

## 5 21164 Microprocessor Functional Overview

This section provides an overview of 21164 external signals that support the following:

- Clocks
- Bcache interface
- System interface
- Interrupts
- Test modes

See Figure 1 for a block diagram of the 21164.

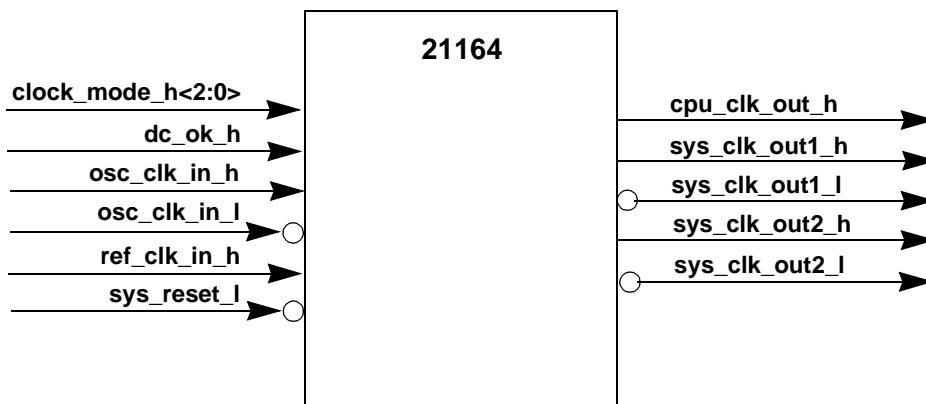
## 5.1 Clocks

The 21164 accepts two clock signal inputs and develops three clock signal outputs:

Signal	Description
<b>Input Clock Signals</b>	
<b>osc_clk_in_h,l</b>	Differential inputs <i>normally</i> driven at the desired internal frequency.
<b>ref_clk_in_h</b>	A system-supplied clock to which the 21164 synchronizes its timing for multiprocessor systems.
<b>Output Clock Signals</b>	
<b>cpu_clk_out_h</b>	A 21164 internal clock that may or may not drive the system clock.
<b>sys_clk_out1_h,l</b>	A clock of programmable speed supplied to the external interface.
<b>sys_clk_out2_h,l</b>	A delayed copy of <b>sys_clk_out1_h,l</b> . The delay is programmable and is an integer number of <b>cpu_clk_out_h</b> periods.

Figure 6 shows the 21164 clock signals.

Figure 6 21164 Clock Signals



## Clocks

### 5.1.1 CPU Clock

The 21164 uses the differential input clock lines **osc\_clk\_in\_h,l** as a source to generate its CPU clock. The input signals **clk\_mode\_h<2:0>** control generation of the CPU clock.

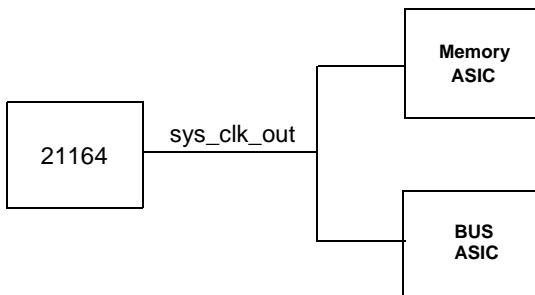
### 5.1.2 System Clock

The CPU clock is divided by a programmable value of 3 to 15 to generate a system clock. The programmable feature allows the system designer maximum flexibility when choosing external logic to interface with the 21164.

The **sys\_clk\_out1\_h,l** signals are delayed by a programmable number of CPU cycles between 0 and 7 to produce **sys\_clk\_out2\_h,l**. The output of the programmable divider is symmetric if the divisor is even. The output is asymmetric if the divisor is odd.

Figure 7 shows the 21164 driving the system clock on a uniprocessor system.

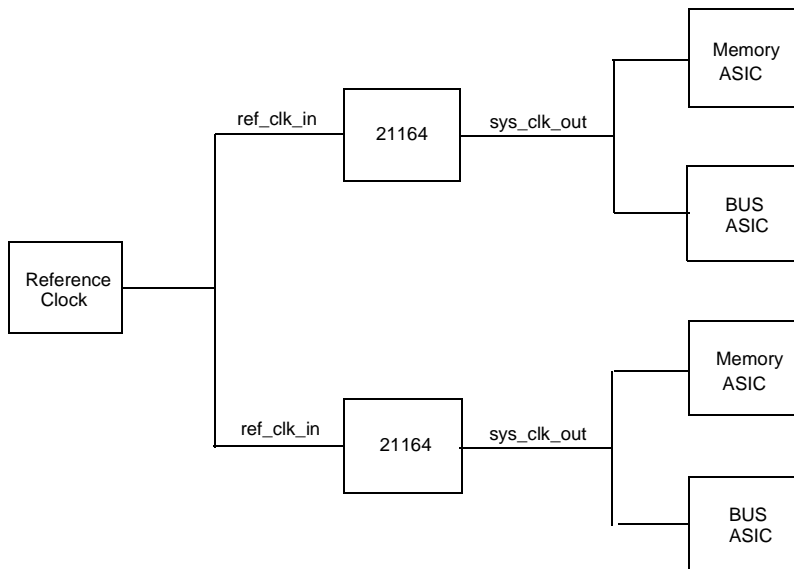
**Figure 7 21164 Uniprocessor Clock**



### 5.1.3 Reference Clock

The 21164 provides a reference clock input so that other CPUs and system devices can be synchronized in multiprocessor systems. If a clock is asserted on signal **ref\_clk\_in\_h**, then the **sys\_clk\_out1\_h,l** signals are synchronized to that reference clock by means of a digital phase-locked loop (DPLL). Figure 8 shows the 21164 synchronized to a system reference clock.

**Figure 8 21164 Reference Clock for Multiprocessor Systems**



## Board-Level Backup Cache Interface

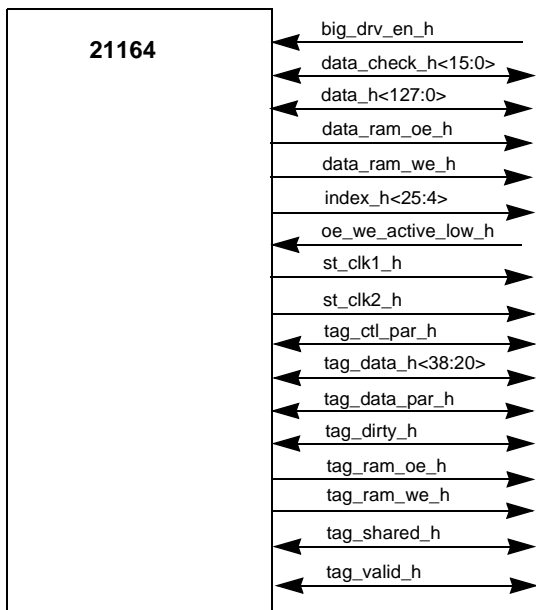
### 5.2 Board-Level Backup Cache Interface

The 21164 includes an interface and control for an optional board-level backup cache (Bcache). This section describes the Bcache interface. The Bcache interface is made up of the following:

- A data bus (which it shares with the system interface)
- Tag and tag control bits for determining hit and coherence
- SRAM output and SRAM write control signals

Figure 9 shows the 21164 system interface signals.

**Figure 9 21164 Bcache Interface Signals**



The Bcache interface is managed by the cache control and bus interface unit (CBU). The Bcache interface is a 128-bit bidirectional data bus. The read and write speed of the Bcache can be programmed independently of each other and independently of the system clock ratio. Optionally, the Bcache can operate in a psuedo-pipeline manner. Internal processor registers are used to program the Bcache timing and to enable wave pipelining. See the *Samsung 21164 Alpha Microprocessor Hardware Reference Manual* for more information.

## Board-Level Backup Cache Interface

The Bcache system supports block sizes of 32 bytes or 64 bytes but it must be set like the secondary cache (Scache). The block size is selected by a mode bit. The Scache is 3-way, set-associative but is a subset of the larger externally implemented, direct-mapped Bcache. In systems with no Bcache, the Scache block size must be set to 64 bytes.

### 5.2.1 Bcache Victim Buffers

The 21164 is designed to support systems with one or more offchip Bcache victim buffers. External victim buffers improve the overall performance of the Bcache. A Bcache victim is generated when the 21164 deallocates a dirty block from the Bcache. Each time a Bcache victim is produced, the 21164 stops reading the Bcache until the system takes the current victim, and then the Bcache operations resume.

### 5.2.2 Cache Coherence Protocol

Cache coherency is a concern for single and multiprocessor 21164-based systems as there may be several caches on a processor module and several more in multiprocessor systems.

The system hardware designer need not be concerned about Icache and Dcache coherency. Coherency of the Icache is a software concern—it is flushed with an IMB (PALcode) instruction. The 21164 maintains coherency between the Dcache and the Scache.

If the system does not have a Bcache, the system designer must create mechanisms in the system interface logic to support cache coherency between the Scache, main memory, and other caches in the system.

If the system has a Bcache, the 21164 maintains cache coherency between the Scache and the Bcache. The Scache is a subset of the Bcache. In this case, the designer must create mechanisms in the system interface logic to support cache coherency between the Bcache, main memory, and other caches in the system.

The following tasks must be performed to maintain cache coherency:

- The CBU in the 21164 maintains coherency in the Dcache and keeps it as a subset of the Scache.
- If an optional Bcache is present, then the 21164 maintains the Scache as a subset of the Bcache. The Scache is set-associative but is kept a subset of the larger externally implemented direct-mapped Bcache.

## System Interface

- System logic must help the 21164 to keep the Bcache coherent with main memory and other caches in the system.
- The Icache is not a subset of any cache and also is not kept coherent with the memory system.

Table 4 describes the Bcache states that determine cache coherence protocol for 21164 systems.

**Table 4 Bcache States for Cache Coherency Protocols**

Valid <sup>1</sup>	Shared <sup>1</sup>	Dirty <sup>1</sup>	State of Cache Line
0	X	X	Not valid.
1	0	0	Valid for read or write operations. This cache line contains the only cached copy of the block and the copy in memory is identical to this line.
1	0	1	Valid for read or write operations. This cache line contains the only cached copy of the block. The contents of the block have been modified more recently than the copy in memory.
1	1	0	Valid for read or write operations. This block may be in another CPU's cache.
1	1	1	Valid for read or write operations. This block may be in another CPU's cache. The contents of the block have been modified more recently than the copy in memory.

<sup>1</sup> The `tag_valid_h`, `tag_shared_h`, and `tag_dirty_h` signals are described in Table 2.

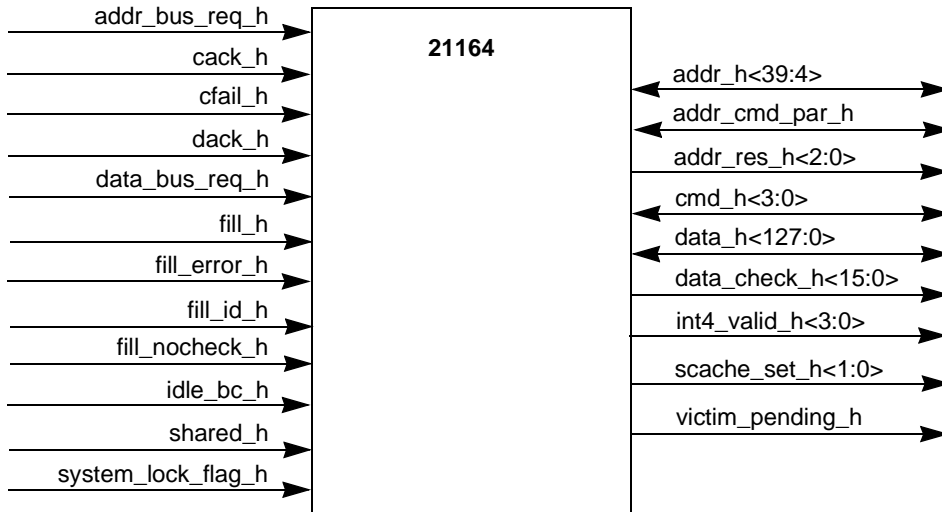
### 5.3 System Interface

The system interface is made up of bidirectional address and command buses, a data bus that it shares with the Bcache interface, and several control signals.

Figure 10 shows the 21164 system interface signals.

## System Interface

**Figure 10 21164 System Interface Signals**



The system interface is under the control of the cache control and bus interface unit (CBU). The system interface is a 128-bit bidirectional data bus. The cycle time of the system interface is programmable to speeds of one-third to one-fifteenth the CPU cycle time. All system interface signals are driven or sampled by the 21164 on the rising edge of `sys_clk_out1_h`.

### 5.3.1 Commands and Addresses

The 21164 can take up to two commands from the system at a time. The bus interface buffer can hold one or two misses and one or two Scache victim addresses at a time. A miss occurs when the 21164 searches its caches but does not find the addressed block. The 21164 can queue two misses to the system. An Scache victim occurs when the 21164 deallocates a dirty block from the Scache.

The system requests the misses, and the victims arbitrate for the Bcache.

- The highest priority for the Bcache is data movement for the system, which includes fill, read dirty data, invalidate, and set shared activities.
- If there are no system requests for the Bcache, then a 21164 command is selected.

## System Interface

Tables 5 and 6 provide a brief description of the commands that the 21164 and the system can drive on the command bus.

**Table 5 21164 Commands for the System**

<b>cmd&lt;3:0&gt;</b>	<b>Command</b>	<b>Meaning</b>
0000	NOP	Nothing.
0001	LOCK	New lock register address.
0010	FETCH	21164 passes a FETCH instruction to the system.
0011	FETCH_M	21164 passes a FETCH_M instruction to the system.
0100	MEMORY BARRIER	MB instruction.
0101	SET DIRTY	Dirty bit set if shared bit is clear.
0110	WRITE BLOCK	Request to write a block.
0111	WRITE BLOCK LOCK	Request to write a block with lock.
1000	READ MISS0	Request for data.
1001	READ MISS1	Request for data.
1010	READ MISS MOD0	Request for data; modify intent.
1011	READ MISS MOD1	Request for data; modify intent.
1100	BCACHE VICTIM	Bcache victim should be removed.
1101	—	Spare.
1110	READ MISS MOD STC0	Request for data, ST $\chi$ _C data.
1111	READ MISS MOD STC1	Request for data, ST $\chi$ _C data.

## Interrupts

**Table 6 System Commands for the 21164**

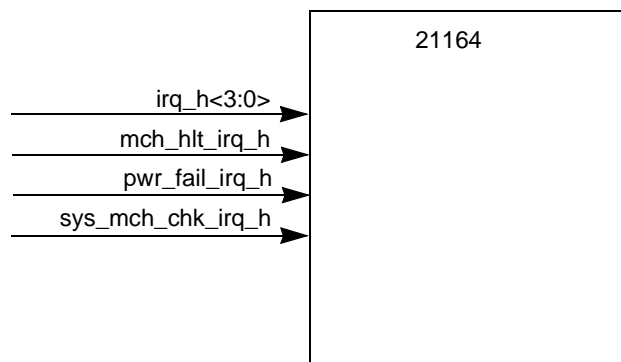
cmd<3:0>	Command	Meaning
0000	NOP	Nothing.
0001	FLUSH	Remove block from caches; return dirty data (flush protocol).
0010	INVALIDATE	Remove the block (write invalidate protocol).
0011	SET SHARED	Block goes to shared state (write invalidate protocol).
0100	READ	Read a block (flush protocol).
0101	READ DIRTY	Read a block; set shared (write invalidate protocol).
0110	READ DIRTY/INV	Read a block; invalidate (write invalidate protocol).

### 5.4 Interrupts

The 21164 has seven interrupt signals that have different uses during initialization and normal operation.

Figure 11 shows the 21164 interrupt signals.

**Figure 11 21164 Interrupt Signals**



## Interrupts

### 5.4.1 Interrupt Signals During Initialization

The 21164 interrupt signals work in tandem with the **sys\_reset\_1** signal to set the values for many of the user-selectable clocking ratios and interface timing parameters. During initialization, the 21164 reads system clock configuration parameters from the interrupt pins.

Table 7 shows the system clock divisor settings. The system clock frequency is determined by dividing the ratio into the CPU clock frequency.

**Table 7 System Clock Divisor**

irq_h<3>	irq_h<2>	irq_h<1>	irq_h<0>	Ratio
Low	Low	High	High	3
Low	High	Low	Low	4
Low	High	Low	High	5
Low	High	High	Low	6
Low	High	High	High	7
High	Low	Low	Low	8
High	Low	Low	High	9
High	Low	High	Low	10
High	Low	High	High	11
High	High	Low	Low	12
High	High	Low	High	13
High	High	High	Low	14
High	High	High	High	15

## Test Modes

Table 8 shows how the three remaining interrupt signals are used to determine the length of the **sys\_clk\_out2** delay. These signals provide flexible timing for system use.

**Table 8 System Clock Delay**

<b>sys_mch_chk_irq_h</b>	<b>pwr_fail_irq_h</b>	<b>mch_hlt_irq_h</b>	<b>Delay Cycles</b>
Low	Low	Low	0
Low	Low	High	1
Low	High	Low	2
Low	High	High	3
High	Low	Low	4
High	Low	High	5
High	High	Low	6
High	High	High	7

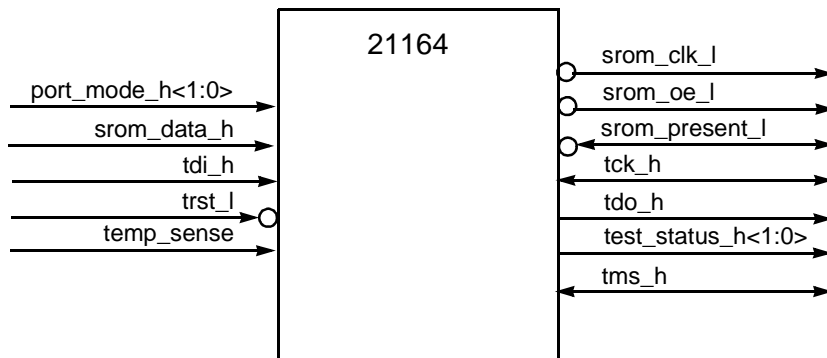
### 5.4.2 Interrupt Signals During Normal Operation

During normal operation, interrupt signals request various interrupts as described in Table 2.

## 5.5 Test Modes

Figure 12 shows the 21164 test signals.

**Figure 12 21164 Test Signals**



## Test Modes

The 21164 test interface port consists of 13 dedicated signals. Table 9 summarizes the 21164 test port signals and their function.

**Table 9 21164 Test Port Pins**

Pin Name	Type	Function
<b>port_mode_h&lt;1&gt;</b>	I	Must be false.
<b>port_mode_h&lt;0&gt;</b>	I	Must be false.
<b>srom_present_l</b>	I	Tied low if serial ROMs (SROMs) are present in system.
<b>srom_data_h/Rx</b>	I	Receives SROM or serial terminal data.
<b>srom_clk_h/Tx</b>	O	Supplies clock to SROMs or transmits serial terminal data.
<b>srom_oe_l</b>	O	SROM enable.
<b>tdi_h</b>	I	IEEE 1149.1 TDI port.
<b>tdo_h</b>	O	IEEE 1149.1 TDO port.
<b>tms_h</b>	I	IEEE 1149.1 TMS port.
<b>tck_h</b>	I	IEEE 1149.1 TCK port.
<b>trst_l</b>	I	IEEE 1149.1 optional TRST port.
<b>test_status_h&lt;0&gt;</b>	O	Indicates Icache BiSt status.
<b>test_status_h&lt;1&gt;</b>	O	Outputs an IPR-written value and timeout reset.

### 5.5.1 Normal Test Interface Mode

The test port is in the default or normal test interface mode when the **port\_mode\_h<1:0>** signals are tied to 00. In this mode, the test port supports the following:

- Serial ROM interface port
- Serial diagnostic terminal interface port
- IEEE 1149.1 test access port

### 5.5.2 Serial ROM Interface Port

The following signals make up the serial ROM (SROM) interface:

**srom\_present\_l**  
**srom\_data\_h**

## Test Modes

### **srom\_oe\_l** **srom\_clk\_h**

During system reset, the 21164 samples the **srom\_present\_l** signal for the presence of SROM. If no SROMs are detected at reset, then **srom\_present\_l** is deasserted and the SROM load is disabled. The reset sequence clears the Icache valid bits, which causes the first instruction fetch to miss the Icache and seek instructions from offchip memory.

If SROMs are present during setup, then the system performs an SROM load as follows:

1. The **srom\_oe\_l** signal supplies the output enable to the SROM.
2. The **srom\_clk\_h** signal supplies the clock to the ROM that causes it to advance to the next bit. The cycle time of this clock is  $126 \pm$  times the system clock ratio.
3. The **srom\_data\_h** signal reads the SROM data.

### 5.5.3 Serial Terminal Port

After the serial ROM data is loaded into the Icache, the three SROM load signals become parallel I/O pins that can drive a diagnostic terminal such as an RS422.

### 5.5.4 IEEE 1149.1 Test Access Port

The test access port complies with all requirements of the IEEE 1149.1 (JTAG) standard. The following signals make up the test access port:

- **tms\_h**—Test access port select.
- **trst\_l**—Test access port reset.
- **tck\_h**—Test access port clock.
- **tdi\_h** and **tdo\_h**—Input and output for serial boundary-scan, die-ID, bypass, and instruction registers.

### 5.5.5 Test Status Signals

The **test\_status\_h** signals extract test status information from the chip.

- The **test\_status\_h<0>** signal indicates when the Icache built-in self-test (BiSt) fails.
- The **test\_status\_h<1>** signal detects unreparable Icache by indicating more than two failing Icache rows.