MANDO Runtime

USER GUIDE

IntervalZero

MANDO Runtime User Guide

IZ-DOC-0296-R1

IntervalZero

200 Fifth Avenue, FL 6, STE 6020
Waltham, MA 02451
Phone: 781-996-4481
www.intervalzero.com

Contents

Overview

In this guide, we introduce key MANDO Runtime features and show you how to configure and control Runtime components. The **MANDO Runtime** includes a Real-time HAL (RTHAL) and a Real-time Kernel (RTKernel) that are necessary to run MANDO executables: Real-time applications (.RTSS) or Real-time DLLs (.RTDLL).

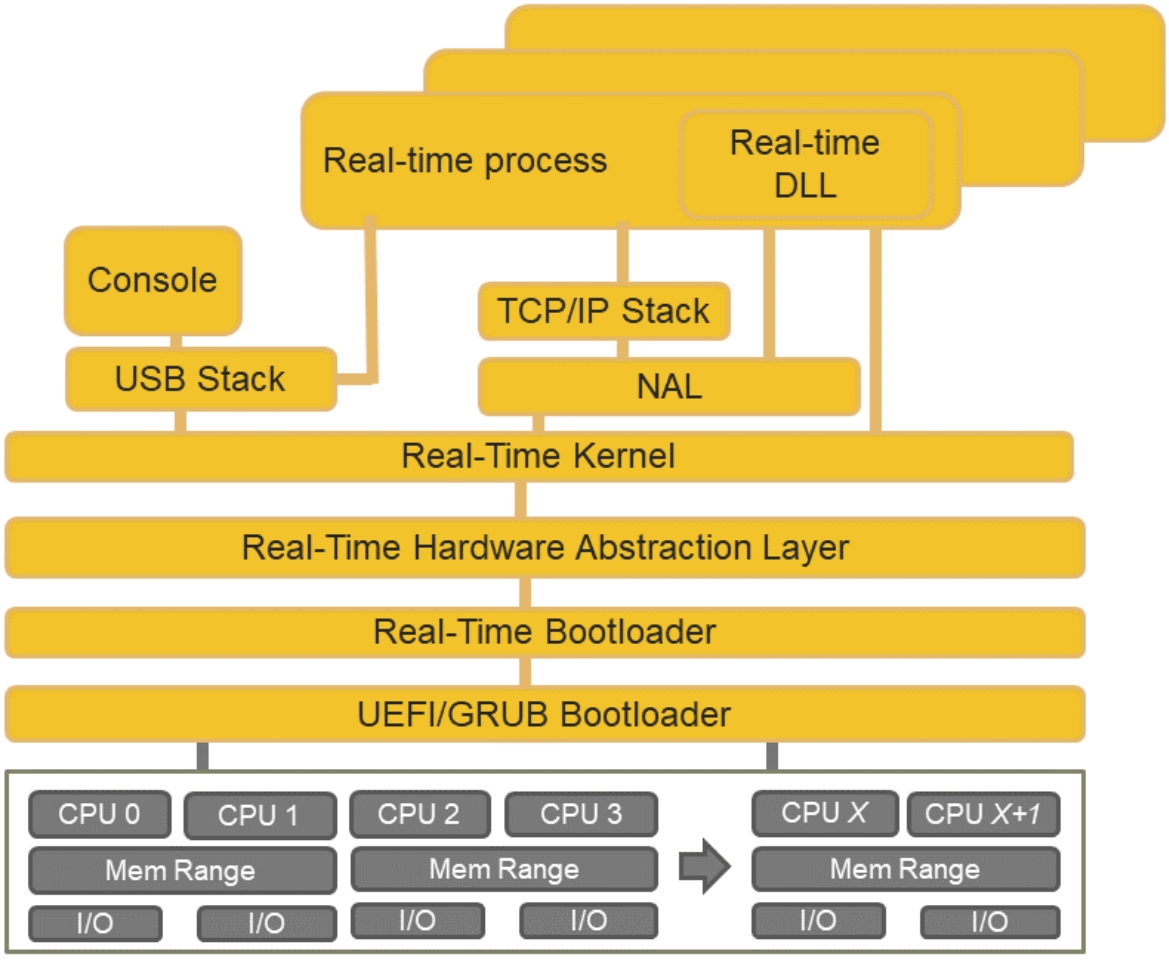
NOTE: This guide assumes you have installed the MANDO Runtime. For more information on installation, see the *MANDO Runtime Install Guide*.

1

MANDO Architecture

MANDO components are loaded in the following order:

1. The system boots up through UEFI/GRUB Bootloader, displaying *GNU GRUB version 2.04* on the screen.
2. The Real-time Bootloader starts to provide IntervalZero MANDO boot options.
3. The Real-time Bootloader loads the Real-time Hardware Abstraction Layer (RTHAL).
4. The RTHAL initializes and loads the Real-time Kernel (RTKernel).
5. RTKernel initializes and loads the USB stack, and optionally starts the Network Abstraction Layer (NAL) and TCP/IP stack as configured in the RtKrnlConfig.ini file.
6. The RTKernel then loads any Real-time processes (RTProcesses) as configured to start up at boot time in RtKrnlConfig.ini and RtJigInput.bat.

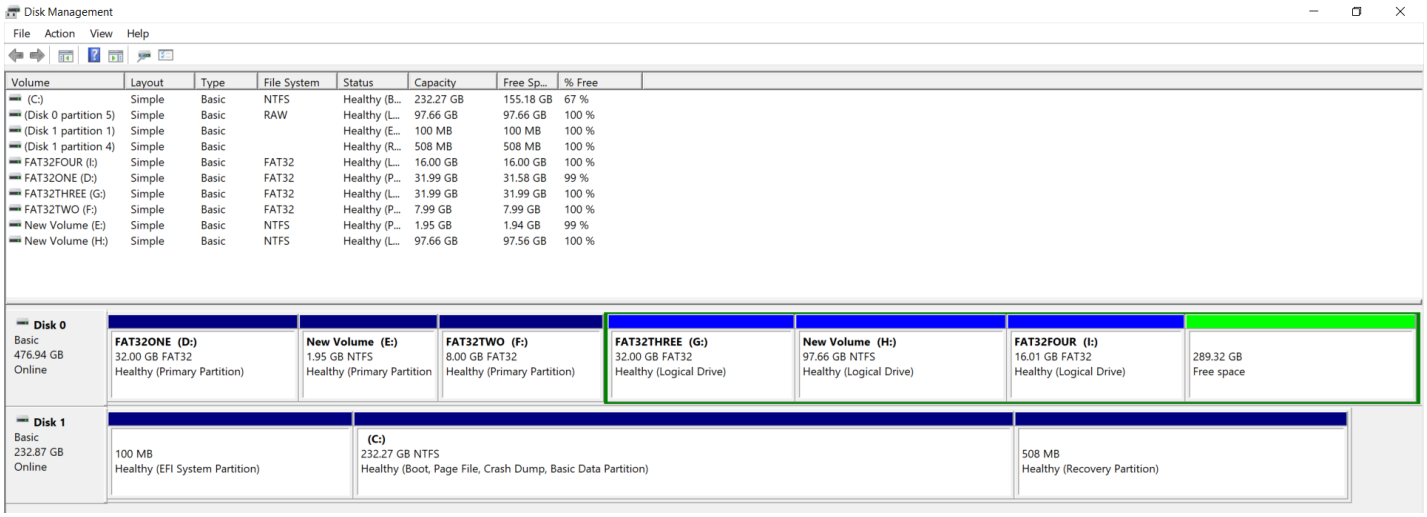


Component	Description
UEFI / GRUB Bootloader	Widely used operating system bootloader, referring to *Multiboot2 Specification version 2.0*.
Real-time Bootloader	Reads Real-time HAL (RTHAL) configuration parameters, then loads RTHAL.

Component	Description
Real-time Hardware Abstraction Layer	• Processor initialization and management • Address space management • I/O device and bus management • High resolution clock and timer • Other
Real-time Kernel	• Thread, process, object, handle management • MSpace memory management • RTProcess/RTDLL loader • File system • Registry database • Other
Real-time Network	• Network Interface Drivers • Network Abstraction Layer (NAL) • TCP/IP protocol stack

Drive Letters in MANDO

In MANDO, drive letters are allocated differently than in Windows. This can result in some confusion when setting paths to binaries in the `RtJigInput.bat` script, or to files which should be manipulated in an application. MANDO allocates drive letters in the order that volumes are found on the machine. For example, take the following disk setup as seen through the Windows Disk Management utility:



In this case, there are four FAT32 Volumes. MANDO will allocate the drive letters as follows:

FAT32 Volume	**MANDO Drive Letters**
FAT32ONE	C:
FAT32TWO	D:
FAT32THREE	E:
FAT32FOUR	F:

NOTE: In this example, adding FAT32 volumes to the second disk will add a G: volume in MANDO. Disk 0 will be found before Disk 1. As a result, volumes on Disk 0 will be found before volumes on Disk 1.

NOTE: Drive letters in Windows have no bearing on the drive letters in MANDO.

2

Configuring MANDO Runtime

There are two MANDO Runtime components that can be configured:

- You can configure the **Real-time HAL (RTHAL)** using the `grub.cfg` file, which can be found and edited at `<GRUB Drive>\boot\grub\` on the GRUB installed drive. See Configuring the Real-Time HAL for a full list of all Real-time HAL configuration settings.

  When the system boots into the GNU GRUB version 2.04-2.06 configuration selection screen, select the appropriate IntervalZero MANDO entry. Once selected, you can optionally edit the command of the entry by pressing the **E** key. The command contains multiple options separated by white space.

- You can configure the **Real-time Kernel (RTKernel)** using the `RtKrnlConfig.ini` file, which is read during Real-time Kernel initialization. You can access and edit this file at `<InstallDrive>\Program Files\MANDO\`. See Configuring the Real-Time Kernel for a full list of all Real-time Kernel configuration settings.

SECTIONS IN THIS CHAPTER:

MANDO Boot Configurations

The table below lists the different MANDO boot configurations from GRUB entries.

Boot Configuration	Description
Standard Mode	Default configuration with 100 us HAL timer period.
Fastest Mode	Fastest configuration with 20 us HAL timer period.
Tick Compensation	Adjust timer ticks based on the Time Stamp Counter (TSC).
SMI Mitigation	Disable Advanced Configuration and Power Interface (ACPI) power management.
Priority-Based CAT/MBA	Enable cache allocation and memory bus throttling based on thread priority.
Debug Mode	Using serial port to display and send shell commands.

NOTE: When the system boots into the Configuration Selection screen, select the appropriate IntervalZero MANDO entry. Once selected, you can edit the entry's command by pressing the **E** key. The command contains multiple options separated by white space.

Configuring the Real-Time HAL

You can configure the Real-time HAL (RTHAL) through the `grub.cfg` file, which you can access and edit at `<GRUB Drive>\boot\grub\` on the GRUB installed drive. The table below details a full list of all Real-time HAL configurable parameters.

Setting	Default Value	Description
NumProc	0x40	Sets the maximum number of processors/cores (x) to run. This value must be an integer in the range 1 to 63.
TickPeriod	0x64	Sets the HAL Timer Period to x micro-seconds. **NOTE:** Use caution when setting the HAL timer period below 20 microseconds. If the HAL timer period is set too low it can inversely impact performance or cause your system to hang.
TickCompd	0	Select **1** to enable adjusting timer ticks based on CPU time stamp counter (TSC). Select **0** without adjustment.
AcpiPwMgr	0	Select **1** to enable SMI mitigation by disabling ACPI power management. Select **0** without disabling ACPI power management.
MultiIntLvl	1	Select **0** to use single interrupt level. Select **1** to use multiple interrupt levels. Select **2** to use two interrupt levels.
DisableRdt	0	Select **1** to ignore Intel RDT capability. Select **0** to use Intel RDT capability.

Setting	Default Value	Description
PriorityCat	0	Select **1** to enable cache allocation based on thread's priority. Select **0** for flat cache allocation.
PriorityMba	0	Select **1** to enable memory bus throttling based on thread's priority. Select **0** without memory bus throttling.
RemoveMemory	0	Removes memory (x MBs) from the total available memory that RTHAL/RTKernel can use.
TruncateMemory	0xFFFFFFFFFFFFFFFF	Limits the amount of physical memory available to RTHAL/RTKernel. When you use this option, RTHAL/RTKernel ignores all memory at or above the specified physical address. Specify the address in bytes.
uart	port@0x3F8	Select **port@** for serial port debugging. Select **bdf@** for PCI serial device debugging. Select **mmio@** for MMIO device debugging.

Configuring the Real-Time Kernel

You can configure the Real-time Kernel (RTKernel) using the `RtKrnlConfig.ini` configuration file available from `<InstallDrive>\Program Files\MANDO\`. The available configuration settings are listed below.

Setting	Default Value	Type	Description
StarvationBehavior	00000000	DWORD	Sets the behavior after WatchDog time-out of a running thread. • 0 = Disabled • 1 = Freeze all active processes
StarvationTimeout	004c4b40	DWORD	Sets the starvation timeout value (in microseconds). • 0 = No time-out
SystemExtMSpacePoolCommit	00000000	DWORD	Determines whether to allocate the system process external MSpace at RTKernel start up. • 0 = No • 1 = Yes
SystemExtMSpacePoolMinThreshold	00100000	DWORD	Sets the minimum size (in bytes) of the system process external MSpace.

Setting	Default Value	Type	Description
SystemIntMSpacePoolMinThreshold	00400000	DWORD	Sets the minimum size (in bytes) of the system process internal MSpace.
DisableSystemMSpacePoolExpand	00000000	DWORD	Determines whether the system process MSpaces expand when depleted. • 0 = Expandable • 1 = Not expandable
SystemMSpacePoolExpandSize	00100000	DWORD	Sets the minimum expand size (in bytes) for the system process MSpaces at each expansion.
DisableSystemMSpacePoolShrink	00000000	DWORD	Determines whether the system process MSpaces are auto shrink when memory frees. • 0 = Auto shrink • 1 = Not auto shrink
IntMSpacePoolMinThreshold	00100000	DWORD	Sets the minimum size (in bytes) of the user process internal MSpace.

Setting	Default Value	Type	Description
ExtMSpacePoolMinThreshold	00100000	DWORD	Sets the minimum size (in bytes) of the user process external MSpace.
DisableMSpacePoolExpand	00000000	DWORD	Determines whether the user process MSpaces expand when depleted. • 0 = Expandable • 1 = Not expandable
MSpacePoolExpandSize	00100000	DWORD	Sets the minimum expand size (in bytes) for user process MSpaces at each expansion.
DisableMSpacePoolShrink	00000000	DWORD	Determines whether user process MSpaces auto shrink when memory frees. • 0 = Auto shrink • 1 = Not auto shrink
ZeroMemoryAtAllocation	00000001	DWORD	Determines whether to initialize memory to zero at allocation. • 0 = memory is not initialized • 1 = memory is initialized to zero

Setting	Default Value	Type	Description
MSpaceFootPrintLimit	0000000000000000 **NOTE:** This is a 64-bit value	QWORD	Sets a limit (in bytes) for the total amount of memory that can be allocated from an MSpace. • 0 = No limit
PoolCacheTrimThreshold	00200000	DWORD	Sets the maximum amount (in bytes) of unused top-most memory to keep in Pool Cache. • ffffffff = Poll Cache trim is disabled
PoolCacheGranularity	00010000	DWORD	Sets the granularity size (in bytes) of allocations from the Pool Cache to Local Pool.
PoolCacheMmapThreshold	ffffffff	DWORD	Sets the minimum size (in bytes) allowed for allocations directly from Local Pool. • ffffffff = Disable direct allocation

Setting	Default Value	Type	Description
FreezeProcessOnMSpaceExhaust	00000000	DWORD	Determines whether to freeze the process when its non-expandable MSpace is exhausted. • 0 = Don't freeze • 1 = Freeze
DefaultTimeQuantum	0000000000000000 **NOTE:** This is a 64-bit value	DWORD	Sets the default thread time quantum value (in micro-seconds).
HardwareExceptionHandling	00000000	DWORD	Determines how to handle hardware exceptions. • 0 = Structure Exception Handling • 1 = Freeze the process, and send an error message to the console

Setting	Default Value	Type	Description
EnterDebuggerOnException	00000000	DWORD	Determines the action following an unhandled exception. • 0 = Freeze the process, and send an error message to the console • 1 = Enter debugger
FreeStackOnTerminateThread	00000000	DWORD	Determines whether to free the thread stack after a thread is terminated. • 0 = Save the thread stack • 1 = Free the thread stack

Setting	Default Value	Type	Description
StopAtFirstException	00000000	DWORD	Determines whether to stop at first exception to avoid extra exception during exception handling. **NOTE:** This is a bitmap of exception indexes. • 0 = Don't stop at first exception • 1 = Stop at first exception
BehaviourStopAtFirstException	00000000	DWORD	Determines the behavior when stopping at first exception. • 0 = Freeze process and send an error message to the console • 1 = Let debugger handle exception

Setting	Default Value	Type	Description
DebugBreakAtFirstException	00000000	DWORD	Sets debug break when stopping at first exception. • 0 = Don't set debug break • 1 = Set debug break
PriorityInversion	00000001	DWORD	Enables or disables Mutex priority inversion. • 0 = Disable Mutex priority inversion • 1 = Enable Mutex priority inversion
SearchPath	"C:\Program Files\IntervalZero\MANDO\bin\"	String	Sets the search path for loading RTSS process and RTDLL files. **NOTE:** The maximum length allowed for the search path is 266 characters. Quotation marks inside strings are not allowed.

Setting	Default Value	Type	Description
LogOutputOnFile	00000001	DWORD	Determines whether RtPrintf outputs to a log file (`RtLogFile.txt`). • 0 = Disable RtPrintf logging • 1 = Enable RtPrintf logging
NALAutoStart	00000000	DWORD	Determines whether the NAL should start automatically.
TCPIPAutoStart	00000000	DWORD	Determines whether the TCPIP Stack should start automatically. • 0 = Don't start the TCPIP Stack • 1 = Start the TCPIP Stack
USBAutoStart	00000001	DWORD	Determines whether the USB Stack should start automatically. • 0 = Don't start the USB Stack • 1 = Start the USB Stack

Setting	Default Value	Type	Description
EnableRtssJig	00000001	DWORD	Determines whether to run the test jig at RTKernel startup. • 0 = Don't run test jig • 1 = Run test jig

Configuring MANDO Components

MANDO uses a configuration file called `RtConfig.rtreg`, accessible from `<InstallDrive>\Program Files\MANDO\`, to configure its components and store custom user values for applications through the Registry API. The file is written in human readable plain text, which makes it easy to read and modify. You can edit the file to modify component settings as needed.

MANDO has the following components:

- Network Abstraction Layer (NAL)
- TCP/IP Stack
- USB Stack

This `RtConfig.rtreg` file is read during startup of each component. If you modify the `RtConfig.rtreg` file after a component is started, the changes will not take place until you restart MANDO.

NOTE: MANDO only recognizes one `RtConfig.rtreg` file and it has a specific syntax.

Syntax

```
[SectionPath1]
"DataItemName1"=DataType1:DataValue1
"DataItemName2"=DataType1:DataValue2
[SectionPath2]
"DataItemName3"=DataType1:DataValue3
```

Parameters

SectionPathx

The unique path that separates each hierarchy. For example: ***HKEY_LOCAL_MACHINE\SOFTWARE\INTERVALZERO\MANDO***. The SearchPath must be surrounded by brackets ([ ]). `RtConfig.rtreg` can contain several unique paths. Each path is case sensitive and can contain one or many data items.

The following SearchPaths locations should not be modified since they are reserved for MANDO components:

- HKEY_LOCAL_MACHINE\SOFTWARE\INTERVALZERO\MANDO – general MANDO settings
- HKEY_LOCAL_MACHINE\SOFTWARE\INTERVALZERO\MANDO\RtNAL – general NAL settings
- Any NAL interfaces should be of the format HKEY_LOCAL_MACHINE\SOFTWARE\INTERVALZERO\MANDO\RtNAL\rtndXX where XX is the order in which NAL will load the interface. XX starts with 00 and goes up by 1 (01, 02,…). Be careful not to skip numbers.
- HKEY_LOCAL_MACHINE\SOFTWARE\INTERVALZERO\MANDO\RTTCPIP
- HKEY_LOCAL_MACHINE\SOFTWARE\INTERVALZERO\MANDO\RTTCPIP\TCPIP00 – general TCP/IP Stack settings
- HKEY_LOCAL_MACHINE\SOFTWARE\INTERVALZERO\MANDO\RtUSB – general USB settings

Below are examples of possible unique paths where a user could store custom data:

- HKEY_LOCAL_MACHINE\MyData\MyApplication
- HKEY_USERS\MyData\MyApplication

DataItemNamex

The name of a unique data item. An equal sign (=) immediately follows the name of the data item. For example, if you wanted to represent the data item **MyDword** with a value of **99** you would write **"MyDword"=dword:99**. If you wanted to represent a string for a data item **MyString** that contains a string value of **This is a string!**, you would write **"MyString"=" This is a string!"**. No data type is needed for strings. If you are using a '\' you must use "\\". The data item name is case-sensitive.

NOTE: You can enter several **DataItemNamex** lines for the same section path.

DataTypex

A keyword that specifies the type of data value assigned to a **DataItemNamex** data item. A string is the default data type, so it does not need to be provided. For all other data types, the type must follow the equal sign (=) without spaces. A colon (:) must immediately follow the data type.

Typical data types:

Data Type	In RtConfig.rtreg
dword	Data represented by a number that is 4 bytes long (a 32-bit integer).
hex(b)	Data represented by a number that is a 64-bit integer.
hex(7)	A multiple string, values that contain lists. Entries are separated by spaces, commas, or other marks.
hex(2)	A variable-length data string.

DataValuex

Immediately follows the colon (or the equal sign with string value) and must be in the appropriate format (for example, string or decimal).

Configuring the Real-Time Network and Interfaces

MANDO provides several options for configuring the real-time network (Network Abstraction Layer and RT-TCP/IP Stack) and for managing network interfaces through the RtConfig.rtreg file.

RELATED TOPICS:

- Network Interface Cards
- Configuring and Controlling the Network
- Managing Network Interfaces
- Configuring Jumbo Frames
- Tips, Tricks and Configurations Hints

Network Interface Cards

The Network Interface Cards (NICs) supported by the Real-Time Network do not require any special hardware configuration. Just install the card in a vacant PCIe slot of your computer.

Each NIC card that is an MANDO device requires its own interrupt, be it line-based, MSI, or MSI-X.

To view a list of supported network cards, go to the IntervalZero Customer Center or contact IntervalZero Customer Support.

Configuring and Controlling the Network

The MANDO RtConfig.rtreg file provides several options for configuring network behavior and performance. This file and the RtKrnlConfig.ini file are available from `\Program Files\MANDO`.

NOTE: The TCP/IP Stack depends on the NAL. Stopping the NAL will also stop the Stack.

Topics:

Configuring and Controlling the Network Abstraction Layer (NAL)

Controlling NAL Startup Behavior

The status of the **NALAutoStart** parameter in the RtKrnlConfig.ini file determines whether the NAL starts automatically when the kernel starts.

TO START THE NAL WITH THE KERNEL:

Set **NALAutoStart** to 1 to start the NAL automatically when the kernel starts. The default value is 0 (do not start the NAL with the kernel).

TO START THE NAL INDEPENDENT OF THE KERNEL:

Set **NALAutoStart** to 0 to start the NAL independent from the kernel. This is the default behavior.

NAL Configuration Settings

The RtConfig.rtreg configuration file contains several options for configuring the Network Abstraction Layer (NAL). You can find these under this Section Path:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\INTERVALZERO\MANDO\RtNAL]
```

Setting	Recommended value	Type	Description
IdealProcessor	0	DWORD	The processor number on which the NAL will run. **NOTE:** Processor numbers are zero based. By default, the first RTSS processor is the ideal processor.

Setting	Recommended value	Type	Description
ExtMSpacePoolMinimumSize	1048576	DWORD	The minimum size of the NAL process's external MSpace in bytes. The minimum is 65536 bytes. The default value is 1048576 bytes. This size will differ based on the number of enabled interfaces. You can use the following formula to determine the amount of memory you need to allocate: Minimum size for external MSpace = 128K + (64K * numberOfEnabledInterfaces)
MSpacePoolExpandable	1	DWORD	Controls whether the NAL process will request additional memory when its MSpace is exhausted. • 1 (Expand) • 0 (Do not expand)
MSpacePoolExpandSize	1048576	DWORD	The size of memory requested if the NAL expands its MSpace.
Verbose	1	DWORD	Allows the NAL to be run in verbose mode. • 1 (Enable verbose logging) • 0 (Disable verbose logging)
IoctlDispatchPriority	40	DWORD	The priority of the NAL's Ioctl Dispatch Thread. This value must be within the range 0 to 127.
IoctlDispatchIdealProcessor	0	DWORD	The ideal processor of the NAL's Ioctl Dispatch Thread.

Configuring and Controlling the TCP/IP Stack

NOTE: These settings require a valid RT-TCP/IP Stack license.

Controlling TCP/IP Stack Startup Behavior

The status of the **TCPIPAutoStart** parameter in the RtKrnlConfig.ini file determines whether the TCP/IP Stack starts automatically when the NAL starts.

TO START THE TCP/IP STACK WITH THE NAL:

Set the **TCPIPAutoStart** setting to 1 to start the TCP/IP Stack with the NAL automatically.

TCP/IP Stack Configuration Settings

The RtConfig.rtreg configuration file contains several options for configuring the TCP/IP Stack. You can find these settings under this Section Path:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\INTERVALZERO\MANDO\RTTCPIP]
```

Setting	Recommended value	Type	Description
Memory	4096	DWORD	The amount of memory, in kilobytes, allocated to the TCP/IP stack, including the heap but not including memory allocated by device drivers at startup.

Setting	Recommended value	Type	Description
MaxConcurrency	0	DWORD	The number of threads that are allowed to run concurrently within the TCP/IP Stack. The range is 0 to 10340. The TCP/IP Stack needs to initialize certain attributes when started to allow a certain number of threads to run and provide services for each client that requests services. For example, running a client and a server application will require 1 concurrency each. The TCP/IP Stack running by itself requires 1 concurrency to run the Loopback service. Loading and managing interfaces requires an average of 3 threads, thus a concurrency of 3 for each interface. **NOTE:** We recommend that this value be calculated automatically (0, default). **NOTE:** If the TCP/IP Stack requires more threads than the initialization process prepared for, the Stack will crash.

You can find these settings under this Section Path:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\INTERVALZERO\MANDO\RTTCPIP\TCPIP00]
```

Setting	Recommended value	Type	Description
IdealProcessor	0	DWORD	The processor number on which the TCP/IP Stack will run. **NOTE:** Processor numbers are zero based. By default, the first RTSS processor is the ideal processor.
MaxSockets	64	DWORD	The maximum number of sockets. The TCP/IP Stack allocates actual socket memory when it creates a socket, so it must know the maximum number of sockets it must create. The specified value must be in the range of 1 to 2551024. The default value is 64. **NOTE:** In the running system, the socket range is 0 to the maximum number of sockets. For example, if the maximum number of sockets is set to 64, the range is 0 to 63.

Setting	Recommended value	Type	Description
TickInterval	100	DWORD	The Stack timer tick interval. The Stack timer is an internal timer used for all internal synchronization. The RT-TCP/IP Stack requires a fixed time notification for every Stack timer interval to update its elapsed time counters. Since there are several protocols implemented within the Stack, dealing with individual timers would be difficult. Therefore, the RT-TCP/IP Stack is optimized to use a single notification for how much time has elapsed. The Stack timer system manages all of the different timers used within the RT-TCP/IP Stack. The specified value must be within the range of 1 to 1000 milliseconds.
TimerIdealProcessor	0	DWORD	The ideal processor for the TCP/IP Stack Timer. **NOTE:** Processor numbers are zero based. By default, the first RTSS processor is the ideal processor.
TimerPriority	66	DWORD	The priority of the TCP/IP Stack's first real-time timer thread, which updates timer variables. This value must be within the range 1-127, where 1 is the lowest priority and 127 is the highest priority.

Setting	Recommended value	Type	Description
IPReassemblyTimeout	60	DWORD	The time-out interval on IP reassembly. **NOTE:** We recommend that you decrease the IP reassembly time-out value so that it is less than the wrap-around time in an IP ID field.
ExtMSpacePoolMinimumSize	6422528	DWORD	The minimum size of the TCP/IP process's external MSpace, in bytes. The minimum is 65536 bytes. This value needs to be large enough to support RT-TCP/IP Stack heap allocation and must be at least the RT-TCP/IP Stack heap allocation size plus 2176 kilobytes. You can use the following formula to determine the amount of memory you need to allocate: *Minimum size for external MSpace = StackHeap(k) + 2176k + (64k * numberOfEnabledInterfaces)*
MSpacePoolExpandable	1	DWORD	Controls whether the TCP/IP process will request additional memory when its MSpace is exhausted. • 1 (Expand) • 0 (Do not expand)

Setting	Recommended value	Type	Description
MSpacePoolExpandSize	1048576	DWORD	Sets the size of memory requested if the TCP/IP Stack expands its MSpace.
MaxArpEntries		DWORD	Sets the maximum number of ARP entries allowed by the TCP/IP Stack. Each ARP cache entry is 100 bytes. It is recommended that the maximum ARP cache entries supported be greater than the total number of devices with which the interface communicates. **NOTE:** If the value is too small, the ARP cache can overflow. The potential for an overflow increases when the majority of network devices are offline. **NOTE:** When an overflow occurs, the TCP/IP Stack presents the warning message *tfRtClone: ARP cache full*, which indicates that the maximum number of entries supported should be increased.

Setting	Recommended value	Type	Description
TimerExecutePriority	60	DWORD	The priority of the RT-TCP/IP Stack's second real-time timer thread, which executes functions for expired timers. This value must be in the range 1-127, where 1 is the lowest priority and 127 is the highest priority. This value must be less than or equal to the value set for Stack timer priority.

Managing Network Interfaces

You can add, delete, set properties for, and associate filters with MANDO network interfaces using the `RtConfig.rtreg` configuration file accessible from `<InstallDrive>\Program Files\MANDO\.`

NOTE: Some changes require a restart of the network.

TOPICS IN THIS SECTION:

Adding, Modifying, and Deleting Interfaces

Before you can manage interfaces, you must first add the MANDO network interface to the RtConfig.rtreg configuration file. See *Configuring MANDO Components* for `RtConfig.rtreg` syntax.

TO ADD AN INTERFACE:

1. Open the RtConfig.rtreg configuration file and add the unique SectionPath surrounded by brackets [].

   For example:

   ```
   [HKEY_LOCAL_MACHINE\SOFTWARE\INTERVALZERO\MANDO\RtNAL\rtndXX]
   ```

   where XX is the order in which the NAL will load the interface. XX starts with 00 and goes up by 1 (01, 02,...) Be careful not to skip numbers.

2. Specify a **FriendlyName** for the interface. The name cannot exceed 64 characters and cannot include spaces.

For example:

```
"FriendlyName"="RtNalIPCH"
```

3. Select the proper MANDO **Driver** for the supported NIC card.

   For example:

   ```
   "Driver"="RtNalIPCH.rtdll"
   ```

   See the *MANDO Supported NICs* document for a list of supported drivers.

5. Specify the PCI bus **Location** of the network interface card for the interface in the form of three semicolon-separated integers. This parameter is optional if only one device of its kind is installed. The default location is 0;0;0. You can run the PciScanBus.rtss sample binary to get location information. See the MANDO SDK Help for information on this sample.
6. If you want the new interface to support TCP/IP functionality, you must specify the following settings:
   - "StackDriver"="RtnMiniport.rtdll"
   - The IPv4 Address ipaddr of the interface in dotted-quad notation. If you specify an IPv4 Address that matches that of another enabled interface, a dialog appears with a list of the duplicate IPv4 address(es). For example:

     "ipaddr"="192.168.100.50"

     - The IPv4 netmask of the interface in dotted-quad notation. For example:

       "netmask"="255.255.255.0"

     - The **IPv4 Address** of the interface in dotted-quad notation. If you specify an IPv4 Address that matches that of another enabled interface, a dialog appears with a list of the duplicate IPv4 address (es).
     - The IPv4 **Netmask** of the interface in dotted-quad notation.

RTCONFIG.RTREG EXAMPLE

In this example configuration, the NAL and TCP/IP Stack support one network interface card which uses the RtNalIPCH driver:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\INTERVALZERO\MANDO\RtNAL\rtnd00]
"Enable"=dword:00000001
"Location"="0;31;6"
"Driver"="RtNalIPCH.rtdll"
"FriendlyName"="RtNalIPCH"
"AdditionalIPAddresses"=hex(7):00,00
"AdditionalNetmasks"=hex(7):00,00
"gateway"="0.0.0.0"
"InterruptIdealProcessor"=dword:00000000
"InterruptPriority"=dword:00000064
"InterruptType"=dword:00000002
"ipv6addr"="Auto"
"ipv6prefix"=dword:00000064
"LinkStatus"=dword:00000001
"LinkStatusIdealProcessor"=dword:00000000
"LinkStatusPriority"=dword:00000000
"MTU"=dword:00001500
"netmask"="255.255.255.0"
"NumRecvBuffers"=dword:00000256
"NumXmitBuffers"=dword:00000256
"ReceiveIdealProcessor"=dword:00000000
"ReceivePriority"=dword:00000063
"ipaddr"="192.168.100.50"
"DefaultQueue"=dword:00000000
"MaxNumberOfRecvQueues"=dword:00000001
"NumberOfRecvQueues"=dword:00000001
```

```
"MaxNumberOfXmitQueues"=dword:00000001
"NumberOfXmitQueues"=dword:00000001
"MaxPacketSize"=dword:00001514
"XmitCompletePriority"=dword:00000062
"XmitCompleteIdealProcessor"=dword:00000000
"InterfaceType"=dword:00000003
"StackDriver"="RtnMiniport.rtdll"
```

TO CHANGE INTERFACE PROPERTIES:

To change an interface property, follow the steps below.

1. Open the RtConfig.rtreg configuration file and find the interface whose properties you want to view or edit under `[HKEY_LOCAL_MACHINE\SOFTWARE\INTERVALZERO\MANDO\RtNAL\rtndXX] SectionPathx`.
2. Edit the properties you want to change. See Network Abstraction Layer (NAL) Properties and TCP/IP Stack Properties below.
3. You must restart the machine running MANDO.

TO DELETE AN INTERFACE:

1. Open the RtConfig.rtreg configuration file and find the interface you want to delete under `[HKEY_LOCAL_MACHINE\SOFTWARE\INTERVALZERO\MANDO\RtNAL\rtndXX] SectionPathx`.
2. Delete the interface.

Network Abstraction Layer (NAL) Properties

To change NAL-specific interface properties for a specific interface, select the interface you want to edit from the RtConfig.rtreg configuration file. You can typically find these settings under the [HKEY_LOCAL_MACHINE\SOFTWARE\INTERVALZERO\MANDO\RtNal\rtndxx] SectionPathx.

NOTE: If the selected interface supports TCP/IP, its TCP/IP-specific settings appear under TCP/IP properties (see below).

Setting	Recommended value	Type	Description
Enable	1	DWORD	Enable/disable the interface.
Location	0;0;0	String	**Optional.** The PCI bus location of the network interface card for the interface in the form of three semicolon-separated integers. **NOTE:** You can run the PciScanBus.rtss sample binary to get location information. See the MANDO SDK Help for more information.
FriendlyName	RtNalPCH	String	The friendly name of the Real-time network device (Rtnd) driver for this interface.
Driver	RtNalPCH.rtdll	String	The MANDO driver for the supported NIC card.
MaxPacketSize	1514	DWORD	The maximum packet size allowed by the interface. See Device-Specific Interface Values for device-specific defaults and value ranges.

Setting	Recommended value	Type	Description
LinkStatus	1	DWORD	Enable/disable link status monitoring for the selected interface.
XmitCompleteIdealProcessor	0	DWORD	The ideal processor for the thread that services transmitted packets.
XmitCompletePriority	62	DWORD	The priority of the thread that services transmitted packets.
ReceiveIdealProcessor	0	DWORD	The ideal processor for the thread that services received packets.
ReceivePriority	63	DWORD	The priority of the thread that services received packets.

Setting	Recommended value	Type	Description
InterruptType	2	DWORD	The type of interrupt that will be used: • 2 (MSI-X Single Vector) • 4 (MSI-X Multi Vector) • 3 (MSI) • 1 (Line-based) **NOTE:** For improved performance, we recommend you use MSI-X Single Vector instead of MSI for interrupts when possible.
InterruptIdealProcessor	0	DWORD	The ideal processor for the thread servicing the interface interrupts. This value must be a valid MANDO processor.
InterruptPriority	64	DWORD	The interrupt priority for the interface. This value must be a valid MANDO priority within the range 1-127.
NumXmitBuffers	256	DWORD	The number of transmit buffers used by the interface. This value must be greater than or equal to 64, and it must be a multiple of 8.

Setting	Recommended value	Type	Description
NumRecvBuffers	256	DWORD	The number of receive buffers used by the interface. This value must be greater than or equal to 64, and it must be a multiple of 8.
DefaultQueue	0	DWORD	The number of the queue that receives and transmits unfiltered data, starting with queue 0. See Device-Specific Interface Values for device-specific defaults and value ranges.
NumberOfXmitQueues	1	DWORD	The number of transmit queues to configure for the interface.
NumberOfRecvQueues	1	DWORD	The number of receive queues to configure for the interface.
Forced Duplex	0	DWORD	The duplex method used to establish the Ethernet link: • 0 (Autonegotiate) • 1 (Half Duplex) • 2 (Full Duplex)

Setting	Recommended value	Type	Description
ForcedSpeed	0	DWORD	The speed used to establish the Ethernet link: • 0 (Autonegotiate) • 1 (10 Mbps) • 2 (100 Mbps)
StackDriver	RtnMiniport.rtdll	String	Adds TCP/IP support to the selected interface. See *TCP/IP Stack Properties* below for more information.

TCP/IP Stack Properties

NOTE: The TCP/IP options will not be available if the TCP/IP component is not activated.

Setting	Recommended value	Type	Description
ipaddr	192.168.100.50	String	The IPv4 address of the interface in dotted-quad notation.
netmask	255.255.255.0	String	The IPv4 network mask of the interface in dotted-quad notation.

Setting	Recommended value	Type	Description
gateway	0.0.0.0	String	**Optional**. The gateway for this interface. If this parameter is not specified, the interface has no gateway. **NOTE:** This value must be a valid IP address. Leading zeroes are not supported.
mtu	1500	DWORD	**Optional**. The Maximum Transmission Unit (MTU) size in bytes for the interface. **NOTE:** This value cannot exceed the interface's Max packet size minus 14. See Device-Specific Interface Values for device-specific defaults and value ranges. For detailed information on configuring Jumbo Frames, see *Configuring Frames* in the MANDO SDK Help.

Setting	Recommended value	Type	Description
ipv6addr	Auto	String	**Optional**. The IPv6 address. If not given, the interface has no IPv6 address.
ipv6prefix	64	DWORD	**Optional**. The IPv6 network prefix.

Filtering Frames

The MANDO Ethernet Filter is an extensible interface between the RT-TCP/IP stack and a Real-time network interface. You can use this interface to filter all frames at the data link layer and send Ethernet frames directly to the RTND driver. See *Using an Ethernet Filter* in the MANDO SDK Help for more information.

Device-Specific Interface Values

This topic lists the device-specific interface defaults and value ranges for all MANDO supported devices. This information is useful when adding or modifying interfaces in the `RtConfig.rtreg` configuration file accessible from `<InstallDrive>\Program Files\MANDO\`.

NOTE: Some devices may have more restrictive limits. To see the most accurate limits, consult the appropriate specification guide for your device.

PROPERTIES:

Maximum packet size

Setting	Description	Recommended value	Type	NAL limits
MaxPacketSize	The maximum packet size allowed by the interface.	1514	DWORD	1514-16380

Driver	Devices	Driver limits	Device limits	Notes
RtNalIGC	All	1514-9212	1514-9212	
RtNalIGB	All	1514-9716	1514-9716	
RtNalIPCH	All	1514-9014	1514-9212	
RtNalI10GB	All	1514-15868	1514-15868	
RtNalE1000	0x1000, 0x104B, 0x104D, 0x104C, 0x108B, 0x108C, 0x109A	1514-1514	1514-1514	
RtNalE1000	Others not listed above	1514-9014	1514-9212	All 82571, 82572, 82574, 82583 devices: 1514-9212. All 80003ESLAN devices: 1514-9212.

Driver	Devices	Driver limits	Device limits	Notes
RtNalRtl	All	1514-9212	1514-9212	Maximum packet size for specific hardware revisions:

Revision	Driver Limits	Device Limits	Tested
0x8110	1514	1514	No
0x8168C, 0x8111C, 0x8168CP, 0x8111CP	6140	6140	No
0x8168C-SPIN2, 0x8111C-SPIN2	6140	6140	Yes
0x8168B, 0x8110S, 0x8169	7436	7436	No
0x8168D, 0x8168DP, 0x8168G, 0x8168EP	9212	9212	No
0x8168EVL, 0x8111EVL, 0x8168H, 0x8111H, 0x8168E, 0x8111E, 0x8168F, 0x8111F	9212	9212	Yes

Number of receive buffers

Setting	Description	Recommended value	Type	NAL limits
NumRecvBuffers	The number of receive buffers used by the interface. Must be greater than or equal to 64.	256	DWORD	8-16384

Driver	Devices	Driver limits	Device limits	Notes
RtNalIGC	All	80-16384	8-16384	Multiple of 8, rounded up.
RtNalIGB	All	80-16384	8-16384	Multiple of 8, rounded up.
RtNalIPCH	All	80-16384	8-16384	Multiple of 8, rounded up.
RtNalI10GB	All	80-16384	8-16384	Multiple of 8, rounded up.
RtNalE1000	All	1514-1514	8-16384	Multiple of 8, rounded up.
RtNalRtl	All	64-1024	8-1024	Multiple of 8, rounded up.

Number of transmit buffers

Setting	Description	Recommended value	Type	NAL limits
NumXmitBuffers	The number of transmit buffers used by the interface. Must be greater than or equal to 64.	256	DWORD	8-16384

Driver	Devices	Driver limits	Device limits	Notes
RtNalIGC	All	80-16384	8-16384	Multiple of 8, rounded up.
RtNalIGB	All	80-16384	8-16384	Multiple of 8, rounded up.
RtNalIPCH	All	80-16384	8-16384	Multiple of 8, rounded up.
RtNalI10GB	All	80-16384	8-16384	Multiple of 8, rounded up.
RtNalE1000	All	1514-1514	8-16384	Multiple of 8, rounded up.
RtNalRtl	All	64-1024	8-1024	Multiple of 8, rounded up.

Default receive queue

Setting	Description	Recommended value	Type	NAL limits
DefaultQueue	The number of the queue that receives unfiltered data, starting with queue 0.	0	DWORD	0 (Number of receive queues -1)

Driver	Devices	Driver limits	Device limits
RtNalIGC	All	0 (Number of receive queues -1)	0 (Number of receive queues -1)
RtNalIGB	All	0 (Number of receive queues -1)	0 (Number of receive queues -1)
RtNalIPCH	All	0	0

Driver	Devices	Driver limits	Device limits
RtNalI10GB	All	0	0
RtNalE1000	All	0	0
RtNalRtl	All	0	0

Number of receive queues

Setting	Description	Default	Type	NAL limits
MaxNumberOfRecvQueues	The number of receive queues to configure for the interface.	1	DWORD	1-128

Driver	Devices	Driver limits	Device limits
RtNalIGC	All	1-1	1-4
RtNalIGB	0x10C9, 0x10E6, 0x10E7, 0x10A7, 0x1526	1-16	1-16
RtNalIGB	0x1521,0x150E, 0x150F, 0x1510, 0x1511, 0x1516	1-8	1-8
RtNalIGB	0x1533, 0x1534, 0x1535, 0x157B	1-4	1-4
RtNalIGB	0x1539	1-2	1-2
RtNalIPCH	All	1-1	1-2
RtNalI10GB	All	1-16	1-128
RtNalE1000	All	1-1	1-2
RtNalRtl	All	1-1	1-1

Number of transmit queues

Setting	Description	Recommended value	Type	NAL limits
MaxNumberOfXmitQueues	The number of transmit queues to configure for the interface.	1	DWORD	1-128

Driver	Devices	Driver limits	Device limits
RtNalIGC	All	1-1	1-4
RtNalIGB	0x10C9, 0x10E6, 0x10E7, 0x10A7, 0x1526	1-16	1-16
RtNalIGB	0x1521,0x150E, 0x150F, 0x1510, 0x1511, 0x1516	1-8	1-8
RtNalIGB	0x1533, 0x1534, 0x1535, 0x157B	1-4	1-4
RtNalIGB	0x1539	1-2	1-2
RtNalIPCH	All	1-2	1-2
RtNalI10GB	All	1-16	1-128
RtNalE1000	All	1-1	1-2
RtNalRtl	All	1-1	1-1

Link configuration

Setting	Description	Recommended value	Type	NAL limits
LinkStatus	The method used to establish the Ethernet link.	1 (Auto Negotiate)	DWORD	Auto Negotiate, 10 Mbps Half Duplex , 10 Mbps Full Duplex, 100 Mbps Half Duplex, 100 Mbps Full Duplex

Driver	Devices	Driver limits	Device limits
RtNalIGC	All	Auto Negotiate, 10 Mbps Half Duplex , 10 Mbps Full Duplex, 100 Mbps Half Duplex, 100 Mbps Full Duplex	Auto Negotiate, 10 Mbps Half Duplex , 10 Mbps Full Duplex, 100 Mbps Half Duplex, 100 Mbps Full Duplex
RtNalIGB	All	Auto Negotiate, 10 Mbps Half Duplex , 10 Mbps Full Duplex, 100 Mbps Half Duplex, 100 Mbps Full Duplex	Auto Negotiate, 10 Mbps Half Duplex , 10 Mbps Full Duplex, 100 Mbps Half Duplex, 100 Mbps Full Duplex

Driver	Devices	Driver limits	Device limits
RtNalIPCH	All	Auto Negotiate, 10 Mbps Half Duplex , 10 Mbps Full Duplex, 100 Mbps Half Duplex, 100 Mbps Full Duplex	Auto Negotiate, 10 Mbps Half Duplex , 10 Mbps Full Duplex, 100 Mbps Half Duplex, 100 Mbps Full Duplex
RtNalI10GB	All	Auto-Negotiate	Auto Negotiate, 10 Mbps Half Duplex , 10 Mbps Full Duplex, 100 Mbps Half Duplex, 100 Mbps Full Duplex
RtNalE1000	All	Auto-Negotiate	Auto Negotiate, 10 Mbps Half Duplex , 10 Mbps Full Duplex, 100 Mbps Half Duplex, 100 Mbps Full Duplex
RtNalRtl	All	Auto-Negotiate	Auto Negotiate, 10 Mbps Half Duplex , 10 Mbps Full Duplex, 100 Mbps Half Duplex, 100 Mbps Full Duplex

Interrupt type

Setting	Description	Recommended value	Type	NAL limits
InterruptType	The type of interrupt used by an MANDO real-time network interface.	2 (MSI-X Single Vector when available, otherwise MSI)	DWORD	Line-Based, MSI, MSI-X Single Vector, MSI-X Multi Vector

Driver	Devices	Driver limits	Device limits
RtNalIGC	All	Line-Based, MSI, MSI-X Single Vector, MSI-X Multi Vector	Line-Based, MSI, MSI-X Single Vector, MSI-X Multi Vector
RtNalIGB	0x1520, 0x1521, 0x1522, 0x1523, 0x1524, 0x152F	Line-Based, MSI, MSI-X Single Vector, MSI-X Multi Vector	Line-Based, MSI, MSI-X Single Vector, MSI-X Multi Vector
RtNalIGB	All others	Line-Based, MSI, MSI-X Single Vector	Line-Based, MSI, MSI-X Single Vector, MSI-X Multi Vector
RtNalIPCH	All	Line-Based, MSI, MSI-X Single Vector	Line-Based, MSI, MSI-X Single Vector, MSI-X Multi Vector

Driver	Devices	Driver limits	Device limits
RtNalI10GB	All	Line-Based, MSI, MSI-X Single Vector	Line-Based, MSI, MSI-X Single Vector, MSI-X Multi Vector
RtNalE1000	All	Line-Based, MSI, MSI-X Single Vector	Line-Based, MSI, MSI-X Single Vector, MSI-X Multi Vector
RtNalRtl	All	Line-Based, MSI, MSI-X Single Vector	Line-Based, MSI, MSI-X Single Vector, MSI-X Multi Vector

See the *MANDO Supported NICs* document for an up-to-date list of supported devices.

Configuring Jumbo Frames

In MANDO, jumbo frame sizes are configured for an interface with Maximum Packet Size, which includes 14 bytes of Ethernet header and does not include FCS. The IP MTU must be configured to be less than or equal to (Maximum Packet Size – 14).

The TCP/IP stack supports Path MTU Discovery. MANDO devices will successfully communicate over TCP with another device that supports Path MTU Discovery, even if MTUs are configured to different sizes.

NOTE: All tested drivers support Jumbo Packets. To view a list of supported network cards, contact IntervalZero Customer Support.

TO CONFIGURE THE STACK AND YOUR NIC CARDS TO USE JUMBO FRAMES:

- Configure the MTU size for the Stack to be less than or equal to (Maximum Packet Size – 14).

NOTE: Maximum frame size is hardware-specific, and you should consult your NIC card's hardware documentation for acceptable values.

UDP or ICMP over Fragmented IP

For User Datagram Protocol (UDP) or Internet Control Message Protocol (ICMP) over fragmented IP, if two devices are configured with different MTUs, they can only communicate with IP frame sizes of smaller MTU. To run UDP or ICMP over fragmented IP protocol, the IP MTU must be configured to the same number. Windows devices typically configure JUMBO frame sizes of IP MTU + 14 bytes of Ethernet header.

Example:

The Intel(R) Gigabit CT Desktop Adapter Windows driver includes 14 bytes of layer 2 frame header in Jumbo Packet size configuration. To run UDP or ICMP over fragmented IP between this driver and any MANDO interface, you would:

- configure the MANDO Interface MTU to *9000* (9014 – 14) through the RtConfig.rtreg configuration file.

  or

- configure the MANDO Interface MTU to *4074* (4088 – 14) through the RtConfig.rtreg configuration file.

Using an Ethernet Filter

The MANDO Ethernet Filter is an extensible interface between the RT-TCP/IP stack and a Real-time network interface. You can use this interface to filter all frames at the data link layer and send Ethernet frames directly to the RTND driver.

The filter layer is created as a Real-Time DLL (RTDLL). The filter is loaded by the stack and is associated with a given interface. Since it is an RTDLL, it can use all supported RTAPI calls.

Loading the Ethernet Filter

The Ethernet Filter is loaded when the RT-TCP/IP stack loads. Once you create a custom filter, you can associate the filter with an interface and configure it to be loaded by the stack using the RtConfig.rtreg configuration file.

Tips, Tricks and Configuration Hints

This topic contains helpful tips, tricks, and configuration recommendations for the real-time network (Network Abstraction Layer and RT-TCP/IP Stack).

Delays in Stack Response

When the stack is started, initialization procedures occur before the stack is online. You will experience a delayed response from the stack if you begin calling the stack before the initialization has completed. MANDO provides API functions that can be used to get online status:

- RtnIsDeviceOnline – Used to get the online status of a network device for link status monitoring.
- RtnIsStackOnline – Used to get the online status of the RT-TCP/IP Stack

See the MANDO SDK Help for information on these API functions.

Running in Verbose Mode

You can run the RT-TCP/IP stack in verbose mode to enable verbose logging, which can be helpful in determining issues with the stack. You can configure the Verbose setting in the RtConfig.rtreg configuration file under following unique SectionPath surrounded by brackets ([]):

```
[HKEY_LOCAL_MACHINE\SOFTWARE\INTERVALZERO\MANDO\RtNAL]
```

Setting	Description	Default	Type
Verbose	Enable verbose logging.	1	DWORD

NOTE: Running the stack in Verbose mode may cause the RT-TCP/IP stack to slow during system shutdown.

Potential Causes of Driver Lockup

If the Interrupt Thread Priority for a driver is configured to a value that is less than the priority of the application thread that sends data, it can result in a lockup of the device interface.

Error Messages

Error	Meaning
tfSheapMalloc: Memory allocation failed. Increase the configuration for the TCP/IP Stack heap.	This error may appear when you attempt to increase the number of Max Sockets without first allocating enough memory for the Stack, or when a buffer to be used in *send* or *sendto* is too large. To resolve this issue, increase the Stack heap size through the RtConfig.rtreg configuration file. The default value is 4096 kilobytes.
tfCountSemAlloc: The configured value of MaxConcurrency is too low. RTTCPIP: FATAL ERROR in TCP/IP STACK. Please terminate attached applications and stop the Stack.	These errors appear if the configured maximum concurrency for the TCP/IP Stack is too low. To resolve the issue, increase the maximum concurrency configuration in the RtConfig.rtreg configuration file, terminate RTSS applications using the Stack, and restart the Stack.
RTTCPIP: PLEASE RESTART TCP/IP STACK! It's not safe after an exception in a socket application.	This error occurs when an RTSS application has generated an exception without completing a socket API call, blocking *recv*, *recffrom*, *accept*, or *select*. It's impossible for the RT-TCP/IP Stack to clean up its resources and continue running, as it might cause a BSOD or lockup.

Error	Meaning
RTTCPIP: Error in RtndInitializeInterface for network card RtNalIGB. RTTCPIP: Error (0x5) initializing interface card RtNalIGB.	Error 0x5 is ERROR_ACCESS_DENIED. This means that the receive and/or transmit queues needed for the Stack are already used by another NAL application. **NOTE:** The name RtNalIGB in these error codes was an example.

Configuring MANDO USB

MANDO provides USB capabilities through a USB Host Stack, which can be optionally loaded when the MANDO system boots. The USB Stack supports the following types of devices:

- Keyboards – to be used with the MANDO Console to control the MANDO system, such as running and terminating processes. See the MANDO SDK Help for information on MANDO Console.
- Mass Storage Devices – supported by the MANDO Filesystem, facilitates file access through the supported File IO APIs.
- Hubs

You can configure MANDO USB through the `RtConfig.rtreg` file under Section Path `[HKEY_LOCAL_MACHINE\SOFTWARE\INTERVALZERO\MANDO\RtUSB]`

Setting	Default value	Type	Description
IdealProcessor	0	DWORD	DWORD Sets the processor on which the USB Stack will run.

3

Using MANDO Runtime

MANDO Runtime includes a MANDO Application batch file (`RtJigInput.bat`) and a keyboard-based console that can be used to run programming samples and real-time applications, as well as run simple commands to query information about the system.

SECTIONS IN THIS CHAPTER:

About the Batch File

The MANDO Runtime includes a MANDO application batch file, `RtJigInput.bat`, located at `<InstallDrive>\Program Files\MANDO\`. This file contains a list of RtssRun commands for all programming samples included with MANDO Runtime. It can also be used to run real-time applications.

Using MANDO Console

The MANDO Runtime includes MANDO Console, a console that supports USB keyboard input. MANDO Console supports the following commands (case-insensitive). You can find full usage information for each of these commands in the MANDO Help:

Command	Description
`Rtssrun <app_path>`	Runs the MANDO application found at the given path.
`Rtsskill`	When no parameter is given, this command lists the currently running MANDO processes.
`Rtsskill <PID>`	When a valid Process ID is given, that process is terminated.
`dir`	Lists all files and folders found in the current directory.
`DisplayVolumes`	Lists all of the FAT32 volumes on the system found by MANDO, as well as what type of media they are on (SATA or USB).
`cd <folder_path>`	Changes the current directory of the console to the given path.
`cls`	Clears the console of all output history.

MANDO Console also supports special keys that provide additional functionality to the console:

Key(s)	Description
Page Up/Down	Scrolls up and down through the console's output history.
Up/Down Arrow	Cycles through the history of previously entered commands.
Escape	Clears any input currently entered into the console.

Running Program Samples

MANDO includes several program samples to help developers create real-time programs that will run in the MANDO Runtime environment. MANDO Runtime provides a set of .rtss executables in the `<InstallDrive>\Program Files\MANDO\bin` directory.

RTSS SAMPLES:

Sample	Description
FastSemaphore	Demonstrates the usage and abilities of the FastSemaphore functions.
IPCLatency	Measures the latency for switching between two threads.
IPCPairLatency	Measures latencies from process B signal to process A wake up.
Mailbox	Creates an instance of a mailbox which is thread safe.
mSRTM	Runs a System Response Timer on different cores simultaneously, measuring the difference between the expected timer interval and the actual timer interval.
PciScanBus	Searches for devices on the machine's PCI buses. When a device is found, the PCI configuration information is printed to the screen.
PingPong	Spawns off two threads that signal each other back and forth for a set amount of time. After the time has expired, the main thread signals the threads to complete and end the program.
RDTPerformance	Provides an example of how to use MANDO-supported Intel® Resource Director Technology (RDT) to optimize the performance of Real-time threads with high performance requirements.
BeginThreadEx	Demonstrates how to use CRT functions - _beginthreadex and _endthreadex.
RTPerfMonitor	Displays system information including speed and type of processor running MANDO, HAL type, CPU utilization and other information that can be used to help measure RTSS application load.
SimpleDataExchange	Detects whether an event is open or not.

Sample	Description
SimpleProducerConsumer	Builds two applications: • the consumer application opens a memory address that is shared between the two Real-time processes. • the producer application creates and writes to a globally-named shared memory address.
SMP	Spawns three threads in a suspended state.
SRTM	Measures timer delivery latencies using the MANDO synchronized clock and timers and prints the results.
TimingAlloc	Measures malloc/free performance.
TimingAllocMt	Measures malloc/free performance in a multi-threading case.
FLSTest	Demonstrates hot to use FLS functions to manipulate FLS associated with the current thread.
IntelProE1000	Demonstrates attaching multiple ISRs and ISTs using line-based or MSI/MSI-X interrupts for multiple Intel Pro 1000 PT server adapters.
MultiIntrModeI225	Demonstrates attaching multiple ISRs and ISTs using line-based or MSI/MSI-X interrupts for multiple Intel I225/I226 adapters.
MultiVectorI350	Demonstrates attaching multiple interrupt vectors using MSI-X interrupts for Intel I350 Ethernet adapter.
PCIDIO	Demonstrates how to attach line based interrupts on PCIDIO line-based interrupt cards.

RTDLL SAMPLES:

Sample	Description
RTDLL	Demonstrates how to use a Real-time Dynamic Linked Library (RTDLL) in a Real-time process. RTDLLs provide library functionality to RTSS applications, similar to how DLLs are used in Windows applications.

Sample	Description
TLSTest	Demonstrates how to use explicit and implicit thread local storage (TLS) in a Real-Time process.

NAL SAMPLES:

Sample	Description
NalDataStream	Demonstrates transmitting and receiving data using Network Abstraction Layer (NAL).
NalMultiplePacketTx	Demonstrates transmitting multiple packets, receiving packets in a callback, and using packet timestamps using Network Abstraction Layer (NAL).
NalTimeStamping	Demonstrates using time stamping functionality in Network Abstraction Layer (NAL).

TCP/IP SAMPLES:

Sample	Description
RtTcpipClient	TCP or UDP echo client application used for testing Real Time TCP/IP protocol stack.
RtTcpipServer	TCP or UDP echo server application used for testing Real Time TCP/IP protocol stack.

To run samples automatically after the Real-time Kernel completes initialization, you must edit the `RtJigInput.bat` file. This file is deployed in the `<InstallDrive>\Program Files\MANDO\` folder.

By default, `RtJigInput.bat` will only run two instances of SRTM.rtss on the 1st and 2nd processors/cores. See below for instructions on how to use `RtJigInput.bat` to run other samples:

TO RUN PROGRAMMING SAMPLES USING RTJIGINPUT.BAT:

1. Boot the system in its Windows Boot Configuration.
2. Navigate to `<InstallDrive>\Program Files\MANDO\`.
3. Right-click `RtJigInput.bat` and select **Edit**.
4. Observe the RtssRun commands for all samples included with MANDO Runtime.

> **NOTE:** For more information on RtssRun, see the MANDO SDK Help installed with MANDO SDK.

5. Find the RtssRun command(s) for the desired sample(s).
6. Remove the comment characters (: :) to enable the Rtssrun command(s).
7. Re-boot the system from a GRUB bootable USB drive or hard drive.
8. Select the desired GRUB boot configuration. See GRUB Boot Configurations for more information.
9. Upon system boot, the sample(s) will run automatically after the MANDO Kernel startup.
10. Sample output will be displayed on the screen when the program ends.
11. Re-boot the system in its Windows Boot Configuration.
12. Navigate to `<InstallDrive>\Program Files\MANDO\`.
13. Open the `RtLogFile.txt` log file to view sample output.

Running User-Developed Real-Time Applications

You can use the MANDO application batch file, `RtJigInput.bat`, to run real-time applications (RTProcesses).

TO RUN REAL-TIME APPLICATIONS USING RTJIGINPUT.BAT:

1. Navigate to `<InstallDrive>\Program Files\MANDO\`.
2. Right-click `RtJigInput.bat` and select **Edit**.
3. Add an RtssRun command pointing to the desired application. For more information on RtssRun, see the MANDO SDK Help.
4. Reboot the system from a GRUB bootable USB drive or hard drive.
5. Select the desired GRUB boot configuration. See GRUB Boot Configurations for more information.
6. Upon system boot, the application(s) will run automatically after the Real-time Kernel startup.
7. Application output will be displayed on the screen when the program ends.
8. Reboot the system in its Windows Boot Configuration.
9. Navigate to `<InstallDrive>\Program Files\MANDO\`.
10. Open the `RtLogFile.txt` log file to view application output.

TO RUN SAMPLES/APPLICATIONS USING THE MANDO CONSOLE:

1. Boot the system from a GRUB bootable USB drive or hard drive.
2. Select the desired GRUB boot configuration. See GRUB Boot Configurations for more information.
3. Once MANDO has started, and the Console has initialized (indicated by a "C:\>" near the bottom of the screen), type `Rtssrun` followed by the path for the application to run.
4. Press **Enter**.
5. Output for the application will be displayed on the screen.
6. Output which exceeds the length of the screen can be inspected by either opening the log file in Windows as described above, or by scrolling through the output history with the Page Up and Page Down keys.

Getting Support

For support of this version of MANDO Runtime, please contact BetaProgram@intervalzero.com.

Index