INIC Hardware Concepts
Technical Bulletin

Supporting MOST® Networks
Media Oriented Systems Transport
Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip’s Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights.

Trademarks

The Microchip name and logo, the Microchip logo, dsPIC, FlashFlex, KEELOG, KEELOG logo, MPLAB, PIC, PICmicro, PICSTART, PIC™ logo, rPIC, SST, SST Logo, SuperFlash and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

FilterLab, Hampshire, HI-TECH C, Linear Active Thermistor, MTP, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

Analog-for-the-Digital Age, Application Maestro, BodyCom, chipKIT, chipKIT logo, CodeGuard, dsPICDEM, dsPICDEM.net, dsPICworks, dsSPEAK, ECAN, ECONOMONITOR, FanSense, HI-TIDE, In-Circuit Serial Programming, ICSP, Mindi, MiWi, MPASM, MFP, MPLAB Certified logo, MPLIB, MPLINK, mTouch, Omniscient Code Generation, PICC, PICC-18, PICDEM, PICDEM.net, PICkit, PICtail, REAL ICE, rLAB, Select Mode, SQI, Serial Quad I/O, Total Endurance, TSHARC, UniWinDriver, WiperLock, ZENA and Z-Scale are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

GestIC and ULPP are registered trademarks of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

flexPWR, JukeBox, Kleer, KleerNet, MediaLB, and MOST

The preceding is a non-exhaustive list of trademarks in use in the US and other countries. For a complete list of trademarks, email a request to legal.department@microchip.com. The absence of a trademark (name, logo, etc.) from the list does not constitute a waiver of any intellectual property rights that SMSC has established in any of its trademarks.

All other trademarks mentioned herein are property of their respective companies.

© 2009-2014, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

Preface

NOTICE TO CUSTOMERS

All documentation becomes dated, and this manual is no exception. Microchip tools and documentation are constantly evolving to meet customer needs, so some actual dialogs and/or tool descriptions may differ from those in this document. Please refer to our web site (www.microchip.com) to obtain the latest documentation available.

Documents are identified with a “DS” number. This number is located on the bottom of each page, in front of the page number. The numbering convention for the DS number is “DSXXXXXA”, where “XXXXX” is the document number and “A” is the revision level of the document.

For the most up-to-date information on Microchip development tools, please visit www.microchip.com.

INTRODUCTION

This chapter contains general information that will be useful to know before using the INIC Hardware Concepts. Items discussed in this chapter include:

- Notice to Customers
- Introduction
- Document Layout
- Conventions Used in this Guide
- The Microchip Website
- Customer Change Notification Service
- Customer Support
- Recommended Reading
- Document Revision History
INIC Hardware Concepts

DOCUMENT LAYOUT

This Specification describes how to use the INIC Hardware Concepts. The document is organized as follows:

• Chapter 1. “Introduction”
• Chapter 2. “Power Management Architecture”
• Chapter 3. “Hardware Blocks”
• Chapter 4. “Implementation Examples”
• Appendix A. “MOST Specifications and INIC”
• Appendix B. “ECL Extensions”
• Appendix C. “Glossary and General Terms”

CONVENTIONS USED IN THIS GUIDE

Within this manual, the following abbreviations and symbols are used to improve readability.

<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIT</td>
<td>Name of a single bit within a field</td>
</tr>
<tr>
<td>FIELD.BIT</td>
<td>Name of a single bit (BIT) in FIELD</td>
</tr>
<tr>
<td>x…y</td>
<td>Range from x to y, inclusive</td>
</tr>
<tr>
<td>BITS[m:n]</td>
<td>Groups of bits from m to n, inclusive</td>
</tr>
<tr>
<td>PIN</td>
<td>Pin Name</td>
</tr>
<tr>
<td>SIGNAL</td>
<td>Signal Name</td>
</tr>
<tr>
<td>msb, lsb</td>
<td>Most significant bit, least significant bit</td>
</tr>
<tr>
<td>MSB, LSB</td>
<td>Most significant byte, least significant byte</td>
</tr>
<tr>
<td>zzzzb</td>
<td>Binary number (value zzzz)</td>
</tr>
<tr>
<td>0xzzz</td>
<td>Hexadecimal number (value zzz)</td>
</tr>
<tr>
<td>zzh</td>
<td>Hexadecimal number (value zz)</td>
</tr>
<tr>
<td>rsvd</td>
<td>Reserved memory location. Must write 0, read value indeterminate</td>
</tr>
<tr>
<td>code</td>
<td>Instruction code, or API function or parameter</td>
</tr>
<tr>
<td>Multi Word Name</td>
<td>Used for multiple words that are considered a single unit, such as: Resource Allocate message, or Connection Label, or Decrement Stack Pointer instruction.</td>
</tr>
<tr>
<td>Section Name</td>
<td>Emphasis, Reference, Section or Document name.</td>
</tr>
<tr>
<td>VAL</td>
<td>Over-bar indicates active low pin or register bit</td>
</tr>
<tr>
<td>x</td>
<td>Don’t care</td>
</tr>
<tr>
<td>&lt;&gt;</td>
<td>&lt;&gt; indicate a Parameter is optional or is only used under some conditions</td>
</tr>
<tr>
<td>{,Parameter}</td>
<td>Braces indicate Parameter(s) that repeat one or more times.</td>
</tr>
<tr>
<td>[Parameter]</td>
<td>Brackets indicate a nested Parameter. This Parameter is not real and actually decodes into one or more real parameters.</td>
</tr>
</tbody>
</table>
THE MICROCHIP WEBSITE

Microchip provides online support via our WWW site at www.microchip.com. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user’s guides and hardware support documents, latest software releases and archived software

- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing

- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip’s customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at www.microchip.com. Under “Support”, click on “Customer Change Notification” and follow the registration instructions.

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: http://microchip.com/support
RECOMMENDED READING

This Technical Bulletin describes INIC Hardware Concepts and references the following documents as recommended and supplemental resources.

Documents listed below and referenced within this publication are current as of the release of this publication and may have been reissued with more current information. To obtain the latest releases of Microchip documentation please visit the Microchip website. Please note, some Microchip documentation may require approval. Contact information can be found at www.microchip.com.

All non-Microchip documentation should be retrieved from the applicable website locations listed below. Microchip is not responsible for the update, maintenance or distribution of non-Microchip documentation.

Because the Internet is a constantly changing environment, all Internet links mentioned below and throughout this document are subject to change without notice.

[1] MOST Specification 2.5


[4] INIC Hardware Data Sheets
MOST25:
OS81050 INIC Data Sheet. DS81050AP11
OS81060 INIC Data Sheet. DS81060AP4

MOST50
OS81082 INIC Data Sheet. DS81082FP5
OS81092 INIC Data Sheet. DS60001271

MOST150:
OS81110 INIC Data Sheet. DS81110AP5
OS81118 INIC Data Sheet. DS60001252


[8] MOST150 oPHY Automotive Physical Layer Sub-Specification

[9] MOST150 cPHY Automotive Physical Layer Sub-Specification

[10] i²C-Bus Specification

[12] INIC API User’s Manuals
   MOST25:
   OS8105x/OS8106x INIC API User’s Manual. UM_OS8105x_6x_INIC_API
   MOST50:
   OS81082 INIC API User’s Manual. UM_OS81082_INIC_API
   OS81092 INIC API User’s Manual. UM_OS81092_INIC_API
   MOST150:
   OS81110 INIC API User’s Manual. UM_OS81110_INIC_API
   OS81118 INIC API User’s Manual. UM_OS81118_INIC_API

   Part 2: Electrical transient conduction along supply lines only. ISO 7637-2.

[14] Road vehicles - 16750-2 Environmental conditions and testing for electrical and electronic equipment
   www.iso.org

[15] MPM85000 Automotive Power Management Device Data Sheet

[16] OS85650/2 I/O Companion Chip Data Sheet

[17] MOST FunctionBlock NetBlock
INIC Hardware Concepts

DOCUMENT REVISION HISTORY

INIC Hardware Concepts. The most extensive and pertinent application changes are listed in Table 1, although various other differences may be observed between document revisions.

TABLE 1: REVISION SUMMARY

<table>
<thead>
<tr>
<th>Location</th>
<th>Description of Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DS60001264A:</strong></td>
<td>May 2014</td>
</tr>
<tr>
<td>Edits made throughout this specification</td>
<td>Reformat to Microchip template and document numbering</td>
</tr>
<tr>
<td>Hardware Blocks</td>
<td>Adopted MOST Specification terminology of <em>Sleep Power State</em> for Sleep Mode and <em>Active Power State</em> for Active Mode throughout document.</td>
</tr>
<tr>
<td>EHC</td>
<td>INIC’s ( t_{\text{TimePwrOff}} ) timer is now used as a backup or failsafe timer in case EHC does not start up. Refer to Section 3.8 “ECU Wakeup”.</td>
</tr>
<tr>
<td>EHC</td>
<td>EHC is now responsible for timer ( t_{\text{PwrSwitchOffDelay}} ) that determines when device shuts down. Refer to Section 3.8 “ECU Wakeup”.</td>
</tr>
<tr>
<td><strong>TB0520AN3d10:</strong></td>
<td>Nov. 2013 (Draft Release)</td>
</tr>
<tr>
<td>Appendix A</td>
<td>Expanded explanation of typical network startup and added Figure A-5 (for clarification).</td>
</tr>
<tr>
<td>Appendix B</td>
<td>Added new section: <em>ECL Start Sequence</em>. And reworked all figures to include the new definition of the start sequence. Added new section: <em>System Test Wakeup Event</em>, which includes all ECL test definitions and clarification of how each test behaves under different network conditions. Also defined new ECL test: <em>stable lock</em>. Removed all references to unidirectional ECLs and ( t_{\text{RBDS}\text{Start}} ). Added System Test start example, Figure B-10 and description. Updated ECL specifications. Expanded <em>ECL Timing Definitions Table B-3</em> table.</td>
</tr>
<tr>
<td><strong>TB0520AN3r8:</strong></td>
<td>Jan. 2013</td>
</tr>
<tr>
<td>General</td>
<td>The use of ECL is emphasized for network startup, and ECL system test becomes the preferred method for <em>Ring Break Diagnosis</em>.</td>
</tr>
<tr>
<td>Appendix A</td>
<td>Updated to add more detail concerning use of ECL in system startup.</td>
</tr>
<tr>
<td>Appendix B</td>
<td>Completely reworked to show ECL system tests and expand the ECL specification table.</td>
</tr>
<tr>
<td><strong>TB0520AN2:</strong></td>
<td>Nov. 2011</td>
</tr>
<tr>
<td>General</td>
<td>Converted to international nomenclature for voltage units (V). Updated references to latest specifications.</td>
</tr>
<tr>
<td>Power Management Architectures</td>
<td>Added dual ECL wire figure. Removed PermissionToWake/CapabilityToWake functions since removed from the MOST spec.</td>
</tr>
<tr>
<td>Hardware Blocks</td>
<td>Network Front End: expanded section and added MOST150 coax. <em>Ring Break Diagnosis</em>: Added description for Phase 2 and Phase 3.</td>
</tr>
<tr>
<td>Applications Examples</td>
<td>Removed non-MOST compliant examples.</td>
</tr>
<tr>
<td>Appendices</td>
<td>Added new Appendix A. “<em>MOST Specifications and INIC</em>” Added new Appendix Appendix B. “<em>ECL Extensions</em>” Added this Section “<em>Document Revision History</em>” Updated the Glossary</td>
</tr>
<tr>
<td><strong>TB0520AN1:</strong></td>
<td>Sep. 2009</td>
</tr>
<tr>
<td>General</td>
<td>Initial version of the document</td>
</tr>
</tbody>
</table>
Chapter 1. Introduction

The MOST Specification 2.5 [1] Chapter 4 contains a hardware section that provides a basic overview of how hardware in a MOST device should operate. This data is targeted for MOST Network Interface Controller (NIC) integrated circuits. Unfortunately, the MOST Specification 3.0 [2] does not contain any hardware-related section. Therefore, this document updates the hardware section of the MOST Specification 2.5 [1], as it relates to INIC (Intelligent Network Interface Controller) integrated circuits, and compliments the MOST Specification 3.0 [2]. Specifically, this document covers the MOST25 INICs (OS81050, OS81060), the MOST50 INICs (OS81082, OS81092), and the MOST150 INIC (OS81110).

The INIC architecture was born out of years of extensive MOST network testing using the NIC architecture, and vastly improves network operation and performance. When using a NIC device, the entire network startup is reliant on the External Host Controller’s power-up time and firmware stability. An INIC is a combination of a NIC with an on-chip CPU that manages all the low-level, real-time network functions. This encapsulation of the low-level network functions significantly simplifies MOST network implementation. In addition, INIC manages network startup and includes a watchdog timer for the External Host Controller (EHC) and application software. Managing network startup on chip provides a quicker and more stable network startup and helps protect network operation from errant application software. The encapsulation of all the low-level network functions also eliminates that complexity from the EHC; thereby shortening development time. The INIC architecture uses a message-based control interface instead of the register-based interface used with the NICs. This message-based API uses small powerful messages to configure INIC, which further reduces the complexity of EHC firmware development. The internal differences between different INICs are hidden under a common API which provides an easy migration path between different network speeds (MOST25, MOST50, and MOST150) with minimal effort.

Due to these differences in software partitioning, the hardware requirements listed in the MOST Specification 2.5 [1] (for a NIC architecture) need to be modified to fit this new architecture.

This document is divided into several chapters. Chapter 2. “Power Management Architecture” on page 10 covers system-wide power management architectures. Chapter 3. “Hardware Blocks” on page 15 describes the hardware blocks needed within a device, similar to the hardware requirements section listed in the MOST Specification 2.5 [1]. Chapter 4. “Implementation Examples” on page 37 provides example implementations of the hardware blocks, for a more detailed understanding of electrical, optical, and coaxial MOST systems. Two major appendices are also provided: Appendix A. “MOST Specifications and INIC” attempts to clarify the MOST Specification as it relates to different INIC network speeds, and Appendix B. “ECL Extensions” attempts to clarify and extend the MOST ECL specification.
Chapter 2. Power Management Architecture

Although the MOST network supports many different network topologies, the ring is used in the majority of designs due to its cost benefit. After choosing the network topology, the architecture for managing network power must be decided upon. In modern automotive infotainment systems, the key-switch position (clamp status) is rarely used to directly control the device or Electronic Control Unit (ECU) power. Doing so would not allow advanced networks to start up and shut down properly, nor save parameters needed at the next power up. Therefore, all systems require some sort of power management. An ECU that manages network startup and shutdown is said to contain the Power Master functionality (see MOST Specification 3.0 [2], Power Management Section 3.1.2.3). All other ECUs in the network that are not the Power Master are referred to as Power Slave devices.

2.1 SWITCHED-BATTERY POWER MANAGEMENT

One method of power management is to have one ECU manage the physical power sent to all other network ECUs, as illustrated in Figure 2-1. In this scenario, the one ECU managing power supports a Sleep Power State, which is defined as a power mode where the current draw from a continuous battery supply is at an absolute minimum. The Sleep Power State is required on any ECU that is connected to the continuous battery supply so the battery is not drained when the car is parked for extended periods of time. ECUs that are disconnected from the battery (shown as BatSwP) do not need to support a Sleep Power State since power to the entire ECU is removed from power when not in use. The one ECU managing power contains the Power Master function that manages network startup and shutdown, as well as the switched battery power to all other network ECUs. Since the Power Master ECU is connected to the continuous battery supply (BatConP) and supports the Sleep Power State, it can transition to a fully-powered state (defined as the Active Power State) based on local events (i.e. non-network events). Examples of local events include a change in key position (clamp status received from the CAN network) or a power on button being pressed. Local events must be qualified before allowing them to wake up other network ECUs. When the Power Master ECU wakes from one of these local events, it enables power to the rest of the network ECUs (BatSwP) and starts up the network. When the Power Master wants to shut down the network (e.g. clamp status indicating key switch position off or a door-open event), it sends commands around the network informing all other network ECUs to prepare for network shutdown, and then after an appropriate amount of time, disconnects all other ECUs from power.
FIGURE 2-1: SWITCHED-BATTERY POWER MANAGEMENT

The benefit of this switched-battery power management approach is that only one ECU needs to support the Sleep Power State, so every other ECU has simpler power management requirements. This approach has the drawbacks of requiring a special switched-battery supply (BatSwP) cable be routed to every ECU from the Power Master ECU, and does not allow any other ECU to wake the network.

2.2 NETWORK ACTIVITY POWER MANAGEMENT

A considerably more flexible power management method supports local wakeup events from other ECUs that do not contain the Power Master function. To support this, the continuous battery power (BatConP) and support for the Sleep Power State must be included in every network ECU, as illustrated in Figure 2-2. An example of local wakeup could be a telematics ECU that wakes up the network due to a received wireless event. Although this scenario supports any ECU waking the network, typically only a few ECUs would have valid reasons to do so. Under normal conditions, the Power Master ECU still wakes (and shuts down) the network. This scenario requires that all ECUs can wake due to network activity detection so that network ECUs downstream from the Power Master ECU will exit the Sleep Power State when activity is detected on the network interface. Once the Power Master starts up the MOST network, each ECU downstream exits the Sleep Power State, and propagates the network activity, causing the next ECU to wake, and so forth. When the Power Master ECU receives activity from the preceding ECU, the network locks and normal communication commences.
When the Power Master wants to shut down the network, it sends commands around the network telling all other ECUs to prepare for network shutdown, and then after an appropriate delay, the Power Master turns off network activity. This in turn, causes the other network ECUs to revert to the Sleep Power State.

The benefits of each ECU being connected to continuous power are that a special power line does not need to be routed to every network ECU and that ECUs other than the Power Master can wake the network when needed. The trade-off is that each ECU must support the Sleep Power State and be able to wake from network activity. For optical and coaxial MOST systems, the fiber optic receiver or the coaxial receiver contains a sleep mode and network activity wakeup support. For electrical MOST50 systems, the network activity detection must be external, since the network front-end circuitry only consists of passive components.

### 2.3 ECL AND/OR NETWORK ACTIVITY POWER MANAGEMENT

A third power management method, illustrated in Figure 2-3, utilizes one unshielded wire, called an Electrical Control Line (ECL, see MOST Electrical Control Line Specification [3]), that is common to all ECUs. This signal is used for ECU wakeup and diagnostics. The ECL interface is low speed, uses the battery supply and ground as signal levels, and can be implemented with discrete components.
circuitry or with a LIN transceiver. Under normal conditions, the Power Master ECU wakes all ECUs in the network by pulling the ECL low (active). All other ECUs exit the Sleep Power State when the ECL is pulled low. Once all ECUs are awake, the Power Master ECU then starts up the network through INIC.

As in the previous method, when the Power Master wants to shut down the network, it sends commands around the network informing all other ECUs to prepare for network shutdown, and then after an appropriate amount of time, the Power Master turns off network activity. This in turn, causes the downstream ECUs to revert to the Sleep Power State.

The ECL provides the benefit that every ECU receives a wakeup indication at the same time, so the network powers up and achieves lock faster than when using network activity alone. Another benefit is that the ECL provides a (low-bandwidth) communication path for quicker diagnostics if the network is not operational. Optical and coaxial MOST systems support network activity by default, so the ECL adds redundancy and better diagnostics. For electrical MOST systems, the ECL is the primary method of waking from the Sleep Power State, in lieu of additional network activity detection circuitry.
2.4 ECL AND/OR NETWORK POWER MANAGEMENT (MORE ROBUST)

Whereas Figure 2-3 illustrates the logical ECL connection, Figure 2-4 depicts a more robust connection method where each ECU uses two connector pins for ECL, which are shorted together on the board. The ECL can then follow the MOST network wiring in the harness, and provides redundancy in case the ECL wire is cut in one location, the ECL continues to operate normally.

Although three power management methods have been described, many other options are viable when designing a MOST network architecture.

FIGURE 2-4: ECL AND/OR NETWORK ACTIVITY POWER MANAGEMENT (MORE ROBUST)
Chapter 3. Hardware Blocks

The fundamental hardware architecture of a MOST ECU, using an INIC for the MOST network interface, is illustrated in Figure 3-1. As this figure is generic, actual implementations may vary from this diagram. In addition, this diagram only illustrates the basic connections between areas. The hardware blocks are:

- **Network Front End** - Interface between the INIC chip and the MOST network physical layer.
- **MOST INIC** - Intelligent Network Interface Controller IC which manages all the lower software layers of the MOST network protocol.
- **External Host Controller (EHC)** - Micro-controller which defines and manages the application layer.
- **Application** - Hardware needed to implement the desired application, if applicable (the EHC block could contain the entire application where no extra application hardware is needed).
- **Temperature Sensor** - Measures ECU temperature, if needed.
- **Power Management** - Manages power supplies and power supply control circuitry.

Figure 3-1 illustrates a MOST ECU that supports the **Sleep Power State** where continuous power is always connected to the ECU. The **Sleep Power State** support allows the ECU to wake due to network activity, a separate control line, or a local event.

**FIGURE 3-1: GENERIC MOST ECU HARDWARE BLOCK DIAGRAM**
In the *Sleep Power State*, the ECU draws a minimal amount of current (typically less than 100 µA). Exact ECU requirements for the *Sleep Power State* are system-integrator/OEM specific. To obtain this low power level, certain blocks are powered off, while others remain powered to support wake up event detection. Special attention must be paid to eliminate parasitic currents between powered and unpowered blocks to achieve minimal power consumption in the *Sleep Power State*.

Figure 3-1 differs from the MOST Specification 2.5 [1] *Hardware Section* in that no external watchdog trigger is shown. A sophisticated watchdog timer is integrated in the INIC and is part of the control communications (see Section 3.2.1 “INIC Watchdog Timer” for more information).

### 3.1 NETWORK FRONT END

The network front end block, composed of a network transmitter (Tx) and receiver (Rx) PHY, can be either electrical (ePHY), optical (oPHY), or coaxial (cPHY) based on the speed grade and INIC used. The layout of electrical RX and TX signals are critical to achieving a clean, low-jitter network. Layout guidelines for INIC and the network front end are provided in the INIC Hardware Data Sheets [4]. Although typical circuitry is shown here as examples, the particular INIC data sheet is the controlling document.

The MOST Cooperation specifies the MOST network physical layer, including signal quality requirements, in the following documents:

- MOST150 (oPHY and cPHY): MOST Physical Layer Basic Specification [7]
  - Optical: MOST150 oPHY Automotive Physical Layer Sub-Specification [8]
  - Coax: MOST150 cPHY Automotive Physical Layer Sub-Specification [9]

#### 3.1.1 MOST25 oPHY

The fiber-optic receiver (FOR) must be connected to continuous power (ContP) to support waking on network activity. When activity is detected, the FOR wakes from the *Sleep Power State*, and drives the STATUS pin low. When the power management block detects STATUS low, it enables network switched power (NwSwP), thereby waking the rest of the ECU. In some optical systems, the receiver PHY is labeled FOR (Fiber Optic Receiver), and the transmitter PHY is labeled FOX (Fiber Optic Transmitter). The combination of the two is labeled FOT (Fiber Optic Transceiver).

For MOST25 networks, series resistors are used in the network transmit and receive paths. The value should be small enough to minimize rise time (which helps meet the MOST Specification of Physical Layer [5]), while also minimizing EMI. The distance between the INIC and the FOT should also be minimized. The TX transmitter line also has a weak pull down which minimizes transients during power up and power down. The transmitter PHY supports a RGAIN pin which is connected to INIC (through support circuitry as shown in Figure 3-2) and manages a half-power transmit mode that can be selected through an INIC command.
For systems that require thermal management of the FOX, a pull-down transistor on the RGAIN pin can be managed by the EHC (FOX_OFF in Figure 3-2), assuming an analog RGAIN pin exists on the FOX. Otherwise, the power supply for the FOX must be controlled.

3.1.1.1 OPTICAL PHYSICAL LAYER (oPHY)

Current MOST optical networks utilize a polymer optical fiber (POF) physical layer with a typical diameter of 1 mm. The POF physical layer was chosen to provide a low-cost physical layer that is immune to EMI/EMC issues. The large diameter core keeps costs down by relaxing the mechanical-interface tolerances; however, this large core also has a minimum bend radius of approximately 25 mm. Two types of MOST network connectors are used to interface ECUs with the MOST network physical layer wiring harness: an integrated pigtail where the FOT is integrated inside the connector housing at the edge of the ECU, and a flex pigtail where the FOT is separate from the connector housing at the edge of the ECU.

The integrated pigtail connector, illustrated in Figure 3-3, has the benefit of one connector; thereby lowering costs. Some trade-offs are that the FOT is closer to the edge of the ECU box which could increase EMI, and the location limits where INIC can be placed. For MOST25 networks, the INIC-FOT interface is single-ended; therefore, INIC should be as close as possible to the FOT, which provides better adherence to the physical layer specifications.
The flex-pigtail connector, illustrated in Figure 3-4, separates the FOT from the ECU connector which attaches to the external wire harness. The benefits of this approach are that the FOT and INIC can be placed close together and at the most beneficial location on the PCB, and since the FOT electronics are located farther from the ECU connector opening, EMI issues are diminished. The downside to this approach is the extra cost associated with the separate connector and the small optical cables needed to connect the two. The extra optical connection also adds some optical loss that must be accounted for.

In both figures above, the physical layer measurement points are illustrated, where SP1 is the electrical interface between the INIC TX and the FOT, SP4 is the electrical interface between the FOR and the INIC RX pin. If space is available, measurement test points should be added to the PCB to allow characterization for adherence to the MOST Specification of Physical Layer [5]. SP2 is the optical interface between the ECU transmitter and the wiring harness, and SP3 is the optical interface between the wiring harness and the ECU receiver.

Currently two temperature grades of POF are used for automotive physical layers: -40 °C to 85 °C (which is typically used in wiring harnesses), and -40 °C to 105 °C (which is typically used for the flex pigtail cables between the FOT and the ECU connector). When the FOX is transmitting, the junction between the fiber
and the FOX is typically 10 °C hotter than the ambient temperature. Some OEMs require temperature management of the FOX, where the FOX is turned off as temperatures get close to the limits of the POF. More information is in Section 3.5 “Temperature Management” while options to shutoff the FOX are mentioned in the previous sections.

3.1.2 MOST50 ePHY

For MOST50 electrical networks (ePHY), the network front-end circuitry is passive and includes a transformer. If waking from network activity is needed, then a separate activity detection circuit operating from continuous power (ContP) is required, as illustrated in Figure 3-5. For the exact circuitry between INIC and the ePHY front end, see the appropriate INIC Hardware Data Sheets [4]. Although a separate ECL is generally used for network wakeup and diagnostics, it is required in systems that do not support network activity wakeup.

**FIGURE 3-5: ELECTRICAL NETWORK FRONT END - MOST50**

3.1.3 MOST150 oPHY and cPHY

For MOST150 networks the network receiver PHY is connected to continuous power to allow waking from the Sleep Power State mode due to network activity. The receiver PHY contains an internal sleep mode to minimize power consumption when no network activity exists. The receiver PHY STATUS line connects to the power management block to wake the rest of the ECU from the Sleep Power State when network activity exists. The STATUS pin also connects to the MOST150 INIC. The MOST150 transmitter PHY supports a RST pin to disable the network output while the power supply is ramping.

*Figure 3-6* illustrates a MOST150 optical network. For systems that require thermal management of the FOX, the INIC TX transmit output can be forced off by the EHC sending an INIC API command, or an EHC pin can be wire-OR’ed into the FOX RST pin (assuming the FOX RST pin is diode isolated from the rest of the reset circuitry). In MOST150, the INIC to FOT interface is differential, which can support longer distances between INIC and the FOT than with the MOST25 interface. For additional detail, refer to the MOST150 oPHY Automotive Physical Layer Sub-Specification [8].
Figure 3-7 illustrates a MOST150 coax network (cPHY), which is similar to the MOST150 optical interface. The network front end components also include a STATUS signal in the receiver and a RST for the transmitter. For coax physical layers, the receiver PHY is labeled CEC (Coaxial to Electrical Converter), and the transmitter PHY is labeled ECC (Electrical to Coaxial Converter). The combination of the two is labeled CTR (Coaxial Transceiver). In some implementations, the ECC is integrated with the CEC, in which case the entire CTR device operates from continuous power (ContP). For additional detail, refer to the MOST150 cPHY Automotive Physical Layer Sub-Specification [9].
3.2 MOST INIC

The INIC chip (INIC Hardware Data Sheets [4]) contains either a single-ended (MOST25 - OS81050, OS81060) or a differential (MOST50/150 - OS81082, OS81092, OS81110) interface to the network front end circuitry. The MOST INIC also supports two interfaces to the EHC for configuration and network management: a Control Port supporting a slave I^2C format (see I^2C-Bus Specification [10]), or the MediaLB® interface (see MediaLB Specification [11]). The EHC uses one of these interfaces to configure INIC and communicate control data across the MOST network. Application hardware that is independent of the EHC usually handles the streaming and/or packet data on different INIC hardware interfaces such as the I^2S streaming port pins connected to a DAC or a video CODEC connected via MediaLB.

Generally, INICs are connected to network switched power (NwSwP) which is switched off in the Sleep Power State. The power management block then manages the wakeup from the Sleep Power State (turns on network switched power), and takes INIC out of reset when the power supplies have stabilized.

Since INIC automatically manages the lower-level functions of the network, it also can react to the ECU voltage state $U_{\text{LOW}}$ (see Section 3.7 “ECU Voltage Levels”). On INIC, the $PS1$ and $PS0$ pins are inputs that receive power state information from the power management block regarding the battery supply voltage level. Using this state information, INIC reacts according to the MOST Specification and can notify the EHC of the current power state. These pins are useful when the EHC doesn’t include power supply voltage detection or management. When the EHC manages the supply voltage states directly, the INIC $PS1/PS0$ pins can be grounded. INICs also contain a $PWROFF$ pin which indicates to the power management block when INIC is ready to be powered off.

Unlike the NIC implementations, timing-slave INICs contain all the low-level functionality to get the MOST network operational without any EHC support. Therefore, INIC’s power-on reset must not come from the EHC, which could slow overall network startup or disable it altogether (if the EHC does not initialize properly). However, the EHC does need the capability to reset INIC in case of extraordinary circumstances or to support INIC flash program memory update (for flash-based INIC devices). The pin used by the EHC to reset INIC must be high-impedance when the EHC is in reset so network operation is not disturbed if the EHC is reset.

INIC also includes a $RSOUT$ pin that should be tied to the EHC’s reset line to allow the ECU to reset INIC, the EHC, or both from across the network via an INIC API reset function, see the appropriate INIC API User’s Manuals [12]. The $RSOUT$ pin can also reset the EHC when the INIC watchdog timer expires, which can happen due to a variety of communication errors. The watchdog timer is configured through an INIC API function (INIC API User’s Manuals [12]). During early code development, the watchdog timer can be turned off to facilitate low-level debugging. However, for normal network operation, the watchdog timer must be enabled to cover application code stability issues. One danger to avoid is forgetting to re-enable the watchdog timer if it had been disabled during code development.

3.2.1 INIC Watchdog Timer

To ensure the EHC is functioning properly, INICs include a sophisticated watchdog timer where, if the watchdog times out, INIC protects itself from the EHC (EHCI Protected State) to keep the MOST network from getting corrupted. Once INIC is in EHCI Protected State, which is also the power-up default state, the EHC
must go through a special sequence to reconnect. The sequence is illustrated in Figure 3-8; INIC starts up in the EHCI Protected State wherein INIC will only respond to a few commands from the EHC. The EHC must go through the proper sequence to get INIC to the EHCI Attached State, where the EHC has full access to INIC and the MOST network. If the EHC does not respond properly and in a timely manner, then INIC will fall back into the EHCI Protected State (watchdog timeout).

If the watchdog timeout is triggered, INIC can pull its RSOUT pin low to reset the EHC. INIC actually contains two watchdog timers, one for transmitted control messages to the EHC, and one for received control messages from the EHC.

FIGURE 3-8: INIC EHCI STATES (WATCHDOG)

Note: When INIC is in reset, all clock outputs (including the MediaLB clock) are off.

3.3 EXTERNAL HOST CONTROLLER (EHC)

Although INICs get the network operational and provide the basic network information (portion of the FBlock NetBlock), INICs are unaware of the functionality of the rest of the ECU application. Therefore, an EHC is required to implement the rest of FBlock NetBlock functionality, along with any other FBlocks that are required for the particular application.

Generally the EHC is connected to network switched power (NwSwP) which is off during the Sleep Power State. In this scenario, the power management block manages all wakeup conditions. If the EHC wakes up from a local event, it can then decide whether to wake the rest of the network ECUs or not. The EHC can
use the **HOLD** line to keep the power supply in the *Active Power State* while deciding whether to wake the network. However, the preferred method is to let INIC directly control the power supply in the *EHCI Protected State*, and the EHC control **PWROFF** (via INIC API commands) when INIC is in the *EHCI Attached State* (see Section 3.8.1 "ECU Power Hold Strategy").

A *wakeup event* is considered a *local wakeup event* if it is local to a single ECU. Examples of *local wakeup events* are a Power-On switch being pressed, or a wireless event that must communicate information to other ECUs (e.g. unlock car). The other type of *wakeup event* is a *network wakeup event*, which is defined as a network-related event affecting all ECUs in the MOST network (e.g. network activity, an electrical control line (ECL), or a *Switch-To-Power* (STP) event). Some ECUs do not support any *local wakeup events* and only power up due to *network wakeup events*.

The EHC only wakes up the other network ECUs if the *wakeup event* has been properly qualified (e.g. debouncing a switch, or meeting a minimum active time for glitch protection, etc.). If the EHC decides to wake up the other network ECUs, it can pull ECL low or start up the network (which causes network activity to propagate through the network). Doing both provides some robustness as a failure in one network wakeup method does not prevent all ECUs from exiting the *Sleep Power State*.

The simplest scenario for a *Power Slave* ECU to wake the network is to assert ECL, which in turn wakes up the *Power Master* ECU (along with all other ECUs). Once the *Power Master* exits the *Sleep Power State*, it can startup the network in a normal fashion (as it would if receiving a local wakeup event, e.g. clamp status). The requirements for a *Power Slave* to wakeup the network, and for the *Power Master* to recognize the wakeup event are described in Section B.2.2 "Power Slave Wakeup".

### 3.4 APPLICATION

The application block refers to the peripherals needed for the particular application; such as receivers, DSPs, amplifiers, disk drives, wireless hardware, etc. The hardware configuration needed for the applications area is very device-specific, and might not even exist if the entire application resides inside the EHC. In ECUs with application areas that have high power consumption, the application area power (ApSwP) should be controlled separately from the EHC/INIC power. The application area might also use the direct battery power with its power managed separately due to stringent voltage requirements. This application hardware can then be switched off during abnormal voltage conditions (voltages outside of $U_{\text{Normal}}$, see Section 3.7 "ECU Voltage Levels").

Examples of this scenario include amplifiers protecting speakers or CD/DVD players protecting the motor drive mechanism. In other systems, the applications area may not use significant power so it could be connected to the same low-voltage regulated supply used by the EHC/INIC (NwSwP).

Additionally, some systems may need the application area to be active in order to wake up the entire network (such as a wireless interface that requires the network to send messages to other ECUs to start the car or open door locks), so the application area would be connected to a continuous power supply. In these systems, careful attention must be paid to minimizing current drawn in the *Sleep Power State* by ensuring no parasitic currents flow between powered and unpowered portions of the ECU.

The *MOST Specification 3.0* [2] *section 3.1.2.3.3* defines a low-power mode, called *Device Shutdown*, which allows network communications to occur, but places the ECU in a low-power state to conserve power. This state could turn off power...
to the applications block (ApSwP) and could also place the EHC in a low-power mode. If the EHC is in a low-power state, INIC commands received from across the network could wake the EHC. A typical method of waking the EHC is to use the INIC $\text{INT}$ line, which can be asserted when the Power Master sends a device wakeup command. Device Shutdown can be used to conserve power while the network is active, but the engine is off (vehicle parked).

3.5 TEMPERATURE MANAGEMENT

Even though ECUs should be designed to handle any automotive temperature extreme, some ECUs could malfunction or experience permanent damage when operating at temperatures beyond their specified limit (HMI displays, for example). In addition, in optical MOST networks, some system-integrators/OEMs require that the FOX Tx PHY be thermally managed to protect the POF fiber. The MOST Specification 3.0 [2] Over-Temperature Management section 3.1.5.6 defines how an ECU should respond to over-temperature events, with the $\text{g}_{\text{Shutdown}}$ and the $\text{g}_{\text{Critical}}$ being mandatory.

The MOST Specification defines the following alert levels, illustrated in Figure 3-9:

- $\text{g}_{\text{AppOff}}$ - Individual application shutdown. ECUs turn off the application circuitry while still leaving the network interface operational. The application must send $\text{NetBlock.FBlockIDs.Status()}$ (with the affected FBlocks removed) to the FBlock NetworkMaster indicating that parts of the application are not presently available.

- $\text{g}_{\text{Shutdown}}$ - Temperature shutdown (request). ECU broadcasts $\text{NetBlock.Shutdown.Status(TemperatureShutdown)}$ request to the network Power Master to shut down the network due to a temperature problem.

- $\text{g}_{\text{Critical}}$ - Critical emergency shutdown. If this temperature is reached, the EHC immediately shuts down all circuitry, including the network interface.

The MOST specification also mentions a fourth optional level (below $\text{g}_{\text{AppOff}}$) where an application internally limits its functionality to minimize heating. Since this level is internal to the ECU, no network notification is needed. The MOST Specification 3.0 [2] also defines two recovery temperatures, illustrated in Figure 3-9:

- $\text{g}_{\text{NetOn}}$ - Network operational. Once an ECU reaches $\text{g}_{\text{Shutdown}}$ or above (where the network is turned off), $\text{g}_{\text{NetOn}}$ is where the network should be restarted.

- $\text{g}_{\text{AppOn}}$ - Individual application restart. ECUs have cooled enough to turn the application circuitry back on. The application must send $\text{NetBlock.FBlockIDs.Status()}$ (with the affected FBlocks added) to the NetworkMaster indicating that the application is available again.
3.6 POWER MANAGEMENT

This section defines a typical power supply architecture, and defines the voltage levels for standard states. Although typical values are given, the actual values used must be defined by the system integrator. To cover the major aspects related to the power management block, this section describes an ECU which supports a *Sleep Power State* and a separate application power-controlled area (enabled with the *Switched-Application Signal*, SA). Figure 3-10 depicts a basic power management block, which includes a wide set of options that may not all be needed in a given application, but are included here for completeness. In addition, this figure only covers the logical aspects of the different sections, and doesn’t cover all conditions that should be taken into account for a real design. One issue that is not covered is when the ECU wakes from the *Sleep Power State*, the wake condition disappears, but INIC or the EHC has not had enough time to assert the PDROFF or HOLD signals. The system should be designed to guarantee enough time for INIC or the EHC to respond before reverting to the *Sleep Power State*. However, if the wakeup event occurs, but the EHC fails to initialize, the system should also be designed so the ECU falls back into the *Sleep Power State* (after some time) when the wake event deasserts, even if the EHC never responds. These issues provide a more robust system design by allowing the network to operate in the presence of EHC application code failure, while not draining the battery.
3.6.1 Load Dump and EMI Filtering

The load dump and EMI filtering block protects the ECU from voltage transients, provides some reserve power for short periods of voltage dropout, provides EMI protection, and prevents EMI radiation. The hardware configuration needed is very application specific. Figure 3-11 illustrates a basic circuit for discussion purposes; however, the actual circuit needed could vary dramatically. Since the automotive power supply consists of a battery and a charging circuit (alternator), the voltage seen by any ECU can vary widely. While most OEMs have their own standards for what power supply transients must be handled by an ECU, the ISO 7637-2 standard, Road vehicles - ISO 7637-2 Electrical disturbances from conduction and coupling [13], provides a common set of tests for load dump circuits. The load dump circuit in Figure 3-11 consists of diode D1 for reverse voltage protection, transient voltage suppressor D2 for limiting voltage spikes, and resettable fuse F1 to limit the current (through D1 and D2) when D2 is
clamping. F1 must be selected so that the minimum trip point over temperature is higher than the maximum current required by the ECU under all conditions. In high powered applications, the application section might be connected directly to the battery power (BatConP) while the regulators for the EHC/INIC are connected to the protected power (ProConP). In this figure, the capacitors and inductor provide EMI protection from high frequency transients, EMI filtering to minimize the radiation caused by the ECU, and reserve power for short periods of voltage dropout. The ISO 16750-2 standard (Road vehicles - 16750-2 Environmental conditions and testing for electrical and electronic equipment [14]), can by used to gauge the robustness of an ECU to power-supply voltage dropouts. If the ECU is connected to continuous power, then D2 and the polarized capacitors should be of the low-leakage variety, since leakage current from these components contributes to the standby current (I_{STBY}) budget when the ECU is in the Sleep Power State. The load dump circuitry also causes a voltage drop, V_{LoadDump}, between the ECU external battery power (BatConP) and the internal protected power (ProConP). This voltage drop should be taken into account when implementing the ECU voltage levels, since internal circuitry usually measures the voltage levels at the protected power (ProConP), but ECU voltage levels are specified from the outside of the ECU (BatConP) V_{LoadDump} will vary over current and temperature.

### FIGURE 3-11: LOAD DUMP AND EMI FILTERING

![Load Dump and EMI Filtering Diagram](image)

#### 3.6.2 Micropower Regulator

The micropower regulator is needed in systems supporting a Sleep Power State, and provides power to each area that is involved with wakeup detection and power-on logic. These areas can include the STP detector, the power-valid comparator, the switched-network regulators’ shutdown current, network activity detectors, and any volatile memory that must be powered in the Sleep Power State. Since some components are powered from the same power domain while in the Sleep Power State and the Active Power State (e.g. the OEC), the micro-power regulator must support their active power requirements as well.

The micro-power regulator and all circuitry powered from the continuous supply must be designed so that the entire ECU meets a manufacturer-specific standby current, I_{STBY} (typically 100 \( \mu \)A) while in the Sleep Power State.

#### 3.6.3 Electrical Control Line (ECL)

The ECL is included in the majority of MOST systems because it provides a quicker network/ECU wakeup mechanism than network activity alone, provides robustness as a secondary wakeup mechanism, and can be used as a diagnostics channel. As described in Chapter 2 "Power Management Architecture", ECL is a wire-OR’ed line connected to all the MOST ECUs. The ECL protocol is extremely simple thereby minimizing the load on EHCs. Since
the ECL logic levels are ground and the battery voltage, level translation and protection are needed once ECL enters the ECU, as illustrated in Figure 3-12. All ECUs must support receiver ECL circuitry (ECL_RX), however, only ECUs supporting Power Slave wakeup or supporting diagnostics across ECL must support the ECL transmitter circuitry (ECL_TX), this choice is system integrator specific.

FIGURE 3-12: ECL AND MOST NETWORK INTERFACE

As illustrated in Figure 3-12, the bi-directional ECL is split into a unidirectional ECL_TX transmit line and ECL_RX receive line. The ECL_RX line connects to the power management block to wake the ECU from the Sleep Power State when ECL asserts, similar to the Rx PHY STATUS line. To support this wake activity, the ECL circuitry must operate from a continuous power supply, typically ProConP. Once the ECU is in the Active Power State, the EHC manages the ECL. Since MOST25 and MOST150 Rx PHYs always include a STATUS line (indicates network activity present), ECL is not absolutely necessary for wakeup; however, ECL provides a faster network wakeup mechanism. For MOST50 electrical designs where the network front end is passive, ECL is the primary ECU wakeup mechanism.

3.6.4 Switched Network Power Regulators

Typically the EHC and INIC are powered from the ECU switched supplies (NwSwP) and are disconnected from power in the Sleep Power State. If the application block is powered separately, then the EHC controls when that block is powered or not (SA signal in Figure 3-10). If the ECU exits the Sleep Power State due to an event other than a network event (i.e. local event), then the EHC (application) must qualify the local wakeup event and decide whether to wake the other network ECUs or not (see Section 3.8 “ECU Wakeup” for more information).

A reset generator is required by INIC to set the part in a known state once the power supplies have stabilized. As previously mentioned, the power-on INIC reset must not come from the EHC; otherwise, network startup could be delayed and the network would be less robust (since INIC would be dependent on the
EHC to release reset). Figure 3-13 illustrates a typical reset configuration, where a separate reset generator resets all hardware on power up (assuming the EHC requires an external reset). INIC also has an open-drain RSOUT pin that connects to the EHC reset. With this connection, INIC can be configured to reset the EHC when the following conditions occur:

- When the watchdog times out
- By using an INIC API function, which has parameters to reset INIC, the EHC, or both simultaneously.

FIGURE 3-13: RESET CIRCUIT

If the EHC does not need an external power-on reset (POR) signal, then the reset generator diode (D4) to the EHC reset can be removed. Assuming D4 is not needed, the diode (D3) between the reset generator and INIC is then only necessary for MOST150 systems. The Tx PHY in MOST150 systems requires a reset signal when initial power is applied. The reset requirements of the Tx PHY differ from the INIC reset requirements; therefore, the EHC reset of INIC must be isolated so it doesn't influence the Tx PHY (D3 shown in Figure 3-13).

Many designs use INIC’s PWROFF pin as an input to the logic used to sustain the switched-network regulator (NwSwP). If the EHC resets INIC, the EHC must make sure the power supply remains in the Active Power State until INIC can reassert the PWROFF pin, assuming INIC solely manages keeping the ECU awake during normal operation. If MediaLB companion devices are used and MediaLB is the master clock source for the companion device, then the companion device should also be reset at the same time that INIC is reset, since the companion clock (MLBCLK) will be disabled while INIC is reset.

Note: Since resetting INIC affects network stability, the EHC should only reset INIC as a last resort, after all other methods of resolving communications problems fail.
3.7 ECU VOLTAGE LEVELS

The MOST Specifications define four generic ECU voltage ranges, which are delineated by three voltage thresholds. While typical values are given here; some of these values are application specific, while others are defined by the system integrator. The four ECU voltage ranges are:

**Low Voltage ($U_{\text{Low}}$):**
Defined as the voltage below the $V_{\text{Th,Low}}$ threshold (typically 7 V). The supply dropping below this threshold indicates that the ECU cannot continue communicating on the network. When INIC is in EHCI Protected State and receives indication that the power dropped below this threshold (through PS1/PS0), it shuts down the network transmitter immediately and then releases the PWROFF pin to indicate that it’s ready to transition to the Sleep Power State. In EHCI Attached State INIC simply forwards PS1/PS0 state to the EHC and it then determines what action to take. The $V_{\text{Th,Low}}$ threshold is typically set to 7 V to allow for internal voltage drops due to load dump circuitry ($V_{\text{LoadDump}}$ in Figure 3-11) and regulator dropout, while still supporting older 5 V applications.

**Critical Voltage ($U_{\text{Critical}}$):**
Defined as the voltage between the $V_{\text{Th,Low}}$ and $V_{\text{Th,Critical}}$ (typically 9 V) thresholds, where the NetInterface (EHCI/INIC) operates normally; however the EHC may need to secure external hardware that cannot operate on this low of a voltage. During the Active Power State, transitions across the $V_{\text{Th,Critical}}$ boundary must not affect the NetInterface operation. An ECU’s response to this range is very application-specific. INICs do not react to this voltage range, other than to notify the EHC.

**Super Voltage ($U_{\text{Super}}$):**
Defined as the voltage above the $V_{\text{Th,Super}}$ threshold (typically 16 V), where the NetInterface (EHCI/INIC) operates normally; however the EHC may need to secure external hardware that cannot function on voltages this high. During the Active Power State, transitions across the $V_{\text{Th,Super}}$ boundary must not affect the NetInterface operation. An ECU’s response to this range is very application-specific. INICs do not react to this voltage range, other than to notify the EHC.

**Normal Voltage ($U_{\text{Normal}}$):**
Defined as the voltage between the $V_{\text{Th,Critical}}$ and $V_{\text{Th,Super}}$ thresholds and covers normal operating voltages where the NetInterface operates normally.

**Note:** The ECU must be designed so that no damage occurs regardless of the voltage level. Hardware sensitive to voltage fluctuations should be shut down outside of its normal operating range for protection (e.g. power amplifiers and disk drive motors or pickups).

These thresholds are defined where power enters the ECU (BatConP and $V_{\text{BAT,ECU}}$), so voltage drops due to internal circuitry ($V_{\text{LoadDump}}$ in Figure 3-11) must be taken into account when setting internal threshold values. In addition, all thresholds must include hysteresis to avoid oscillations between voltage states.

Once in the Active Power State, ECUs must be able to communicate normally on the network while above the $V_{\text{Th,Low}}$ threshold, without interruption or re-initialization. $V_{\text{Th,Low}}$ should be set as low as possible to maintain network communication as long as possible. Although the threshold values are application dependent, the following relationship is always maintained: $U_{\text{Low}} < U_{\text{Critical}} < U_{\text{Normal}} < U_{\text{Super}}$. Figure 3-14 illustrates the relationship between the thresholds and the voltage ranges. $V_{\text{Th,Super}}$ is a rising-voltage threshold, whereas $V_{\text{Th,Critical}}$ and $V_{\text{Th,Low}}$ are falling-voltage thresholds. The rising-voltage
thresholds have hysteresis on the falling edge, while the falling-voltage thresholds have hysteresis on the rising edge.

INIC manages the low-level network functions, and can be told what condition the power supply is in, as well as whether an STP event occurred. These voltage conditions are priority-encoded onto the PS1 and PS0 pins as shown in Table 3-1.

### TABLE 3-1: INIC POWER MANAGEMENT ENCODING

<table>
<thead>
<tr>
<th>PS1</th>
<th>PS0</th>
<th>Status</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>$U_{\text{Normal}}$ (OK)</td>
<td>4th</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Switch-To-Power (STP)</td>
<td>2nd</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>$U_{\text{Critical}}, U_{\text{Super}}$</td>
<td>3rd</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>$U_{\text{Low}}$</td>
<td>1st (highest)</td>
</tr>
</tbody>
</table>

The priority indicates which encoding should be used (higher one) when simultaneous events are detected by the power management circuitry. $U_{\text{Low}}$ is the highest priority for INIC since it indicates that the supply is dropping below a level where the ECU can operate network communications, in which case INIC prepares for an immediate shutdown.

INIC can also notify the EHC of changes in the power management encodings/states through the INIC.PMIState.Status() function. Other than notifying the EHC, INIC internally does not react to the $U_{\text{Critical}}, U_{\text{Super}}$ voltage condition (treated the same as $U_{\text{Normal}}$).

Some OEMs specify that an ECU must not disrupt network communications during a brief excursion into the $U_{\text{Low}}$ region. If power supply capacitance alone cannot keep the ECU core supply (ProConP) above the $V_{\text{Th_Low}}$ threshold and if the EHC/INIC circuitry can function well below $V_{\text{Th_Low}}$, then the EHC can use a more sophisticated algorithm to manage power levels in lieu of INIC. In this case, the INIC PS1/0 pins should be tied to ground, and the EHC should tell INIC when network shutdown is required (via the INIC.PMIState.Set() function [INIC API User’s Manuals [12]]. One scenario could be an EHC software timer which manages the time between $V_{\text{Th_Low}}$ and the voltage where the INIC or EHC is reset ($V_{\text{Th_Reset}}$).

**FIGURE 3-14: VOLTAGE LEVELS AND THRESHOLDS**

![Voltage Levels and Thresholds Diagram](image-url)
3.8 ECU WAKEUP

Assuming an ECU supports a *Sleep Power State*, the ECU should not wakeup the other network ECUs unless the following two conditions are met:

- A qualified *local wakeup event* exists
- The power supply (V) is in the proper region (V > V\textsubscript{Th_Active})

The system integrator defines the *V\textsubscript{Th_Active}* threshold to be either *V\textsubscript{Th_Critical}* (typical) or *V\textsubscript{Th_Low}*. The power management block should qualify *wakeup events* with this voltage region before waking the system from the *Sleep Power State*. If *V\textsubscript{Th_Active}* is set to *V\textsubscript{Th_Low}*, then the hysteresis value used is critical to keeping the power supply from oscillating between the on and off states. The more robust implementation is to use *V\textsubscript{Th_Critical}* for *V\textsubscript{Th_Active}*, and have enough separation between the *V\textsubscript{Th_Critical}* wakeup threshold and the *V\textsubscript{Th_Low}* shutdown threshold to cover the threshold tolerances as well as the voltage offset of all ECUs across the vehicle (shown in Figure B-11 as *V\textsubscript{BAT_SHIFT}* and *V\textsubscript{GND_SHIFT}*). The ECU device behavior relative to supply voltage is illustrated in Figure 3-15.

**FIGURE 3-15: SUPPLY VARIATION ECU (DEVICE) BEHAVIOR**

![Diagram showing device state transitions](image)

Different systems are designed to wakeup using different methods, based on the system integrator’s or OEM’s desires. Not all *wakeup events* listed below are supported in a given system; however, all *wakeup events* can be grouped into two distinct categories:

- **Network Wakeup Events**: Network related (or external) *wakeup events*. Events designed to wake the entire network. Examples include:
  - Network activity - Network start-up on one ECU ripples through the network. Activity on the MOST network causes ECUs to wake from the *Sleep Power State* into normal operation.
  - Electrical Control Line (ECL) (see MOST Electrical Control Line Specification [3]) - The ECL is a wire-OR’ed signal connected to every network ECU, and can wake an ECU into normal operation or into a system diagnosis/test mode (also referred to as the ECL System Test), based on the length of the initial ECL pulse. The ECL can also be used to convey debug information between the ECL initiator and all the other ECUs.
  - Switch-To-Power (STP) Event - Is indicated the first time the continuous power is connected to the ECU, or through a continuous power supply disruption for a long enough period. The INIC can be configured to go into RBD mode (see Section 3.9 “Ring Break Diagnosis”) when a STP event occurs (deprecated).
• **Local Wakeup Events**: Internal events that are local to a single ECU and wakeup the EHC, but might not startup the MOST network. Typically the power management block recognizes one of these events and wakes up the ECU (i.e. transitions to the *Active Power State*) so the EHC (application) can determine whether to start the network or not. An event that causes the EHC to generate a network-related event is designated a *qualified local event*. When STP isn’t supported some ECUs wakeup on application of initial power, which allows the EHC to configure the module. Initial power is not considered a *wakeup event* because initial power is not a qualified reason to wakeup the rest of the network ECUs. In this scenario, the EHC configures the rest of the device and typically reverts to the *Sleep Power State*. Examples of *local wakeup events* (that could be qualified) include:

- Local button pressed (Power-On switch)
- Wireless event which needs network access
- Other buses such as CAN indicating key position (clamp status), or LIN indicating a door was opened

Systems supporting a *Sleep Power State* typically use network activity, ECL, or both (for redundancy) as a normal network *wakeup event*. In the past, STP was used in some systems to force RBD mode (see Section 3.9 “Ring Break Diagnosis”) when the ECL was not supported. The ECU containing the *Power Master* function generally wakes up the rest of the network ECUs (through ECL and/or network activity). Some other ECUs could wake the rest of the network ECUs, based on a local event.

Once an ECU exits the *Sleep Power State*, wake conditions have priority over **PWROFF** and **HOLD**. Therefore, as long as the wake condition is present, the power management section should not transition to the *Sleep Power State*, regardless of the **PWROFF** and **HOLD** status. For example, as long as network activity is present, the ECU should stay in the *Active Power State*. The power management section can support overrides to handle exceptional cases (such as a stuck wakeup condition); however, this is not part of normal operation.

### 3.8.1 ECU Power Hold Strategy

In Figure 3-10, the **PWROFF** signal comes from INIC and the **HOLD** signal comes from the EHC. The default state for these signals must be inactive (high) so that if an EHC does not properly initialize, the ECU will fall back into the *Sleep Power State* once the wake condition is removed. The preferred and most robust method of managing ECU power is to have INIC manage the regulators through the **PWROFF** pin. The INIC PMICConfig.TimePwrOff value is a backup timer in case the EHC fails and should be set to a value much greater than $t_{\text{PwrSwitchOffDelay}}$. The value could be set to a time longer than the time it takes to flash the EHC code, 60 s for example. This allows the ECU to power up and power down independent of the EHC providing for a more robust network, and a requirement for some OEMs. Once INIC transitions to the *EHCI Attached State*, the EHC is responsible for handling the $t_{\text{PwrSwitchOffDelay}}$ timer and controlling the **PWROFF** pin state using INIC API functions.

INIC powers up in the *EHCI Protected State*, and the EHC must set INIC to the *EHCI Attached State* which happens as part of the MOST NetServices™ initialization. If there are communication problems between INIC and the EHC, INIC falls back to the *EHCI Protected State* and can release **PWROFF** if the EHC does not recover. Under normal operation, when INIC is ready to power down, it is still in the *EHCI Attached State* so it does not release **PWROFF**. When the EHC is ready to power down ($t_{\text{PwrSwitchOffDelay}}$ expires), it sends the
INIC.PMIState.Set(PMI_PowerOffControl) message to INIC, which causes INIC to release the PWROFF signal (causing the ECU to enter the Sleep Power State).

If the EHC is able to control the INIC reset line, then the EHC or power management circuitry must make sure that the power supply stays in the Active Power State, since INIC cannot control the PWROFF pin while in reset.

The EHC must manage the MOST tPwrSwitchOffDelay timer. The timer should be set to its initial value, and start counting down under the following conditions:

- At initial wakeup from the Sleep Power State
- When entering the NetInterface state Off
- Upon receipt of a valid ECL start sequence (electrical wakeup)
- Upon completion of an ECL System Test

The EHC tPwrSwitchOffDelay timer should be halted (not counting and never expire) under the following conditions:

- When exiting the NetInterface state Off (i.e. going to state Init or Normal Operation).
- Upon receipt of a valid ECL system test start impulse

Utilizing the above methodology, the typical wakeup sequence would be:

- Wakeup event causes ECU to exit the Sleep Power State
- INIC powers up and asserts PWROFF (for up to tTimePwrOff)
- EHC powers up, boots, and starts tPwrSwitchOffDelay
- EHC (Power Master) starts the network
  - NetServices initialization causes INIC to reach the EHCl Attached State, causing INIC to stop timer tTimePwrOff
  - Power Master starts network activity
  - EHC detects activity (via NetInit or NetOn notification) and stops the tPwrSwitchOffDelay timer
- Network achieves the NetOn state and network communication commences

The typical shutdown sequence would be:

- Power Master determines that the network should be shutdown
- Power Master sends Shutdown request
- Power Master EHC eventually turns off network
- Lack of network activity results in all EHCs being notified of NetOff which causes them to reinitialize and start their tPwrSwitchOffDelay timer
- When each EHC’s tPwrSwitchOffDelay timer expires, it directs INIC to release the PWROFF pin, and the ECU enters the Sleep Power State.

If the EHC is unstable, then it never initializes NetServices, so the INIC stays in the EHCl Protected State. When INIC is in the EHCl Protected State and the network is not active, the INIC tTimePwrOff timer is started. When tTimePwrOff expires, INIC will release the PWROFF pin independent of the EHC, allowing the node to revert to the Sleep Power State, which keeps the errant module from draining the battery. If network activity recommences before tTimePwrOff expires, then INIC stops tTimePwrOff and locks to the network. Therefore, the network remains stable even when the EHC is not.
3.9 RING BREAK DIAGNOSIS

The majority of MOST networks use a ring topology due to the cost benefit. During normal operation, an ECU wakes up due to network activity or ECL. The Power Master ECU (typically the same ECU that contains the Timing Master) starts the network. When the last ECU in the MOST network ring changes to the Active Power State, it sends network traffic (activity) back to the Power Master, and the network achieves lock. If a cable break exists between any two ECUs, then the downstream ECUs will not propagate the network activity, since network activity never reaches them. Ring Break Diagnosis (RBD) is a process included in INIC that can be used to determine whether a break exists in the network (ring) and between which two ECUs the break resides. When ECL is supported, the stable lock test is preferred over RBD due to its simpler nature. RBD, as described in the MOST Specification 3.0 [2] Addendum A - MOST50 RBD, consists of three phases:

• Phase 1: Activation (critical that all ECUs start RBD within $t_{Diag\_Start}$)

• Phase 2: Diagnosis (handled within INIC)

• Phase 3: Delivery of results (optional, INIC produces results; delivery managed by the EHC)

Phase 1, activation, is started via one of three methods, and all INICs must start RBD within $t_{Diag\_Start}$:

• RBD activation method 1 (preferred method): The Electrical Control Line (ECL). The ECL wakes up all ECUs and can initiate an RBD test through the ECL parameters. In this scenario, the EHC has to detect the RBD test sequence and then start RBD through the INIC.RBDTrigger() function. When using ECL for RBD activation, the INIC.RBDOptions() function can be used to block STP events (coded on PS1/PS0) from causing RBD activation. For more information on this method, see Appendix B, “ECL Extensions”.

• RBD activation method 2: A disruption in the ECU’s power, defined as an STP pulse, activates RBD. The power management block should recognize the first time power is applied to the ECU, or when power is removed for a long enough period to be recognized as an STP event (see Section 3.9.1 “Switch-To-Power Event”). The power management block then encodes the STP on to the PS1/PS0 pins for INIC. The INIC.RBDOptions() function can configure INIC for this method.

• RBD activation method 3: Always startup in RBD mode, and then switch to normal mode when activity is detected on the incoming network link. The INIC.RBDOptions() function can configure INIC for this method. Care must be taken to make sure all ECUs start RBD within $t_{Diag\_Start}$ which can be difficult with ECUs that support local wakeup events (which have to be qualified).

Phase 2, diagnosis, is managed by INIC directly. INIC can notify the EHC when in the diagnosis phase through the INIC.NIState(NET_RBD) status function. Phase 3, delivery of results, is system dependent. When INIC finishes RBD (exits Phase 2), it informs the EHC through either:

• INIC.NIState(NET_ON) - where the ring obtained lock and no break exists, or

• INIC.NIState(NET_OFF) - where the ring failed to lock
Or the EHC can sign up for notification on \texttt{INIC.RBDResult()} where INIC will inform the EHC of the RBD results once the diagnosis phase is finished. The delivery of results for any particular ECU is system dependent and should be defined by the system integrator. Some examples are sending the results across a different diagnostics bus (e.g. CAN, LIN), or if RBD was started via an ECL System Test, the results would be returned across ECL.

### 3.9.1 Switch-To-Power Event

If the ECL is not supported in the system, then a Switch-To-Power (STP) event is another method for initiating Ring Break Diagnosis (RBD). An STP event is generally not triggered during normal operation of the vehicle, but in a car repair shop or on the assembly line. This method is deprecated due to the difficulty in synchronizing the power cycling on all ECUs simultaneously.

An STP event is defined as the first time continuous power is applied to an ECU, or when an ECU is removed from continuous power for a specified minimum amount of time ($t_{\text{STP}}$). Assuming STP events are supported, then when all ECUs are disconnected from main power by a central power switch or directly from the battery for long enough, then the power management block recognizes this event, it exits the Sleep Power State, and encodes the STP setting on the $\text{PS}_1$ and $\text{PS}_0$ pins, which causes INIC to enter RBD mode (if configured to do so). When INIC finishes RBD, the EHC reads the results through the \texttt{INIC.RBDResults()} function. Once RBD is finished, the ECU should revert to the Sleep Power State to conserve power (assuming the network is still broken and cannot obtain a stable lock).

The STP event detector must be connected to the continuous supply before the load dump and filtering (BatConP, $V_{\text{BAT}_{\text{ECU}}}$), so it can detect dropouts in the continuous supply. However, the STP detector must be designed so that a STP event is only generated when the continuous power drops below a minimum threshold for the appropriate amount of time. Short supply dropouts (e.g. transient dropouts from engine start) must not lead to a STP event. The supply dropout period to trigger an STP event, $t_{\text{STP}}$, is typically between two and four seconds. The power supply for the STP detector must be designed to survive periods shorter than $t_{\text{STP}}$ so that false STP triggering does not occur. High accuracy is not needed for the STP detector period as long as times shorter than two seconds do not trigger STP events. The first time power is applied to the ECU should also be encoded as an STP event. Even though each ECU in the network will have a slightly different STP power dropout time, whenever main power is removed for more than a few seconds, all ECUs will still indicate an STP event as long as the minimum time is observed.

When a proper STP event is detected (and the power supply returns above $V_{\text{Th, Active}}$), the ECU exits the Sleep Power State and the STP event must be encoded on $\text{PS}_1/\text{PS}_0$ long enough for INIC to initialize and start RBD.
Chapter 4. Implementation Examples

This chapter describes implementation examples that cover the power management, INIC, and the EHC areas. These examples include more details than can be covered through a generic power management discussion; however they are implementation-specific and are related to the particular power management hardware described. The application area is not covered since it is very device specific and not directly related to the network functionality. While these examples show detailed circuitry, no warranty is given regarding their suitability for a specific design.

4.1 MOST150 DESIGN

The following design example uses the MPM85000 Power Management device (see MPM85000 Automotive Power Management Device Data Sheet [15]) to simplify the MOST network power management area. All MPM85000 features are not needed in every design. For this particular implementation example, the assumption is made that support for switch-to-power event detection (for Ring Break Diagnosis activation) is not required (see Section 3.9.1 “Switch-To-Power Event”). Figure 4-1 and Figure 4-2 illustrate how the MPM85000 implements the power management functionality; therefore, these figures only illustrate the circuitry relevant to the MPM85000. Power supply decoupling required for other devices is not shown (see the respective device data sheets).

The WAKEHI pin determines the $V_{\text{Th_Active}}$ threshold and is shown configured for the $V_{\text{Th_Critical}}$ threshold. Therefore, the MPM85000 will only wake the ECU when a qualified wake-up event occurs and when the supply voltage is greater than the $V_{\text{Th_Critical}}$ threshold. In MOST optical networks, the network activity detection comes from the OEC STATUS pin. Since the activity detector is inside the OEC, it requires constant power during the Sleep Power State. More power is necessary for the OEC during the Active Power State. Similarly, in a coax system with a coax to electrical converter (CEC) that supports activity detection, such as the OS82150, the micropower regulator must be able to supply the full operational current to the PHY. To support this extra power, an external transistor is used with the MPM85000 to boost the output current needed in the Active Power State. In Figure 4-1, a PTC fuse is added to the transistor supply, and is only needed if short-circuit protection is required on the low-voltage continuous supply (ContP).
The MPM85000 also supports an ECL through the LIN pin, which is level-translated and split out into the unidirectional RXD and TXD pins for the EHC (illustrated in Figure 4-2). Since the MPM85000 has a LIN compliant driver, the system integrator could use the LIN protocol for a more sophisticated diagnostics interface, rather than the bit-banged approach mentioned in the MOST Electrical Control Line Specification [3]. If the EHC does not use this interface configured for LIN, then registers inside the MPM85000 could be used to bit-bang the ECL functions in lieu of using the dedicated RXD and TXD pins. If the ECL function is not used, then the associated circuitry can be omitted.
Figure 4-2 illustrates the communication section for a MOST150 application, where the EHC is assumed to have an internal power on reset (POR) generator. The Tx PHY for MOST150 systems requires a power-on reset. The MPM85000 reset generator provides a proper POR signal for the INIC and the Tx PHY. The EHC is depicted as having an open-drain GPIO (\texttt{NET\_RESET}) connected to the INIC \texttt{RST} pin to support INIC flash memory updates as well as an INIC reset for exceptional conditions (such as failed communications between INIC and the EHC). The power-up/reset default for this GPIO must be high-impedance so an EHC that gets reset does not interfere with network operation. Since the MPM85000 contains a delay associated with \texttt{PWROFF} signal recognition, short INIC resets by the EHC do not require the EHC to keep the MPM85000 in the \textit{Active Power State}.

In Figure 4-2, the EHC is shown communicating with the MPM85000 (and INIC) over an I\textsuperscript{2}C bus as an I\textsuperscript{2}C master. The MPM85000 contains internal registers which indicate the reason for exiting the \textit{Sleep Power State} as well as other status information. These registers can be read by the EHC and used to write the \texttt{NetBlock.DeviceInfo(WakeInfo)} parameters. The MPM85000 also includes a temperature sensor with multiple programmable interrupt capabilities which can be used to implement the over-temperature management mentioned in Section 3.5 \textit{“Temperature Management"}. The MPM85000 \texttt{INT} pin signals interrupt events and are requests for EHC intervention (support for \texttt{INT} and the MPM85000 Control Port are optional). Interrupt events include local/external wakeup and activity events, and temperature or voltage thresholds crossed.
This design supports power condition signaling at the INIC with the PS0/PS1 pins connected between the MPM85000 and INIC. When a power state change occurs (i.e. transition on PS0/PS1), INIC notifies the EHC through the INIC.PMIState.Status message, which causes MOST NetServices to fire the pmistate_changed_fptr() callback to the application. When using the latest INIC Configuration String settings for the PMIConfig.Config parameter (available on OS81110 INIC firmware revision 1.2.5 or later) INIC exclusively manages the PWROFF pin during the EHCI Protected State; however, the EHC manages the PWROFF pin (via API commands) when INIC is in the EHCI Attached State or the EHCI SemiProtected State.

Using the PS0/PS1 pins in this manner allows INIC to manage ECU power based on the current voltage level and the network state while INIC is in the EHCI Protected State (and the EHC is presumably unavailable). If the network is in the NetOn state, and the voltage level is above $U_{\text{Low}}$, then INIC keeps the PWROFF pin asserted. If the network goes to the NetOff state, INIC releases PWROFF after $t_{\text{TimePwrOff}}$ expires. If the voltage level drops below the $U_{\text{Low}}$ threshold while INIC is in EHCI Protected State, INIC immediately releases the PWROFF pin.

Once the EHC is up and running and driving NetServices properly such that INIC is in EHCI Attached State, INIC keeps the PWROFF pin asserted regardless of the power level or network state. However, INIC continues to inform the EHC of those states, and the EHC can control power according to its own logic, using INIC.PMIState.Set(Control) to have INIC release the PWROFF pin when desired.

As mentioned in Section 3.7 “ECU Voltage Levels”, if the OEM specifies that the device must remain operational during brief excursions into the $U_{\text{Low}}$ region without powering down even while INIC is still in EHCI Protected State, then the INIC PS0/PS1 pins should be tied to ground so that INIC will not release the PWROFF pin. In this configuration, all power management logic is under EHC control. Note that with the PS0/PS1 pins grounded, INIC cannot notify the EHC about power level boundary crossings. However, the MPM85000 can be configured to interrupt the EHC directly when a power threshold is crossed.

The INIC is attached to the $I^2C$ bus used by the EHC. To support higher throughput to the MOST network, the EHC can also be connected to the MediaLB interface on INIC. If the selected EHC does not natively support MediaLB (see MediaLB Specification [11]), then an OS85650/2 (see OS85650/2 I/O Companion Chip Data Sheet [16]) can be used to translate an EHC parallel bus into MediaLB channels.

One of the benefits to using the MPM85000 is the ability to override stuck wakeup events (such as a Power-On switch or ECL shorted to ground) from the EHC to provide a more robust ECU architecture. Once a wakeup event is ignored, transitions on that event clear the override automatically. Another benefit of the MPM85000 is that the high-value, low-current resistors used to measure the battery voltage are on-chip thereby reducing component count and eliminating the need for conformal coating.
4.2 MOST50 DESIGN

Figure 4-3 shows a design using the MPM85000 to simplify the MOST network power management area (see MPM85000 Automotive Power Management Device Data Sheet [15]). As with the last example, this implementation assumes that STP event detection is not required.

This design is very similar to the previous one, except that the electrical network front-end circuitry is passive and does not include a network activity detector. In addition, since no OEC exists, the continuous power output and related circuitry of the MPM85000 are not needed. The MPM85000 contains an on-chip low-power MOST50 network activity detector to support network wakeup in an electrical MOST network. Since the OS81092 is covered under the MOST Specification 3.0 [2], no persistent parameter storage is needed. Therefore, the OS81092 VDDU continuous power supply can be tied to the switched 3.3 V supply.

FIGURE 4-3: MOST50 APPLICATION EXAMPLE - POWER/NETWORK SECTION

This design supports power condition signaling at the INIC with the PS0/PS1 pins connected between the MPM85000 and INIC. When a power state change occurs (i.e. transition on PS0/PS1), INIC notifies the EHC through the INIC.PMState.Status message, which causes MOST NetServices to fire the pmistate_changed_fptr() callback to the application. When using the latest INIC Configuration String settings for the
INIC Hardware Concepts

PMICfg.Config parameter (available on OS81092 hardware revision C1D or later) INIC exclusively manages the PWROFF pin during the EHCI Protected State; however, the EHC manages the PWROFF pin (via API commands) when INIC is in the EHCI Attached State or the EHCI SemiProtected State.

Using the PS0/PS1 pins in this manner allows INIC to manage ECU power based on the current voltage level and the network state while INIC is in the EHCI Protected State (and the EHC is presumably unavailable). If the network is in the NetOn state, and the voltage level is above $V_{\text{low}}$, then INIC keeps the PWROFF pin asserted. If the network goes to the NetOff state, INIC releases PWROFF after $t_{\text{TimePwrOff}}$ expires. If the voltage level drops below the $V_{\text{Low}}$ threshold while INIC is in EHCI Protected State, INIC immediately releases the PWROFF pin.

Once the EHC is up and running and driving NetServices properly such that INIC is in EHCI Attached State, INIC keeps the PWROFF pin asserted regardless of the power level or network state. However, INIC continues to inform the EHC of those states, and the EHC can control power according to its own logic, using INIC.PMIState.Set(Control) to have INIC release the PWROFF pin when desired.

As mentioned in Section 3.7 “ECU Voltage Levels”, if the OEM specifies that the device must remain operational during brief excursions into the $V_{\text{Low}}$ region without powering down even while INIC is still in EHCI Protected State, then the INIC PS0/PS1 pins should be tied to ground so that INIC will not release the PWROFF pin. In this configuration, all power management logic is under EHC control. Note that with the PS0/PS1 pins grounded, INIC cannot notify the EHC about power level boundary crossings. However, the MPM85000 can be configured to interrupt the EHC directly when a power threshold is crossed.

The INIC is attached to the I2C bus used by the EHC. To support higher throughput to the MOST network, the EHC can also be connected to the MediaLB interface on INIC. If the selected EHC does not natively support MediaLB (see MediaLB Specification [11]), then an OS85650/2 (see OS85650/2 I/O Companion Chip Data Sheet [16]) can be used to translate an EHC parallel bus into MediaLB channels.

One of the benefits to using the MPM85000 is the ability to override stuck wakeup events (such as a Power-On switch or ECL shorted to ground) from the EHC to provide a more robust ECU architecture. Once a wakeup event is ignored, transitions on that event clear the override automatically. Another benefit of the MPM85000 is that the high-value, low-current resistors used to measure the battery voltage are on-chip thereby reducing component count and eliminating the need for conformal coating.
The communication section for a MOST50 system, shown in Figure 4-4, is very similar to the previous example, with the exceptions that MOST50 systems do not use an active PHY, and that the MOST50 INIC shown (OS81092) is not flash-memory based.

**FIGURE 4-4: MOST50 APPLICATION EXAMPLE - COMMUNICATIONS SECTION**
4.3 MPM85000 EHC POWER ON RESET

In the previous examples, the assumption was made that the EHC contains an internal power-on-reset circuit, or that the EHC POR was handled independently from INIC and the MPM85000. However, if the EHC requires an external POR and operates from the same power supply as INIC, the MPM85000 reset can be used, as illustrated in Figure 4-5. In this scenario diode isolation is required so that when either the INIC or EHC drives the reset of the other component, the driving component does not reset itself.

FIGURE 4-5: EHC POWER ON RESET FROM MPM85000
Appendix A. MOST Specifications and INIC

This section adds clarification to the MOST Specification 2.5 [1] and MOST Specification 3.0 [2] as they relate to different MOST network speeds and the INIC architectures.

The MOST Specification mentions sleep mode, which is defined as a low-power ECU state where the ECU is connected to the continuous battery power (BatConP) and most of the circuitry is powered off to minimize current draw. Within this document, sleep mode is referred to as the Sleep Power State. The current draw of each ECU in the Sleep Power State ($I_{STBY}$) is specified by the system integrator. The opposite of the Sleep Power State is defined as the Active Power State, in which part or all of the ECU is powered up and operational. For the MOST network to be operational and locked, all network ECUs must be in the Active Power State.

A.1 ECU WAKEUP VS. NETWORK STARTUP

Older MOST systems only used network activity to wakeup the network ECUs. In these systems, ECU wakeup and network startup occurred at approximately the same time. Newer MOST systems support other network wakeup mechanisms, such as ECL, where ECU wakeup can occur at a distinctly different time from actual network startup. In the MOST Specifications, the terms wakeup and startup are sometimes used interchangeably; however, they are both distinct phases of getting to an operational network state. Some MOST network timers start based on ECU wakeup, whereas others are based on MOST network startup. To clarify the situation, these terms are defined as follows:

- **Wakeup** is an ECU power transition from the Sleep Power State to the Active Power State.
- **Startup** is a MOST network NetInterface transition going from state Off (no network activity) to state Init (network activity exists).

In current MOST systems, signals other than network activity, such as ECL, wake an ECU from the Sleep Power State (wakeup) independent from network activity (startup). Figure A-1 illustrates a MOST150 wakeup from network activity. In this scenario, the ECU is in the Sleep Power State when network activity causes the network front-end STATUS line to assert (logic low), which wakes up the ECU (NwSwP enabled) causing the ECU to enter the Active Power State. Before the power supplies stabilize, the reset line is asserted (logic low), which blocks bypass mode in the Tx PHY. Once reset is released, the Tx PHY propagates the bypass mode network activity to the next ECU downstream causing it to wakeup (exit the Sleep Power State). Activity must propagate through every ECU in the network to achieve network lock. Once INIC has initialized, it exits bypass mode becomes a visible node in the network.
In Figure A-1, $t_{ON4}$ is part of the MOST150 oPHY Automotive Physical Layer Sub-Specification [8], $t_{INICINIT}$ is the initialization time of INIC once reset is released (see the appropriate INIC Hardware Data Sheets [4]), and $t_{PowerReset}$ is the ECU reset generator time from power good to reset released. The $t_{Wakeup}$ and $t_{WaitNodes}$ times are defined in the MOST Specification 3.0 [2].

MOST25 systems are similar with the exception that the optical transmitter (EOC) does not include a reset line; therefore, the bypass mode exists while in reset.

For MOST50 electrical systems that support network activity wakeup, the timing, shown in Figure A-2, is similar with the exception that INIC does not support a bypass mode; therefore, network activity is output on completion of the INIC $t_{INICINIT}$ time.

FIGURE A-2: MOST50 NETWORK ACTIVITY WAKEUP
A typical MOST system supports ECL wakeup, where ECL is used to wakeup the network ECUs, and the **Power Master** initiates network startup. Using ECL for wakeup provides redundancy to network activity and also provides a low-speed diagnostic path. Another benefit of ECL wakeup is that all ECUs power up at the same time, providing a faster overall network initialization than when using network activity alone (where each ECU only wakes up after the previous ECU transmits network activity). Figure A-3 depicts ECU wakeup via an ECL pulse (\(t_{EWU}\), see **MOST Electrical Control Line Specification** [3]). If the network activity reaching an ECU occurs after the completion of \(t_{INICINIT}\), then the time from RX activity to TX activity through an INIC, \(t_{NtwStartup}\) (defined in the **INIC Hardware Data Sheets** [4]) is much smaller than the INIC power up initialization time (generally an order of magnitude smaller). The ECL assertion could come from the **Power Master**, or from an **Power Slave** ECU that has qualified a local wakeup event.

**FIGURE A-3: MOST50/MOST150 ECL WAKEUP**
The previous figures focused on network startup; whereas, Figure A-4 illustrates network shutdown and the transition from the Active Power State to the Sleep Power State. For the MOST150 optical and coaxial PHYs, the Rx PHY off time and the Tx PHY off time are negligible, and in MOST50 electrical networks the PHY is passive. Therefore, the INIC time $t_{\text{NtwShutdown}}$ is approximately the MOST Specification time $t_{\text{shutdown}}$.

**FIGURE A-4: NETWORK SHUTDOWN**

![Network Shutdown Diagram]

A.2 NETWORK STARTUP

Typically the Power Master ECU wakes up and starts up the network based on a qualified local wakeup event, such as clamp status (key position switched to ON). This scenario is illustrated in Figure A-5, where the network achieves lock and the network operates in a typical fashion. Some Power Slave ECUs can also be designed to wake up and startup the network, based on their own local wakeup event.

**FIGURE A-5: NETWORK STARTUP**

![Network Startup Diagram]

If an INIC is triggered to startup the network, but the network fails to achieve lock within $t_{\text{config}}$ time, then the INIC NetInterface goes back to state off and reports an error to the EHC. As per the MOST Specification 3.0 [2] Section 3.1.5.1.2 Waking, the EHC must try and restart the network. As illustrated in Figure A-6, the first retry is mandatory, whereas the second and third retries are optional. The sequence used to startup the network, including retries, is defined as a startup sequence, and the number of attempts to startup the network within that sequence is defined as $N_{\text{NtwStartup}}$. Figure A-6 illustrates the use of ECL and network activity; however, ECL support is optional. In Figure A-6, the top signal (when high) indicates a local wakeup event is active.
INIC Hardware Concepts

The EHC qualifies this event and then initiates the startup sequence, which consists of trying to startup the network, and if the network doesn’t lock, the EHC tries to restart the network one to three more times. The local wakeup event that instigated the startup sequence need not be active for the entire sequence.

In Figure A-6, tConfig and tRestart times are defined in the MOST Specification 3.0 [2], and are managed by INIC once network startup is triggered.

**FIGURE A-6: NETWORK STARTUP SEQUENCE**

When the startup sequence finishes, if the network has not obtained lock and the local wakeup event is still active (i.e., key still in the ON position, engine running), the Power Master can initiate another startup sequence. The Power Master should continue to do so as long as the wakeup event is valid. The response to this situation and the determination of which local wakeup events allow the Power Master to continue initiating startup sequences is system integrator or OEM specific. Figure A-7 illustrates a Power Master network and startup scenario in which the local wakeup event is active for the start of three startup sequences. In this figure, the local wakeup event goes away sometime during the third startup sequence. Upon completion of the third startup sequence the EHC determines that the local wakeup event is no longer valid, so the EHC logs the error and shuts down. Prior to shutting down, the Power Master may optionally initiate an ECL System Test or Ring Break Diagnosis mode, the results of which can also be logged prior to shutdown.

**FIGURE A-7: POWER MASTER NETWORK STARTUP RETRIES**
Figure A-8 illustrates the flow for the Power Master, integrated with the ECL System Test. In this example, the system test is executed after all the network startup attempts fail.

FIGURE A-8: POWER MASTER NETWORK STARTUP FLOW

- **Wakeup Event**
- Initialize software
- start timer \( t_{\text{SwitchOffDelay}} \)
- Qualified
  - Wakeup event?
    - yes
      - ECL system test?
        - no
          - Voltage in \( U_{\text{Normal}} \)?
            - yes
              - ECL system test?
                - no
                  - Start up network
                    - Generate \( N_{\text{EWU}} \) pulses
      - Voltage in \( U_{\text{Normal}} \)?
        - no
          - Return to Sleep
- ECL system test?
  - no
    - Voltage in \( U_{\text{Normal}} \)?
      - yes
        - \( t_{\text{SwitchOffDelay}} \) timeout?
          - yes
            - Network Ready
          - no
            - Increment network startup attempts
  - no
    - Wakeup event still valid?
      - yes
        - Wakeup event
          - New Wakeup event qualified?
            - yes
              - Initiate ECL system test
            - no
              - Return to Sleep
          - ECL system test?
            - no
              - Voltage in \( U_{\text{Normal}} \)?
                - yes
                  - \( t_{\text{SwitchOffDelay}} \) timeout?
                    - yes
                      - Return to Sleep
                    - no
                      - Network startup attempts < \( N_{\text{Startup}} \)?
                        - yes
                          - Participate in ECL system test
                        - no
                          - Wakeup event
                            - \( t_{\text{SwitchOffDelay}} \) timeout?
                              - yes
                                - Return to Sleep
                              - no
                                - Wakeup event still valid?
                                  - yes
                                    - Initiate ECL system test
                                  - no
                                    - Return to Sleep
                                - yes
                                  - New Wakeup event qualified?
                                    - yes
                                      - Initiate ECL system test
                                    - no
                                      - Return to Sleep
                                  - no
                                    - Return to Sleep
                                - no
                                  - Return to Sleep
For *Power Slave* ECUs (i.e. every ECU in the network except the one containing the *Power Master*) with capability to wakeup from local events and start the network, the procedure is slightly different: *Power Slave* ECUs perform a startup sequence similar to the *Power Master* (as shown in Figure A-6); however, if the first startup sequence completes unsuccessfully (i.e. the network doesn’t achieve lock), then the *Power Slave* does not initiate another startup sequence even if the local wakeup event is still asserted. This rule prevents a *Power Slave* from draining the battery by constantly trying to startup a malfunctioning network. In Figure A-9 the *Power Slave* qualifies a local wakeup event, and initiates a startup sequence. Once the startup sequence finishes, the *Power Slave* logs the error even though the local wakeup event is still active.

**FIGURE A-9: POWER SLAVE NETWORK STARTUP RETRIES**

Most50 electrical systems do not support slave network activity wakeup (*INIC.NWStartup()*). In these systems, *Power Slaves* simply assert ECL low to wakeup the other network ECUs. Then the ECU containing the *Power Master* (which includes the network *Timing Master*) can startup the MOST network.
Figure A-10 illustrates the flow for the Power Slave that initiates network wakeup, integrated with the ECL System Test. In this example, the Power Slave only utilizes ECL to wake the Power Master. Once the Power Master wakes up, it starts the network as usual.

**FIGURE A-10: POWER SLAVE NETWORK STARTUP FLOW**

- **Wakeup Event**
  - Initialize software
  - start timer \( t_{PwrSwitchOffDelay} \)
  - Qualified wakeup event? (yes/no)
    - yes
      - go to Normal Network Slave
    - no
      - Local wakeup event? (yes/no)
        - yes
          - go to Normal Network Slave
        - no
          - Voltage in \( U_{Norm} \)?
            - yes
              - return to Sleep
            - no
              - \( t_{PwrSwitchOffDelay} \) timeout?
                - yes
                  - return to Sleep
                - no
                  - ECL system test?
                    - yes
                      - Start \( t_{SysSystOffDelay} \), Generate \( N_{Syst} \) pulses
                    - no
                      - Network startup attempts < \( N_{Startup} \)?
                        - yes
                          - Increment network startup attempts
                        - no
                          - Participate in ECL system test
                            - yes
                              - New wakeup event qualified?
                                - yes
                                  - \( t_{PwrSwitchOffDelay} \) timeout?
                                    - yes
                                      - return to Sleep
                                    - no
                                      - Local wakeup event?
                                        - yes
                                          - go to Normal Network Slave
                                        - no
                                          - \( t_{PwrSwitchOffDelay} \) timeout?
                                            - yes
                                              - return to Sleep
                                            - no
                                              - Restart network (MOST150)
                                                - yes
                                                  - Network wakeup event received?
                                                    - yes
                                                      - go to Normal Network Slave
                                                    - no
                                                      - \( t_{PwrSwitchOffDelay} \) timeout?
                                                        - yes
                                                          - return to Sleep
                                                        - no
                                                          - ECL system test?
                                                            - yes
                                                              - Network activity, or valid ECL start sequence
                                                            - no
                                                              - return to Sleep
                                                          - no
                                                            - Participate in ECL system test
                                                              - yes
                                                                - return to Sleep
                                                              - no
                                                                - Local wakeup event?
                                                                  - yes
                                                                    - go to Normal Network Slave
                                                                  - no
                                                                    - \( t_{SysSystOffDelay} \) timeout?
                                                                      - yes
                                                                        - return to Sleep
                                                                      - no
                                                                        - ECL should be monitored continuously in the background.

To get the Network interface states, NetServices should be running.
**INIC Hardware Concepts**

Figure A-11 illustrates the flow for a *Power Slave* that cannot initiate network wakeup, integrated with the *ECL System Test*. This ECU can only wake from the *Sleep Power State* due to *network wakeup events*.

**FIGURE A-11: POWER SLAVE WAKEUP FLOW**

A.3 EHC SOFTWARE RESET

In MOST150 systems, INIC enters bypass mode during reset. In the *MOST Specification 3.0 [2]*, when an ECU drops out of the network (goes into bypass mode), the ECU is constrained on when it can return back into the network (visible node) so that other ECUs can recognize that it exited the network. This time in bypass mode is defined as $t_{\text{Bypass}}$. Additionally, INIC requires some further time after reset for initialization, defined as $t_{\text{INICINIT}}$ (see *INIC Hardware Data Sheets [4]*). Therefore, if the EHC holds INIC in reset for a time period ($t_{\text{SwReset}}$), then the following constraint on the EHC reset time exists (see Equation A-1):

**EQUATION A-1:**

$$t_{\text{Bypass}}_{\text{Min}} < t_{\text{SwReset}} + t_{\text{INICINIT}} < t_{\text{Bypass}}_{\text{Max}}$$
The constraint described by Equation A-1 is illustrated in Figure A-12. In addition, the Tx PHY requires a reset at initial power-up which disables its network output while in reset. As previously mentioned, this power-on reset cannot come from the EHC. In addition, the EHC reset of INIC must be isolated from the Tx PHY reset; otherwise, the Tx PHY would disable its network output and not allow the bypass mode to propagate network activity through the network.

**FIGURE A-12: EHC INIC RESET (MOST25/MOST150)**

In MOST25 systems, INIC goes into bypass mode while in reset, similar to MOST150 systems. The equation above is also valid for MOST25 systems, but the bypass time $t_{\text{Bypass}}$ is different (see MOST Specification 2.5 [1] and MOST Specification 3.0 [2]).

In MOST50 electrical systems, INIC transmit output is disabled while in reset, as illustrated in Figure A-13. The reset-assert time ($t_{\text{SwReset}}$) should be as short as possible to minimize network startup and disruptions. Since asserting an INIC reset blocks network activity from propagating across the network, long reset assertion times can cause the network to shut down. Therefore, MOST50 $t_{\text{Bypass}}$ and the EHC reset of INIC, $t_{\text{SwReset}}$, must meet the following equations (see Equation A-2):

**EQUATION A-2:**

\[
\begin{align*}
&t_{\text{Bypass}}_{\text{Min}} < t_{\text{SwReset}} + t_{\text{INICINIT}} < t_{\text{Bypass}}_{\text{Max}} \\
\text{and} \\
&t_{\text{Bypass}}_{\text{Max}} < t_{\text{Unlock}}_{\text{Min}}
\end{align*}
\]

**FIGURE A-13: EHC INIC RESET (MOST50)**

For MOST50, $t_{\text{Bypass}}_{\text{Min}}$ is 1 ms (absorbed in the $t_{\text{INICINIT}}$ time), and $t_{\text{Bypass}}_{\text{Max}}$ is 26 ms. For MOST50, $t_{\text{SwReset}}_{\text{Max}}$ is 1 ms.
Appendix B. ECL Extensions

This section adds (does not replace) clarifications and extensions to the ECL protocol defined in the MOST Electrical Control Line Specification [3].

The ECL supports two separate protocols:

- An electrical wakeup (EWU) protocol for the network ECUs. Essentially a Network Wakeup Event triggered by EWU pulses on the ECL.

- An ECL System Test protocol that can be used by a test tool or an EHC to initiate tests and gather results from each ECU. Triggered by test start impulse (TSI) pulses on the ECL.

Both of these protocols are optional; support for either one must be defined by the system integrator.

An ECU connected to ECL can be either an initiator or a participant. For the ECL System Test, the initiator transmits the start sequence on ECL and indicates the test mode. Once the test is completed, the participants respond with their results (two bits per ECU).

The ECL protocols described in the following sections support the both MOST150 oPHY and cPHY systems with a maximum of 20 ECUs as well as MOST50 ePHY systems with up to 40 ECUs.

B.1 ECL START SEQUENCE

A device supporting both electrical wakeup and an ECL System Test is required to differentiate the two events. Electrical wakeup is defined as a pulse (or series of pulses) where the ECL is pulled low for time $t_{EWU}$. The start of a system test is triggered by a series of pulses of time $t_{TSI}$. To delineate between these two events, a generic start sequence is defined (see Figure B-1.)

FIGURE B-1: ECL START SEQUENCE

Typically an ECL monitoring software module is running in the background. If an ECL pulse is detected, then the module continues to watch for ECL pulses until $t_{SSend}$ is detected, which delineates the start sequence. When the end of the start sequence is detected, the module informs the application that it has received a valid start...
sequence containing either *electrical wakeup* (EWU) or *test start impulse* (TSI) pulses. A benefit of this approach is that the number of pulses need not be fixed in every module; thereby allowing the ECL initiator to change the number of pulses without having to change all the other ECU modules. Figure B-1 illustrates the start sequence detection at a participant node. The pulses are listed with generic times, $t_{SSID}$, since the determination of EWU or TSI has not been established yet.

When the application is informed that a valid EWU start sequence is detected, it initializes its network software and, if it is a *Power Slave*, it waits for network startup. For the *Power Master*, a valid EWU pulse indicates a qualified wakeup event (from a *Power Slave*) and the device starts the network normally.

If the application is informed that a valid TSI start sequence is detected, it must then block any *local wakeup events* and refrain from asserting ECL (per the ECL specification). For the *ECL System Test* (shown in Figure B-8 on page 61), the software module also informs the application, (at the end of $P_{Sync}$) which system test is starting. Then the application reacts appropriately (e.g. initiates RBD, or in the case of the *Power Master*, starts up the network for the stable lock and coding violation tests.

**FIGURE B-2: ECL START SEQUENCE - INITIATOR**

![ECL Start Sequence Diagram](image)

**B.2 NETWORK WAKEUP EVENTS**

For MOST50 systems, ECL is the primary method of waking all ECUs from the *Sleep Power State*, although supporting network activity as a *wakeup event* can add more robustness to the system design. For MOST150 systems, ECL wakeup allows all the ECUs to power up simultaneously providing a faster network readiness than with network activity alone.

A significant advantage of the INIC architecture is its ability to get the network operational without assistance from the EHC. Therefore, the network can power up independent of the *Power Slave* EHC initialization time. The only exception is the one ECU containing the *Power Master*, which must send a command to initiate network startup (since the *Power Master* EHC must qualify the *local wakeup event* before starting up the network).

**B.2.1 Power Master Wakeup**

Typically the ECU containing the *Power Master* wakes up the other network ECUs. When the *Power Master* gets a *wakeup event* that causes it to exit the *Sleep Power State* (and the power supply is in the proper region), then the *Power Master* wakes up all other network ECUs by pulling the ECL low for an appropriate amount of time ($t_{EWU}$), and starts up the network by sending the following INIC control message:

$$\text{EHC} \rightarrow \text{INIC.NWStartup.StartResult}()$$
The **Power Master** wakeup and network startup sequence is illustrated in Figure B-3. All INIC commands are described in detail in the INIC API User's Manuals [12]. When using NetServices, the INIC.NWStartup() method is called via the MostStartup() function.

For MOST50 networks the **Timing Master** must reside within the **Power Master** ECU; therefore, using the INIC.NWStartup() INIC function assumes that INIC is already configured as the network **Timing Master** ECU (INIC.DeviceMode()). When the network achieves normal operation (INIC.NIState.Status(NET_ON)), then INIC responds with the following result:

\[
\text{EHC} \leftarrow \text{INIC.NWStartup.Result()}
\]

and NetServices, sends the msval_state_changed(MSVAL_S_ON) callback.

**FIGURE B-3: POWER MASTER NETWORK WAKEUP/STARTUP**

When the ECL is pulled low, all other ECUs which are not the **Power Master** (i.e. all **Power Slave** ECUs), exit the Sleep Power State and each INIC starts its tTimePwrOff timer and pulls PWROFF low (which keeps the ECU awake for a minimum amount of time). If this timer expires with no network activity and before INIC enters the EHCU Attached State, then the **Power Slave** ECU reverts to the Sleep Power State. Under normal operation, network activity starts (NIState.Status(NET_INIT), at which time INIC turns off its tTimePwrOff timer and continues to assert PWROFF as long as network activity persists. When the EHC initializes and attaches to INIC via the NetServices initialization process, INIC also stops its tTimePwrOff timer. After the EHC initializes, it should start its tPwrSwitchOffDelay timer as stated in Section 3.8.1 “ECU Power Hold Strategy”.

If the MOST Specification 3.0 [2] time of tConfig plus tRestart expires without the network achieving lock, then INIC responds with the following message:

\[
\text{EHC} \leftarrow \text{INIC.NIState.Status(NET_OFF)}
\]

In the **Power Master** EHC, INIC also responds with the following error message:

\[
\text{EHC} \leftarrow \text{INIC.NWStartup.Error(Processing Error)}
\]

and NetServices sends the msval_error(ERR_STARTUP_FAILED) callback.

Since network startup failure is a very serious problem, if this error message is received by the **Power Master** EHC, then it should try and restart the network by pulling ECL low again (NEWU times), and then try and repeat the INIC.NWStartup() function, as illustrated in Figure B-4. For the **Power Master**, after a qualified local wakeup
In the event of a network startup attempt, the number of network startup attempts \( N_{\text{NtwStartup}} \) is system integrator specific, and is defined as a startup sequence. At the end of a startup sequence, if the network fails to lock, the Power Master can initiate a new startup sequence as long as the local wakeup event is still valid (e.g. key position, engine still running).

**FIGURE B-4: POWER MASTER NETWORK STARTUP RETRY**

For an individual startup sequence, the MOST Specification 3.0 [2] Section 3.1.5.1.2 Waking, requires at least one retry to startup the network \( (N_{\text{NtwStartup}}=2) \) and an optional two more retries \( (N_{\text{NtwStartup}}=4) \). The minimum EHC \( t_{\text{wrSwitchOffDelay}} \) time should be long enough to cover the delay between any of the timer start triggers stated in Section 3.8.1 "ECU Power Hold Strategy", plus any post-startup diagnostics. This timer is defined by the OEM/system integrator thereby ensuring all ECUs behave similarly across different vendors.

After the Power Master EHC has attempted to start a network a number of times \( (N_{\text{NtwStartup}}) \), if the wake condition is no longer valid, the Power Master can optionally initiate an ECL System Test (illustrated in Figure B-5). Having the Power Master automatically run a system test after failure of the network to lock and before powering down is useful to store error events for later diagnostics, or to display to the user before powering down (indication of error condition). The level of diagnostics support included in the Power Master is OEM/system integrator specific.

**FIGURE B-5: POWER MASTER WAKEUP SEQUENCE WITH SYSTEM TEST**
### B.2.2 Power Slave Wakeup

Generally, only a few Power Slaves support local wakeup events which, once qualified, allow the Power Slave to generate a network wakeup event; thereby waking all ECUs. In some systems, no Power Slaves support local wakeup events. As stated in the MOST Specification, any local event must be qualified before generating a network wakeup event. Qualified is defined as a valid event with a basis for waking up the other network ECUs. For example, a wireless call received that requires other network resources, or a debounced power on switch press constitute qualified local wakeup events. Whereas power supply fluctuations, a wireless call that does not need any other network resources, or glitches in local wakeup events are not considered qualified local wakeup events.

For MOST150 systems that support ECL, Power Slaves can support two network wakeup events: asserting ECL, and starting up the MOST network. Supporting both is a more robust approach as it adds redundancy. In addition, ECL assertion causes all the ECUs to power up (i.e. exit the Sleep Power State) simultaneously, thereby enabling faster network startup times. Using only network activity startup requires that each ECU powers up in series. MOST50 INIC devices do not support Power Slave startup (INIC.NWStartup()); therefore, asserting the ECL is the only option.

Network activity startup is described in detail in the MOST Specification. Figure B-6 illustrates a Power Slave wake up exclusively through the ECL. For the Power Slave to assert ECL, the local event is first qualified and the ECL must not already be asserted. Then the Power Slave asserts ECL causing all ECUs to wake from the Sleep Power State. The Power Slave then reasserts ECL ($N_{EWU} = 2$) so the Power Master can measure (i.e. qualify) the pulse. The Power Slave must assert enough EWU pulses to ensure the Power Master $t_{ECLDetect}$ time expires and the Power Master detects at least one valid ECL pulse. In the Figure B-6 example, the Power Master $t_{ECLDetect}$ time expires during the first $t_{Pause}$ time. Therefore, $N_{EWU}$ must be at least two. If $t_{ECLDetect}$ is greater than $t_{EWU} + t_{Pause}$, then $N_{EWU}$ would need to be three. Once the Power Master has initialized ($t_{PMInit}$) it begins the standard Power Master network start up sequence, as described in Section B.2.1 "Power Master Wakeup". If the Power Slave detects ECL asserted externally during subsequent retry attempts, it should assume the Power Master is starting the network and should abort its retry attempt.

### FIGURE B-6: POWER SLAVE NETWORK WAKEUP VIA ECL

![Power Slave Network Wakeup via ECL Diagram]
If the Power Slave does not detect another ECL pulse after $t_{PSW\_Retry}$ time, then the Power Slave should reassert $N_{EWU}$ wakeup pulses. The number of ECL assertions attempts by a Power Slave ECU is also limited to $N_{Net\_Startup}$. Unlike the Power Master, the Power Slave cannot exceed the maximum attempts regardless of the state of the local wake event (the MOST Specification limits the maximum number of retries to three). This restriction is required to keep the battery from draining in the event of a Power Slave repeatedly attempting to wake all ECUs.

### B.3 ECL SYSTEM TEST

Figure B-7 illustrates a typical network where two devices can be the system test initiator:

- The ECU with a diagnostic interface, connected to an external diagnostics tool. For this system test to work properly, the ECU connected to the diagnostics tool must be functioning properly. If this ECU contains the Power Master, it can also initiate the system test if the network fails to achieve lock.
- An external ECL test tool can also be connected directly to ECL through an ECL diagnostic connector. The benefit of this approach is that the ECU with the diagnostic interface is also tested.

**FIGURE B-7: EXTERNAL TOOL ECL SYSTEM TEST BLOCK DIAGRAM**
In addition to the methods listed above, a system test can also be triggered through FBlock Enhanced Testability (ET). Since the ECU acting as the test initiator receives the ET command(s) from across the network, the network must be locked to trigger the system test in this manner.

All ECUs must be capable of detecting the start of an ECL System Test from the Sleep Power State. Therefore, when an initiator initiates a system test from the Sleep Power State, the system test start sequence must be long enough to ensure the longest tECLDetect of any ECU can still detect a proper system test start sequence. When a tTSI pulse is recognized, all ECUs capable of asserting ECL, must block all network wakeup events (including assertion of ECL, even by the Power Master), while the system test is running. If the system test start sequence occurs after the Power Master has initiated network startup, then the Power Master continues to startup the network simultaneously to the system test sequence. Similarly, if the network is locked when a system test is initiated, the Power Master leaves the network locked through the system test.

**FIGURE B-8: ECL SYSTEM TEST SEQUENCE**

If the ECL System Test initiator is one of the ECUs and not an external test tool, the initiator also drives the test results out on ECL at the appropriate time during the results sequence.

### B.3.1 Parameter Sequence

The P1 to P5 parameters define the particular system test to execute. The rising edge of PSync indicates the start or trigger for the test. For the tests below, the On bit cannot be interpreted unless the En bit is set low indicating that the ECU is responding to a supported system test.

- **P[1:5] = 00000b**
  - MOST signal result, RBD test. The test must be completed within 7 s.
  
  If the network is off, all ECUs initiate Ring Break Diagnosis (RBD) on the rising edge of PSync within tDiagStart time. For the results sequence, the corresponding On bit is set to 0 if stable lock has been achieved on the network before the end of the RBD test. If network lock is achieved, then the Power Master initiates network shutdown after tTestPause.
  
  If the network is in NetInterface Init or locked, then the corresponding On bit is set to 0 if stable lock existed at the rising edge of PSync or occurred anytime during tTestPause.

- **P[1:5] = 10000b**
  - Only alive result. The test must be completed within 100 ms.
  
  Regardless of the network state, the participating device sets the corresponding En bit to 0, thereby indicating that it received a proper system test parameter sequence. The On bit is not used, and remains at 1 for this test.
P[1:5] = 01000b
MOST signal result, coding error test. Test must be completed within 7 s.
If the network is off, the Power Master initiates network startup on the rising edge of P\textsubscript{Sync} (Power Master could be system test initiator or participant). Then all ECUs start counting coding errors from stable lock to the end of t\textsubscript{TestPause}. For the results sequence, the On bit is set to 0 if a stable lock has been achieved and the number of coding errors detected were lower than a predefined threshold (system integrator specific). If the network never leaves the NetInterface Off state for the duration of t\textsubscript{TestPause} or does not achieve stable lock, the corresponding On bit is set to 1. If network lock is achieved, then the Power Master initiates network shutdown after t\textsubscript{TestPause}.
If the network is in NetInterface state Init or Normal Operation, all ECUs start counting coding errors from the rising edge of P\textsubscript{Sync} to the end of t\textsubscript{TestPause}. For the results sequence, the On bit is set if a stable lock has been achieved and the number of coding errors detected were lower than a predefined threshold (system integrator specific).

P[1:5] = 11000b
MOST signal result, SSO/CU result. Test must be completed within 100 ms. (MOST150 only)
For the results sequence, the corresponding On bit is set to 0 if the INIC.SSOSResult() function returns NoResult or NoFaultSaved. The On bit is set to 1 if the function returns SuddenSignalOff or CriticalUnlock.

P[1:5] = 00100b
MOST signal result, stable lock test. Test must be completed within 2.5 s.
If the network is off, the Power Master initiates network startup on the rising edge of P\textsubscript{Sync} (Power Master could be system test initiator or participant). For the results sequence, the corresponding On bit is set to 0 if a stable lock has occurred anytime during t\textsubscript{TestPause}. If the network never leaves the NetInterface Off state for the duration of t\textsubscript{TestPause}, the corresponding On bit is set to 1. If network lock is achieved, then the Power Master initiates network shutdown after t\textsubscript{TestPause}.
If the network is in NetInterface Init or Normal Operation, then the corresponding On bit is set to 0 if stable lock existed at the rising edge of P\textsubscript{Sync} or occurred anytime during t\textsubscript{TestPause}.

B.4 ECL DETECTION SCENARIOS

In MOST network systems supporting a Sleep Power State, the system integrator must decide how fast an ECU must exit the Sleep Power State and have the EHC measure the different ECL assertion pulse widths supported in the system. The time from exiting the Sleep Power State to the EHC being able to start measuring ECL is defined as t\textsubscript{ECLDetect} and includes:
• Hardware detection of ECL assertion (including any receiver glitch protection circuitry)
• EHC power supply ramp time, if the EHC was powered down during the Sleep Power State
• Reset deassertion, if the EHC power supply was off in the Sleep Power State
• EHC firmware initialization time

All previous figures illustrate only one ECL pulse per event; however, the system integrator could decide to use multiple pulses per event, as illustrated in Figure B-9. The first three scenarios rely on the EHC being able to measure an ECL event pulse width, and support all relevant ECL pulse widths. The fourth scenario (D) relies on hardware glitch protection and the EHC doesn’t measure the wakeup pulse.
If an ECL System Test is initiated from the Sleep Power State, the system test start sequence (tSTSS) must be long enough (N TSI pulses) to provide at least one valid TSI pulse after the longest tECLDetect time (tECLDetectmax) of any networked ECU. Assuming tECLDetect starts from the first falling edge of ECL (see Equation B-1):

**EQUATION B-1:**

\[ t_{STSS} > t_{ECLDetect_{max}} + t_{TSI} + t_{SSEnd} \]

Or, knowing the longest tECLDetect time (see Equation B-2):

**EQUATION B-2:**

\[ N_{TSI} > \frac{t_{ECLDetect_{max}}}{t_{TSI} + t_{Pause}} + 1 \]

Using the example illustrated in Figure B-10 where tECLDetectmax is between 950 ms and 1150 ms, N_{TSI} would have to be 5 to guarantee that every ECU detected a full TSI pulse.

---

**FIGURE B-9: ECL DETECTION SCENARIOS**

A: EHC can partially measure (tECL_Delta) the smallest ECL pulse. EHC knows how long tECLDetect is and assumes ECL low for that portion.

B: The EHC isn’t ready until after the initial wake pulse. Each event requires two ECL pulses. EHC measures the second pulse.

C: Hardware doesn’t wake until the rising edge, therefore the first pulse is missed. Each event requires two ECL pulses. EHC measures the second pulse.

D: EHC does not measure the wake pulse (power slaves only); relies on receiver hardware glitch protection. System test start sequence must be longer than the longest tECLDetect in the system.
B.5 ECL ROBUSTNESS

Since the ECL is used in automotive environments, certain robustness requirements are generally specified by the system integrator:

- The ECL must work above and below the battery voltage entering the ECU. This is due to voltage shifts on both the power and ground supply lines. These excursions beyond the ECU power supply are illustrated in Figure B-13 and in the ECL receiver specifications for $V_{ECL_{IH}}$ and $V_{ECL_{IL}}$.

- The ECL receiver should contain glitch protection, for the specified time $t_{Glitch}$, so that it does not exit the *Sleep Power State* due to noise on the ECL.

- The ECL transmitter must not be damaged by a short-to-power ($V_{BATTERY}$). The ECL transmitter circuit typically handles this via multiple mechanisms:
  - The ECL must limit the transmitter current, as specified in $I_{ECL_{SC}}$.
  - The ECL circuit could contain a thermal shutdown which disables the ECL transmitter when the ECL circuit gets too hot.
  - The ECU must also monitor the ECL receiver while transmitting. The ECL circuit could also contain error detection that stops asserting the ECL transmitter when the ECL receiver indicates that ECL is still high (shorted to power). This error should be recognized by the EHC and reported through the diagnostics reporting mechanisms (e.g. MOST, CAN, etc).

- The ECL must be able to handle short-to-ground. A short-to-ground does not have the potential to damage an ECL transmitter, as a short-to-power can. Handling short-to-ground conditions involves the detection of the condition $t_{ECL_{Low}}$, and the ability of the ECU to revert to the *Sleep Power State* when the MOST network shuts down. This error should be recognized by the EHC and reported through the diagnostics reporting mechanisms (e.g. MOST, CAN, etc) before the network shuts down. If short-to-ground detection is supported, then the following should also be supported:
  - If the EHC reverts to the *Sleep Power State* during a short-to-ground fault, the EHC must recover from a short-to-ground condition automatically (in hard-
ware), as this could be the only ECU waking mechanism. If network activity wakeup is not supported, then this requirement is mandatory.

- To minimize $I_{ECL}$ current during the fault condition, a larger ECU pull up resistor value ($R_{PU}$) is used during the Sleep Power State.

• ECL transmitter assertion time out.
If the EHC drives ECL low and never releases (code fault), then the ECL transmitter times out and releases the ECL to allow others to transmit. The timeout must be set larger than the maximum allowable ECL assertion pulse (but shorter than the $t_{ECL,Low}$ short-to-ground time).

B.6 ERRANT ECL PULSES

For the ECL System Test, reaction to errant ECL pulses is described in the MOST Electrical Control Line Specification [3]. However, reaction to errant ECL wakeup pulses ($t_{EWU}$) is complicated since the INIC chip manages network activity independent of the EHC. In general, network activity should take precedence over errant ECL pulses.

For Power Slaves which are not waking from a local wakeup event, an ECL pulse wakes the EHC and INIC from the Sleep Power State. The EHC measures the ECL pulse to determine which type of wake event occurred. However, once in the Active Power State, INIC manages network activity independent of the EHC. Upon exiting the Sleep Power State, INIC starts its backup $t_{TimePwrOff}$ timer. If network activity is not received before the timer expires, INIC indicates it is ready to revert to the Sleep Power State. If network activity occurs before the timer expires, then INIC tries to lock to the network and resets its $t_{TimePwrOff}$ timer. INIC also informs the EHC of these events. When the INIC enters the EHCI Attached State, INIC stops its $t_{TimePwrOff}$ timer and the EHC starts its $t_{PwrSwitchOffDelay}$ timer.

If the wakeup ECL pulse measured by the EHC is not valid, but the network starts up, then the EHC should log the errant ECL pulse, forward the error to any available diagnostic reporting mechanisms, and assume a normal wakeup occurred (network activity takes precedence over errant ECL pulses). Similarly, if the network starts up and an ECL short-to-ground is detected, then the EHC should log and forward the error to the diagnostic reporting mechanism(s). In either case, the network should stay active until the Power Master determines that network shutdown should occur.

Power Masters woken from ECL react similarly in systems where Power Slaves waking from a local event pull ECL low while initiating network startup. In the Power Master, if the network starts up after receiving an errant ECL pulse, then the network activity takes precedence, so the EHC should log the errant ECL pulse, forward the error to the diagnostic reporting mechanism(s), and assume a normal wakeup occurred. In systems where Power Slaves do not (or cannot) startup the network, then Power Masters should ignore errant ECL pulses and not start the network (and revert back to the Sleep Power State instead).

For all ECUs, the OEM may support a second EHC $t_{PwrSwitchOffDelay}$ time for ECL wakeup separate from normal network activity. This second timer allows the ECU to revert to the Sleep Power State in a shorter period of time if an errant ECL pulse wakes the ECU but no network activity is received.
B.7 ECL CIRCUIT CHARACTERISTICS

This section describes the electrical characteristics of the ECL. To support the ECL System Test, the ECL must be a bi-directional line for all ECUs, as illustrated below in Figure B-11. Note that Figure B-11 is a generic circuit for discussion purposes; actual implementations can vary greatly.

FIGURE B-11: GENERIC ECL CIRCUIT (BI-DIRECTIONAL)

In Figure B-11, the voltage potential at the ECU connector is defined as $V_{BAT_ECU}$. This voltage can be shifted from the car battery voltage ($V_{BATTERY}$) due to impedances in the wiring harness. Generally ECUs also contain load dump and reverse voltage protection between the connector and the internal supply as mentioned previously (Section 3.6.1 “Load Dump and EMI Filtering”). This circuitry causes a voltage drop, $V_{LoadDump}$, between the ECU power connector and the protected internal supply ($V_{SUPPLY}$) which powers the ECL circuitry.

Note: In relation to Chapter 3 and Chapter 4, $V_{BAT_ECU}$ in Figure B-11 is equivalent to the BatConP supply, and $V_{SUPPLY}$ is equivalent to ProConP.
B.8 ECL ELECTRICAL CHARACTERISTICS

The transmitter ECL output (Figure B-12) voltages generally do not extend all the way to the ECU’s power supply and ground values due to internal protection circuitry. Also, these output values are further reduced relative to the battery due to impedances in the supply wiring to the ECU.

Figure B-12: ECL TRANSMITTER WAVEFORM

Figure B-13 illustrates the ECL receiver waveform with a worst-case maximum ECL received signal, which can extend outside both the ECU power and ground references. These excursions beyond the ECU power supply range can occur when the ECL transmitting device is close to the car battery and the receiving ECL circuitry is far away from the car battery thus adding considerable voltage drop in its supply lines. The ECL receiver circuitry must be capable of handling these voltage extremes.

Figure B-13: ECL RECEIVER WAVEFORM

Table B-1 lists global specifications for the entire system, not just one ECU device.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECL total line capacitance</td>
<td>$C_{ECL}$</td>
<td>20</td>
<td></td>
<td>nF</td>
<td></td>
</tr>
<tr>
<td>ECL total line pull-up resistance</td>
<td>$R_{ECL}$</td>
<td>1</td>
<td></td>
<td>30 kΩ</td>
<td></td>
</tr>
<tr>
<td>ECU voltage shift relative to the battery.</td>
<td>$V_{BAT_SHIFT}$</td>
<td>0</td>
<td></td>
<td>0.115 $V_{BAT_ECU}$</td>
<td>V</td>
</tr>
<tr>
<td>ECU ground shift relative to the battery.</td>
<td>$V_{GND_SHIFT}$</td>
<td>0</td>
<td></td>
<td>0.115 $V_{BAT_ECU}$</td>
<td>V</td>
</tr>
</tbody>
</table>
Table B-2 lists specifications for an individual ECU device. All specifications listed below could be modified by the system integrator.

### TABLE B-2: ECU ECL SPECIFICATIONS

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECU connector voltage</td>
<td>$V_{BAT_{ECU}}$</td>
<td>$V_{Low_{Th}}$</td>
<td></td>
<td></td>
<td>V</td>
<td>(Note 1)</td>
</tr>
<tr>
<td>ECL circuit voltage</td>
<td>$V_{SUPPLY}$</td>
<td>$V_{Low_{Th}} - 1$</td>
<td></td>
<td></td>
<td>V</td>
<td>(Note 1)</td>
</tr>
<tr>
<td>$I_{BAT_{ECU}}$</td>
<td>$I_{STBY}$</td>
<td>100</td>
<td></td>
<td></td>
<td>µA</td>
<td>$V_{ECL} \geq V_{SUPPLY}$</td>
</tr>
<tr>
<td>ECL capacitance</td>
<td>$C_{ECU}$</td>
<td>300</td>
<td></td>
<td></td>
<td>pF</td>
<td></td>
</tr>
<tr>
<td>Pull up resistor, Active Power State</td>
<td>$R_{PU}$</td>
<td>20</td>
<td>30</td>
<td>60</td>
<td>kΩ</td>
<td></td>
</tr>
<tr>
<td>Pull up resistor, Sleep Power State</td>
<td>$R_{PU_{S}}$</td>
<td>700</td>
<td></td>
<td></td>
<td>kΩ</td>
<td>fault condition</td>
</tr>
<tr>
<td>ECL loss of power</td>
<td>$I_{ECL_{NO_{BAT}}}$</td>
<td></td>
<td></td>
<td>±100</td>
<td>µA</td>
<td>$V_{BAT_{ECU}} = V_{BAT_{GND}}$ $V_{ECL} = V_{Super_{Th}}$</td>
</tr>
<tr>
<td>ECL loss of ground</td>
<td>$I_{ECL_{NO_{GN}}}$</td>
<td></td>
<td></td>
<td>±1000</td>
<td>µA</td>
<td></td>
</tr>
<tr>
<td>Load dump voltage drop</td>
<td>$V_{LoadDump}$</td>
<td></td>
<td></td>
<td>1</td>
<td>V</td>
<td>Active Power State</td>
</tr>
<tr>
<td>ECL pull-up diode voltage drop</td>
<td>$V_{DPU}$</td>
<td></td>
<td></td>
<td>1</td>
<td>V</td>
<td>$I_{ECL}$ maximum</td>
</tr>
</tbody>
</table>

#### ECL Receiver:

$V_{ECL}$ receiver inactive/recessive state (high) | $V_{ECL_{IH}}$ | $0.62 \times V_{SUPPLY}$ | | | V | maximum $\leq V_{Super_{Th}}$ |

$V_{ECL}$ receiver active/dominant state (low) | $V_{ECL_{IL}}$ | $-0.12 \times V_{BAT_{ECU}}$ | | | V | |

$V_{ECL}$ receiver glitch protection | $t_{Glitch}$ | | | 50 | µs | Sleep Power State |

#### ECL Transmitter:

$V_{ECL}$ transmitter inactive/recessive state (high) | $V_{ECL_{OH}}$ | $V_{SUPPLY} - 1$ | | | V | Active Power State |

$V_{ECL}$ transmitter active/dominant state (low) | $V_{ECL_{OL}}$ | $V_{ECU_{GND}}$ | | | V | $I_{ECL}$ minimum |

ECL fall time | $t_{fall}$ | | | 1 | ms | 0.8 $\times V_{ECL}$ to 0.2 $\times V_{ECL}$, $R_{ECL}$ minimum, and $C_{ECL}$ maximum |

ECL rise time | $t_{rise}$ | | | 2 | ms | 0.2 $\times V_{ECL}$ to 0.8 $\times V_{ECL}$, $R_{ECL}$ maximum, and $C_{ECL}$ maximum |

ECL current | $I_{ECL}$ | | | 20 | mA | $V_{ECL_{OL}} \leq 0.2 \times V_{SUPPLY}$ |

ECL short-circuit current | $I_{ECL_{SC}}$ | | | 200 | mA | $V_{ECL} = V_{BAT_{ECU}}_{max}$ |

**Note 1**: Whether the ECU must operate above $U_{Super}$ is system integrator specific.

**2**: Current in the *Sleep Power State* can vary by ECU type and must be specified by the system integrator for each ECU type.

**3**: System integrator defines whether optional or not. Since ECL asserted low typically keeps an ECU in the *Active Power State*, to support this value, the ECU must have the ability to override the wake on ECL low condition, and revert to the *Sleep Power State*. If supported, the ECU must be able to cancel the override when ECL ceases to be stuck low (rising edge detected).

**4**: Minimum transition times (slew-rate limiting) may be specified by the system integrator to minimize EMI.
B.9 ECL TIMING DEFINITIONS

All devices (EHCs or test tools) which can drive ECL (initiators) must first make sure that ECL is high before driving ECL low. If ECL is already low, then the Power Master EHC must not drive low until ECL is high for \( t_{\text{Pause}} \). In a Power Slave, it is system-integrator-specific as to whether the EHC waits similarly to the Power Master, or cancels its attempt to drive ECL since it is already being driven.

All EHCs must continually monitor the ECL receive line, even when transmitting. Another ECU could be driving the line simultaneously, which must be seen by all ECUs (even the transmitting one).

### TABLE B-3: EHC ECL TIMING SPECIFICATIONS

<table>
<thead>
<tr>
<th>Description</th>
<th>Name</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network startup (wakeup) attempts (^\d)</td>
<td>( N_{\text{NetW.Startup}} )</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>--</td>
</tr>
<tr>
<td>( \text{Power Master} )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \text{Power Slave} )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECL short-to-ground detection</td>
<td>( t_{\text{ECL.Low}} )</td>
<td>1</td>
<td></td>
<td></td>
<td>s</td>
</tr>
<tr>
<td>( \text{Power Master} ) time from receiving a wakeup event from the Sleep Power State to being able to assert ECL and start the network.</td>
<td>( t_{\text{PMInit}} )</td>
<td></td>
<td></td>
<td></td>
<td>ms</td>
</tr>
<tr>
<td>( \text{Power Slave} ) time from receiving a wakeup event from the Sleep Power State to being able to assert ECL, if supported.</td>
<td>( t_{\text{PSInit}} )</td>
<td></td>
<td></td>
<td></td>
<td>ms</td>
</tr>
<tr>
<td>( \text{Power Slave} ) wakeup retry time. Time between network wakeup attempts if network activity or ECL assertion by another ECU does not occur.</td>
<td>( t_{\text{PSW_Retry}} )</td>
<td></td>
<td></td>
<td></td>
<td>ms</td>
</tr>
<tr>
<td>Time from inactivity to reverting to the Sleep Power State. (^\d)</td>
<td>( t_{\text{PwrSwitchOffDelay}} )</td>
<td>5</td>
<td>20</td>
<td></td>
<td>s</td>
</tr>
<tr>
<td>INIC Configuration String backup timer (PMICfg.Time-PwrOff) to keep the ECU in the Active Power State if the EHC is malfunctioning.</td>
<td>( t_{\text{TimePwrOff}} )</td>
<td></td>
<td>60</td>
<td></td>
<td>s</td>
</tr>
<tr>
<td>Time from receiving a wakeup event from the Sleep Power State to being able to measure the ECL pulse. ( t_{\text{ECLDetect}} \leq t_{\text{PMInit}} ) or ( t_{\text{PSInit}} )</td>
<td>( t_{\text{ECLDetect}} )</td>
<td></td>
<td></td>
<td></td>
<td>ms</td>
</tr>
<tr>
<td>( \text{Power Master} )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \text{Power Slave} )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Longest ( t_{\text{ECLDetect}} ) time of any ECU in the network.</td>
<td>( t_{\text{ECLDetect}_{\text{max}}} )</td>
<td></td>
<td></td>
<td></td>
<td>ms</td>
</tr>
<tr>
<td>( \text{ECL System Test} ) start sequence time.</td>
<td>( t_{\text{STSS}} )</td>
<td></td>
<td></td>
<td></td>
<td>ms</td>
</tr>
</tbody>
</table>

**Note 1:** The maximum number of network wakeup/startup attempts by the Power Master is system-integrator-specific, but is generally based on how long the local wakeup event remains asserted. For example, if the car engine is running (clamp status), the Power Master may continuously try to start the network. For the Power Master, the maximum value assumes that the qualified local wake event deasserts before the maximum number was reached. For Power Slave ECUs, the number of retries is limited so as not to drain the battery.

**2:** System-integrator-specific. Longer times relax the requirements on the EHC and its initialization code at the expense of longer network startup times.

**3:** Section 3.8.1 “ECU Power Hold Strategy” defines when the EHC should start and stop this timer.

**4:** Equal to \( N_{\text{TSI}} \times t_{\text{TSI}} + (N_{\text{TSI}} - 1) \times t_{\text{Pause}} + t_{\text{SSEnd}} \)
### TABLE B-3: EHC ECL TIMING SPECIFICATIONS (CONTINUED)

<table>
<thead>
<tr>
<th>Description</th>
<th>Name</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indicates the end of the ECL start sequence. Must be at least 1.5(t_{\text{pause}}) to discern between the two.</td>
<td>(t_{\text{SSEnd}})</td>
<td>150</td>
<td></td>
<td></td>
<td>ms</td>
</tr>
<tr>
<td>Generic term for number of ECL start pulses (either TSI or WI). Used in the ECL “start sequence participant” software module. Is either (N_{\text{TSI}}) or (N_{\text{EWU}}).</td>
<td>(N_{\text{SSID}})</td>
<td>1</td>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td>Number of ECL System Test start pulses. Indicates how many system test start pulses are used during the system test start sequence. For system test initiators, (N_{\text{TSI}}) must be longer than the longest ECU (t_{\text{ECLDetect}}) time to guarantee that all ECUs detect a proper ECL System Test start sequence.</td>
<td>(N_{\text{TSI}})</td>
<td>1</td>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td>Number of electrical wakeup start pulses. Indicates how many EWU start pulses are used during the electrical wakeup start sequence.</td>
<td>(N_{\text{EWU}})</td>
<td>1</td>
<td>2</td>
<td></td>
<td>--</td>
</tr>
</tbody>
</table>

**Note 1:** The maximum number of network wakeup/startup attempts by the Power Master is system-integrator-specific, but is generally based on how long the local wakeup event remains asserted. For example, if the car engine is running (clamp status), the Power Master may continuously try to start the network. For the Power Master, the maximum value assumes that the qualified local wake event deasserts before the maximum number was reached. For Power Slave ECUs, the number of retries is limited so as not to drain the battery.

**2:** System-integrator-specific. Longer times relax the requirements on the EHC and its initialization code at the expense of longer network startup times.

**3:** Section 3.8.1 “ECU Power Hold Strategy” defines when the EHC should start and stop this timer.

**4:** Equal to \(N_{\text{TSI}} \times t_{\text{TSI}} + (N_{\text{TSI}} - 1) \times t_{\text{pause}} + t_{\text{SSEnd}}\)
### TABLE C-1: GLOSSARY

<table>
<thead>
<tr>
<th>Term</th>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active Power State</td>
<td></td>
<td>ECU state in which the device is connected to a continuous battery supply and fully active (i.e. EHC and INIC powered). This state does not automatically indicate that the MOST network is operational. The opposite of the Active Power State is the Sleep Power State.</td>
</tr>
<tr>
<td>Application Switched Power</td>
<td>ApSwP</td>
<td>Power management conceptual power supply net. This optional power block can be implemented in high power consumption ECUs to power the application circuitry separately from the EHC/INIC power. Power to this block is controlled by the Switched-Application Signal (SA).</td>
</tr>
<tr>
<td>Battery Continuous Power</td>
<td>BatConP</td>
<td>Power management conceptual power supply net which is always connected directly to the battery. This net can have different names based on where the signal is viewed. BatConP is V_BATTERY at the battery terminals, and V_BAT_ECU at an ECU power connector; the difference being the voltage drop that occurs in the wiring between the two points.</td>
</tr>
<tr>
<td>Battery Switched Power</td>
<td>BatSwP</td>
<td>Power management conceptual power supply net which is controlled by the Power Master ECU and provides power to all other network ECUs. In this scenario, the Power Master ECU must be on BatConP.</td>
</tr>
<tr>
<td>Bypass Mode</td>
<td></td>
<td>INIC state where the incoming network data is bypassed directly to the TX output of INIC. While in bypass mode, INIC is invisible to the network and cannot transmit any data. The electrical MOST50 INIC does not support bypass mode. The opposite of bypass mode is a visible node.</td>
</tr>
<tr>
<td>Clamp Status</td>
<td></td>
<td>Indication of the ignition key position (off, accessory, on).</td>
</tr>
<tr>
<td>Coaxial Physical Layer</td>
<td>cPHY</td>
<td>Coax network cable used in MOST150 networks.</td>
</tr>
<tr>
<td>Coaxial to Electrical Converter</td>
<td>CEC</td>
<td>Devices designed for MOST150 networks using a coax PHY layer to receive coax signals and convert them to electrical signals for INIC to use. Includes circuitry for supporting a Sleep Power State.</td>
</tr>
<tr>
<td>Coaxial Transceiver</td>
<td>CTR</td>
<td>Standard MOST network connector unit for a MOST150 cPHY network connection consisting of a CEC and ECC.</td>
</tr>
<tr>
<td>Continuous Power</td>
<td>ContIP</td>
<td>Power management conceptual power supply net. Continuous power available for a small portions of the design that remain operational (for wakeup event detection) during the Sleep Power State.</td>
</tr>
<tr>
<td>Controller Area Network</td>
<td>CAN</td>
<td>A vehicle bus standard designed to allow microcontrollers and devices to communicate with each other within a vehicle without a host computer.</td>
</tr>
<tr>
<td>ECL System Test</td>
<td></td>
<td>ECL System Test is an ECL protocol that defines using the ECL for purposes of test/diagnosis as well as results reporting. An ECL System Test is the preferred method of performing Ring Break Diagnosis.</td>
</tr>
<tr>
<td>EHC Interface State</td>
<td>EHCI</td>
<td>States that define the level of EHC interaction with the INIC and MOST network: - The EHCI Attached State is the normal mode operation in which the EHC has full access to INIC and the MOST network. - In the EHCI Protected State, the EHC is inaccessible to the network and EHC communication with the INIC is limited. - The EHC SemiProtected State is a transitional state in which the INIC and EHC can fully communicate; however, the EHC is still inaccessible to the MOST network.</td>
</tr>
<tr>
<td>Electrical to Coaxial Converter</td>
<td>ECC</td>
<td>Devices designed for MOST150 networks using a coax PHY layer to convert INIC electrical signals into coax signals for transmission on the MOST network.</td>
</tr>
<tr>
<td>Electrical Control Line</td>
<td>ECL</td>
<td>A wire-OR’ed unshielded cable attached to every ECU. Defined by the MOST Cooperation in the MOST Electrical Control Line Specification [3] and used for either electrical wakeup or system test protocol.</td>
</tr>
<tr>
<td>Electrical Optical Converter</td>
<td>EOC</td>
<td>Produces a digital signal in the electrical domain from an incoming optical signal.</td>
</tr>
</tbody>
</table>
**TABLE C-1: GLOSSARY (CONTINUED)**

<table>
<thead>
<tr>
<th>Term</th>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Electrical Physical Network</td>
<td>ePHY</td>
<td>Unshielded twisted pair network cable used in MOST50 networks. ePHY network standards can be found in the MOST Electrical Physical Layer Specification [6].</td>
</tr>
<tr>
<td>Electrical Wakeup</td>
<td>EWU</td>
<td><em>Electrical Wakeup</em> is an ECL protocol that defines using the ECL for purposes of generating a Network Wakeup Event. $EWU$ is the duration of the ECL assertion and $NEWU$ is the number of ECL assertions.</td>
</tr>
<tr>
<td>Electronic Control Unit</td>
<td>ECU</td>
<td>Also known as a “device” in the MOST Specification 3.0 [2]. An entire box or unit consisting of an INIC, EHC, power management circuitry and application circuitry.</td>
</tr>
<tr>
<td>External Host Controller</td>
<td>EHC</td>
<td>Microcontroller or microprocessor that manages the ECU and defines what applications exist.</td>
</tr>
<tr>
<td>Function Block</td>
<td>FBlock</td>
<td>Logical group of functions (commands) that are related. For example, an FBlock Tuner contains all functions associated with the radio tuner hardware. FBlocks are required to support common functions for plug-and-play operation. In addition, all ECUs are required to support some standard FBlocks such as NetBlock (also used for plug-and-play operation/system enumeration). Some network system services also reside in FBlocks, such as the FBlock NetworkMaster. See the Device Model in the MOST Specification 3.0 [2] Section 2.1.2 for additional information.</td>
</tr>
<tr>
<td>Fiber Optic Receiver</td>
<td>FOR</td>
<td>Also known in the MOST Specification as optical electrical converter, OEC. Devices designed for MOST networks using an optical PHY layer to receive optical signals and convert them to electrical signals for INIC to use. Includes circuitry for sleep mode support.</td>
</tr>
<tr>
<td>Fiber Optic Transceiver</td>
<td>FOT</td>
<td>Standard MOST network connector unit for an oPHY MOST network consisting of an FOR (OEC) and FOX (EOC).</td>
</tr>
<tr>
<td>Fiber Optic Transmitter</td>
<td>FOX</td>
<td>Also known in the MOST specification as electrical optical converter, EOC. Devices designed for MOST networks using an optical PHY layer to convert INIC electrical signals to optical signals.</td>
</tr>
<tr>
<td>Intelligent Network Interface Controller</td>
<td>INIC</td>
<td>Manages all time critical low-level network functions to off load the EHC and provide a more stable network. Protects the network from errant EHC code. INIC devices have been developed for all MOST speed grades and provide simple migration paths between grades.</td>
</tr>
<tr>
<td>Local Interconnect Network</td>
<td>LIN</td>
<td>A low-cost low-speed automotive network managed by the LIN Consortium (<a href="http://www.lin-subbus.org">www.lin-subbus.org</a>). LIN transceivers can be used to implement the MOST Electrical Control Line (ECL).</td>
</tr>
<tr>
<td>Media Local Bus</td>
<td>MediaLB or MLB</td>
<td>Board-level high-speed bus that connects INICs to EHCs and other peripherals that can carry all MOST data types.</td>
</tr>
<tr>
<td>Media Oriented Systems Transport</td>
<td>MOST®</td>
<td>High-speed networks, designed for automotive use, that efficiently carry streaming data (audio/video), network control data and packet data. MOST150 also carries multiple types of isochronous data as well as Ethernet packet data. The MOST standard is managed by the MOST Cooperation (<a href="http://www.mostcooperation.com">www.mostcooperation.com</a>). MOST25 - is a 25 Mbit/s automotive bus MOST50 - is a 50 Mbit/s automotive bus MOST150 - is a 150 Mbit/s automotive bus that supports extra data types</td>
</tr>
<tr>
<td>NetInterface State</td>
<td></td>
<td>State of the INIC physical layer with respect to the rest of the network. Defined in the MOST Specification 3.0 [2] NetInterface Section 3.1.2.2. - The Normal Operation state means the INIC is communicating normally. - During the Off state, there is no modulated signal at the device output. - During the Init state, the INIC is initializing and is able to communicate with other nodes. - During Ring Break Diagnosis state, errors in the network can be localized. - The Diagnosis Result state is an optional state used to report RBD results.</td>
</tr>
<tr>
<td>Network Interface Controller</td>
<td>NIC</td>
<td>Network Interface Controller. Integrated circuits that implement the original MOST25 speed grade, and require all network startup intelligence to be managed by the EHC.</td>
</tr>
<tr>
<td>Network Switched Power</td>
<td>NwSwP</td>
<td>Power management conceptual power supply net used to power the INIC and the EHC. If a separate application power supply (ApSwP) is not implemented, then NwSwP also powers the application.</td>
</tr>
</tbody>
</table>
# INIC Hardware Concepts

## Glossary (Continued)

<table>
<thead>
<tr>
<th>Term</th>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original Equipment Manufacturer</td>
<td>OEM</td>
<td>A company who manufactures products or components that are purchased by another company and retailed under that purchasing company's brand name. In the scope of this document, it refers to a vehicle manufacturer.</td>
</tr>
<tr>
<td>Optical Electrical Converter</td>
<td>OEC</td>
<td>Also known as fiber optic receiver, FOR. Devices designed for MOST networks using an optical PHY layer to receive optical signals and convert them to electrical signals for INIC to use. Includes circuitry for supporting a Sleep Power State.</td>
</tr>
<tr>
<td>Optical Physical Network</td>
<td>oPHY</td>
<td>Network connections that use FOTs and plastic optic fiber for ECU-to-ECU connections.</td>
</tr>
<tr>
<td>Polymer Optical Fiber</td>
<td>POF</td>
<td>A 1 mm diameter plastic fiber used in MOST25 and MOST150 optical physical networks.</td>
</tr>
<tr>
<td>Power Master</td>
<td></td>
<td>An ECU containing a logical software block which manages power up and power down of the network. Typically the same ECU that contains the Timing Master. Only one Power Master can exist in a MOST network. All other ECUs are designated as Power Slaves.</td>
</tr>
<tr>
<td>Power Slave</td>
<td></td>
<td>Refers to ECUs in a network that are not responsible for managing network power up and power down behavior. Opposite of a network Power Master device.</td>
</tr>
<tr>
<td>Ring Break Diagnosis</td>
<td>RBD</td>
<td>Diagnosis mode built into INIC chips to help determine where a break exists in the MOST network. Defined in the MOST Specification 3.0 for MOST50 and MOST150.</td>
</tr>
<tr>
<td>Sleep Power State</td>
<td></td>
<td>ECU state in which the device is connected to a continuous battery supply, but is drawing minimal current (I_{STBY}). Most of the circuitry in the ECU is powered off; only circuitry needed to wake up is powered. The opposite of a Sleep Power State is an Active Power State.</td>
</tr>
<tr>
<td>Standby Current</td>
<td>I_{STBY}</td>
<td>The total current draw of the ECU during the Sleep Power State. Maximum value for each ECU is generally specified by the system integrator or OEM.</td>
</tr>
<tr>
<td>Startup Sequence</td>
<td></td>
<td>The sequence (including retries) used by the Power Master to startup the network.</td>
</tr>
<tr>
<td>Switched-Application Signal</td>
<td>SA</td>
<td>Optional digital signal from the EHC that controls power to the applications switched power (ApSwP) block of circuitry. This signal allows the EHC to power down the application circuitry while still keeping the network alive.</td>
</tr>
<tr>
<td>Switch-To-Power</td>
<td>STP</td>
<td>Method to start Ring Break Diagnosis by removing battery power for greater than 2 s. This method is deprecated due to the complexity of synchronizing all ECUs regarding the power event.</td>
</tr>
<tr>
<td>System Integrator</td>
<td></td>
<td>Entity responsible for coordination of all devices in the network from all suppliers. This entity could be the OEM, or an ECU supplier designated by the OEM.</td>
</tr>
<tr>
<td>Test Start Impulse</td>
<td>TSI</td>
<td>The Test Start Impulse is an assertion on the ECL that indicates an ECL System Test is commencing. t_{TSI} is the duration of the ECL assertion and N_{TSI} is the number of ECL assertions.</td>
</tr>
<tr>
<td>Timing Master</td>
<td></td>
<td>ECU containing an INIC that is configured as the master clock for the MOST network. The Timing Master INIC generates the system clock and framing signals for the entire network. All other network ECUs are designated as Timing Slave devices, which derive their local timing from the incoming network bitstream.</td>
</tr>
<tr>
<td>Timing Slave</td>
<td></td>
<td>An ECU containing an INIC that is configured to derive its local timing from the incoming network bitstream. Opposite of a network Timing Master device.</td>
</tr>
<tr>
<td>Visible Node</td>
<td></td>
<td>Normal INIC state where INIC is visible to all other network ECUs, has a node position, and can communicate with all other ECUs. The opposite of a Visible Node is a node in Bypass Mode.</td>
</tr>
<tr>
<td>Wakeup Event</td>
<td></td>
<td>An event in a vehicle ECU that plays a role in waking up one or more ECUs from the Sleep Power State. These events can be grouped into two categories: - A Network Wakeup Event is intended to wakeup all network ECUs from the Sleep Power State. Examples include network activity, ECL assertion, or an STP pulse. - A Local Wakeup Event occurs within a single ECU. Examples include clamp status, a Power-On switch press, or an incoming call to a wireless receiver. The ECU must properly qualify (debounce, validate, etc.) the event before generating a Network Wakeup Event.</td>
</tr>
</tbody>
</table>
# AMERICAS

**Corporate Office**  
2355 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 480-792-7200  
Fax: 480-792-7277  
Technical Support: [http://www.microchip.com/support](http://www.microchip.com/support)  
Web Address: [www.microchip.com](http://www.microchip.com)

**Atlanta**  
Duluth, GA  
Tel: 678-957-9614  
Fax: 678-957-1455

**Austin, TX**  
Tel: 512-257-3370

**Boston**  
Westborough, MA  
Tel: 508-756-0087  
Fax: 508-756-0088

**Chicago**  
Itasca, IL  
Tel: 630-285-0071  
Fax: 630-285-0075

**Cleveland**  
Independence, OH  
Tel: 216-447-0464  
Fax: 216-447-0643

**Dallas**  
Addison, TX  
Tel: 972-818-7243  
Fax: 972-818-2924

**Detroit**  
Novi, MI  
Tel: 248-848-4000

**Houston, TX**  
Tel: 281-894-5983

**Indianapolis**  
Noblesville, IN  
Tel: 317-773-8323  
Fax: 317-773-5453

**Los Angeles**  
Mission Viejo, CA  
Tel: 949-462-9523  
Fax: 949-462-9608

**New York, NY**  
Tel: 631-435-6000

**San Jose, CA**  
Tel: 408-735-9110

**Canada - Toronto**  
Tel: 905-673-0699  
Fax: 905-673-6509

# ASIA/PACIFIC

**Asia Pacific Office**  
Suites 3707-14, 37th Floor  
Tower 6, The Gateway  
Harbour City, Kowloon  
Hong Kong  
Tel: 852-2943-5100  
Fax: 852-2401-3431

**Australia - Sydney**  
Tel: 61-2-9868-6733  
Fax: 61-2-9868-6755

**China - Beijing**  
Tel: 86-10-8569-7000  
Fax: 86-10-8528-2104

**China - Chengdu**  
Tel: 86-28-8665-5511  
Fax: 86-28-8665-7889

**China - Chongqing**  
Tel: 86-23-8980-9588  
Fax: 86-23-8980-9500

**China - Hangzhou**  
Tel: 86-571-8792-8115  
Fax: 86-571-8792-8116

**China - Hong Kong SAR**  
Tel: 852-2943-5100  
Fax: 852-2401-3431

**China - Nanjing**  
Tel: 86-25-8473-2460  
Fax: 86-25-8473-2470

**China - Qingdao**  
Tel: 86-532-8502-7355  
Fax: 86-532-8502-7205

**China - Shanghai**  
Tel: 86-21-5407-5533  
Fax: 86-21-5407-5066

**China - Shenyang**  
Tel: 86-24-2334-2829  
Fax: 86-24-2334-2393

**China - Shenzhen**  
Tel: 86-755-8864-2200  
Fax: 86-755-8203-1760

**China - Wuhan**  
Tel: 86-27-5980-5300  
Fax: 86-27-5980-5118

**China - Xian**  
Tel: 86-29-8833-7252  
Fax: 86-29-8833-7256

**China - Xiamen**  
Tel: 86-592-2388138  
Fax: 86-592-2388130

**China - Zhuhai**  
Tel: 86-756-3210040  
Fax: 86-756-3210049

# ASIA/PACIFIC

**India - Bangalore**  
Tel: 91-80-3090-4444  
Fax: 91-80-3090-4123

**India - New Delhi**  
Tel: 91-11-4160-8631  
Fax: 91-11-4160-8632

**India - Pune**  
Tel: 91-20-3019-1500

**Japan - Osaka**  
Tel: 81-6-6152-7160  
Fax: 81-6-6152-9310

**Japan - Tokyo**  
Tel: 81-3-6880-3770  
Fax: 81-3-6880-3771

**Korea - Daegu**  
Tel: 82-53-744-4301  
Fax: 82-53-744-4302

**Korea - Seoul**  
Tel: 82-2-554-7200  
Fax: 82-2-558-5932 or 82-2-558-5934

**Malaysia - Kuala Lumpur**  
Tel: 60-3-6201-9857  
Fax: 60-3-6201-9859

**Malaysia - Penang**  
Tel: 60-4-227-8870  
Fax: 60-4-227-4068

**Philippines - Manila**  
Tel: 63-2-634-9065  
Fax: 63-2-634-9069

**Singapore**  
Tel: 65-6334-8870  
Fax: 65-6334-8850

**Taiwan - Hsin Chu**  
Tel: 886-3-5778-366  
Fax: 886-3-5770-955

**Taiwan - Kaohsiung**  
Tel: 886-7-213-7830

**Taiwan - Taipei**  
Tel: 886-2-2508-1012

**Thailand - Bangkok**  
Tel: 66-2-694-1351  
Fax: 66-2-694-1350

# EUROPE

**Austria - Wels**  
Tel: 43-7242-2244-39  
Fax: 43-7242-2244-393

**Denmark - Copenhagen**  
Tel: 45-4450-2828  
Fax: 45-4485-2829

**France - Paris**  
Tel: 33-1-69-53-63-20  
Fax: 33-1-69-30-90-79

**Germany - Dusseldorf**  
Tel: 49-2129-3766400

**Germany - Munich**  
Tel: 49-89-627-144-0  
Fax: 49-89-627-144-44

**Germany - Pforzheim**  
Tel: 49-7231-424750

**Italy - Milan**  
Tel: 39-0331-742611  
Fax: 39-0331-466781

**Italy - Venice**  
Tel: 39-049-7625286

**Netherlands - Drunen**  
Tel: 31-416-690399  
Fax: 31-416-690340

**Poland - Warsaw**  
Tel: 48-22-3325737

**Spain - Madrid**  
Tel: 34-91-708-08-90  
Fax: 34-91-708-08-91

**Sweden - Stockholm**  
Tel: 46-8-5090-4654

**UK - Wokingham**  
Tel: 44-118-921-5800  
Fax: 44-118-921-5820

03/25/14