IntervalZero

ProductBrief

April 2021

# RTX64

## Overview

RTX64 is a key component of the IntervalZero RTOS Platform that also includes x86 and x64 multi-core multiprocessors, and the Windows operating system to outperform real-time hardware such as DSPs and MCUs and reduce the development costs for systems that require determinism or hard real-time.

Symmetric multiprocessing-enabled RTX64 takes full advantage of 64-bit memory and performance capabilities. Uniquely, the RTX64 RTOS scheduler enables embedded real-time applications to directly access the 512GB of addressable physical memory available on 64-bit Windows.

This is critical to modern day real-time systems and represents a gigantic leap from the 4GB physical memory limit of traditional 32-bit Windows systems. The 4GB barrier has stymied innovation in many industries that depend on real-time systems and that require memory access far beyond 4GB.

## Determinism

- Guaranteed Precision – set timer periods down to 1 microsecond, and Interrupt Service Thread (IST) latencies of less than 10 microseconds
- Separation from Windows – Windows processes cannot interfere with real-time applications
- Scalability – one scheduler is used across all real-time processors. Symmetric multiprocessing (SMP) aware scheduler utilizes both priority-driven and pre-emptive algorithms to ensure critical thread context switches; and yields to threads of high priority occur in the sub-microsecond range

## Control

- Flexibility to configure how much or little processing capability is used for real-time processes (1 to 63 processors)
- Full control of real-time process threads with the ability to load balance as needed. RTX64 provides the ability to set thread and interrupt affinities
- Peace of mind if Windows issues a STOP message or shutdown; real-time applications have the ability to continue running to safety shutdown

## Simplify

- Use a single operating system for applications.  RTX64 is supported on Windows 10
- Use commercial off-the shelf (COTS) target system; no special hardware required
- Use one development environment - Visual studio 2015, 2017, and 2019
- Use common languages (C/C++) for Windows and real-time applications
- Use common Win32 API; same code can be run as a Windows or real-time process
- Use managed code for Windows application and still communicate with your real-time applications
- No driver model to follow; real-time process can talk directly to hardware
- Use standard IPC communication between Windows applications and real-time processes (events, mutexes, and semaphores)
- Use shared memory between Windows and real-time process for sharing of data

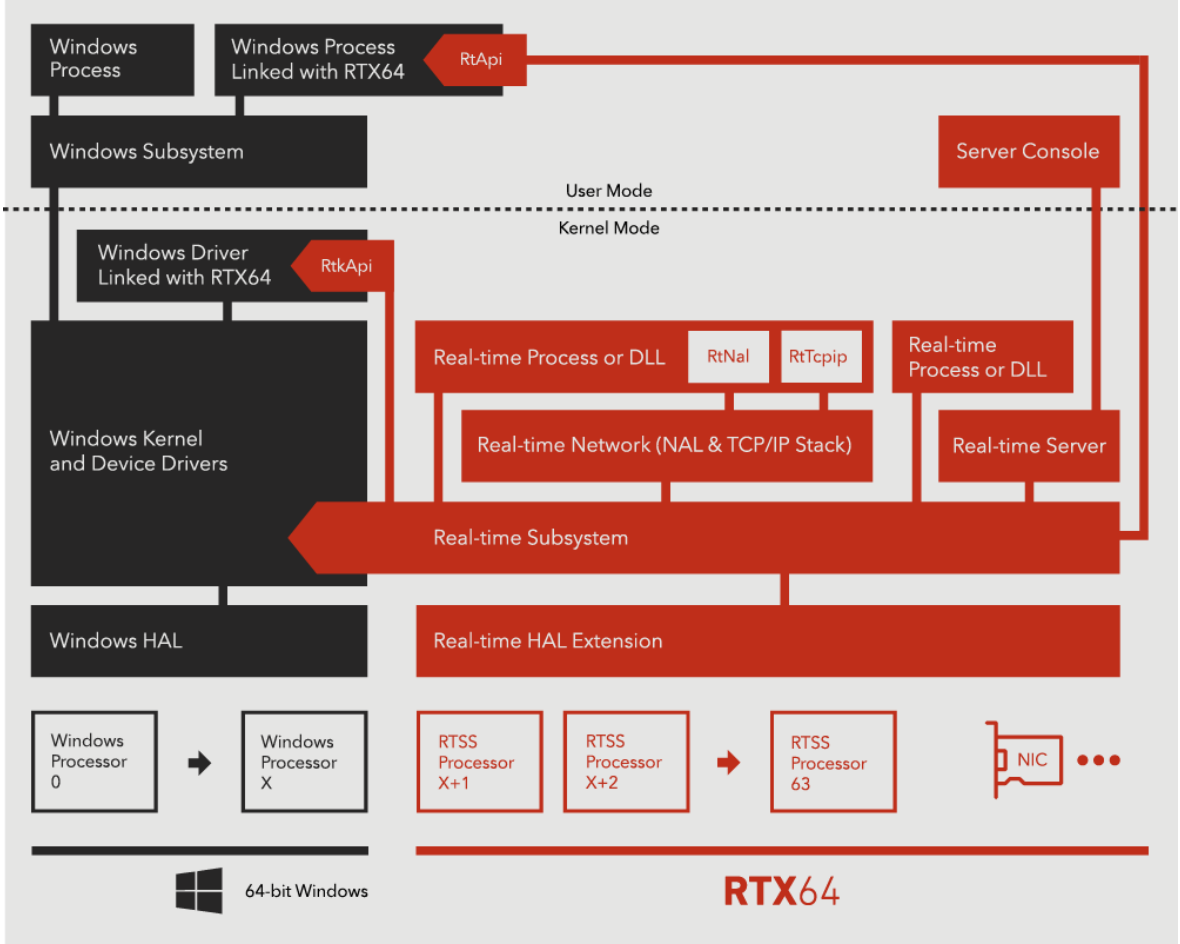
## Reduce Costs

- Eliminate additional system to perform the HMI
- Eliminate proprietary controller and communications cards
- Improved asset utilization: Take advantage of underused multi-core capacity
- Reduced manufacturing costs and fewer physical parts

## Improve Efficiency

- Eliminate some inventory costs and reduce maintenance costs
- Field upgrades are accomplished through software download rather than board replacement

## Architecture




## Key Features

### Real-time Runtime

- Scalable from 1 to 63 real-time processors
- SMP aware scheduler utilizes both priority-driven and pre-emptive algorithms to ensure critical thread context switches; and yields to threads of high priority occur in the sub-microsecond range
- Configurable thread and interrupt affinity
- Configurable timer period

- Ability to attach to line-based and message-based (MSI/MSI-X) interrupts
- Shutdown handling on Windows STOP or shutdown
- Deterministic memory
- Access to Windows file system and registry
- Ability to set Search paths for process creation and loading of RTDLLs
- Dynamic-link library support through RTDLLs, which can be loaded implicitly or explicitly
- Ability to profile application behavior by monitoring internal objects and custom events
- Real-time Inter Process Communication between Windows user processes and real-time processes
    - Native and managed interface for 32-bit or 64-bit Windows processes
    - Objects available: events, mutexes, and semaphore
    - Data sharing through shared memory
- Real-time Inter Process Communication between Windows kernel drivers and real-time processes
    - Native interface for 64-bit Windows drivers
    - Objects available: events, mutexes, and semaphore
    - Data sharing through shared memory
- RTX64 Network Abstraction Layer (NAL) provides the following networking capabilities to the RTX64 Subsystem:
    - Per-frame callbacks for high performance, low latency transmit and receive at Layer 2
    - Ability to transmit Ethernet frames close to line-speed
    - Provides burst transmission of Ethernet frames allowing line speed with small packets
    - Supports zero-copy when transmitting multiple frames
    - Supports IEEE-1588 timestamping
    - Network drivers for a number of common Network Interface Cards
    - Enables the use of NIC hardware queues
    - Supports polling of devices
- Windows user groups for limiting access to RTX64 features
- Tools and Utilities
    - Activation and Configuration – activate subsystem components and configure RTSS cores
    - Control Panel – configuration the subsystem
    - Console – display print messages
    - SRTM – view system timer to timer handler response on a given core
    - KSRTM – view system timer to interrupt service routine (ISR) response
    - Latency View – view and compare system timer response latencies on multiple cores at the same time
    - Task Manager – display a list of running RTSS processes and Windows processes and drivers linked to RTX64, also display pre-process, -thread, and –processor CPU usage
    - Monitor – configures profiling of RTSS
    - RTX64Objects – view internal objects and states
    - RTX64MSpaces – view internal memory allocations

**Software Development Kit**

- Headers and libraries for application development
    - Real-time API (RTAPI) similar to Windows Win32 API
    - Real-time Kernel API (RTKAPI)

    - Real-time Network API (RTNAPI)
    - Real-time Network Driver API (RTNDAPI)
    - Managed Code Framework (IntervalZero.RTX64) – Subsystem setup and configuration through managed code interface
    - Native framework – Subsystem setup and configuration through C/C++ interface
- Microsoft Visual Studio 2015, 2017, and 2019 support
    - Wizard for application and dll development
    - API Code snippets
    - C-Runtime support
    - Local and remote debugger support via launch within Visual Studio
    - Local and remote attach support
- Microsoft WinDbg extension and RTSS symbols
- Percepio Tracealyzer for RTX64 – graphical tool to analyze monitoring data
- Sample source to show basic concepts

**Product Documentation**

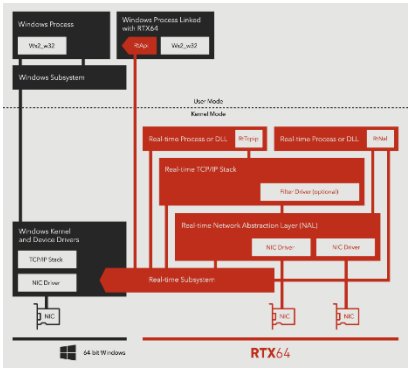
- Documentation consisting of installation and user guides, API references, and details on real-time programming concepts

## Additional Purchasable Features

**RT-TCP/IP Stack**

The RT-TCP/IP Stack provides the following networking capabilities to the RTX64 Subsystem:

- TCP/UDP/IP networking for RTX64 processes
- Support for IPv4 and IPv6
- Winsock support
- RAW Sockets
- MAC layer filtering
- Virtual Network – point to point connection between Windows and RTSS
- Utilities (RtssArp, RtssIpConfig, RtssPing, and RtssRoute)




**RTX64 Vision**

RTX64 Vision provides functionality for using GigE Vision Cameras within the real-time RTX64 environment:

- Real-time GigE Vision filter driver
- Camera Setup Tool
- Real-time GigE Vision Interface & Communication library
- Built version of OpenCV for use with RTSS Vision applications