MOST®

Media **O**riented **S**ystems **T**ransport

Multimedia and Control Networking Technology

OS8104

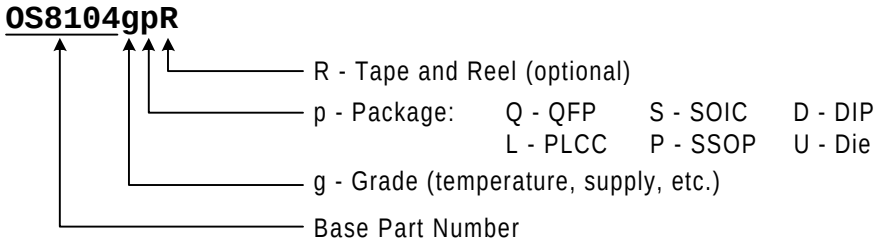
Preliminary Product Data Sheet

DS8104PP2

Sept. 2000

OASIS SiliconSystems

Ordering Information



Valid Part Numbers:

Order Number	Grade		Package
	Temperature	Supply	
OS8104AQ	-40 to +85° C	4.5 to 5.5 V	44-pin TQFP
OS8104AQR	-40 to +85° C	4.5 to 5.5 V	44-pin TQFP, Tape and Reel

This table represents parts that were available at the time of printing and may not represent parts that are currently available. For the latest list of valid ordering numbers for this product, please contact your local sales office.

Support and Further Information

For more information on the MOST technology, product line, and custom IC development using MOST tools, contact one of our offices below.

Oasis SiliconSystems AG
1101 S. Capital of Texas Highway
Building B, Suite 101
Austin, Texas 78746 USA

Tel: (+1) 512 306-8450
Fax: (+1) 512 306-8442
america@oasis.com

Oasis SiliconSystems AG
Bannwaldallee 48
D-76185 Karlsruhe
Germany

Tel: (+49) (0) 721 6 25 37 - 0
Fax: (+49) (0) 721 6 25 37 - 119
europe@oasis.de

Oasis SiliconSystems AG
4-16 Oomaru Tsuzuki-ku
Yokohama 224-0061, Japan

Tel: (+81) (90) 2757 1419
Fax: (+81) (45) 941 7818
pacrim@oasis.de

Technical Support

For technical support please refer to one of the following email addresses:

support@oasis.de
support@oasis.com

Intellectual Property

Trademarks

Patents

There are a number of patents and patents pending on the MOST technology. The rights to these patents are not granted without any specific Agreement between the users and the patent owners.

Preliminary Product Data Sheet

A Preliminary Product Data Sheet describes a product which is in limited production and subject to change. Oasis SiliconSystems has worked diligently to ensure that the information in this document is accurate and reliable. However, the information in this document is subject to change without notice and is provided "as is" without warranty of any kind (expressed or implied).

Conventions

Within this manual, the following abbreviations and symbols are used for improving readability:

	Comment
PIN	Name of a PIN
BIT	Name of a single bit within a register
x..y	Range from x to y. Used as abbreviation e.g. for a group of bits like *D7..0*
[a,b,c]	List of alternative elements to choose from.
0xzz	Hexadecimal number (value zz)
bREG	Single-byte (8-bit) register
wREG	Single-word register (16-bit)
mREG	Multi-byte register (e.g. message buffer)
rsvd	The respective bit or register is reserved for future use
/	Inverter. Attached pin or bit uses inverted logic (low or 0 active)
x	Don't care
↑	Rising edge
↓	Falling edge

Revision History

Revision	Date	Description
1.1		Initial Data Sheet
2	Sept. 2000	Fully revised.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Features

Lowest Cost Solution for high quality multimedia networking

- Data rate of over 24.5 Mbit/s
- ⇒ Bandwidth allocable for both asynchronous and synchronous data
- ⇒ Independent 768 kbit/s control channel
- Designed for ultra-low cost network nodes
- ⇒ On-chip support of fiber optic physical layer
- ⇒ Stand-alone operation (no MCU) by remote access
- ⇒ No buffering required for audio applications

On-chip Network Management Functions

- Automatic multimedia channel allocation
- Automatic wake-up on sensing network activity
- Node position and delay detection
- Fail-safe node bypass and error reporting

Flexible I/O

- Consumer Electronic / Multimedia I/O
- ⇒ Multiple speed CD & Media player interfaces
- ⇒ Real-time I/O with various data formats
- ⇒ 1-8x speed IEC 958 (S/PDIF or AES-EBU) port
- ⇒ Transparent channel with sample rate up to 3 MHz
- Control I/O
- ⇒ High speed IIC/SPI serial control interface
- ⇒ FIFO buffered 8-bit parallel control interface
- ⇒ Transparent interface up to 128 kbit/s.

Easy System Integration

- Low power and zero power mode
- Low power 5 V operation
- 44-pin TQFP package

Description

The OS8104 is a highly integrated CMOS Transceiver IC with full support for the 24.5 Mbit/s MOST Multimedia and Control network.

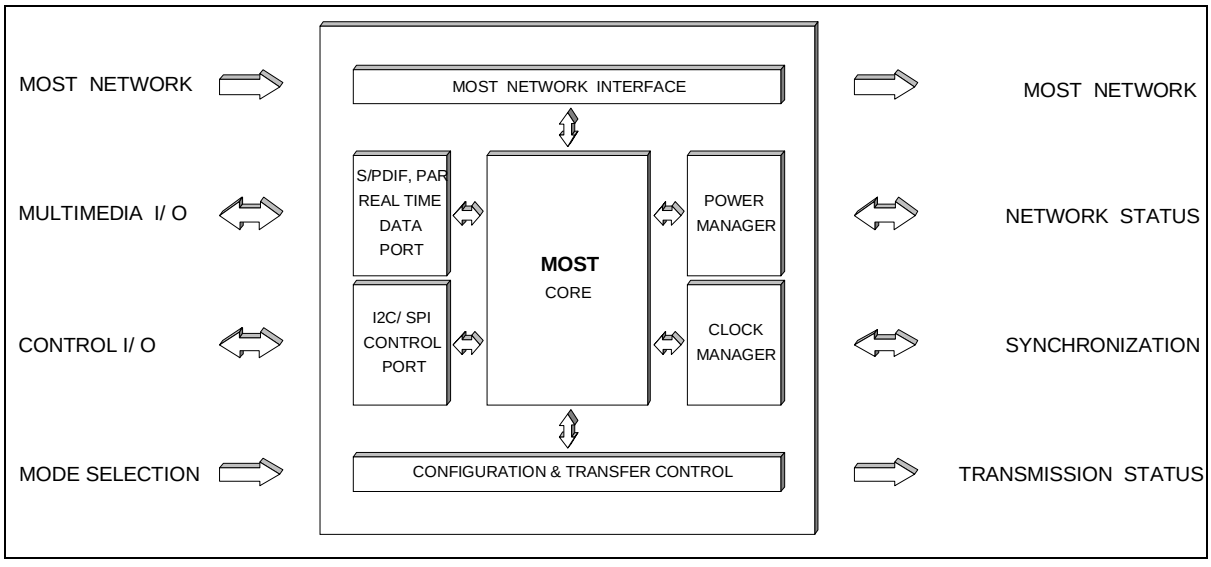
This high performance, low cost, real-time network utilizes a ring topology and a plastic optical fiber (POF) physical layer, both of which are supported directly by the OS8104.

All relevant network management functions are handled on-chip. An ultra-low jitter PLL guarantees high quality audio and video transmission, and clock recovery over a wide frequency range.

The 24.5 Mbit/s aggregate bit rate provides the capability to network up to 60 byte-wide synchronous data channels (equivalent to 15 stereo, CD-audio channels) or any appropriate combination of synchronous and asynchronous data streams. Control data is also networked, transparently to the source data channels, at an aggregate bit rate of 768 kbits/s.

The OS8104 can be managed remotely, from any other network node; or locally via the Control Port in I^2C or SPI more, or through a parallel format.

Programmable serial and parallel real-time data interfaces, with different clock and data modes, support connections to multiple speed CD drives and other consumer electronic devices, as well as comple–mentary ICs such as A/D and D/A converters or DSPs.



1 Introduction

The OS8104 is a complete MOST (Media Oriented Systems Transport) network Transceiver (**trans**mitter/re**ceiver**) device capable of more than 24 Mbit/s data throughput that provides a "stand-alone" solution when interfacing to a MOST optical network. The only additional components required to implement a node are a few capacitors and resistors required for the PLL loop filter. The architecture of the OS8104 is based on a RISC microcontroller to achieve the maximum performance and flexibility at the lowest possible price. A complete MOST network node can even be implemented without any other network-specific hardware, just the OS8104 and related passive components.

The network is peer-to-peer and requires no hardware overhead such as a hub (although it supports hub-based architectures). In addition to handling network interface and communication management functions, the OS8104 also handles all of the important network management functions such as node position sensing, network delay detection, start up and shut down, as well as error reporting, fail safe operation, and channel allocation.

An ideal application for a MOST network is in the multimedia field where a number of audio, video and communication devices are connected with each other and require flexible control and high speed real-time data exchange.

2 General Overview

2.1 Functional Description

The OS8104 chip contains a RISC microcontroller-based MOST Routing Engine and several peripherals including a clock manager, Source Data Ports, wake-up logic and an I^2C/SPI/parallel Control Port.

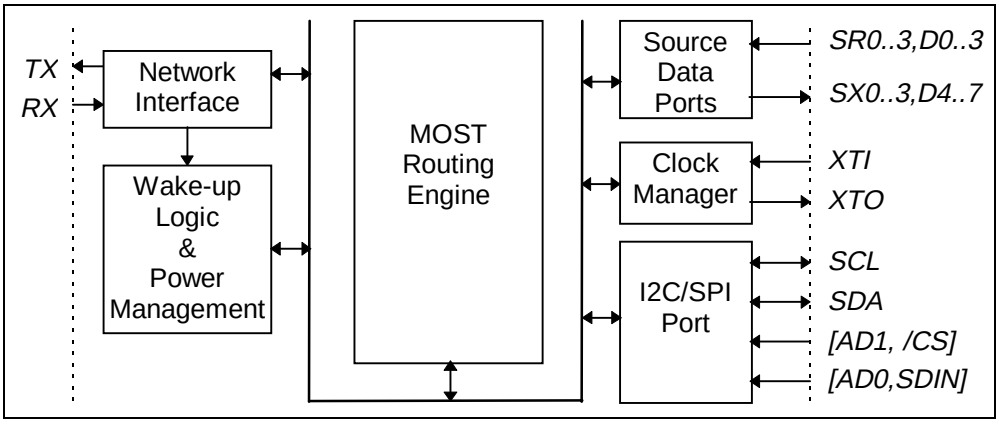


Figure 2-1: MOST Routing Engine

The network interface includes an ultra-low jitter Phase Lock Loop (PLL) and a channel demodulator on the input side (*RX*). The network output (*TX*) provides a channel-coded signal. Those signals can be connected directly to optical receiving/transmitting devices (FOT units) or balanced line drivers. The line quality can be monitored using the built-in error detection circuit.

Each device can generate the network clock or synchronize to it; however, only one device can generate the clock and the frame structure for a particular network. Various A/D converters, D/A converters, digital signal processors, and Media Players (CD-AUDIO/VIDEO/ROM etc.) can be synchronized through one of the many clock and serial interface modes.

The Source Data Ports provide a synchronous data interface to the network, generally used for multimedia devices. Data sources and sinks can be connected to the Source Ports to transfer data to another node on the network, or receive data from another node on the network. The Source Ports

support up to four simultaneous inputs and four simultaneous outputs for transferring real-time data between external peripherals and the MOST network.

2.2 Network Interface & Compatibility

The Source Data Ports support many different clock modes providing an easy interface to single- and multiple-speed CD devices (i.e. I^2S, Sony, Matsushita format). In addition, one port can be configured for an S/PDIF (IEC 60958 parts 1 and 3) input and another as an S/PDIF output. The S/PDIF interface can operate up to eight times the normal CD data rate; therefore, the maximum S/PDIF data rate at a network frame rate of 48 kHz is:

$$(24 \text{ bit} \times 2) \times 8 \times Fs = 18{,}432 \text{ bit/s} = 2.304\ 10^6 \text{ byte/s (approx. 2.3 Mbytes/s)}$$

Real-time routing of source data through the network is managed automatically by the on-chip routing algorithm. This algorithm handles the channel allocation and supplies the labels for the requested channels (as a result of an allocation request). Unused channels will be flagged and can be de-allocated in the event of unplugging or de-installation of a device. Real-time (stream) data can have one or multiple destinations. All real-time channels are available to all nodes on the network (simulcast).

Each node has a network address (logical) and a node position address (physical) for control purposes and system level network management functions such as initialization, fault reporting or delay detection. Control messages can be single, group or broadcast. The Control Port can be configured for I^2C or SPI formats. The highest data throughput for the OS8104 is achieved by configuring the part for parallel port mode which supports control and real-time data access. Parallel mode can also be used just for real-time data, leaving the control data accesses to the Control Port in serial mode. Transferring real-time data in parallel mode is ideal for burst type data and supports buffering of up to 8 bytes.

Remote access to all control functions and to the on-chip Control Port (configured as an I^2C-master interface) supports stand-alone operation and diagnostics. This allows entire nodes to be configured and controlled remotely (from other network nodes) and checked. Even external devices connected to the OS8104 at a node can be remotely controlled (via the Control Port) in this manner.

For low power consumption, the chip can be switched into a low-power, or zero-power mode. The zero-power mode can be exited by a local reset or remotely through network activity detection.

2.3 On-Chip Network Management

The core network management functions in a MOST network are handled automatically, on a distributed basis, and are embedded into the OS8104 MOST transceiver itself. Since channel allocation, physical addressing, fault monitoring and power-down/wake-up is provided on-chip, the implementation of the network is very simple and a high level of network protection and reliability is achieved. Remote access allows for network management functions, such as network diagnostics, to be handled centrally or in a decentralized manner within each node, depending on the higher layer software structure.

Physical position sensing, MOST network delay detection, and node alive supervision are some of the essential mechanisms provided on-chip. Part of the initialization of the network is the node position sensing mechanism which provides a unique physical address for every chip in the network. This initialization procedure is part of the network management and is done on-chip.

A device can also be configured and controlled with a logical address (DeviceID in the MOST Specification) within the network. The application can write the target logical address into the chip and the on-chip network management verifies its unique existence. If the address is already used by another device, the address will be rejected and the application will be notified. All data requirements and channel connections are established during initialization as well as dynamically in the background during normal operation.

An OS8104 device, in a network node, can be either active or passive. During active operation the request for data capacity comes from the application using a particular communication mode. As soon as a connection has been made the required data capacity is provided by the network.

The application doesn't have to deal with resource management within the network, as long there is no resource conflict reported. Resource conflict will be detected as soon as the maximum network capacity of more than 24 Mbits/s is not sufficient to serve the applications running on the system. In this case the unavailability of additional data capacity will be flagged. For the majority of applications, the large data capacity of the network prevents the system from running into capacity conflicts, making high-level software administration and segmented transmission unnecessary.

The source data allocation algorithm is managed on the bit-stream level. This results in fast response times (in milliseconds) and high data efficiency at the same time. Once a synchronous connection is built, it will stay in place while the application needs the channels. Once the application is finished using the channels, the application can de-allocate the channels allowing others to use them. Burst data channels are managed at the frame level, since their loading might change very quickly, and the latency time for that data transfer should be low.

If activities at several nodes must be synchronized and in phase (for example, different speaker connections) the inherent network delay can be compensated by using the network delay-reporting mechanism. This makes the information on relative network delay, with respect to each channel, available to every node.

All other bits within the Frame are for management purposes on the network level. For example, while the Preamble provides synchronization and clock regeneration, the Parity Bit indicates reliable data content and is used for error detection and Phase Lock Loop operation.

2.3.1 Channel Allocation

Within each MOST frame, 60 byte-wide physical channels are available for transporting source data (any data coming from or going to a Source Data Port). Up to eight physical channels (64 bits) can be allocated with a single allocation request, thus forming a logical channel.

For example, a stereo CD-audio channel requires four of the byte-wide physical channels, running at a frame rate of 44.1 kHz. Thus, up to 15 stereo CD-audio channels can be supported simultaneously using the network's 60 available byte-wide physical channels. Logical channels can be clustered into multi-channel streams in higher software layers.

Source data routing becomes an easy task, since all necessary functionality is supported by the bit-stream structure and managed on-chip. Channel allocation (also referred to as resource allocation) is done by a network-level channel-allocation algorithm, embedded in the OS8104. Each node contains a channel allocation table containing labels associated with each logical channel (connection labels) available at any point in time. Since synchronous source data channels (bytes) are quasi static, two functions for allocation and de-allocation are available. Nodes that want to place source data onto the network can identify the existence of free channels and request allocation from the network. Nodes that want to sink data can identify the correct channels by the connection label without needing to send messages to the source nodes. Up to 60 synchronous byte sized data channels are potentially available. The channel allocation can be changed during runtime. The entire channel building procedure has a maximum latency time of 25 ms after network lock.

2.3.2 Physical Position Sensing

The node position relative to the Frame Generator (timing-master node) is available in each node, once the network is running. The node position determines the node's unique location (and physical address) in the network. The frame generator senses the number of nodes in the network and provides this information to the application. Using the node position for configuration sensing, enables "hot" plugging supporting dynamic system reconfiguration.

2.3.3 Network Delay Detection

The delay in a network, relative to the timing-master, is not necessarily directly related to the node position, since nodes can be active or passive. Only active nodes (Source Data Bypass inactive) add delay to source data. The network delay detection provides an accurate number of frame delays and supports delay compensation. Similar to the network position, every node also knows the maximum delay in the network. Network delay compensation can be very important for high fidelity audio, noise canceling, speech recognition, and multi-channel sound applications.

2.3.4 Node Alive Supervision

The node-alive supervision mechanism is used for network management, such as channel allocation, error management and power management. To avoid channel blocking, due to unplugged or defective devices, devices that are detected as not alive or sleeping can be taken out (de-allocated) of the resource request list. By this means the corresponding channels will become available to other nodes.

2.4 On-Chip Power Management

Two different power saving modes are provided by the OS8104 transceiver chip. The Zero-Power mode is available when not using the receive FOT to manage system power, as described in the MOST Specification and illustrated in Section A.1. Per the MOST Specification, the receive FOT will power down the entire node when no light is detected on the RX pin. When not using an FOT with power management capability (or not using FOTs at all), the OS8104's Zero-Power mode can minimize node power (albeit not as low as when using the receive FOT to manage power).

The Low-Power mode will place the device in the lowest power state where the network is still running, but the particular node is not being used. As of the writing of this Data Sheet, the Low-Power mode is not supported by the existing MOST Specification.

2.4.1 Low-Power Mode

The MOST core can enter a low-power mode where all functions except the transceiver are stopped. Normal activity is resumed after receipt of a wake-up signal. Setting the low-power wake-up bit of the timing-master node will cause a wake-up signal to be generated and sent around the network.

2.4.2 Zero Power Mode

The Zero-Power mode can be initiated via the network or control messages. During zero-power operation, the network is constantly checked for activity. As soon as activity is detected, the device will power up.

2.5 Data Transfer Methods

The OS8104 supports three methods for transferring data:

- Synchronous, or Stream Data Transfer
- Control Message Data Transfer
- Asynchronous, or Packet Data Transfer

The synchronous data transfer method uses a circuit-switched approach for very-low overhead support of streaming real-time data, such as audio or video. A typical application for synchronous data transfer is sending streaming audio from a CD transport unit to an amplifier, that converts the audio data to analog and plays the data out of speakers. Once the logical connection is setup, via Control messages, no

overhead is wasted while streaming the data across the network, providing the maximum efficiency for real-time data.

The Control message data transfer method operates simultaneously with the other transfer methods and supports control, status queries, and notification status among all nodes in the network. This transport mechanism is supported by every node and is the backbone of the MOST Specification. Messages can be sent to all devices (broadcast), to just a group of devices (groupcast), or to a specific node (using logical or physical addressing). A large set of standard functions and function classes are pre-defined to support peer-to-peer interoperability. Control messages can turn on functions in other nodes (such as starting a CD player), as well as to get status from a node (such as the current track and time from the CD player).

The packet data transfer method uses a channel shared by all nodes to transfer asynchronous burst-type data. This method arbitrates for the channel and supports sending messages to individual nodes. The portion of the network bandwidth allocated to asynchronous vs. synchronous data is system-controllable and can be optimized for each system. Examples of asynchronous packet data include internet data, GPS map data, or still video images that are only occasionally sent across the network. The packet data transfer method is similar to Ethernet-style communication.

Once the network is configured, the synchronous, asynchronous, and control data streams run concurrently supporting a very predictable, low-latency, high-performance network. High data loading on one method does not affect the other transport methods. Supporting these three methods simultaneously allows different types of data to operate across the same network, providing higher bandwidth utilization than with single-method networks.

2.5.1 Bandwidth

Bandwidth can be divided into two categories: MOST network bandwidth and OS8104 interface bandwidth. Each of these two categories can be applied to each of the transport methods (synchronous, asynchronous, and control).

2.5.1.1 Control Messaging

A control message requires 16 MOST frames to be transported across the MOST network. Therefore, the message rate depends upon the network frame or sample rate (Fs). In addition, some control message bandwidth is used for network administration. The final application available throughput on the MOST network is determined by:

$62 \times Fs/1024$ = control messages per second (2906 msg/s at Fs = 48 kHz)

Each control message contains 17 bytes of useable data which leads to a net data rate of:

$62 \times Fs/1024$ msg/s $\times$ 17 bytes/msg $\times$ 8 bits/byte = bits per second, or
$8432 \times Fs/1024$ bits/s (395.3 kbits/s at Fs = 48 kHz)

This Control message bandwidth is shared between all nodes on the network by a fair arbitration mechanism.

The OS8104 is capable of arbitrating for a new TX message in the third control message after it finishes sending the last control message. Therefore, assuming no other devices are arbitrating for the control message channel, the maximum control data rate for a single OS8104 node is:

$(8432 \times Fs/1024) \div 3$ bits/s (131.8 kbits/s at Fs = 48 kHz)

The actual control message throughput realized by a particular OS8104 device depends on how the Control Port interface is configured (SPI, I^2C, or through the parallel port).

2.5.1.2 Synchronous (Stream) Data

The synchronous bandwidth available in the MOST system at any particular time depends on the bSBC (synchronous bandwidth control) value and the network frame or sample frequency (Fs). bSBC is set in the timing-master node and sent to all the timing-slave nodes in the network. bSBC determines the division of network source data between synchronous and asynchronous transfer methods (for an overview of the MOST network architecture, see Figure 6-1). The bSBC value is specified in quadlets (four bytes equals one quadlet), where the minimum is 6 (6 quadlets reserved for synchronous data, 9 quadlets for asynchronous data), and the maximum is 15 (15 quadlets for synchronous data and no asynchronous data). The MOST network available synchronous bandwidth is:

SBC quadlets × 4 bytes/quadlet × 8 bits/byte × Fs 1/s

Therefore the minimum MOST network synchronous bandwidth is:

6 quadlets × 4 bytes/quadlet × 8 bits/byte × Fs 1/s, or
192 × Fs bits/s (9.216 Mbits/s at Fs = 48 kHz)

And the maximum MOST network synchronous bandwidth is:

15 quadlets × 4 bytes/quadlet × 8 bits/byte × Fs 1/s, or
480 × Fs bits/s (23.04 Mbits/s at Fs = 48 kHz)

Since the SBC unit is 1 quadlet, the step size for synchronous bandwidth is 1.54 Mbits/s. Synchronous bandwidth is allocated similar to a circuit-switched topology; therefore, none of the real-time bandwidth is used for addressing. Control messages setup the virtual circuit between the source and one or more destinations. Individual nodes can request (Allocation request) synchronous bandwidth in steps of one byte (one physical channel).

The volume of synchronous source data that the OS8104 can transfer is dependent on the configuration of the Source Ports. The Source Ports can be configured in three modes. In serial mode, synchronous source data is transferred serially into the chip on the SR0..3 pins and out of the chip on the SX0..3 pins. The format chosen for those pins also determines the maximum data rate. When the OS8104 is configured for parallel operation, in either Parallel-Synchronous or Parallel-Combined (Physical) modes, the part can transfer the entire MOST synchronous source bandwidth. When the OS8104 is configured for parallel operation in the Parallel-Asynchronous mode, no synchronous source data is transferred.

When the Source Ports are configured for one of the standard serial formats (I^2S, S/PDIF, etc.), the maximum synchronous data rate, in each direction, is:

4 × 64 × Fs bits/s (12.288 Mbits/s at Fs = 48 kHz)

Using a special S/PDIF mode, this synchronous data rate can be doubled, although only one direction is supported (in or out):

8 × 64 × Fs bits/s (24.576 Mbits/s at Fs = 48 kHz)

which supports the maximum MOST network synchronous bandwidth of 23 Mbits/s at Fs = 48 kHz.

When the Source Ports are configured in Parallel-Synchronous or Parallel-Combined (Physical) mode, the OS8104 parallel port can handle the maximum MOST network synchronous bandwidth, in each direction (along with one status quadlet in Parallel-Combined mode), supporting a bandwidth of:

15 quadlets × 4 bytes/quadlet × 8 bits/byte × Fs 1/s, for Parallel-Synchronous, or
480 × Fs bits/s (23.04 Mbits/s at Fs = 48 kHz)

and

16 quadlets × 4 bytes/quadlet × 8 bits/byte × Fs 1/s, for Parallel-Combined, or
512 × Fs bits/s (24.576 Mbits/s at Fs = 48 kHz)

2.5.1.3 Asynchronous (Packet) Data

As with the synchronous bandwidth, the asynchronous bandwidth available on the MOST network at any particular time depends on the bSBC (synchronous bandwidth control) value and the network frame or sample frequency (Fs). bSBC determines the division of network source data between synchronous and asynchronous transfer methods. The bSBC value is specified in quadlets (four bytes equals one quadlet), where the minimum is 6 (6 quadlets reserved for synchronous data, 9 quadlets for asynchronous data), and the maximum is 15 (15 quadlets for synchronous data and no asynchronous data). The MOST network available asynchronous bandwidth is:

(15 – SBC) quadlets × 4 bytes/quadlet × 8 bits/byte × Fs 1/s

Therefore the maximum MOST network asynchronous bandwidth is:

9 quadlets × 4 bytes/quadlet × 8 bits/byte × Fs 1/s, or
288 × Fs bits/s (13.8 Mbits/s at Fs = 48 kHz)

The minimum MOST network asynchronous bandwidth is no asynchronous bandwidth, where all the source data quadlets are allocated to synchronous source data. The maximum packet data length when the Source Ports are in Parallel-Combined mode is 1014 bytes. In all other modes, the maximum packet length is limited to the on-chip buffer size of 48 bytes.

Since asynchronous data transfer is done via packets, and all nodes share the same channel, packet headers (arbitration, target address, source address, CRC) lower the actual user data transfer rate (by 10 bytes). Since packets can be variable lengths, the ratio of actual user data (net data) transfer to overall MOST network asynchronous data transfer is not constant. However, the longer the packet, the higher the net data rate, since the packet header size is constant.

When a packet is ready to be sent out onto the MOST network, the OS8104 must arbitrate for the asynchronous data channel. The chip can arbitrate for a new packet in the fifth frame after completion of a previous packet. When no other device is arbitrating for the channel, the OS8104 has a minimum delay of four frames between packets being sent out. The maximum net data rate that one device can achieve on the network is defined below.

R	net data rate, bits/s
Pd	packet data length (net data), bytes
Ph	packet header, bytes
Tp	transmission time for packets, frames
Ta	delays between successive packets from one device, frames
Af	Asynchronous bytes per frame, bytes
Fs	MOST network frame rate, sample frequency, 1/s
SBC	synchronous bandwidth control, quadlets/frame

For the OS8104:

Ta = 4 frames
Pd maximum value defined by Source Port mode on the OS8104 (48 or 1014 bytes)
Ph = 10 bytes
Af = (15 – SBC) × 4

Therefore, the needed frames to consider back-to-back packet transmission is:

$$Tp = \text{roundup}\left(\frac{Pd + Ph}{Af}\right) = \text{roundup}\left(\frac{Pd + Ph}{(15\text{-}SDC) \times 4}\right) = \text{roundup}\left(\frac{Pd + 10}{(15\text{-}SDC) \times 4}\right) \text{ frames}$$

The average net data rate = net data in a packet / (time for packet transmission + time between packets).

$$R = \frac{Pd \times 8}{\frac{Tp + Ta}{Fs}} = \frac{Pd \times 8 \times Fs}{Tp + Ta} = \frac{Pd \times 8 \times Fs}{\text{roundup}\left(\frac{Pd + 10}{(15\text{-}SDC) \times 4}\right) + Ta} \quad \text{bits/s}$$

The Ta value listed is the time required by the OS8104. If the external application needs more time to prepare and transfer the next packet to the chip, Ta will grow appropriately. Table 2-1 illustrates the net data rate for asynchronous transfer for different bSBC settings and different packet lengths (Pd). When the OS8104 Source Port is in Parallel-Combined mode, the maximum packet size is 1014 bytes. When the Source Ports are configured for any other mode, the maximum packet size is 48 bytes.

bSDC	Pd						Units
	1014	512	256	128	64	48	
6	11.799	10.348	8.192	6.144	3.511	3.072	Mbits/s
7	10.816	9.362	7.562	5.461	3.511	3.072	Mbits/s
8	9.497	8.548	7.022	5.461	3.511	2.633	Mbits/s
9	8.285	7.562	6.144	4.915	3.072	2.633	Mbits/s
10	6.953	6.342	5.461	4.468	3.072	2.633	Mbits/s
11	5.726	5.314	4.681	3.781	2.731	2.304	Mbits/s
12	4.326	4.096	3.641	3.072	2.234	2.048	Mbits/s
13	2.950	2.809	2.587	2.234	1.755	1.536	Mbits/s
14	1.498	1.456	1.385	1.260	1.069	0.970	Mbits/s
15	0	0	0	0	0	0	Mbits/s

Assumes Ta = 4 and a network frame rate of Fs = 48 kHz.

Table 2-1: Net Asynchronous Data Bandwidth (R)

2.6 MOSTNetServices API

For speeding up the development of applications using the OS8104, Oasis SiliconSystems offers the MOST NetServices API. The MOST NetServices API provides software access to the MOST network. All services that are relevant for MOST network are available as a software library. This includes basic services like initialization, up to high-level communication tasks. MOST NetServices is modular and can be customized for the target system. The MOST NetServices API is implemented in ANSI C, which can be adapted to individual requirements through configuration files.

The MOST NetServices API is organized into two layers: Basic Services (Layer 1) and the Applications Socket (Layer 2). The Basic Services provides low-level services such as network initialization, Control message management through the Control Port, Source Port configuration, synchronous channel allocation on the network, and asynchronous data transmission services.

The Applications Socket (Layer 2) operates on top of Layer 1 and provides a command interpreter and the NetBlock function required on all network devices. The command interpreter provides a simple API for developing new functions within a node. It also supports the MOST Specification Notification Services and functional addressing.

With respect to the ISO communications model, the OS8104 chip supports the Physical and Data-Link layers. MOST NetServices Layer 1 supports the Network layer through the Session layer. MOST NetServices Layer 2 supports the Presentation and part of the Application layer.

More information on regarding MOST NetServices can be found on the Oasis SiliconSystems web page:

http://www.oasis.com

3 Main Functional Blocks

The OS8104 has six main blocks that physically interface with the network and the application:

Functional Block	Description
Configuration Interface	For general Source and Control Data Port mode selection at start up
Network Interface	Providing control, status and connectivity to the MOSTnetwork
Clock Manager	Synchronizes the entire MOSTTransceiver to various clock sources. It provides a scaleable output of the system clock and an input for an external clock signal or a crystal.
Source Data Ports	To source and sink either synchronous source data (streams) in serial or parallel format, or asynchronous source data (packets) in a parallel format.
Control Data Port	This port exchanges control data (packets) in a serial or parallel* format. It controls the chip locally by writing and reading internal registers. In addition to that, remote control of other chips via the network is possible as well as, the requesting/allocating of system resources on the network. This port provides the exchanging of control messages to other applications within the network.
Wake-up Logic & Power Management	Activates or deactivates Power Down and Low Power mode controlled e.g. by network activity or a dedicated command.

* Parallel operation mode for the Control Port is only available if Source Data Port is in parallel mode

Table 3-1: Overview of functional blocks of OS8104

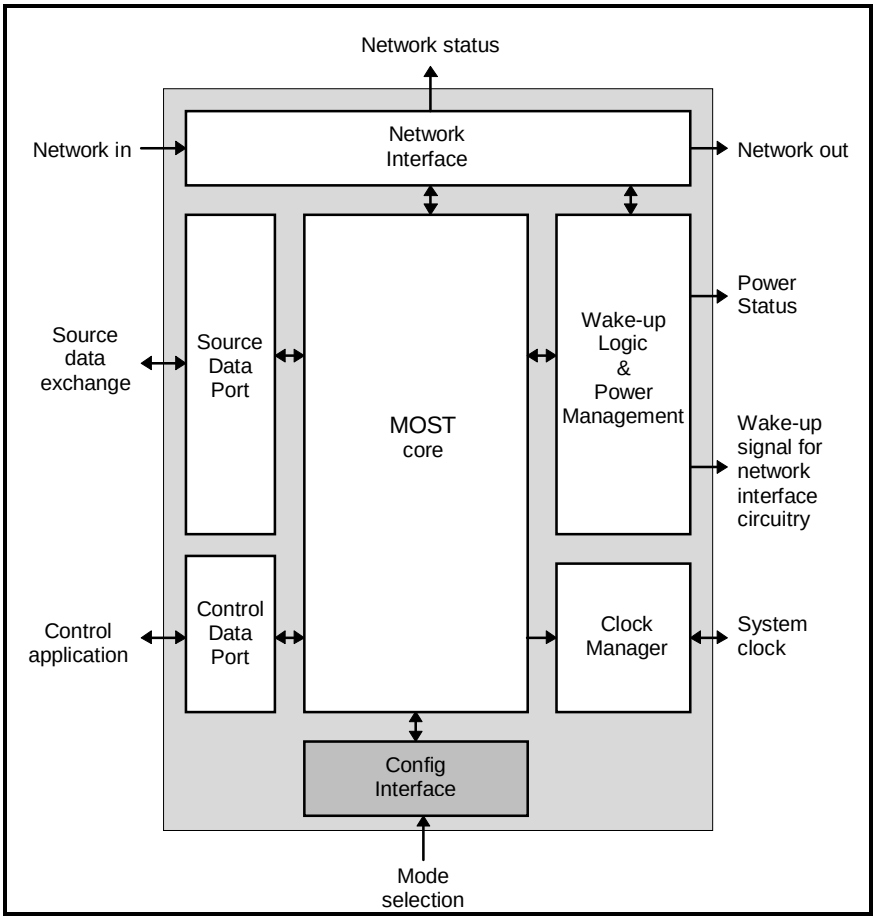


Figure 3-1: OS8014 Functional Blocks

4 OS8104 Configuration

Before being able to access OS8104 for operation, the chip must be configured. As previously described, the Control Port provides access to the internal functions of OS8104. The Source Port transports source data into and out of the chip. For both ports, either a serial or a parallel interface is available. Figure 4-1 illustrates the valid combinations of interface formats.

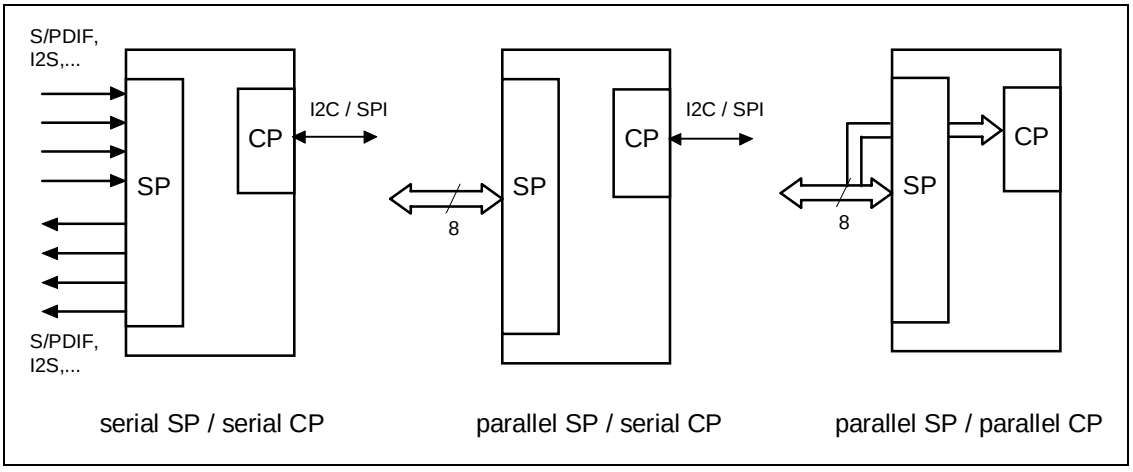


Figure 4-1: Source and Control Port Interface Options

The external interface format for the Control Port and the Source Data Port is selected via the Configuration Interface, which consists of four pins.

- *PAR_CP* (Parallel/serial mode configuration of Control Port)
- *PAR_SRC* (Parallel/serial mode configuration of Source Port)
- *ASYNC* (Source Port parallel mode: Parallel-Synchronous or Parallel-Asynchronous)
- *SCL* (At reset time, selects the serial Control Port format, I^2C or SPI mode)

In addition, /RD and /WR are used to configure the part in "Stand-Alone" mode.

PAR_CP, PAR_SRC and *ASYNC* must be valid before /RS rises and cannot change during normal operation. SCL selects the Control Port serial format on the rising edge of /RS. After initialization, SCL is the clock signal for the Control Port in serial mode.

The Control Port supports the following interfaces:

- Serial – PAR_CP tied low. Then the SCL pin selects the format for the serial Control Port.
  - A pull-up resistor on SCL selects the I^2C serial format. The pull-up on SCL is also needed since SCL is an open-drain bi-directional signal in the I^2C serial format.
  - A pull-down resistor on SCL selects the SPI serial format.
- Parallel – PAR_CP tied high.

The Source Ports use the SR3..0 and SX3..0 pins in serial mode. In parallel mode the SRn and SXn pins are configured for an 8-bit parallel port, D7..0. The Source Port supports the following interfaces:

- Serial – PAR_SRC tied low.
  - I^2S, Sony, Matsushita and various others.
- Parallel Synchronous – PAR_SRC tied high and ASYNC tied low.
  - Source data (stream data) handled in fixed time scheme via the parallel port. Asynchronous data must be handled through the Control Port (if any).
- Parallel Asynchronous – PAR_SRC tied high and ASYNC tied high.
  - No fixed timing scheme. Handshaking between the OS8104 and the external application is used. The RAM of OS8104 is accessible. No synchronous source data is handled.
- Parallel Combined – PAR_SRC tied high, ASYNC tied high, and the APCM bit in bPCMA is set.
  - Synchronous source data (stream data) and asynchronous source data (packet data) is handled in a fixed time scheme, at high speed, via the parallel port.

Table 4-1 shows the available configurations for the given signal combinations.

PAR_CP	*PAR_SRC*	*ASYNC*	*SCL*	Description
0	0	x	0	SP in serial mode, CP in serial SPI mode
0	0	x	1	SP in serial mode, CP in serial I^2C mode
0	1	0	0	SP in Parallel-Synchronous mode CP in serial SPI mode
0	1	0	1	SP in Parallel-Synchronous mode CP in serial I^2C mode
0	1	1	0	SP in Parallel-Asynchronous mode CP in serial SPI mode
0	1	1	1	SP in Parallel-Asynchronous mode CP in serial I^2C mode
1	1	0	x	SP in Parallel-Synchronous mode CP in parallel mode
1	1	1	x	SP in Parallel-Asynchronous mode CP in parallel mode
1	0	x	x	invalid combinations
The SCL value is latched at the rising edge of /RS. The Parallel-Combined mode is derived by placing the part in Parallel-Synchronous mode and setting the APCM bit in bPCMA.				

Table 4-1: OS8104 SP and CP Configuration Options

The desired mode is configured when the chip comes out of reset. Once a mode is selected it is kept until the next hardware reset or power-up reset.

The OS8104 can be controlled remotely by placing the part in Stand-alone mode, which needs no external intelligence in the local node. The OS8104 receives all commands via the network and can be used to control external peripherals. In this mode, the Control Port powers-up in serial mode as an I^2C master. The Stand-alone mode is described in detail in Section 17. The OS8104 is configured for Stand-alone mode by tying /RD and /WR to ground before /RS rises, and must stay at ground during Stand-alone operation.

If not in Stand Alone mode, the /RD and /WR pins must both remain high between the rising edge of /RS and the first falling edge of /INT.

5 Control Port in Serial Mode

The Control Port (CP) provides access to all on-chip registers and operates in either serial or parallel mode. The CP in parallel mode is discussed in Section 8. Selecting the respective mode is done by using the configuration interface. Table 4-1 on page 28 shows all available modes. The respective signals for CP are:

/RS	PAR_CP	SCL	Description
0	x	x	chip is being reset
↑	0	0	CP in serial SPI mode
↑	0	1	CP in serial I2C mode
↑	1	x	CP in parallel mode

Table 5-1: Configuration interface for CP

In serial mode, CP hardware operates at up to 400 kbit/s (clock rate), and supports either an I^2C or an SPI transmission protocol. The data received via CP is processed in bytes by the internal firmware of the OS8104. The processing time varies based on of the overall load of tasks. In I^2C mode, clock stretching provides an appropriate handshake mechanism to adapt the data transfer (byte) rate dynamically, thereby maximizing the transfer rate at any given time. In SPI mode, byte rate must be kept below the worst case processing time of 200 kbit/s.

When used as mentioned above, the OS8104 Control Port is operating as a slave device. The valid address range of any access is within a memory page, address range 0x00 through 0xFF. The default memory page is 0. Memory address 0xFF is reserved for switching between memory pages. Therefore, writing address 0xFF to the values 0 through 3 places all further memory accesses within one of the four memory pages 0 through 3, respectively.

5.1 I^2C Mode

In I^2C mode, the I^2C address of the MOST chip can be one of 4 address pairs (0x40/0x41, 0x42/0x43, 0x44/0x45, 0x46/0x47). This allows a maximum of four MOST devices on one I^2C bus. The signals AD0 and AD1 specify the device address, SCL clocks data in and out, and SDA is the bi-directional data pin. An even address indicates a write access, an odd address a read access to the respective chip.

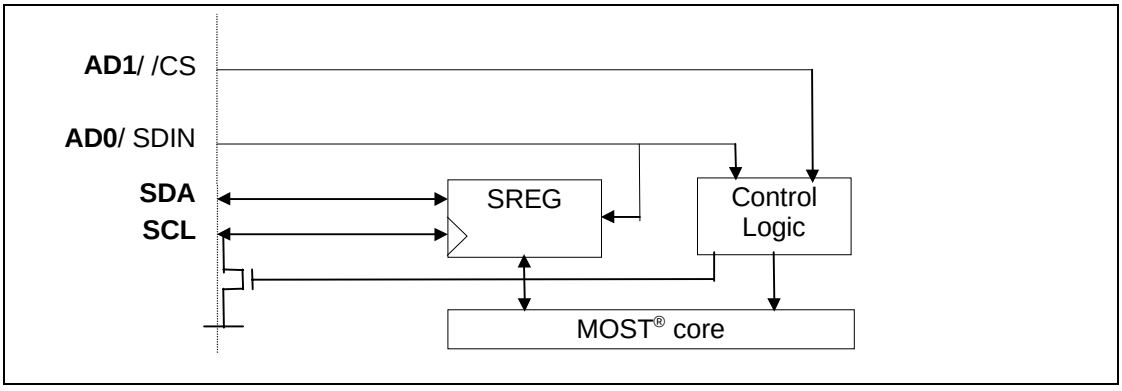


Figure 5-1: Control Port Block Diagram (I^2C Mode)

For information about I^2C standard, please refer to the I^2C Specification from Philips. The following diagram shows the connection of a MOST transceiver in an I^2C environment:

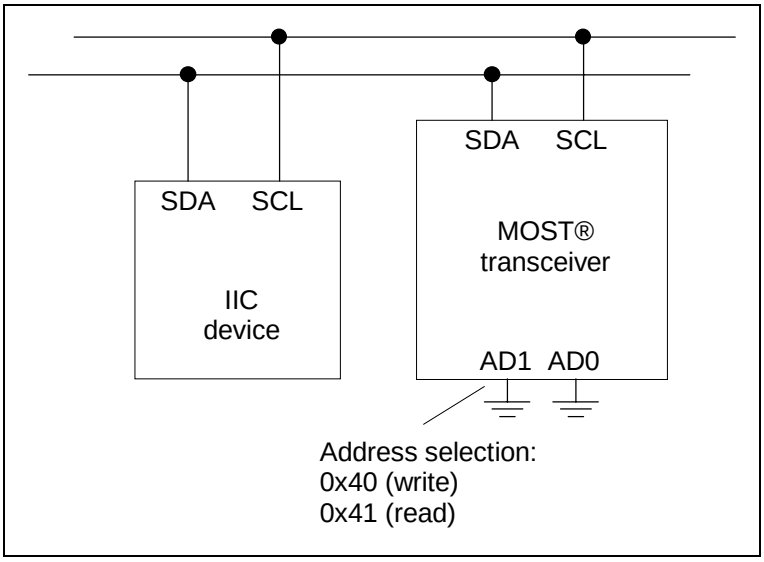


Figure 5-2: Control Port Pin Connections (I^2C Mode)

5.1.1 Writing To Control Port

The beginning of the transmission is marked by a start condition. The first byte (Address) specifies the chip address and whether the operation is a read or write. The desired operation is encoded in the LSB. A 1 stands for a read operation, a 0 for a write operation. The second byte contains the target address (Memory Address pointer, MAP), which is the address of the memory location the operation is directed to. The following bytes are interpreted as data to be written to the OS8104, until a stop condition occurs.

In the following diagram, the I^2C transmission scheme is shown. The characters "S" and "P" represent the start and stop conditions for messages in I^2C mode. "R" indicates a read operation and stands for a 1 at the respective position, while "W" indicates a write operation and stands for a 0.

	Address								MAP	DATA1	...	DATAn	
S	0	1	0	0	0	a1	a0	W	7..0	7..0	...	7..0	P

Figure 5-3: Control Port Writing in I^2C mode

The bits a1 and a0 match the values derived from the pins AD1 and AD0 and specify the chip address as follows:

AD1..0	Addr. for writing	Addr. for reading
00	0x40	0x41
01	0x42	0x43
10	0x44	0x45
11	0x46	0x47

Table 5-2: Control Port I^2C Addresses

After the MAP byte is sent, data can be continually written until a stop condition occurs. MAP will be incremented automatically. For writing to other memory pages, write the desired page number address location (MAP) 0xFF. This special address is reserved for page switching purpose in every memory page.

5.1.2 Reading From Control Port

In contrast to the write access, the read access consists of two parts. First, the target address (Memory Address pointer, MAP) must be sent to the chip using a separate transmission cycle (start/stop pair). Then the data can be read from the MAP address using a separate transmission.

The beginning of the first transmission is marked by a start condition. The first byte (Address) specifies the chip address and the operation. The desired operation is encoded in the LSB, where a 1 stands for a read operation, a 0 for a write operation. To write the MAP, the LSB is set to 0. The second byte contains the target address (MAP). This transmission is ended by a stop condition.

The second part of data transfer starts with a start condition, followed by the Address byte, which must specify a read operation now (LSB set to 1). The OS8104 will then transmit byte after byte (auto-incremented), until a stop condition occurs.

In the following diagrams, the characters "S" and "P" represent the start and stop conditions for messages in I^2C mode. "R" indicates a read operation and stands for a 1 at the respective position, while "W" indicates a write operation and stands for a 0.

Transmitting MAP	Reading DATA

Transmitting MAP:

	Address								MAP	
S	0	1	0	0	0	a1	a0	W	7..0	P

Reading DATA:

	Address								DATA1	...	DATAn	
S	0	1	0	0	0	a1	a0	R	7..0	...	7..0	P

Figure 5-4: Control Port Reading in I^2C mode

The bits a1 and a0 match the values derived from the pins AD1 and AD0 and specify the chip address as shown in Table 5-2 on page 30.

If clock rate on the I^2C bus exceeds the maximum, clock stretching will occur. The timing diagram for I^2C communication is in Section 18.4.6 on page 158.

5.2 SPI Mode

When the Control Port is in SPI mode, a unique chip select, /CS, is used in lieu of an address byte. SCL clocks data in and out of the chip, SDIN is the data input, and SDA is the data output.

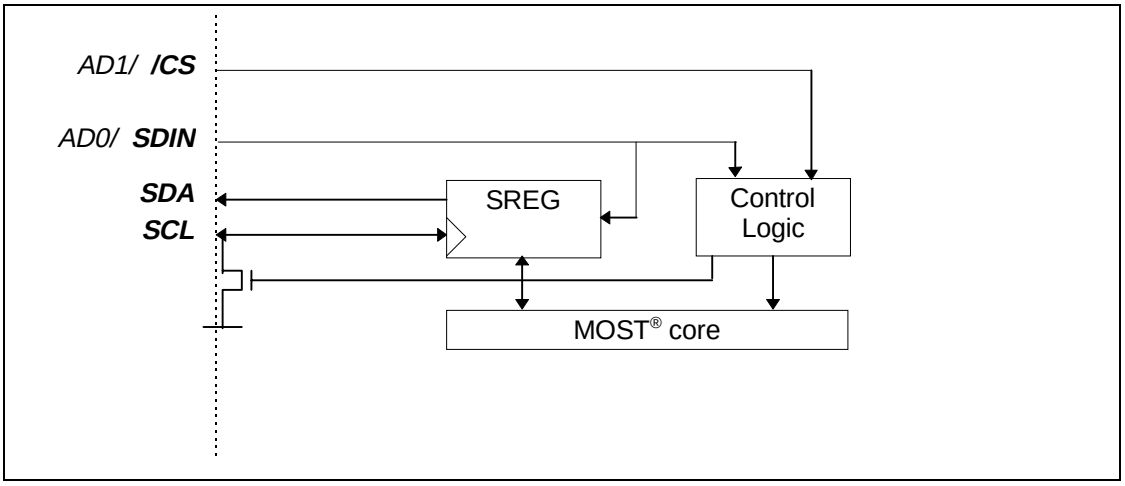


Figure 5-5: Pins used for CP in SPI mode.

Reading and writing is controlled by a unique address, which is defined as 0x40 for writing and as 0x41 for reading. The timing diagram for SPI communication is in Section 18.4.5 on page 157.

5.2.1 Writing To Control Port

The beginning of the transmission is marked by a falling edge at */CS*. Then 0x40 must be sent, followed by the MAP byte. Each byte which is sent after that sequence is written to the memory locations starting with the MAP address, until */CS* is set to high again. The target address for the memory location is automatically incremented:

	Address	MAP	DATA1	...	DATAn	
CS↓	0x40	7..0	7..0	...	7..0	*CS*↑

Figure 5-6: SPI Transmission Scheme

Similar to I^2C mode, the default memory page for writing or reading is page 0. To switch to another memory page, memory location 0xFF must be written with the desired page number (0 through 3). This special address is reserved for page switching purpose in every page.

5.2.2 Reading From Control Port

The reading from CP in SPI mode is similar to the writing to CP. The main difference is that there must be two separate cycles. One transmits the MAP byte (memory address to read from) to the chip and the second reads the data. The sequence is:

	Address	MAP			Address	DATA1	...	DATAn	
CS↓	0x40	7..0	*CS*↑	*CS*↓	0x41	7..0	...	7..0	*CS*↑

Figure 5-7: Scheme for reading from CP via SPI

Data output will continue until */CS* is set high again. After address 0x41 was transferred to the chip, any data on SDIN is treated as don't care.

6 Network Interface

The MOST transceiver interfaces to the MOST network. The *RX* and *TX* receive and transmit pins are TTL compatible and can be directly interfaced with fiber optic receiver/transmitter devices (FOT Units).

The main task of the network interface is to decode and encode the bit stream. The master clock is recovered from the bit stream, and the node is synchronized on a Frame and Block level. The clock recovery and synchronization is accomplished by a high-precision Phase Lock Loop (PLL) in combination with the data decoding circuit.

The network interface can be configured as the timing-master or a timing-slave mode. Only one timing-master node exists in the network with the rest being timing-slave nodes. The timing-master node operates from its own clock source that can be derived from a crystal oscillator or an external clock. All the timing-slave nodes use a recovered clock from the network (RX pin) as the internal master clock. In a network, one single master node generates the timing for all other nodes (slaves).

6.1 MOST Frame Structure

Data on the MOST network are organized in frames. Frame generation is provided by a single device in the system, also referred to as the *timing-master* or *frame generator*. Although this can be any device in the network it would typically be a control unit or human-machine interface (HMI) device.

The MOST frame structure provides maximum flexibility in terms of compatibility with a number of existing communication and data transport requirements, without any drawbacks in implementation cost or processing overhead. Built-in structures allow simple network management on the lowest layers, avoiding overhead and cost shortcomings.

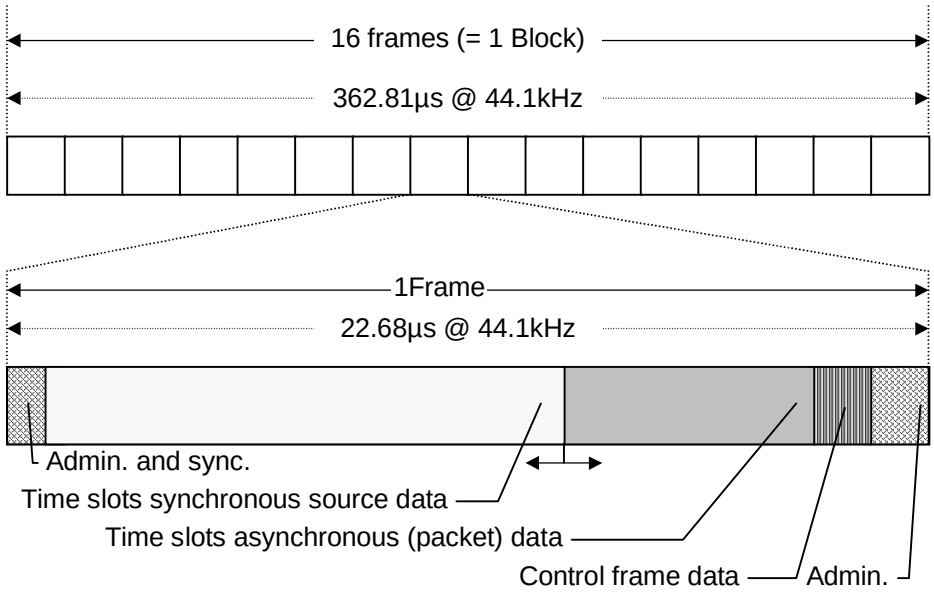


Figure 6-1: General MOST Frame Structure

In addition to a small amount of administrative data, each MOST frame transports two bytes of control data, along with 60 byte of source data. A unit of 16 MOST frames is called "Block". Since each frame transports two bytes of control data, a block transports in total 32 bytes of control data, defined as a "Control message frame" or "Control frame".

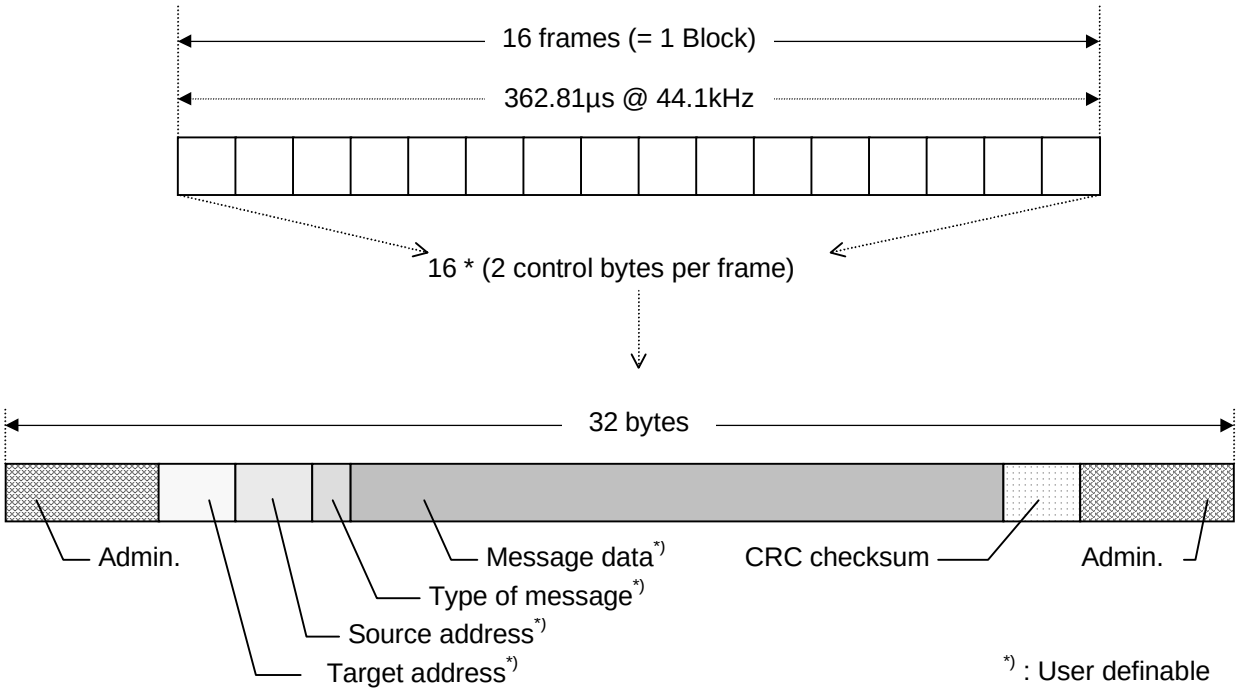


Figure 6-2: Control Message Frame

Within a control frame, Control messages are transported. For more information on Control messages, see Section 13 on page 107.

6.2 Network Configuration

The MOST network-related functions of the OS8104 are controlled via the following set of registers:

- bXCR
  Transceiver Control Register. Control of bypasses, output enable, master/slave mode
- bXSR
  Transceiver Status Register. Controls reporting of errors and reports error events
- bSBC
  Source Bandwidth Control. Controls number of bytes used for synchronous source data transfer vs. number of bytes used for asynchronous packet data transfer.
- bNDR
  Node Delay Register. Reports source data delay between timing master and local node.
- bNPR
  Node Position Register. Reports physical position of a node in the MOST network.
- bMPR
  Maximum Position Register. Reports total number of nodes in the MOST network.
- bMDR
  Maximum Delay Register. Reports total source data delay in the MOST network.

All registers are accessible via the Control Port. While bXCR, bXSR, and bSBC are written by application, the contents of bNDR, bNPR, bMPR, and bMDR are calculated automatically. The values in bNDR, bNPR, bMPR, and bMDR are only valid when the network is in a locked state.

6.2.1 bXCR (Transceiver Control Register)

bXCR controls the main MOST network-related functions of the OS8104. After reset/power up reset, the transceiver comes up as a timing-slave in 'all-bypass' mode where received data (at the RX pin) is directly passed to the TX pin, unaltered.

0x80	bXCR	Transceiver Control Register	
Bit	**Name**	**Description**	**Default**
7	MTR	Master/Slave select	0
6	OE	Transmitter output enable	0
5	rsvd	Write as 1	1
4	LPW	Low power wake-up	0
3	SAN	Standalone mode; Write as 0	0
2	SBY	Source data bypass	0
1	/ABY	All bypass	0
0	/REN	RMCK enable	0

Table 6-1: bXCR (Transceiver Control Register)

MTR (Timing Master/Slave select)

When set to 1, the transceiver acts as the timing-master for the whole network. When the node is configured as a timing-master, it also handles the resource allocation, network supervision, etc.

OE (Transmitter output enable)

If the transceiver is configured as an active part in the network (/ABY=1), OE controls the *TX* output, where OE set to 1 enables the transmitter pin and OE set to 0 forces TX low.

LPW (Low power wake-up)

This bit is available only if MTR is set to 1 (in the timing-master). When LPW is set to 1, the transceiver will generate a wake-up signal to the network that wakes up all nodes which are in low-power mode. This bit is write only and always reads as 0.

SAN (Stand-Alone mode)

SAN read as 1 indicates that the chip is running in Stand-Alone mode (described in Section 17 on page 141). When writing to this register, SAN must not be modified (written as 1 in Stand-Alone mode or written as 0 otherwise). Stand-Alone mode is set via hardware at power-up by tying the /RD and /WR pins low at reset.

SBY (Source Data Bypass)

Nodes that do not "source" or process source data to be sent out onto the network can set this bit to 1. This will minimize the delay through the network. When SBY is set to zero, the node increments the system's delay counter (bNDR) by one and the delay through the node for synchronous source data is two frames.

/ABY (All-bypass mode enable)

When set to 0, the signal received at *RX* from the network is directly connected to the TX pin and is unaltered by the node. This keeps the overall locking time in the network small since all PLLs come up simultaneously and also ensures that only nodes with a running application can enter into the network. To disable the *all-bypass* mode and enter into the network, bit /ABY must be set to 1, where the node increments the Node Position Register (bNPR).

/REN (RMCK enable)

When /REN is set to 0, the RMCK pin outputs a clock based on the RD2..0 bits in bCM1. When /REN is set to 1, the RMCK output is high impedance. Changing /REN can cause glitches on RMCK.

6.2.2 bXSR (Transceiver Status Register)

Register bXSR indicates errors and status of the MOST network. Errors are described in more detail in Section 6.2.4.

0x81	bXSR	Transceiver Status Register	
Bit	**Name**	**Description**	**Default**
7	rsvd	Reserved. Write as 0	0
6	MSL	Mask S/PDIF lock error	1
5	MXL	Mask transceiver lock error	0
4	ME	Mask coding error	1
3	ERR	All error capture	0
2	rsvd	Reserved. Write as 0	0
1	ESL	Error capture "S/PDIF lock error"	0
0	EXL	Error capture "transceiver lock error"	0

Table 6-2: bXSR (Transceiver Status Register)

MSL (Mask S/PDIF-lock error)
: If set to 1, bit ERR will not be set on occurrence of an S/PDIF lock error.

MXL (Mask transceiver-lock error)
: If set to 1, bit ERR will not be set on occurrence of a transceiver lock error.

ME (Mask coding error)
: If set to 1, bit ERR will not be set on occurrence of a coding (bi-phase or parity) error.

ERR (All error capture)
: ERR will be set to 1 by the transceiver whenever an unmasked-error has occurred (based on MSL, MXL, and ME). To generate an interrupt (/INT going low), the IERR bit in bIE must be set to 1. The ERR bit is sticky and must be written to 0, to clear it. The *ERROR* pin is not sticky and only indicates an error for the length of the event causing the error (see Figure 6-3).

ESL (Error capture S/PDIF)
: This bit flags the occurrence of S/PDIF lock errors. ESL is sticky and can be cleared/re-triggered for a new capture by writing a zero to it. If multiple events are connected to the ERR bit, the ESL and EXL bits can help determine the cause of ERR being set to 1.

EXL (Error capture transceiver)
: This bit flags the occurrence of transceiver lock errors. EXL is sticky and can be cleared/re-triggered for a new capture by writing a zero to it. If multiple events are connected to the ERR bit, the ESL and EXL bits can help determine the cause of ERR being set to 1.

6.2.3 bXSR2 (Transceiver Status Register 2)

0x97	bXSR2	Transceiver Status Register 2	
Bit	**Name**	**Description**	**Default**
7..2	rsvd	Reserved. Write as 0	000000
1	INV	RX Inversion Control. Sets the polarity of the network received data (RX), which affects the polarity of pulse width distortion. This value should be optimized for the respective FOT device used.	0
0	rsvd	Reserved. Write as 0	0

Table 6-3: bXSR2 (Transceiver Status Register 2)

6.2.4 Capturing Error Events

Bit ERR in bXSR is set when the transceiver obtains lock or when S/PDIF or line-coding errors occur. Each of these events can be masked out by an individual mask bit (MSL, MXL and ME). Figure 6-3 shows how the mask bits, the *ERROR* pin, and the respective flags are related.

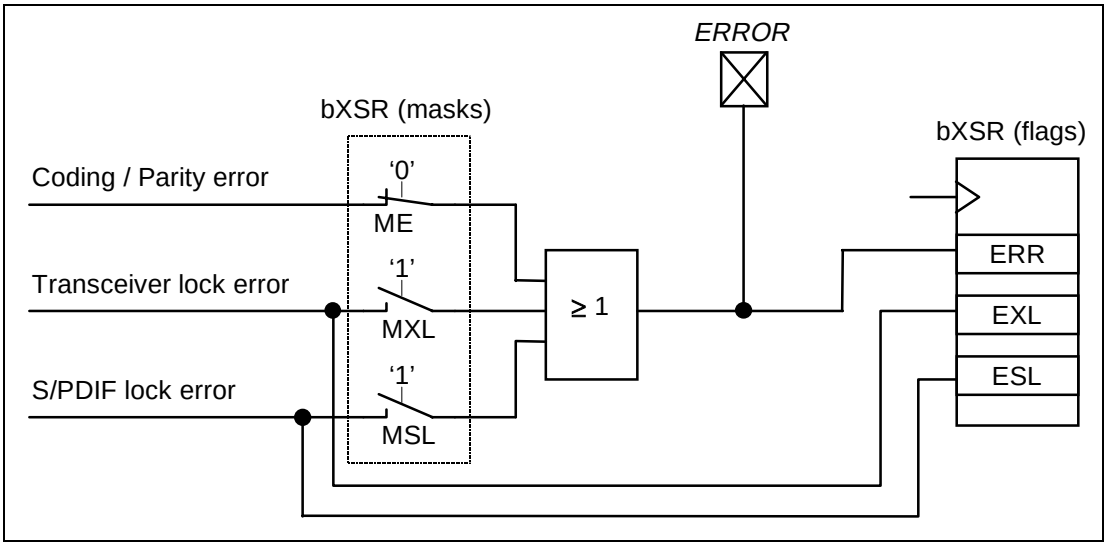


Figure 6-3: Error Flags and Error Masks

If the ERR bit was set due to the occurrence of an error, it can be cleared for a new capture by writing a 0 to it. The same applies to ESL and EXL. An interrupt is generated when ERR is set to 1 if the IERR bit in the interrupt enable register bIE is set to 1 (see Section 11.1 on page 95). This error interrupt can be reset by writing a 1 to bit RERR in bMSGC.

If the interrupt service routine is finished before the error event disappears, another interrupt will be generated. It is possible either to:

- Mask the respective error event and then clear the ERR bit.
  or to:
- Clear the respective interrupt bit in bIE and then clear the ERR bit.

The *ERROR* pin is active for the period of time an error event lasts. Short pulses can occur on the *ERROR* pin. Since the ERR bit is sticky, applications may find it more convenient to use than the *ERROR* pin.

6.2.5 bSBC (Synchronous Bandwidth Control register)

In each MOST frame, 60 bytes are available for transporting source data. Those bytes are referred to as "physical channels". The physical channels are organized in units of four, called "quadlets". Therefore, the OS8104 provides 15 quadlets of source data. Two main services use the source data: Synchronous source data transfer, and Packet data transfer. The bSBC register determines how the quadlets are divided between the two services.

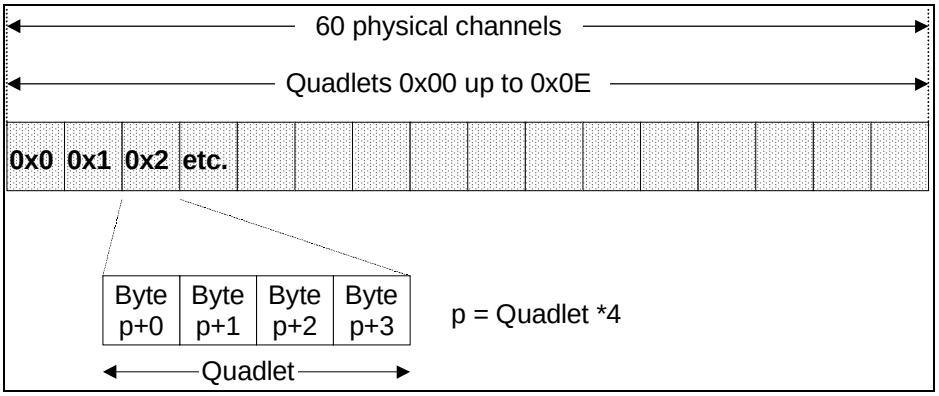


Figure 6-4: Source data organized in quadlets

The division between the two transfer types is controlled by the timing-master node by specifying the number of quadlets used for synchronous data transfer, in the bSBC register The bSBC register in timing-slave nodes will be updated automatically.

Between 6 and 15 quadlets can be assigned to synchronous data transfer, providing anywhere from 24 to 60 physical channels for synchronous data transfer. Although the OS8104 supports all 15 quadlets reserved for Synchronous data, the MOST Specification requires that at least one quadlet be reserved for Packet data. The number of quadlets left for asynchronous data transfer is calculated as follows:

Number of asynchronous quadlets = 15 - contents of bSBC

When assigning the maximum of 60 physical channels (15 quadlets) to synchronous source data transfer, no asynchronous data transfer is possible.

The organization of physical channels in quadlets is only relevant for specifying the bSBC value. Each physical channel assigned to synchronous source data transfer is independent from each other and can be allocated individually for resource management or "routing" of synchronous data. The synchronous source bytes are numbered from 0x00 up to 0x3B (if all channels are assigned to synchronous source data transfer).

The byte number is calculated by using the following formula:

Byte Number = (Number of quadlet *4) + Number of Byte within quadlet

The maximum number of asynchronous bytes per frame is 36 (9 quadlets). The minimum number of synchronous quadlets is 6.

During normal operating conditions, bSBC will be written once during the general initialization phase of the MOST network. Every time bSBC is changed, the complete allocation mechanism of the network must be re-initialized by sending a "De-allocate All" message to the timing master. This also applies to reset or power-up reset. For more information about de-allocating network resources see Section 13.5.2.5.

0x96	bSBC	Synchronous Bandwidth Control	
Bit	**Name**	**Description**	**Default**
7..4	rsvd	Reserved. Write as 0	0x0
3..0	SBC3..0	Number of synchronous quadlets. Read-only when MTR = 0 (timing-slave device). Read/write when MTR = 1	0x0

Table 6-4: bSBC (Synchronous Bandwidth Control register)

Immediately after reset, every node is configured as slave. A slave node extracts bSBC information from the incoming MOST frame. The value in bSBC is only valid if the chip is in lock state. If the node is configured as the timing master, the application has to set bSBC to an appropriate value.

The default value in bSBC is not valid. The timing-master node must initial SBC3..0 to a value between 6 and 15. (The MOST Specification only allows up to 14.)

6.2.6 bNDR (Node Delay Register)

0x8F	bNDR	Node Delay Register	
Bit	**Name**	**Description**	**Default**
7..0	NDR7..0	Network node delay (read-only)	0x00

Table 6-5: bNDR (Node Delay Register)

The Node Delay register determines how many node delays are between this node and the master node, for synchronous source data. The node delay value in the timing-master is 0 and is incremented by one in every node that has SBY set to 0. In addition, the node will delay synchronous source data by two frames before passing it to the next node. If the SBY bit of a node is set to 1, the node will not send source data to the network. Therefore, the received source data will be directly retransmitted incurring no frame delays, and bNDR will not be incremented. bNDR is only valid after lock is established and network initialization has finished.

6.2.7 bNPR (Node Position Register)

0x87	bNPR	Node Position Register	
Bit	**Name**	**Description**	**Default**
7..0	NPR7..0	Network node position (read-only)	0x00

Table 6-6: bNPR (Node Position Register)

The Node Position register indicates the physical position of a node in the network. The node position value in the timing-master node is 0. The timing-slave node connected to the TX output of the master node has a bNPR of 1. Nodes in all-bypass mode are invisible to the network and do not increment bNPR. bNPR is only valid after lock is established and network initialization has finished.

6.2.8 bMPR (Maximum Position Register)

0x90	bMPR	Maximum Position Register	
Bit	**Name**	**Description**	**Default**
7..0	MPR7..0	Number of nodes in the network (read-only)	0x00

Table 6-7: bMPR (Maximum Position Register)

The Maximum Position register indicates the total number of nodes in the MOST network (not counting nodes in all-bypass mode). If this register changes, the ALC bit in bMSGS is set to 1 and an interrupt is generated, if IALC bit in bIE is set. bMPR is only valid after lock is established and network initialization has finished.

6.2.9 bMDR (Maximum Delay Register)

0x91	bMDR	Maximum Delay Register	
Bit	**Name**	**Description**	**Default**
7..0	MDR7..0	Total delay in the network (read-only)	0x00

Table 6-8: bMDR (Maximum Delay Register)

The Maximum Delay Register indicates the total number of node delays, for synchronous source data, within the network. If this register changes, the ALC bit in bMSGS is set to 1 and an interrupt is generated, if IALC bit in bIE is set. bMDR is only valid after lock is established and network initialization has finished.

6.2.10 Network Registers After Lock

The contents of bNPR, bNDR, bMPR, bMDR are updated automatically after the network achieves lock. Table 6-9 indicates when the register contents will be valid.

Register	Best Case	Worst Case
bNPR	46 µs after lock of local chip	92 µs after lock of local chip
bNDR	46 µs after lock of local chip	24 ms after lock of local chip
bMPR	24 ms (after lock of complete network)	Timing Master: 24 ms Slave: 47 ms (after lock of complete network)
bMDR	24 ms (after lock of complete network)	Timing Master: 24 ms Slave: 47 ms (after lock of complete network)

Network Frame rate (Fs) = 44.1 kHz

Table 6-9: Network Register Update Times

7 Source Data Ports in Serial Mode

The term *Source Data* refers to any data which is transmitted, transported, and received in a continuous stream, meeting real-time requirements. A typical application for source data is audio data transmitted from an A/D converter to an amplifier. Therefore, the hardware interface to external applications is called the *Source Data Ports*.

The OS8104 can receive and transmit data between external applications and the MOST network simultaneously. The SRn pins (n = 0 to 3) receive source data externally for transmission onto the network, and the SXn pins transmit data to external applications from the network receiver. The Source Data Ports support many different data formats and can operate in serial or parallel modes. Serial formats such as I^2S, Matsushita, Sony, and S/PDIF are supported. In parallel mode, both synchronous source data and asynchronous packet data can be transferred. The parallel access mode of the Source Data Ports is described in Section 8.

For transporting any data via Source Data interface, the chip must be in lock state. In Un-Lock state, synchronization to the network is lost; therefore, no source data exchange is possible. The external interface itself is still active and does not need to be deactivated by the application.

7.1 General Description

The Source Data Ports can be used for synchronous or asynchronous data input and output. The synchronous format supports the highest bandwidth and highest efficiency, while requiring the smallest amount of external memory.

7.1.1 Source Data Port Registers

Source data control registers select the serial mode format for the Source Data Port.

- bSDC1
  Source Data Control register 1. Control of polarity of different interface signals, muting, S/PDIF enable.
- bSDC2
  Source Data Control register 2. Control of clock rates (SCK, S/PDIF, Transparent channel, Multi speed FSY).
- bSDC3
  Source Data Control register 2. Special S/PDIF control (Sync Source, and I/O direction in S/PDIF 8x mode).

7.1.2 Serial Source Port Interface

The Source Data Ports are typically connected to multimedia sources and/or sinks that handle audio, video or CD data streams. The SR0..3 pins receive data from external sources. The SX0..3 pins transmit data to external sources. The SCK pin is the bit clock for all four inputs and all four outputs. The FSY pin is the Frame Sync delineating the word/channel (left vs. right channel) boundaries for all the data pins. FSY and SCK can be configured as outputs, or inputs.

For I^2S format compatibility, a delay bit in the source data control register is available to shift the data relative to *FSY* by one bit. A variety of clock modes are selectable by control register as well. Possible clock modes are 8/16/24/32/48/64/128/256 times the sampling frequency (Fs).

7.1.3 Internal Structure of Serial Source Data Interface

Each Source Port data pin is connected to an 8-bit shift register. Figure 7-1 shows how the shift registers are arranged.

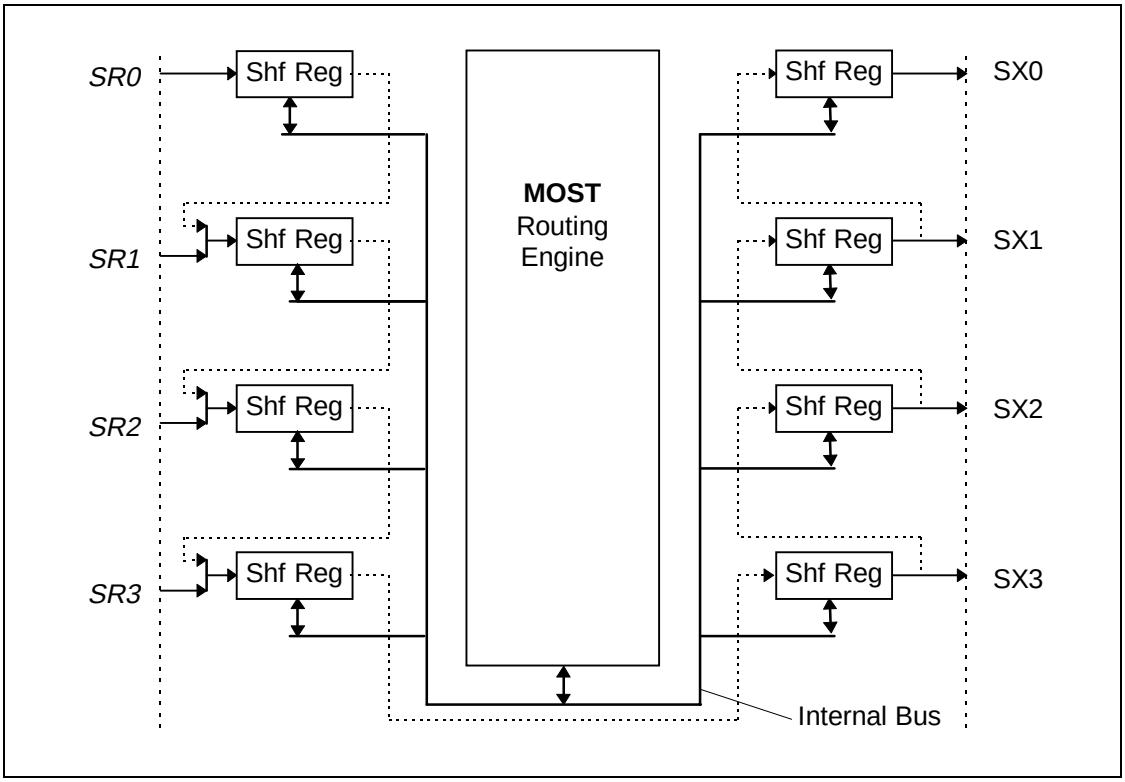


Figure 7-1: Source Data Port Block Diagram (serial mode).

To support higher data rates, the shift registers can be cascaded. The possible connections for cascading are drawn as a dotted line in the Figure above. A typical bit rate for serial Source Ports is 64Fs, wherein *SCK* runs 64 times faster than the MOST network frames. Therefore, up to 64 bits per frame are shifted into each shift register. The internal MOST routing engine will read each Source Port eight times (when set to 64Fs) per frame to get all 64 bits.

Figure 7-2 shows the waveform of a synchronous serial data transfer for one port running at 64Fs:

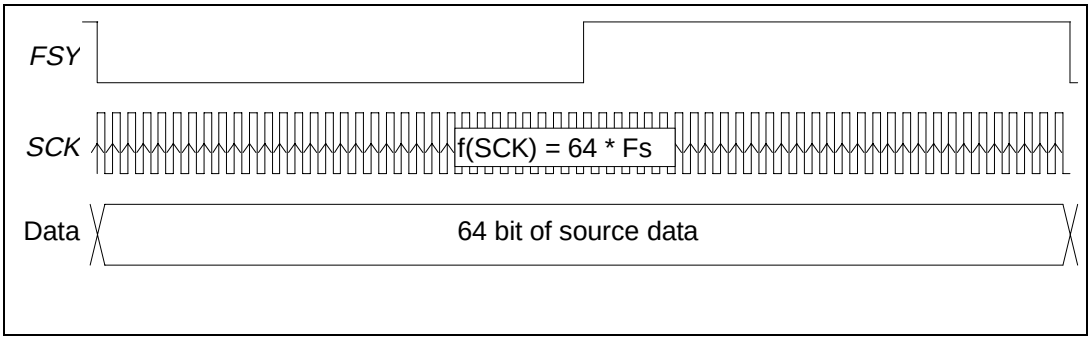


Figure 7-2: Source Data Port at 64Fs

When cascading shift registers, the number of available Source Port pins will be reduced. Cascading all the shift registers produces a 64-bit shift register. Since the Source Port is read eight times per Fs interval, 512 bits can be shifted into or out of the chip. In this mode, only one Source Data Port pin is

available in one direction. Either SR0 can input the 512 bits or SX0 can output the 512 bits. This mode is only available when SR0/SX0 is configured for S/PDIF mode running at 8x standard speed.

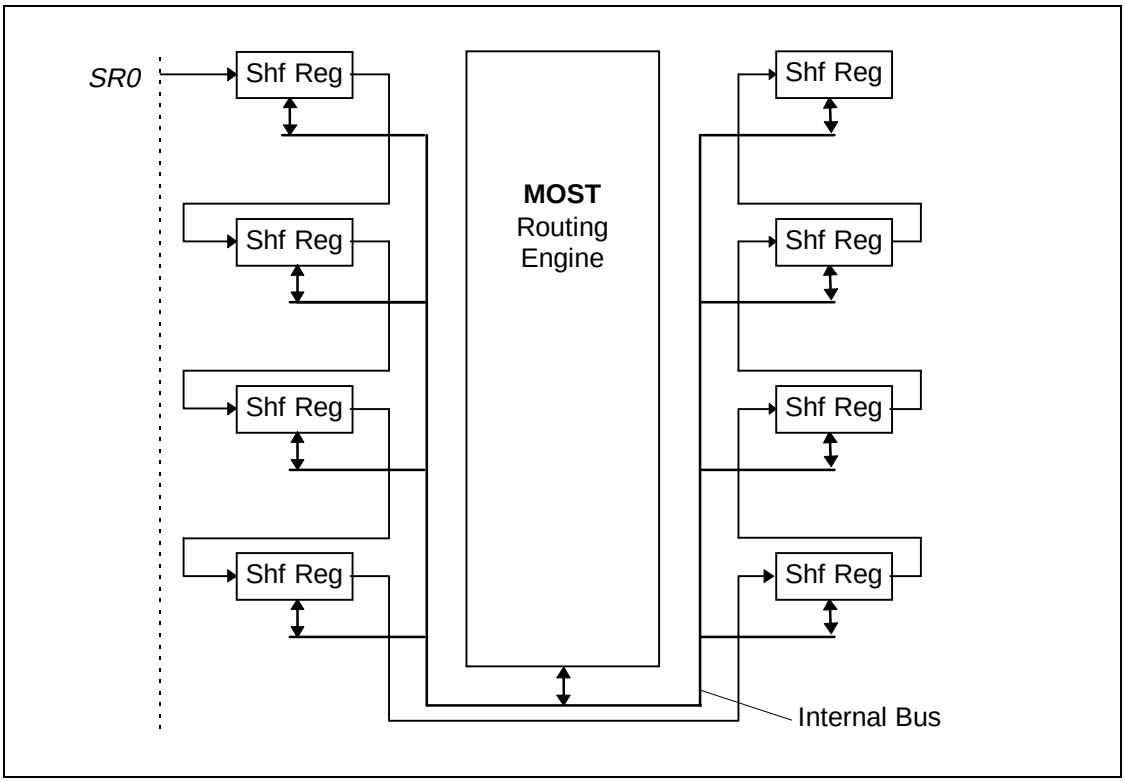


Figure 7-3: Source Data Port Cascaded for 512Fs Input mode.

7.1.4 S/PDIF (IEC 60958-3) Data Transport

As mentioned above, *SR0* and *SX0* can be configured for S/PDIF (IEC-60958 parts 1 and 3) format. These pins can decode and encode the bi-phase data, handle the preambles, and generate an S/PDIF data stream, based on internal timing. SR0 and SX0 can operate at single to octal the standard S/PDIF data rate (Fs).

In S/PDIF mode, *SX0* outputs the block start preamble by receiving a block bit from the received bit stream or by automatic insertion in case the block bit is not in the bit stream (i.e. received data stream is not S/PDIF).

7.1.5 Transparent Data Transport

Two of the Source Data Port pins (*SR1*, *SX1*) can be switched to transparent mode. In this mode, the input port SR1 samples the incoming signal at regular intervals. The sample rate depends on the current clock rate and an additional scale factor. By adjusting clock rate and scale factor, the appropriate sample rate for a given application may be chosen. The over-sampling ratio can be programmed with respect to the signal's frequency and the maximum acceptable jitter on the receiver side's *SX1* port. The following diagram shows the effect of the sampling on a transmitted signal.

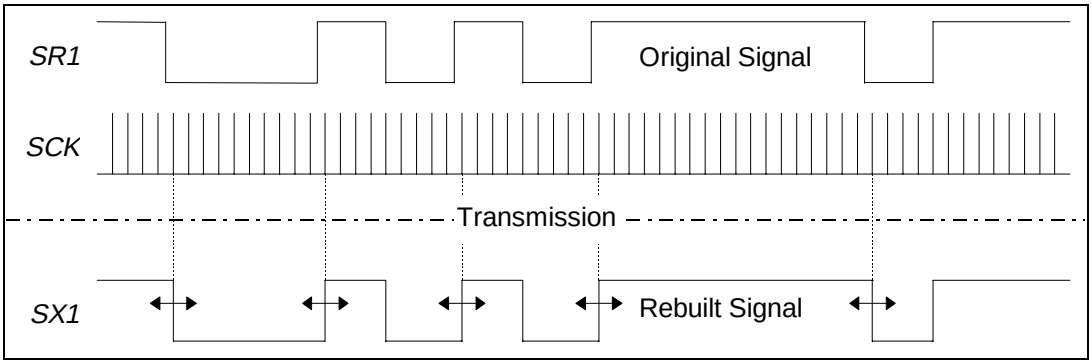


Figure 7-4: Transparent Signal Transmission

The network source data channel capacity used depends on the over-sampling ratio. Over-sampling ratios between 1Fs and 64Fs can be selected. This transparent transport mechanism can handle different application protocols and formats such as RS232, DAB, ITTS, GSM AT, GPS, ISDN, and UNILINK.

Since the sampled data is handled via the standard SR1 and SX1 ports, it can be routed onto and off of the network similar to the other Source Ports.

7.2 Source Data Port Configuration (Serial)

The Source Ports must be configured during the initialization procedure after the chip is reset.

The general configuration, i.e. whether the Source Data Ports work in serial or parallel mode, is to be done by hardware during the reset procedure. When transmitting data onto the MOST network, the MSB is transmitted first and when transmitting multi-byte words, the most significant byte is transmitted first. In addition, when transmitting stereo data, left is generally sent before right.

7.2.1 bSDC1 (Source Data Control Register 1)

The Source Data Ports of the MOST transceiver chip support a variety of serial data formats. The main register for controlling the serial data format is called Source Data Control Register 1 (bSDC1).

0x82	bSDC1	Source Data Control 1 Register	
Bit	**Name**	**Description**	**Default**
7	EDG	Active edge of *SCK*	0
6	DEL	Delay first bit against *FSY*	0
5	POL	Polarity of *FSY*	0
4	I/O	Input/Output select of *FSY* and *SCK*	0
3	NBR	Number of *SCK* cycles per frame	0
2	SPD	S/PDIF port enable	0
1	/MT	Mute source data outputs	0
0	/TCE	Transparent channel enable	0

Table 7-1: bSDC1 (Source Data Control Register 1)

EDG (Active edge of *SCK*)

When set to 0, SRn pins are sampled on the falling edge of SCK and SXn pins transition on the rising edge. When set to 1, SRn pins are sampled on the rising edge of SCK, and SXn pins transition on the falling edge.

DEL (Delay first bit against *FSY*)

If set to 1, there is a one *SCK* cycle delay between the change of *FSY* (Frame Sync) and the first data bit sampled. Such a delay is used in I^2S format. When DEL is set to 0, the first data bit is at the FSY edge (no delay).

POL (Polarity of *FSY*)

If set to 1, a high level at signal *FSY* is interpreted as indicator for a left sample. When set to 0, a left sample is indicated by a low level at *FSY*.

I/O (Input/Output select of pins *FSY* and *SCK*)

If SPD is set to 1, I/O is a don't care (*FSY* and *SCK* are forced to output). If SPD is 0, setting I/O to 0, configures the *FSY* and *SCK* pins as inputs. If SPD is 0, setting I/O to 1, configures the *FSY* and *SCK* pins as outputs.

NBR (Number of *SCK* cycles per frame)

This bit is only used if the clock rate of the Source Data Ports is set to 64Fs (SPR2..0 in register bSDC2). If SPR2..0 is 64Fs and NBR is 0, SCK is set to 64Fs. If SPR2..0 is 64Fs and NBR is 1, SCK is set to 48Fs. When SCK is an output at 48Fs, the actual clock frequency is 64Fs with 8 clock cycles removed. In each half of the FSY cycle, the third (out of four) set of 8 clocks are removed. See Figure 7-5. When SCK is an input at 48Fs, it must be continuous (non-gated).

SPD (S/PDIF port enable)

When set to 1, the SR0 and SX0 pins switch to S/PDIF format. The S/PDIF speed is selected by SDR1..0 in register bSDC2. When SPD is 1, the *FSY* and *SCK* are forced to outputs. SPD set to 0 enables standard serial modes for SR0 and SX0.

/MT (Mute Source Port outputs)

When set to 0, SX0..3 pins are forced low. When /MT is set to 1, the SXn pins output data.

/TCE (Transparent Channel enable)

When set to 0, SR1 and SX1 are configured for transparent mode operation. Therefore, the signal at SR1 is sampled with a frequency calculated as follows:

fsample = (current *SCK* clock rate) / div

where div = 1, 2, 4 or 8. The scale factor div is specified by the TCR1..0 bits in register bSDC2.

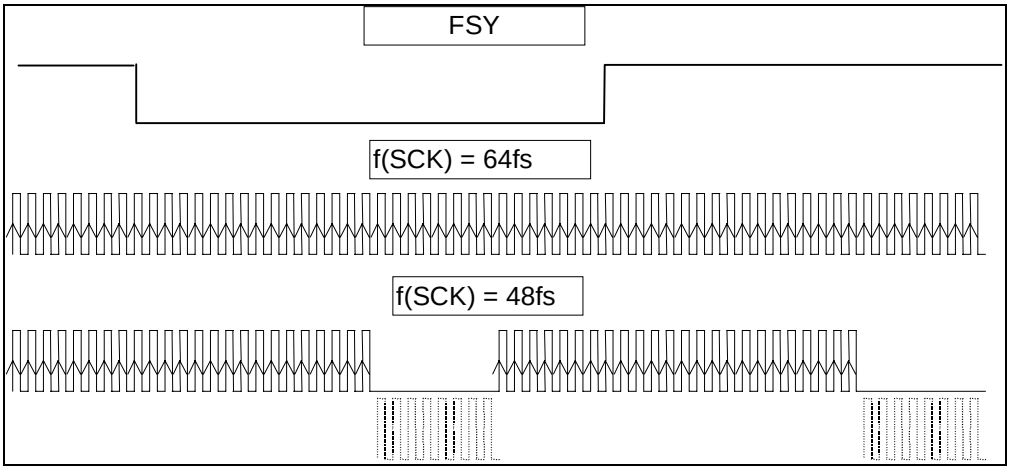


Figure 7-5: Clock gating for SCK = 48Fs

If SCK is configured as input and NBR is set to 1, SCK must be a continuous clock at 48Fs. The gated clock format used to output a 48Fs clock cannot be used.

7.2.1.1 I²S (Philips) Source Data Format

For selecting I²S format, bSDC1 must be configured as shown in the table below (irrelevant bits omitted). Figure 7-6 shows a sample waveform.

bSDC1 bit:	EDG	DEL	POL	I/O	NBR
Value	1	1	0	1	0

Table 7-2: Selecting I²S (Philips) source data format

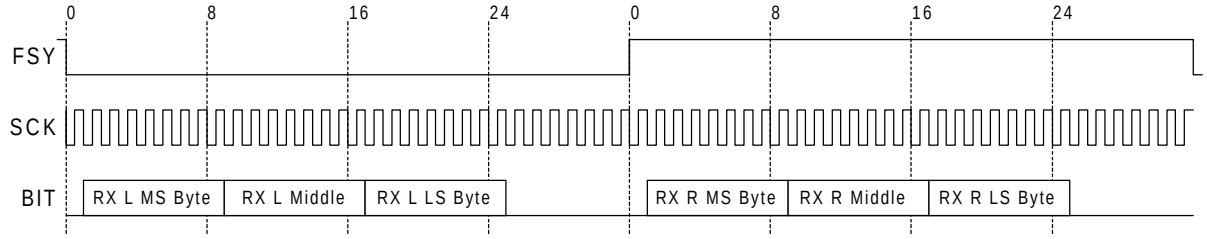


Figure 7-6: I²S source data format

In the above Figure, data is arranged MSB first, high byte first.

7.2.1.2 SONY Source Data Format

For selecting SONY format, bSDC1 must be configured as shown in the table below (irrelevant bits omitted). Figure 7-7 shows a sample waveform. The usage of NBR implies a *SCK* rate of 64Fs.

bSDC1 bit:	EDG	DEL	POL	I/O	NBR
Value	1	0	1	1	1

Table 7-3: Selecting SONY source data format

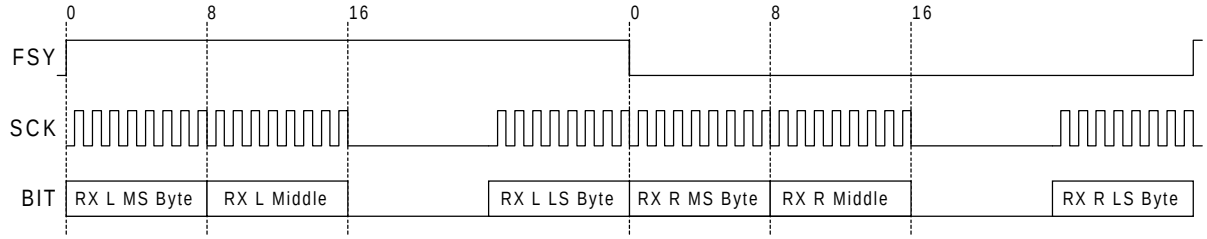


Figure 7-7: Sony source data format

In the above diagram, data is arranged MSB first, high byte first.

7.2.1.3 Matsushita Source Data Format

For selecting Matsushita format, bSDC1 must be configured as shown in the table below (irrelevant bits omitted). Figure 7-8 shows a sample waveform.

bSDC1 bit:	EDG	DEL	POL	I/O	NBR
Value	1	0	1	1	0

Table 7-4: Selecting Matsushita Source Data Format

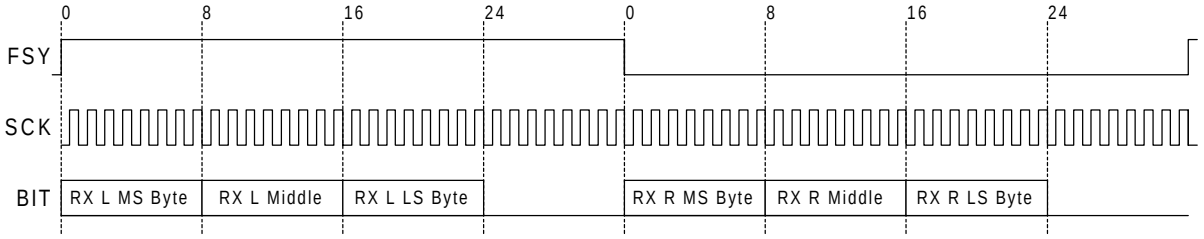


Figure 7-8: Matsushita source data format

In the above diagram, data is arranged MSB first, high byte first.

7.2.2 bSDC2 (Source Data Control Register 2)

Register bSDC2 contains the control bits for Source Data Port *SCK* rate, Transparent Channel Clock rate, Multi-speed FSY, and the S/PDIF speed rate.

0x8C	bSDC2	Source Data Control 2 Register	
Bit	**Name**	**Description**	**Default**
7..5	SPR2..0	Source port *SCK* rate	011
4	MFSY	Multi-speed FSY enable	0
3..2	TCR1..0	Transparent channel clock rate	00
1..0	SDR1..0	S/PDIF speed rate	00

Table 7-5: bSDC2 (Source Data Control Register 2)

SPR2..0 (*SCK* rate)

These bits select the *SCK* clock (bit) rate of the Source Data Ports in serial mode. On higher clock rates than 64Fs, the shift registers in the Source Data Ports are cascaded, thus decreasing the number of active Source Data Port pins. When the OS8104 is configured for Parallel-Synchronous or Parallel-Combined mode, SPR2..0 must be set to 101.

000 – 8Fs
001 – 16Fs
010 – 32Fs
011 – 64Fs
100 – 128Fs. Ports SR1, SX1, SR3, and SX3 are not available.
101 – 256Fs. Ports SR1..3 and ports SX1..3 are not available.

MFSY (Multi-speed FSY enable)

This bit depends on the *SCK* clock rate (set via SPR2..0), and is only enabled at 128Fs or 256Fs when *FSY* is an output. At SCK rates below 128Fs, the *FSY* output is always 1Fs. As an input, *FSY* must always be 1Fs.
If MFSY is set to 1 and SCK rate is set to 128Fs, FSY is 2Fs.
If MFSY is set to 1 and SCK rate is set to 256Fs, FSY is 4Fs.

TCR1..0 (Transparent Channel SCK divider)

These bits select the sample rate of SX1 and SR1 when configured in transparent mode (/TCE in register bSDC1 set to 0). The bits TCR1..0 control a clock divider which scales down the currently selected *SCK* clock rate (SPR2..0). Transparent mode is not available when SPR2..0 is 128Fs or 256Fs.
00 – SCK divided by 1
01 – SCK divided by 2
10 – SCK divided by 4
11 – SCK divided by 8

SDR1..0 (S/PDIF speed rate)

These bits specify the speed (and bit rate) at which Source Data Port 0 is running when configured in S/PDIF mode (Bit SPD in register bSDC1 set to 1). On higher clock rates than 64Fs (double-speed S/PDIF and faster), the shift registers in the Source Data Ports are cascaded, thus decreasing the number of active Source Data Port pins.

00 – 64Fs (1x).
01 – 128Fs (2x). Ports SR1 and SX1 are not available.
10 – 256Fs (4x). Ports SR1..3 and SX1..3 are not available.
11 – 512Fs (8x). Ports SR1..3 and SX1..3 are not available. In addition, the S/PDIF stream is in only one direction (in or out), specified by SIO in bSDC3.

7.2.3 bSDC3 (Source Data Control Register 3)

Register bSDC3 contains the control bits for S/PDIF operation.

0x8D	bSDC3	Source Data Control 3 Register	
Bit	**Name**	**Description**	**Default**
7	SIO	S/PDIF ‘in’ or ‘out’ in 8x mode	0
6..4	rsvd	Reserved; Write as 0	0
3	SPS	S/PDIF sync source	0
2..0	rsvd	Reserved; Write as 0	0

Table 7-6: bSDC3 (Source Data Control Register 3)

SIO (S/PDIF ‘in’ or ‘out’ in 8x mode)

When 8xS/PDIF is selected (SDR1..0 = 11), SIO set to 1 supports S/PDIF data in on SR0, and SIO set to 0 supports S/PDIF data out on SX0.

SPS (S/PDIF sync source)

When set to 0, the S/PDIF output is synchronized to the S/PDIF input data stream. For more information, refer to the Section titled *Synchronizing to S/PDIF*.

7.2.4 Serial Source Data Port Modes

There are eleven serial modes available, which differ in bit rates and number of available ports. In the following descriptions "Port 0" means SR0 and SX0, except where otherwise stated. The table below shows an overview of the eleven serial modes:

Mode	Comment	
1	Ports 0,1,2,3	in serial port format; Clock/bit rate 8Fs, 16Fs, 32Fs or 64Fs
2	Ports 0,2,3 Port 1	in serial port format; Clock/bit rate 8Fs, 16Fs, 32Fs or 64Fs in transparent mode; Sample rate = (Clock/bit rate) / div [div = 1,2,4 or 8]
3	Ports 0, 2	in serial port format; Clock/bit rate 128Fs
4	Port 0	in serial port format; Clock/bit rate 256Fs
5	Port 0 Ports 1,2,3	in S/PDIF format (1x speed) in serial port format; Clock/bit rate 8Fs, 16Fs, 32Fs or 64Fs
6	Port 0 Port 1 Ports 2,3	in S/PDIF format (1x speed) in transparent mode; Sample rate = (Clock/bit rate) / div [div = 1,2,4 or 8] in serial port format; Clock/bit rate 8Fs, 16Fs, 32Fs or 64Fs
7	reserved	
8	Port 0 Ports 2,3	in S/PDIF format (2x speed) in serial port format; Clock/bit rate 8Fs, 16Fs, 32Fs or 64Fs
9	Port 0 Ports 2	in S/PDIF format (2x speed) in serial port format; Clock/bit rate 128Fs
10	Port 0	in S/PDIF format (4x speed)
11	Port 0	in S/PDIF format (8x speed) as input (SR0 only)
12	Port 0	in S/PDIF format (8x speed) as output (SX0 only)

Table 7-7: Serial Source Data Port Modes

The term "serial port format" stands for an industry-standard format like I^2S or Sony, configured using bits in bSDC1.

For the Serial Source Data modes described below, Table 7-8 lists the symbols and abbreviations used in the descriptions.

Symbol	Comment
N_{nnnFs}	Represents a serial port format (I^2S, Sony,...selected by bSDC1) and the clock/bit rate 'nnn' the port is running on
N_a	Represents a serial port format (I^2S, Sony,... selected by bSDC1). The clock/bit rate of the port is determined by 'a': SPR2..0 000 001 010 011 a = 8Fs 16Fs 32Fs 64Fs
S_{nx}	S/PDIF format at a certain speed where n = 1, 2, 4, or 8
T_b	represents a transparent format with a sample rate of clock/bit rate divided by 'b': TCR1..0 00 01 10 11 b = 1 2 4 8
---	port not available
x	don't care

Table 7-8: Symbols and Abbreviations

In addition to the clock rates listed (8Fs, 16Fs, 32Fs and 64Fs), a 48Fs clock rate is supported by setting SPR2..0 in register bSDC2 to 64Fs, and setting the NBR bit in register bSBC to 1.

The mapping of the respective bytes from the serial Source Port pins to and from the MOST network is described in Section 12 starting on page 97.

7.2.4.1 Mode 1

Source Port Mode 1									
Ports				Control Registers					
SR/X0	SR/X1	SR/X2	SR/X3	bSDC1		bSDC2			bSDC3
				SPD	TCE	SPR2..0	TCR1..0	SDR1..0	SIO
N(a)	N(a)	N(a)	N(a)	0	0	**a**	x	x	x

Table 7-9: Source Port Mode 1 Register Settings

In Mode 1, all four Source Ports are running at the same serial data format and clock rate. The selection of a serial data format is described in Section 7.2.1 on page 44. N(a) indicates that the clock rate is specified by setting the bits SPR2..0 in bSDC2:

a	8Fs	16Fs	32Fs	64Fs
SPR2..0:	000	001	010	011

Table 7-10: Source Port Mode 1 SCK Rates

7.2.4.2 Mode 2

Source Port Mode 2									
Ports				Control Registers					
SR/X0	SR/X1	SR/X2	SR/X3	bSDC1		bSDC2			bSDC3
				SPD	TCE	SPR2..0	TCR1..0	SDR1..0	SIO
Na	Tb	Na	Na	0	1	**a**	**b**	x	x

Table 7-11: Source Port Mode 2 Register Settings

In Mode 2, the Source Ports 0, 2 and 3 are running at the same serial data format and clock rate. The selection of a serial data format is described in Section 7.2.1 on page 44. N(a) indicates that the clock rate is specified by setting the bits SPR2..0 in bSDC2:

a	8Fs	16Fs	32Fs	64Fs
SPR2..0:	000	001	010	011

Table 7-12: Source Port Mode 2 SCK Rates

Source Port 1 is running in transparent mode. In this mode, its sample clock rate is set via bits SPR2..0 and TCR1..0 in register bSDC2:

Scale factor b	1	2	4	8
TCR1..0:	00	01	10	11

Table 7-13: Source Port Mode 2 Transparent Channel Clock Rate

The sample clock rate is calculated a follows:

sample clock rate = a/b

For an overview of transparent mode, refer to Section 7.1.5 on page 43.

7.2.4.3 Mode 3

Source Port Mode 3									
Ports				Control Registers					
SR/X0	SR/X1	SR/X2	SR/X3	bSDC1		bSDC2			bSDC3
				SPD	TCE	SPR2..0	TCR1..0	SDR1..0	SIO
N128Fs	---	N128Fs	---	0	x	100	x	x	x

Table 7-14: Source Port Mode 3 Register Settings

In Mode 3, Source Ports 0 and 2 are running at the same serial data format and at a fixed clock rate of 128Fs. The selection of a serial data format is described in Section 7.2.1 on page 44. Source Ports 1 and 3 are not available in this mode. This mode is enabled by setting SPD to 0 in register bSDC1, and setting bits SPR2..0 to 100 in register bSDC2.

7.2.4.4 Mode 4

Source Port Mode 4									
Ports				Control Registers					
SR/X0	SR/X1	SR/X2	SR/X3	bSDC1		bSDC2			bSDC3
				SPD	TCE	SPR2..0	TCR1..0	SDR1..0	SIO
N256Fs	---	---	---	0	x	101	x	x	x

Table 7-15: Source Port Mode 4 Register Settings

Mode 4 supports only one Source Port (*SR0*/*SX0*) running at a clock rate of 256Fs. The selection of a serial data format is described in Section 7.2.1 on page 44. Source Ports 1 and 3 are not available in this mode. This mode is enalbed by writing a 0 to bit SPD in register bSDC1, and 101 to the bits SPR2..0 in register bSDC2.

7.2.4.5 Mode 5

Source Port Mode 5									
Ports				Control Registers					
SR/X0	SR/X1	SR/X2	SR/X3	bSDC1		bSDC2			bSDC3
				SPD	TCE	SPR2..0	TCR1..0	SDR1..0	SIO
S1x	Na	Na	Na	1	0	**a**	x	00	x

Table 7-16: Source Port Mode 5 Register Settings

In Mode 5, Source Port 0 runs in S/PDIF format, at single S/PDIF speed. Source Ports 1 through 3 run at the same serial data format and clock rate. The selection of a serial data format is described in Section 7.2.1 on page 44. N(a) indicates that the clock rate is specified by setting the bits SPR2..0 in bSDC2:

a	8Fs	16Fs	32Fs	64Fs
SPR2:0:	000	001	010	011

Table 7-17: Source Port Mode 5 SCK Rates

7.2.4.6 Mode 6

Source Port Mode 6									
Ports				Control Registers					
SR/X0	SR/X1	SR/X2	SR/X3	bSDC1		bSDC2			bSDC3
				SPD	TCE	SPR2..0	TCR1..0	SDR1..0	SIO
S1x	Tb	Na	Na	1	1	**a**	**b**	00	x

Table 7-18: Source Port Mode 6 Register Settings

In Mode 6, Source Port 0 runs in S/PDIF format, at single S/PDIF speed. Source Ports 2 and 3 run at the same serial data format and clock rate. The selection of a serial data format is described in Section 7.2.1 on page 44. N(a) indicates that the clock rate is specified by setting the bits SPR2..0 in bSDC2:

a	8Fs	16Fs	32Fs	64Fs
SPR2..0:	000	001	010	011

Table 7-19: Source Port Mode 6 SCK Rates

Source Port 1 is running in transparent mode, where the sample clock rate is determined by SPR2..0 and TCR1..0 in register bSDC2:

Scale factor b	1	2	4	8
TCR1..0:	00	01	10	11

Table 7-20: Source Port Mode 6 Transparent Channel Clock Rate

The sample clock rate is calculated a follows:

sample clock rate = a/b

For an overview of Source Port in transparent mode, refer to Section 7.1.5 on page 43.

7.2.4.7 Mode 8

Source Port Mode 8									
Ports				Control Registers					
SR/X0	SR/X1	SR/X2	SR/X3	bSDC1		bSDC2			bSDC3
				SPD	TCE	SPR2..0	TCR1..0	SDR1..0	SIO
S2x	---	Na	Na	1	x	**a**	x	01	x

Table 7-21: Source Port Mode 8 Register Settings

In Mode 8, Source Port 0 runs in S/PDIF format at double S/PDIF speed. Source Port 1 is not available in this mode. Source Ports 2 and 3 run at the same serial data format and clock rate. The selection of a serial data format is described in Section 7.2.1 on page 44. N(a) indicates that the clock rate is determined via bits SPR2..0 in bSDC2:

a	8Fs	16Fs	32Fs	64Fs
SPR2..0:	000	001	010	011

Table 7-22: Source Port Mode 8 SCK Rates

7.2.4.8 Mode 9

Source Port Mode 9									
Ports				Control Registers					
SR/X0	SR/X1	SR/X2	SR/X3	bSDC1		bSDC2			bSDC3
				SPD	TCE	SPR2..0	TCR1..0	SDR1..0	SIO
S2x	---	N128Fs	---	1	x	100	x	01	x

Table 7-23: Source Port Mode 9 Register Settings

In Mode 9, Source Port 0 runs in S/PDIF format, at double S/PDIF speed. Source Ports 1 and 3 are not available in this mode. Source Port 2 runs at a standard serial data format and a fixed clock rate of 128Fs. The selection of a serial data format is described in Section 7.2.1 on page 44.

7.2.4.9 Mode 10

Source Port Mode 10									
Ports				Control Registers					
SR/X0	SR/X1	SR/X2	SR/X3	bSDC1		bSDC2			bSDC3
				SPD	TCE	SPR2..0	TCR1..0	SDR1..0	SIO
S4x	---	---	---	1	x	x	x	10	x

Table 7-24: Source Port Mode 10 Register Settings

In Mode 10, Source Port 0 runs in S/PDIF format, at quadruple S/PDIF speed. Source Ports 1 through 3 are unavailable in this mode.

7.2.4.10 Mode 11

Source Port Mode 11									
Ports				Control Registers					
SR/X0	SR/X1	SR/X2	SR/X3	bSDC1		bSDC2			bSDC3
				SPD	TCE	SPR2..0	TCR1..0	SDR1..0	SIO
S8x in	---	---	---	1	x	x	x	11	0

Table 7-25: Source Port Mode 11 Register Settings

In Mode 11, Source Port 0 runs in S/PDIF format, at octal S/PDIF speed. Source Ports 1 through 3 are unavailable in this mode. In addition, Source Port 0 can only receive S/PDIF data on SR0. SX0 is not available.

7.2.4.11 Mode 12

Source Port Mode 12									
Ports				Control Registers					
SR/X0	SR/X1	SR/X2	SR/X3	bSDC1		bSDC2			bSDC3
				SPD	TCE	SPR2..0	TCR1..0	SDR1..0	SIO
S8x out	---	---	---	1	x	x	x	11	1

Table 7-26: Source Port Mode 12 Register Settings

In Mode 12, Source Port 0 runs in S/PDIF format, at octal S/PDIF speed. Source Ports 1 through 3 are unavailable in this mode. In addition, Source Port 0 can only transmit S/PDIF data on SX0. SR0 is not available.

7.2.5 Mode Configuration Register Overview

Source Port Format Combinations										
	Ports				Control Registers					
Mode	SR / SX0	SR / SX1	SR / SX2	SR / SX3	bSDC1		bSDC2			bSDC3
					SPD	TCE	SPR2..0	TCR1..0	SDR1..0	SIO
1	Na	Na	Na	Na	0	1	**a**	x	x	x
2	Na	Tb	Na	Na	0	0	**a**	**b**	x	x
3	N128Fs	---	N128Fs	---	0	x	100	x	x	x
4	N256Fs	---	---	---	0	x	101	x	x	x
5	S1x	Na	Na	Na	1	1	**a**	x	00	x
6	S1x	Tb	Na	Na	1	0	**a**	**b**	00	x
8	S2x	---	Na	Na	1	x	**a**	x	01	x
9	S2x	---	N128Fs	---	1	x	100	x	01	x
10	S4x	---	---	---	1	x	x	x	10	x
11	S8x in	---	---	---	1	x	x	x	11	0
12	S8x out	---	---	---	1	x	x	x	11	1

NnnnFs represents a serial port format (I2S, Sony,...selected by bSDC1) and the clock/bit rate the port is running on

Na represents a serial port format (I2S, Sony,...selected by bSDC1) and the clock/bit rate of the port is determined by

a	8Fs	16Fs	32Fs	64Fs
a	000	001	010	011

Snx represents S/PDIF format with a certain speed

T_b represents a transparent format with a sample rate of clock/bit rate divided by **b**, there b can be

b	1	2	4	8
b	000	001	010	011

--- port not available

x don't care

Table 7-27: Serial Source Data Port Modes Versus Registers

7.2.6 S/PDIF (IEC-60958)

The OS8104 can handle S/PDIF data from single speed mode up to 8x speed mode through the Source Data Ports *SR0* and *SX0*.

S/PDIF mode is enabled by setting the SPD bit in register bSDC1 to 1. When S/PDIF is enabled, the pins *FSY* and *SCK* are always configured as outputs. The diagram below shows single speed S/PDIF data, and how *FSY* is synchronized to it:

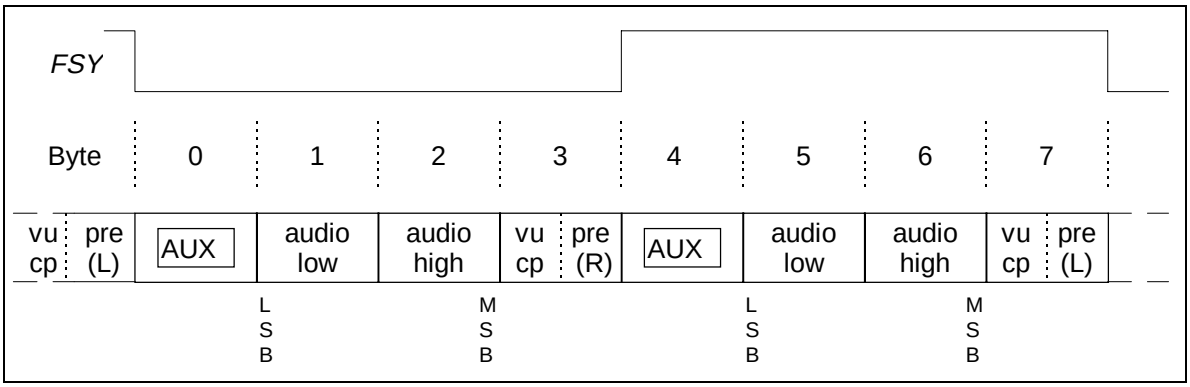


Figure 7-9: Standard S/PDIF Data Stream

Audio data in S/PDIF format is transported LSB first. The OS8104 changes the sequence of bits automatically to MSB first, when receiving S/PDIF data via SR0. However, the data byte order is Low Byte/High Byte in the input buffer, so the contents of RE must be adapted to change sequence on a byte level. Incoming single-speed S/PDIF data is handled like any serial data received at 64Fs SCK clock rate. The respective address references can be derived from the Tables in Section 12 on page 97.

The VUC bits of the S/PDIF data stream, and the respective preambles, are received in the same manner as the other data. This makes it possible to receive S/PDIF data, transport the entire stream via the MOST network, and restore the S/PDIF signal at a different node.

In total there are three ways to process S/PDIF data in OS8104:

- S/PDIF data to S/PDIF data
  Receiving S/PDIF data, transmitting the entire data stream to any node and restoring of the original S/PDIF data stream at the destination node and outputting it on SX0 in SPDIF mode.
- S/PDIF data to non-S/PDIF data
  Receiving S/PDIF data, and transmitting only the audio related parts of the data stream.
- Non-S/PDIF data to S/PDIF data
  Routing any non-S/PDIF audio data to SX0 and output the data as S/PDIF data stream. Dummy VUC and S/PDIF timing are generated by OS8104 automatically.

7.2.6.1 Synchronizing To S/PDIF

If a node is the timing-master (MTR set to 1), its PLL can be synchronized to the incoming S/PDIF data stream. Since the master node generates the timing for the entire network, all nodes will be synchronized to that S/PDIF data stream too.

As an alternative, the master's PLL can be locked to a crystal connected to the *XTI*/*XTO* pins. In this case, the source of S/PDIF data at the input of the master node must be synchronized to the master's clock by using the *RMCK* output pin of OS8104.

If a node is a timing-slave, the network input must be chosen as the synchronization source. In this case, a S/PDIF device connected to this node must always be synchronized to the network by using the *RMCK* output of the OS8104.

Setting bit SPS in register bSDC3 to 1 separates the internal timing generator from the S/PDIF data stream at *SR0*. Otherwise the timing is synchronized to the S/PDIF data stream at *SR0*.

Bit SPS can solve two basic problems:

- If an external S/PDIF device synchronizes its output to the incoming S/PDIF data and the same is done within the MOST transceiver, a closed loop is generated. No S/PDIF transfer is possible since both devices keep trying to resynchronize their block positions. This closed-loop is avoided by setting SPS to 1.
- A corrupt signal at the S/PDIF input will corrupt S/PDIF output as well. Switching synchronization from S/PDIF input to internal synchronization, by setting SPS to 1, solves this problem as well.

The following diagram shows the function of SPS:

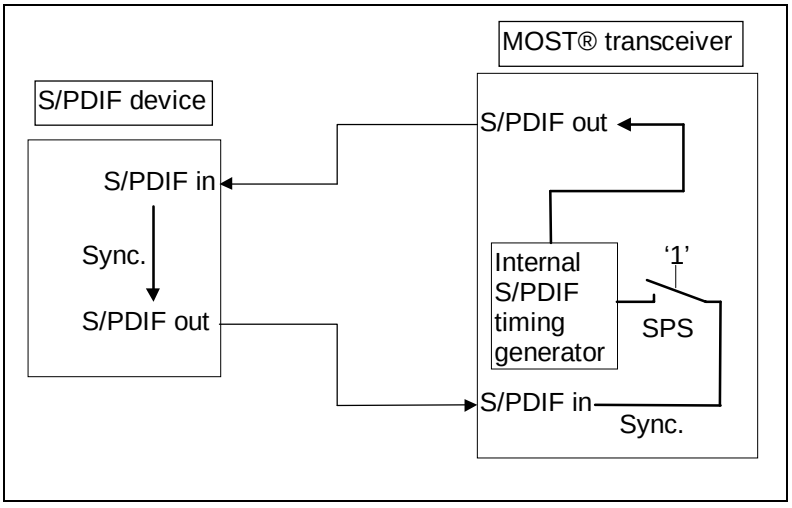


Figure 7-10: SPS bit Block Diagram

7.2.6.2 S/PDIF Speed Modes

The OS8104 supports several different S/PDIF speeds. The following table lists the possible S/PDIF speed selections, selected through bits SDR1..0 in register bSDC2:

SDR1	SDR0	Speed	Bytes/Frame	Clock rate [Fs]
0	0	1x (64 bits/frame)	8	64
0	1	2x (128 bits/frame)	16	128
1	0	4x (256 bits/frame)	32	256
1	1	8x (512 bits/frame)	64	512

Table 7-28: S/PDIF Mode Speeds

If 8x-S/PDIF mode is selected, all the Source Port shift registers (in both directions) are cascaded. Therefore, only one direction can be supported (input or output), selected via bit SIO in register bSDC3. A 0 configures *SR0* as S/PDIF "input only" and a 1 configures *SX0* as S/PDIF "output only". SIO is not used for S/PDIF speeds slower than 8x, where *SR0* and *SX0* both are available as source data ports. Routing information for S/PDIF data can be found in tables in Section 12 on page 97.

In S/PDIF modes, the FSY signal is aligned to the first data byte after the left preamble as illustrated in Figure 7-11.

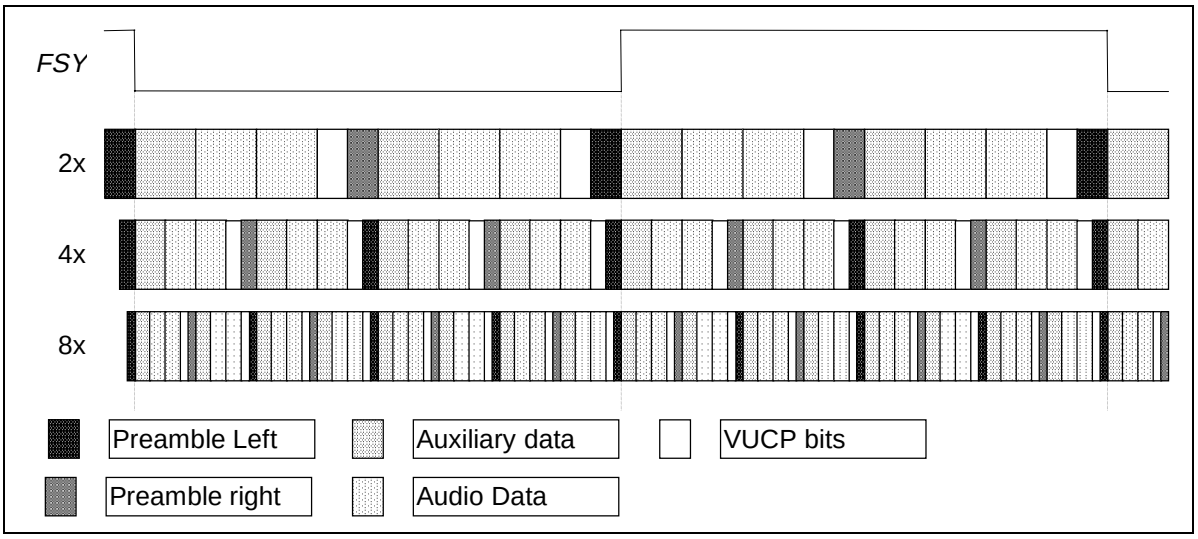


Figure 7-11: FSY- S/PDIF Alignment

7.2.6.3 S/PDIF Data To S/PDIF Data

When transmitting S/PDIF data on a MOST network, the VUC data bits and the preambles must be transmitted, if the final data output at the receiver node is also in the S/PDIF format.

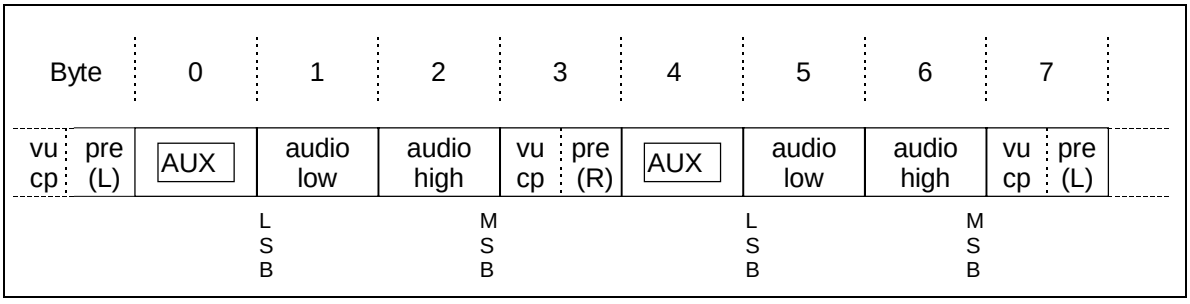


Figure 7-12: Sequence of audio bytes in S/PDIF (2 times 16-bit stereo audio)

At the target node, the correct sequence of bytes must be restored. Although changing the order of audio data bytes is not absolutely necessary when sending pure S/PDIF data from an S/PDIF source to an S/PDIF sink, the byte order, within a word or channel, should be reversed on the MOST network to conform to the MOST standard. Then any other node may also use the data traversing the MOST network, even if they are not outputting S/PDIF streams. The MOST network standard is left channel before right (same as S/PDIF). However, the MOST standard is MSbyte first, which is opposite to S/PDIF. This byte-reversal is handled easily through reconfiguration of the MOST routing engine. As previously mentioned, the bit reversal of each byte in the S/PDIF stream is handled automatically by the OS8104 (S/PDIF data bytes are LSB first and the MOST standard is MSB first).

7.2.6.4 S/PDIF Data To Non-S/PDIF Data

When building a MOST standard data stream based upon data from a source data port in S/PDIF mode, the preambles and VUC bits can be omitted. In that case, only the bytes listed below must be transmitted in the given sequence:

- (byte2, byte1) and (byte6, byte5) (16 bit audio)
- (byte2, byte1, byte0) and (byte6, byte5, byte4) (24 bit audio)

The byte numbers refer to Figure 7-12.

7.2.6.5 Non-S/PDIF Data To S/PDIF Data

When building an S/PDIF data stream based upon audio data coming from a Source Data Port (running in one of the standard MOST modes), the S/PDIF specific preambles (left, right and block) and VUC are not transmitted with the data. In this case, the OS8104 automatically adds these bits to the outgoing S/PDIF stream.

The non-S/PDIF data must be routed to the respective positions in Source Port *SX0,* by configuring the Routing Engine appropriately. For setting the VUC bits to a defined value of 000, the address reference 0xF8 must be placed at the respective location in the RE. 0xF8 is a special Routing Engine address reference that outputs 0x00 for that particular byte (see Section 12 for more information on the Routing Engine).

8 Parallel Access

Both the Source Data Port (SP) and the Control Port (CP) can be configured for parallel mode operation. The SP can be configured for parallel mode while CP still operates in serial mode; however, to use the CP in parallel mode, the SP must be in parallel mode.

In parallel mode, the *SR0..3* and *SX0..3* pins form an 8-bit parallel interface:

D7	D6	D5	D4	D3	D2	D1	D0
SX1	*SR1*	*SR3*	*SR2*	*SR0*	*SX3*	*SX2*	*SX0*

Table 8-1: Source Data Port pins configured for 8-bit parallel I/O

The Source Data Port shift registers are re-configured to an 8-byte FIFO. All data is handled via this FIFO, with the exception of reading the status registers bCP and bSP.

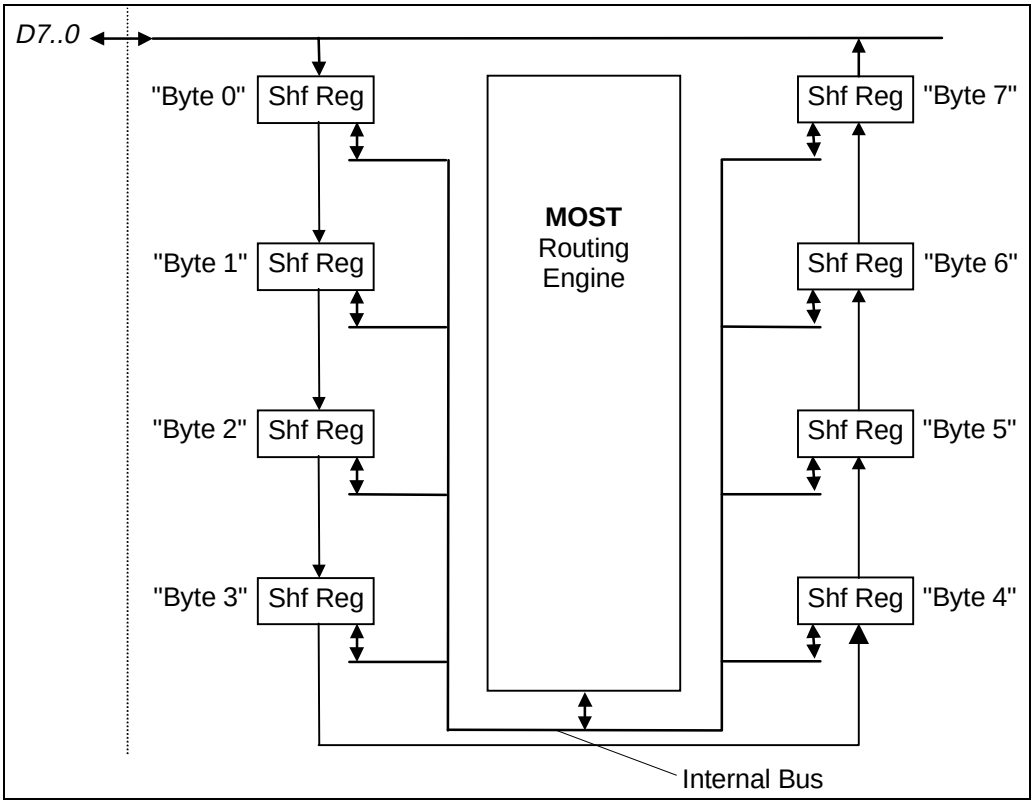


Figure 8-1: OS8104 in Parallel Mode

For source data, there are three parallel configuration options: Parallel-Synchronous, Parallel-Asynchronous, or Parallel-Combined mode. Regardless of the particular Source Port parallel mode, the Control Port can also be configured to use the parallel port (or the Control Port can remain in serial mode).

- Parallel-Synchronous (Stream data)
  - Source data handled in fixed time scheme controlled by *FSY* and *SRC_FLOW*.
  - No packet data handling
- Parallel-Asynchronous (Packet data)
  - No fixed time scheme.
  - SRC_FLOW provides data transfer handshaking (OS8104 busy signal)
  - RAM areas accessible without switching memory pages :
    Registers from 0xE2 up to 0xF2 on page 0
    mARP
    mAXP
  - No source data handling available.
- Parallel-Combined (Packet and Stream)
  - Source and packet data handled in fixed time scheme controlled by *FSY* and *SRC_FLOW*.

All parallel port data formats use a common set of signals and registers. Data flow is controlled by the pins:

- */RD* – Read access
- */WR* – Write access
- *PAD1..0* – address, which determine the type of operation

The Control signals are:

- CP_FLOW (Control Port) – for control message handshaking (Control Port busy)
- SRC_FLOW (Source Port) – Fixed timing scheme for Parallel-Synchronous and Combined modes, and packet data handshaking (port busy) for Parallel-Asynchronous mode.
- FSY (Source Port in Synchronous mode)

Status is available through special status registers that do not reside in the standard memory area of the OS8104 (similar to the other registers). These registers are only available through the parallel interface.

- bCP (Control Port parallel status)
- bSP (Source Port parallel status)

8.1 Control Port in Parallel Mode

The Control Port can only be configured in parallel mode, when the Source Data Port is also configured for parallel mode. Table 4-1 on page 28 shows all available modes. The respective signals for configuring CP in parallel mode are:

/RS	PAR_CP	PAR_SRC	Description
0	x	x	chip is being reset
↑	1	1	CP in parallel mode

Table 8-2: Configuring CP in Parallel Mode

Data flow is controlled by the pins /RD, /WR, and PAD1..0. *PAD1..0* specify the kind of operation to be performed, and */RD* and */WR* specify a read or write operation, respectively. Before the first data can be read or written, a MAP (Memory Address Pointer) byte must be written, which specifies where the following data will be read from or written to.

The signal *CP_FLOW* indicates when the chip is ready to receive/transmit data via the parallel port. The Control Port Status register bCP indicates the current status of the Control Port.

8.1.1 Writing CP MAP Data Register

The CP MAP data register contains the address at which a read or write operation shall start. The MAP value must be written before the first access to the Control Data port.

This operation is selected by setting the control pins as follows:

PAD1	*PAD0*	*/RD*	*/WR*	Selected Operation
0	0	1	↑	Write CP MAP Data Register

Table 8-3: Writing CP MAP Data Register in Parallel Mode

Data is written into the register at the rising edge of */WR*.

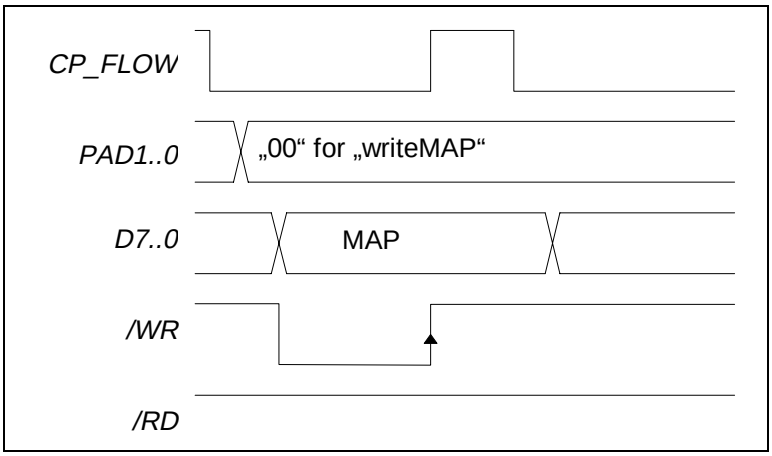


Figure 8-2: CP MAP Data Register Write Timing

The MAP value only needs to be written the first time as it is auto-incremented. As long as data is read from or written to sequential addresses within a memory page, a new MAP is not needed.

8.1.2 Writing To Control Port

Before writing data to CP, the destination address for the data must be specified by writing the MAP byte. The MAP specifies the address within the current memory page (default page is 0). To change to another memory page, the page number (0 to 3) must be written to memory location 0xFF. The write operation to CP is selected by setting the control pins as follows:

PAD1	*PAD0*	*/RD*	*/WR*	Selected Operation
0	1	1	↑	Write CP Data Register

Table 8-4: Writing to CP in Parallel Mode

The data is internally latched at the rising edge of */WR* . The first write cycle after the specification of MAP writes data directly to the target address specified in MAP. Each write access auto-increments MAP, so MAP must be specified only once when transferring larger blocks of data to the same memory page. Figure 8-3 shows a sample data transfer for writing N bytes. Pin /RD is constantly high in this example:

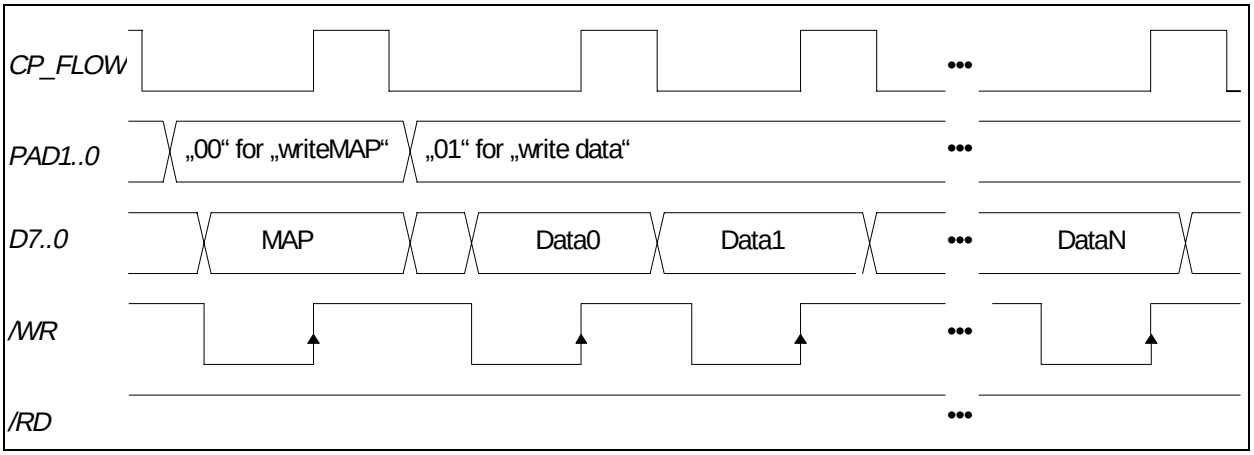


Figure 8-3: CP Write Timing in Parallel Mode

When data is latched CP_FLOW goes high indicating the Control Port is busy. Control Port busy is also indicated by the CP_BUSY2..0 bits in bCP. The Control Port is ready for another access when either *CP_FLOW* returns to zero, or the CP_BUSY2..0 bits in bCP are all zero.

8.1.3 Reading From Control Port

Before reading data from CP, the target address for the data must be specified by writing the MAP byte. The MAP specifies the address within the current memory page (default page is 0). To change to another memory page, the page number (0 to 3) must be written to memory location 0xFF. The read operation is selected by setting the control pins as follows:

PAD1	PAD0	/RD	/WR	Selected Operation
0	1	↓	1	Read CP Data Register

Table 8-5: Reading from CP in Parallel Mode

Figure 8-4 shows the reading of "N" bytes:

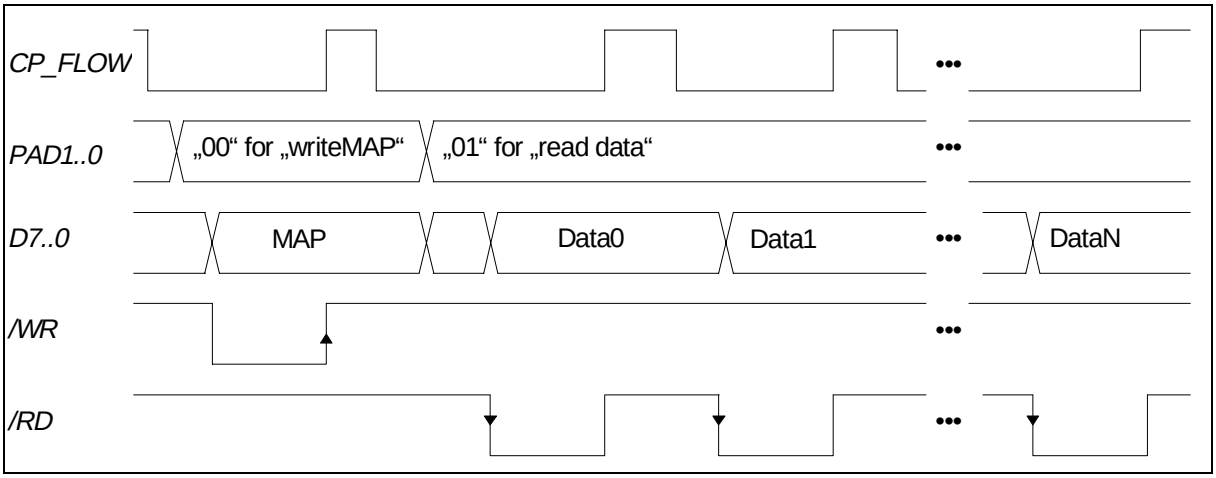


Figure 8-4: CP Read Timing in Parallel Mode

When reading data, *CP_FLOW* goes high indicating the Control Port is busy readying the next data byte. Control Port busy is also indicated by the CP_BUSY2..0 bits in bCP. The Control Port is ready for another access when either *CP_FLOW* returns to zero, or the CP_BUSY2..0 bits in bCP are all zero.

8.1.4 Control Port Status Register (bCP)

Since bCP is handled independently of the FIFO, it is always available for reading, regardless of the state of CP_FLOW. For reading bCP, the following signals must be set:

PAD1	PAD0	/RD	/WR	Selected Operation
0	0	↓	1	Read CP Data Register bCP

Table 8-6: Reading bCP Status in Parallel Mode

	bCP	Control Port Status (in parallel mode, read-only)	
Bit	**Name**	**Description**	**Default**
7	ZP	Zero Power	0
6	LP	Low Power	0
5	rsvd	Reserved.	0
4	AINT	Asynchronous Packet interrupt (opposite of /AINT pin)	0
3	INT	Control and Error interrupt (opposite of /INT pin)	0
2..0	CP_BUSY2..0	Control Port busy (when non-zero)	000

Table 8-7: bCP (Control Port Status)

ZP (Zero Power Mode)
When set to 1, the chip is in zero power mode.

LP (Low Power Mode
When set to 1, the chip is in low power mode.

AINT Status of */AINT* pin used for Packet data Transfer)
When set to 1, indicates either the reception of or the transmission of a data packet. This bit reflects an inverted version of the /AINT pin. If the interrupt occurs, this bit will be set to one while the /AINT pin will be driven low.

INT (Status of */INT* pin used For Control and Error Handling)
When initially powered up, INT set to 1 indicates that the OS8104 is ready to be accessed. External hardware should wait for INT before any other accesses are made to the OS8104. INT also indicates a control message has been received or an error occurred. The bIE register enables various error events that cause INT to be set. INT reflects an inverted version of the /INT pin. If the interrupt occurs, this bit will be set to one while the /INT pin will be driven low.

CP_BUSY2..0
If any of these bits are set to 1, the MOST core is busy and cannot respond to accesses. If all three bits are set to 0, the Control Port is ready for another access.

8.1.5 Control Signal Overview

PAD1	PAD0	/RD	/WR	Register Selection
0	0	↓	1	Read CP Status Register bCP
0	0	1	↑	Write CP MAP Data Register
0	1	↓	1	Read CP Data Register
0	1	1	↑	Write CP Data Register

Table 8-8: Control Signals in Parallel Mode

8.2 Source Port (SP) In Parallel Mode

The Source Port supports three parallel modes, two of which are selected via configuration pins (Table 8-9): Parallel-Synchronous and Parallel-Asynchronous. The third mode, Parallel-Combined, is derived from the Parallel-Synchronous mode and the setting of the APCM in bPCMA (see Section 8.3).

PAR_SRC	ASYNC	Description
0	x	Source Port (and Control Port) in serial mode
1	0	Source Port in Parallel-Synchronous mode (or Parallel-Combined)
1	1	Source Port in Parallel-Asynchronous mode

Table 8-9: Configuration Interface for SP

For Parallel-Asynchronous transfer, *SRC_FLOW* indicates whether the chip is ready to receive/transmit data via parallel port, or not. SRC_FLOW is active whenever the OS8104 is in Lock state. *FSY* provides synchronization with the network frame for Parallel-Synchronous mode. The bSP register provides status for the Source Port when in parallel mode.

> Data is only transported via the Source Port interface when the chip is in the LOCK state. When the chip is in the unlock state, synchronization to the network is lost and no source data exchange is possible (control data can still be transferred).

8.2.1 Parallel-Synchronous mode

In Parallel-Synchronous mode, source data is transferred into and out of the chip via a FIFO. Since synchronous source data is transferred continuously, it follows a fixed timing scheme. The SPR2..0 bits in bSDC2 must be set to 101 for Parallel-Synchronous operation.

Each frame is divided up into eight time intervals, called SF0 up to SF7, which are identified by *SRC_FLOW* (programmed for 8Fs). Within each SF interval, eight bytes of parallel synchronous data can be read and eight bytes can be written (16 bytes transferred in all). Therefore, a total of 64 bytes of synchronous source data can be transferred in each direction during each network frame period. The routing address location associated with each byte is determined by the SF interval and the FIFO byte position. Even if no data is needed for a particular SF interval, at least one read or write access is required for each SF interval. *FSY* going low indicates the start of SF0.

In reset, *SRC_FLOW* is held high and will go low approximately 3 µs after /RS is released. *SRC_FLOW* will stay low until the OS8104 is in lock. Once in lock , SRC_FLOW and FSY determine the particular SF interval. If the OS8104 loses lock, SRC_FLOW will go low and stay low until lock is reacquired.

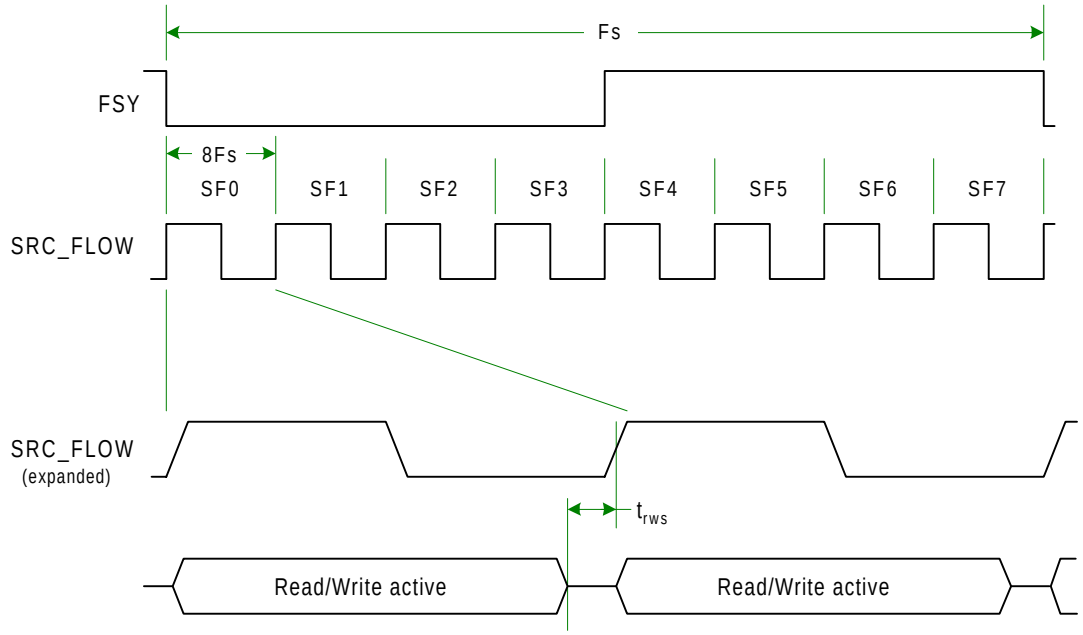


Figure 8-5: Source Port Parallel-Synchronous Mode Timing Overview

Just prior to the beginning of each SF interval , the routing engine fills the 8-byte FIFO with received data. At the rising edge of *SRC_FLOW*, data is ready for reading out of the FIFO. Data must be read from the FIFO before writing the FIFO, as writing to the FIFO overwrites the read data. If received data is not needed, the FIFO can be written immediately after SRC_FLOW rises. There is no requirement to read or write all eight bytes; however, at least one byte must be accessed (read or written) during each SF interval.

At the end of an SF interval, no reading or writing can occur for a brief period of time equal to 1/(512Fs). This time is indicated in Figure 8-5 as t_{rws} and is specified in Section 18.4.2 on page 153.

The FSY signal indicates the start of a frame and aligns the routing engine data with the particular SF interval. SF0 is defined as the first SRC_FLOW interval after FSY falls. The *SRC_FLOW* signal is time-aligned with the *FSY* signal and toggles at an 8Fs rate.

8.2.1.1 Reading from FIFO

For reading data from the FIFO, pins *PAD1* and *PAD0*, as well as */WR* must be set to 1. A falling edge at pin */RD* starts the output of data:

PAD1	*PAD0*	*/RD*	*/WR*	Selected Operation
1	1	↓	1	Read Data from FIFO

Table 8-10: Reading from FIFO in Parallel-Synchronous Mode

The figure below shows the timing diagram for reading from the FIFO.

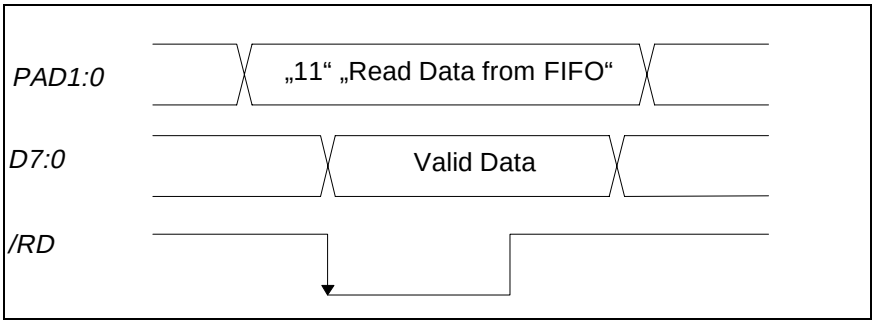


Figure 8-6: Source Port Parallel-Synchronous Mode Read Timing

8.2.1.2 Writing into FIFO

For writing data to the FIFO, pins *PAD1* and *PAD0*, as well as */RD* must be set to 1. A rising edge at pin */WR* stores the data in the FIFO cell:

PAD1	*PAD0*	*/RD*	*/WR*	Selected Operation
1	1	1	↑	Write Data into FIFO

Table 8-11: Writing into FIFO in Parallel-Synchronous Mode

The figure below shows the timing diagram for writing into the FIFO.

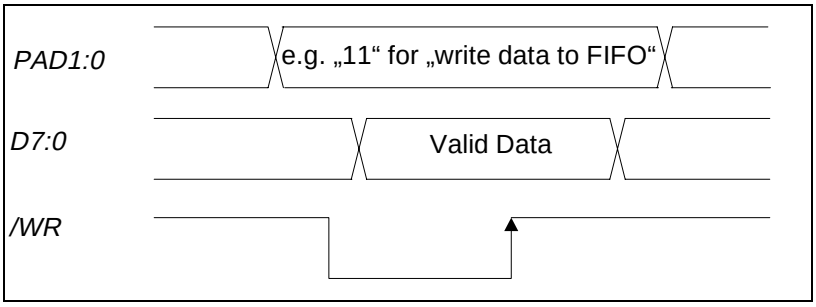


Figure 8-7: Source Port Parallel-Synchronous Mode Write Timing

8.2.2 Parallel-Asynchronous mode

Parallel-Asynchronous data transfer is not restricted to a fixed timing scheme synchronized to the frame on the MOST network. The Parallel-Asynchronous data transfer mode provides access to the RAM area of the OS8104. By sending an appropriate Memory Address Pointer (MAP), the asynchronous transmit and receive buffers, and control registers, may be read or written at a faster speed than can be accomplished through the Control Port. In this mode, *SRC_FLOW* is a hand-shaking signal that indicates when the FIFO is busy, or when it can be accessed.

When in Parallel-Asynchronous data transfer mode, access to synchronous source data is not available. To access both synchronous and asynchronous data, see the Parallel-Combined transfer mode.

With the exception of the status register bSP, eight bytes must always be read or written. When the 8 bytes are transferred, SRC_FLOW goes high. While SRC_FLOW is high no parallel accesses are permitted. When SRC_FLOW goes low, the FIFO is ready for another transfer.

In total, there are three transfer modes available:

- Reading a block of 8 bytes
- Writing a block of 8 bytes
- Writing a single byte (8 bytes written but only one byte stored)

For transmitted packets, /AINT will go low when the packet is completely transferred, the transmit status is available, and the TX buffer, mAXP, is available. For received packets, /AINT will go low when a valid packet is received, with the entire packet available in the Packet receive buffer mARP. A valid received packet is one where the logical address (bNAH/bNAL) or the alternate packet address (bAPAH/bAPAL) is correct, **AND** the message has a valid CRC.

8.2.2.1 Memory Address Pointer (MAP)

The start address must be written before any data accesses can occur. The start address, or memory address pointer (MAP) consists of two bytes: MAP1 and MAP2.

- MAP1:
  Contains the offset within a RAM page (0x00 up to 0xFF).
- MAP2:
  Contains the RAM page (0x00 up to 0x03) and the control bit specifying a single byte transfer (bit 7). Therefore, when performing single byte transfers, 0x80 must be added to the memory page value.

For example, to write the first eight data bytes of the Asynchronous Transmit Packet buffer mAXP (RAM locations 0x1C0 through 0x1F1), the MAP value would be:

MAP1: 0xC0
MAP2: 0x01

Byte(n) in the FIFO will be written to/read from the address location MAP + n. Therefore, byte0 – the first byte entered will be written to location 0x1C0 in the example above.

To write only the first byte of mAXP, the MAP values are:

MAP1: 0xC0
MAP2: 0x81

Even though only the last byte is stored, eight bytes must still be written to the FIFO due to the hardware handshaking mechanism.

8.2.2.2 Writing to FIFO

When *SRC_FLOW* changes from high to low, the FIFO is ready for being accessed. As with reading from the FIFO, eight bytes must always be transferred even if only the last byte is actually stored. Figure 8-8 illustrates the general signal flow for writing to the FIFO.

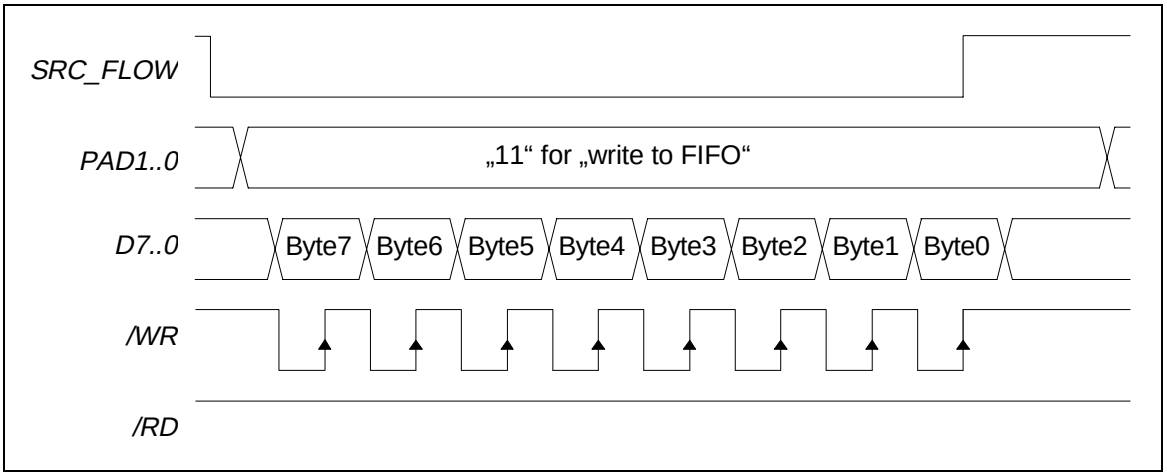


Figure 8-8: Source Port Parallel-Asynchronous Mode Timing Overview

The *PAD1..0* specify the kind of operation to be performed, and */RD* and */WR* define a read or a write access, respectively. Before writing to FIFO, either the *SRC_FLOW* signal must be low, or the SRCF bit in bSP must be 0.

When *SRC_FLOW* is low, the *PAD1..0* signals select the operation, data is then setup and finally a rising edge of /WR latches the data in the chip.

After eight write cycles, *SRC_FLOW* changes from low to high, indicating that the FIFO is busy being processed. *SRC_FLOW* will stay high until the chip has finished processing the FIFO, at which time SRC_FLOW will go low and more FIFO transfers can occur. As shown in Figure 8-8, the data byte written to the FIFO last will be stored in "Byte 0" of FIFO. For more information about the internal structure of FIFO, refer to Figure 8-1 on page 59. Figure 8-9 illustrates how the bytes in FIFO are associated with the bytes in memory.

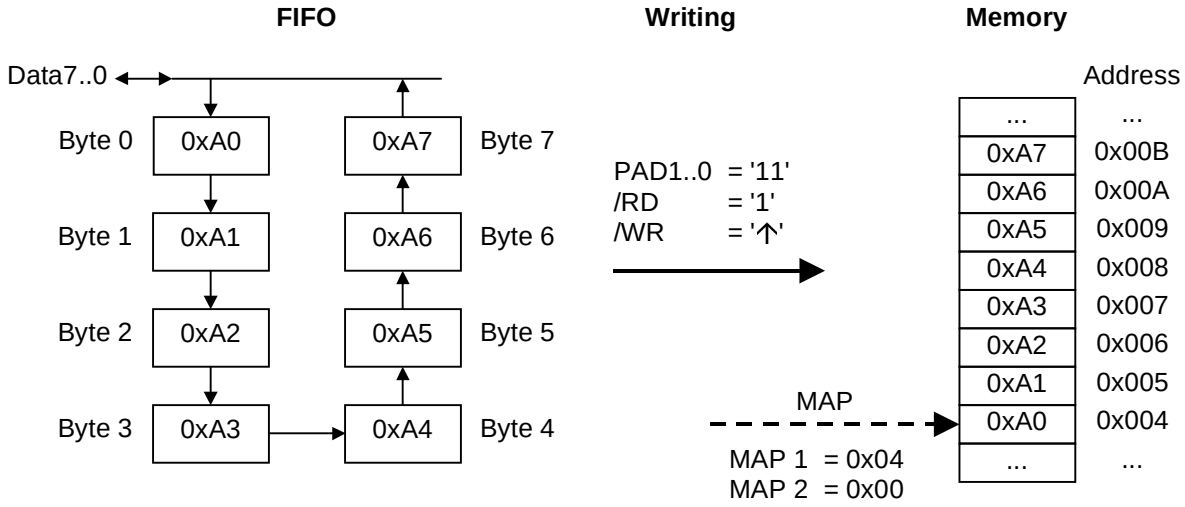


Figure 8-9: SP Parallel-Asynchronous Write Data Mapping Example

8.2.2.3 Writing MAP

For writing the MAP value into the chip, the following signals must be set:

PAD1	*PAD0*	*/RD*	*/WR*	Selected Operation
1	0	1	↑	Write MAP to FIFO

Table 8-12: Writing MAP in Parallel-Asynchronous Mode

A rising edge at */WR* stores the data into the FIFO. The order of bytes (xx = don't care) within the FIFO is:

FIFO	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
Contents	MAP1	MAP2	xx	xx	xx	xx	xx	xx

Table 8-13: Contents of FIFO for Writing MAP

To achieve this order, 6 arbitrary bytes must be written to the FIFO first. Then MAP2 and finally MAP1 must be written as shown below:

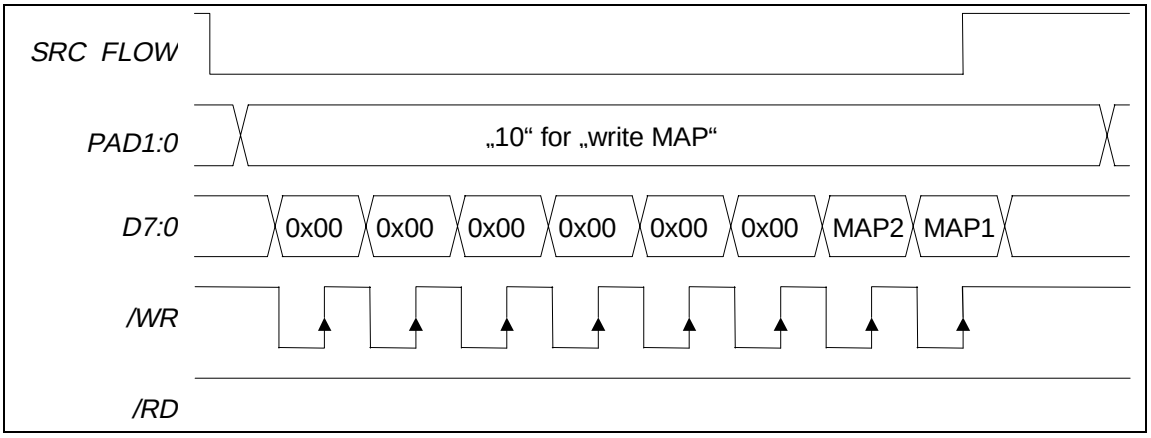


Figure 8-10: Writing MAP Timing

8.2.2.4 Writing 8 bytes to the FIFO

Before writing data to the FIFO the first time, the MAP data must be initialized. Then data can be continually transferred up to a page boundary. To cross a page boundary, the new page value must be written into the MAP. After writing MAP, the application has to wait for *SRC_FLOW* changing back to low level. Then the data can be written to the FIFO. For this operation, the following signals must be set:

PAD1	PAD0	/RD	/WR	Selected Operation
1	1	1	↑	Write DATA to FIFO

Table 8-14: FIFO Signals for Writing in Parallel-Asynchronous mode

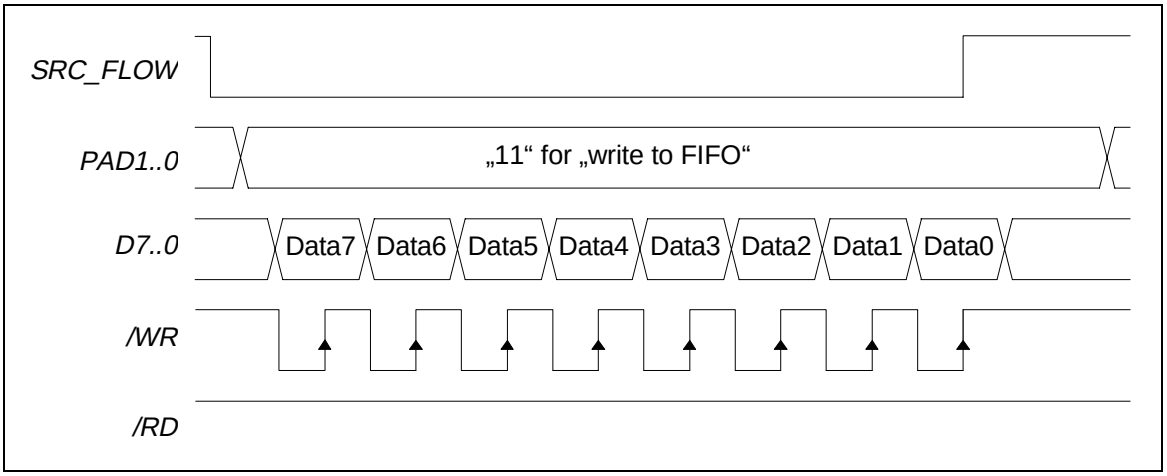


Figure 8-11: Writing 8 bytes Timing (SP In Parallel-Async. Mode)

The sequence of data in the FIFO needed for this operation is described in Section 8.2.2.2 on page 68. MAP is auto-incremented, so that it needs to be written only once when writing larger blocks of data. After the next falling edge at *SRC_FLOW*, the next set of eight bytes can be written (if desired).

8.2.2.5 Writing 1 Byte to the FIFO

To write only one byte to memory, the data byte is sent along with the MAP value. The MSB of MAP2 must be set to indicate a single byte transfer. Therefore, the FIFO write sequence consists of writing five arbitrary bytes followed by the data byte, followed by the two MAP bytes, as shown in the figure below:

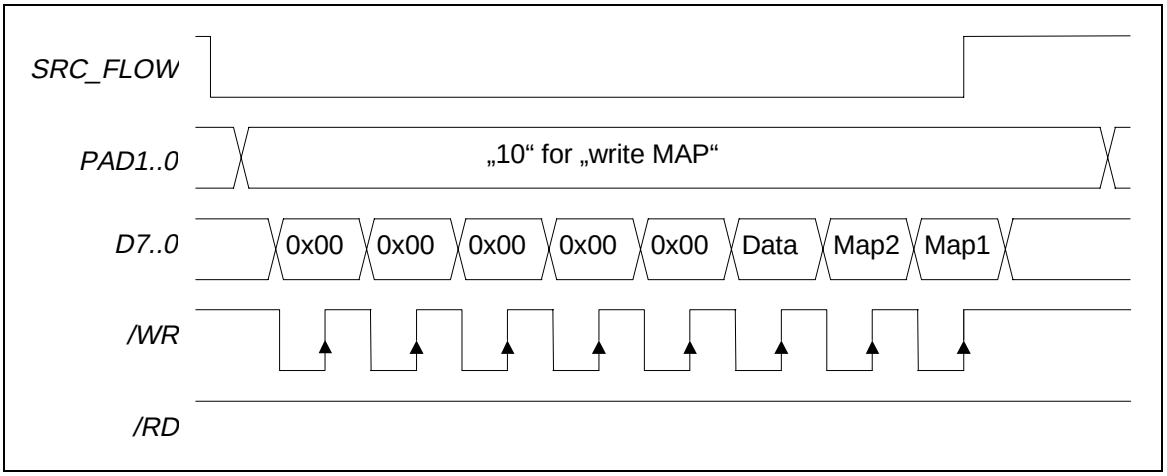


Figure 8-12: Writing 1 byte of data (SP In Parallel-Async. Mode)

8.2.2.6 Reading 8 Bytes from the FIFO

As with the write sequence, the MAP value must be initialized before any data can be read from the FIFO. After writing MAP, the application has to wait for *SRC_FLOW* to change back to a low level before data can be read from FIFO. For this operation, the following signals must be set:

PAD1	*PAD0*	*/RD*	*/WR*	**Selected Operation**
1	1	↓	1	Read Data from FIFO

Table 8-15: Reading 8 Bytes Of Data (SP in Parallel-Async. Mode)

Eight bytes must always be read from the FIFO. The following figure illustrates the signal flow for reading data in Parallel-Asynchronous data transfer mode:

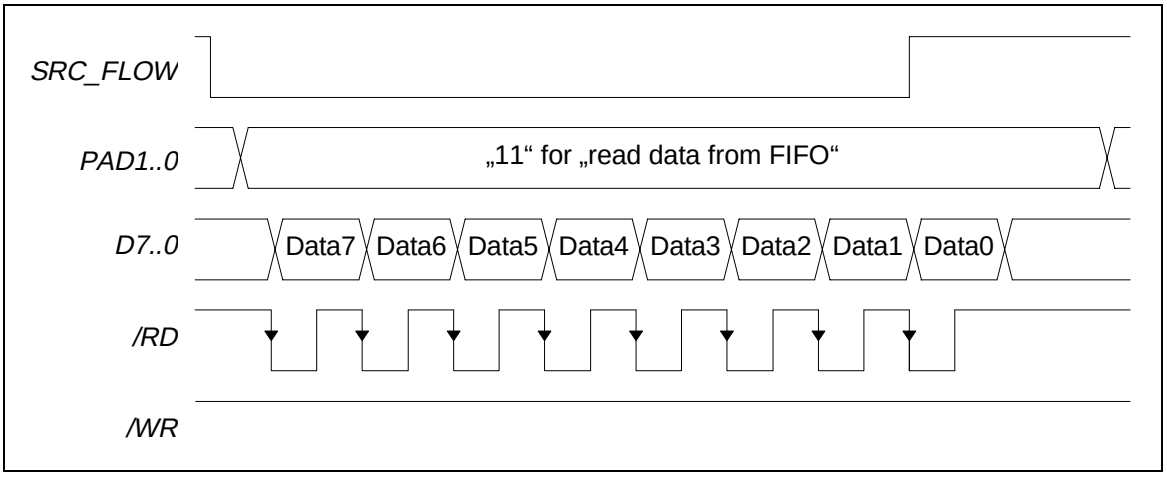


Figure 8-13 Reading 8 bytes Timing (SP in Parallel-Async. mode)

After *SRC_FLOW* has changed to low level, *PAD1..0* must be set to '11'. Then a falling edge at */RD* makes the data accessible at D7..0. After eight read cycles, *SRC_FLOW* changes from low to high again, indicating that the FIFO is busy being processed. MAP is auto-incremented, so it needs to be written once when reading larger blocks of data. After the next falling edge of *SRC_FLOW*, the next set of eight bytes can be read (if desired). The last byte of the 8-byte FIFO is read out first. The first byte in FIFO contains the first byte read from memory (The data MAP pointed at).

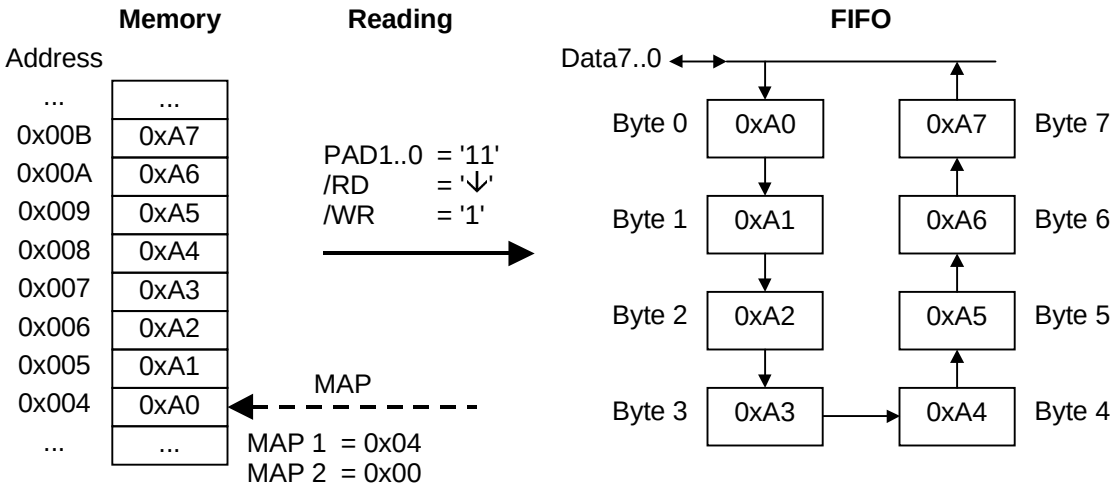


Figure 8-14: SP Parallel-Asynchronous Read Data Mapping Example

8.2.3 Reading the Source Port Status Register (bSP)

For getting information about the current status of the Source Data Port in parallel mode, the status register bSP can be read. bSP is a special register that is only available through the parallel interface and does not reside in the OS8104 internal memory map. For reading bSP, the following signals must be set:

PAD1	*PAD0*	*/RD*	*/WR*	Selected Operation
1	0	↓	1	Read SP Status Register bSP

Table 8-16: Reading the Source Port Status Register (bSP)

A single read cycle gets the contents of bSP, as opposed to the eight read cycles required to transfer data.

	bSP	Source Port Status (in parallel mode, read-only)	
Bit	**Name**	**Description**	**Default**
7	ZP	Zero Power	0
6	LP	Low Power	0
5	rsvd	Reserved.	0
4	AINT	Asynchronous Packet interrupt (opposite of /AINT pin)	0
3	rsvd	Reserved.	0
2	FIFO Empty	FIFO Empty	0
1	FSYNC	Frame Sync (FSY pin)	0
0	SRCF	Source Port Busy (SRC_FLOW pin)	0

Table 8-17: bSP (Source Port Status – parallel mode)

ZP (Zero Power Mode)
This bit indicates, that the chip is in zero power mode.

LP (Low Power Mode)
This bit indicates, that the chip is in low power mode.

AINT (Asynchronous Data interrupt)
This bit shows the status of the AINT pin. The value in this register is inverted from the value on the /AINT pin. If an interrupt is active, this bit will be 1 and the /AINT pin will be low.

FIFO Empty
This bit indicates, that the FIFO is ready to receive the first byte. This bit can be used to re-synchronize the application to FIFO in Parallel-Asynchronous mode.

FSYNC (frame sync)
This bit indicates the status of the *FSY* pin. FSYNC changing from high to low indicates the start of the SF0 interval.

SRCF SRC_FLOW bit.
This bit indicates the status of the *SRC_FLOW* pin. In Parallel-Synchronous (or Parallel-Combined) this bit (changing from low to high) along with FSYNC indicate the particular SF interval. In Parallel-Asynchronous mode, SRCF set to 1 indicates the FIFO is busy being processed and cannot transfer data. SRCF set to 0 in Parallel-Async. mode indicates the FIFO is ready for more transfers.

When the chip is either in LP or ZP mode, two writes to any register (in parallel mode) will wake up the chip.

8.2.4 Source Port Control Signal Overview

PAD1	PAD0	/RD	/WR	Selected Operation
1	0	↓	1	Read SP Status Register bSP
1	0	1	↑	Write MAP to FIFO
1	1	↓	1	Read Data from FIFO
1	1	1	↑	Write Data to FIFO

Table 8-18: SP Control Signal Overview

8.3 Parallel-Combined Mode

The Parallel-Combined mode is a high-speed parallel interface mode designed to support the maximum bandwidth for both synchronous source data and asynchronous source data (packet data). All 60 source data bytes (divided between synchronous and asynchronous data) of the MOST frame are accessible in this mode. In addition, status and control information are provided to manage the asynchronous (packet) data transfers externally. Since the data throughput is very high, the device controlling the OS8104 must be able to handle large amounts of high-speed data. The Control frame data can also be managed through the parallel interface when in Parallel-Combined mode.

For Parallel-Combined mode, the OS8104 must be initially configured for the Parallel-Synchronous (stream) mode. Then setting the APCM bit to 1 in bPCMA changes the OS8104 parallel interface from Parallel-Synchronous to Parallel-Combined mode. All source data transfers are handled via the FIFO. The synchronous source data routing is handled as in the Parallel-Synchronous mode. The FIFO is processed eight times per frame in each direction for source data. The frame is divided up into eight intervals of identical length, labeled SF0 to SF7. During each interval at least one read or write access must be performed. In addition, the transmit status quadlet must be fully written (all four bytes) during the proper SF interval.

Figure 8-15 illustrates the eight SF intervals and reading and writing the parallel port. As in the Parallel-Synchronous mode, accesses to the parallel port must not be made at the end of the SF interval for t_{rws} amount of time. During an SF interval the FIFO can be read from, then written to. Also illustrated in the Figure is control data accesses after all the source data is transferred (assuming the Control Port is configured to use the parallel interface).

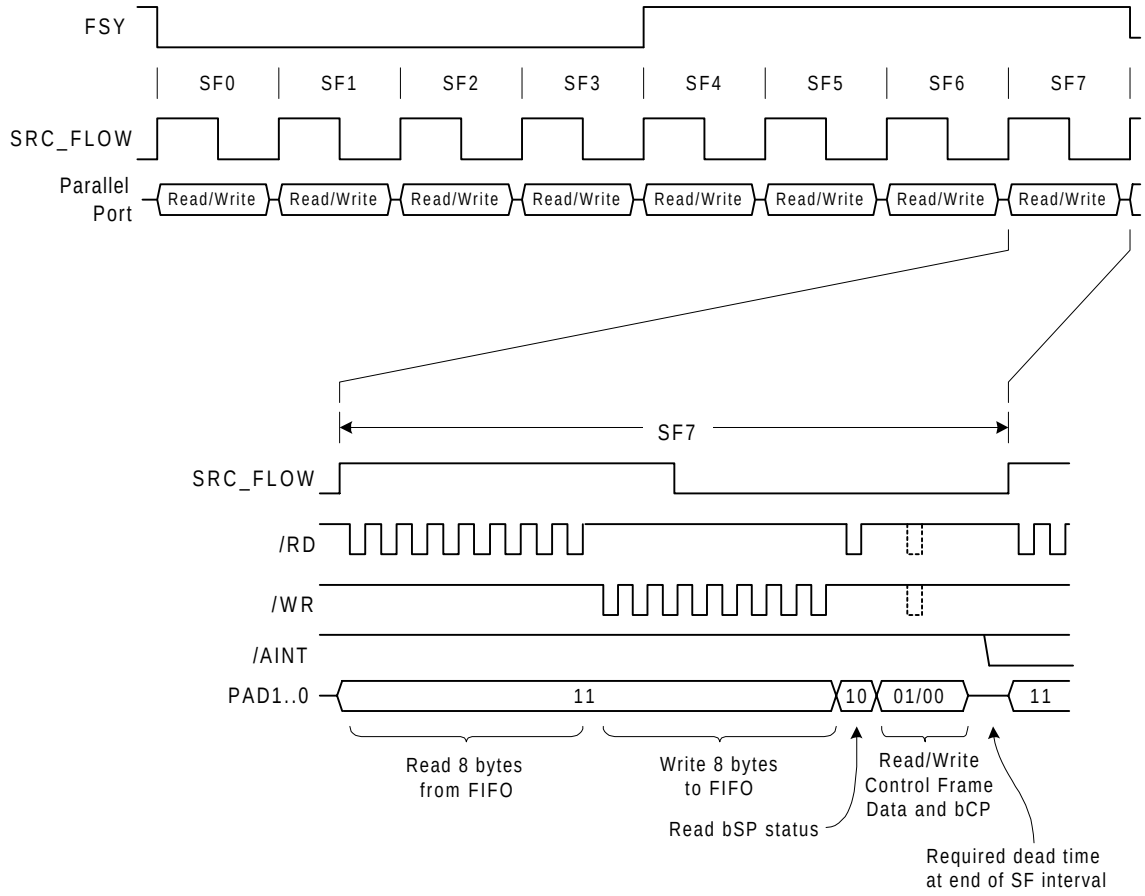


Figure 8-15: Parallel-Combined Mode Timing

Although asynchronous packets of up to 1024 bytes can be transferred in this mode, the packet length must not exceed 48 bytes when sending to nodes that are not configured in Parallel-Combined mode. For asynchronous data handling in the other modes (serial and Parallel-Asynchronous), the OS8104 buffers the packet on chip, which limits the packet size to the internal buffer size of 48 bytes. In Parallel-Combined mode, the asynchronous data is transferred through the external port and the packets must be assembled by the external device. The OS8104 only acts as a transport mechanism and all packet management and control must be handled by the external device. However, the OS8104 does add the CRC data to the end of a transmit packet, and checks the CRC on received packets.

8.3.1 Configuring Parallel-Combined Mode

Since the Parallel-Combined mode is based on the Source Port's Parallel-Synchronous mode, the OS8104 must be configured in this mode first (See Section 8.2 on page 65). Then, the Routing Engine (RE) registers 0x40 up to 0x7F must be filled with address references 0x00 up to 0x3F. This will map all 60 bytes of source data, received from the network, to the parallel port (along with 4 additional status bytes). The upper half of the RE registers will look as follows:

RE-Reg.	+0	+1	+2	+3	+4	+5	+6	+7	SF:
0x40	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	← 0
0x48	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F	← 1
0x50	0x10	0x11	0x12	0x13	0x14	0x15	0x16	0x17	← 2
0x58	0x18	0x19	0x1A	0x1B	0x1C	0x1D	0x1E	0x1F	← 3
0x60	0x20	0x21	0x22	0x23	0x24	0x25	0x26	0x27	← 4
0x68	0x28	0x29	0x2A	0x2B	0x2C	0x2D	0x2E	0x2F	← 5
0x70	0x30	0x31	0x32	0x33	0x34	0x35	0x36	0x37	← 6
0x78	0x38	0x39	0x3A	0x3B	0x3C	0x3D	0x3E	0x3F	← 7
FIFO Byte:	↑ Byte0	↑ Byte1	↑ Byte2	↑ Byte3	↑ Byte4	↑ Byte5	↑ Byte6	↑ Byte7	

Table 8-19: Upper Half of RE Registers in Parallel-Combined mode

The table above shows the location where each byte of the incoming frame is found in the FIFO, for the respective SF interval.

For sending data out to the network, the lower half of the RE (locations 0x00 up to 0x3F) must be configured. By default (after reset), the lower half of the RE registers is configured for "direct throughput", where the bytes received from RX are retransmitted out of TX, in the same channel they came in. For routing synchronous source data, entered through the parallel port, onto the network, the lower half of the RE registers must be modified as described in Section 12.4 on page 104, replacing the "direct throughput" data with the externally-sourced data address reference.

8.3.1.1 bPCMA (Parallel-Combined Mode Activate register)

0xE6	bPCMA	Parallel-Combined Mode Activate Register	
Bit	**Name**	**Description**	**Default**
7..1	rsvd	Reserved; Write as 0	0000000
0	APCM	Activate Parallel-Combined mode	0

Table 8-20: bPCMA (Parallel-Combined Mode Activate Register)

APCM (Activate Parallel-Combined Mode)

If SP is in Parallel-Synchronous mode, writing a 1 to APCM, enables Parallel-Combined mode.

8.3.2 Receiving Source Data

Based on the configuration of the RE registers shown in Table 8-19, all the network-received source data (60 bytes) is sent out the parallel port, via the FIFO, followed by 4 status bytes. The division between the synchronous and asynchronous data is contained in bSBC, and is also part of the status bytes at the end of the frame. The external device would generally collect the entire frame of data into a buffer, and then use the SBC data in the status to determine the start of asynchronous data. The external device does not have to read synchronous source data that it doesn't need; however, at least one read or write access must occur during each SF interval, and the full TX status quadlet must be written. Figure 8-16 illustrates reading the source data read from the OS8104's FIFO.

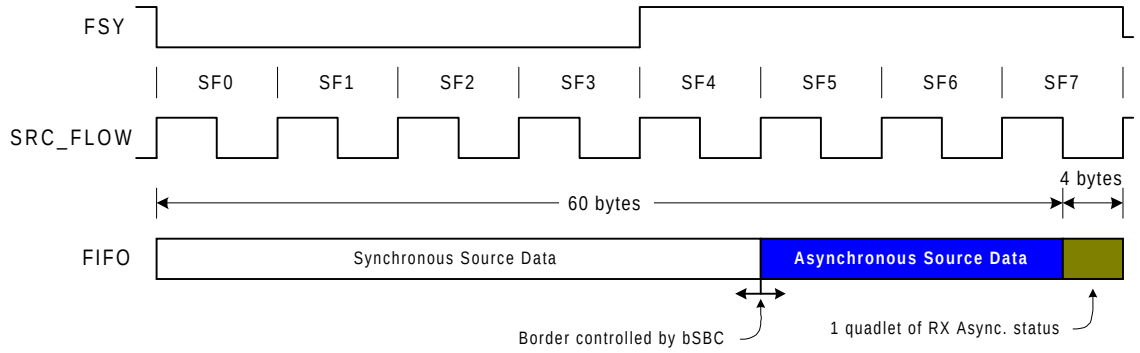


Figure 8-16: Parallel Port Data Output (Network Received Data)

8.3.3 Asynchronous Data Packets

In Parallel-Combined mode, an asynchronous data packet has the following structure:

Byte	Function
0	Arbitration
1	Destination address (high byte)
2	Destination address (low byte)
3	Length of data (in quadlets) valid values 0x01 up to 0xFE
4	Source address (high byte)
5	Source address (low byte)
6	Data byte 0
7	Data byte 1
...	...
N	Data byteV
N+1 .. N+4	CRC

Table 8-21: Data Packet Architecture

The bytes shaded (4-N) are relevant for the length calculation. For transmit packets, the CRC is added by the OS8104 hardware. For receive packets, the CRC is validated by the OS8104, with the result available in the last status byte (source data byte 63).

8.3.3.1 Length

The length value is in quadlets (4 bytes) and is calculated as described in Section 15.1.2 on page 130. The maximum length is 0xFE, as opposed to 0xD for Packet Data transfer when not in Parallel-Combined. The number of data bytes transmitted in a packet is:

number of data bytes = (Length × 4) - 2

Two bytes must be subtracted since two bytes of the first quadlet, used in the length calculation, are the source address. The maximum number of bytes per packet is:

(0xFE × 4) - 2 = (254 × 4) - 2 = 1014

8.3.3.2 Received Async. Status Bytes

Starting with the first frame source data byte at 0, the first status byte is 60. The table below shows the status bytes and their function.

Number	Name	Inactive	Function
60	Synchronous Bandwidth Control (SBC)	---	Contains the information about the number of bytes used for synchronous source data transfer (in quadlets). See Section 6.2.5 on page 37.
61	Start	0x00	If non-zero, this byte indicates that this frame contains the start of a packet.
62	rsvd	rsvd	Reserved
63	Error	0x00	If non-zero, this byte indicates that an error (incorrect CRC or packet ended prematurely) occurred during reception of the current packet.

Table 8-22: Asynchronous RX Status Bytes

When receiving a packet, the position of the destination address and packet length (contained in the packet) can be derived from the current bSBC value:

Destination Address, high byte location = (SBC × 4) + 1
Destination Address, low byte location = (SBC × 4) + 2
Packet length, in quadlets = (SBC × 4) + 3

The Destination Address and Packet length are only available in the first buffer of a packet (when the "Start" status flag is non-zero).

8.3.3.2.1 Moving Received Async. Status Bytes

The asynchronous received status bytes have been previously illustrated in a fixed position at the end of the MOST network frame. These bytes may also be placed between the asynchronous and the synchronous source data; however, their position would not be fixed and would now be determined by the bSBC value. Assuming the application can manage the dynamic nature of this location, having the status quadlet before the asynchronous data bytes helps minimize the external buffering needed.

The upper half of the RE registers would need to be modified whenever the bSBC value changed. The special address references 0x3C through 0x3F correspond to the four received status bytes. To move the status bytes to the synchronous/asynchronous boundary, the following Routing Engine registers must be changed:

RE[0x40+(bSBC × 4)] = 0x3C
RE[0x41+(bSBC × 4)] = 0x3D
RE[0x42+(bSBC × 4)] = 0x3E
RE[0x43+(bSBC × 4)] = 0x3F

Then the succeeding RE registers must be filled with the network received asynchronous source data address references as follows:

RE[0x44+(bSBC × 4) + n] = bSBC × 4 + n

where n starts at 0 and ends when the rest of the RE registers are filled (RE0x7F is the last register). Placing the status before the asynchronous data changes the position of the destination address and packet length to:

Destination Address, high byte location = (bSBC × 4) + 5
Destination Address, low byte location = (bSBC × 4) + 6
Packet length, in quadlets = (bSBC × 4) + 7

The following example uses a bSBC value of 0x0A, or 10 quadlets reserved for synchronous source data and 5 quadlets reserved for asynchronous packet data. To place the status between the synchronous and asynchronous data, the first status byte is placed in RE register 0x68 as illustrated below:

RE-Reg.	+0	+1	+2	+3	+4	+5	+6	+7	SF:
0x40	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	← 0
0x48	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F	← 1
0x50	0x10	0x11	0x12	0x13	0x14	0x15	0x16	0x17	← 2
0x58	0x18	0x19	0x1A	0x1B	0x1C	0x1D	0x1E	0x1F	← 3
0x60	0x20	0x21	0x22	0x23	0x24	0x25	0x26	0x27	← 4
0x68	**0x3C**	**0x3D**	**0x3E**	**0x3F**	0x28	0x29	0x2A	0x2B	← 5
0x70	0x2C	0x2D	0x2E	0x2F	0x30	0x31	0x32	0x33	← 6
0x78	0x34	0x35	0x36	0x37	0x38	0x39	0x3A	0x3B	← 7
FIFO Byte:	↑ Byte0	↑ Byte1	↑ Byte2	↑ Byte3	↑ Byte4	↑ Byte5	↑ Byte6	↑ Byte7	

Table 8-23: Upper Half of RE Registers: Moving Received Status Bytes Example

The asynchronous received source data address references (0x28 - 0x3B) are shifted up in the Routing Engine registers to make room for the status bytes. Therefore, the destination address and packet length (in the buffer where the 'Start" status is set) are now moved to (the first async. byte is arbitration):

Destination Address, high byte location = (bSBC × 4) + 5 = 0x2D
Destination Address, low byte location = (bSBC × 4) + 6 = 0x2E
Packet length, in quadlets = (bSBC × 4) + 7 = 0x2F

This assumes that buffer address 0 contains the first byte of synchronous source data. Figure 8-17 illustrates the received data flow with the status bytes moved.

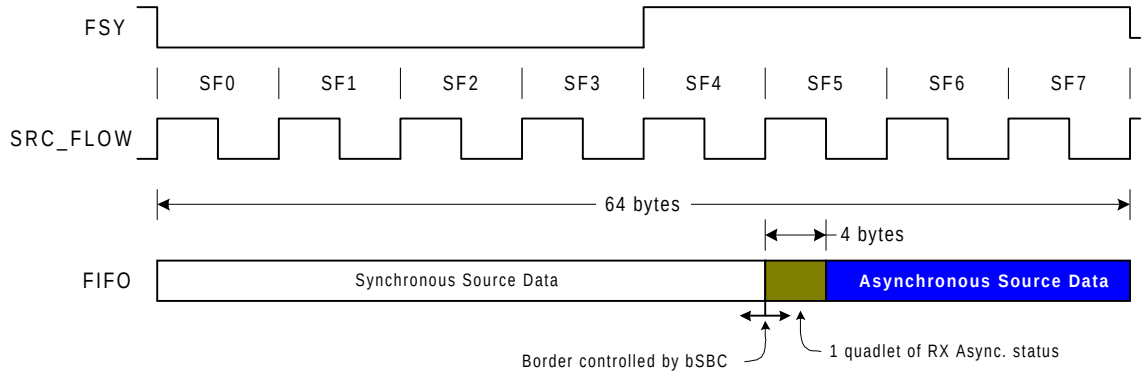


Figure 8-17: Parallel Port Data Output (Network Received Data) – Moved Status

8.3.3.3 Handling Received Data

Figure 8-18 on page 80 illustrates a program-flow example for handling received data in Parallel-Combined mode. In this example, all 60 data bytes of a frame, plus the four status bytes are read. In the OS8104, the asynchronous packet status bytes are not available until after the synchronous source data; therefore, the four status bytes must be the last four bytes of the frame, or placed between the synchronous and asynchronous source data. The routine described in this example uses two additional variables:

- Packet reception:
  Flags whether the routine expects the current frame to contain an intermediate part of the packet (Packet reception set)
- Length:
  Contains the number of bytes to be received.

After receiving a complete frame, the status byte containing SBC should be read first. Then the division of source data and packet data can be calculated. After the synchronous source data is transferred to the appropriate sections of the application, packet data handling is performed.

The start of a packet reception is flagged by status byte "Start", which is set to a nonzero value. When a start of packet is received, the destination address must be compared with the local node address. If the address matches, the length of the packet must be calculated and data must be copied. If the number of bytes to be received is greater than the data contained by the current frame, the local variable "Packet reception" provides reading of packet data in succeeding frames. "Packet reception" should be reset after the last byte is read. This flow doesn't handle errors and, as shown in Figure 8-18, some additional checks are needed to detect possible problems.

If the Start status byte is non-zero, this frame contains the beginning of a Packet transmission. An error exists if "Packet reception" is active at the same time Start is received, since the receiving of the preceding packet was not finished correctly. In this case an error must be reported, "Packet reception" must be cleared and packet data handling is finished.

If Start is not zero and "Packet reception" is inactive, the destination address must be read and compared with the local node address. This is the task of the application, since the chip does not perform address comparison in Parallel-Combined mode. If the address does not match, packet handling is finished.

If the address matches, "Packet reception" must be set, and the length of the packet, in bytes, must be calculated, based on the value contained in the packet (see 15.1.2 on page 130). This value is stored in variable "Length". Then the read pointer must be adjusted to the start of packet data. The received status also includes an Error status byte, which should be checked. If the error byte is zero, data can be read until the last byte of the packet is read, or the end of the buffer is reached. Whenever the last byte was read, "Packet reception" must be cleared and Packet handling is finished.

If the error status bit is set, the error should be reported, "Packet reception" should be cleared, and Packet handling is finished.

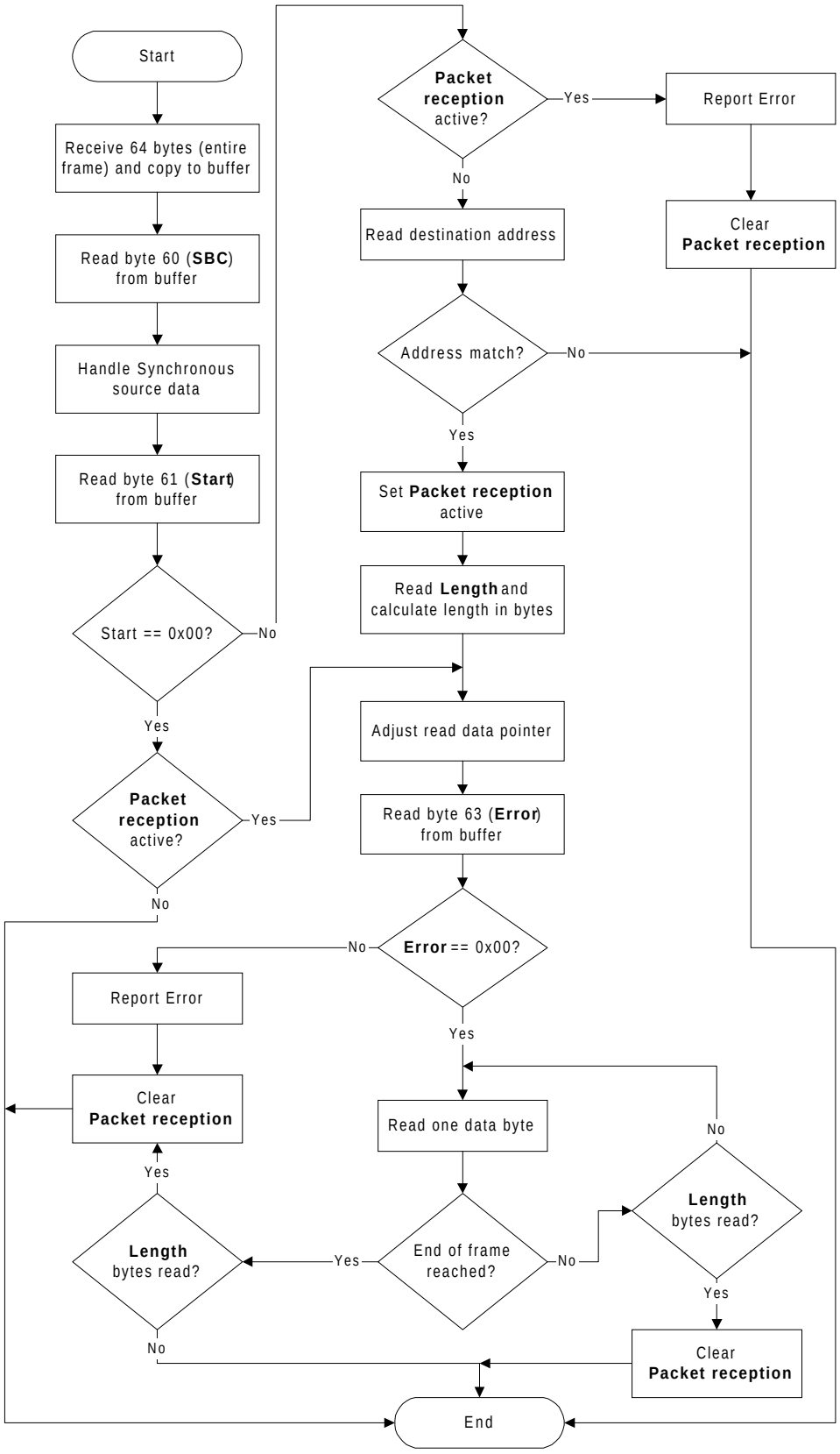


Figure 8-18: Program Flow for Receiving Packet Data (Parallel-Combined mode)

8.3.4 Transmitting Data

For transmitting synchronous source data, the same rules used for the Source Data Port (SP) in Parallel-Synchronous data mode apply. The address references for routing synchronous source data are identical, and are described in Section 12 on page 97.

When sending out synchronous source data and packet data in Parallel-Combined mode, the synchronous source data must be transmitted first, followed by the status bytes (1 quadlet), and then the packet data.

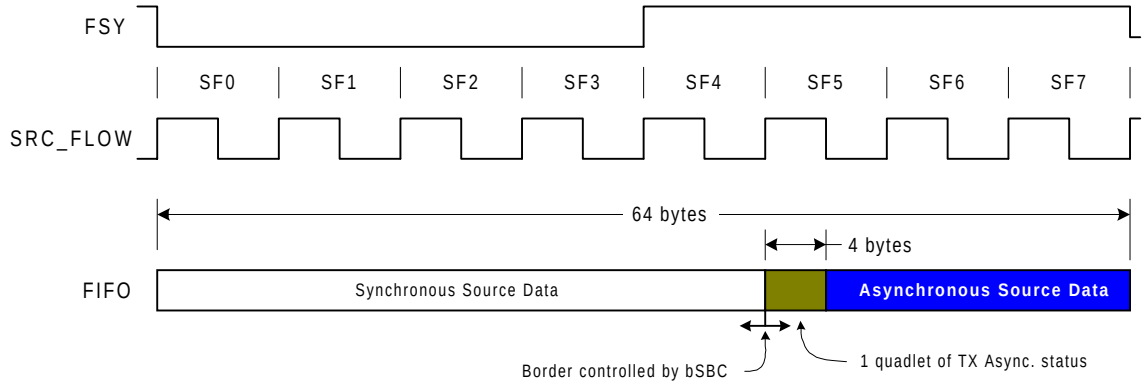


Figure 8-19: Parallel Port Data Input (Network Transmit Data)

8.3.4.1 Transmit Asynchronous Status Bytes

When transmitting packet data, one quadlet contains status information for the Routing Engine and must be prepared by the external application.

Number	Name	Inactive	Function
0x00	Start Transmit	0x00	Indicates to the chip that the external application wants to start to transmit a packet of data.
0x01	rsvd	rsvd	Reserved, write as 0x00
0x02	ACK	0x00	Indicates to the chip that the application has recognized the activation of the */AINT* pin and is sending the fourth frame of data.
0x03	rsvd	rsvd	Reserved, write as 0x00

Table 8-24: Asynchronous TX Status Bytes

All four TX status bytes must be transmitted each frame, even if no packet data is being sent.

8.3.4.2 Preparing Packet Data

The table below illustrates a transmit Packet. The CRC is calculated automatically by the chip and added to the end of the packet.

Byte	Function
0	Arbitration
1	Destination address (high byte)
2	Destination address (low byte)
3	Length of data (in quadlets) valid values 0x01 up to 0xFE
4	Source address (high byte)
5	Source address (low byte)
6	Data byte 0
7	Data byte 1
...	...
N	Data byte V

Table 8-25: TX Packet in Parallel-Combined mode

The arbitration value must be calculated by using the following formula (The Node Position Address is in register bNPR):

Arbitration = (Node Position Address × 2) +1

The destination address is the logical node address or the alternate node address of the node that the packet is being sent to. The length is a one byte value that contains the length of the data packet, bytes 4 through N, in quadlets. The length calculation includes the two source address bytes and must be rounded up to whole quadlets.

length = roundup ((number of data bytes + 2) / 4) = roundup ((number of packet bytes 4-N)/4)

The maximum length value is 0xFE, as opposed to 0x0D for packets when not in Parallel-Combined mode. The Source address is the logical node address of the node transmitting the packet, and the high and low address bytes are included in the length calculation.

Packet length must not exceed 48 bytes when sending to nodes that are not configured in Parallel-Combined mode.

The external application must fill in the source address bytes with the node's logical address (bNAH/bNAL).

When the external application has a packet ready for transmission, it starts transmission by setting the "Start Transmit" status byte to a value other than zero. Then the rest of the frame contains the first buffer of the packet data, stored in one of the internal frame buffers. When the frame of data is received by the OS8104 with the "Start Transmit" status flag set, the chip starts to arbitrate for the MOST network asynchronous data channel.

If the packet fits into three frames or less, the packet data can be written, frame by frame and no further action is needed. When the chip wins arbitration, the */AINT* pin is driven low, indicating that arbitration is won. If more than three frames are needed for the packet, then the external device must send the fourth buffer of data in the frame after /AINT goes low, and continue sending portions, frame by frame, until the entire packet is sent. The fourth buffer must have "Ack" set to acknowledge that this buffer includes the fourth buffer of asynchronous data. When the final buffer of packet data is transmitted by the chip, */AINT* will be driven high, indicating to the external device that the OS8104 is ready to arbitrate/send a new packet, if needed.

In the example below, 2 quadlets per frame are reserved for asynchronous data transfer. Therefore, SBC has the value 0x0D, which reserves 52 bytes (13 quadlets) for synchronous source data transfer.

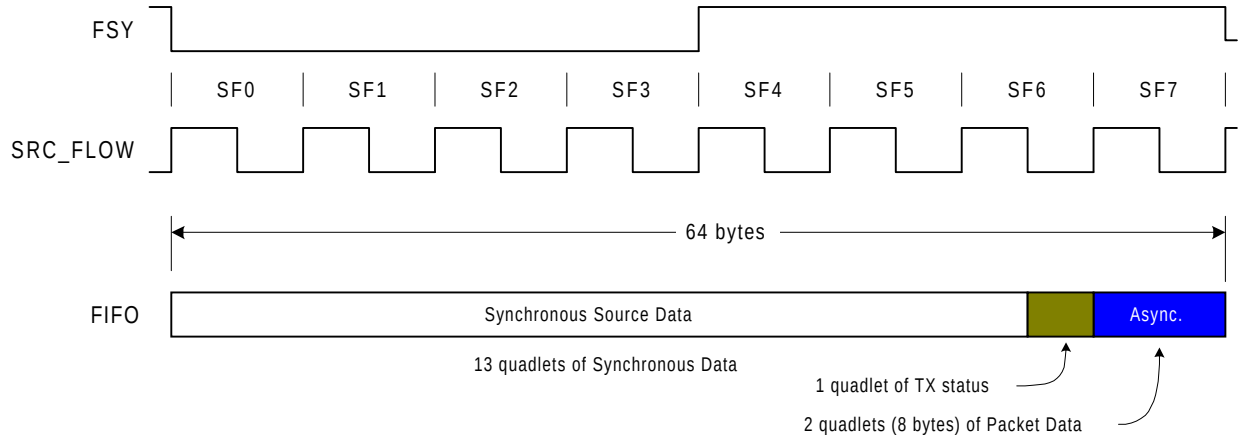


Figure 8-20: Packet Example - Source Data Allocation

In this example, the packet is comprised of nine data bytes sent, from logical node address 0x0222 (at node position 0x03), to destination address 0x0123. The assembled packet would look like:

Byte	Value	Function
0	0x07	Arbitration (two times node position plus one)
1	0x01	Destination address (high byte)
2	0x23	Destination address (low byte)
3	0x03	Length of data (in quadlets). Valid values are 0x01 through 0xFE
4	0x02	Source address (high byte)
5	0x22	Source address (low byte)
6	0x10	Data byte 0
7	0x11	Data byte 1
8	0x12	Data byte 2
9	0x13	Data byte 3
10	0x14	Data byte 4
11	0x15	Data byte 5
12	0x16	Data byte 6
13	0x17	Data byte 7
14	0x18	Data byte 8

Table 8-26: Sample Asynchronous Data Packet

The packet takes two frames to transmit since the total packet size is 14 bytes and only 8 bytes per frame are allocated to asynchronous data. SBC does not change dynamically, so the Start TX status byte will always be located at byte position 52 of the frame. The top portion of Figure 8-21 illustrates the data sent by the external device, through the parallel port, and into the FIFO. The bottom portion of the figure illustrates the FIFO being read by the Routing Engine and stored in an internal Frame buffer.

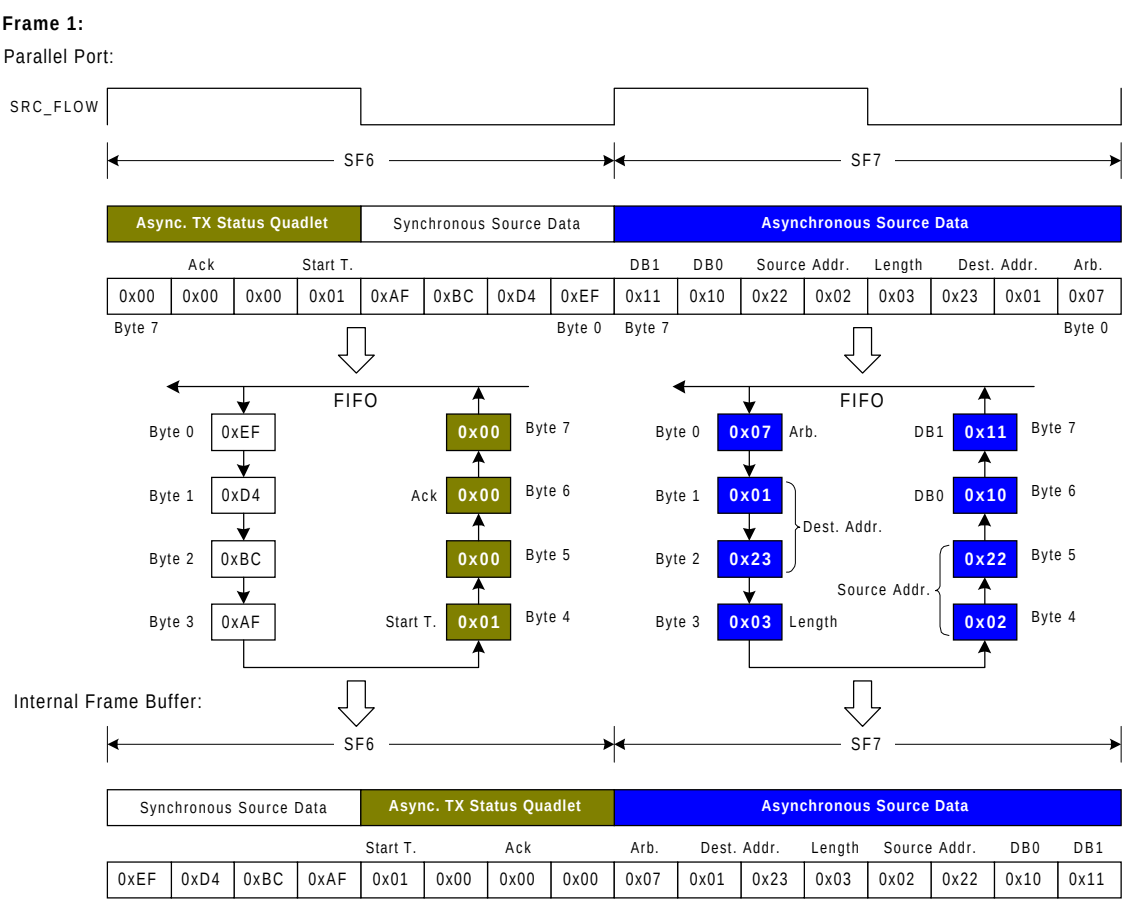


Figure 8-21: Async. Packet Example – Frame 1

The frame containing the first buffer (Frame 1), containing the initial portion of the asynchronous packet, must have the "Start Transmit" flag set (shown as "Start T." in the Figure). Frame 2 will contain the rest of the packet. The "Start Transmit" flag must be zero for all buffers/frames except the first one.

In SF6 of the first frame, the external device sends eight bytes to the FIFO. These bytes are half synchronous source data and half async. status bytes. The FIFO is filled in reverse order (Byte 7 to Byte 0), placing the Ack. status flag at the second byte sent to the FIFO and the "Start T." flag at the fourth byte sent to the FIFO. The Routing Engine reads the data out of the FIFO in reverse order, placing the "Start T." as the first async. status byte in the internal frame buffer.

SF7 contains the start of the asynchronous packet. The first byte of the packet is the arbitration byte, which is sent to the parallel port last (due to the FIFO byte-reversal). The last byte of the packet, in SF7 of this frame, is data byte 1 (DB1).

Frame 2 of this example is illustrated in Figure 8-22. This frame sends the second (and last) buffer for the example presented. Half of SF6 contains the Asynchronous TX Status bytes. The "Start T." status byte is cleared in this frame, since it is not the first buffer of the asynchronous packet. SF7 contains the last seven data bytes of the asynchronous packet. One extra byte is added to complete the eight-byte buffer since the length must be rounded up to full quadlets.

Frame 2:

Parallel Port:

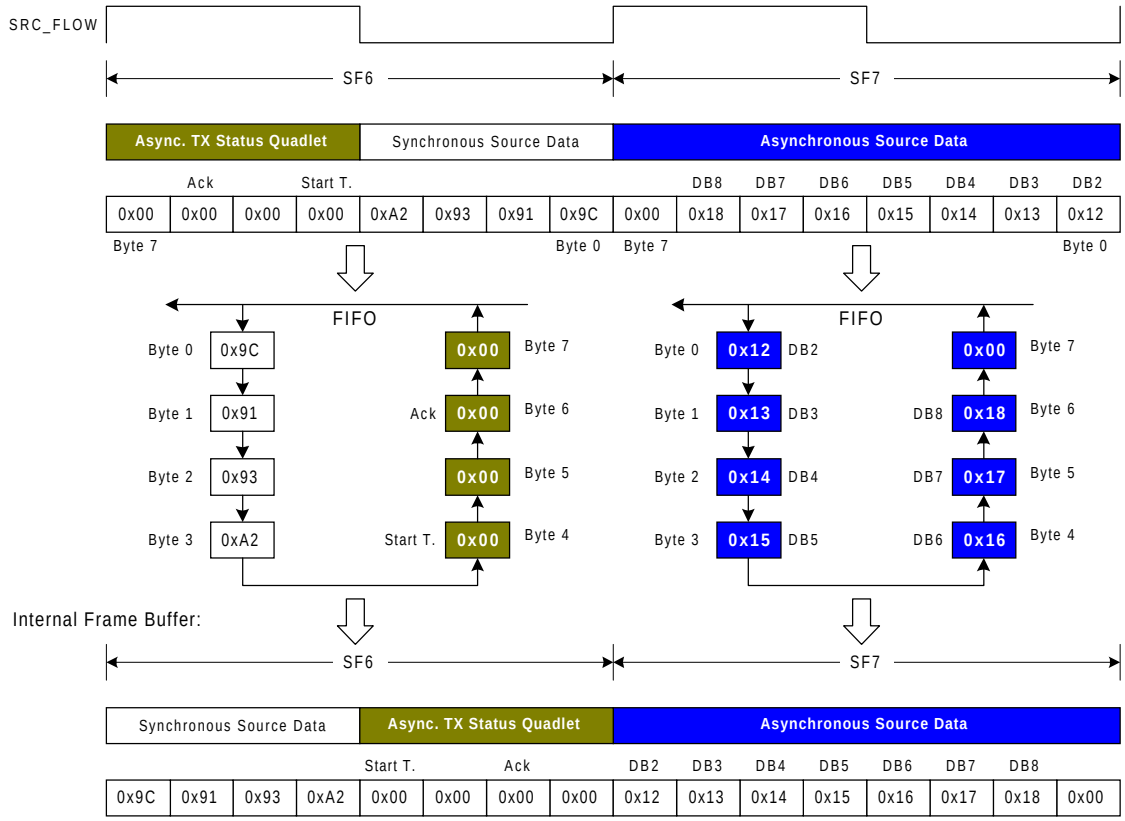


Figure 8-22: Async. Packet Example – Frame 2

The two internal frame buffers used in this example are illustrated in Figure 8-23. The frame buffers are 64 bytes long and store a combination of synchronous and asynchronous data, along with the four asynchronous TX status bytes. The asynchronous buffer size is 8 bytes (set by the bSBC register value). Since the OS8104 has three internal frame buffers, an asynchronous message that fits into three asynchronous buffers (24 bytes in this example) can be written in back-to-back frames. Asynchronous packets that take more than three buffers have to use the */AINT* pin and the "Ack" status byte for handshaking with the OS8104.

The synchronous source data stored in the internal frame buffers is only transmitted onto the MOST network if the lower half of the RE registers are configured as shown in Section 12.4.

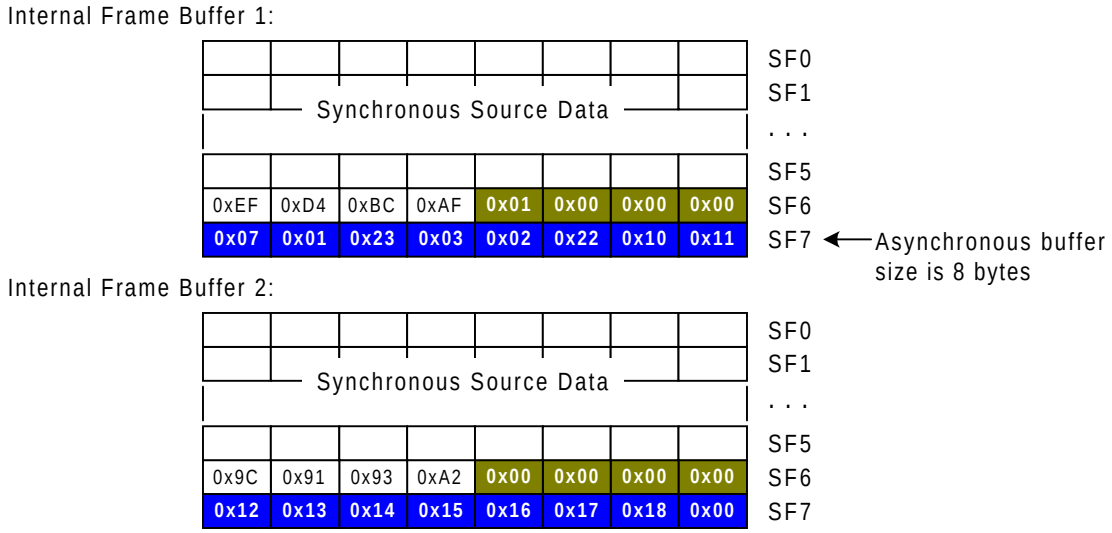


Figure 8-23: OS8104 Internal Frame Buffers

8.3.4.3 Multi-Frame Packets and /AINT Handshaking

To start transmitting a packet, the OS8104 has to arbitrate for the asynchronous data channel. There may be a delay between the start of transmission and when the node wins arbitration. Therefore a certain amount of data must be stored in internal data buffers. The OS8104 contains three frame buffers which can each hold one complete frame of source data. The Asynchronous buffer size is that part of the frame dedicated to asynchronous data (determined by bSBC).

If the packet fits into three frames or less, the packet data can be written, frame by frame and no further action is required. When the chip wins arbitration, the */AINT* pin is driven low. */AINT* will go high at the end of the message transmission, indicating a new message can now be sent. Using the example from the last section, the asynchronous message fits into two internal asynchronous buffers (Short Packet). When the first buffer is loaded into the chip (in Frame 1), with the "Start Transmit" (Start T.) status byte set, the OS8104 starts arbitrating for the asynchronous channel in the next frame (Frame 2). The second and final buffer is loaded in Frame 2, as illustrated in Figure 8-24 as "b2" If the asynchronous channel is free, the node will win arbitration in Frame 2 and drive /AINT low. However, if the asynchronous channel is busy, the node may not win arbitration until a later frame. In Figure 8-24, arbitration is won in the 4th frame where the chip drives /AINT low. The first asynchronous buffer b1, (loaded into the chip in Frame 1) is output on the network in the Frame 4 time period. Then the second (and last) asynchronous buffer for this message, b2, is output on the network in the Frame 5 time period. Since only two buffers were used for the asynchronous packet, /AINT is driven high at the end of Frame 5 indicating the message has been sent. In Frame 6, a new Packet can be started.

The time /AINT changes with respect to FSY is based on the bSBC value; however, /AINT will change at least a minimum of 5/(128Fs) before FSY changes, or 0.88 ms when Fs is 44.1 kHz.

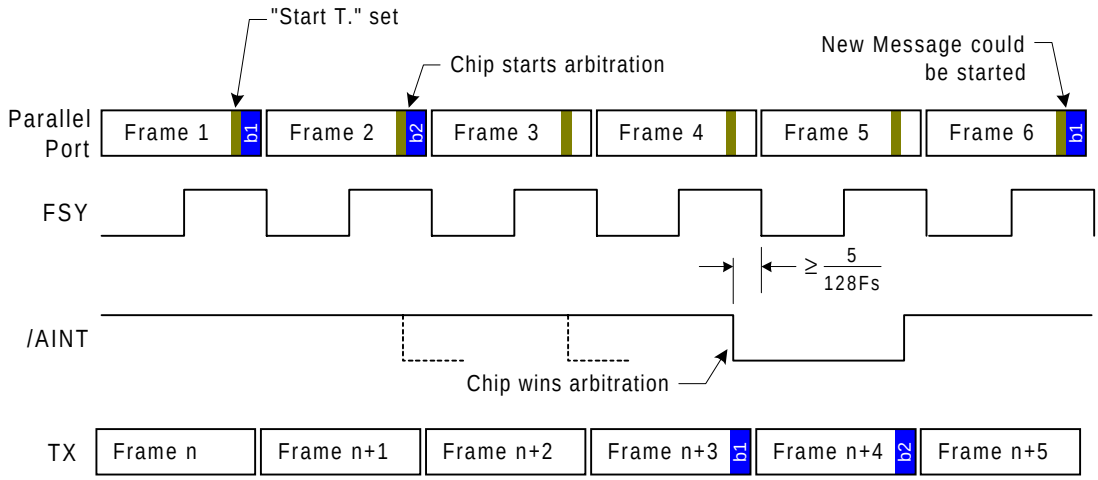


Figure 8-24: Asynchronous. Packet Example Arbitration

If the packet is larger than can be contained in the three internal buffers, then the external device should load the first three buffers in three consecutive frames. If /AINT is low by the beginning of the fourth frame, the external device can continually load buffers until the complete message is transferred. The "Ack" status byte in the Frame containing the fourth asynchronous buffer must be non-zero or the OS8104 will ignore the asynchronous data buffer. This scenario is illustrated in Figure 8-25, where the node wins arbitration while the second or third buffer is being sent to the parallel port. At Frame 4, /AINT is low so the external device continually sends asynchronous buffers.

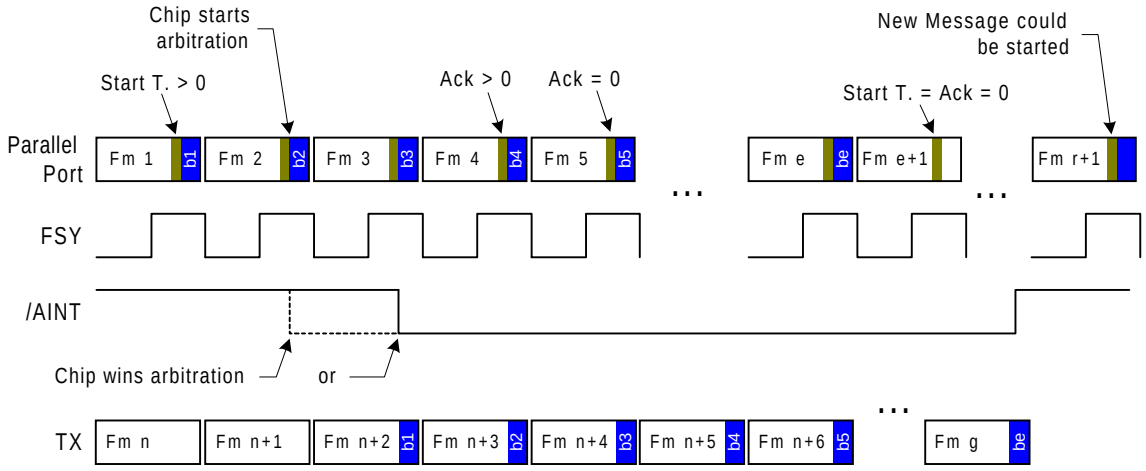


Figure 8-25: Packet Buffering with No Delay

/AINT goes high when the OS8104 has transmitted the last frame of the packet. Due to internal buffering, this packet is generally sent to the OS8104 three frames before it gets transmitted out the TX pin onto the network. The frames sent to the OS8104 after the last packet ("Fm e" in Figure 8-25), but before /AINT goes high, must not have the "Start Transmit" status flag set and must not have the Ack status flag set. (shown as Fm e+1 … in Figure 8-25).

If /AINT is still high by the beginning of the fourth frame, then the asynchronous channel on the MOST network is in use and the external device must wait until /AINT goes low before loading any more asynchronous data buffers. Synchronous source data should still be sent since the asynchronous source data operates independently of the synchronous source data. Typically, when an external device is waiting for /AINT to go low, it just continually repeats sending the third asynchronous data buffer since at least one access in each SF interval is required for normal operation. The actual data sent is irrelevant since it is not stored in an internal buffer.

Figure 8-26 illustrates sending a Packet larger than three buffers, where the network asynchronous channel is in use. The network asynchronous channel frees up at parallel port Frame 6 time, where the local node wins arbitration. The external device notices /AINT low and, starting in the 7th frame, sends the fourth asynchronous buffer (b4) with the Ack status byte greater than 0. The fifth asynchronous buffer (b5) is sent in Frame 8 with the Ack status byte reset to 0, and buffers are continually transmitted until the entire packet is sent. A new message can be sent in the frame following the one where /AINT goes high.

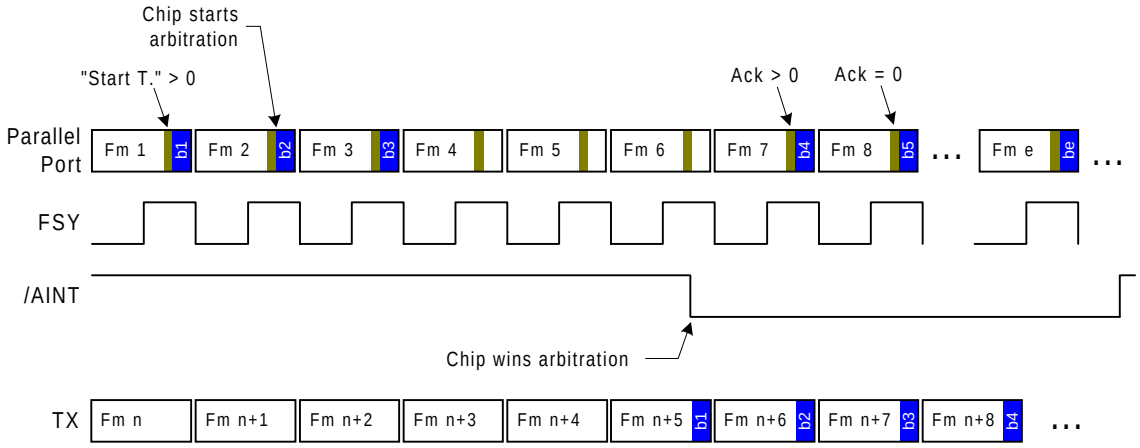


Figure 8-26: Packet Buffering with Delay

The fourth packet buffer (with the Ack status byte greater than 0) must be sent in the frame following the one where /AINT went low. Otherwise, invalid data may be sent out.

8.3.4.4 Priority

Packet priority is specified in register bPPI (Section 15.1.3 on page 130).

8.3.4.5 Idle SF Intervals

If no data transfer is required during an SF interval, it is sufficient to perform a single read or write access. However, all four bytes of the asynchronous transmit status quadlet must always be written (to all zeros if no status).

9 Clock Manager

The clock manager generates all internal clocks and an additional clock signal called *RMCK*, which can drive external devices such as CD and DVD drives, and other digital devices. The core of the clock manager is a PLL, which can be synchronized to one of four possible clock sources:

- Crystal oscillator (or external clock source)
- Bit clock (*SCK*)
- S/PDIF data stream (through *SR0*)
- Incoming data from MOST network (*RX*)

The clock manager is controlled by two registers: bCM1 and bCM2. Figure 9-1 shows a block diagram of the clock manager. The PLL always provides a clock source to the devices on the OS8104; however, when the PLL is unlocked (i.e. no data on RX pin), the PLL will drift to its lowest frequency. The OS8104 is always accessible through the Control Port, although source data is not transferred when unlocked.

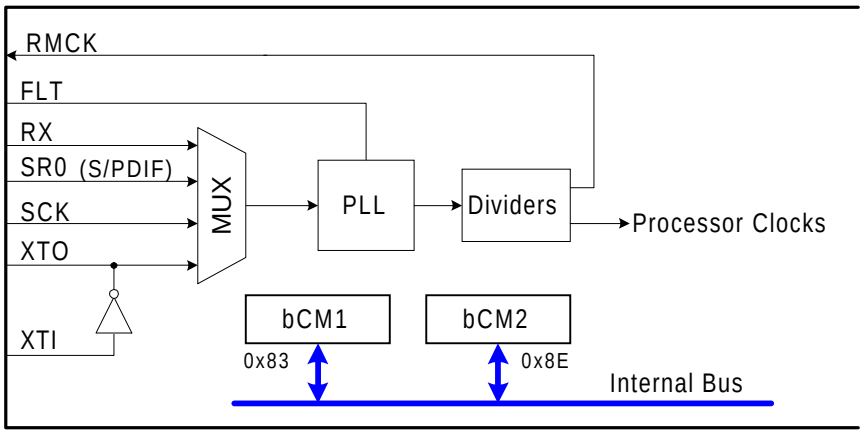


Figure 9-1: Clock Manager

9.1 bCM1 (Clock Manager 1 register)

0x83	bCM1	Clock Manager 1 Register	
Bit	**Name**	**Description**	**Default**
7	PLD	PLL disable	0
6..4	RD2..0	RMCK divider	000
3..2	XTL1..0	Oscillator divider	00
1..0	MX1..0	PLL input select	00

Table 9-1: bCM1 (Clock Manager Register 1)

PLD (PLL disable)

When set to 1, the PLL stops trying to lock to the selected source and drifts to its lowest frequency.

RD2..0 (RMCK divider)

These bits select the frequency of the *RMCK* output pin. RMCK is derived off the OS8104 master internal timing source; therefore, this output may be used by external devices to operate synchronously to the OS8104. Changing these bits can produce glitches on RMCK.

RD2	RD1	RD0	RMCK frequency (default: 50 % duty cycle)
0	0	0	384 Fs
0	0	1	256 Fs
0	1	0	128 Fs
0	1	1	64 Fs
1	0	0	1536 Fs
1	0	1	1024 Fs (33% duty cycle)
1	1	0	768 Fs
1	1	1	512 Fs

XTL1..0 (Oscillator divider)

The value selected is based on the crystal oscillator attached to the OS8104. Three different Fs (network frame rate) over-sampling ratios are supported.

00 – 256Fs
01 – 384Fs
10 – 512Fs
11 - Reserved

MX1..0 (PLL input select)

These bits determine the signal source the PLL is synchronized to. RX is the default source after reset.

00 – RX (MOST)
01 – SR0 (S/PDIF)
10 – Crystal Oscillator
11 - SCK

9.2 bCM2 (Clock Manager 2 register)

0x8E	bCM2	Clock Manager 2 Register	
Bit	**Name**	**Description**	**Default**
7	/LOK	PLL lock status (read-only)	1
6	NAC	Network activity (read-only)	0
5	ZP	Zero Power mode enable	0
4	LP	Low Power mode enable	0
3..0	rsvd	Reserved; Write as 0	0000

Table 9-2: bCM2 (Clock Manager Register 2)

/LOK (PLL lock status)

When set to 0, indicates the PLL is locked to the selected source. When set to 1, the PLL is unlocked.

NAC (Network Activity)

When set to 1, indicates that activity exists on the network (rising or falling edges detected on RX). This bit does not indicate whether that data is being transferred correctly or not. When NAC is 0, no activity is detected.

ZP (Zero Power mode enable)

When set to 1, the OS8104 is placed in low power mode and enters zero power if network activity ceases. The OS8104 powers up again after network activity detection, or access to the chip via the Control Port or parallel port.

LP (Low Power mode enable)

When set to 1, the OS8104 is placed in low power mode. The OS8104 powers up after access to the chip via the Control Port or parallel port, or reception of the wake up signal from the network.

9.3 PLL Lock Status

The /LOK bit in bCM2 indicates the current lock status of the PLL. When /LOK is 0, the PLL is locked, and when /LOK is 1 the PLL is not locked. When set to 1, the EXL bit in bXSR indicates a lock error occurred since the last time it was cleared. EXL is sticky and can be cleared (armed for capturing a new lock error) by writing EXL to zero.

When the OS8104 initially obtains lock, it takes three frames for the OS8104 to synchronize source data. Therefore, if the OS8104 is not in all bypass mode (bXCR bit /ABY = 1) and not in source data bypass (bXCR bit SBY = 0), then source data won't be transferred properly between the network and the Source Ports for three frames.

For timing-slave nodes, if no light is present, the PLL drifts to its lowest frequency. Source data cannot be transferred while the PLL is unlocked. Only Control data can be transferred (via the Control Port or the parallel interface).

9.4 VREF and FLT Pins

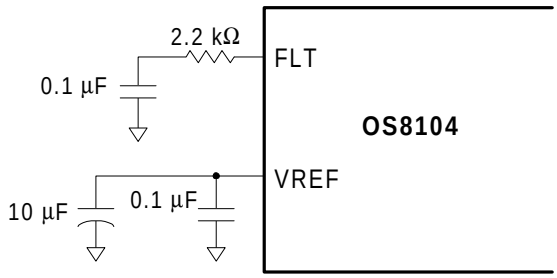


Figure 9-2: Standard application for VREF pin and FLT pin

The circuitry illustrated in Figure 9-2 is the optimum setup for FLT and VREF. The 0.1 µF capacitors should be of a ceramic type. The VREF 0.1 µF capacitor should be placed as close as possible to the VREF pin and AGND. Similarly, the FLT capacitor and resistor should be placed as close as possible to the FLT and AGND pins.

9.5 Crystal Oscillator (XTI/XTO)

The crystal oscillator is used by the timing-master node to set the timing for the entire ring. Slave nodes generally have a crystal oscillator to allow the node to run diagnostics when the ring is down.

The crystal oscillator should be fundamental mode, parallel resonant with a load capacitor specified as mentioned below. Figure 9-3 depicts the external circuitry connected to the OS8104 oscillator circuit. Since the internal inverter/amplifier is operated in its linear region, external series resistors must not be used, as they will lower the gain and cause start-up problems.

When the crystal oscillator is not selected as the timing source for the node (MX1..0 not equal to 10), the oscillator is powered down to minimize noise and power.

If an external clock is used in lieu of a crystal oscillator, it must support CMOS drive levels and be connected to XTO, with XTI grounded (see Figure 9-3). For the timing-master node, this clock must be jitter free. For a slave node, the clock must also be jitter free to support ring-down diagnostics where the slave might be the timing-master for the rest of a broken ring.

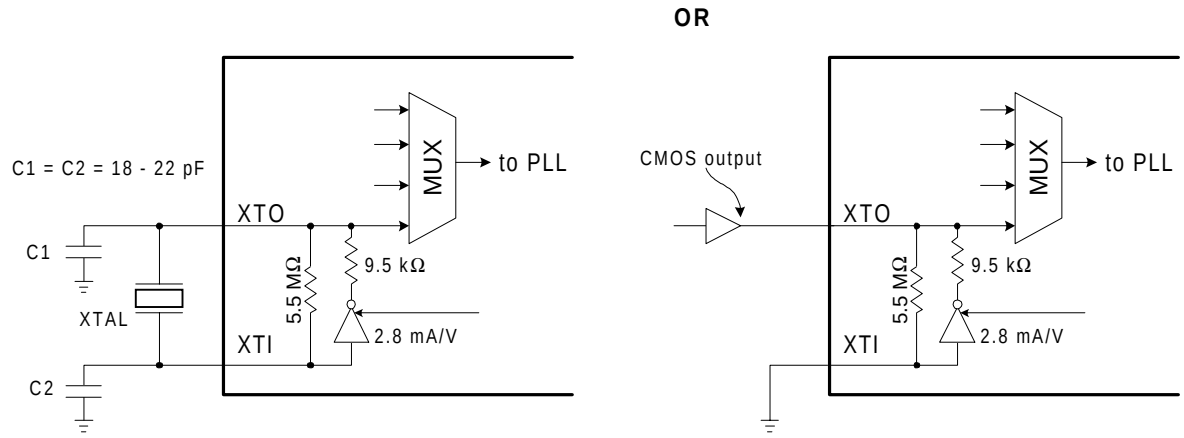


Figure 9-3: Crystal Oscillator Input

The crystal frequency can be 256xFs, 384xFs, or 512xFs. The selection is made via the XTL1..0 bits in bCM1. The load capacitor, specified when ordering the crystal, is the series combination of the capacitance on each leg of the crystal, in parallel with the crystal shunt capacitance (C_O). The capacitance on each leg includes the pin capacitance of XTO or XTI (about 4 pF), the capacitance of the board traces (anywhere from 4 to 10 pF), and the external added capacitor (C1 or C2). Using the typical load capacitor (C_L) value of 20 pF and a shunt capacitance of 7 pF, each leg of the crystal would have a total capacitance of 26 pF. The series combination of two 26 pF loads is the required 13 pF (C_L - C_O). Subtracting out the pin capacitance and the PCB trace capacitance would leave the external capacitors on each leg (C1 and C2) in the 18 to 22 pF range.

Name	Value	Description
Correlation	Parallel Resonant	Mode of oscillation
Osc. Mode	Fundamental	Oscillation mode or Operation mode.
C_L	20 pF	Load Capacitance. Series combination of external capacitors with board capacitance and pin capacitance added.
Max. ESR	30 Ω 20 Ω	Maximum Equivalent Series Resistance. When crystal frequency is 256Fs When crystal frequency is 384Fs or 512Fs
Maximum Drive Level	1 mW 2 mW	When crystal frequency is 256Fs or 384Fs When crystal frequency is 512Fs
T_A	-40 to 85 °C	Operating temperature range
cut	AT	AT cut produces the best temperature stability.
Tolerance	±50 ppm	Frequency tolerance at 25 °C. Typical value.

Table 9-3: Crystal Oscillator Specifications

XTL1..0:	00	01	10	Units
Fs	256	384	512	
38 kHz	9.728	14.592	19.456	MHz
44.1 kHz	11.2896	16.9344	22.5792	MHz
48 kHz	12.288	18.432	24.576	MHz

Table 9-4: Crystal Oscillator Frequencies

The crystal cut and tolerance value listed in Table 9-3 are typical values and may be changed to suit differing system requirements. Higher ESR values, than those listed in the Table, run the risk of having start-up problems.

10 Power Management

Two power-saving modes are available in the OS8104 to minimize the nodes power when no activity exists on the network or from the external application. The Zero-Power mode is available when not using the receive FOT to manage system power, as described in the MOST Specification and illustrated in Section A.1. Per the MOST Specification, the receive FOT powers down the entire node when no light is detected on the RX pin. When not using an FOT with power management capability (or when not using FOTs at all), the Zero-Power mode can minimize power when the network is inactive (although it will not be as low as when using the receive FOT to control power).

The Low-Power mode will place the device in the lowest power state where the network is still active, but the particular node is not being used. As of the writing of this Data Sheet, the Low-Power mode is not supported by he existing MOST Specification.

The power saving modes are entered by writing the appropriate bit in the bCM2 register. The power-saving mode is exited when activity is detected on the network (RX pin) or by a Control Port accesses from the external application. The following table describes the function of the power management pins:

Pin	Comment
STATUS	When high, indicates that the chip is in Zero-Power mode. *STATUS* can be used to control the external application's power supply.
/WAKE_UP	This signal enables or disables the power supply of the external signal reception circuitry e.g. a FOT unit. If the chip pulls */WAKE_UP* low, the reception circuitry should be active.
R_TIMER	Connector for external resistor (390 kΩ or 400 kΩ). Used by the internal clock section of the wake-up logic.

Table 10-1: Power Management Pins

Power and network activity status as well as the enable bits for Zero-Power mode and Low-Power mode can be accessed in the bCM2 register (Section 9.2 on page 90). Bit LPW in bXCR (Section 6.2.1 on page 35) initiates the transmission of a wakeup-frame sent out onto the network. LPW should only be set by the timing-master node.

10.1 Low Power Mode

10.1.1 Entering Low Power Mode

The chip switches to Low-Power mode when bit LP in register bCM2 is set to 1. This can be done via Control Port. The following list shows the characteristic features of this mode:

- Only the transceiver section of the MOST chip and the wake-up logic are active. The rest of the chip (e.g. Source Data Ports) is powered down.
- The SBY bit in register bXCR is activated:
  - Node is still visible for the network
  - Node delay count for source data is reduced
    Since the transceiver is still active, the device is visible for the rest of the network. Therefore, the device is counted during Node Position counting. The Node Delay count will NOT be incremented due to activation of SBY (source data bypass).

10.1.2 Leaving Low Power Mode

The chip will wake up whenever one of the following events occur:

- Wake-up frame is received (generated by the timing-master node)
- Two falling edges are detected on *SD*A of the Control Port in I^2C mode.
- Two falling edges are detected on */CS* of the Control Port in SPI mode.
- Two write accesses are detected on /WR when in parallel port .

The SBY bit stays active, after the waking up procedure is finished.

At the end of the wake up procedure, the /INT interrupt pin will be driven low. The register values bNAL, bGA, bXTIM, bXRTY, bAPAH, bAPAL are unaltered. All the other registers will be reset to default values. Since the network configuration may have changed during Low Power mode, the Routing Engine table is reset to its default values.

10.2 Zero Power Mode

10.2.1 Entering Zero Power Mode

Zero-Power mode is enabled, when bit ZP in register bCM2 is set to 1. When ZP is set to 1, the chip activates Low-Power mode first. If no activity is detected on the network, the chip will then automatically go to Zero-Power mode. If ZP is set while there is still network activity, the chip will wait for network activity to cease. As long as the chip is in Low-Power mode, it will wake up when a wakeup frame is sent via the network. The following list shows the characteristic features of Zero-Power mode:

- Low-Power mode is entered immediately after setting the ZP bit in bCM2.
- Zero-Power mode is entered only if there is no network activity detected.
- In Zero-Power mode, only the wake-up logic is active
- The wake-up logic scans the network input for activity in regular intervals.

10.2.2 WAKE_UP and R_TIMER Pins

In Zero-Power mode, the wake-up logic scans the network input for activity every 50 ms (20 Hz, if resistor at *R_TIMER* has 400 kΩ). Therefore, the */WAKE_UP* pin is driven low for approx. 150 µs. It is possible to save more power when switching the external receiving circuitry's power off when */WAKE_UP* is high. If no activity is detected, OS8104 will return to Zero-Power mode.

Changing the resistor at *R_TIMER* to 390 kΩ has no significant effect on scan interval (20.5 Hz/48.78 ms instead of 20 Hz/50 ms). If the wake-up logic and Zero-Power mode is not used, *R_TIMER* must be tied to VDDA (analog power supply pin).

10.2.3 Leaving Zero-Power Mode

The chip wakes up whenever one of the following events occurs:

- There is network activity.
- Two falling edges are detected on *SD*A of the Control Port in I^2C mode.
- Two falling edges are detected on */CS* of the Control Port in SPI mode.
- Two write accesses are detected on /WR when configured for parallel port operation.

After finishing the wake-up procedure, the /INT pin will be driven low. The registers bNAH, bNAL, bGA, bXTIM, bXRTY, bAPAH, bAPAL are unaltered. All the other registers, including the Routing Engine table (RE), will be reset to default values.

11 Handling Interrupts

OS8104 has two interrupt pins:

- */AINT*
  Used when sending asynchronous (packet) data. */AINT* is described in Section 5.
- /INT
  Indicates power-up initialization complete and Control message received or error.

There is a variety of events that can be signaled by the */INT* pin. An interrupt can be generated when a "network configuration changed" event occurs, when a control message was transmitted/received, or in case of errors. Section 6.2.4 on page 37 describes which error events can affect the /INT pin. Register bIE determines which events affect the interrupt pin, while the Message Control Register bMSGC (see Section 13.2.1 on page 110) can clear interrupt events.

11.1 bIE (Interrupt Enable register)

0x88	bIE	Interrupt Enable register	
Bit	**Name**	**Description**	**Default**
7..4	rsvd	Reserved; Write as 0	0000
3	IALC	Network configuration changed	0
2	IERR	Error or Power-on initialization complete	0
1	IMTX	Message transmitted	0
0	IMRX	Message received	0

Table 11-1: bIE (Interrupt Enable register)

IALC (Interrupt on: network configuration changed)
: If set to 1, an interrupt will be generated when bit ALC in bMSGS is set to 1. ALC indicates that bMPR or bMDR have changed.

IERR (Interrupt on: error or power-up after startup)
: If set to 1, an interrupt will be generated when bit ERR in bMSGS is set to 1. Although /INT will always go low after power-on initialization is complete, ERR can also indicate network receive errors, controlled through the bXSR register.

IMTX (Interrupt on: message transmitted)
: If set to 1, an interrupt will be generated when bit MTX in bMSGS is set to 1. MTX indicates that a Control message has been transmitted (TXR in bMSGS indicates successful transmission; otherwise register bXTS indicates transmission problem)

IMRX (Interrupt on: message received)
: If set to 1, an interrupt will be generated when bit MRX in bMSGS is set to 1. MRX indicates that a Control message has been received (received message type indicated in bRTYP).

All interrupts can be cleared by setting the respective reset bits in register bMSGC (see Section 13.2.1 on page 110).

11.2 Power-On Interrupt

When the OS8104 is powering up or after being reset, the */INT* pin will be kept high until the chip has finished its initialization. Then */INT* will be pulled low to indicate that OS8104 is ready for external accesses.

11.3 Behavior On Multiple Interrupts

If multiple events are enabled, more than one event could be pending at any given time. Figure 11-1 illustrates the occurrence of two interrupt events. ALC is triggered first, which generates an interrupt. During the time the ALC interrupt is pending, another event (MTX) occurs.

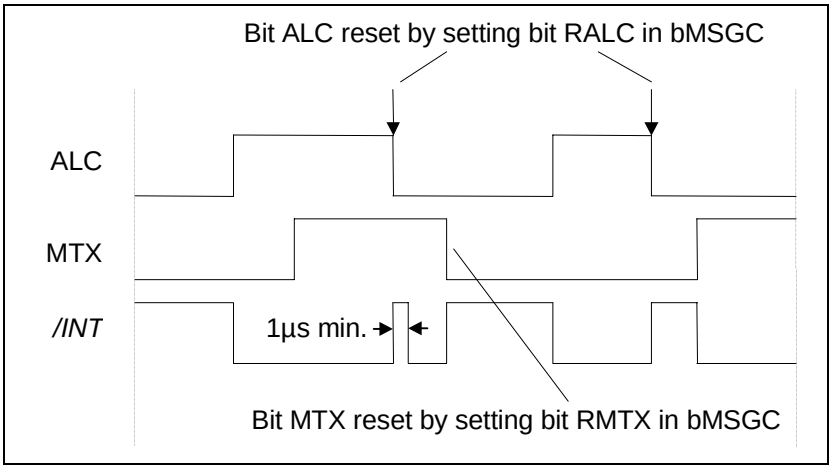


Figure 11-1: Multiple-interrupt events

When the first event (ALC) is cleared, the */INT* pin will return high (inactive) for a short period of time, since the second event is still pending. OS8104 will generate a new interrupt, which will last until the second event is cleared.

12 Routing Synchronous Data

Synchronous source data can be "routed" from the Source Data Port inputs (SRn pins) to the MOST network transmitter. Likewise, source data can be routed from the MOST network receiver to the Source Data Port outputs (SXn pins). Lastly, data can be routed directly from the Source Data Port inputs to the Source Data Port outputs. The OS8104 contains a Routing Engine that acts like a cross-point switch to manage the transfer of data from data inputs to data outputs.

The Routing Engine contains a set of Routing Engine registers (RE0x00 to RE0x7F) that determine where routing data goes to (sink devices). This set of registers is divided into two halves. The lower half (RE0x00 to RE0x3C) routes data to the MOST network transmitter to be sent out onto the network. The upper half (RE0x40 to RE0x7F) of the RE registers route data to the Source Data Port outputs (SXn) or to the parallel port (Source Port in Parallel-Synchronous or Parallel-Combined mode).

The Routing Engine registers are filled with "address references" that determine what data (data source) is to be sent out to that particular sink location. Address references are also divided into two halves. Address references 0x00 through 0x3B (up to 60 bytes) are associated with synchronous source data coming from the MOST network receiver (RX pin). Some of these address references may not be used if network source data bandwidth is allocated to asynchronous packet data. Register bSBC determines the dividing point between synchronous and asynchronous data. The upper half of the address references, 0x40 to 0x7F, are associated with the data arriving on the Source Data Port input pins (SRn) or from the parallel port when the Source Port is in Parallel-Synchronous or Parallel-Combined mode.

When "address references" are placed in the RE registers, the Routing Engine transfers the data from the address-referenced source (network receive data, SRn pins, or from the parallel port) to the RE-register destination (network transmit data, SXn pins, or to the parallel port).

After reset, the RE registers are configured such that the network's receive data in a frame is passed directly to the network transmitter. Therefore, physical channel 1 of the incoming network data stream on RX will be sent out as physical channel 1 of the outgoing source data stream on TX. Channel 2 will be sent out as channel 2, etc.

All incoming data is buffered before being sent out. If the node is placing data onto the network (SBY set to 0), then the incoming source data is delayed two frames before being sent out (to the network or the Source Port outputs). In Figure 12-1, the arrows between the RE registers and input buffer point to those memory locations the data will be copied from.

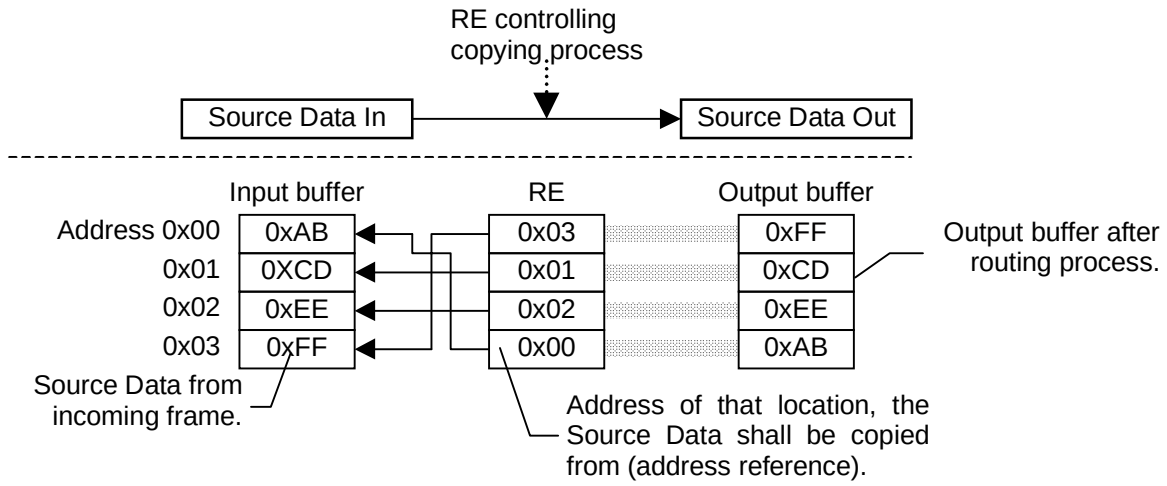


Figure 12-1: Routing Process (Simplified) Example

The number of usable synchronous source data bytes per frame is specified via register bSBC (Section 6.2.5 on page 37). RE registers associated with source data not reserved for synchronous data must not be used.

12.1 Incoming Network Data to Outgoing Network Data

Each MOST frame contains a maximum of 60 bytes of source data. When routing data from the incoming frame to the outgoing frame, the respective address reference value is equal to the data byte number within the frame (valid numbers are 0x00 through 0x3B). For example, to output byte number 8 of the received data on byte 8 of the transmitted data, the number (address reference) 0x08 (received data) must be written to RE register 0x08 (transmitted location).

The table below illustrates the lower half of the Routing Engine (RE) registers. The locations 0x00 up to 0x3B refer to the synchronous source data locations to transmit onto the outgoing MOST network frame.

RE-Regs.	+0	+1	+2	+3	+4	+5	+6	+7
0x00	D0x00	D0x01	D0x02	D0x03	D0x04	D0x05	D0x06	D0x07
0x08	D0x08	D0x09	D0x0A	D0x0B	D0x0C	D0x0D	D0x0E	D0x0F
0x10	D0x10	D0x11	D0x12	D0x13	D0x14	D0x15	D0x16	D0x17
0x18	D0x18	D0x19	D0x1A	D0x1B	D0x1C	D0x1D	D0x1E	D0x1F
0x20	D0x20	D0x21	D0x22	D0x23	D0x24	D0x25	D0x26	D0x27
0x28	D0x28	D0x29	D0x2A	D0x2B	D0x2C	D0x2D	D0x2E	D0x2F
0x30	D0x30	D0x31	D0x32	D0x33	D0x34	D0x35	D0x36	D0x37
0x38	D0x38	D0x39	D0x3A	D0x3B				

Table 12-1: Lower Half of Routing Engine Registers (Network Transmit locations)

To send byte D0x00 of the incoming frame to byte D0x11 in the outgoing frame:

- Determine the address reference of byte D0x00 in the incoming frame – address reference 0x00
- Determine the RE register that refers to byte D0x11 of the outgoing frame – register 0x11
- Write the address reference 0x00 to RE register location 0x11.

Originally, incoming channel 0x11 was sent out the outgoing channel 0x11. After writing RE register 0x11 to 0x00, the contents of incoming channel 0x00 is sent out the outgoing channel 0x11, and the incoming data channel D0x11 is discarded (unless it is routed elsewhere).

12.2 Serial Source Port Inputs To Network

Since the Serial Source Port inputs are "data sources", they are associated with address references that need to be placed into the RE register to transfer data. The address references used for the Source Data Port inputs (SRn pins) are dependent on the selected speed of the Source Ports.

Source Port Inputs - Address References									
bytes/frame	64	32	16	8	6		4	2	1
Bit rate [xFs]	512	256	128	64	48†		32	16	8
FSY rate [xFs]	8	4	2	1	In	Out	1/2	1/4	1/8
SR **n**	0*	0	0,2	0,1, 2,3	0,1, 2,3	0,1, 2,3	0,1, 2,3	0,1, 2,3	0,1, 2,3
Addressing Index **x** [hex]	47/46	43	41	40	40	40	48	58	78
	45/44	42							
	43/42	41	40						
	41/40	40							
	4F/4E	4B	49	48	48	48			
	4D/4C	4A							
	4B/4A	49	48						
	49/48	48							
	57/56	53	51	50					
	55/54	52							
	53/52	51	50		50	58	58		
	51/50	50							
	5F/5E	5B	59	58					
	5D/5C	5A							
	5B/5A	59	58						
	59/58	58							
	67/66	63	61	60	58	60	68	78	
	65/64	62							
	63/62	61	60						
	61/60	60							
	6F/6E	6B	69	68	60	68			
	6D/6C	6A							
	6B/6A	69	68						
	69/68	68							
	77/76	73	71	70	68	78	78		
	75/74	72							
	73/72	71	70						
	71/70	70							
	7F/7E	7B	79	78					
	7D/7C	7A							
	7B/7A	79	78						
	79/78	78							
Address Reference = **x** + **n**									

*SR0 in 8x S/PDIF input or output mode. All numbers in hexadecimal.

† If I/O bit in bSDC1 is 1, column 'out' applies, else column 'in' applies.

Table 12-2: Source Port Input (SRn) Address References

In Table 12-2, the first three rows specify the speed in bytes/frame, bit rate, and FSY rate. The fourth row specifies which of the source data input ports are available at the respective clock rate.

The address reference used for a particular SRn location is calculated by the formula shown at the bottom of the table, and is equal to the number in the table, offset by the particular Source Port Data pin number (SRn).

Network resource should be reserved (allocated) before routing source data to the network. Reserving source data resources is accomplished through the Resource Allocation feature of the OS8104.

Since the source data bytes of the outgoing network frame are assigned to the RE register locations 0x00 to 0x3B, an address reference written to one of these locations routes the respective source data input byte to the outgoing frame.

The total number of available synchronous source data bytes per frame is specified via register bSBC (Section 6.2.5 on page 37). If not all bytes of the MOST frame are reserved for synchronous source data transfer (some may be reserved for asynchronous data), the respective RE registers and address references must not be used.

RE-Regs.	+0	+1	+2	+3	+4	+5	+6	+7
0x00	D0x00	D0x01	D0x02	D0x03	D0x04	D0x05	D0x06	D0x07
0x08	D0x08	D0x09	D0x0A	D0x0B	D0x0C	D0x0D	D0x0E	D0x0F
0x10	D0x10	D0x11	D0x12	D0x13	D0x14	D0x15	D0x16	D0x17
0x18	D0x18	D0x19	D0x1A	D0x1B	D0x1C	D0x1D	D0x1E	D0x1F
0x20	D0x20	D0x21	D0x22	D0x23	D0x24	D0x25	D0x26	D0x27
0x28	D0x28	D0x29	D0x2A	D0x2B	D0x2C	D0x2D	D0x2E	D0x2F
0x30	D0x30	D0x31	D0x32	D0x33	D0x34	D0x35	D0x36	D0x37
0x38	D0x38	D0x39	D0x3A	D0x3B				

Table 12-3: Lower Half of RE Registers for Network Transmit Data.

Using an example where the Source Ports are configured in Mode 3 (Source Ports 0 and 2 running at 128Fs). If bytes 0x00 to 0x02 of the incoming frame at Source Port 2 should be transmitted at position 0x39 to 0x3B in the outgoing frame. The following steps must be performed:

- Determine the address reference of byte 0x00 up to 0x02 at Source Port 2 in Mode 3.
  - the correct values are 0x43, 0x42, 0x4B
- Determine which RE registers refer to bytes D0x39 up D0x3B of the outgoing frame
  - The correct values are 0x39 to 0x3B).
- Write the address references to the respective RE registers as follows:

RE registers:	0x39	0x3A	0x3B
Contents of RE registers:	0x43	0x42	0x4B

The number of available synchronous source data bytes per frame is specified in register bSBC (Section 6.2.5 on page 37). If all source bytes of the MOST frame are not reserved for synchronous source data transfer, the respective bytes in RE must not be used.

12.3 Network to Serial Source Port Outputs

Similar to the previous section, the speed of the Source Data Ports determines the address reference used for routing incoming network data out of the serial Source Ports (SXn). When routing data to the Source Output ports, two values must be determined: the correct address reference for the data to be shifted out, and the appropriate Routing Engine registers that the address reference should be written to.

Source Port Outputs - RE Register Locations									
bytes/frame	64	32	16	8	6		4	2	1
clock rate [xFs]	512	256	128	64	48†		32	16	8
FSY rate [xFs]	8	4	2	1	In	Out	1/2	1/4	1/8
SX **n**	0*	0	0,2	0,1, 2,3	0,1, 2,3	0,1, 2,3	0,1, 2,3	0,1, 2,3	0,1, 2,3
Addressing Index **x** [hex]	47/46	47	47	47	47	47	47	47	47
	45/44	46							
	43/42	45	46						
	41/40	44							
	4F/4E	4F	4F	4F	4F	4F			
	4D/4C	4E							
	4B/4A	4D	4E						
	49/48	4C							
	57/56	57	57	57			57		
	55/54	56							
	53/52	55	56		57	5F			
	51/50	54							
	5F/5E	5F	5F	5F					
	5D/5C	5E							
	5B/5A	5D	5E						
	59/58	5C							
	67/66	67	67	67	5F	67	67	67	
	65/64	66							
	63/62	65	66						
	61/60	64							
	6F/6E	6F	6F	6F	67	6F			
	6D/6C	6E							
	6B/6A	6D	6E						
	69/68	6C							
	77/76	77	77	77			77		
	75/74	76							
	73/72	75	76		6F	7F			
	71/70	74							
	7F/7E	7F	7F	7F					
	7D/7C	7E							
	7B/7A	7D	7E						
	79/78	7C							
RE register = **x** - **n**									

* SR0 in 8x S/PDIF input or output mode, all numbers in hexadecimal.

† If I/O bit in bSDC1 is 1, column 'out' applies, else column 'in' applies.

Table 12-4: Source Port Output (SXn) RE Registers

The appropriate Routing Engine registers can be determined by referring to Table 12-4.

The first row of the table contains the number of bytes shifted out per frame, while the second row contains the respective serial clock rate. The third row lists the Frame rate and the fourth row specifies which of the Source Port output pins are available at the respective clock rate. The RE register used for a particular SXn location is calculated by the formula shown at the bottom of the table, and is equal to the number in the table, offset by the particular Source Port Data pin number (SXn).

The address reference, that must be written to the RE register, depends on where the source of the data comes from. If the source data comes from a serial source data input port, Table 12-2 must be used. If the data comes from the incoming MOST frame, the address references 0x00 up to 0x3B are relevant (refer to Table 12-1, on page 98).

Figure 12-2 shows the conjunction between the entries in Table 12-2 and the incoming bytes at a serial source data port. SCK runs at 64Fs. The top of the table was turned counterclockwise (90 degrees).

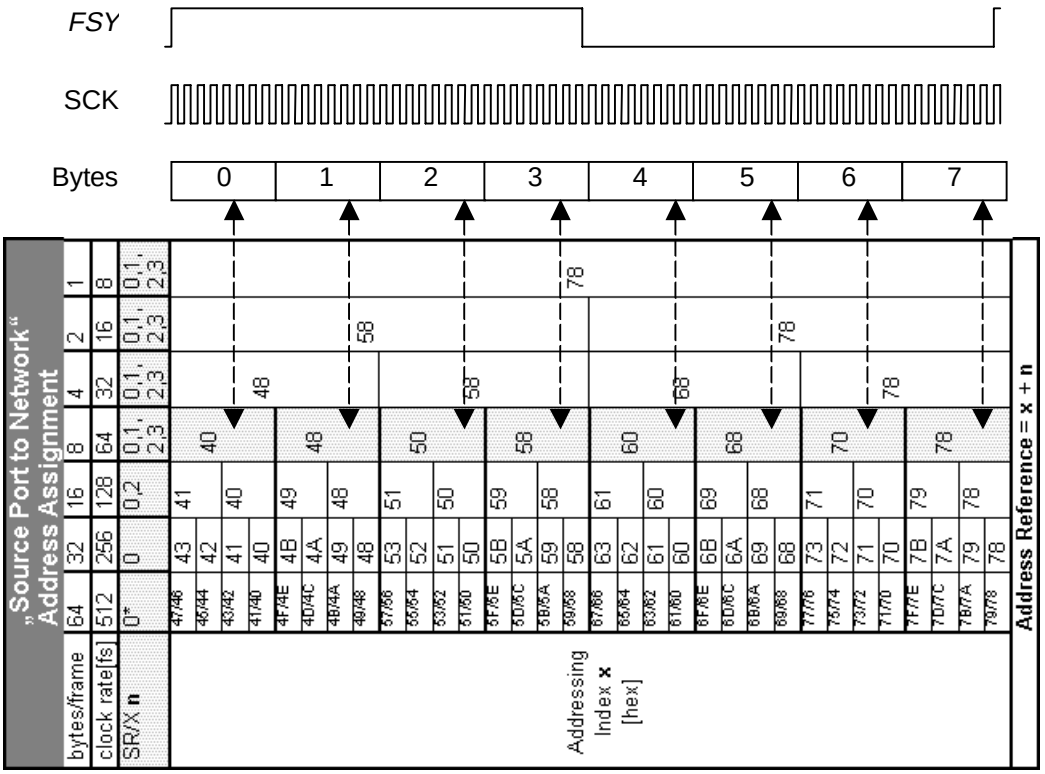


Figure 12-2: Table versus data frame at serial Source Port

The same applies to Table 12-4. When turning the top of the table counterclockwise (90 degrees), then the sequence of the entries in the table meets the sequence of data bytes in the outgoing frame at a serial source data port (SCK speed must always be taken into consideration).

For example:

The local Source Ports are configured in Mode 1 and are running at 64Fs, and byte 0x04 of the incoming MOST frame is to be transported to Source Port 3 and shifted out as the second byte.

The RE register location, related to SX3, must be calculated for respective byte at the chosen speed.

Using Table 12-4, the RE register is 0x4C (0x4F - 3). Since byte 0x04 of the incoming MOST frame has address reference 0x04, RE would look like:

RE-Regs.	+0	+1	+2	+3	+4	+5	+6	+7
0x40								
0x48					0x04			
0x50								
0x58								
0x60								
0x68								
0x70								
0x78								
					↑ SX3	↑ SX2	↑ SX1	↑ SX0

Table 12-5: Upper Half of Routing-Engine Registers (Example).

Table 12-5 illustrates the Routing-Engine registers that are used to send data out the Source Port pins SX3..0, when configured in Mode 1, with an SCK speed of 64Fs. The eight bytes for SX3, starting at the beginning of FSY, are 0x44 for the first byte, 0x4C for the second byte, 0x54 for the third byte, up to 0x7C for the eighth byte. Figure 12-3 illustrates the RE register locations associated with each byte of the Source Port outputs in Mode 1 with an SCK speed of 64Fs.

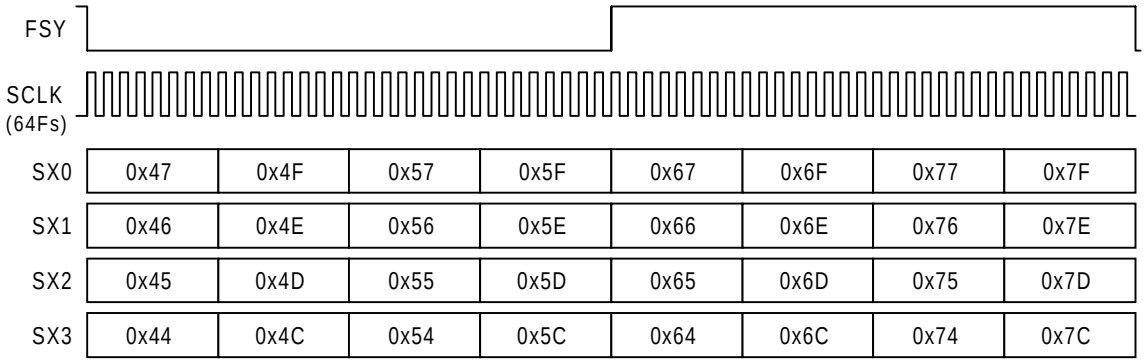


Figure 12-3: Source Port Output Routing Example (RE Register Locations)

12.4 Synchronous Parallel Port Data Transfers

When the Source Ports are configured in Parallel-Synchronous or Parallel-Combined mode, the Source Port pins are configured to form an 8-bit parallel port, that transfers data through a FIFO that is 8 bytes deep. Each frame is divided up into eight intervals of identical length, called SF0 up to SF7. During each SF interval, a total of 16 bytes may be transferred (8 received, 8 sent), for a total of 64 bytes per frame A more detailed description can be found in Section 8.2.1 for Parallel-Synchronous and Section 8.3 for Parallel-Combined mode. For source data transferred from the external device, through the parallel port, and onto the network, address references for each of the parallel bytes are needed to load into the lower half of the Routing Engine Registers, which determine what gets transmitted out the MOST network (TX pin). For sink data transferred from the MOST network receiver, through the parallel port, and to the external device, the upper half of the Routing Engine registers must be programmed with the address references of the network received data. Each register in the upper half of the Routing Engine corresponds to one of the 64 parallel port bytes that can be transferred each frame.

12.4.1 Synchronous Parallel Data To Network

When sending data from an external device, through the parallel port, in Parallel-Synchronous or Parallel-Combined mode, a maximum of 64 bytes per frame can be written. For assigning a particular parallel port byte to one of the available positions within the MOST network transmit frame (the maximum number of source data bytes in the frame is 60), address references for the parallel port data are needed. The following table shows the relation between the position of a byte in the FIFO, the SF interval during which the byte is transferred, and the address reference needed to write into the lower half of the Routing Engine registers.

	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
SF0	0x40	0x41	0x42	0x43	0x44	0x45	0x46	0x47
SF1	0x48	0x49	0x4A	0x4B	0x4C	0x4D	0x4E	0x4F
SF2	0x50	0x51	0x52	0x53	0x54	0x55	0x56	0x57
SF3	0x58	0x59	0x5A	0x5B	0x5C	0x5D	0x5E	0x5F
SF4	0x60	0x61	0x62	0x63	0x64	0x65	0x66	0x67
SF5	0x68	0x69	0x6A	0x6B	0x6C	0x6D	0x6E	0x6F
SF6	0x70	0x71	0x72	0x73	0x74	0x75	0x76	0x77
SF7	0x78	0x79	0x7A	0x7B	0x7C	0x7D	0x7E	0x7F

Table 12-6: Parallel Port Synchronous Data In Address References

For example, to route the data byte positioned in Byte 5 during SF interval 6 to the outgoing frame, the correct address reference is 0x75. To send this byte onto the first byte of the outgoing network frame, 0x75 must be written to Routing Engine register 0x00 (first Routing Engine register).

12.4.2 Network to Synchronous Parallel Data

Likewise, when sending data to an external device, from the parallel port, in Parallel-Synchronous or Parallel-Combined mode, a maximum of 64 bytes per frame can be read. The upper half of the Routing Engine registers are mapped (using SF intervals and FIFO byte position) to each of the parallel port bytes. Filling these upper Routing Engine registers with the address references of the network receive data maps the data to the parallel port. The following table shows which Routing Engine register is associated with which parallel port byte, based on the SF interval and the particular FIFO byte number. Generally all these locations are filled with the full network bandwidth (address references 0x00 through 0x3C). In Parallel-Combined mode, the last four locations must be packet status addresses 0x3D through 0x3F).

RE-Regs.	+0	+1	+2	+3	+4	+5	+6	+7	SF:
0x40									← 0
0x48									← 1
0x50									← 2
0x58									← 3
0x60									← 4
0x68									← 5
0x70									← 6
0x78									← 7
FIFO byte:	↑ Byte0	↑ Byte1	↑ Byte2	↑ Byte3	↑ Byte4	↑ Byte5	↑ Byte6	↑ Byte7	

Table 12-7: Upper Half of RE Registers for Parallel Port Output Data.

For example, to route the first eight bytes of the received MOST frame to the Source Port in Parallel-Synchronous mode, one byte will be sent per SF interval. This meets the requirement that at least one read or write access must occur during each SF interval. When reading from the parallel port, FIFO byte 7 is the first byte read. Therefore, FIFO byte 7 of each SF interval will be used to transfer the received MOST network data. The bottom of Table 12-7 defines which column refers to which byte within the FIFO. The right-most column gives information about the respective SF interval.

Using the FIFO "Byte7" column of Table 12-7, the address references for the first eight network receive bytes (0x00-0x07) are placed in the Byte7 column, as illustrated in Table 8-8.

RE-Regs.	+0	+1	+2	+3	+4	+5	+6	+7	SF:
0x40								0x00	0
0x48								0x01	1
0x50								0x02	2
0x58								0x03	3
0x60								0x04	4
0x68								0x05	5
0x70								0x06	6
0x78								0x07	7

Table 12-8: Parallel Synchronous Source Data Routing Example

12.5 Transparent Channel Data Routing

Transparent Channel Input to MOST Network				
bytes/frame	8	4	2	1
clock rate [Fs]	64	32	16	8
Address Reference **x** [hex]	41	49	59	79
	49			
	51	59		
	59			
	61	69	79	
	69			
	71	79		
	79			

Depending on the selected sample rate, a different number of bits are sampled at SR1 or transmitted at SX1. At the highest sample rate, a maximum of 8 bytes are available per frame. Only one byte per frame is sampled at the lowest sample rate. When routing data from SR1 to the outgoing MOST frame, Table 12-9 lists the correct address references for the selected clock rate.

Table 12-9: SR1 Transparent Channel Address References

Transparent Channel Output To SX1				
bytes/frame	8	4	2	1
clock rate [Fs]	64	32	16	8
RE Register Location [hex]	46	46	46	46
	4E			
	56	56		
	5E			
	66	66	66	
	6E			
	76	76		
	7E			

When routing transparent data from the incoming MOST frame to SX1, the network receive data address reference must be programmed into the appropriate SX1 Routing Engine register. Table 12-10 lists the appropriate RE register locations for the selected clock rate.

For example, in two MOST nodes the transparent channel is configured to run at an 8Fs clock rate. One byte of transparent data per frame. The first node (Node 1) receives a signal via the SR1 pin, and puts the data onto physical channel 0 of the network frame.

Table 12-10: SX1 Transparent Channel RE Register Locations

The second node (Node 2) takes the data from the received physical channel 0, and restores the signal at SX1. Using the 8Fs clock rate column in Table 12-9 and Table 12-10, Node 1 must program RE register 0x00 (physical channel 0) to address reference 0x79. And Node 2 must program RE register 0x46 (SX1 in 8Fs mode) to address reference 0x00 (network receive physical channel 0 address). With this setup, data on Node 1, SR1 pin, will be transmitted across the network and output on Node 2 SX1 pin.

12.6 RE Address Reference 0xF8

The address reference 0xF8 sets the respective target byte to zero. To force a zero into a destination (sink device), set the RE register associated with that sink byte to 0xF8. A 0xF8 in the lower half of the RE registers transmits a zero to the appropriate byte output on the MOST network. A 0xF8 in the upper half of the RE registers transmits a zero to the appropriate Source Port output pin (SXn) in serial mode, or to the appropriate FIFO byte in parallel mode.

13 Control Messages

In addition to transferring synchronous and asynchronous source data, a MOST network supports a special messaging service. Control messages can get status information about devices connected to the MOST network, and can control devices remotely. Traffic on the Control Message channel does not influence the bandwidth of source data transfer. The Control message service supports many different message types. The diagram below shows the existing types of messages.

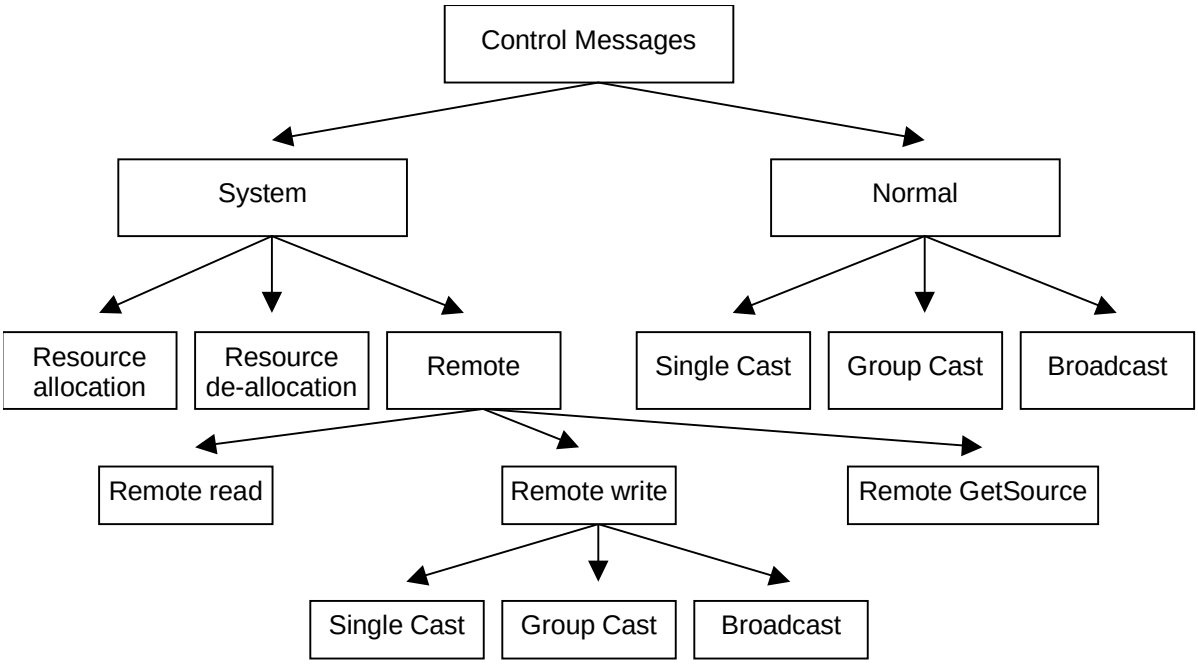


Figure 13-1: Control Message Types

For example, normal messages can control the functionality of a CD player. Normal messages can be sent to a single node (single cast), to a group of nodes (groupcast), or to all nodes in a network (broadcast).

System messages handle MOST related functions like resource management (allocation/de-allocation) and remote operation of a MOST Transceiver (Remote read/write, finding nodes routing source data on allocated channels).

The kind of transmission (single cast, groupcast or broadcast) is determined through special addresses or address ranges. System messages of type "resource allocation" or "resource de-allocation" are always directed to the timing-master node and are not sent to other nodes.

Control messages have a maximum length of 17 bytes. Through OS8104 internal registers, the maximum number of retries, the retry time interval, and the message priority can all be configured. The built-in arbitration guarantees fair handling of the messages. An acknowledge mechanism helps to track the status of a message and the contents of a control message is protected by CRC (Cyclic Redundancy Check).

The following registers are used when transmitting or receiving Control Messages:

- bNAH, bNAL - Node Address Registers (Logical Address register)
- bGA – Group Address Register (determines Groupcast address)
- bMSGC – Message Control Register
- bMSGS – Message Status Register

- bXTS – Transmit Transfer Status Register
- bXTRY – Transmit Retry Register
- bXTIM – Transmit Retry Time Register
- mRCMB – Receive Control Message Buffer
- mXCMB – Transmit Control Message Buffer
- bIE – Interrupt Enable Register

13.1 Transmit Message Addressing

13.1.1 Node Address (Logical Addressing)

Each OS8104 can have a node address (logical address) which is stored in two registers called Node Address High register (bNAH) and Node Address Low register (bNAL). After reset, the default address is 0x0FFF. Valid logical addresses are 0x0001 through 0x02FF, and 0x0500 through 0xFFFF; although 0x0500 through 0xFFFF are currently reserved, and should not be used.

The gap in the address space is caused by special addressing modes described later. The device address can either be set manually by application software, or by the SAI (Start Address Initialization) procedure.

The SAI procedure provides automatic testing whether the desired address is unique, and automatic setting of bNAH and bNAL if the address is okay. For using SAI, the desired address must be written to 0xC3 (low byte to bXTAL in mXCMB) and 0xC2 (high byte to bXTAH in mXCMB). Then bit SAI in the message control register bMSGC must be set to 1. After verification, the OS8104 sets bit MTX (Message transmitted) in message status register bMSGS, and reports the result of the SAI procedure by setting bit TXR. TXR=1 means that the address was unique and successfully stored in bNAH and bNAL.

13.1.2 Node Position Address (Physical Addressing)

Each node in a network has a unique node position address (physical address), which is determined by its position in the network. Counting starts at the timing-master node with node position 0x00. Node position address register is automatically configured by the OS8104, after lock is established.

The valid address range for Node Position Addressing is 0x0400..0x04FF. The destination address for sending control messages to a single node by accessing its physical address, is calculated as follows:

Node physical address = 0x0400 + bNPR

The bNPR register stores the Node Position Address which is the lower byte used in the physical address calculation (see Section 6.2.7 on page 39).

13.1.3 Group Address/Groupcast

Several nodes in an optical network can be combined into a group, by setting their Group Address register bGA to the same value. Valid bGA values are 0x00..0xC7 and 0xC9..0xFF. For sending a control message to a group of nodes, the Groupcast address in the sending node must be calculated as follows:

Node groupcast address = 0x300 + bGA

The bGA registers stores the lower byte of the Groupcast address. The valid address range for Group Addressing is 0x0300..0x03C7 and 0x03C9..0x03FF. Address 0x03C8 is reserved for a network-wide broadcast address.

> When using Groupcast addressing, the number of available bytes to send is 16. One byte is used to secure groupcast.

13.1.4 Special Group Address/Broadcast

One group-address value is reserved for sending messages to all nodes of an optical network. This address has the value 0xC8 with the respective destination address for all nodes of 0x03C8. During the sending of a broadcast message, all other control message transmissions are delayed until the end of the broadcast process, to ensure reception of the message at each node. Therefore excessive use of broadcast messages is discouraged as it will decrease the overall bandwidth of the control message service.

13.1.5 Address Ranges Vs. Addressing Modes

When transmitting messages, there are four addressing modes available for reaching individual nodes as well as groups of nodes. The addressing modes differ by the address range:

Address	Addressing Mode
0x0001..0x02FF	Sending messages to a single node. (Logical addressing) Address 0x000 is not valid.
0x0300..0x03C7 0x03C9..0x03FF	Sending messages to a group of nodes (Groupcast addressing)
0x03C8	Sending messages to all nodes (Broadcast addressing);
0x0400..0x04FF	Sending messages to a single node based on node position; The Node Position Register contains the current node position number. The master node has always position number 0x00, so that the master's address would be 0x0400 (Physical addressing)
0x0500..0x0FFF	Reserved

Table 13-1: Address Ranges vs. Addressing Modes

When sending control messages via Broadcast addressing, the sending of other control messages is disabled until all nodes have received the broadcast message. Excessive use of broadcast addressing keeps the network from sending normal control messages.

13.1.6 bNAH, bNAL (Node Address Registers)

The registers bNAH and bNAL contain the node logical address used for control messages. After reset, the default address is 0x0FFF. An address of 0x0000 is not allowed.

0x8A	bNAH	Node Address High Register	
Bit	**Name**	**Description**	**Default**
7..0	NAH7..0	Logical Node address high byte	0x0F

Table 13-2: bNAH (Node Address High Register)

0x8B	bNAL	Node Address Low Register	
Bit	**Name**	**Description**	**Default**
7..0	NAL7..0	Logical Node address low byte	0xFF

Table 13-3: bNAL (Node Address Low Register)

13.1.7 bGA (Group Address Register)

0x89	bGA	Group Address Register	
Bit	**Name**	**Description**	**Default**
7..0	GA7..0	Lower byte of the Group address. 0x03 is the upper address byte.	xx

Table 13-4: bGA (Group Address Register)

The value in bGA determines which group a node belongs to. In total there are 254 different addresses available (Group address 0xC8 is reserved for broadcast messages which all nodes in the network respond to). A message sent to a group address is received by all nodes that have the same group address (bGA value). The transmit retry time in register bXTIM must be the same for all nodes, if the network supports the Broadcast message type.

13.2 Controlling Transfer Of MOST Control Messages

MOST Control Messages are handled via two buffers. The Receive Control Message Buffer mRCMB contains received control message data along with information about the sender and the addressing mode used. The Transmit (Xmit) Control Message Buffer is used when preparing messages to be sent out onto the network. Along with the actual message, the addressing mode, the message type (normal or system), and the message priority must be specified.

The Message Control Register bMSGC is used to manage Control messages, with the results in the message status register bMSGS. Additional information about the transmit status is provided by the Transmit Status Register bXTS. The Xmit retry register bXRTY and the Xmit retry time register bXTIM control the number and retry interval for re-sending a transmission that failed.

13.2.1 bMSGC (Message Control Register)

0x85	bMSGC	Message Control Register (write only)	
Bit	**Name**	**Description**	**Default**
7	STX	Start transmission	0
6	RBE	Receive buffer enable	0
5	rsvd	Reserved; Write as 0	0
4	SAI	Start address initialization	0
3	RALC	Reset network configuration changed interrupt	0
2	RERR	Reset Error or Power-on after start-up interrupt	0
1	RMTX	Reset Message transmitted interrupt	0
0	RMRX	Reset Message received interrupt	0

Table 13-5: bMSGC (Message Control Register)

This register controls sending and receiving of control messages. The status of the operations triggered by bMSGC is reported in the Message Status Register bMSGS.

STX (Start Transmission)

When writing a 1 to STX, the message currently stored in the Xmit Control message buffer mXCMB is transmitted. The transceiver writes the result of the transmission operation into bit TXR in bMSGS and to register bXTS.

RBE (Receive Buffer Enable)

Incoming control messages are stored within the Receive Control Message Buffer mRCMB. If a message was received, the reception of further messages will be blocked until bit RBE (Receive Buffer Enable) is set to 1, releasing mRCMB. Therefore, once the contents of mRCMB is read or if the application doesn't care about the message, RBE must be set to 1 to allow further reception of control messages. Bit RBS in bMSGS provides the current status of mRCMB. When RBE is

written to 1, it will be reset to 0 automatically when the transceiver finished the releasing procedure.

SAI (Start Address Initialization)
Bit SAI is used to set a valid node address. First. the desired node address must be written to mXCMB (bytes XTAH, XTAL). Then SAI is set to 1. The transceiver will check whether the desired address is correct and unique within the network or not. If the address is valid and unique, it will be written to the Node Address register bNAH/bNAL. If the address is valid and the Node address register is written, bit TXR in bMSGS will be set to indicate success. SAI will be reset to 0 automatically.

RALC (Reset "Network configuration changed" interrupt)
Setting this bit, resets bit ALC in bMSGS and the interrupt pin */INT*. RALC will be reset to 0 automatically.

RERR (Reset "Error or Power-on after start-up" interrupt)
Setting this bit, resets bit ERR in bMSGS and the interrupt pin */INT*. RERR will be reset to 0 automatically.

RMTX (Reset "Message transmitted" interrupt)
Setting this bit, resets bit MTX in bMSGS and the interrupt pin */INT*. RMTX will be reset to 0 automatically.

RMRX (Reset "Message received" interrupt)
Setting this bit, resets bit MRX in bMSGS and the interrupt pin */INT*. RMRX will be reset to 0 automatically.

13.2.2 bMSGS (Message Status Register)

0x86	bMSGS	Message Status Register (read-only)	
Bit	**Name**	**Description**	**Default**
7	RBS	Receive buffer status	0
6	TXR	Transmission result	0
5..4	rsvd	Reserved. Write as 0	00
3	ALC	Network configuration changed	0
2	ERR	Error or Power-on initialization complete	1
1	MTX	Message transmitted	0
0	MRX	Message received	0

Table 13-6: bMSGS (Message Status Register)

The Message Status Register bMSGS reports the status of several operations in conjunction with the sending and receiving of Control Messages.

RBS (Receive Buffer Status)
A value of 0 indicates that the Receive Control Message Buffer mRCMB is ready to receive a new Control Message. When set to 1, mRCMB contains a message and no further messages will be received. mRCMB status is controlled via bit RBE in the Message Control Register bMSGC.

TXR (Transmission Result)
TXR flags the status of all transmissions. A value of 1 indicates a successful transmission. In the case of an error, TXR will be set to 0. The respective error code is stored in register bXTS.

ALC (Network Configuration Changed)
A 1 indicates that either bMPR (Maximum Position Register) or bMDR (Maximum Delay Register) has changed (or both).
If bit IALC in register bIE is set, then an interrupt will be generated when ALC changes to 1. ALC is reset by setting bit RALC in register bMSGC.

ERR (Error, or Power-on Initialization Complete)

This bit indicates either the completion of power-on initialization, or an error event. If bit IERR in bIE is set to 1, then interrupt will be generated when ERR changes to 1. Register bXSR determines which error events affect ERR. ERR is reset by setting bit RERR in register bMSGC.

MTX (Message Transmitted)

MTX indicates that a message has been transmitted. When MTX was set to 1, TXR indicates if the transmission was successful and register bXTS indicates the actual error condition. If bit IMTX in bIE is set to 1, then an interrupt will be generated when MTX changes to 1. MTX is reset by setting bit RMTX in register bMSGC.

MRX (Message Received)

MRX indicates that a control message has been received. If bit IMRX in bIE is set to 1, then an interrupt will be generated when MRX changes to 1. MRX is reset by setting bit RMRX in register bMSGC.

13.2.3 bXTS (Transmit Status Register)

0xD5	bXTS	Xmit Transfer Status Register (read-only)	
Bit	**Name**	**Description**	**Default**
7..0	XTS7..0	Transmission result	0x00

Table 13-7: bXTS (Transmit Status Register)

This register contains the result of a transmission including the error code in case of a transmission failure.

XTS7..0 (Transmission result)

0x00 - No response from the target address
0x10 - Message was sent successfully
0x11 - Message was received, but receiving node does not support XMIT message type
0x20 - Transmission failed because receiving node got a bad CRC
0x21 - Transmission failed because the receiving node's receive buffer was full.

13.2.4 bXRTY (Transmit Retry Register)

0xBF	bXRTY	Xmit Retry Register	
Bit	**Name**	**Description**	**Default**
7..0	XRTY7..0	Total transmission attempts	0x06

Table 13-8: bXRTY (Transmit Retry Register)

This register specifies how often the transceiver attempts a retransmission when the transmission fails. This is the **total** number of attempts, so the first attempt is included. The default value of ‘0x06’ will lead to a maximum of five additional attempts, if the original transmission fails. The minimum value is 0x01 and the maximum value 0xFF.

13.2.5 bXTIM (Transmit Retry Time Register)

0xBE	bXTIM	Xmit Retry Time Register	
Bit	**Name**	**Description**	**Default**
7..0	XTIM7..0	Time between transmission retries. Valid values are 0x03 to 0xFF.	0x0B

Table 13-9: bXTIM (Transmit Retry Time Register)

bXTIM specifies the delays between transmission retries, when an initial transmission fails. The value represents the number of time units between the retries, where the time unit depends on the network's frame rate, Fs. The formula for calculating the retry time is:

Retry time = <Time Unit> × XTIM

The time units are approximately:

421 µs at Fs = 38 kHz
363 µs at Fs = 44.1 kHz
333 µs at Fs = 48 kHz

The minimum value in bXTIM is 0x03 and the maximum value 0xFF. In a network running at at a frame rate of 44.1 kHz , the retry time for the default value 0x0B would be:

363 µs × 11 = 3.99 ms

Therefore, if transmission of a message fails, the OS8104 will wait approximately 4 ms before retrying the transmission.

All nodes in the network must have the same bXTIM value for proper operation of Broadcast Messages.

13.2.6 mRCMB (Receive Control Message Buffer)

0xA0	mRCMB	Receive Control Message Buffer	
Addr.	**Name**	**Description**	**Default**
0xA0	bRTYP	Type of received control message	0x00
0xA1	bRSAH	Source address high	0x00
0xA2	bRSAL	Source address low	0x00
0xA3	bRCD0	Received control data byte 0x0	0x00
...	...	Received control data bytes 0x1.. 0xF	0x00
0xB3	bRCD16	Received control data byte 0x10	0x00

Table 13-10: mRCMB (Receive Control Message Buffer)

After a successful reception, the Receive Control Message Buffer mRCMB contains a complete control message.

bRTYP (Received control message Type)
bRTYP indicates the address type of message received. The receive message types are described in more detail in Section 13.5.1. The definition of bRTYP is different from the definition of bXTYP in mXCMB.
0x00 – Logical Addressing (address in bNAH/bNAL)
0x01 – Physical Addressing (based on bNPR)
0x02 – Broadcast Addressing (using address 0x3C8)
0x03 – Groupcast Addressing (based on bGA)

bRSAH (Source Address High)
bRSAH contains the high byte of the message sender's logical address (the node who sent the message).

bRSAL (Source Address Low)
bRSAH contains the low byte of the message sender's logical address.

bRCDn (Received control data byte n)
The bytes bRCD0 up to bRCD16 contain the currently received message.

13.2.7 mXCMB (Transmit Control Message Buffer)

0xC0	mXCMB	Xmit Control Message Buffer	
Addr.	**Name**	**Description**	**Default**
0xC0	bXPRI	Priority for Xmit control message	0x01
0xC1	bXTYP	Type of Xmit control message	0x00
0xC2	bXTAH	Target address high	0x00
0xC3	bXTAL	Target address low	0x00
0xC4	bXCD0	Xmit control data byte 0x0	0x00 *
...	...	Xmit control data bytes 0x1.. 0xF	0x00
0xD4	bXCD16	Xmit control data byte 0x10	0x00

* After reset, bXCD0..bXCD2 contain the OS8104 version number (refer to Section 16.4 on page 140)

Table 13-11: mXCMB (Transmit Control Message Buffer)

The Xmit Control Message Buffer mXCMB contains the entire control message to be sent.

bXPRI (Priority for Xmit control message)

bXPRI specifies the priority at which the transmit arbitration handles the current message, when it is sent. The value 0x00 represents the lowest, 0x0F the highest priority. If several nodes try to send a message on the same priority level, the MOST network will grant access to the Control message channel based on a fair arbitration algorithm. If a node sends a message on a higher priority level than all the other nodes, the MOST network will directly grant access to the next time slot for control messages.

bXTYP (Type of Xmit control message)

bXTYP determines the type of the message to be sent. The meaning of bXTYP is different from the definition of bRTYP (in mRCMB) for received messages. bXTYP is described in more detail is Section 13.5.2.

0x00 – Normal message, using any of the various addressing options
0x01 – System message: Remote Read
0x02 – System message: Remote Write
0x03 – System message: Resource Allocation
0x04 – System message: Resource De-allocation
0x05 – System message: Remote GetSource

bXTAH/bXTAL (Target address high/Target address low)

bXTAH contains the high byte and bXTAL the low byte of the target address. The value determines the addressing mode used (see Section 13.1).

bXCDn (Xmit data byte n)

The bytes XCD0 up to XCD16 contain the data of the message that shall be sent. When using Broadcast or Groupcast addressing, only 16 data bytes are available for messages. Therefore, XCD16 is not available when using Broadcast or Groupcast addressing.

13.3 Control Message Reception

The maximum number of received data bytes per message is 17 (16 when using Broadcast or Groupcast addressing). Either polling or interrupts can be used to determine when a control message has been received. When using interrupts, bit IMRX in the Interrupt Enable register bIE enables received message interrupts. The left side of the Figure below illustrates the flow for an interrupt service routine for receiving control messages.

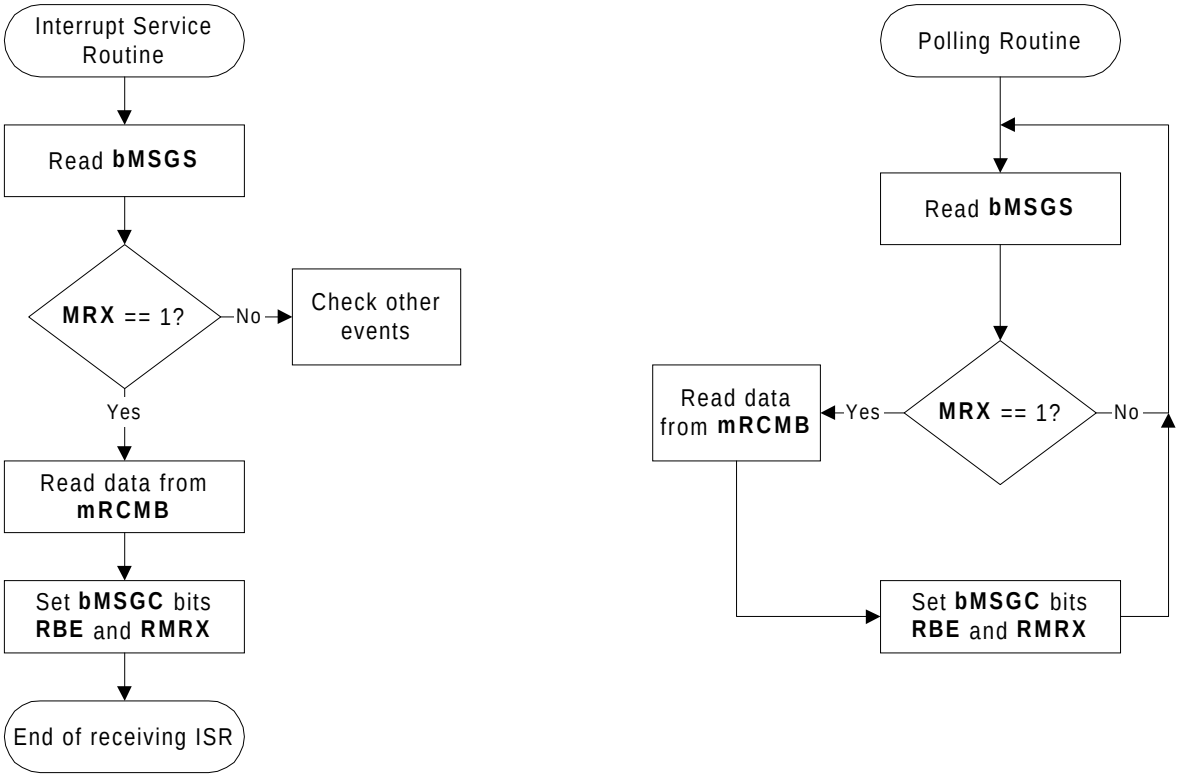


Figure 13-2: Control Message Reception Flow

If RBE is not set to 1 after reading the message, further control messages will be blocked from being received.

The mRCMB register is described in detail in Section 13.2.6 on page 113.

The right side of Figure 13-2 illustrates the use of polling to determine receive message status. Bit IMRX in bIE register must be set to 0 to disable interrupts.

13.4 Control Message Transmission

Similar to Control message reception, message transmission can be done through polling or by using interrupts. To use the interrupt method, bit IMTX in bIE register must be set to 1. Figure 13-3 shows the program flow for transmitting a control message.

To use the polling method, bit IMTX in bIE register should be set to 0 to disable interrupts, and the Message Status Register bMSGS must be polled in regular intervals.

Figure 13-4 illustrates the program flow for sending MOST Control Messages based on polling.

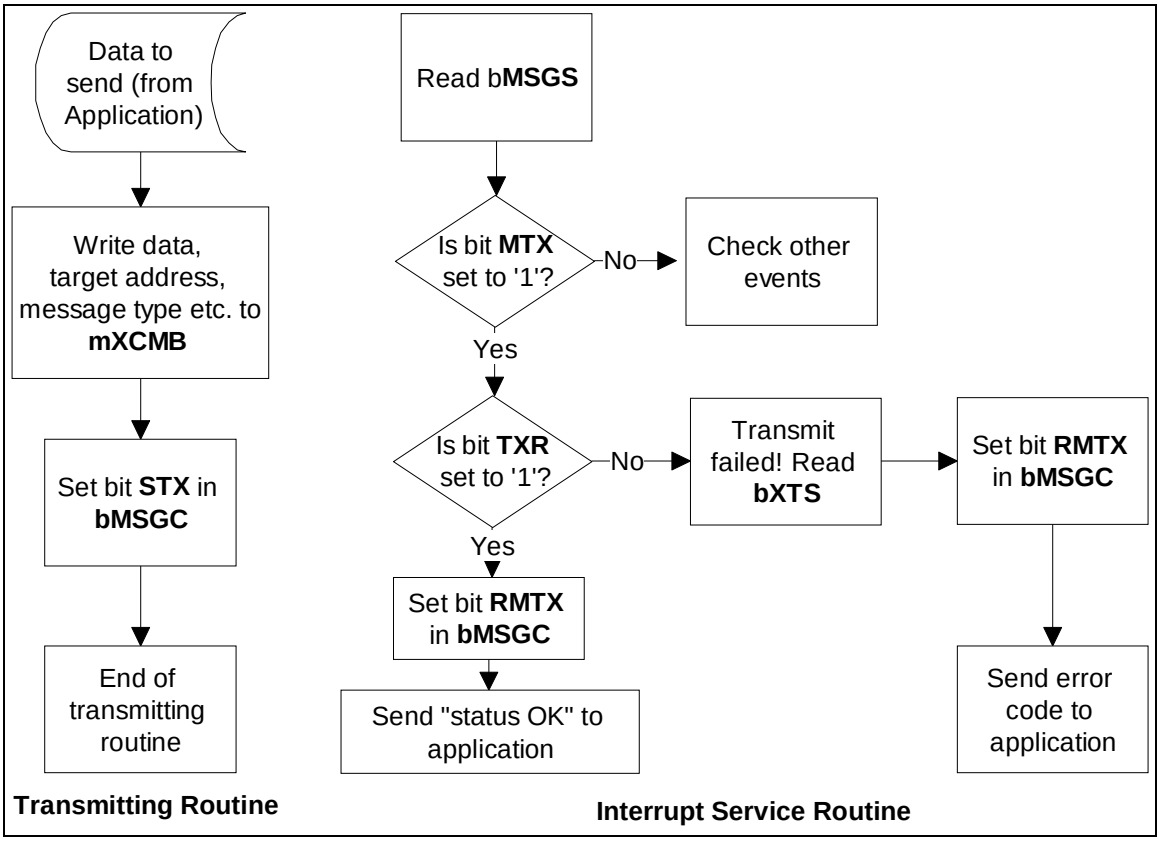


Figure 13-3: Sending MOST Control messages: Interrupt service routine

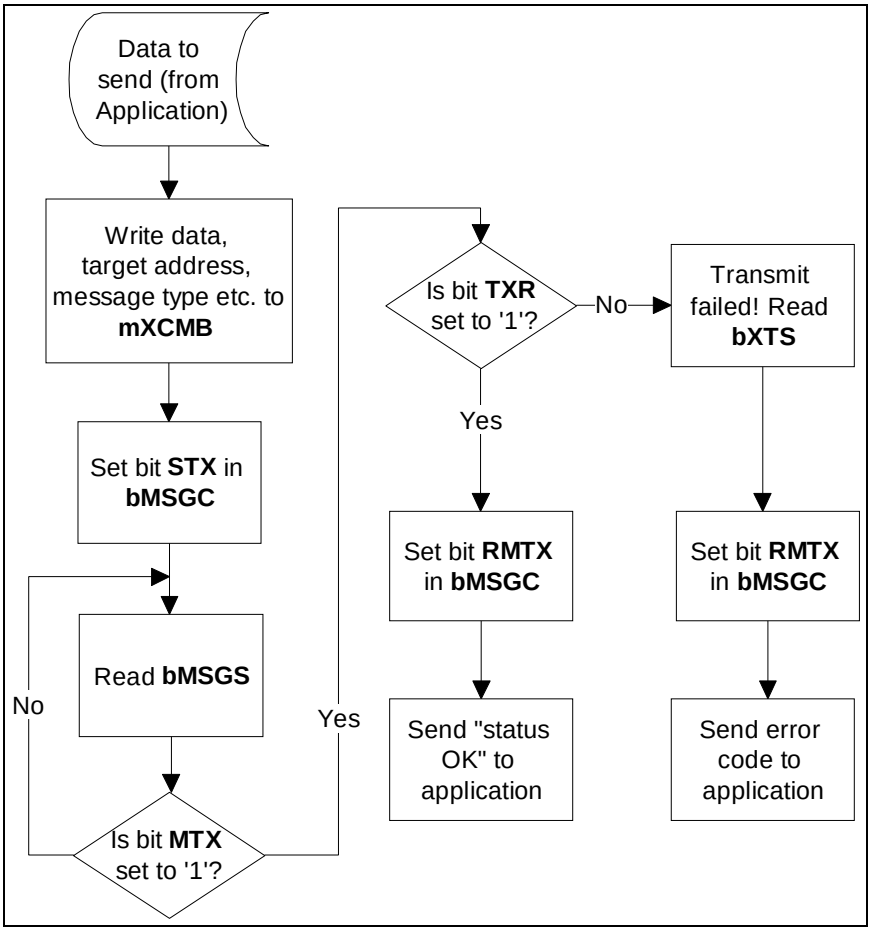


Figure 13-4: Sending MOST Control messages: Polling

13.5 Message Types (Encoding And Description)

13.5.1 Received Control Message Types

When receiving MOST Control Messages, register bRTYP in mRCMB determines the addressing mode used for the received message. (The transmit message type is very different than the receive message type).

bRTYP	Description
0x00	Message was normally addressed (logical node address matched)
0x01	Message was addressed to node position (Value in Node Position Register i.e. the physical address matched)
0x02	Message was broadcasted (address was 0x03C8)
0x03	Message was groupcasted (Value in Group Address Register matched)

Table 13-12: bRTYP (Received Control Message Types)

13.5.2 Transmit Message Types

For transmitting control messages, the following types of messages are available:

- Normal messages:
  - Single cast (Logical or Physical Addressing)
  - Groupcast
  - Broadcast

- System messages:
  - Resource allocate
  - Resource de-allocate
  - Remote read
  - Remote write:
    Single cast
    Groupcast
    Broadcast
  - Remote GetSource

The type of message is encoded in a single byte. The respective value must be written to bXTYP in mXCMB (Refer to Section 13.2.7 on page 114). (The Receive message type indicates the addressing mode.)

bXTYP	Description
	Standard message:
0x00	Message will be sent as normal message.
	System messages:
0x01	Message will be sent as "remote read" message.
0x02	Message will be sent as "remote write" message.
0x03	Message will be sent as "resource allocation" message.
0x04	Message will be sent as "resource de-allocation" message.
0x05	Message will be sent as "remote GetSource" message.

Table 13-13: bXTYP (Transmit Control Message Type)

13.5.2.1 Normal Messages (Code 0x00)

By using this message type code, data of any kind can be sent to any node within the net. The following table shows the contents of mXCMB for sending a normal message:

Address	Contents	Description
0xC0	0x01	Priority; Default 0x01
0xC1	0x00	Normal message
0xC2	XTAH	Target address high
0xC3	XTAL	Target address low
0xC4..0xD4	D0..D16	17 data bytes

Table 13-14: Sending a normal Control Message (Code 0x00)

Since the type of addressing is encoded in the actual address, the target address can be set for Logical or Physical addressing to a single node, Groupcast addressing to multiple nodes, or Broadcast addressing to all nodes.

13.5.2.2 Remote Read Message (Code 0x01)

This message type provides reading of registers on any other node, without influencing the target node's transmit and receive buffer or the respective bits. Therefore, the Remote Read is hidden from the application running on the target node. For the sending node, the transmission of a Remote Read message appears as a normal transmission; however, the result (the received data) will be stored in the transmit buffer mXCMB. Remote Read accesses are limited to the target node's memory page 0. The following table shows the contents of mXCMB after having sent a Remote Read message. The bytes read from the target device are located in mXCMB bytes D0 to D7 (0xC7..0xCE).

Address	Contents	Description
0xC0	0x01	Priority; Default 0x01
0xC1	0x01	Remote read
0xC2	XTAH	Target address high
0xC3	XTAL	Target address low
0xC4	rsvd	Reserved; Write as 0x00
0xC5	MAP	Address to read from
0xC6	rsvd	Reserved
0xC7..0xCE	D0..D7	8 data bytes read from target node (after the target node has answered the remote read request). Within D0 the contents of the memory location MAP pointed to is stored, within D1 the contents of MAP+1 etc.
0xCF..0xD4	rsvd	Reserved; Write as 0x00

Table 13-15: mXCMB after a Remote Read message

The result of sending the Remote Read message is indicated by the bits MTX and TXR in register bMSGS (refer to Section 13.2.2 on page 111). If a successful transmission is indicated, the data read from the remote node is available in the transmit buffer (D0..D7 bytes).

13.5.2.3 Remote Write Message (Code 0x02)

This message type allows modification of the remote node's memory locations on memory page 0. Like the remote read message, the remote write access is a special mode of operation, since the target node's transmit and receive buffers are NOT influenced in any way. Therefore, a remote write access is hidden from the application running on the target node. For the sending node, the transmission of a Remote Write message is a normal control message transmission. One to eight bytes in the remote node can be written starting at the memory address pointer (MAP). Table 13-16 shows the contents of mXCMB when sending a Remote Write message:

Address	Contents	Description
0xC0	0x01	Priority; Default 0x01
0xC1	0x02	Remote write
0xC2	XTAH	Target address high
0xC3	XTAL	Target address low
0xC4	rsvd	Reserved. Write as 0x00
0xC5	MAP	Address to write to
0xC6	LENGTH	Count of data bytes to be written (max. is 8)
0xC7..0xCE	D0..D7	Up to 8 data bytes to be written to target node. The contents of D0 will be written into address MAP, D1 into MAP+1 etc..
0xCF..0xD4	rsvd	Reserved

Table 13-16: mXCMB when sending a Remote Write message

The result of sending the Remote Write message is indicated by the bits MTX and TXR in register bMSGS (refer to Section 13.2.2 on page 111). If a successful transmission is indicated, the data was written successfully to the remote node.

13.5.2.4 Resource Allocate Message (Code 0x03)

Before routing any synchronous source data to the MOST network, the synchronous source channels (physical channels) should be allocated. Allocation of synchronous source data, managed by the timing-master node in the network, is needed to guarantee that different nodes wanting synchronous source channels don't use the same physical channels (causing collisions). When a node wants to place synchronous source data onto the network, it should send a "Resource Allocate" message to the timing-master node.

For the node requesting synchronous resources, the transmission of a Resource Allocate message appears as a normal transmission; however, the result (the answer from the timing-master node) is stored in the Xmit buffer. When an OS8104 is configured as the timing-master, it internally manages the incoming Resource Allocate messages and responds without added software support.

Remote Allocate messages can either be sent using Logical addressing or physical addressing. Using physical (node position) addressing is the most effective approach, since the timing-master in a MOST network always has node position 0x00. Therefore, the physical address is always 0x0400 and bXTAH should be set to 0x04, and bXTAL to 0x00.

One to eight physical channels can be allocated per Allocate request. When the timing-master responds, bit MTX and TXR in bMSGS will be set to 1, and Answer1 and Answer2 in mXCMB will indicate the response. If the Answer1 result is 0x01, then the allocation request was granted and P0..P7 contain the physical channel numbers to use. For the node requesting the channels, these physical channel numbers are the Routing Engine registers to load the address references of the source data into. This will connect the local nodes source data to the MOST network. The first channel that was granted (located in P0 at 0xC9, is also considered the Connection Label (CL) which is stored in the allocation table (mCRA) and represents the entire set of channels allocated with this request. This CL is used to de-allocate this set of channels. Resource Allocation and the mCRA is discussed in more detail in Section 14 on page 125. Table 13-17 shows the contents of mXCMB for sending a Resource Allocate message:

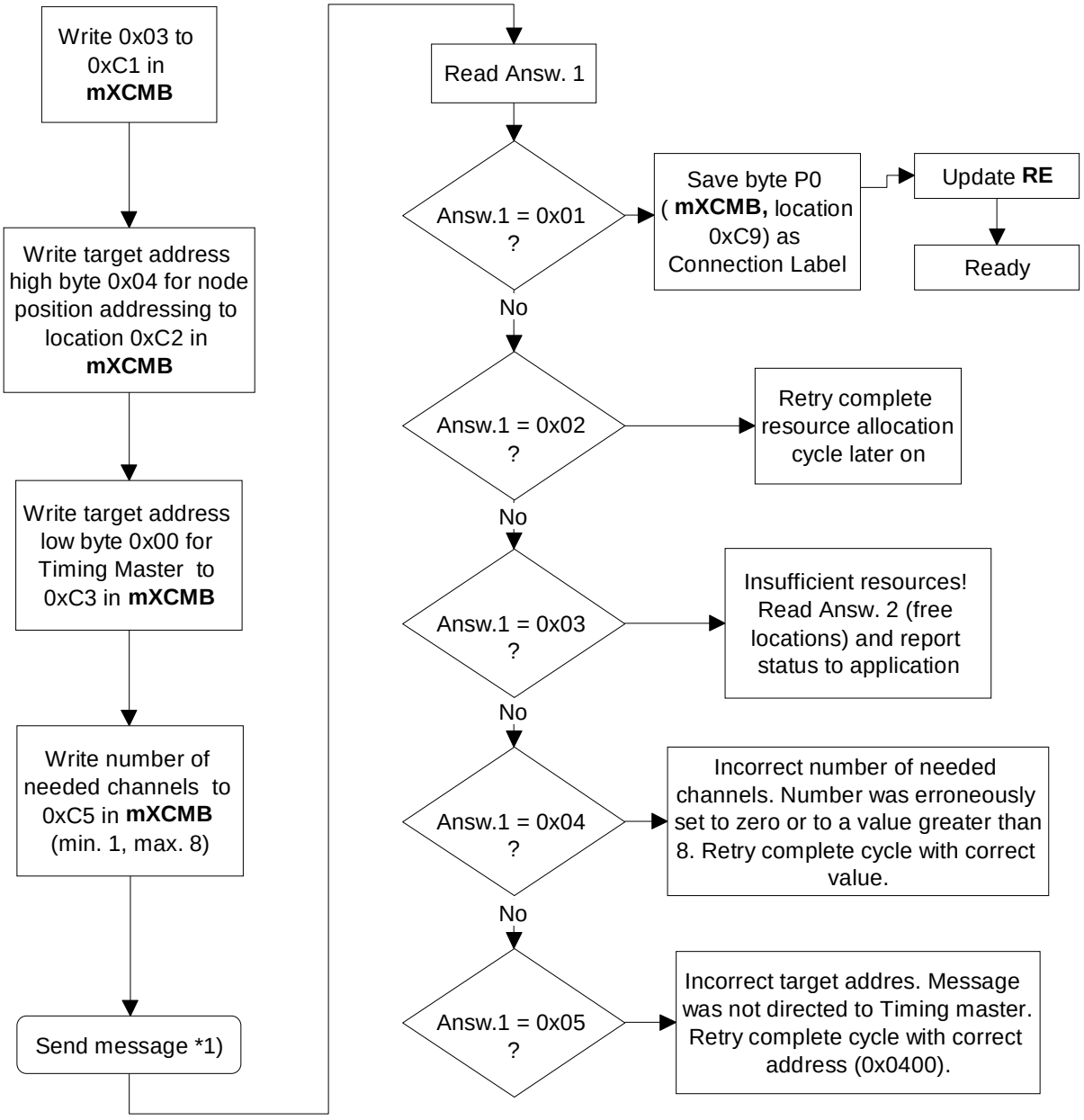
Address	Contents	Transmit Control Message Buffer mXCMB
0xC0	0x01	Priority; Default 0x01
0xC1	0x03	Resource Allocate
0xC2	0x04	Target address high (for physical addressing = 0x04)
0xC3	0x00	Target address low (for physical addressing = 0x00)
0xC4	rsvd	Reserved; Write as 0x00
0xC5	Request	Number of channels to be allocated (maximum value is 8)
0xC6	rsvd	Reserved. Write as 0x00
0xC7	Answer1	Allocation status from timing master (Note 1)
0xC8	Answer2	Allocation status from timing master (Note 1)
0xC9..0xD0	P0..P7	Routing Engine Register locations. (Note 2) 0xFF means "not usable. P0 is Connection Label (CL) which is used when de-allocating this channel.
0xD1..0xD4	rsvd	Reserved
Note 1: Refer to Table 13-18 for Answer 1 and Answer 2 decoding. Note 2: The value returned by the master refers to the physical channel address within the MOST frame. These address are the Routing Engine register locations used to transmit data across the network. The source data address references should be placed in these RE register locations.		

Table 13-17: mXCMB when Sending a Resource Allocate Message

Answer1	Comment	Answer2	Comment
0x01	ALLOC_GRANT; There are enough free channels	AL_FREE	Total number of free locations at that time (current request included)
0x02	ALLOC_BUSY; Master is still busy working at allocation requests.	AL_FREE	Total number of free locations at that time
0x03	ALLOC_DENY; Request was denied. Not enough free channels to fill request.	AL_FREE	Total number of free locations at that time
0x04	ALLOC_WRONG; The REQUEST value was set to 0 or to a value greater than 8.	AL_FREE	Total number of free locations at that time
0x05	WRONG_TARGET; Request was directed to a slave node instead the timing master.	0x00	Constantly set to zero

Table 13-18: Answers from a Resource Allocate message

The result of the current resource allocate request is returned by the timing master node and written into the bytes Answer1 and Answer2 of mXCMB. The following flow chart illustrates how to sending allocate requests and how to respond to the answers.



*1): The sending of a message is described in Section 13.4 on page 116.

Figure 13-5: Resource Allocation Flow

13.5.2.5 Resource De-Allocate message (Code 0x04)

If network synchronous source data is no longer needed by the node, the resources should be de-allocated, so they are available for other applications. De-allocating allocated resources is a simple task using the "Resource De-Allocate" message.

The Connection Label, provided by the timing-master node when the resources were allocated, is sent in the de-allocate message and tells the timing-master which set of physical channels to de-allocate. The Connection Label (CL) is the physical channel ID of the first channel and is stored in the current allocation table (mCRA) for all channels assigned to this ID. The mCRA is described in more detail in 14.

One special CL (0x7F) forces the timing-master to de-allocate all resources that have previously been allocation and is labeled "De-allocate All". After power-up or reset, the application of the timing-master node must send a "De-allocate All" message to physical address 0x0400 (himself), or to its own logical address to initialize the mCRA table.

The following table shows the contents of mXCMB for sending a Resource De-Allocate message:

Address	Contents	Transmit Control Message Buffer mXCMB
0xC0	0x01	Priority; Default 0x01
0xC1	0x04	Resource De-Allocate
0xC2	0x04	Target address (for physical addressing = 0x04)
0xC3	0x00	Target address (for physical addressing = 0x00)
0xC4	rsvd	Reserved; Write as 0x00
0xC5	CL	Connection Label assigned during allocation process (Note 1)
0xC6	rsvd	Reserved; Write as 0x00
0xC7	Answer1	De-Allocation status from Master (Note 2)
0xC8	0x00	Constant 0x00
0xC9..0xD4	rsvd	Reserved
Note 1: See the "Resource Allocate message" Note 2: See Table 13-20 for list of answers from the timing-master node.		

Table 13-19: mXCMB for sending a Resource De-Allocate message

Answer1	Comment
0x01	DEALLOC_GRANT; De-allocation successful.
0x02	DEALLOC_BUSY; Master is still busy in working at allocation/de-allocation requests.
0x04	ALLOC_WRONG; The connection label had a value greater than 0x7F.
0x05	WRONG_TARGET; Request was directed to a slave node instead to the master.

Table 13-20: Answers on Resource De-Allocate messages

After every change of bSBC, the allocation mechanism must be initialized by sending the "de-allocate all" message 0x7F to the timing-master node.

13.5.2.6 Remote GetSource Message (Code 0x05)

This message finds the address (logical, physical, and group) of the node that is placing source data onto the network, using a particular Connection Label (CL). Similar to other system messages, this message does not interfere with the target node's transmit or receive buffers, and is handled in the background, without interfering with the normal Control message traffic.

The Remote GetSource message is sent as a broadcast message (address 0x03C8) to all nodes. This system message is unique since it doesn't block the Control message channel even though it uses broadcast addressing. The Connection Label associated with the source channels is placed in the mXCMB buffer at location 0xC5. The node associated with that Connection Label will respond with its addresses.

If bits MTX and TXR are set to 1 in the bMSGS register, then the addresses sought are stored back in the Xmit buffer (D0..D7). Transmit buffer register 0xCA contains the node position (part of the physical address) of the node using the specified Connection Label. Register 0xCC contains the group address, and registers 0xCD and 0xCE contain the node's logical address.

If TXR is 0, no node responded indicating that no current node in the network is associated with that Connection Label.

Address	Contents	Description
0xC0	0x01	Priority; Default 0x01
0xC1	0x05	Remote GetSource message code
0xC2	0x03	Broadcast address high. Set to 0x03
0xC3	0xC8	Broadcast address low. Set to 0xC8
0xC4	rsvd	Reserved; Write as 0x00
0xC5	CL	Connection Label
0xC6..0xC9	rsvd	Reserved; Write as 0x00
0xCA	NPR	Returned Node Position of the node that routes stream data to the channels specified by CL
0xCB	rsvd	Reserved; Write as 0x00
0xCC	GA	Returned Group Address of the node that routes stream data to the channels specified by CL
0xCD	NAH	Returned high byte of logical address of the node that routes stream data to the channels specified by CL
0xCE	NAL	Returned low byte of logical address of the node that routes stream data to the channels specified by CL
0xCF..0xD4	rsvd	Reserved; Write as 0x00

Table 13-21: mXCMB when Sending a Remote GetSource Message

14 Resource Administration

The MOST network supports a maximum of 60 bytes per frame for transporting source data (see Section 6.1 on page 33). These bytes can be divided between synchronous source data and asynchronous packet data. Register bSBC (see Section 6.2.5 on page 37) controls the division between the two, and contains the number of quadlets (four bytes) reserved for synchronous source data. Once bSBC is initialized, the timing-master knows how many synchronous source physical channels are available, and where to start asynchronous channel arbitration.

To manage the synchronous source data on a network-wide basis, the master node maintains a Channel Resource Allocation Table (mCRA) to keep track of the status of each synchronous source physical channel. This table is distributed to all nodes on the network at regular intervals (every 1024 frames). Having one node (the timing-master) manage the allocation of synchronous source data, keeps other network nodes from placing source data on the same physical source channels causing collisions.

14.1 mCRA (Channel Resource Allocation Table)

The Channel Resource Allocation table (mCRA) contains the current allocation status for every synchronous data byte in the MOST frame. In each node, mCRA is located at memory locations 0x380 up to 0x3BB. Since the number of synchronous data bytes per frame is variable, the length of mCRA is variable too. The maximum memory address for mCRA is calculated as follows:

last mCRA address = 0x380 + (bSBC $\times$ 4) - 1

The example below shows the structure of mCRA when 14 quadlets of the available 15 quadlets are assigned to synchronous source data. The variables CRAnn in the table refer to the synchronous source data physical byte position in the MOST frame (CRA0F stands for the synchronous source physical channel number 0x0F in the MOST frame). The maximum address is for this example is:

0x380 + (14 $\times$ 4) - 1 = 0x3B7

Therefore, the contents of addresses 0x3B8 to 0x3BB must be ignored since they do not represent valid synchronous source bytes.

mCRA	+0	+1	+2	+3	+4	+5	+6	+7
0x380	CRA00	CRA01	CRA02	CRA03	CRA04	CRA05	CRA06	CRA07
0x388	CRA08	CRA09	CRA0A	CRA0B	CRA0C	CRA0D	CRA0E	CRA0F
0x390	CRA10	CRA11	CRA12	CRA13	CRA14	CRA15	CRA16	CRA17
0x398	CRA18	CRA19	CRA1A	CRA1B	CRA1C	CRA1D	CRA1E	CRA1F
0x3A0	CRA20	CRA21	CRA22	CRA23	CRA24	CRA25	CRA26	CRA27
0x3A8	CRA28	CRA29	CRA2A	CRA2B	CRA2C	CRA2D	CRA2E	CRA2F
0x3B0	CRA30	CRA31	CRA32	CRA33	CRA34	CRA35	CRA36	CRA37
0x3B8	CRA38	CRA39	CRA3A	CRA3B				

Table 14-1: mCRA built for 56 bytes of synchronous source data.

After reset, all values in the mCRA table are set to 0x70, indicating that the byte is not allocated and not in use.

14.1.1 mCRA in Slave Nodes

The allocation status of a byte is encoded in the value stored to its associated location within mCRA. The following values may occur in any node that is not timing master:

Value (note 1)	Description
0x70	Corresponding byte in frame is not allocated
0x00 .. 0x3B	Corresponding byte is allocated. The value specifies the Connection Label.

Note 1. Assumes MSB masked off before interpreting value.

Table 14-2: Valid mCRA Values in a Slave Node

The most significant bit of the values in mCRA must be ignored (masked off) in all slave nodes (all nodes except the timing master). The Connection Label is the first physical channel number returned from an allocation request and is stored at every physical channel location in the mCRA associated with that request.

14.1.2 mCRA in the Timing-Master

In a timing-master node, the most significant bit of the values in mCRA has significance. The MSB indicates whether a byte is actually in use or not. Therefore, the lower seven bits indicate whether a byte is allocated or not and the MSB indicates whether its currently in use or not.

Value	MSB	bit 6..bit 0	Description
0x70	0	0x70	Corresponding byte is not allocated and not in use
0xF0	1	0x70	Corresponding byte in frame is not allocated, but is being used by a node within the network. This is an error, since the byte was never allocated before being used by a node.
0x80	1	0x00	Corresponding byte is allocated to Connection Label 0x00 and is in use.
0x00	0	0x00	Corresponding byte is allocated to Connection Label 0x00 but is not in use. The node which allocated this resource is not using it at this time.

Table 14-3: Valid mCRA Values in Timing-Master Node

14.2 Allocating Network Resources

For reducing the complexity of application software, the timing-master node manages the synchronous source data resources automatically. When the timing-master is initializing the node and setting the SBC value, it must send a "De-Allocate All" Control message to initialize the mCRA.

Each application that needs synchronous source channels, must send a *Remote Allocate* control message to the timing-master. The timing-master checks whether there are enough channels unused to fill the request, and if so, returns the physical channel numbers to use. If there are not enough free physical channels to fill the request, the timing-master sends an error message to the requesting node indicating so. Each physical channel number corresponds to an 8-bit channel in the frame which is now reserved for the requesting-node's application. These physical channel numbers are also the Routing Engine register locations that the application fills with the address references of the data to send onto the network. When the application places address references into the lower half of the Routing Engine registers, the OS8104 moves the source data associated with that address reference onto the MOST network at the position (physical source channel number) indicated by the particular RE register number. Section 12 explains the routing of synchronous source data in detail.

The first physical channel number returned by the timing-master is also used for the "Connection Label" (CL), and is used to ID all the channels allocated on this particular request. Therefore the mCRA only stores Connection Labels for allocated channels (each physical channel location in the mCRA contains the Connection Label associated with that allocation request).

For example, if an application requests 8 channels. The timing-master could grant the request and might return the physical channel numbers 0x00 through 0x07. 0x00 is the CL for this request, and must be stored by the application and used to de-allocate these channel when no longer in use. The CL 0x00 would be stored in the mCRA at the first eight locations (those locations associated with physical channel numbers 0x00 through 0x07).

When the application is finished using these channels, it must de-allocate the channels by sending the CL to the timing-master, and then remove the address references from the corresponding RE register locations. The RE registers should be filled with the corresponding received network data to allow those synchronous source channels to pass through the node unaltered.

After numerous allocate/de-allocate operations, the mCRA may be fragmented since the physical channels associated with a Connection Label need not be continuous. Table 14-4 illustrates an example of an mCRA table, in a slave node, when 24 bytes are reserved for synchronous source data (6 quadlets). These values are after the MSB is masked off.

Byte	+0	+1	+2	+3	+4	+5	+6	+7
0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0x08	0x08	0x08	0x08	0x0B	0x0B	0x0D	0x0B	0x0D
0x10	0x0D	0x0D	0x0D	0x0D	0x0D	0x0D	0x70	0x70

Table 14-4: mCRA Example

The physical source channels of the MOST frame are assigned as follows:

Connection Label (CL)	Physical Source Channels in MOST frame
0x00	0x00 through 0x07 (8 physical channels)
0x08	0x08 through 0x0A (3 physical channels)
0x0B	0x0B, 0x0C, and 0x0E (3 physical channels)
0x0D	0x0D, 0x0F, and 0x10 through 0x15 (8 physical channels)
0x70	0x16 and 0x17 are unallocated (unused) channels

Table 14-5: Connection Label Example

The maximum number of allocable channels per request is eight. The number of requests per node is not limited. Therefore a node could allocate 8 bytes at once, or could send eight requests for a single channel (although the application would now have to track 8 Connection Labels). Since the maximum number of allocable channels per node is only limited by the number of available source data bytes, a single node could allocate all 60 bytes.

14.3 De-Allocating Network Resources

If allocated synchronous source channels are no longer needed by a node, they should be de-allocated to allow other nodes to allocate them. This is done by sending a *Resource De-Allocate* Message to the timing-master of the network. The *Resource De-Allocate* format is described in Section 13.5.2.5.

For de-allocating resources, the Connection Label, given when the resources were allocated, must be sent to the timing-master in the de-allocation request. Using the example from the previous section, the following tables show a mCRA before the de-allocation request.

mCRA	+0	+1	+2	+3	+4	+5	+6	+7
0x380	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0x388	0x08	0x08	0x08	0x0B	0x0B	0x0D	0x0B	0x0D
0x390	0x0D	0x0D	0x0D	0x0D	0x0D	0x0D	0x70	0x70

Table 14-6: mCRA Example before Resource De-allocation

The node sends a de-allocation request to the timing-master using the Connection Label 0x0B. Table 14-7 illustrates the mCRA after the resources have been de-allocated. Their status (not allocated) is indicated by the value 0x70.

mCRA	+0	+1	+2	+3	+4	+5	+6	+7
0x380	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
0x388	0x08	0x08	0x08	0x70	0x70	0x0D	0x70	0x0D
0x390	0x0D	0x0D	0x0D	0x0D	0x0D	0x0D	0x70	0x70

Table 14-7: mCRA Example after Resource De-allocation

15 Packet Data Transfer

The Packet Data Transfer service utilizes the portion of the MOST network reserved for the asynchronous channel, and is useful for applications which need to transfer data in bursts, instead of a continuous data stream (Internet data, GPS map data, email, etc). In all Source Port modes, except Parallel-Combined, the maximum packet is size contains 48 data bytes (When the Source-Ports are in Parallel-Combined mode, the maximum packet data size is 1014). The packet is protected by a trailing CRC (Cyclic Redundancy Check).

15.1 Packet Transfer Registers

The following registers are associated with asynchronous Packet:

- Alternative Packet Address High Register (bAPAH)
- Alternative Packet Address High Register (bAPAL)
- Packet Length For Data Transfer Register (bPLDT)
- Packet Priority Register (bPPI)
- Packet Control Register (bPCTC)
- Packet Start Tx Register (bPSTX)
- Packet Status Register (bPCTS)
- Asynchronous Receive Packet Buffer (mARP)
- Asynchronous Xmit Packet Buffer (mAXP)

In addition, the Logical node address in registers bNAH and bNAL are sent as the Source address of a packet.

15.1.1 Address Registers

When sending a packet, the target address can either be the target node's logical address (specified in the bNAH/bNAL registers), or it can be the address stored in the target node's alternate packet address registers. Sending packets to the logical address uses the same addressing as used for the Control messages. If there is a need to have separate addresses for Control and Packet messages, the alternate packet address registers (bAPAH and bAPAL) can be used.

15.1.1.1 bAPAH (Alternate Packet Address High Register)

0xE8	bAPAH	Packet Address High Register	
Bit	**Name**	**Description**	**Default**
7..0	APAH7..0	Alternative Packet Address High. This value cannot be the same as bNAH.	0x0F

Table 15-1: Alternative Packet Address High Register (bAPAH)

bAPAH keeps the higher address part (bits 15 through 8) of the alternative address for Packet Data Transfer. The default address for alternate Packet Data Transfer after reset is 0x0FFF. bAPAH cannot be the same value as bNAH. When a node receives a packet, it checks the target address high byte against its bNAH value. If they don't match, then the alternate packet address registers are checked. bAPAH and bAPAL can be as an asynchronous packet group cast address to allow multiple nodes to receive the same message. However, using this register as a form of groupcast address, doesn't have the same protection and acknowledges that are associated with the Control message groupcast addressing.

15.1.1.2 bAPAL (Alternate Packet Address Low Register)

0xE9	bAPAL	Packet Address Low Register	
Bit	**Name**	**Description**	**Default**
7..0	APAL7..0	Alternative Packet Address Low	0xFF

Table 15-2: Alternative Packet Address Low Register (bAPAL)

bAPAL keeps the lower address portion (bits 7 through 0) of the alternative address for Packet Data Transfer.

15.1.2 bPLDT (Packet Transmit Length Register)

0xEC	bPLDT	Transmit Packet Length Register	
Bit	**Name**	**Description**	**Default**
7..0	PLDT7..0	Packet length for data transfer in quadlets	0x00

Table 15-3: Packet Length Register (bPLDT)

The value in this register specifies the number of data bytes (in quadlets) which shall be sent in Packet Data Transfer mode, and includes the data bytes along with the two source address bytes. Therefore, the length value is calculated as follows:

bPLDT = roundup ((number_of_data_bytes +2) / 4)

Valid values for bPLDT are 0x01 up to 0x0D. If the length value is rounded up (fractional), then it is recommended that the extra filler bytes at the end of the packet be set to 0x00. For example, sending three data bytes will take two quadlets (due to the two source address bytes). Three filler bytes exist at the end of the packet, which should be filled with zeros. The following table shows the bPLDT value for various data byte lengths.

Data bytes	Quadlets (bPLDT value)
1	0x1
2	0x1
3	0x2
4	0x2
5	0x2
6	0x2
7	0x3
...	...
48	0xD

Table 15-4: bPLDT Length Values

15.1.3 bPPI (Packet Priority Register)

0xF2	bPPI	Packet Priority Register	
Bit	**Name**	**Description**	**Default**
7..0	PPI7..0	Packet Priority. Valid values are 0x01 (highest) to 0x07 (lowest)	0x01

Table 15-5: Packet Priority Register (bPPI)

The value in bPPI determines the priority by which a packet shall be handled. Valid values are 0x01 up to 0x07, where 0x01 stands for the highest priority level.

15.1.4 bPCTC (Packet Control Register)

0xE2	bPCTC	Packet Control Register	
Bit	**Name**	**Description**	**Default**
7..5	rsvd	Reserved; Write as 0	0
4	RAF	Reset "Packet rejected status (mARP full)" bit AF in bPCTS	0
3	RAC	Reset "Packet rejected status (CRC failed)" bit AC in bPCTS	0
2	rsvd	Reserved. Write as 0	0
1	RATX	Clear packet transmitted interrupt	0
0	RARX	Unlock the Asynchronous Receive Packet Buffer mARP	0

Table 15-6: Packet Control Register (bPCTC)

RAF (Reset "Packet rejected status (mARP full)" bit AF in bPCTS)
When writing a 1 to RAF, bit AF in bPCTS is set to 0.

RAC (Reset "Packet rejected status (CRC failed)" bit AC in bPCTS)
When writing a 1 to RAC, the bit AC in bPCTS is set to 0.

RATX (Clear Packet Transmitted interrupt)
When writing a 1 to RATX, bit ATX in register bPCTS will be reset to 0. Additionally the interrupt will be set to inactive, and pin /AINT will change to 1.

RARX (Unlock the Asynchronous Receive Packet Buffer mARP, and clear packet received interrupt)
Writing a 1 to RARX resets bit ARX in register bPCTS and unlocks the Asynchronous Receive Packet Buffer mARP. When ARX is set to 1, mARP contains a packet and is locked until RARX unlocks it. While locked, mARP will not receive any other packets.

Buffer mARP should be read before unlocking it, otherwise the data could be over-written by a new packet.

15.1.5 bPSTX (Packet Start Tx Register)

0xEA	bPSTX	Packet Start Tx Register	
Bit	**Name**	**Description**	**Default**
7	ASTX	Start packet transmission	0
6..0	rsvd	Reserved. Write as 0	0

Table 15-7: Packet Start Tx Register (bPSTX)

ASTX (Start packet transmission)
When set to 1, the OS8104 starts the transmission of the current packet. ASTX is cleared automatically when the transmission is finished.

15.1.6 bPCTS (Packet Status Register)

0xE3	bPCTS	Packet Status Register	
Bit	**Name**	**Description**	**Default**
7..5	rsvd	Reserved	0
4	AF	Packet rejected - mARP full	0
3	AC	Packet rejected - CRC failed	0
2	rsvd	Reserved	0
1	ATX	Packet transmitted	0
0	ARX	Packet received	0

Table 15-8: Packet Status Register (bPCTS)

AF (Packet rejected, mARP full)

AF set to 1 indicates that the last reception failed due to the packet receive buffer mARP being full (locked). mARP is unlocked by setting RARX to 1 after having read its contents. AF is cleared by setting bit RAF in bPCTC to 1.

AC (Packet rejected, CRC failed)

AC set to 1 indicates that the last reception failed due to a bad CRC value. AC is cleared by setting bit RAC in bPCTC to 1.

ATX (Packet transmitted event)

ATX set to 1 indicates that a packet transmission is completed. ATX will be set after the last byte of a packet is processed. The /AINT pin will go low shortly after the ATX bit is set. ATX is cleared by setting bit RATX in register bPCTC to 1.

ARX (Packet received event)

When set to 1, ARX indicates that a packet has been received and that mARP is locked. Shortly after ARX is set, the /AINT pin will go low. ARP can be cleared by setting bit RARX in register bPCTC to 1.

As long as ARX is set, no reception of further packets is possible. Clearing ARX unlocks mARP. The contents of mARP can then be overwritten by the next packet.

15.1.7 mARP (Asynchronous Receive Packet Buffer)

The Asynchronous Receive Packet Buffer contains the last packet which was successfully received.

0x180	mARP	Asynchronous Receive Packet Buffer	
Byte	**Name**	**Description**	**Default**
0x00	bARTH	Received Target address high	0x00
0x01	bARTL	Received Target address low	0x00
0x02	bASAH	Source address high	0x00
0x03	bASAL	Source address low	0x00
0x04	bARD0	Asynchronous receive data byte 0	0x00
0x05 ... 0x32	bARD1 ... bARD46	Asynchronous receive data byte 1 to 46	0x00
0x33	bARD47	Asynchronous receive data byte 47	0x00

Table 15-9: Asynchronous Receive Packet Buffer (mARP)

bARTH/bARTL (Received Target Address high/Received Target Address low)
These bytes contain the target address to which the received packet was sent. Since two addresses are possible (bNAH/bNAL and bAPAH/bAPAL) it is easy to distinguish which address was used.

bASAH/bASAL (Source Address high /Source Address low)
These bytes contain the Logical address of the sending node, the origin of the packet data in the MOST network.

bARD0..47 (Asynchronous Receive Data Bytes 0 up to 47)
These bytes contain the actual packet data which was received.

15.1.8 mAXP (Asynchronous Transmit Packet Buffer)

0x1C0	mAXP	Asynchronous Xmit Packet Buffer	
Byte	**Name**	**Description**	**Default**
0x00	bATAH	Target address high	0x0F
0x01	bATAL	Target address low	0xFF
0x02	bAXD0	Asynchronous Xmit data byte 0	0x00
0x03 ... 0x30	bAXD1 ... bAXD46	Asynchronous Xmit data byte 1 to 46	0x00
0x31	bAXD47	Asynchronous Xmit data byte 47	0x00

Table 15-10: Asynchronous Xmit Packet Buffer (mAXP)

bATAH/bATAL (Target Address high/Target Address Low)
These bytes contain the target address of the node to send the packet to. This should be the Logical node address or the Alternate Packet address.

bAXD0..47 (Asynchronous Transmit Data Bytes 0 up to 47)
These bytes contain the source data which shall be sent.

If the number of bytes to be sent is not divisible by 4, it is recommended to add filler bytes (generally 0x00).

15.2 Asynchronous Interrupt Pin /AINT

The asynchronous interrupt pin */AINT* indicates either the reception or transmission of packet data. For transmitted packets, /AINT will go low when the packet is completely transferred, the transmit status is available, and the Packet transmit buffer (mAXP) is available. For received packets, /AINT will go low when a valid packet is received, with the entire packet available in the Packet receive buffer mARP. A valid received packet is one where the logical address (bNAH/bNAL) or the alternate packet address (bAPAH/bAPAL) is correct, **AND** the message has a valid CRC.

15.3 Packet Data Handling

The chip will transmit Packet Data using the portion of the MOST frame reserved for asynchronous packet data transfer. The amount of source data reserved for asynchronous packet data is defined in the bSBC register. If there is not sufficient space for sending a packet within a single frame, data will be segmented automatically and sent in smaller portions until the transmission is finished.

15.3.1 Preparing Packet Data For Transmission

When sending data as packet data, a maximum of 48 bytes can be sent per packet. The packet size can be increased by running the Source Ports (SP) in Parallel-Combined mode (see Section 8.3 on page 73).

As preparatory work, the destination address and the priority level must be written to the respective registers. The length of data (in quadlets) must be calculated according to Section 15.1.2 on page 130 and placed in register bPLDT. The number of user data bytes per packet must be reported from the sending node to the receiving node. This is the task of the controlling application software and must be done before starting transmission (when receiving a packet, the actual data length in the buffer is not provided).

After length was calculated and written to bPLDT, user data can be written to the Asynchronous Xmit Packet Buffer mAXP.

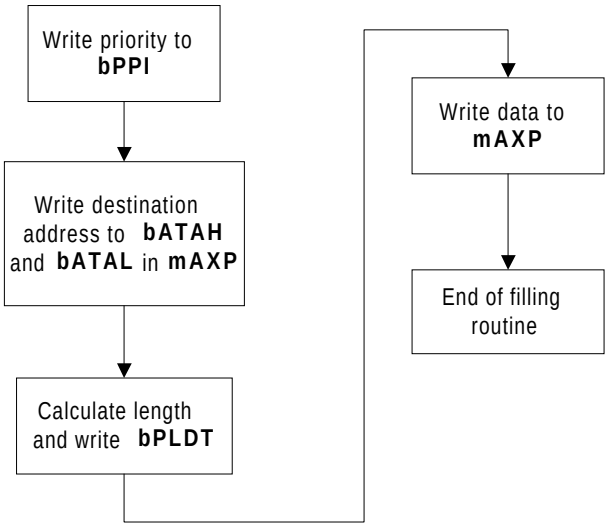


Figure 15-1: Preparing packet data for being sent

The small routine shown in Figure 15-1 is used when describing the handling of packet data transfer, noted as "†" in the Figures to follow.

15.3.2 Interrupt-Based Packet Data Handling

As mentioned in Section 15.2 on page 133, an interrupt will occur on /AINT when a packet is finished being transmitted or a valid packet has been received (correct target address and CRC). After having entered the interrupt service routine, the Packet Status Register must be read. If needed, the information about eventual errors (Bits AF and AC) can be stored for later use in the main loop of the application. If bit ARX is set, a packet has been received, and the received packet data is in the Asynchronous Receive Packet Buffer mARP. After having read mARP, the buffer must be released (unlocked) to being able to receive the next packet. This is done by setting bit RARX to 1 in the packet control register mPCTC.

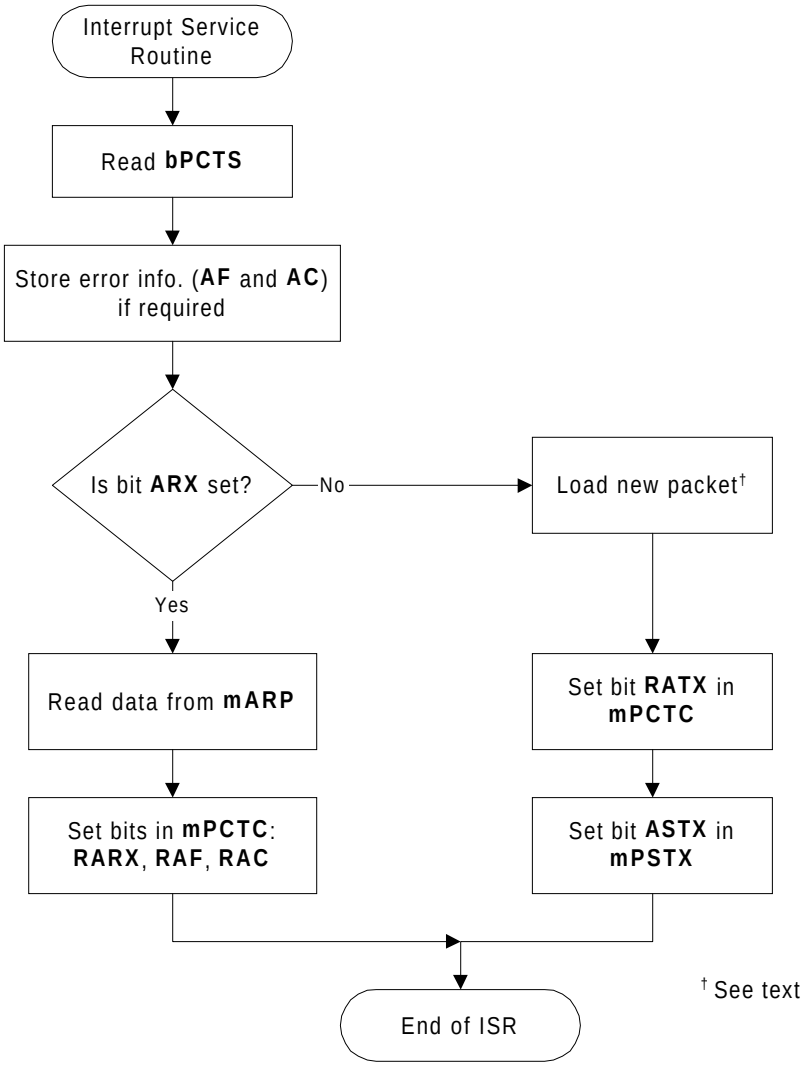


Figure 15-2: Handling Packet Data transfer via interrupt

If the error bits RAF and RAC have been set, they should also be cleared to prepare them for the next message. This ends the handling of a packet received event.

If the interrupt was caused by a packet transmit event, a new packet can be prepared for transmission. Assuming more packets need to be sent, the new packet data and all the header information should be loaded into the appropriate registers, then bit ASTX in register bPSTX must be set to 1 (which starts a new transmission). This ends the handling of a packet transmitted event.

Once the asynchronous channel is arbitrated for and the packet is sent, /AINT will go low. The ASTX bit in bPSTX will be reset automatically once transmission is finished.

15.3.3 Packet Data Handling Based On Polling

The routine below must be called periodically by the main routine of the application. First, the Packet Control register bPCTC must be read and compared with 0x00. If the contents of this register is equal to zero, no operation (e.g. reset bit ARX) is pending, OS8104 is ready for further operations. If either one or both of the bits ATX and ARX are set, then packet data handling is required. The information about eventual errors (Bits AF and AC) can be stored now for later use in the main loop, if needed.

If bit ARX is set, a packet was received. Now the received packet data is available in the Asynchronous Receive Packet Buffer mARP. After having read mARP, the buffer must be released again for being able to receive the next packet. This is done by setting bit RARX in the packet control register mPCTC.

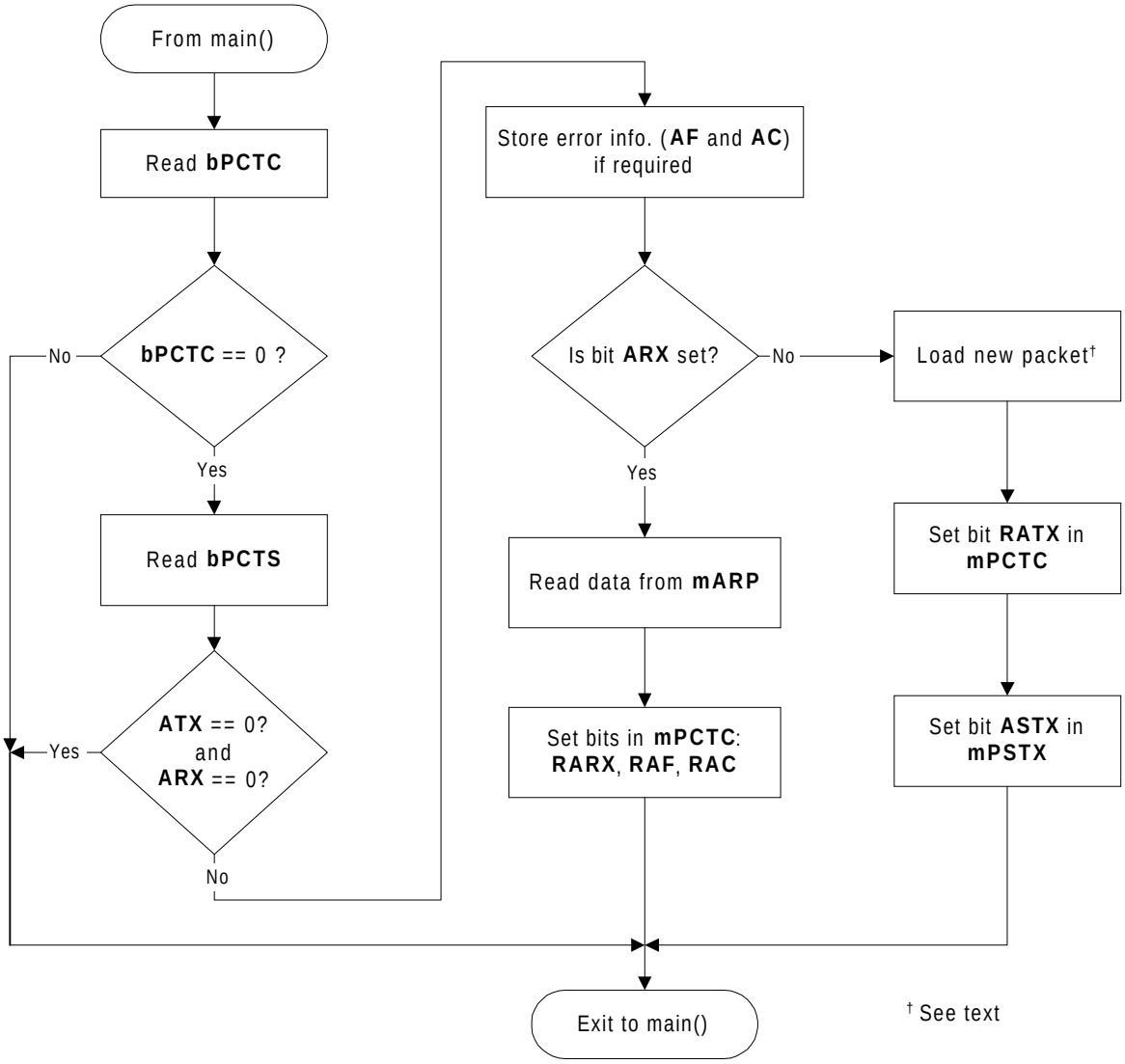


Figure 15-3: Handling Packet Data transfer by polling

If the error bits RAF and RAC have been set, they should also be cleared to prepare them for the next message. This ends the handling of a packet received event.

If ATX is set, a packet transmitted event occurred, and a new packet can be prepared for transmission. For starting transmission, bit ASTX in register bPSTX (See 15.1.5 on page 131) must be set. Bit ASTX will be reset automatically, after the transmit packet is sent.

16 OS8104 Startup

This section describes the start-up procedure for the MOST transceiver OS8104. The examples shown in the flow charts below describe the basic steps which must be performed to set up the standard configuration of the MOST chip. One main flow must be followed every time the OS8104 is powered up. After having processed the main flow, two branches configure the chip either as the timing-master or as a timing-slave mode. The Control Port is configured in I^2C mode for this example (set by tying PAR_CP low and placing a pull-up on SCL).

After power-up, the OS8104 must be reset by a rising edge on the /RS pin after the power supplies have stabilized. The Control Port and Source Port configuration pins must be set for the intending modes to be used in the application. The configuration options determine whether the Ports are used in serial or parallel mode, along with the format of the Control Port when used in serial mode. The configuration options are described in Section 4 on page 27.

To configure the Source Ports in serial mode and the Control Port in I^2C serial mode, the *PAR_CP* and *PAR_SRC* pins must be tied to ground, and the *SCL* pin must have a pull-up to VDD (*SCL* is an open-drain pin in I^2C mode and requires a pull-up anyway).

After reset, the /ABY bit is set configuring the part for all-bypass mode, where the data from RX is sent directly to TX, unaltered. The node is not "visible" to the network (it has no valid node position number and is not addressable by other nodes).

In slave mode there is always a system clock. If there is no signal at RX, the system clock's frequency is determined by the PLL running at its lowest frequency.

After reset, the chip is configured in slave mode, so the crystal oscillator is disabled. It will start to oscillate, if the clock manager is configured for using the crystal as a clock source.

The start-up procedures listed in this section are generic and useful for understanding the OS8104 requirements; however, these procedures do not cover all the start-up requirements of the MOST Specification. For the MOST Specification requirements, see the MOST Specification, *NetInterface* Section.

16.1 Set Up After Power Up Reset

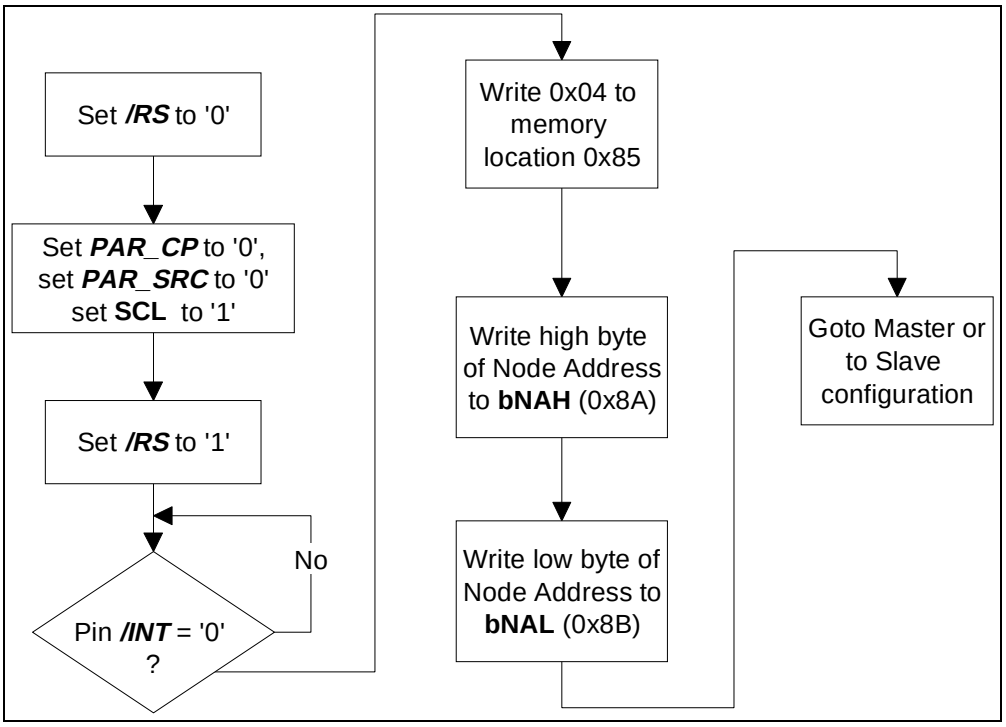
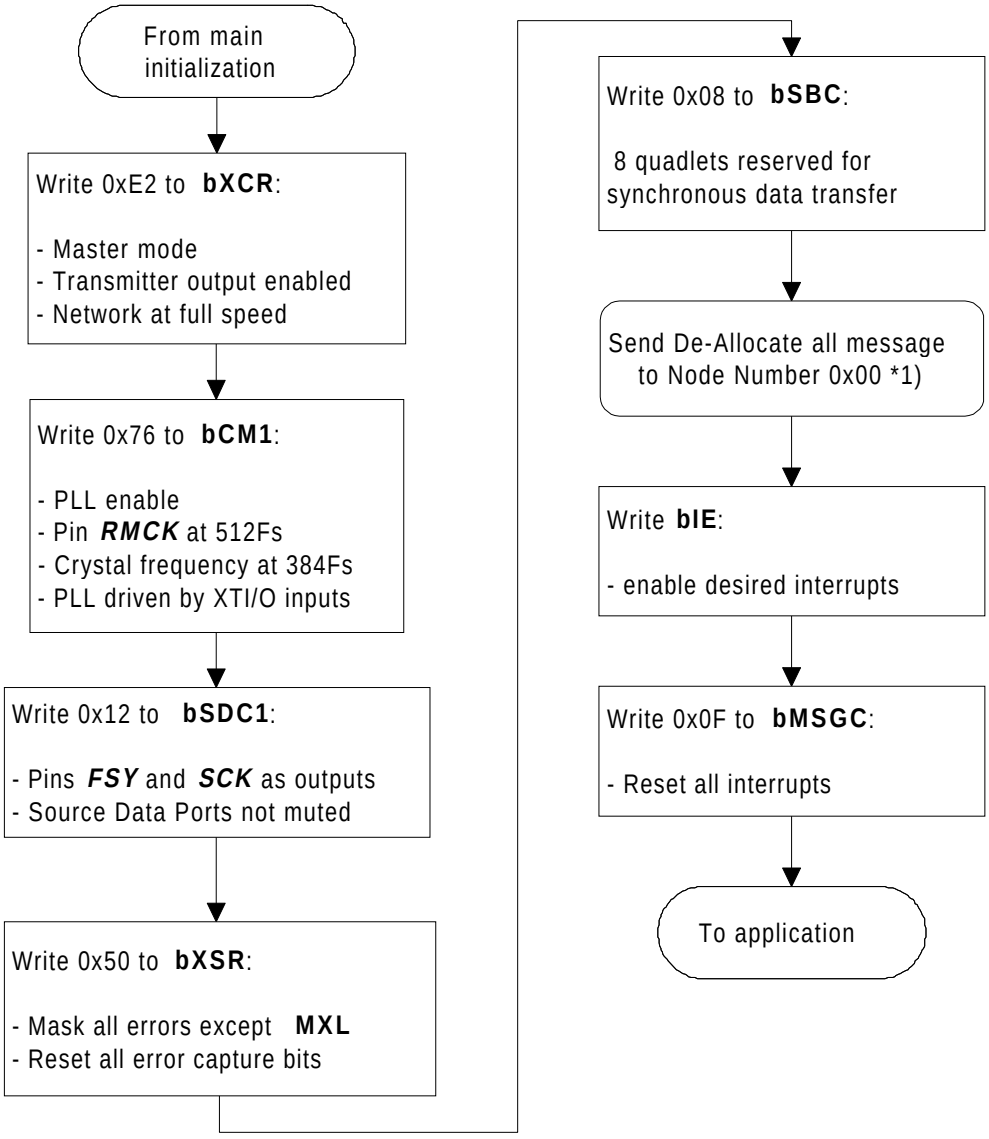


Figure 16-1: Starting up OS8104; Initial Flow

16.2 Master Mode



*1): Sending a "De-Allocate All" message is described in Section 13.5.2.5 on page 123.

Figure 16-2: Starting up OS8104; Configuring as Timing-Master

16.3 Slave Mode

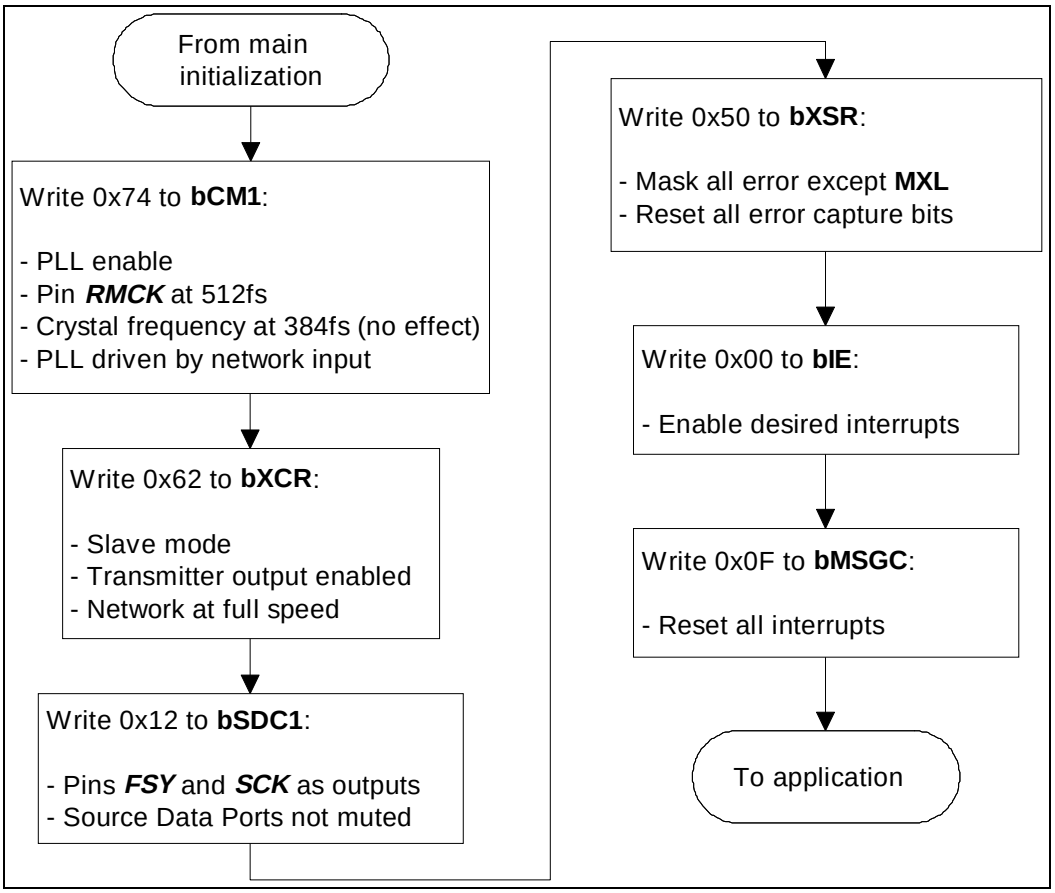


Figure 16-3: Starting up OS8104; Configuring as Slave

16.4 Version Number

The current chip version is provided after hardware reset (*/RS*) or power-up reset by reading memory locations 0xC4, 0xC5 and 0xC6, as illustrated in the Table below.

Revision		Month	Year
	0xC4	0xC5	0xC6
A	0x10	0x12	0x97
B	0x11	0x10	0x98
C	0x12	0x06	0x99
F	0x13	0x07	0x00

Table 16-1: OS8104 Version Numbers

The version number will be overwritten by the first write access to mXCMB

17 Stand-Alone Mode

The OS8104 can be configured in "Stand-Alone" or "Remote-Controlled" mode, where no local controlling hardware is needed. In this mode, the OS8104 receives all commands via the network and can control peripherals through the Control Port, which is automatically configured as an I^2C master. Another network node can then configure and control the OS8104 using "remote read" or "remote write" control messages (described in Section 13.5.2 on page 118). The OS8104 can also transmit MOST Control messages when interrupts occur. Currently the MOST Specification does not support Stand-Alone mode.

17.1 Entering Stand-Alone mode

To configure the OS8104 for Stand-Alone mode, the pins */RD* and */WR* must be tied to GND before the rising edge of /RS or before the power-up reset.

The chip will start up with the following configuration:

- MOST timing-slave mode (Clock recovered from the network)
- Output enabled
- Source bypass bit SBY active (all source data transferred directly from RX to TX)
- All-bypass bit /ABY inactive (set to 1)
- Automatic mute on error
  Whenever an error occurs, the outputs of the source data ports will be automatically muted by setting them constantly to 0. On start-up, only coding errors and transceiver errors are enabled. The S/PDIF lock error condition is masked.
- Control Port configured as I^2C master. The OS8104 only supports single-master operation. The I^2C speed is kept below 100 kHz to avoid clock stretching.
- Automatic Zero Power Mode on missing network activity
- The */INT* pin is configured as an input for receiving external application interrupts.

17.2 Registers

When Stand-Alone mode is active, the following registers are accessible:

- bCM1 (Clock Manager register 1; Refer to Section 9.1 on page 89)
- bCM2 (Clock Manager register 2; Refer to Section 9.2 on page 90)
- bSDC1 (Source Data Control register 1; Refer to Section 7.2.1 on page 44)
- bSDC2 (Source Data Control register 2; Refer to Section 7.2.2 on page 47)
- bSDC3 (Source Data Control register 3; Refer to Section 7.2.3 on page 48)
- bNDR (Node Delay register; Refer to Section 6.2.6 on page 39)
- bNPR (Node Position register; Refer to Section 6.2.7 on page 39)
- bXCR (Transceiver Control Register, Bit SBY only; Refer to Section 6.2.1 on page 35)
- mSIMB (Standalone I^2C Messaging Buffer)
- bXRTY (Xmit Retry Register; Refer to Section 13.2.4 on page 112)
- bXTIM (Xmit Retry Time Register; Refer to Section 13.2.5 on page 112)
- mXCMB (Xmit Control Message Buffer; Refer to Section 13.2.7 on page 114)

A unique message buffer, mSIMB, is provided to support I^2C-master communication in Stand-Alone mode.

17.3 mSIMB (Stand-Alone Control Port Messaging Buffer)

0xEB	mSIMB	Stand-Alone Control Port Messaging Buffer	
Byte	**Name**	**Description**	**Default**
0x00	bSIMC	Count of bytes sent	0x00
0x01	bSITA	I^2C target address	0x00
0x02	bSIMA	I^2C MAP	0x00
0x03	bSITD0	I^2C transfer data byte0	0x00
0x04	BSITD1	I^2C transfer data byte1	0x00
0x05	BSITD2	I^2C transfer data byte2	0x00
0x06	bSITD3	I^2C transfer data byte3	0x00
0x07	bSITC	I^2C transfer control byte/status	0x00

Table 17-1: mSIMB (Standalone CP Messaging Buffer)

bSIMC (Count of bytes sent)

bSIMC contains the count of bytes (after bSIMC) to be sent via the Control Port:

Minimum value is 2 (only bSITA and bSIMA are sent).

Maximum value is 6 (bSITA up to bSITD3 are sent)

bSITA (I^2C target address)

bSITA contains the address of the particular external I^2C device being accessed. I^2C devices are connected to the Control Port (pins *SCL* and *SDA*) which is configured as an I^2C master. The correct setting of the LSB in the I^2C address (for read or write access) is done automatically by the OS8104.

bSIMA (I^2C MAP)

bSIMA contains the Memory Address Pointer for the I^2C peripheral.

bSITD0..3 (I^2C transfer data byte0..3)

Data to be transferred to the external peripheral.

bSITC (I^2C transfer control byte)

bSITC controls and monitors the Control Port data transfer.

Command values (when writing to bSITC):

0x06: Execute I^2C write operation

0x07: Execute I^2C read operation

Status values (when reading from bSITC):

0x00 : Operation terminated/ready for new job

any other value : Busy

17.4 Writing to External Peripheral

The Stand-Alone Message buffer (mSIMB) must be filled before the external peripheral can be written to. The first value in the buffer is the count of bytes to be transmitted, and comprises all bytes from bSITA through bSITD3 (inclusive). The count is 6 to transfer all data bytes through bSITD3. The minimum count is 2 to transfer only bSITA and the MAP bSIMA.

In the following example, four bytes are transmitted, starting at memory location 0x33 on the external I^2C peripheral which has an I^2C address of 0x50. The OS8104, in Stand-Alone mode, has node position 0x05 in the MOST network.

Two devices are involved in transferring data:

- The chip running in Stand-Alone mode (RCC)
- The node sending the remote read and remote write Control Messages (CMS)

SITD0..3 contain the four data bytes, bSITA has the I^2C target address of 0x50, and bSIMA contains the MAP address 0x33 of where to start storing the data. The count is bSIMC = 4 + 2 = 6.

The correct contents for mSIMB in RCC is:

mSIMB		Stand-Alone Control Port Messaging Buffer
Name	**Value**	**Description**
bSIMC	0x06	Count of bytes to transmit
bSITA	0x50	I^2C address of target device
bSIMA	0x33	I^2C MAP
bSITD0	0x48	any data...
bSITD1	0x75	any data...
bSITD2	0x68	any data...
bSITD3	0x75	any data...
bSITC	0x06	I^2C write operation

Table 17-2: mSIMB when writing to I^2C In Stand Alone Mode

To get mSIMB of the RCC node filled, the CMS node must send a "remote write" control message. The table below shows the correct contents of the transmit control message buffer mXCMB of CMS. Since mSIMB is located at memory location 0xEB in RCC, this value must be written to the MAP byte in mXCMB (address 0xC5) of CMS.

mXCMB	Contents	Transmit Control Message Buffer mXCMB
0xC0	0x01	Priority; Default 0x01
0xC1	0x02	Remote write operation
0xC2	0x04	Target address high (of RCC in MOST network)
0xC3	0x05	Target address low
0xC4	0x00	rsvd
0xC5	0xEB	Memory location in RCC to write to (mSIMB buffer)
0xC6	0x08	Count of data bytes to be written
0xC7	0x06	D0 of Control Message. (bSIMC)
0xC8	0x50	D1 of Control Message. (bSITA)
0xC9	0x33	D2 of Control Message. (bSIMA)
0xCA	0x48	D3 of Control Message. (bSITD0)
0xCB	0x75	D4 of Control Message. (bSITD1)
0xCC	0x68	D5 of Control Message. (bSITD2)
0xCD	0x75	D6 of Control Message. (bSITD3)
0xCE	0x06	D7 of Control Message. (bSITC)

Table 17-3: Stand-Alone mode: Write Example

The mechanism for sending this remote write Control Message is described in Section 13.5.2.3 on page 119.

When RCC received this message, the I^2C data transmission will start immediately. The current status of this transmission can be obtained by reading bSITC (address 0xF2) via "remote read" Control message. As long as the chip is busy transmitting through the Control Port, bSITC will be non-zero.

Reading a zero from bSITC (STATUS) indicates the termination of the remote node's Control Port transmission. The "remote read" Control message for checking the transfer status would be:

mXCMB	Contents		Transmit Control Message Buffer mXCMB
	Sent	Returned	
0xC0	0x01	0x01	Priority; Default 0x01
0xC1	0x01	0x01	Remote read operation
0xC2	0x04	0x04	Target address high (of RCC in MOST network)
0xC3	0x05	0x05	Target address low
0xC4	0x00	0x00	rsvd
0xC5	0xEB	0xEB	Memory location in RCC to read from (mSIMB buffer)
0xC6	0x00	0x00	rsvd
0xC7	x	x	D0 of Control Message. (bSIMC)
0xC8	x	x	D1 of Control Message. (bSITA)
0xC9	x	x	D2 of Control Message. (bSIMA)
0xCA	x	x	D3 of Control Message. (bSITD0)
0xCB	x	x	D4 of Control Message. (bSITD1)
0xCC	x	x	D5 of Control Message. (bSITD2)
0xCD	x	x	D6 of Control Message. (bSITD3)
0xCE	x	STATUS	D7 of Control Message. (bSITC) Contains current status

Table 17-4: Stand-Alone mode: Write Example Status Check

More information about handling remote read messages can be found in Section 13.5.2.2 on page 119. The flow chart below shows how a Control Port transmission can be done:

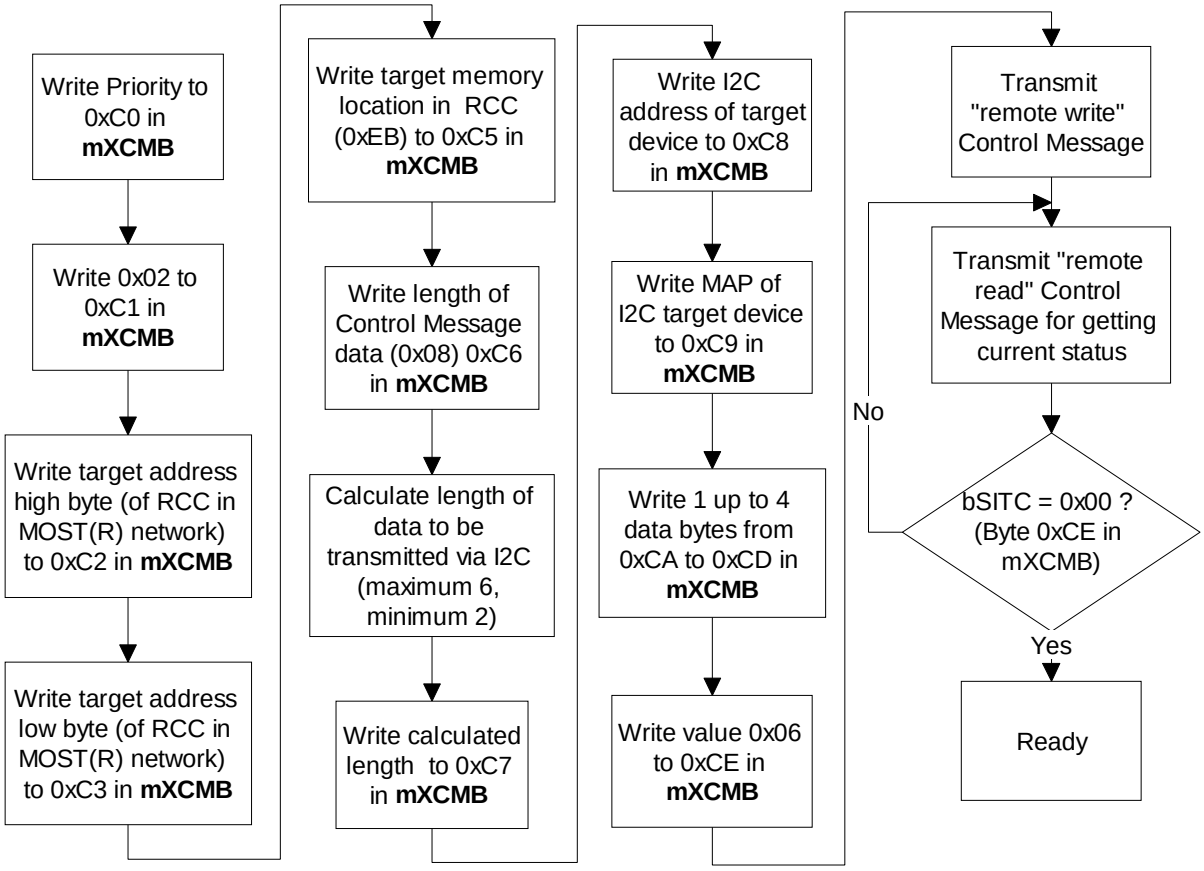


Figure 17-1: Stand-Alone Mode: Write Flow

17.5 Reading from External Peripheral

When reading data from an external peripheral via the I^2C bus, several MOST Control messages are needed. The controlling node must send one message to transfer the I^2C address, I^2C MAP, and the number of bytes to be read. The controlling node must then send one or more messages to transport data (read via I^2C) and status back. OS8104 handles the read cycles automatically, so that only one remote write access is needed for initializing the data transfer via I^2C bus. The number of bytes read from the external peripheral is always done in increments of four.

In the following example, data is read from an OS8104 running in Stand-Alone mode, with node position 0x05 in the MOST network.

When transferring data, two MOST devices are involved in this operation:

- The chip running in Stand-Alone mode (RCC)
- The controlling node sending the *remote read* and *remote write* Control messages (CMS)

For reading I^2C data, only the I^2C target address and the I^2C MAP must be transferred to the external peripheral. Since the number of bytes to be read from the external peripheral is always 4, the count of byte to transmit (bSIMC) is always 6. The external peripheral has an I^2C address of 0x50, and data will be read starting from memory location 0x07. Therefore, the correct data to transfer to mSIMB is:

mSIMB		Stand-Alone Control Port Messaging Buffer
Name	**Value**	**Description**
bSIMC	0x06	Count of bytes to transmit (always 0x06)
bSITA	0x50	I^2C address of target device
bSIMA	0x33	I^2C MAP
bSITD0	0x00	fill up data (don't care)
bSITD1	0x00	fill up data (don't care)
bSITD2	0x00	fill up data (don't care)
bSITD3	0x00	fill up data (don't care)
bSITC	0x07	I^2C read operation

Table 17-5: Stand-Alone mode: mSIMB Read Example

To get mSIMB of the RCC node filled, the CMS node must send a "remote write" control message. The table below shows the correct contents of the transmit control message buffer mXCMB of CMS. Since mSIMB is located at memory location 0xEB in RCC, this value must be written to the MAP byte in mXCMB (address 0xC5) of CMS.

mXCMB	Contents	Transmit Control Message Buffer mXCMB
0xC0	0x01	Priority; Default 0x01
0xC1	0x02	Remote write operation
0xC2	0x04	Target address high (of RCC in MOST network)
0xC3	0x05	Target address low
0xC4	0x00	rsvd
0xC5	0xEB	Memory location in RCC to write to (mSIMB buffer)
0xC6	0x08	Count of data bytes to be written
0xC7	0x06	D0 of Control Message. (bSIMC) – Must be 0x06.
0xC8	0x50	D1 of Control Message. (bSITA)
0xC9	0x33	D2 of Control Message. (bSIMA)
0xCA	x	D3 of Control Message. (don't care)
0xCB	x	D4 of Control Message. (don't care)
0xCC	x	D5 of Control Message. (don't care)
0xCD	x	D6 of Control Message. (don't care)
0xCE	0x07	D7 of Control Message. (bSITC)

Table 17-6: Stand-Alone mode: mXCMB Read Example

When RCC receives this message, the Control Port data transmission will start immediately. The current status of this transmission can be obtained by reading bSITC (address 0xF2) using a *remote read* Control message. As long as the chip is busy with the Control Port, bSITC will be non-zero. Reading a zero from bSITC flags the completion of the Control Port read.

The data read from the external peripheral is stored in bytes bSITD0..3. Therefore, reading the status and data can be accomplished using a single *remote read* message.

mXCMB	Contents		Transmit Control Message Buffer mXCMB
	Not Valid	Valid	
0xC0	0x01	0x01	Priority; Default 0x01
0xC1	0x01	0x01	Remote read operation
0xC2	0x01	0x01	Target address high (of RCC in MOST network)
0xC3	0x23	0x23	Target address low
0xC4	0x00	0x00	rsvd
0xC5	0xEB	0xEB	Memory location in RCC to read from (mSIMB buffer)
0xC6	0x00	0x00	rsvd
0xC7	x	0x06	D0 of Control Message. (bSIMC)
0xC8	x	0x50	D1 of Control Message. (bSITA)
0xC9	x	0x33	D2 of Control Message. (bSIMA)
0xCA	x	0xdd	D3 of Control Message. (Data byte0 from I^2C device)
0xCB	x	0xdd	D4 of Control Message. (Data byte1 from I^2C device)
0xCC	x	0xdd	D5 of Control Message. (Data byte2 from I^2C device)
0xCD	x	0xdd	D6 of Control Message. (Data byte3 from I^2C device)
0xCE	0x07	0x00	D7 of Control Message. (bSITC)

Table 17-7: Stand-Alone mode: mXCMB Read Example Status

If bSITC has the value 0x00, the remote mode has completed the Control Port read and the data bytes are valid. More information about handling remote read messages can be found in Section 13.5.2.2 on page 119. The flow chart below illustrates the reading of Control Port data on a remote node.

17.6 Message On Interrupt

When the OS8104 is configured for Stand-Alone mode, it can send a Control message when the external application pulls the */INT* pin low (which is configured as an input in Stand-Alone mode).

The minimum external interrupt pulse width is 2 µs.

This mechanism works in "single shot" mode. Therefore, after an interrupt event has been reported, the interrupt mechanism must be re-armed. All Control message addressing modes and transmit message types are supported.

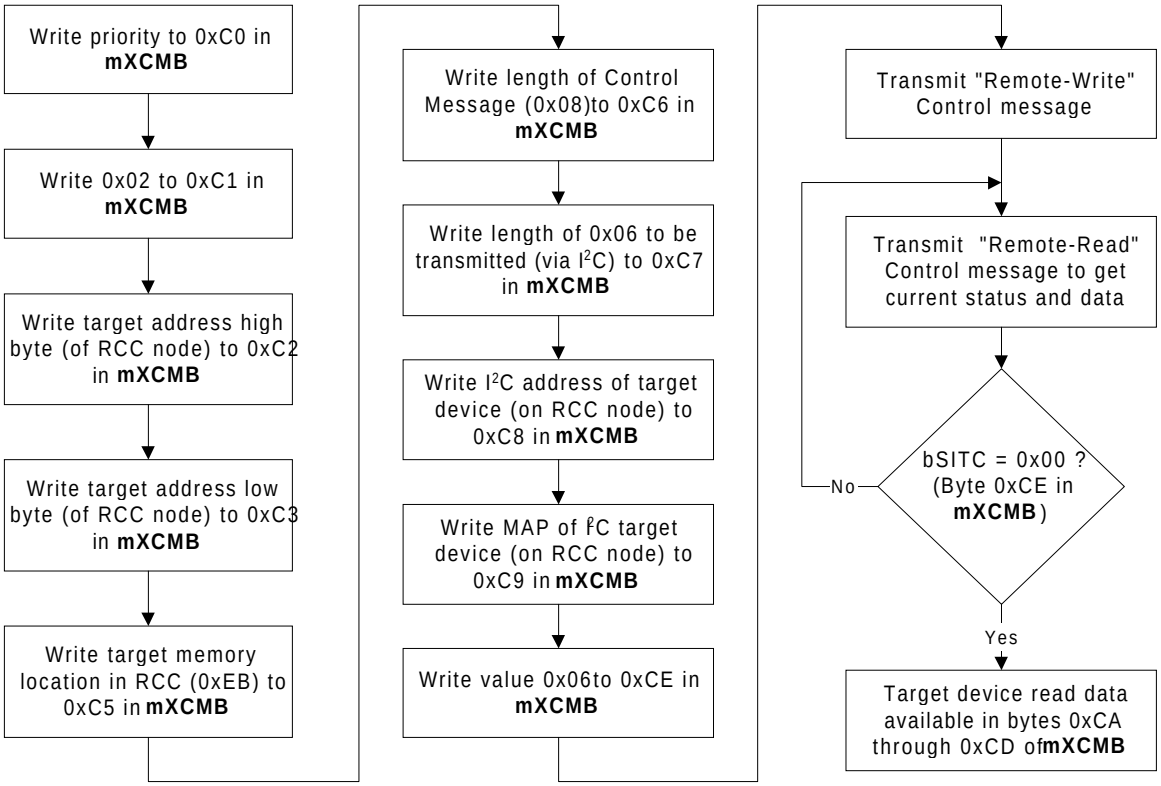


Figure 17-2: Stand-Alone mode: Read Example Flow

First, the Transmit Control Message Buffer mXCMB must be filled with the message to be sent out when an interrupt occurs. Since the remote interrupt mechanism is armed by setting the MSB in bXTYP, the remote transmit message buffer should be filled, starting from location 0xC2 (bXTAH). Once mXCMB on the remote node is filled, using the *remote write* control message type, the interrupt mechanism can be armed by writing the remote bXTYP byte in mXCMB. The bXTYP on the remote node is filled with the appropriate message typed, ORed with 0x80, thereby setting the MSB. For exampe, to send a normal message (type 0x00), the value in bXTYP would be 0x80.

Priority, number of retries, and retry time can also be changed, if required (registers bXPRI, bXRTY, bXTIM).

As soon as an interrupt occurs, the OS8104 in Stand-Alone mode will send out the prepared message and dis-arm the interrupt mechanism.

There will be no error reporting in case of transmission failure.

The message on interrupt mechanism is re-armed by writing to bXTYP again.

For more information on sending Control messages, see Section 13.4 on page 116).

18 Electrical Characteristics

Specifications are subject to change without notice.

18.1 Absolute Maximum Ratings

Parameter (note 1)	Min	Max	Unit
Storage Temperature	-65	150	°C
Ambient Temperature Under Bias	-55	125	°C
Power Supply Voltage	-0.5	6.0	V
Power Dissipation		990	mW
DC Current to Any Pin Except Power		±10	mA
Voltage on Any Pin	-0.3	VDD+0.3	V

Note:

1. Operation at or above these limits may damage the device.

18.2 Guaranteed Operating Conditions

The OS8104 operation is only guaranteed to function, as specified in this Data Sheet, within the limits listed below.

Parameter	Min	Max	Unit
Operating Ambient Temperature	-40	85	°C
Power Supply Voltage	4.5	5.5	V
Operating Network Sample Frequency (Fs)	38	48	kHz

18.3 DC Characteristics

T_A = -40 to 85 °C; VDDD,VDDA = 5 V ±10 %; GNDD,GNDA = 0.0 V

Parameter	Symbol	Min	Typ	Max	Unit	Test Conditions
Low Level Input Voltage: XTO Other Input pins	V_{IL}			 0.1×VDD 0.8	 V V	 (note 1)
High Level Input Voltage: XTO Other Input pins	V_{IH}	 0.9×VDD 2.0			 V V	 (note 1)
Low Level Output Voltage: RMCK Other Output pins	V_{OL}			 0.4 0.4	 V V	 I_{OL} = 2.0 mA I_{OL} = 2.4 mA
High Level Output Voltage: RMCK (note 2) Other Output Pins	V_{OH}	 VDD-0.5 VDD-0.5			 V V	 I_{OH} = -1.0 mA I_{OH} = -2.4 mA
Input Leakage Current	I_L			±10	µA	0 < Vin < VDD
Digital Input Pin Capacitance				10	pF	
Analog Supply Current: Normal: Low-Power mode:	I_A			 20 4	 mA mA	 Fs = 44.1 kHz
Digital Supply Current: Normal Low-Power mode	I_D			 160 19	 mA mA	 Fs = 44.1 kHz
Zero-Power mode: Analog and Digital Supplies combined	$I_A + I_D$			400	µA	
Thermal Resistance: junction-to-case junction-to-ambient junction-to-ambient	 θ_{JC} θ_{JA} θ_{JA}		 9.3 45 68		 °C/W °C/W °C/W	 note 3 note 4

Notes:

1. When driven externally (not using a crystal oscillator).
2. Except for open-drain outputs: */WAKE_UP*, */AINT*, */INT*, and in I^2C mode: *SDA* and *SCL*.
3. When soldered to a multi-layer PCB
4. When soldered to a single-layer PCB (no power or ground plane)

18.4 Switching Characteristics

18.4.1 Clocks and Reset

T_A = -40 to 85 °C; VDDD, VDDA = 5 V ±10 %; GNDD,GNDA = 0.0 V, PLL locked at Fs = 44.1 kHz, Load Capacitance = 40 pF; unless otherwise stated.

Parameter	Sym.	Min	Typ	Max	Unit	Comments
Crystal Oscillator	f_{xtal}	9.7024		24.6272	MHz MHz	256x(Fs = 37.9 kHz) 512x(Fs = 48.1 kHz)
Recovered Master Clock, *RMCK*	f_{rmck}	2.4256		73.8816	MHz MHz	64x(Fs = 37.9 kHz) 1536x(Fs = 48.1 kHz)
RMCK rise/fall time	t_{rmck}, t_{fmck}		7	10	ns	Load capacitance of 20 pF
Network Frame Frequency	Fs	37.9	44.1	48.1	kHz	
Pulse Width Distortion, *RX*	t_{pwd}	-1		+6	ns	
Pulse Width Distortion, *SR0*	t_{pwds}	-0.2		+0.2	UI	note 1
Jitter Tolerance: *RX (timing-slave)* *XTO, SR0, SCK (timing-master)*	t_{jit}			 6 0.8	 ns (pp) ns (pp)	 note 2
RX input rise/fall time	t_{rrx}, t_{frx}		10		ns	note 3
TX rise/fall time	t_{rtx}, t_{ftx}		2	7	ns	
Reset Pulse Width	t_{rspw}	10			ns	notes 4, 5
Power-up Access	t_{puac}			1	ms	note 6
Configuration pin setup to /RS rising: SCL, PAR_CP, PAR_SRC, ASYNC, /WR, /RD	t_{cpsrs}	20			ns	
Config. pin hold from /RS rising: SCL PAR_CP, PAR_SRC, ASYNC /WR, /RD	t_{cphrs}	 20 (note 7) 1			 ns ms	 Until t_{puac} time expires.

Notes:

1. *SR0* configured for S/PDIF and not as the timing-master node. One UI (unit interval) is defined as a single bi-phase period (one half of a bit period). Therefore, for 1xSPDIF mode with Fs = 44.1 kHz, one UI is 177 ns.
2. The MOST Specification requires each node have a crystal-based master-clock source to support ring-break diagnostics, where nodes normally configured as timing-slaves can be reconfigured as the timing-master.
3. Recommended. *RX* rise and fall time should be as short as possible to minimize jitter and PWD.
4. The OS8104 must be reset after the power supplies have stabilized for proper operation.
5. If */INT* was low prior to */RS* going low, short t_{rspw} times might not give */INT* time to rise properly, with */INT* still appearing low after */RS* rises. */INT* should be checked for a high-to-low going edge before accessing the chip.
6. Indicated via the power-on interrupt, */INT* pin. All accesses to the OS8104 must wait until the power-on interrupt occurs before accessing the chip via serial or parallel interfaces.
7. *PAR_CP*, *PAR_SRC*, and *ASYNC* can only change when */RS* is asserted and must remain constant when */RS* is de-asserted.

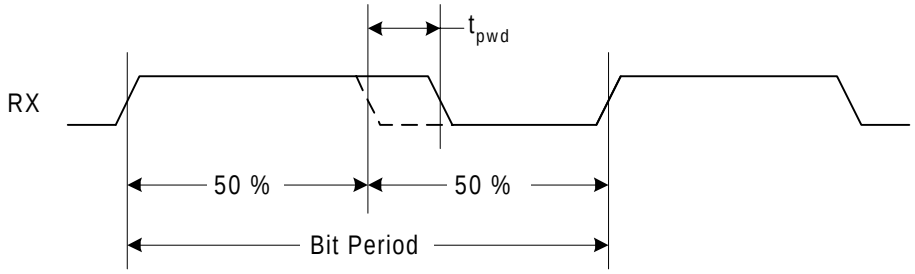


Figure 18-1: RX Pulse-Width Distortion

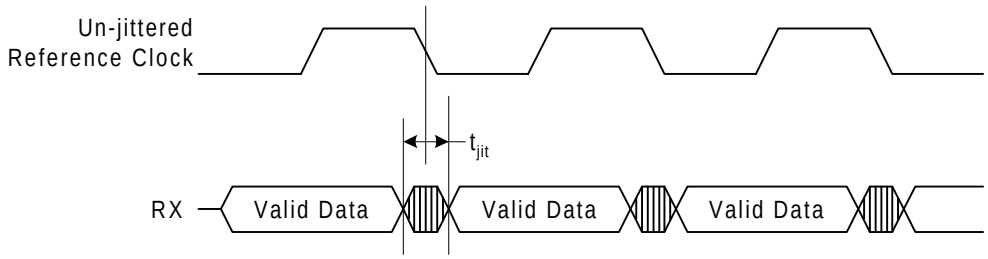


Figure 18-2: RX Jitter Tolerance

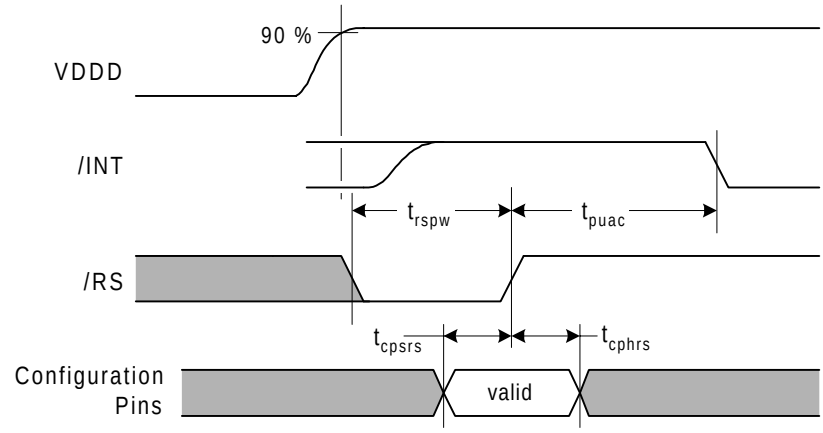


Figure 18-3: Reset Pulse Width

18.4.2 Parallel Interface

T_A = -40 to 85 °C; VDDD, VDDA = 5 V ±10 %; GNDD,GNDA = 0.0 V, PLL locked at Fs = 44.1 kHz, Load Capacitance = 40 pF

Parameter	Symbol	Min	Typ	Max	Unit	Comments
Read Operation:						
/RD low time	t_{rdp}	50			ns	
/RD high time	t_{rdh}	25			ns	
/RD low to D7..0 valid	t_{drd}			23	ns	
D7..0 hold from /RD high	t_{dhd}	0	2	10	ns	
PAD1..0 valid to /RD low	t_{asur}	10			ns	
PAD1..0 hold from /RD high	t_{ahdr}	4			ns	
Write Operation:						
/WR low time	t_{wrp}	50			ns	
/WR high time	t_{wrh}	25			ns	
D7..0 valid to /WR high	t_{dsu}	10			ns	
D7..0 hold from /WR high	t_{dhd}	4			ns	
PAD1..0 valid to /WR low	t_{asuw}	10			ns	
PAD1..0 hold from /WR high	t_{ahdw}	4			ns	
Parallel-Synchronous and Parallel-Combined modes:						
SRC_FLOW period	t_{ssf}	2.598		 3.298	µs µs	8x(Fs = 48.1 kHz) 8x(Fs = 37.9 kHz)
/WR, /RD inactive to SRC_FLOW high	t_{rws}	44.29			ns	note 1
SRC_FLOW high to /WR, /RD low	t_{ssfh}	0			ns	
SRC_FLOW rise/fall time	t_{rsf}, t_{fsf}		7	10	ns	
Parallel-Asynchronous mode:						
/WR, /RD high to SRC_FLOW high	t_{asfd}			40	ns	
SRC_FLOW high time	t_{asfh}		1/16Fs	1.1/8Fs	s	
Control Port Parallel Access:						
/WR, /RD high to CP_FLOW high	t_{cfd}			20	ns	
CP_FLOW high time	t_{cfh}	1	8		µs	

Notes:

1. In Parallel-Synchronous and Parallel-Combined modes, */WR* and */RD* must be inactive before the end of an *SRC_FLOW* period for this amount of time – which is dependant on the main frequency (Fs) of the network. Bits SPR2..0 in bSDC2 must be set to "101".
   The following formula can be used for calculating t_{rws} then:
   $t_{rws} = 1 / (512 \times Fs)$

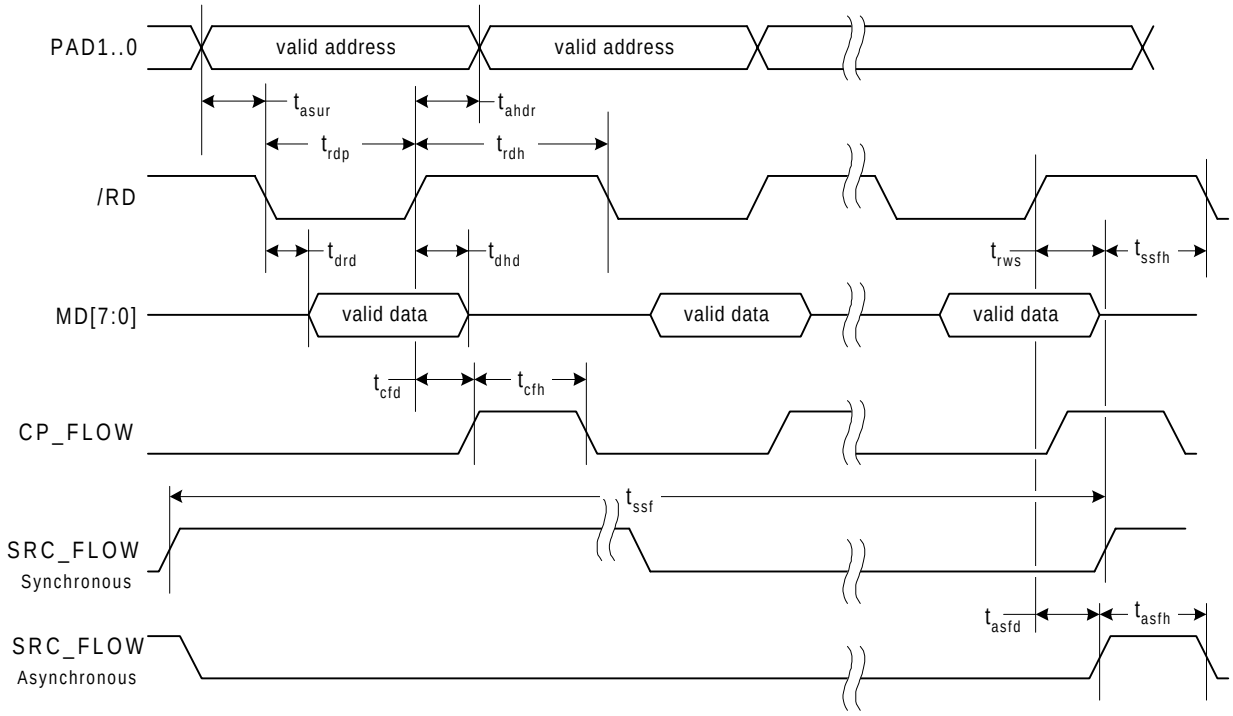


Figure 18-4: Parallel Read Operation

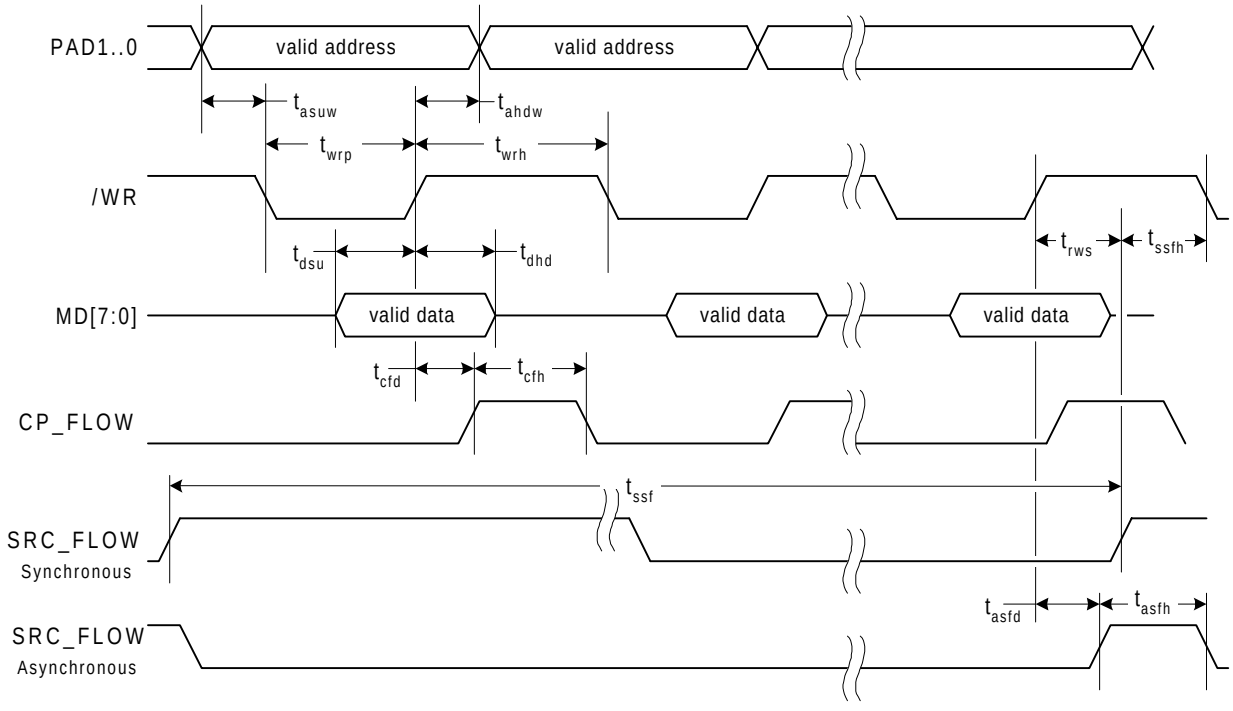


Figure 18-5: Parallel Write Operation

18.4.3 Source Data Ports (Serial) External Clocking

T_A = -40 to 85 °C; VDDD, VDDA = 5 V ±10 %; GNDD,GNDA = 0.0 V, PLL locked at Fs = 44.1 kHz, Load Capacitance = 40 pF

Parameter	Symbol	Min	Typ	Max	Unit	Comment
FSY frequency (note 1)	f_{fsy}	37.9	44.1	48.1	kHz	
SCK frequency (note 1)	f_{sck}	0.3032		12.3136	MHz MHz	Min.: 8Fs @ 37.9 kHz Max.:256Fs @ 48.1 kHz
SCK low time	t_{sckl}	25			ns	
SCK high time	t_{sckh}	25			ns	
FSY valid to SCK rising	t_{fsys}	25			ns	notes 2, 3
FSY hold from SCK rising	t_{fsyh}	25			ns	notes 2, 3
SR0..3 valid to SCK rising	t_{srs}	25			ns	notes 2, 4
SR0..3 hold from SCK rising	t_{srh}	25			ns	notes 2, 4
SCK falling to SX0..3 valid	t_{sxv}			30	ns	notes 2, 4

Notes:

1. *SCK* and *FSY* inputs must be frequency locked to OS8104 master clock (*RMCK* output clock).
2. *SCK* active edge (edge where data is stable and not changing) is determined by the EDG bit in bSDC1. The parameters and Figure 18-6 are illustrated with EDG set to 1. If EDG is zero, reverse the edge in the parameters listed above and invert *SCK* in the diagram below.
3. *FSY* polarity is determined by the POL bit in bSDC1.
4. The MSB of *SR0..3* and *SX0..3* is either the first or the second bit after *FSY* changes, based on the DEL bit in bSDC1.

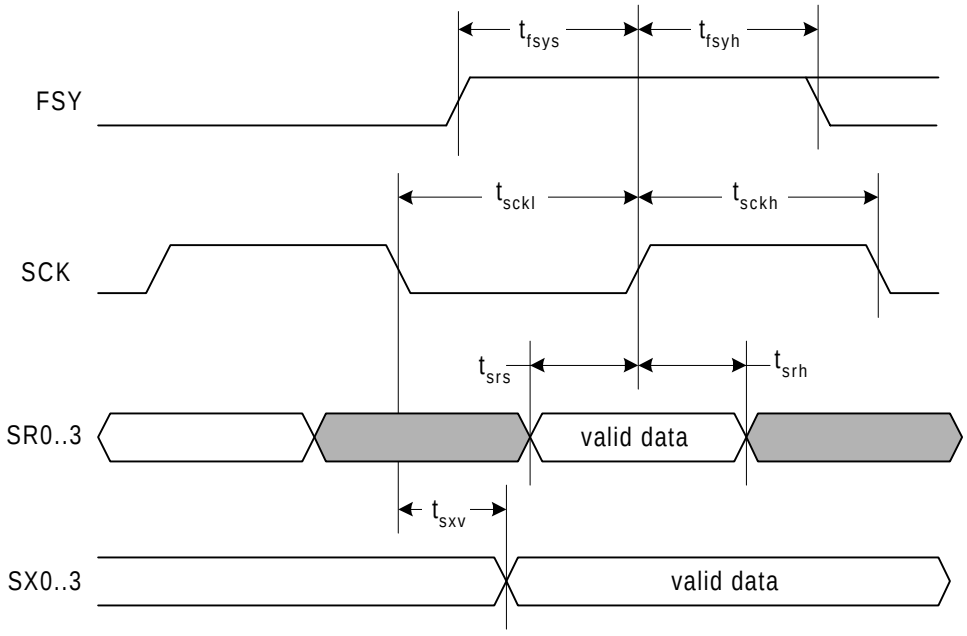


Figure 18-6: Source Port External Timing

18.4.4 Source Data Ports (Serial) Internal Clocking

T_A = -40 to 85 °C; VDDD, VDDA = 5 V ±10 %; GNDD,GNDA = 0.0 V, PLL locked at Fs = 44.1 kHz, Load Capacitance = 40 pF

Parameter	Symbol	Min	Typ	Max	Unit	Comment
FSY frequency	f_{fsy}	37.9	44.1	48.1 96.2 192.4	kHz kHz kHz	MFSY = 0 MFSY = 1, *SCLK* = 128xFs MFSY = 1, *SCLK* = 256xFs
SCK frequency	f_{sck}	0.3032		 12.3136	MHz MHz	Min.: 8xFs @ 37.9 kHz Max.:256xFs @ 48.1 kHz
SCK low time	t_{sckl}	25			ns	
SCK high time	t_{sckh}	25			ns	
SCK falling to FSY valid	t_{fsyv}	-25		25	ns	notes 1, 2
SCK/FSY rise/fall time	t_{rsp}, t_{fsp}		7	10	ns	
SR0..3 valid to SCK rising	t_{srs}	25			ns	notes 1, 3
SR0..3 hold from SCK rising	t_{srh}	25			ns	notes 1, 3
SCK falling to *SX0..3* valid	t_{sxv}	-25		30	ns	notes 1, 3

Notes:

1. *SCK* active edge (edge where data is stable and not changing) is determined by the EDG bit in bSDC1. The parameters and Figure 18-7 are illustrated with EDG set to 1. If EDG is zero, reverse the edge in the parameters listed above and invert *SCK* in the diagram below.
2. *FSY* polarity is determined by the POL bit in bSDC1. *FSY* is a 50 % duty cycle clock where the number of *SCK* cycles per *FSY* period is defined by the SPR2..0 bits in bSDC2 and the NBR bit in bSDC1.
3. The MSB of *SR0..3* and *SX0..3* is either the first or the second bit after *FSY* changes, based on the DEL bit in bSDC1.

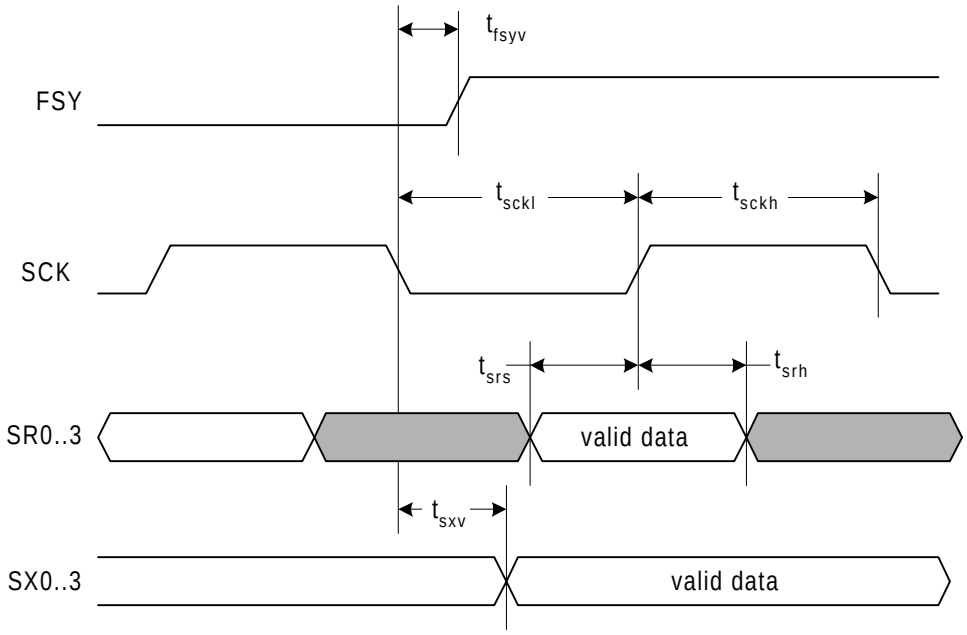


Figure 18-7: Source Port Internal Timing

18.4.5 Control Port In SPI Mode

T_A = -40 to 85 °C; VDDD, VDDA = 5 V ±10 %; GNDD,GNDA = 0.0 V,
PLL locked at Fs = 44.1 kHz, Load Capacitance = 40 pF

Parameter	Symbol	Min	Typ	Max	Unit
SCL frequency	f_{scl}			200	kHz
SCL low time	t_{scll}	1			µs
SCL high time	t_{sclh}	1			µs
SCL low to /CS low setup	t_{sklcsl}	250			ns
SCL low to /CS high	t_{sklcsh}	1			µs
/CS low to *SCL* rising	t_{css}	1			µs
/CS high time	t_{cht}	1			µs
SDIN valid to *SCL* rising	t_{sds}	500			ns
SDIN hold from *SCL* rising	t_{sdh}	500			ns
SCL fall to *SDOUT* valid (note 1)	t_{sdv}			1	µs
/CS low to *SDOUT* driven	t_{cdv}	500			ns
/CS high to *SDOUT* Hi-Z	t_{sdz}			1	µs

Notes:

1. When the PLL is unlocked, t_{sdv} increases to 2 µs.

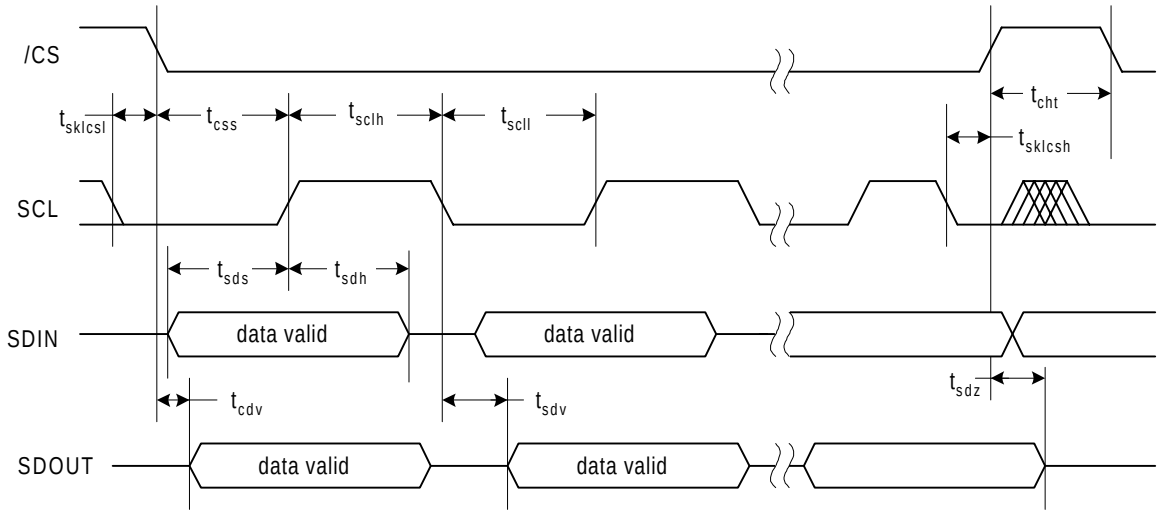


Figure 18-8: Control Port Timing in SPI mode.

18.4.6 Control Port In I²C Mode

T_A = -40 to 85 °C; VDDD, VDDA = 5 V ±10 %; GNDD,GNDA = 0.0 V,
PLL locked at Fs = 44.1 kHz, Load Capacitance = 40 pF

		I²C Master (note 1)			
Parameter	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
SCL frequency	f_{scl}			100	kHz
Bus free between transmissions (*SDA* high time between start and stop)	t_{buf}	4.7			µs
Start condition hold time (*SDA* falling to *SCL* falling)	t_{stah}	4			µs
SCL low time	t_{scll}	4.7			µs
SCL high time	t_{sclh}	4			µs
SCL falling to *SDA* input hold time	t_{sdah}	0			µs
SDA input setup time to *SCL* rising	t_{sdas}	250			ns
(repeated) start condition setup time	t_{stas}	4.7			µs
SDA and *SCL* rise time	t_{riic}			1	µs
SDA and *SCL* fall time	t_{fiic}			300	ns
Stop condition setup time (*SCL* rising to *SDA* rising)	t_{stps}	4			µs

Note:

1. The I²C Master format is only available when the OS8104 is configured for Stand-Alone mode. See Section 17 for more details.

		I²C Slave			
Parameter	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
SCL frequency (note 1)	f_{scl}			400	kHz
Bus free between transmissions (*SDA* high time between start and stop)	t_{buf}	1.3			µs
Start condition hold time (*SDA* falling to *SCL* falling)	t_{stah}	0.6			µs
SCL low time	t_{scll}	1.3			µs
SCL high time	t_{sclh}	0.6			µs
SCL falling to *SDA* input hold time	t_{sdah}	0		0.9	µs
SDA input setup time to *SCL* rising	t_{sdas}	100			ns
(repeated) start condition setup time	t_{stas}	0.6			µs
SDA and *SCL* rise time	t_{riic}			300	ns
SDA and *SCL* fall time	t_{fiic}			300	ns
Stop condition setup time (*SCL* rising to *SDA* rising)	t_{stps}	0.6			µs

Note:

1. Above 200 kHz, clock stretching may occur.

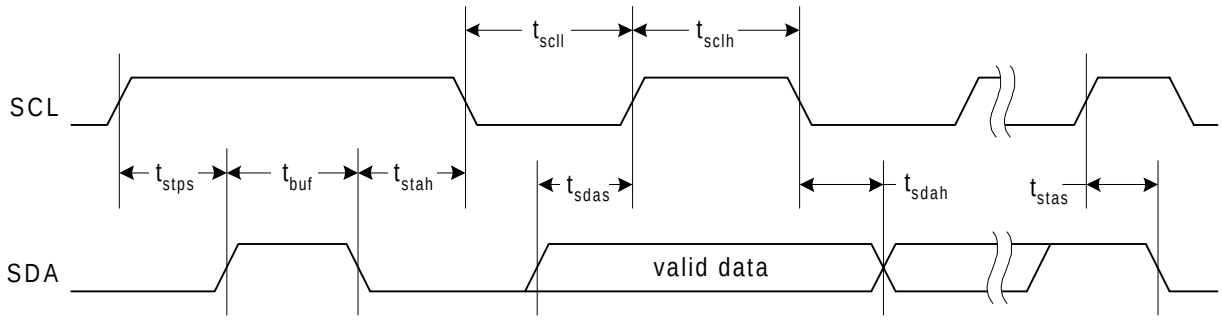


Figure 18-9: Control Port Timing in I²C mode.

19 Packaging and Pinout

19.1 Pinout List

Pin	Serial Name	Parallel Name	Type	Pin Description
1	PAR_SRC	PAR_SRC	D_{IN}	Parallel Source Data Port select input
2	Tie to GNDD	PAD0	D_{IN}	Parallel address select input bit 0
3	TX	TX	D_{OUT}	MOST transmitter output
4	RMCK	RMCK	D_{OUTZ}	Recovered master clock output
5	VDDD	VDDD	P	Digital power supply
6	GNDD	GNDD	G	Digital ground
7	Tie to VDDD	/RD	D_{IN}	Parallel mode read enable input
8	Tie to VDDD	/WR	D_{IN}	Parallel mode write enable input
9	SX0	D0	$D_{I/O}$	Source port data output 0 (Bit 0 of parallel data i/o)
10	SX2	D1	$D_{I/O}$	Source port data output 2 (Bit 1 of parallel data i/o)
11	Tie to GNDD	PAD1	D_{IN}	Parallel address select input bit 1
12	SX3	D2	$D_{I/O}$	Source port data output 3 (Bit 2 of parallel data i/o)
13	FSY	FSY	$D_{I/O}$	Frame sync i/o
14	SCK	SRC_FLOW	$D_{I/O}$	Serial bit clock i/o (Parallel data FIFO flow control output)
15	SR0	D3	$D_{I/O}$	Source port data input 0 (Bit 3 of parallel data i/o)
16	SR2	D4	$D_{I/O}$	Source port data input 2 (Bit 4 of parallel data i/o)
17	SR3	D5	$D_{I/O}$	Source port data input 3 (Bit 5 of parallel data i/o)
18	VDDP	VDDP	P	Periphery power supply
19	GNDP	GNDP	G	Periphery ground
20	XTI	XTI	X	Crystal oscillator input
21	XTO	XTO	X	Crystal oscillator output
22	/WAKE_UP	/WAKE_UP	D_{OODP}	Zero power mode wake-up control
23	STATUS	STATUS	D_{OUT}	Chip power mode status. A 1 indicates the chip is powered down
24	R_TIMER	R_TIMER	A	Wake-up timer resistor input
25	VREF	VREF	A	Voltage reference input
26	FLT	FLT	A	PLL loop filter input
27	GNDA	GNDA	G	Analog ground
28	GNDA	GNDA	G	Analog ground
29	VDDA	VDDA	P	Analog power supply
30	VDDA	VDDA	P	Analog power supply
31	/RS	/RS	D_{IN}	Hardware reset input
32	/CS AD1	/CS AD1	D_{IN}	Chip select input for SPI mode Address bit 1 input for I^2C mode
33	Do not connect	CP_FLOW	D_{OUT}	Control port flow control indicator output
34	ASYNC	ASYNC	D_{IN}	1: Chip is configured for asynchronous data i/o 0: Chip is configured for source data i/o
35	SR1	D6	$D_{I/O}$	Source port data input 1 (Bit 6 of parallel data i/o)
36	SX1	D7	$D_{I/O}$	Source port data output 1 (Bit 7 of parallel data i/o)
37	ERROR	ERROR	D_{OUT}	Error indicator output
38	SDIN AD0	SDIN AD0	D_{IN}	Serial data input in SPI mode Address bit 0 input for I^2C mode
39	SCL	SCL	$D_{I/O+CONF}$	I^2C clock (I/O) SPI clock (I)
40	SDA SDOUT	SDA SDOUT	$D_{I/O+CONF}$	I^2C data (I/O) SPI output data (O)
41	/INT	/INT	$D_{I/O+OD}$	Control message and power-on interrupt pin
42	RX	RX	D_{IN}	MOST receiver input
43	/AINT	/AINT	$D_{I/O+OD}$	Asynchronous message interrupt
44	PAR_CP	PAR_CP	D_{IN}	Parallel Control Port select input

19.2 State Of Pins in Low Power/Zero Power Mode

Pin	Name (serial)	Name (parallel)	Low Power	Zero Power
1	PAR_SRC	PAR_SRC	input inactive	input inactive
2	Tie to GNDD	PAD0	input inactive	input inactive
3	TX	TX	output active	output 0
4	RMCK	RMCK	output active	output inactive (Hi-Z, should be pulled high or low)
5	VDDD	VDDD	---	---
6	GNDD	GNDD	---	---
7	Tie to VDDD	/RD	input inactive [3]	input inactive [3]
8	Tie to VDDD	/WR	input inactive (wake-up active)[2]	input inactive (wake-up active)[2]
9	SX0	D0	output 0	output 0
10	SX2	D1	output 0	output 0
11	Tie to GNDD	PAD1	input inactive	input inactive
12	SX3	D2	output 0	output 0
13	FSY	FSY	input inactive	input inactive
14	SCK	SRC_FLOW	output 0	output 0
15	SR0	D3	input inactive	input inactive
16	SR2	D4	input inactive	input inactive
17	SR3	D5	input inactive	input inactive
18	VDDP	VDDP	---	---
19	GNDP	GNDP	---	---
20	XTI	XTI	normally inactive [4]	inactive
21	XTO	XTO	normally inactive [4]	inactive
22	/WAKE_UP	/WAKE_UP	output 0	output strobing
23	STATUS	STATUS	output 0	output 1
24	R_TIMER	R_TIMER	---	---
25	VREF	VREF	---	---
26	FLT	FLT	---	---
27	GNDA	GNDA	---	---
28	GNDA	GNDA	---	---
29	VDDA	VDDA	---	---
30	VDDA	VDDA	---	---
31	/RS	/RS	input active	input active
32	/CS AD1	/CS AD1	input inactive (wake-up active)[2]	input inactive (wake-up active)[2]
33	Do not connect	CP_FLOW	output keeps last value	output 0
34	Tie to GNDD	ASYNC	input inactive	input inactive
35	SR1	D6	input inactive	input inactive
36	SX1	D7	output 0	output 0
37	ERROR	ERROR	output active [5]	output 1
38	SDIN AD0	SDIN AD0	input inactive	input inactive
39	SCL	SCL	input inactive	input inactive
40	SDA SDOUT	SDA SDOUT	input or output inactive [1] [2]	input or output inactive [1] [2]
41	/INT	/INT	keeps last value	Hi-Z (pulled up by external resistor)
42	RX	RX	input inactive	input inactive
43	/AINT	/AINT	Hi-Z (pulled up by external resistor)	Hi-Z (pulled up by external resistor)
44	PAR_CP	PAR_CP	input inactive	input inactive

Pin	Name (serial)	Name (parallel)	Low Power	Zero Power

Notes:

Active means that the circuitry behind the pin will function in response to it, or will generate responses on the pin

1. When output (SPI mode), it is in Hi-Z state.
2. These signals can be used to wake up the chip in low power or zero power mode. /CS can only wake up in SPI mode, SDA only in I^2C mode.
3. In parallel mode, normal functionality disabled, but status registers can be read.
4. In slave mode generally disabled (clock recovered from RX), in master mode active.
5. If bXSR is set in the respective way, this pin can reflect bXCR lock status.

Table 19-1: Behavior of pins in low power, zero power

19.3 State Of Pins During Reset

Pin	Name (serial)	Name (parallel)	State during Reset (/RS active)
1	PAR_SRC	PAR_SRC	input
2	Tie to GNDD	PAD0	input
3	TX	TX	output driven low
4	RMCK	RMCK	output, driven low
5	VDDD	VDDD	---
6	GNDD	GNDD	---
7	Tie to VDDD	/RD	input
8	Tie to VDDD	/WR	input
9	SX0	D0	Hi-Z
10	SX2	D1	Hi-Z
11	Tie to GNDD	PAD1	input
12	SX3	D2	Hi-Z
13	FSY	FSY	Hi-Z output, driven low
14	SCK	SRC_FLOW	Hi-Z output, driven opposite polarity to ASYNC
15	SR0	D3	Hi-Z
16	SR2	D4	Hi-Z
17	SR3	D5	Hi-Z
18	VDDP	VDDP	---
19	GNDP	GNDP	---
20	XTI	XTI	input
21	XTO	XTO	output, driven to inverse of XTI
22	/WAKE_UP	/WAKE_UP	output, driven low
23	STATUS	STATUS	output, driven low
24	R_TIMER	R_TIMER	---
25	VREF	VREF	---
26	FLT	FLT	---
27	GNDA	GNDA	---
28	GNDA	GNDA	---
29	VDDA	VDDA	---
30	VDDA	VDDA	---
31	/RS	/RS	input, must be driven low externally
32	/CS AD1	/CS AD1	input
33	Do not connect	CP_FLOW	output, driven high (note 1)
34	Tie to GNDD	ASYNC	input
35	SR1	D6	Hi-Z
36	SX1	D7	Hi-Z
37	ERROR	ERROR	output, driven high
38	SDIN AD0	SDIN AD0	input
39	SCL	SCL	Hi-Z
40	SDA SDOUT	SDA SDOUT	Hi-Z
41	/INT	/INT	Hi-Z
42	RX	RX	input
43	/AINT	/AINT	Hi-Z
44	PAR_CP	PAR_CP	input

Note: 1. Older revisions drove CP_FLOW low in the reset state.

Table 19-2: State of pins during reset

19.4 Equivalent Schematics For Pins

19.4.1 Analog Pin (A)

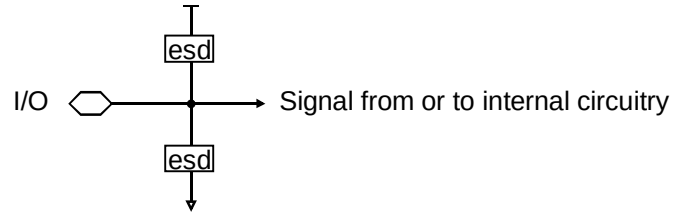


Figure 19-1: Pin equivalent schematic for analog input pins and output pins (A)

19.4.2 Digital Input Pin (D_{IN})

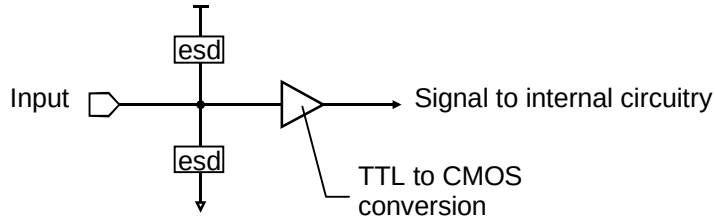


Figure 19-2: Pin equivalent schematic for digital input pin (D_{IN})

19.4.3 Digital Output Pin (D_{OUT})

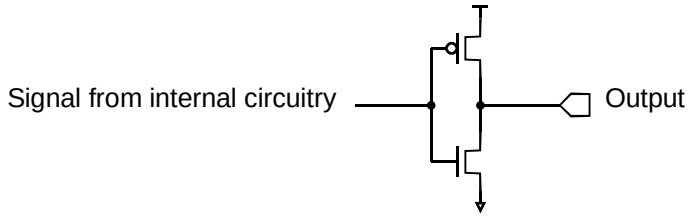


Figure 19-3: Pin equivalent schematic for digital output pin (D_{OUT})

19.4.4 Digital Output Pin With Tri-State (D_{OUTZ})

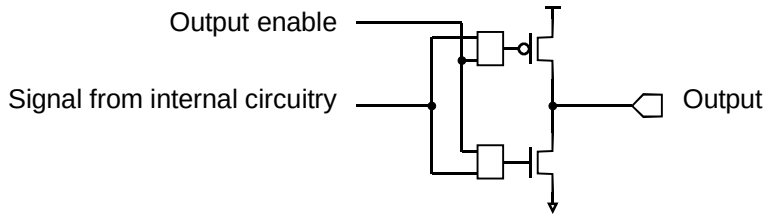


Figure 19-4: Pin equivalent schematic for digital output pin with Tri-State (D_{OUTZ})

19.4.5 Bi-directional Digital Pin ($D_{I/O}$)

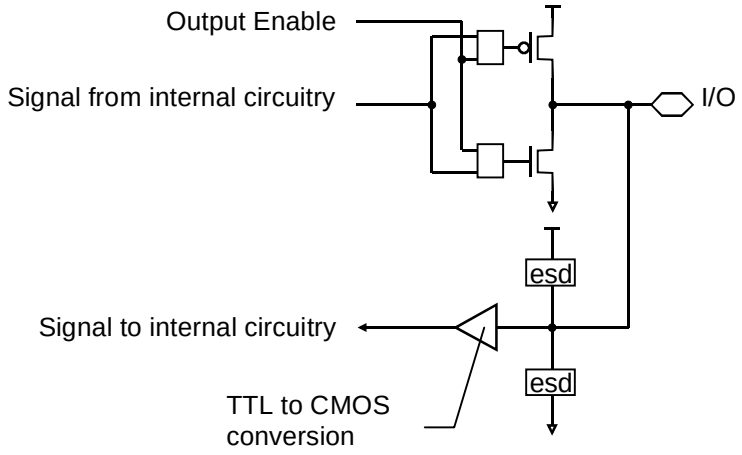


Figure 19-5: Pin equivalent schematic for bi-directional digital output pin ($D_{I/O}$)

19.4.6 Digital Open-Drain Output Pin With internal Pull-up (D_{OODP})

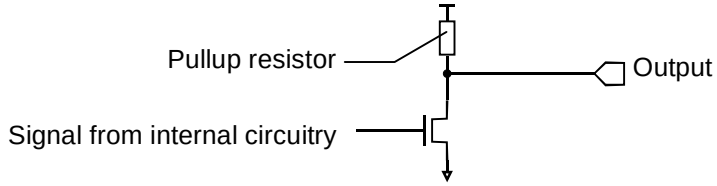


Figure 19-6: Pin equiv. schematic for digital open-drain output pin with internal pull-up (D_{OODP})

19.4.7 Bi-directional Digital And Configurable Pin ($D_{I/O+CONF}$)

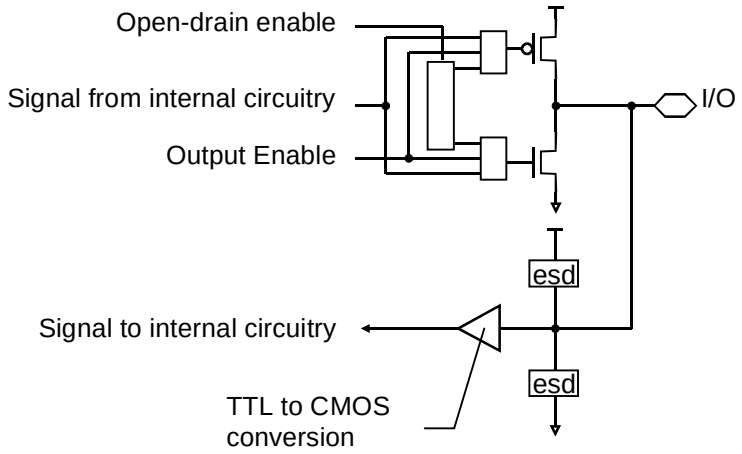


Figure 19-7: Pin equivalent schematic for bi-directional digital and configurable pin ($D_{I/O+CONF}$)

19.4.8 Bi-directional Digital Pin With Open-Drain ($D_{I/O+OD}$)

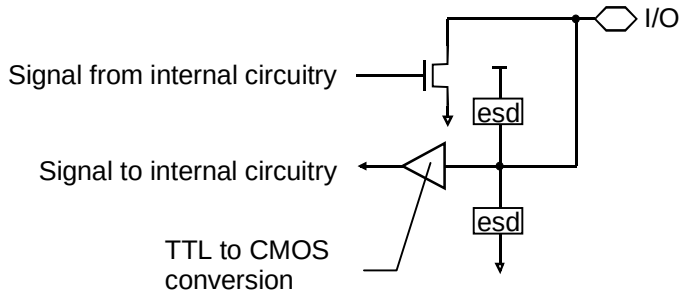
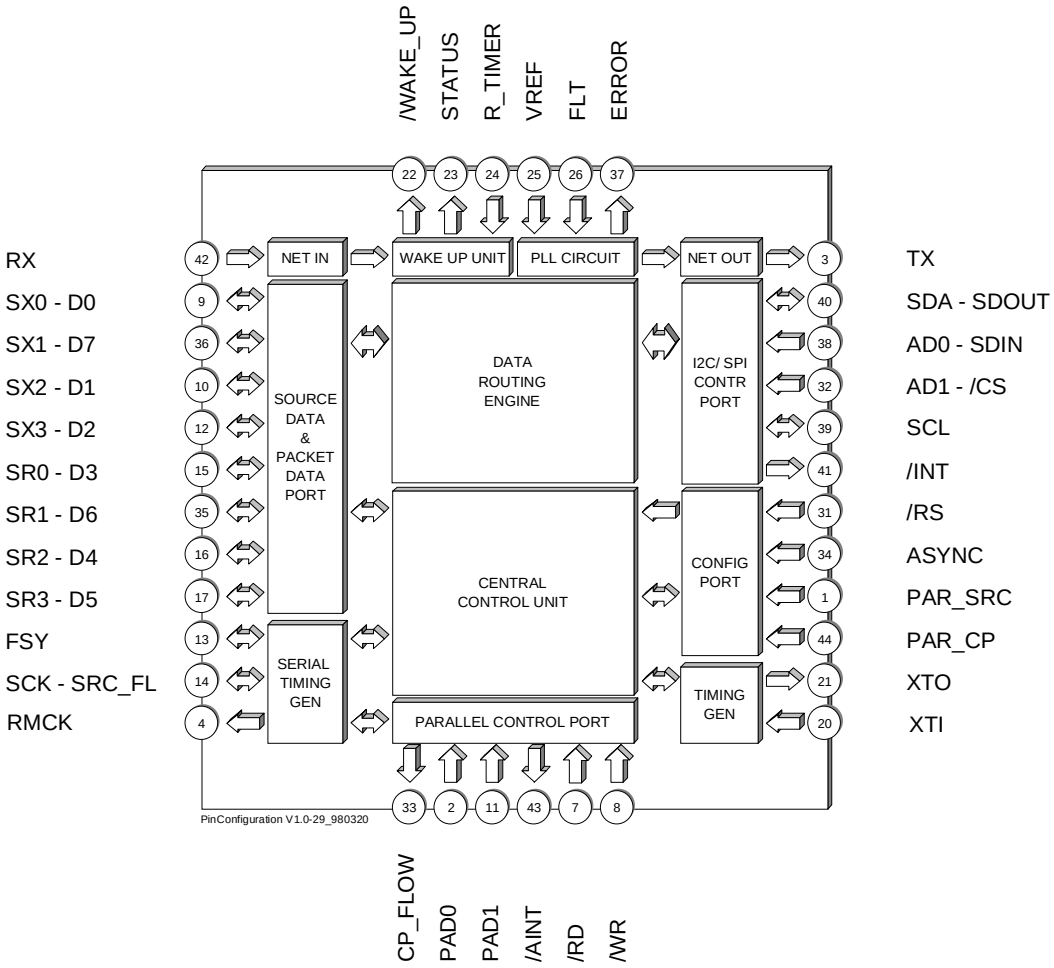
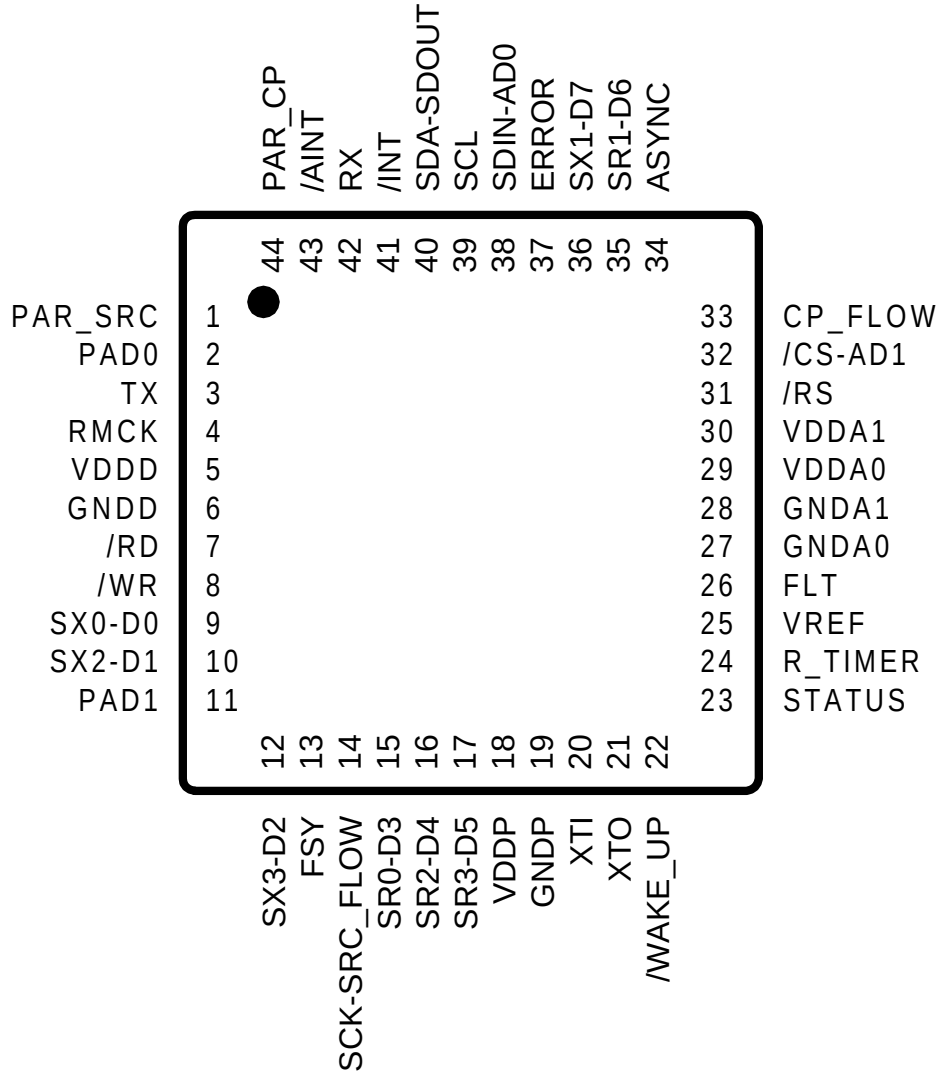


Figure 19-8: Pin equivalent schematic for bi-directional digital pin with open-drain ($D_{I/O+OD}$)

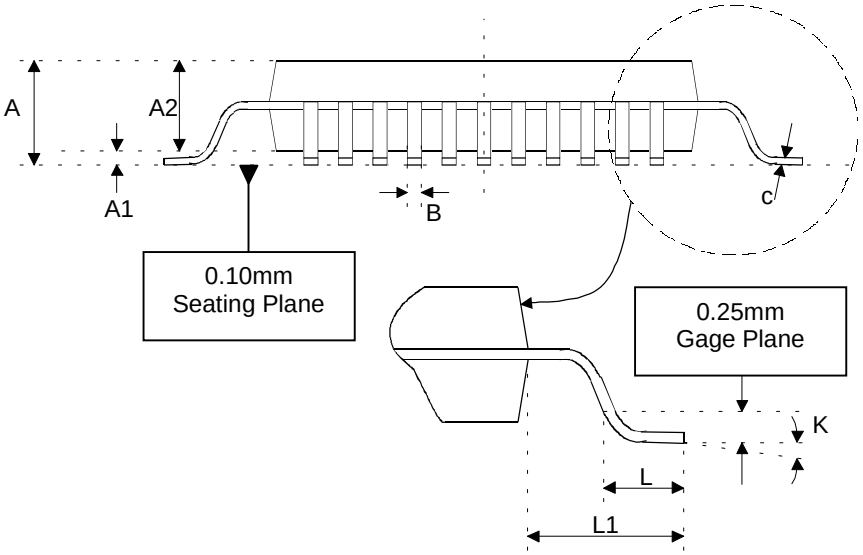
19.5 Block Diagram

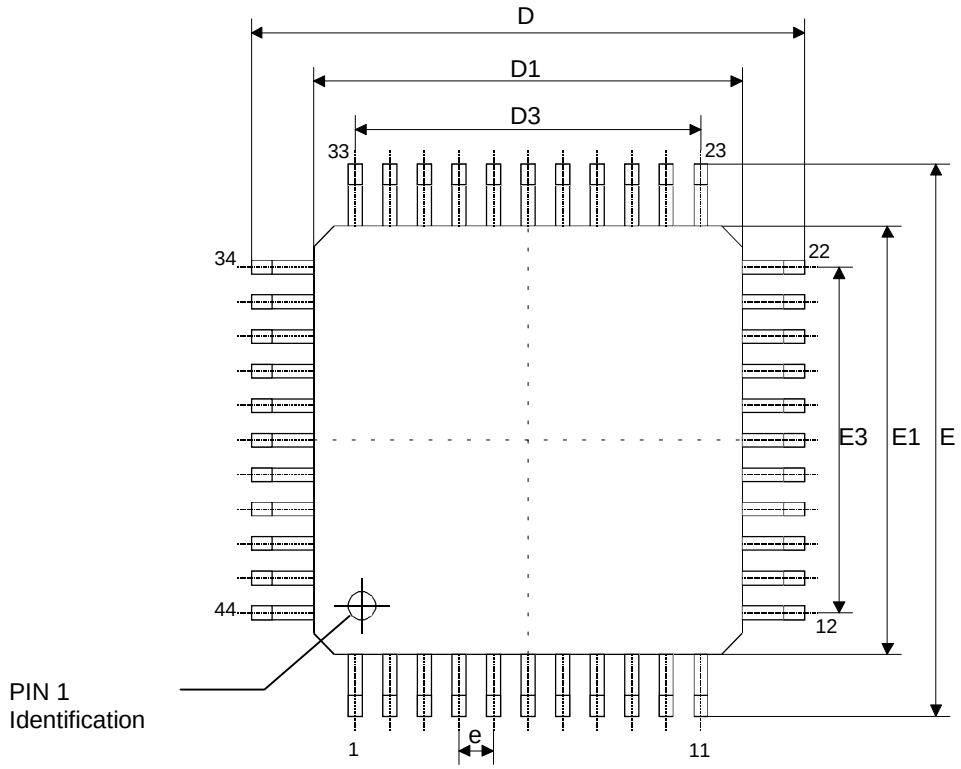


19.6 Pinout



19.7 Package Outline (TQFP 44)





	A	A1	A2	B	c	D	D1	D3	e	E	E1	E3	L	L1	K
Min		0.05	1.35	0.30	0.09								0.45		0°
Typ			1.40	0.37		12.00	10.00	8.00	0.80	12.00	10.00	8.00	0.60	1.00	3.5°
Max	1.60	0.15	1.45	0.45	0.20								0.75		7°

Table 19-3: Package Outline Dimensions in mm and degrees

20 EMI Considerations

As general rule, all tracks should be kept as short as possible to avoid capacitive loads at driver outputs. All components should be placed as close to the OS8104 as possible. The most critical signals with respect to EMI are:

- RMCK
- TX
- RX
- SX0..3, SR0..3
- SCK

The reduction of EMI is most effective, if the current is minimized at the signals' source. For lowering radiation, the best way is to use serial resistors and ferrite beads. :

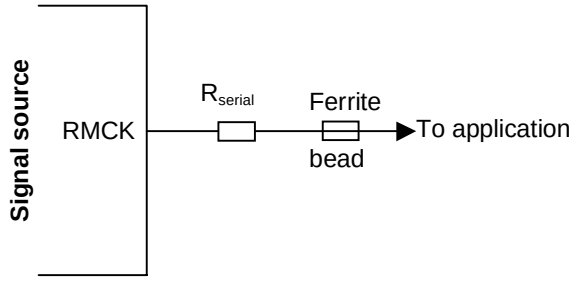


Figure 20-1: EMI reducing circuit

The serial resistor must be placed as close as possible to the output pin to keep track capacitance for the output driver low. The serial resistor as well as the ferrite bead reduce the current through the critical wires, which reduces radiation. Since the value of the resistor depends directly on the application (capacitive load, slew rate etc.), no detailed values can be given. It is advisable to use resistor values between 100Ω and 560Ω.

Since the main goal is to reduce current, the approach using a parallel capacitor as shown below is not suitable for EMI reduction purpose. The capacitor increases the current flowing through the chip and the power lines, which causes additional radiation:

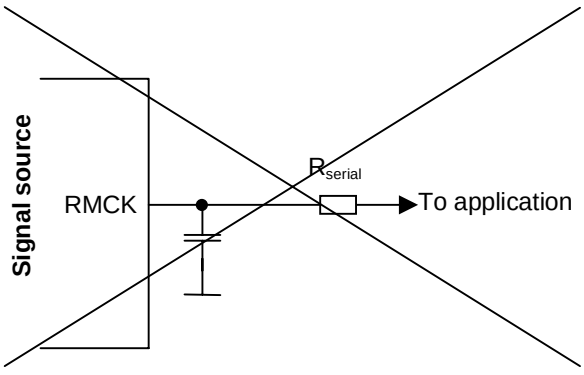


Figure 20-2: Not suitable approach for reducing EMI

Appendix A. Typical Applications

A.1 Power Supplies and Analog Components

The following Figure illustrates the standard power arrangement for the OS8104. The 0.1 µF (±10 %) capacitors must be of ceramic type and should be placed as close as possible to the OS8104. The 10 µF capacitors must have a low Equivalent Series Resistance (ESR).

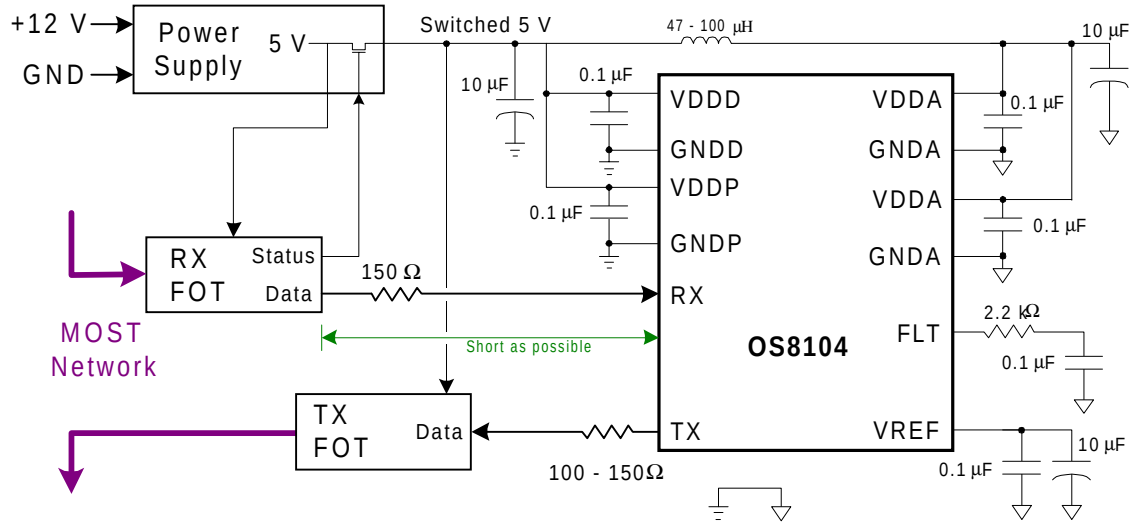


Figure A-1: Typical Power Supply Connection Diagram

The *FLT* and *VREF* analog components should be placed as close as possible to the *GNDA* pins to minimize loop currents. The analog and digital grounds, if separated, should be tied together at only one point on the board.

The distance between the receiver FOT unit and the OS8104 should be as short as possible to minimize capacitance on the Data line. This will shorten transition times out of the FOT thereby minimizing pulse width distortion and jitter.

The series resistor in the TX path is for series-termination and should be as close as possible to the OS8104. The chosen value is based on the board layout and should match the line impedance. This resistor will help minimize reflections and lower EMI.

In the example above, the receive FOT controls the power to the node. All power supplies, except the receive FOT are on the *Switched 5 V* circuit. When there is no light at the RX FOT input, the Status output turns off the switched power supply to minimize the power drain. Having the receive FOT control the node's power complies with the MOST Specification; whereas, using the Zero-Power mode of the OS8104 does not (and uses more power).

When light is initially detected at the FOT, the FOT's Status output turns on the Switched 5 Volt supply thereby powering up the node. The series resistor on the Data output provides current limiting for the short period of time between the receive FOT driving data out and the Power Supply switch-on time.

A.2 Serial modes

A.2.1 Control Port (CP) in I²C mode

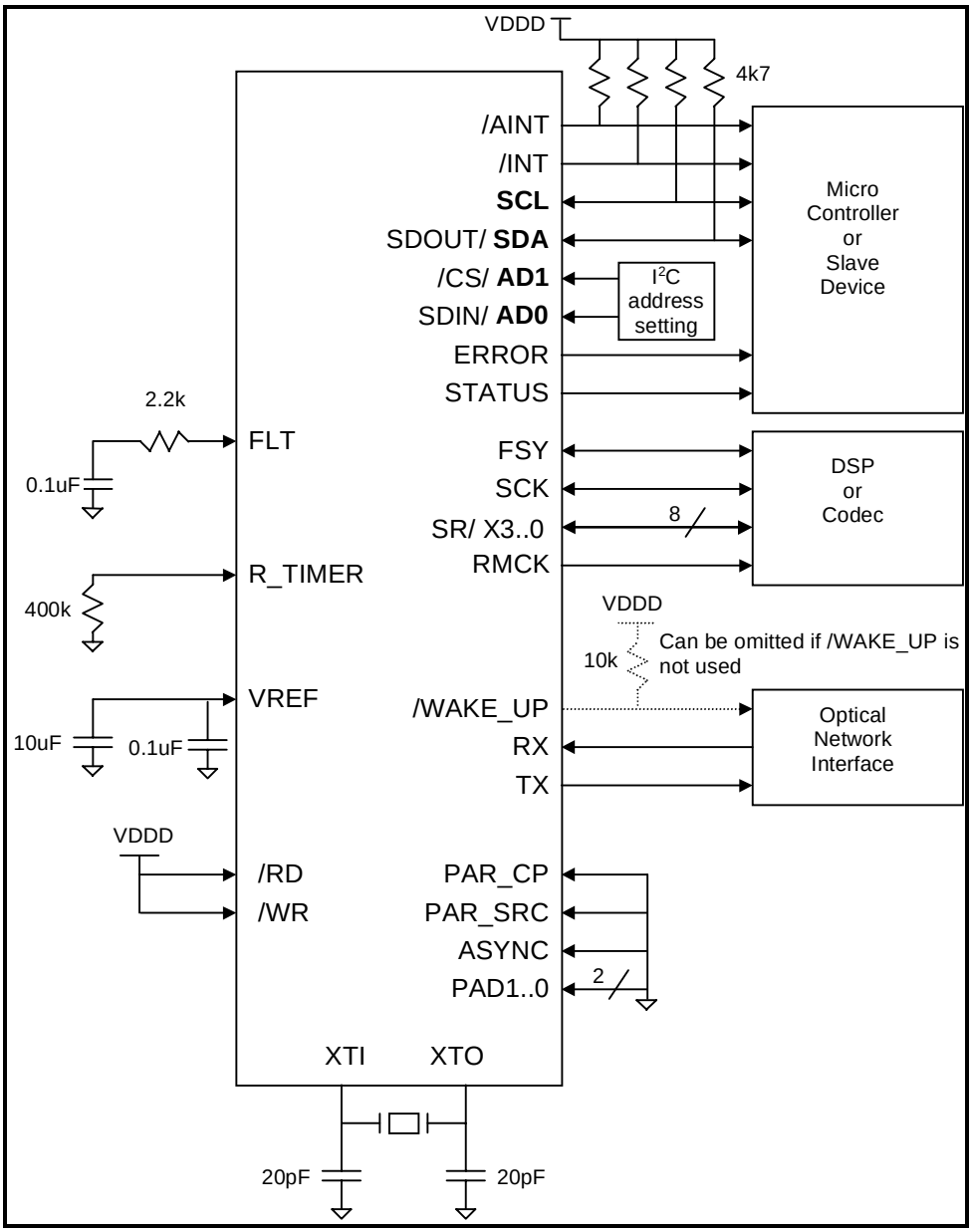


Figure A-2: Typical Application – Source Port in Serial Mode, Control Port in I²C Mode

A.2.2 Control Port (CP) in SPI mode

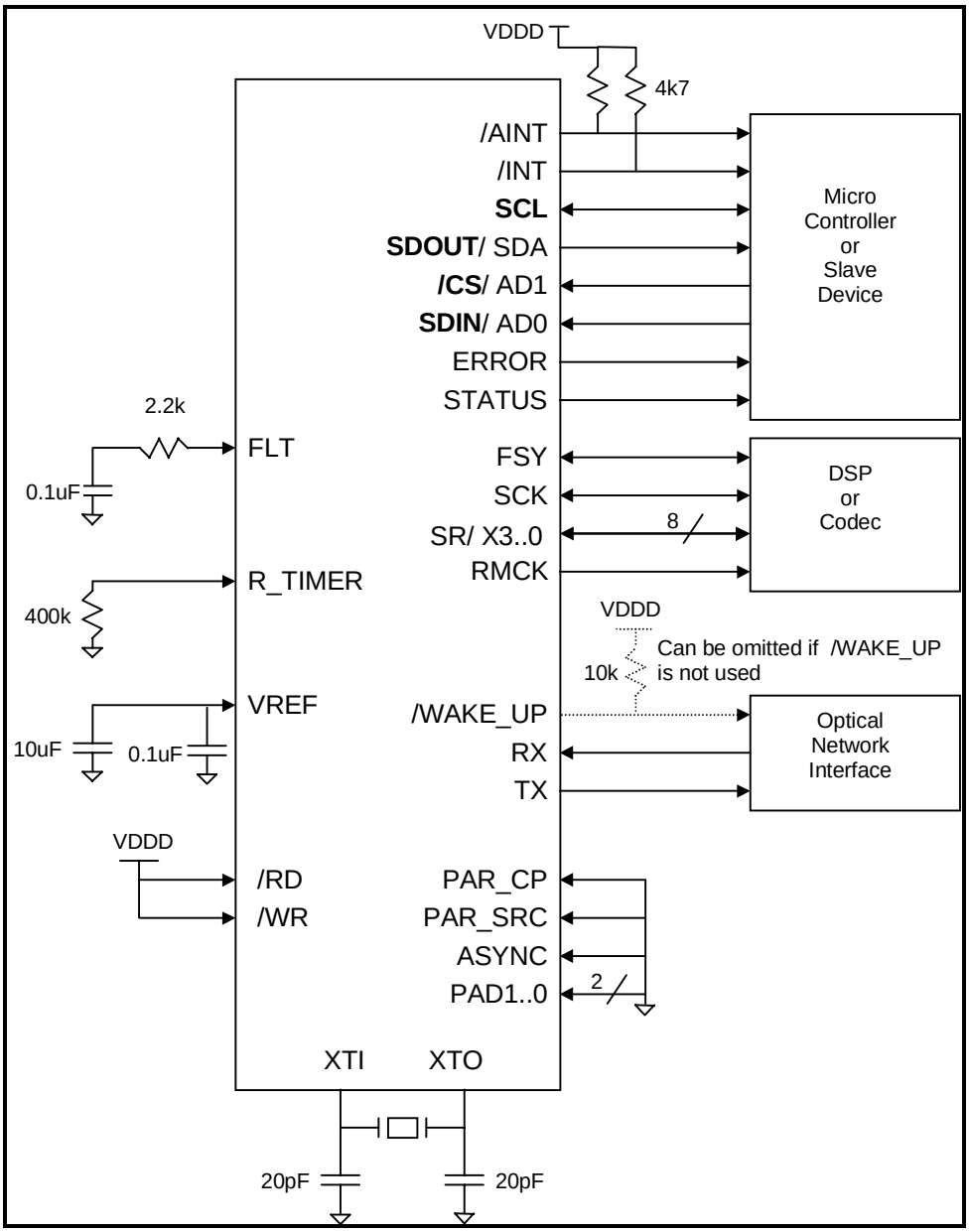


Figure A-3: Typical Application –Source Port in Serial Mode, Control Port in SPI Mode

A.3 Parallel modes

A.3.1 SP In Parallel, CP in Serial (I^2C)

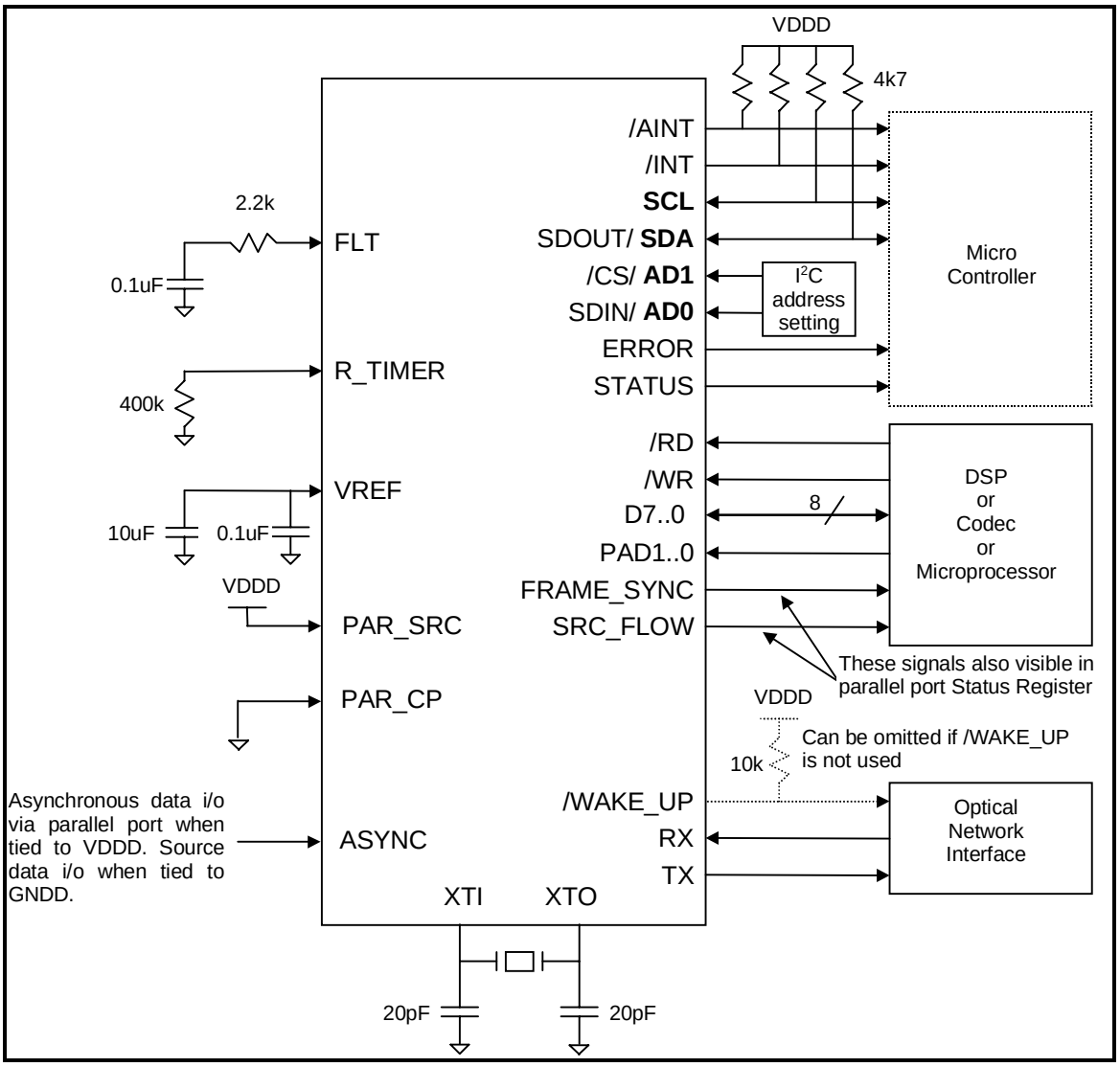


Figure A-4: Typical application – Source Port in Parallel, Control Port in Serial I^2C mode.

A.3.2 SP and CP In Parallel Mode

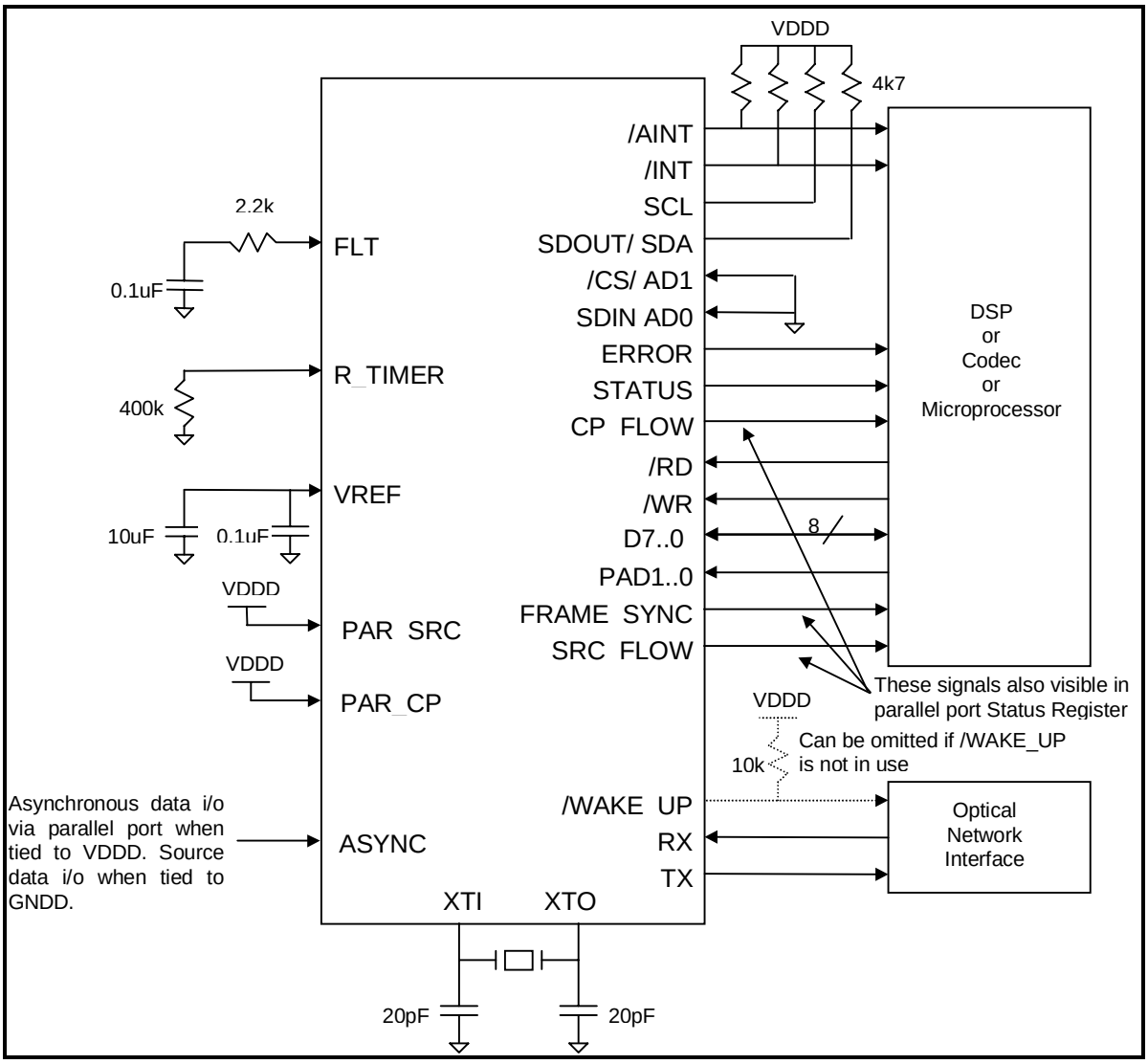


Figure A-5: Typical Application – Source Port and Control Port in Parallel Mode

The pull-up resistors at the pins /AINT, /INT, SCL and SDA/SDOUT are mandatory.

A.4 Stand-Alone Mode

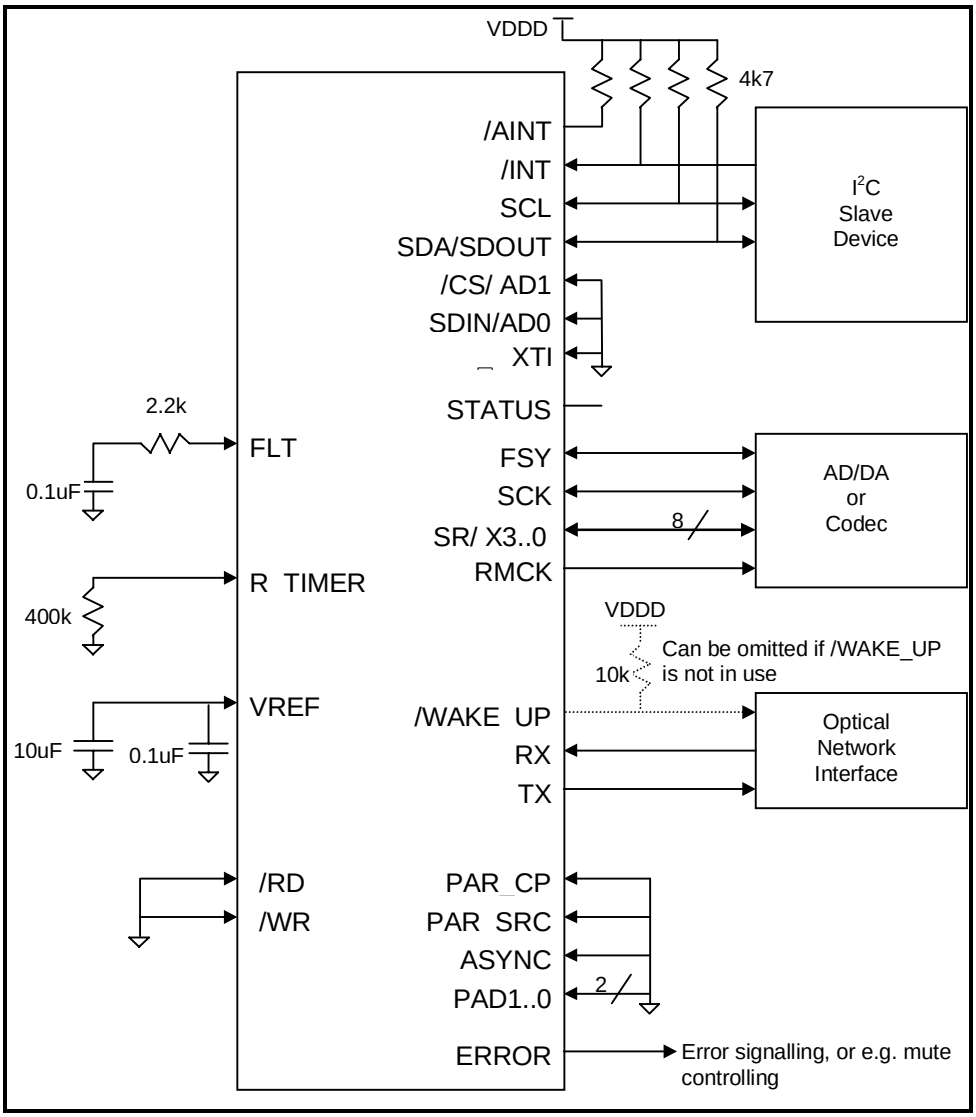


Figure A-6: Typical Application - Stand-Alone Mode

Appendix B. Register Overview

Addr.	Name	B7	B6	B5	B4	B3	B2	B1	B0	Register Name	Page
										Routing Section	
0x00	**bRE0**									RE[0x00]	98
0x01 0x7E	**bRE1** **bRE7E**									RE[0x01] RE[0x7E]	
0x7F	**bRE7F**									RE[0x7F]	101
										HW Control Section	
0x80	**bXCR**	MTR	OE	1	LPW	SAN	SBY	/ABY	/REN	Transceiver Control	35
0x81	**bXSR**		MSL	MXL	ME	ERR		ESL	EXL	Transceiver Status	36
0x82	**bSDC1**	EDG	DEL	POL	I/O	NBR	SPD	/MT	/TCE	Source Data Control 1	44
0x83	**bCM1**	PLD	RD2	RD1	RD0	XTL1	XTL0	MX1	MX0	Clock Manager 1	89
0x85	**bMSGC**	STX	RBE		SAI	RALC	RERR	RMTX	RMRX	Message Control	110
0x86	**bMSGS**	RBS	TXR			ALC	ERR	MTX	MRX	Message Status	111
0x87	**bNPR**									Node Position	39
0x88	**bIE**					IALC	IERR	IMTX	IMRX	Interrupt Enable	95
0x89	**bGA**									Group Address	110
0x8A	**bNAH**									Node Address High	109
0x8B	**bNAL**									Node Address Low	109
0x8C	**bSDC2**	SPR2	SPR1	SPR0	MFSY	TCR1	TCR0	SDR1	SDR0	Source Data Control 2	47
0x8D	**bSDC3**	SIO				SPS				Source Data Control 3	48
0x8E	**bCM2**	/LOK	NAC	ZP	LP					Clock Manager 2	90
0x8F	**bNDR**									Node Delay	39
0x90	**bMPR**									Maximum Position	39
0x91	**bMDR**									Maximum Delay	40
0x96	**bSBC**					SBC3	SBC2	SBC1	SBC0	Synchronous Bandwidth Control	37
0x97	**bXSR2**							INV		Transceiver Status 2	36
	mRCMB									**Rec. Ctrl Msg Buffer**	113
0xA0	**bRTYP**									Receive Message Type	
0xA1	**bRSAH**									Source Address High	
0xA2	**bRSAL**									Source Address Low	
0xA3	**bRCD0**									Receive Control Data 0	
0xA4 … 0xB2	**bRCD1** **…** **bRCD15**									Receive Control Data 1 ... Receive Control Data 15	
0xB3	**bRCD16**									Receive Control Data 16	
										Msg. Transmit Section (1)	
0xBE	**bXTIM**									Transmit Retry Time	112
0xBF	**bXRTY**									Transmit Retries	112
	mXCMB									**Xmit Ctrl Msg Buffer**	114
0xC0	**bXPRI**									Transmit Priority	
0xC1	**bXTYP**									Transmit Message Type	
0xC2	**bXTAH**									Target Address High	
0xC3	**bXTAL**									Target Address Low	
0xC4	**bXCD0**									Xmit Control Data 0	
0xC5 0xD3	**bXCD1** **bXCD15**									Xmit Control Data 1 ... Xmit Control Data 15	
0xD4	**bXCD16**									Xmit Control Data 16	
										Msg. Transmit Section (2)	
0xD5	**bXTS**									Xmit Transfer Status	112

Addr.	Name	B7	B6	B5	B4	B3	B2	B1	B0	Register Name	Page
										Packet Ctrl Section	
0xE2	**bPCTC**				RAF	RAC		RATX	RARX	Packet Control	131
0xE3	**bPCTS**				AF	AC		ATX	ARX	Packet Status	132
0xE6	**bPCMA**								APCM	Parallel-Combined mode Activate	75
0xE8	**bAPAH**									Alternative Packet Address High	129
0xE9	**bAPAL**									Alternative Packet Address Low	130
0xEA	**bPSTX**	ASTX								Packet Start Tx	131
0xEC	**bPLDT**									Packet Length For Data Transmit	130
0xF2	**bPPI**									Packet Priority	130
	mARP									**Packet Receive Sect.**	**132**
0x180	**bARTH**									Received Target Address High	
0x181	**bARTL**									Received Target Address Low	
0x182	**bASAH**									Source Address High	
0x183	**bASAL**									Source Address Low	
0x184	**bARD0**									Async. Receive Data 00	
0x185 0x1B2	**bARD1** **bARD46**									Async. Receive Data 01 Async Receive Data 46	
0x1B3	**bARD47**									Async Receive Data 47	
	mAXP									**Packet Xmit Section**	**133**
0x1C0	**bATAH**									Target Address High	
0x1C1	**bATAL**									Target Address Low	
0x1C2	**bAXD0**									Async Xmit Data 00	
0x1C3 0x1F0	**bAXD1** **bAXD46**									Async Xmit Data 01 Async Xmit Data 46	
0x1F1	**bAXD47**									Async Xmit Data 47	
	mCRA									**Current Resource Allocation**	**125**
0x380	**bCRA0**									Current allocation for frame byte0	
...	**...**									...	
0x3BB	**bCRA59**									Current allocation for frame byte59	

The following are only available in Standalone mode:

Addr	Name	B7	B6	B5	B4	B3	B2	B1	B0	Register Name	
	mSIMB									**Standalone Messaging**	**142**
0xEB	**bSIMC**									Count of bytes to send	
0xEC	**bSITA**									I^2C target address	
0xED	**bSIMA**									I^2C MAP	
0xEE	**bSITD0**									I^2C transfer data byte0	
...	**...**									...	
0xF1	**bSITD3**									I^2C transfer data byte3	
0xF2	**bSITC**									I^2C transfer control byte	

Appendix C. Data Sheet Revision History

Section/Page	Change
DS8104PP2 Changes:	
Text was edited throughout the Data Sheet. The following are the major changes.	
Section 2.5	Expanded to include chip bandwidth numbers for each data transfer method
Section 6.2.3	Added bXSR2 register
Section 8.3	Expanded *Parallel-Combined mode* Section
Section 9.5	Added *Crystal Oscillator* Section
Section 12	Clarified *Routing Synchronous Data* Section
Tables 12-2, 12-4	Added 48Fs Routing Data to Source Port In and Source Port Out Tables.
Section 16.4	Expanded chip version number table
Section 18	Updated all timing diagrams in *Electrical Characteristics* Section.
Section 18.2	Added *Recommended Operating Conditions* Section Fs range changed to 37.9 kHz to 48.1 kHz
Section 18.3	Added Low- and Zero-Power mode current values Added thermal resistances
Section 18.4.1	Added RMCK rise/fall time Added Pulse Width Distortion and Jitter numbers Added configuration pin setup and hold times
Section 18.4.2	Changed t_{drd} to 23 ns (was 10 ns) Added /RD, /WR minimun high time Changed PAD and data hold times Added t_{dhd} hold time maximum Added SRC_FLOW specifications Added CP_FLOW specifications
Section 18.4.3	Changed t_{sxv} to 30 ns (was 25 ns) Added FSY frequency values
Section 18.4.4	Changed t_{sxv} to 30 ns (was 25 ns) Added FSY frequency values Added FSY, SCK rise/fall times
Section 18.4.5	Added t_{sklcsl} and t_{sklcsh} SPI timing parameters Added t_{cdv} SDOUT time

INDEX

/

A

B

C

D

E

F

G

I

L

M

N

O

P

Q

R

S

T

V

X

Z

Notes: