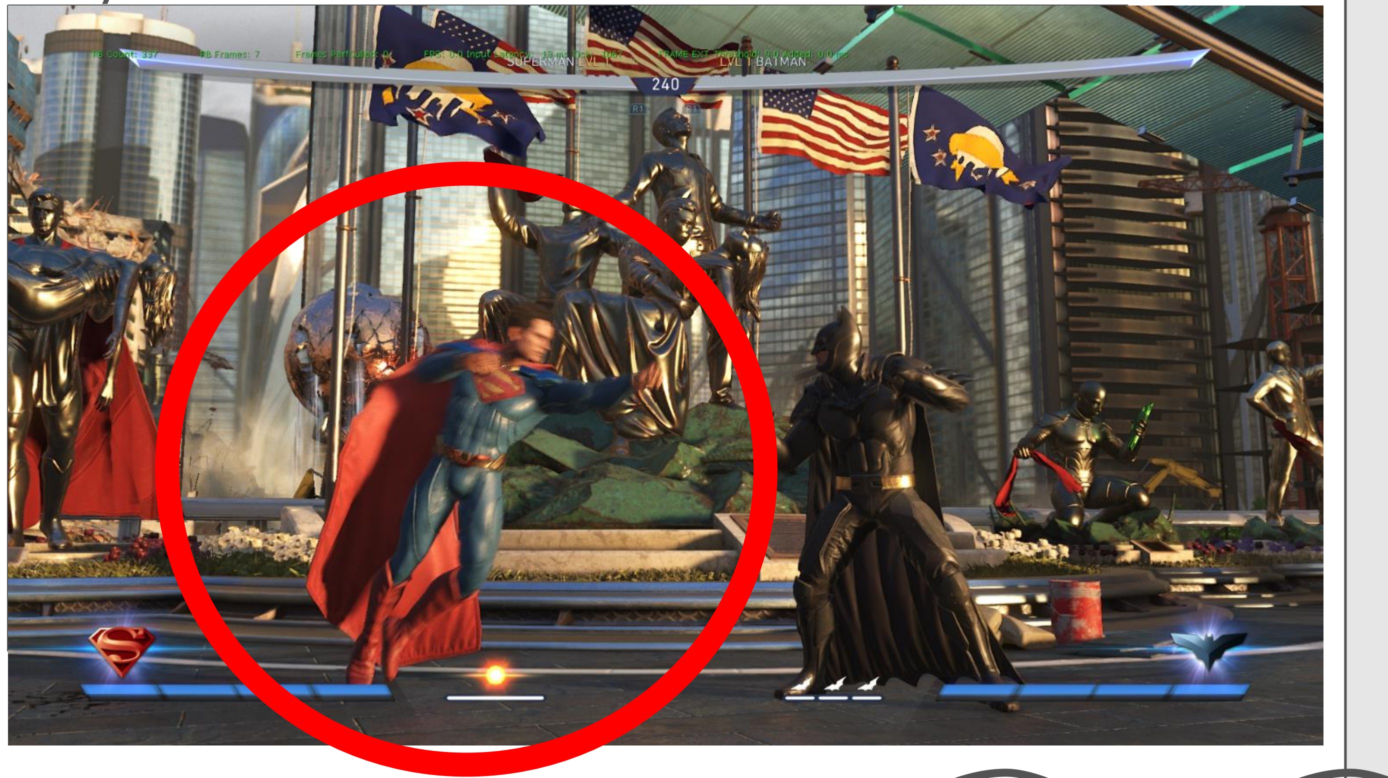
Current Frame

Future Frames

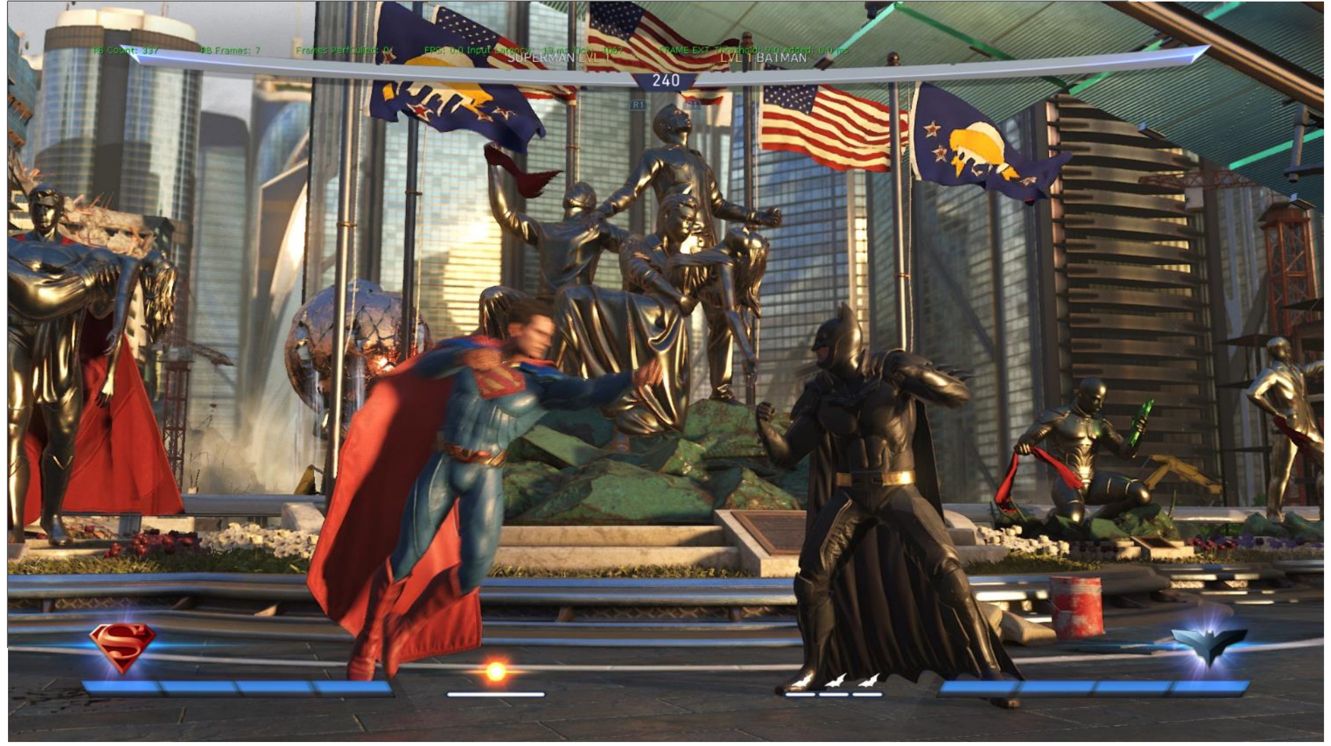
Player 1 1 2 3

Pad Input → X 4

Player 1



Player 2



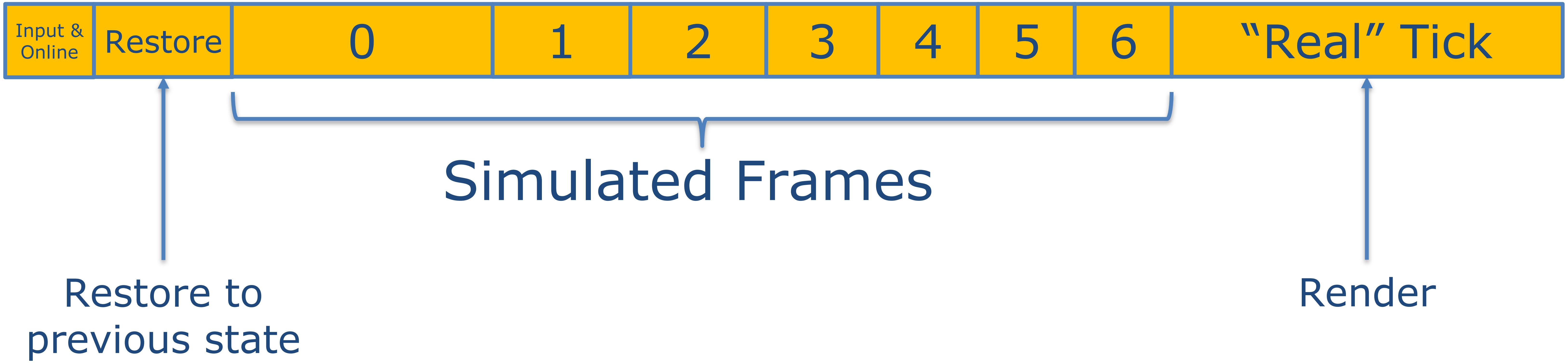
Player 2 4 5 6 1 2 3

	Rollback	Lockstep
Simple		X
Visually Smooth		X
Performant		X
Robust	X	X
Low Bandwidth	X	X
Responsive	X	
Single Frame Latency	X	

What did we do first?

- First goal was to get an idle character rolling back
- Turn off almost everything
- Serialization (Saving/Restoring previous state)
- Debug mode that constantly rolled back (SetRollbackFrames 7)

Tick Timeline (when rolling back)



Input & Online	Restore	0	1	2	3	4	5	6	"Real" Tick

Simulated Frames

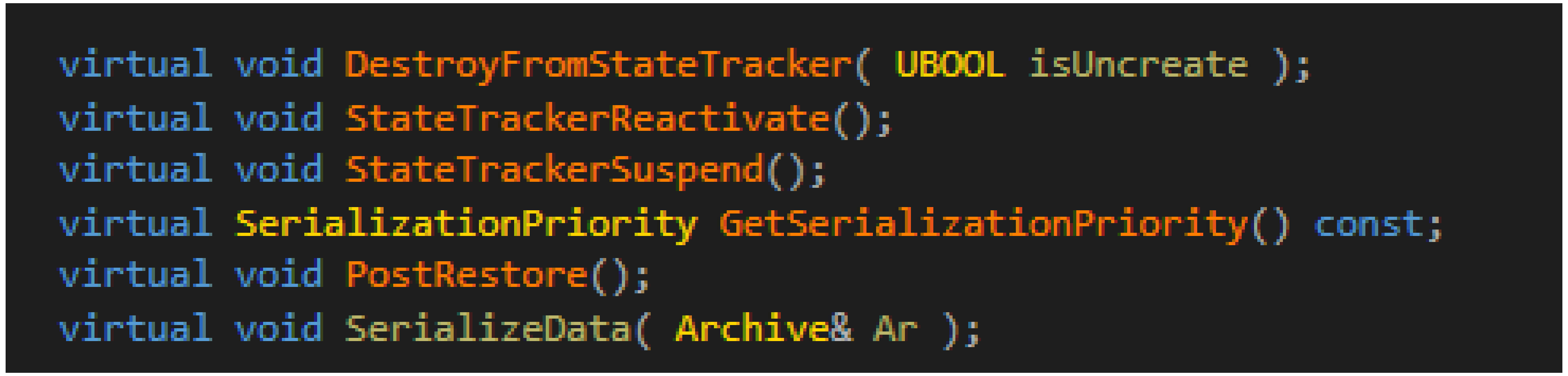
Restore to previous state

Render

Serialization: Save

Rollback framework:

- Ring buffer (sized to rollback window)
- Object serialization interface
- Contains entries for object creation/destruction
- Only save mutable data
- Not delta based



```
virtual void DestroyFromStateTracker( UBOOL isUncreate );
virtual void StateTrackerReactivate();
virtual void StateTrackerSuspend();
virtual SerializationPriority GetSerializationPriority() const;
virtual void PostRestore();
virtual void SerializeData( Archive& Ar );
```

Serialization: Restore

- Parallel serialization
  - Cannot use shared_ptr
  - Simple load balancing and priority scheme
  - Large perf win (2.7ms > 1.3ms for double the work)
  - Waking threads is a bit slow
- Single threaded post-serialization fixup
  - Can coordinate with non-rollback systems
- Bulk-serialization and immutable data are hugely preferred

Object lifetime

Deferred Deletion

- Objects remain alive until their time of death is outside the rollback window
- Generally easier
- Code in destructors is dangerous
- Use handles
- Usually more performant

Delete and Recreate

- Delete objects as normal and create them again if needed
- This is the default
- Slow (unless reusing objects)
- Increased serialization
- Follows “normal” construction and destruction patterns

Recreatables

- Avoid creating the same object using “Re-Creatables”
- Used per type hashing to detect when an object was “identical”
- Sounds & particles were recreatable
  - Can be nondeterministic
  - Nondeterministic simulation means object reuse was mandatory
- Avoids wasteful creation
- Visual/Audio “correctness” without full serialization burden

What about gameplay script?

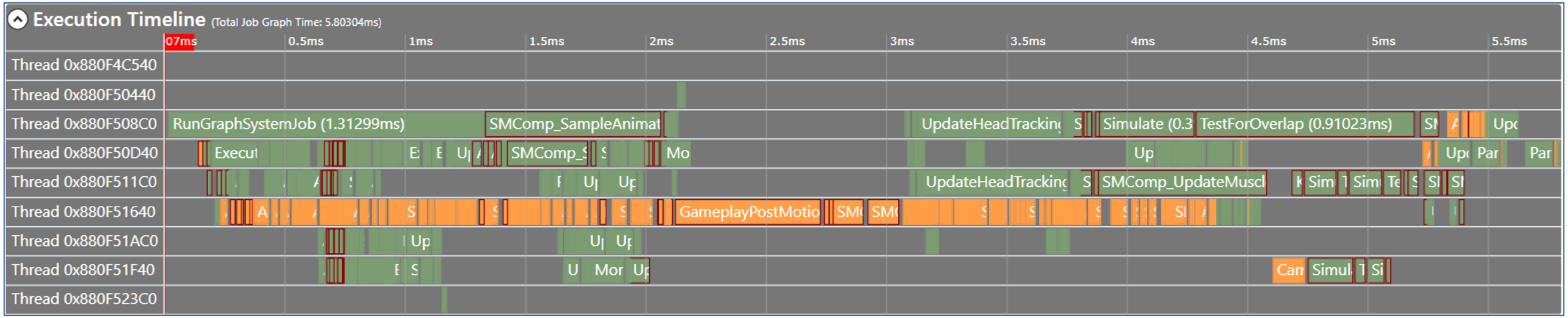
- Fiber based proprietary gameplay script
  - Fiber stack unwinding
  - Fiber stack serialization
  - Objects on the stack that require destructors can be a problem
    - We registered these objects with another system for cleanup

Rollback Artifacts

- When rollbacks occur, there can be a visual pop
- The extent of divergence varies wildly
  - Mostly minor
- Avoid rolling back large visual changes

How was performance?

- Bad. Real bad.
- Before rollbacks, we idled at 9-10ms on the CPU
- After initial rollback support, we idled at 30+ms
- Headroom due to console generation jump … GONE!
- Tons of free cores

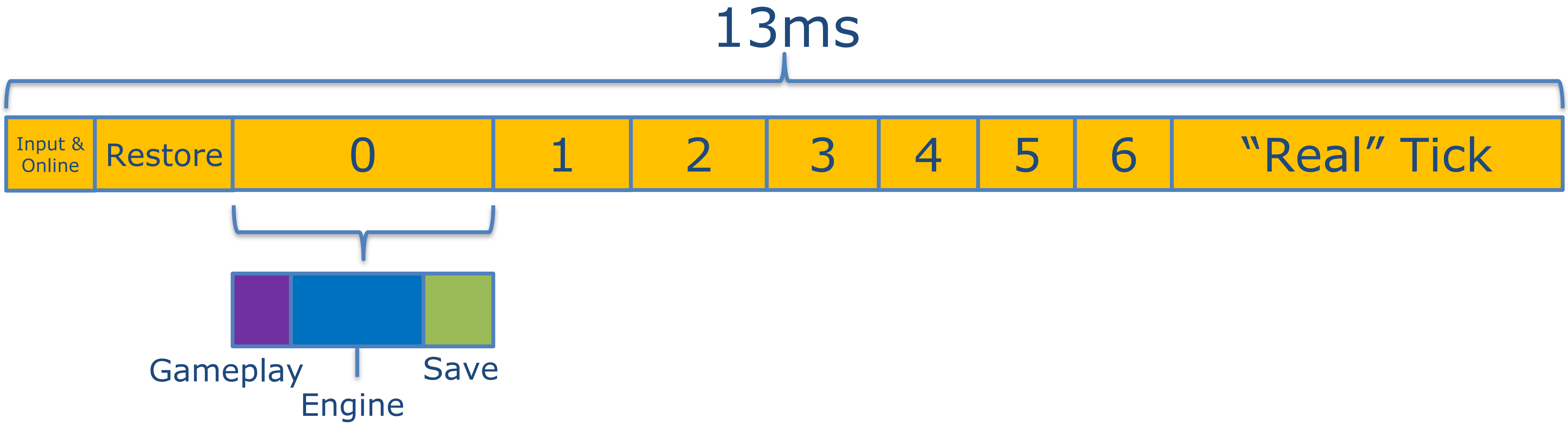


Performance Tools

- Sony/Microsoft perf tools
- Job Graph visualizer (task graph)
- Rollback loop (SUPER PAUSE!)
- PET Profiler
- Performance bots

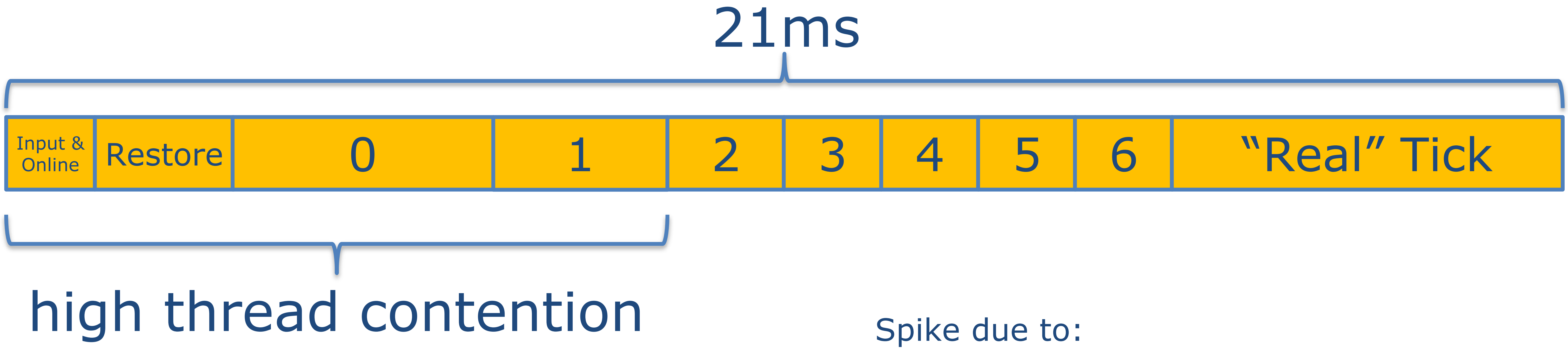
Tick Timeline

7 frame rollback, shipped Injustice 2 (idle)



Tick Timeline

7 frame rollback, shipped Injustice 2
(Green Arrow spike frame)



21ms

Input & Online	Restore	0	1	2	3	4	5	6	"Real" Tick

high thread contention

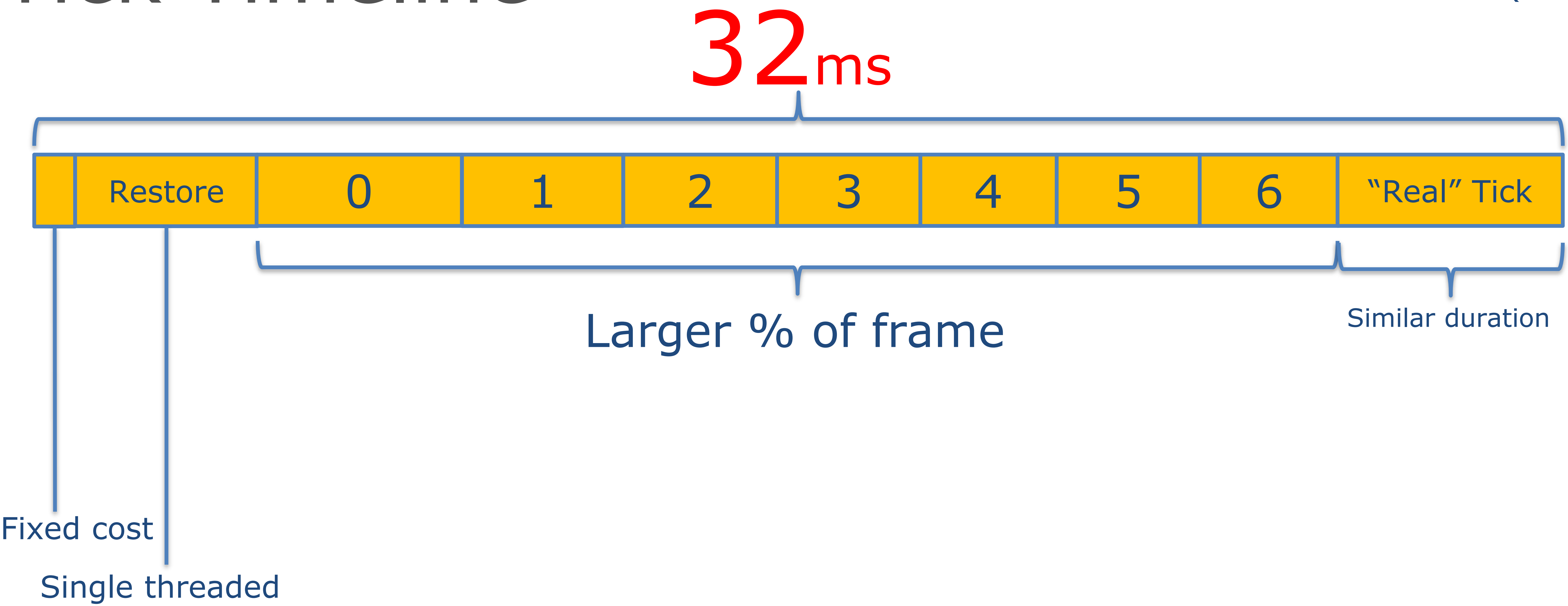
Spike due to:

- Mesh spawning
- Particle spawning
- Particle attachment
- Extra gameplay procs

Spike can persist for 8 frames!

Tick Timeline

7 frame rollback, Mortal Kombat X (idle)



32ms

Restore

0

1

2

3

4

5

6

“Real” Tick

Larger % of frame

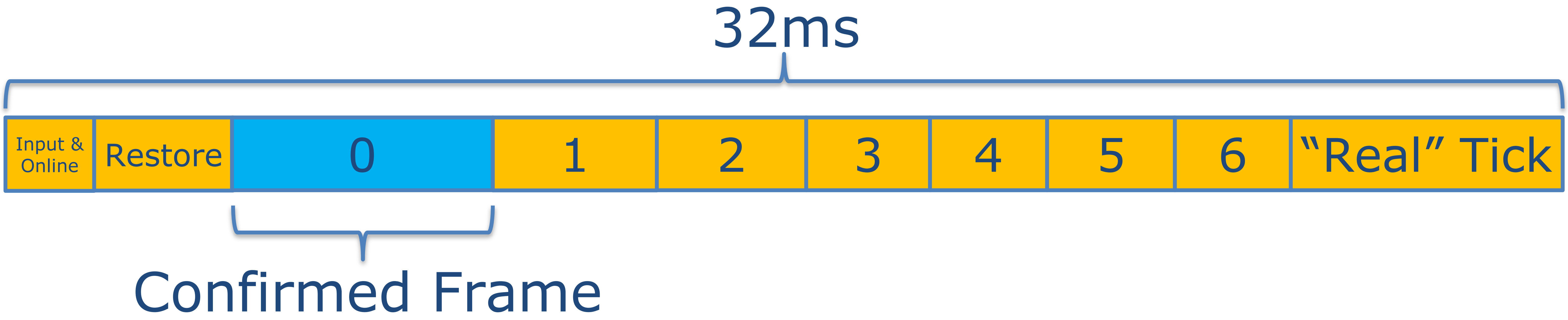
Similar duration

Fixed cost

Single threaded

Tick Timeline

7 frame rollback, Mortal Kombat X (idle)



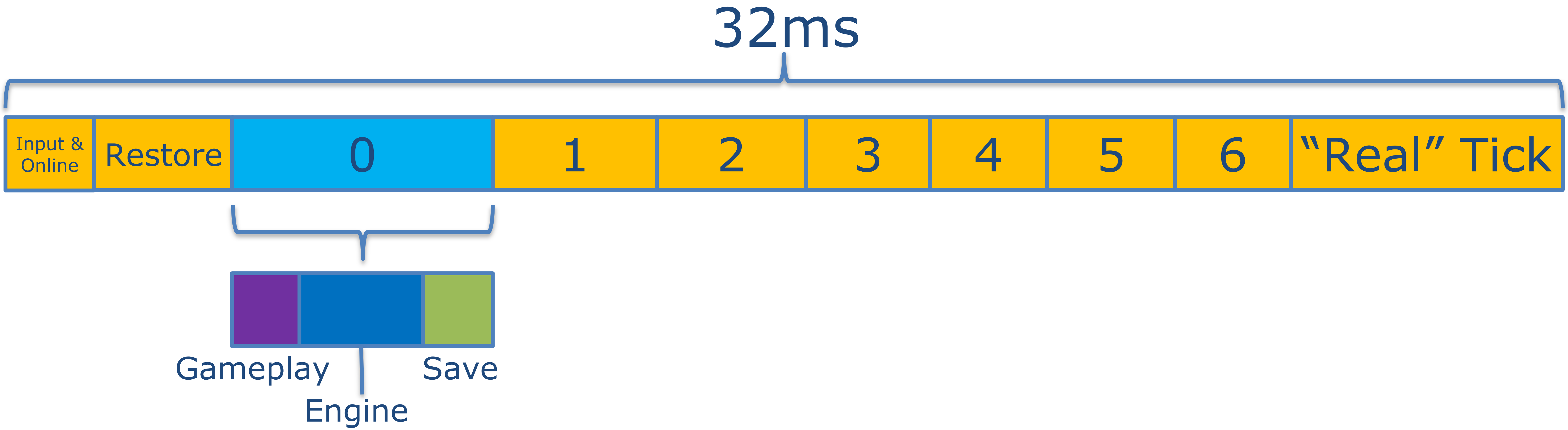
32ms

Input & Online	Restore	0	1	2	3	4	5	6	"Real" Tick

Confirmed Frame

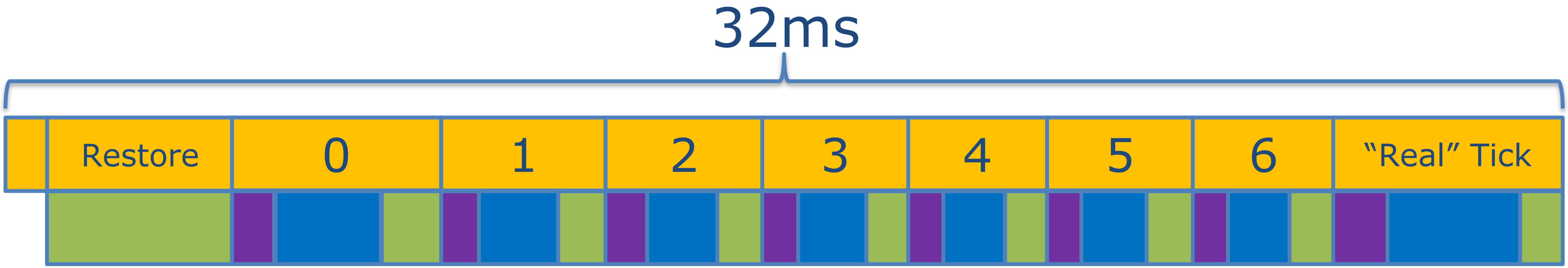
Tick Timeline

7 frame rollback, Mortal Kombat X (idle)



Tick Timeline

7 frame rollback, Mortal Kombat X (idle)



32ms

Restore

0

1

2

3

4

5

6

“Real” Tick

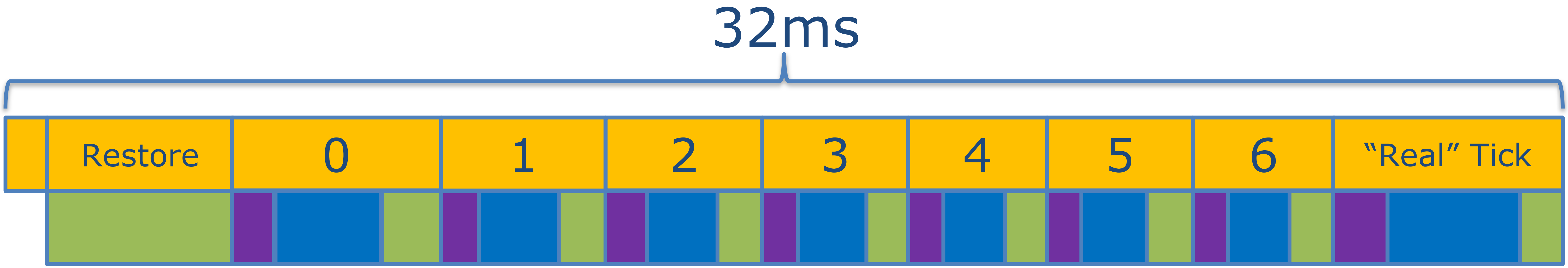
- 🟪 Gameplay
- 🟦 Engine
- 🟩 Save/Restore

Turn off everything cool

- Physics/Cloth
- Raycasts that don’t effect gameplay
- IK
- Particle effects
- Online
- Desync detection

Tick Timeline

7 frame rollback, Mortal Kombat X (idle)



32ms

Restore 0 1 2 3 4 5 6 "Real" Tick

🟪 Gameplay

🟦 Engine

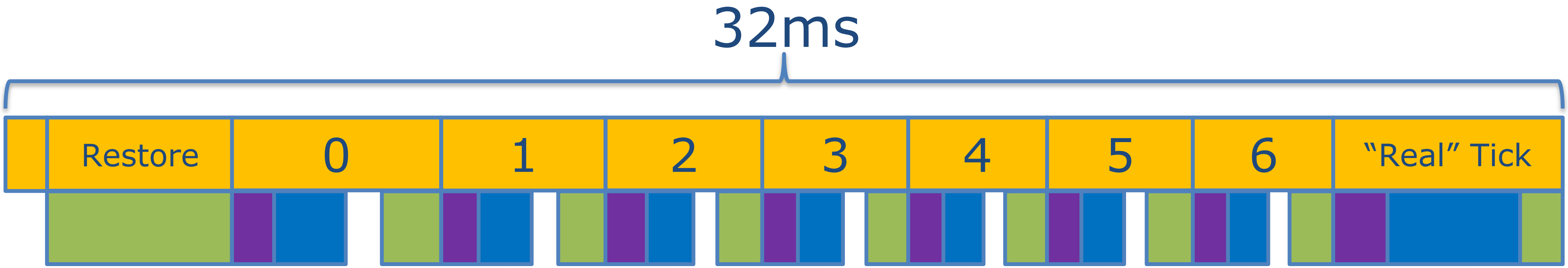
🟩 Save/Restore

Easy performance wins

- Why are we strcmping?
- Don’t do that 8 times
  - Controller polling
  - Garbage collection
- Opt out of system/object updates during simulation
- Death by a thousand cuts
  - Dynamic memory allocs
  - Pointer chasing
  - Walking sparse lists

Tick Timeline

7 frame rollback, Mortal Kombat X (idle)



🟪 Gameplay
🟦 Engine
🟩 Save/Restore

More difficult performance wins

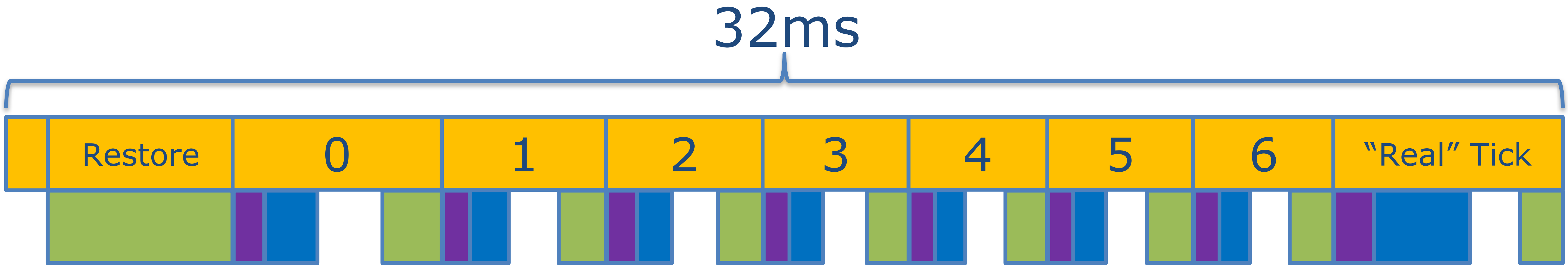
- Promotable re-simulation behavior
- Aggressive parallelization
- Graph optimizations

- Asynchronous UI/Audio ticking
- Automatic emitter parallelization
- Animation pre-sampling
- Simplify common graphs

- Special case complex cases
- Change graph types JIT
- Remove false dependencies
- More job priority levels

Tick Timeline

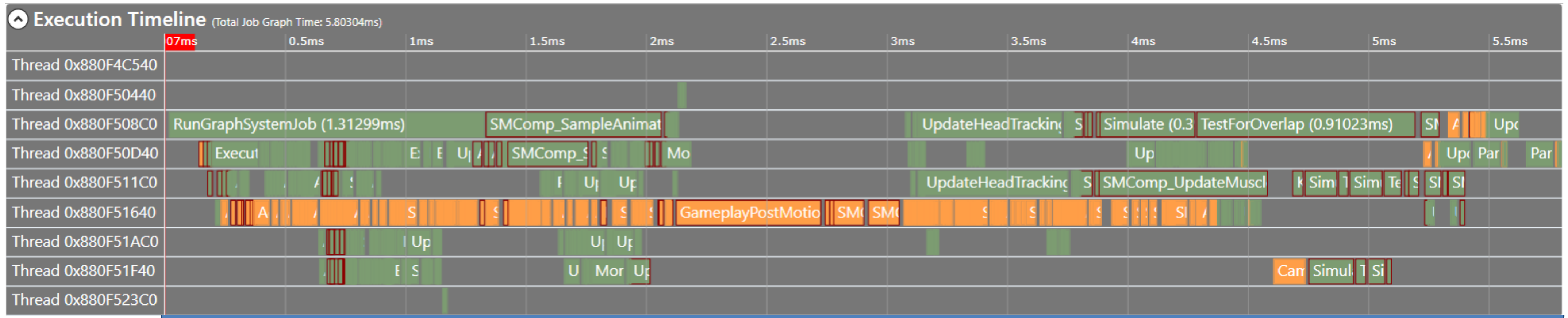
7 frame rollback, Mortal Kombat X (idle)



🟪 Gameplay
🟦 Engine
🟩 Save/Restore

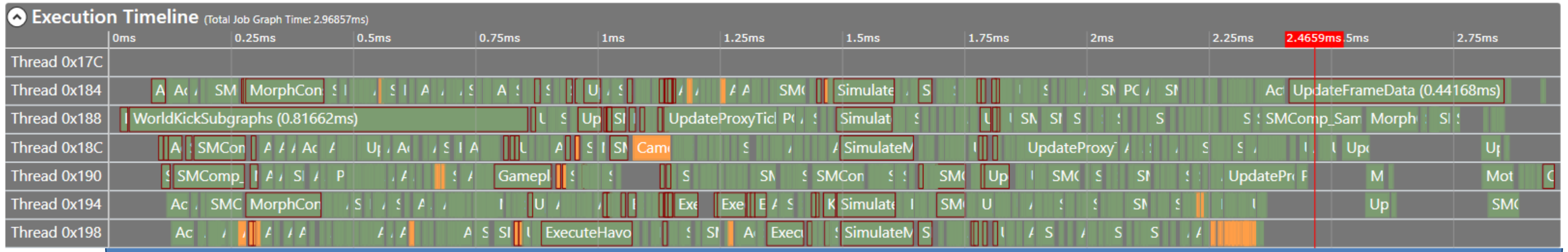
You’re only as fast as the critical path

Early Graph



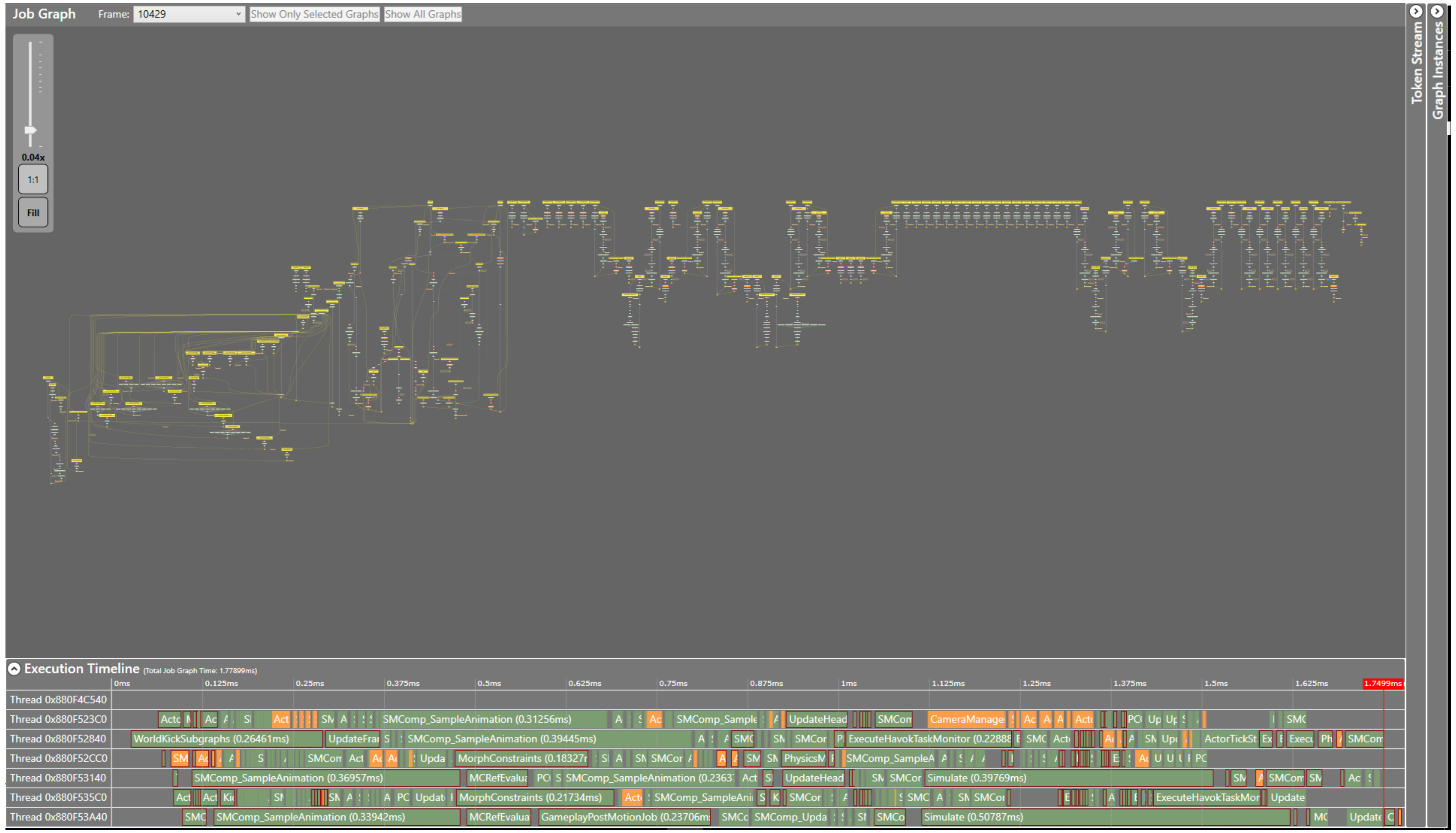
~6ms

Shipping Graph

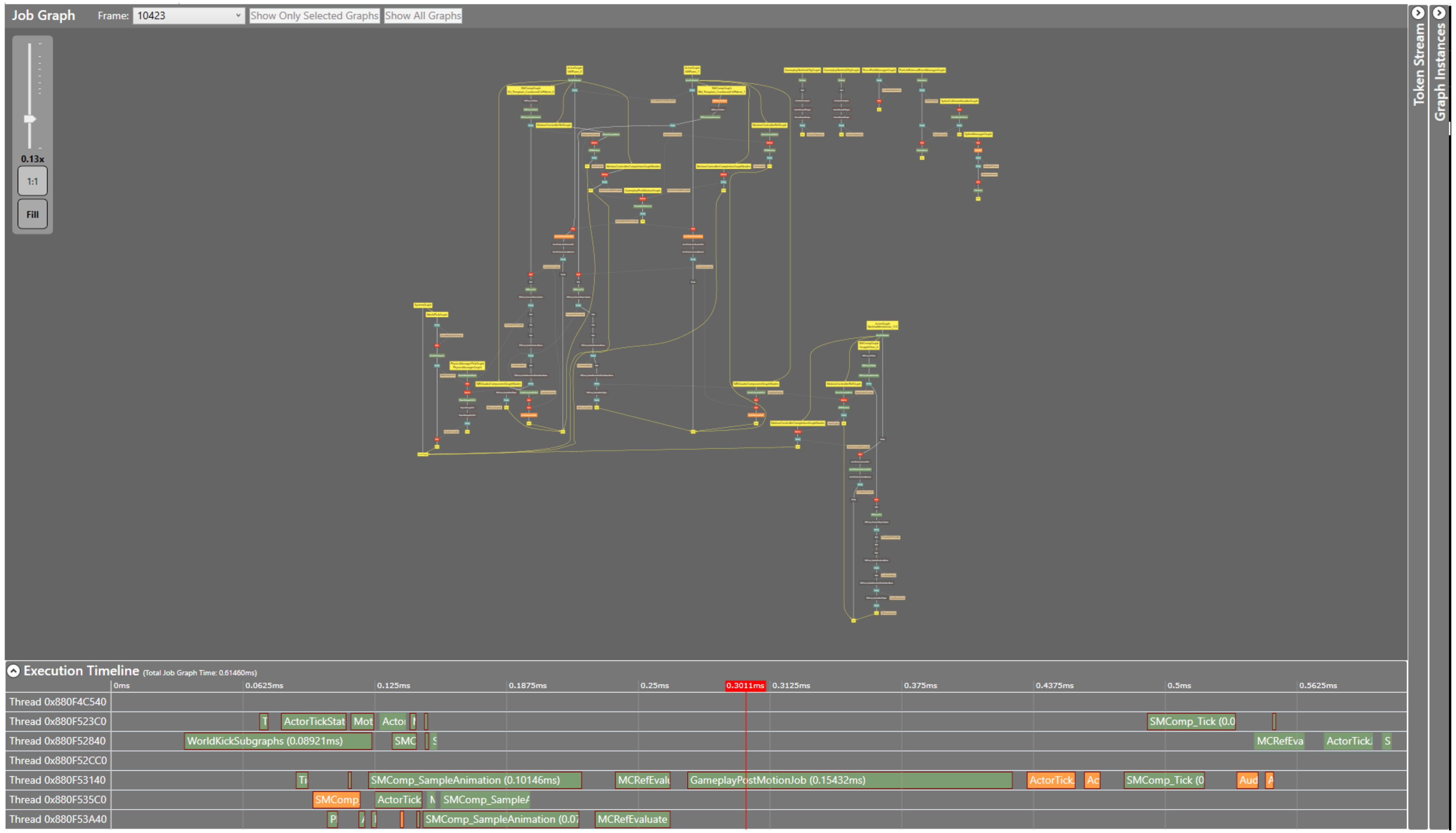


~3ms for a LOT more work!

Job Graph – Full (~2.0ms)



Job Graph – Sim (~0.5ms)

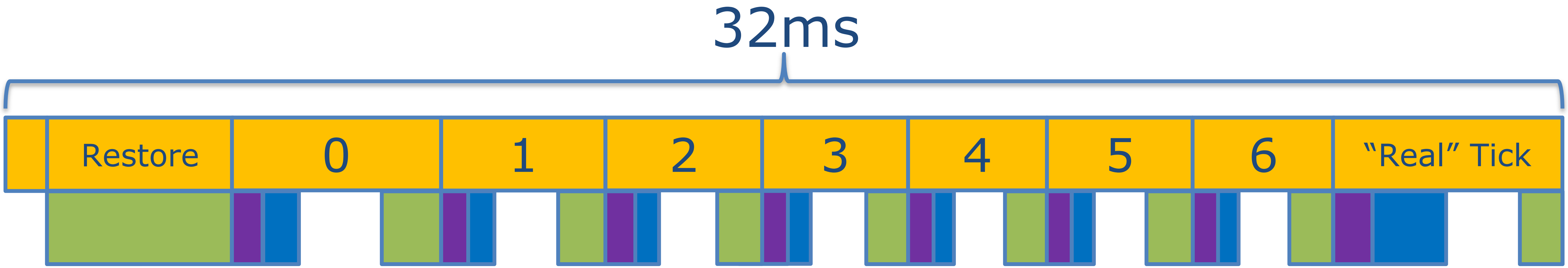


So… about that threading

- Thread contention is real
  - Manage thread priorities and affinities
  - Don’t over-subscribe threads
  - Drop thread priority for low priority or latency tolerant work
  - Careful of priority inversion and starvation!
- Threading primitives can cost more than they are worth
  - Useful migration pattern
  - Use Move semantics to avoid unnecessary atomic operations
    - E.g. Handle copying

Tick Timeline

7 frame rollback, Mortal Kombat X (idle)



🟪 Gameplay

🟦 Engine

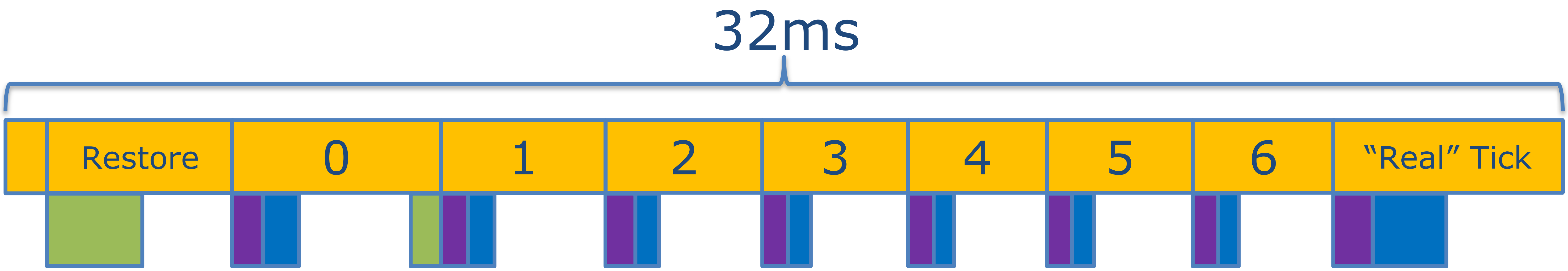
🟩 Save/Restore

Do you have to save 8 times?

- **KEY INSIGHT!** You only need to save the confirmed frame!
  - Large optimization for the worst case
  - Makes average case slower (rollback further)

Tick Timeline

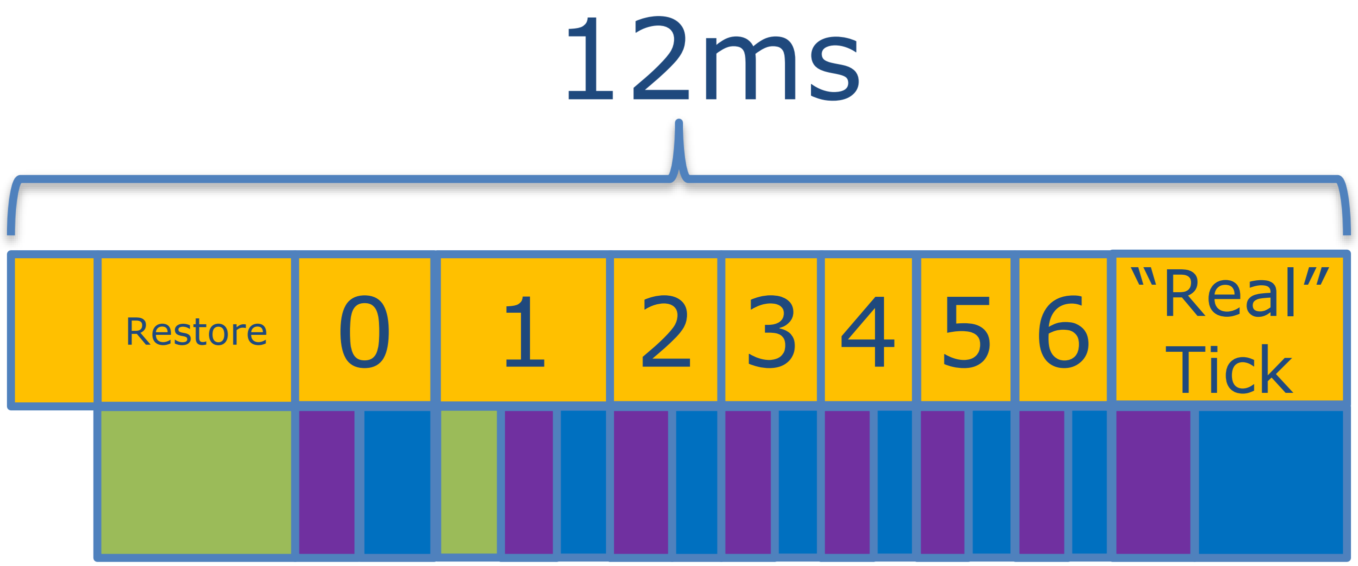
7 frame rollback, Mortal Kombat X
(idle)



🟪 Gameplay
🟦 Engine
🟩 Save/Restore

Tick Timeline

7 frame rollback, Mortal Kombat X (idle)



🟪 Gameplay
🟦 Engine
🟩 Save/Restore

Particle Performance

- Particles were special
  - Naïve approaches WAY too expensive
- Particle systems were the largest cause of performance spikes
- Heavy caching
- Deferred async initialization of particle systems
- Automatic emitter parallelization

Particle Resim Modes

- **RESIM_ALWAYS** – N simulations, 1 serialization
  - Simulate this particle every frame
- **RESIM_NEVER** – 1 simulation, 1 serialization
  - Simulate on the render frame
- **RESIM_PREDICTIVE** – 2 simulations, 1 serialization
  - Simulate on the confirm and the render frame
- **RESIM_NOT_TRACKED** – 1 simulation, 0 serializations
  - Simulate on the render frame

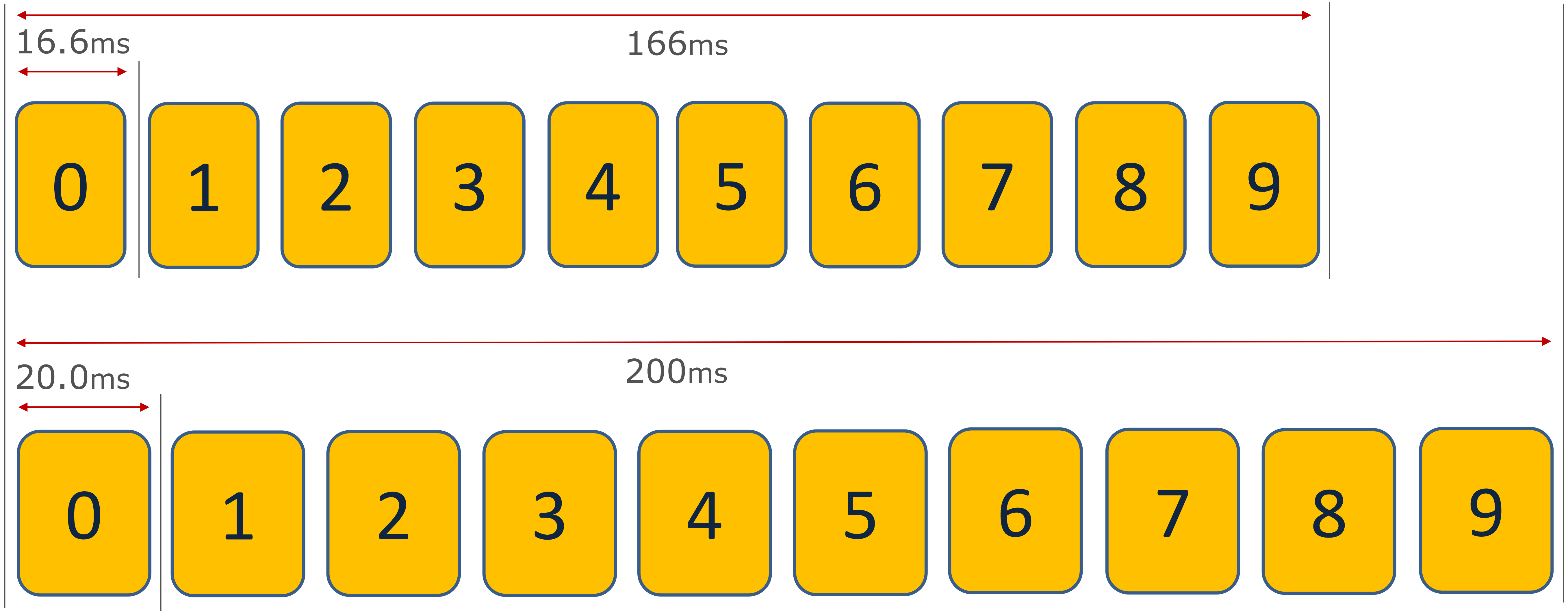
Predictive Particle Cache

- Predictive ticking/serialization
  - May cause visual discontinuities
  - Visual defects mitigated with custom particle state cache
  - Hashed each frame (not just on creation)
  - If particle simulation inputs match cache entry, use cache
- This was EXTREMELY effective
- This is a good template for areas that do not have to be perfect

Checking our work

- QA told us the game was playing GREAT
  - Had been focused on SetRollbackFrames 7
- We were still bogging and net-pausing in our worst cases
  - The net-pauses felt MUCH worse than bogging
- Enter Frame Lengthening!

Frame Lengthening



Beta

- Run a beta!
  - ~20,000 users
- Very positive public response
  - 95% of the players rated it as “as good or better”
- Solidified our performance and network targets

Curveball

- First beta telemetry demonstrated unexpected results
  - Most matches ended up constantly rolling back the maximum
    - Caused by one player getting ahead of the other player
    - Effectively a performance feedback loop
  - Players loved it anyway!
  - Solved by artificially slowing down the player who was ahead
    - Re-used the Frame Lengthening tech

Fine Tuning

- Analyzed our rollback counts
- Used “speculative saves” to reduce rollbacks
- You don’t have to save more than once, but maybe you should…

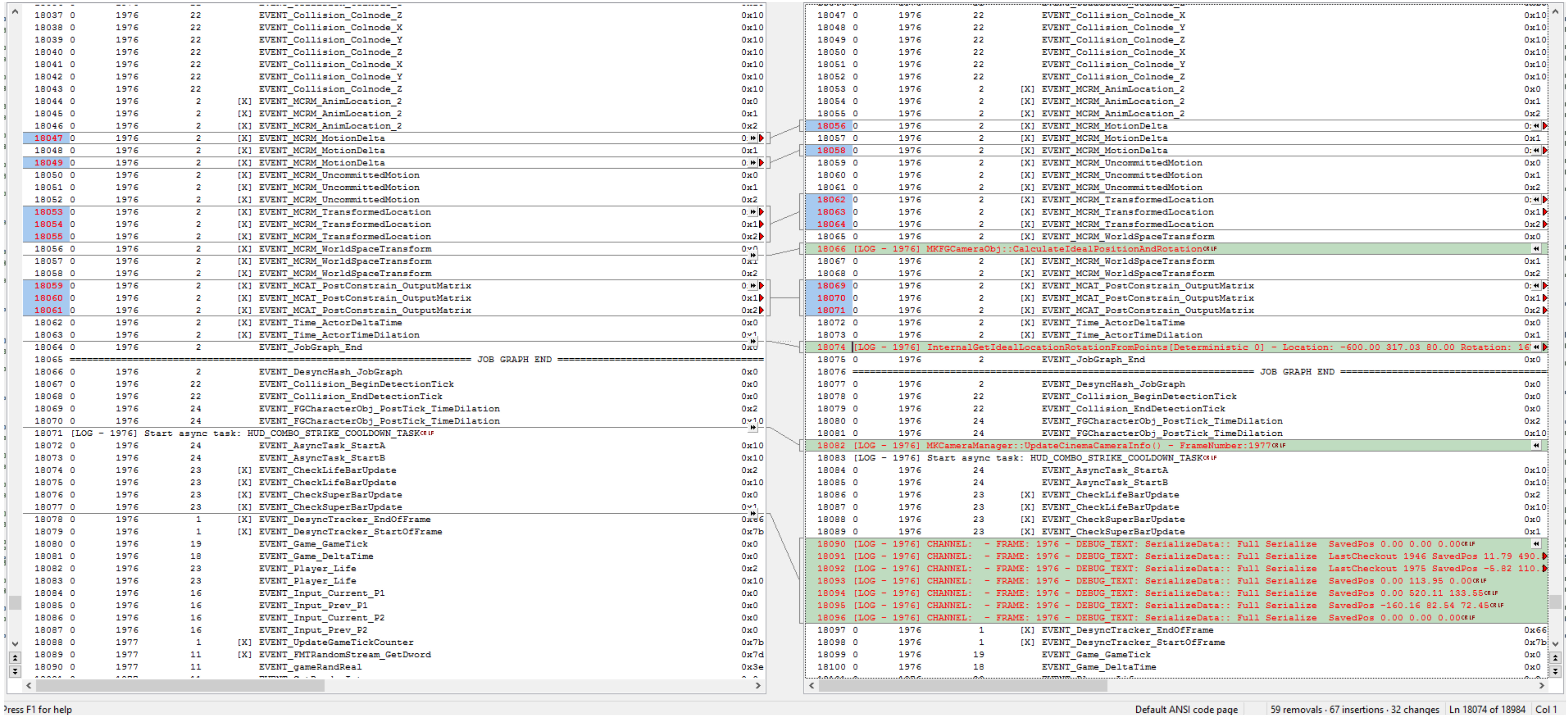
Speculative saves (spec saves)

- Save the confirmed frame (mandatory)
- Save after the simulation mid point (time permitting)
  - Bias this save closer to the confirmed frame
- Save at the end of the frame (time permitting)
- Thresholds are tweakable without patching
- Spec saves reduced total rollback count by 30%

What about all the desyncs?

- Not running procedural systems during simulation caused desyncs
- Luckily, our tools improved to compensate!
- Offline desync detection
- Remote input capture with network delays
  - Allows the match to be replayed
  - Allows breadcrumbs to be added after the fact
  - Invaluable
- Final desync rate less than 0.1%

Desync Log



```
18037 0         1976        22          EVENT_Collision_Colnode_Z                                  0x10
18038 0         1976        22          EVENT_Collision_Colnode_X                                  0x10
18039 0         1976        22          EVENT_Collision_Colnode_Y                                  0x10
18040 0         1976        22          EVENT_Collision_Colnode_Z                                  0x10
18041 0         1976        22          EVENT_Collision_Colnode_X                                  0x10
18042 0         1976        22          EVENT_Collision_Colnode_Y                                  0x10
18043 0         1976        22          EVENT_Collision_Colnode_Z                                  0x10
18044 0         1976         2      [X] EVENT_MCRM_AnimLocation_2                                  0x0
18045 0         1976         2      [X] EVENT_MCRM_AnimLocation_2                                  0x1
18046 0         1976         2      [X] EVENT_MCRM_AnimLocation_2                                  0x2
18047 0         1976         2      [X] EVENT_MCRM_MotionDelta                                     0:
18048 0         1976         2      [X] EVENT_MCRM_MotionDelta                                     0x1
18049 0         1976         2      [X] EVENT_MCRM_MotionDelta                                     0:
18050 0         1976         2      [X] EVENT_MCRM_UncommittedMotion                               0x0
18051 0         1976         2      [X] EVENT_MCRM_UncommittedMotion                               0x1
18052 0         1976         2      [X] EVENT_MCRM_UncommittedMotion                               0x2
18053 0         1976         2      [X] EVENT_MCRM_TransformedLocation                             0:
18054 0         1976         2      [X] EVENT_MCRM_TransformedLocation                             0x1
18055 0         1976         2      [X] EVENT_MCRM_TransformedLocation                             0x2
18056 0         1976         2      [X] EVENT_MCRM_WorldSpaceTransform                             0x0
18057 0         1976         2      [X] EVENT_MCRM_WorldSpaceTransform                             0x1
18058 0         1976         2      [X] EVENT_MCRM_WorldSpaceTransform                             0x2
18059 0         1976         2      [X] EVENT_MCAT_PostConstrain_OutputMatrix                      0:
18060 0         1976         2      [X] EVENT_MCAT_PostConstrain_OutputMatrix                      0x1
18061 0         1976         2      [X] EVENT_MCAT_PostConstrain_OutputMatrix                      0x2
18062 0         1976         2      [X] EVENT_Time_ActorDeltaTime                                  0x0
18063 0         1976         2      [X] EVENT_Time_ActorTimeDilation                               0x1
18064 0         1976         2          EVENT_JobGraph_End                                         0x0
18065 ====================================================================== JOB GRAPH END ==============================
18066 0         1976         2          EVENT_DesyncHash_JobGraph                                  0x0
18067 0         1976        22          EVENT_Collision_BeginDetectionTick                         0x0
18068 0         1976        22          EVENT_Collision_EndDetectionTick                           0x0
18069 0         1976        24          EVENT_FGCharacterObj_PostTick_TimeDilation                 0x2
18070 0         1976        24          EVENT_FGCharacterObj_PostTick_TimeDilation                 0x10
18071 [LOG - 1976] Start async task: HUD_COMBO_STRIKE_COOLDOWN_TASKCRLF
18072 0         1976        24          EVENT_AsyncTask_StartA                                     0x10
18073 0         1976        24          EVENT_AsyncTask_StartB                                     0x10
18074 0         1976        23      [X] EVENT_CheckLifeBarUpdate                                   0x2
18075 0         1976        23      [X] EVENT_CheckLifeBarUpdate                                   0x10
18076 0         1976        23      [X] EVENT_CheckSuperBarUpdate                                  0x0
18077 0         1976        23      [X] EVENT_CheckSuperBarUpdate                                  0x1
18078 0         1976         1      [X] EVENT_DesyncTracker_EndOfFrame                             0x66
18079 0         1976         1      [X] EVENT_DesyncTracker_StartOfFrame                           0x7b
18080 0         1976        19          EVENT_Game_GameTick                                        0x0
18081 0         1976        18          EVENT_Game_DeltaTime                                       0x0
18082 0         1976        23          EVENT_Player_Life                                          0x2
18083 0         1976        23          EVENT_Player_Life                                          0x10
18084 0         1976        16          EVENT_Input_Current_P1                                     0x0
18085 0         1976        16          EVENT_Input_Prev_P1                                        0x0
18086 0         1976        16          EVENT_Input_Current_P2                                     0x0
18087 0         1976        16          EVENT_Input_Prev_P2                                        0x0
18088 0         1977         1      [X] EVENT_UpdateGameTickCounter                                0x7b
18089 0         1977        11      [X] EVENT_FMTRandomStream_GetDword                             0x7d
18090 0         1977        11          EVENT_gameRandReal                                         0x3e
```

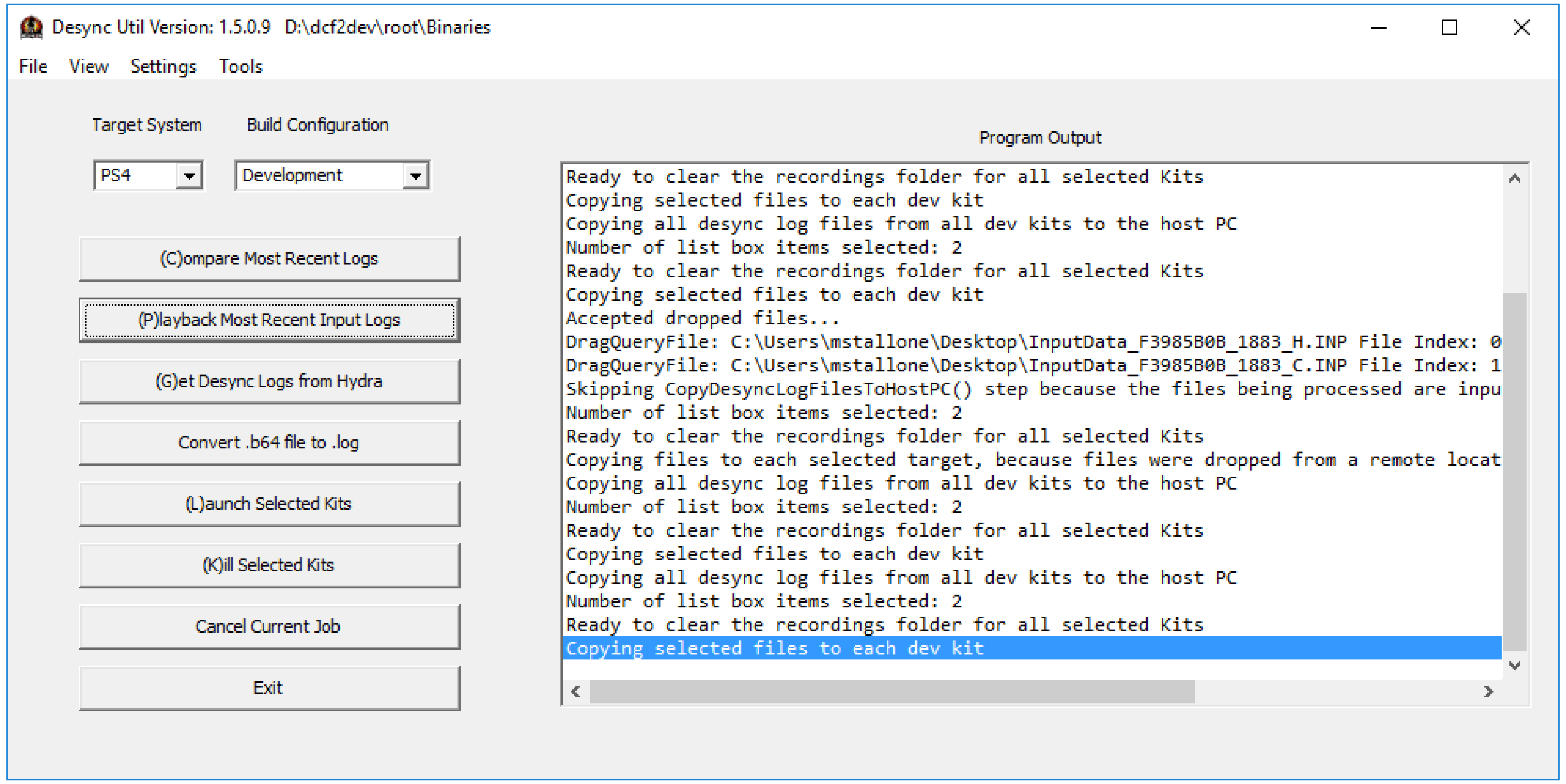
```
18047 0         1976        22          EVENT_Collision_Colnode_X                                  0x10
18048 0         1976        22          EVENT_Collision_Colnode_Y                                  0x10
18049 0         1976        22          EVENT_Collision_Colnode_Z                                  0x10
18050 0         1976        22          EVENT_Collision_Colnode_X                                  0x10
18051 0         1976        22          EVENT_Collision_Colnode_Y                                  0x10
18052 0         1976        22          EVENT_Collision_Colnode_Z                                  0x10
18053 0         1976         2      [X] EVENT_MCRM_AnimLocation_2                                  0x0
18054 0         1976         2      [X] EVENT_MCRM_AnimLocation_2                                  0x1
18055 0         1976         2      [X] EVENT_MCRM_AnimLocation_2                                  0x2
18056 0         1976         2      [X] EVENT_MCRM_MotionDelta                                     0:
18057 0         1976         2      [X] EVENT_MCRM_MotionDelta                                     0x1
18058 0         1976         2      [X] EVENT_MCRM_MotionDelta                                     0:
18059 0         1976         2      [X] EVENT_MCRM_UncommittedMotion                               0x0
18060 0         1976         2      [X] EVENT_MCRM_UncommittedMotion                               0x1
18061 0         1976         2      [X] EVENT_MCRM_UncommittedMotion                               0x2
18062 0         1976         2      [X] EVENT_MCRM_TransformedLocation                             0:
18063 0         1976         2      [X] EVENT_MCRM_TransformedLocation                             0x1
18064 0         1976         2      [X] EVENT_MCRM_TransformedLocation                             0x2
18065 0         1976         2      [X] EVENT_MCRM_WorldSpaceTransform                             0x0
18066 [LOG - 1976] MKFGCameraObj::CalculateIdealPositionAndRotationCRLF
18067 0         1976         2      [X] EVENT_MCRM_WorldSpaceTransform                             0x1
18068 0         1976         2      [X] EVENT_MCRM_WorldSpaceTransform                             0x2
18069 0         1976         2      [X] EVENT_MCAT_PostConstrain_OutputMatrix                      0:
18070 0         1976         2      [X] EVENT_MCAT_PostConstrain_OutputMatrix                      0x1
18071 0         1976         2      [X] EVENT_MCAT_PostConstrain_OutputMatrix                      0x2
18072 0         1976         2      [X] EVENT_Time_ActorDeltaTime                                  0x0
18073 0         1976         2      [X] EVENT_Time_ActorTimeDilation                               0x1
18074 [LOG - 1976] InternalGetIdealLocationRotationFromPoints[Deterministic 0] - Location: -600.00 317.03 80.00 Rotation: 16
18075 0         1976         2          EVENT_JobGraph_End                                         0x0
18076 ====================================================================== JOB GRAPH END ==============================
18077 0         1976         2          EVENT_DesyncHash_JobGraph                                  0x0
18078 0         1976        22          EVENT_Collision_BeginDetectionTick                         0x0
18079 0         1976        22          EVENT_Collision_EndDetectionTick                           0x0
18080 0         1976        24          EVENT_FGCharacterObj_PostTick_TimeDilation                 0x2
18081 0         1976        24          EVENT_FGCharacterObj_PostTick_TimeDilation                 0x10
18082 [LOG - 1976] MKCameraManager::UpdateCinemaCameraInfo() - FrameNumber:1977CRLF
18083 [LOG - 1976] Start async task: HUD_COMBO_STRIKE_COOLDOWN_TASKCRLF
18084 0         1976        24          EVENT_AsyncTask_StartA                                     0x10
18085 0         1976        24          EVENT_AsyncTask_StartB                                     0x10
18086 0         1976        23      [X] EVENT_CheckLifeBarUpdate                                   0x2
18087 0         1976        23      [X] EVENT_CheckLifeBarUpdate                                   0x10
18088 0         1976        23      [X] EVENT_CheckSuperBarUpdate                                  0x0
18089 0         1976        23      [X] EVENT_CheckSuperBarUpdate                                  0x1
18090 [LOG - 1976] CHANNEL:  - FRAME: 1976 - DEBUG_TEXT: SerializeData:: Full Serialize  SavedPos 0.00 0.00 0.00CRLF
18091 [LOG - 1976] CHANNEL:  - FRAME: 1976 - DEBUG_TEXT: SerializeData:: Full Serialize  LastCheckout 1946 SavedPos 11.79 490.
18092 [LOG - 1976] CHANNEL:  - FRAME: 1976 - DEBUG_TEXT: SerializeData:: Full Serialize  LastCheckout 1975 SavedPos -5.82 110.
18093 [LOG - 1976] CHANNEL:  - FRAME: 1976 - DEBUG_TEXT: SerializeData:: Full Serialize  SavedPos 0.00 113.95 0.00CRLF
18094 [LOG - 1976] CHANNEL:  - FRAME: 1976 - DEBUG_TEXT: SerializeData:: Full Serialize  SavedPos 0.00 520.11 133.55CRLF
18095 [LOG - 1976] CHANNEL:  - FRAME: 1976 - DEBUG_TEXT: SerializeData:: Full Serialize  SavedPos -160.16 82.54 72.45CRLF
18096 [LOG - 1976] CHANNEL:  - FRAME: 1976 - DEBUG_TEXT: SerializeData:: Full Serialize  SavedPos 0.00 0.00 0.00CRLF
18097 0         1976         1      [X] EVENT_DesyncTracker_EndOfFrame                             0x66
18098 0         1976         1      [X] EVENT_DesyncTracker_StartOfFrame                           0x7b
18099 0         1976        19          EVENT_Game_GameTick                                        0x0
18100 0         1976        18          EVENT_Game_DeltaTime                                       0x0
```

'ress F1 for help

Default ANSI code page | 59 removals · 67 insertions · 32 changes | Ln 18074 of 18984 | Col 1

Desync tools

- General desync detection and logging
- Replay files
- DesyncUtil
- NRSSoak


Desync Util Version: 1.5.0.9 D:\dcf2dev\root\Binaries
File
View
Settings
Tools
(C)ompare Most Recent Logs
(P)layback Most Recent Input Logs
(G)et Desync Logs from Hydra
Convert .b64 file to .log
(L)aunch Selected Kits
(K)ill Selected Kits
Cancel Current Job
Exit

Target System: PS4

Build Configuration: Development

Program Output

```
Ready to clear the recordings folder for all selected Kits
Copying selected files to each dev kit
Copying all desync log files from all dev kits to the host PC
Number of list box items selected: 2
Ready to clear the recordings folder for all selected Kits
Copying selected files to each dev kit
Accepted dropped files...
DragQueryFile: C:\Users\mstallone\Desktop\InputData_F3985B0B_1883_H.INP File Index: 0
DragQueryFile: C:\Users\mstallone\Desktop\InputData_F3985B0B_1883_C.INP File Index: 1
Skipping CopyDesyncLogFilesToHostPC() step because the files being processed are inpu
Number of list box items selected: 2
Ready to clear the recordings folder for all selected Kits
Copying files to each selected target, because files were dropped from a remote locat
Copying all desync log files from all dev kits to the host PC
Number of list box items selected: 2
Ready to clear the recordings folder for all selected Kits
Copying selected files to each dev kit
Copying all desync log files from all dev kits to the host PC
Number of list box items selected: 2
Ready to clear the recordings folder for all selected Kits
Copying selected files to each dev kit
```

Low-Level Lessons Learned

- Limit mutable state
- Prefer handles over pointers where performance allows
- Avoid shared ownership of mutable resources
- Avoid work in constructors/destructors
- Lean on memcopies/buffer swaps instead of dynamic fixup

High-Level Lessons Learned

- Design game systems to drive visual state, *not* depend on it
- Design systems to update with variable time steps
    - Parametrically is even better
- Everyone should work with debug rollback systems enabled
- Defer processing until after the rollback window if reasonable
- Bog is no longer a function of a single frame

Future work

- Multithread gameplay script
- Extend state based serialization
- Simplify particle serialization/simulation (parametric?)
- Game state separation from the visual state
- Add rollback support for more systems

Questions?

NETHERREALM STUDIOS is hiring

Mike Stallone
mstallone@netherrealm.com