# Table of Contents

## Rtc Index

**Generated:** 2025-12-06

## APB RTC Specification - Table of Contents

**Component:** APB Real-Time Clock (RTC) Controller **Version:** 1.0 **Last Updated:** 2025-12-01 **Status:** Production Ready

---

## Document Organization

This specification is organized into five chapters covering all aspects of the APB RTC component:

## Chapter 3: Interfaces

**Location:** `ch03_interfaces/`

- [00_overview.md](#) - Interface summary
- [01_apb_slave.md](#) - APB protocol specification
- [02_interrupt.md](#) - Interrupt output
- [03_system.md](#) - Clock and reset interface

## Chapter 4: Programming Model

**Location:** `ch04_programming/`
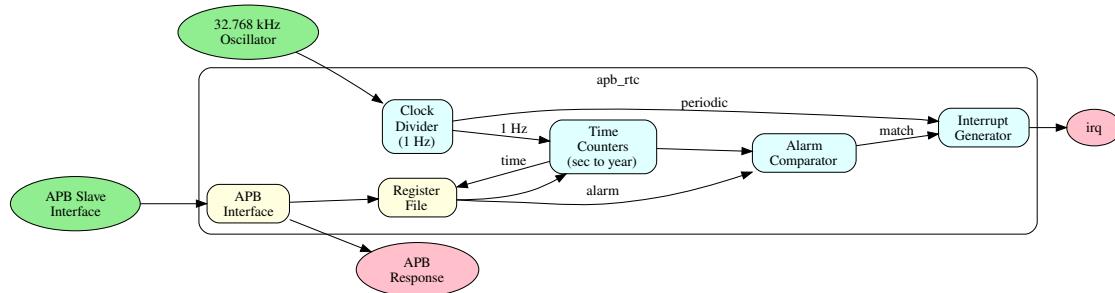
- [00_overview.md](#) - Programming overview
- [01_initialization.md](#) - RTC initialization
- [02_time_operations.md](#) - Reading/setting time
- [03_alarm.md](#) - Alarm configuration
- [04_examples.md](#) - Programming examples

## Chapter 5: Registers

**Location:** `ch05_registers/`

- [01_register_map.md](#) - Complete register map

---

## Block Diagram



*APB RTC Block Diagram*

---

## Quick Navigation

### For Software Developers

- Start with Chapter 4: Programming Model

## Version History

| Version | Date | Author | Changes |
|---------|------|--------|---------|
| 1.0 | 2025-12-01 | RTL Design Sherpa | Initial specification |

**Related Documentation:** - PRD.md - Product Requirements Document

# APB RTC - Overview

## Introduction

The APB RTC is a Real-Time Clock controller with an APB slave interface. It maintains time and date with battery backup support and provides alarm and periodic interrupt capabilities.

## Key Features

### Time Keeping
- Seconds, minutes, hours (12/24-hour mode)
- Day of week, date, month, year
- Century support (2000-2099)
- Leap year calculation
- BCD format storage

### Alarm Function
- Configurable alarm time
- Second, minute, hour, date match
- Daily or specific date alarm

### Interrupt Support
- Alarm match interrupt

- Periodic interrupt (1 Hz)
- Update-ended interrupt
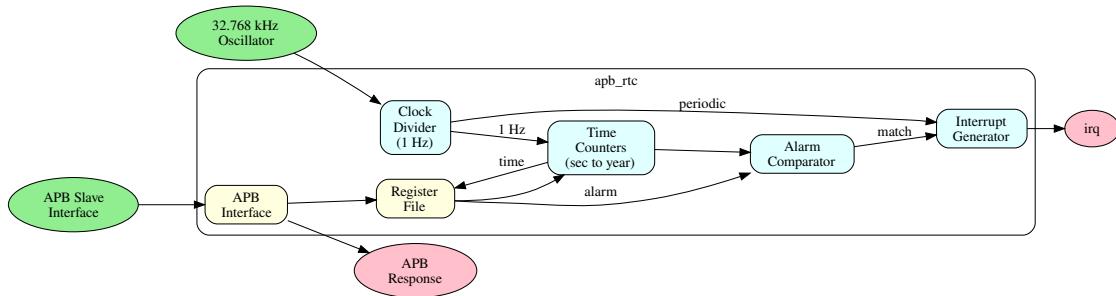
## Power Management
- Low-power 32.768 kHz oscillator
- Battery backup domain support
- RAM retention (optional)

## Applications
- System timekeeping
- Scheduled wake-up
- Event timestamping
- Calendar functions
- Alarm clock

## Block Diagram



*RTC Block Diagram*

## Register Summary

| Offset | Name | Access | Description |
|--------|------|--------|-------------|
| 0x00 | RTC_SECONDS | RW | Seconds (0-59) |
| 0x04 | RTC_MINUTES | RW | Minutes (0-59) |
| 0x08 | RTC_HOURS | RW | Hours (0-23 or 1-12) |
| 0x0C | RTC_DAY | RW | Day of week (1-7) |
| 0x10 | RTC_DATE | RW | Day of month (1-31) |
| 0x14 | RTC_MONTH | RW | Month (1-12) |
| 0x18 | RTC_YEAR | RW | Year (0-99) |

| Offset | Name | Access | Description |
|--------|------|--------|-------------|
| 0x1C | RTC_CENTURY | RW | Century (20-29) |
| 0x20 | RTC_ALARM_SEC | RW | Alarm seconds |
| 0x24 | RTC_ALARM_MIN | RW | Alarm minutes |
| 0x28 | RTC_ALARM_HOUR | RW | Alarm hours |
| 0x2C | RTC_ALARM_DATE | RW | Alarm date |
| 0x30 | RTC_CONTROL | RW | Control register |
| 0x34 | RTC_STATUS | RO/W1C | Status register |
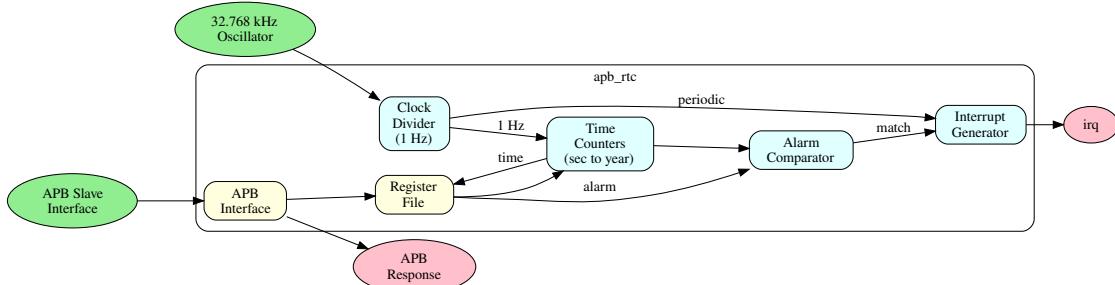
## Parameters

| Parameter | Default | Description |
|-----------|---------|-------------|
| CDC_ENABLE | 0 | Clock domain crossing |

**Next:** - Architecture details

# APB RTC - Architecture

## High-Level Block Diagram



*RTC Architecture*

## Module Hierarchy

```
apb_rtc (Top Level)
+-- apb_slave
```
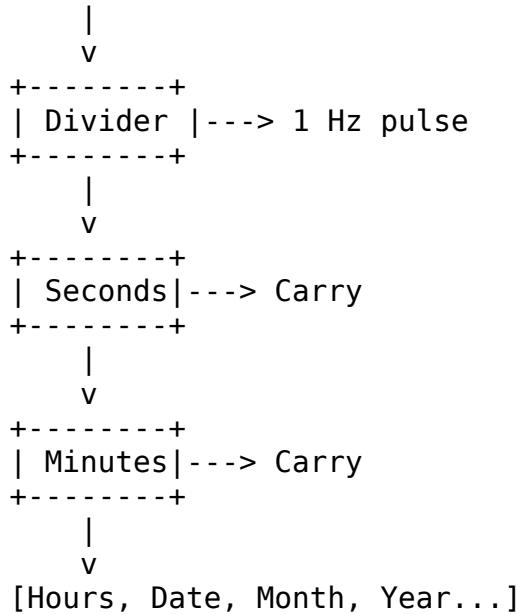
```
+-- rtc_config_regs (Register Wrapper)
|    +-- rtc_regs (PeakRDL Generated)
|
+-- rtc_core
     +-- Time Counter (seconds to century)
     +-- Alarm Comparator
     +-- Interrupt Generator
     +-- BCD Logic
```

## Data Flow
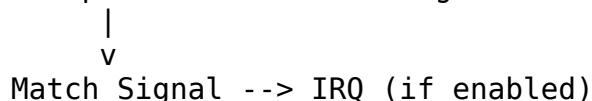
### Time Update Flow

```
32.768 kHz Clock
      |
      v
 +--------+
 | Divider |---> 1 Hz pulse
 +--------+
      |
      v
 +--------+
 | Seconds|---> Carry
 +--------+
      |
      v
 +--------+
 | Minutes|---> Carry
 +--------+
      |
      v
 [Hours, Date, Month, Year...]
```

### Alarm Match Flow

```
Time Registers --> Comparator <-- Alarm Registers
                        |
                        v
              Match Signal --> IRQ (if enabled)
```

## Clock Domains

- APB domain (pclk): Register access
- RTC domain (32.768 kHz): Time counting
- CDC when clocks are asynchronous

---

# APB RTC - Register Map

## Register Summary

| Offset | Name | Access | Reset | Description |
| --- | --- | --- | --- | --- |
| 0x00 | RTC_SECONDS | RW | 0x00 | Seconds (BCD 0-59) |
| 0x04 | RTC_MINUTES | RW | 0x00 | Minutes (BCD 0-59) |
| 0x08 | RTC_HOURS | RW | 0x00 | Hours (BCD 0-23/1-12) |
| 0x0C | RTC_DAY | RW | 0x01 | Day of week (1-7) |
| 0x10 | RTC_DATE | RW | 0x01 | Day of month (BCD 1-31) |
| 0x14 | RTC_MONTH | RW | 0x01 | Month (BCD 1-12) |
| 0x18 | RTC_YEAR | RW | 0x00 | Year (BCD 0-99) |
| 0x1C | RTC_CENTURY | RW | 0x20 | Century (BCD 20-29) |
| 0x20 | RTC_ALM_SEC | RW | 0x00 | Alarm seconds |
| 0x24 | RTC_ALM_MIN | RW | 0x00 | Alarm minutes |
| 0x28 | RTC_ALM_HOUR | RW | 0x00 | Alarm hours |
| 0x2C | RTC_ALM_DATE | RW | 0x00 | Alarm date |
| 0x30 | RTC_CONTROL | RW | 0x00 | Control |
| 0x34 | RTC_STATUS | RO/W1C | 0x00 | Status |

## RTC_CONTROL (0x30)

| Bit | Name | Access | Description |
| --- | --- | --- | --- |
| 0 | RTC_EN | RW | RTC enable |
| 1 | ALM_EN | RW | Alarm enable |
| 2 | PIE | RW | Periodic interrupt enable |
| 3 | AIE | RW | Alarm interrupt enable |
| 4 | UIE | RW | Update interrupt enable |
| 5 | HR24 | RW | 24-hour mode (0=12hr, 1=24hr) |
| 7:6 | Reserved | RO | Reserved |

## RTC_STATUS (0x34)

| Bit | Name | Access | Description |
| --- | --- | --- | --- |
| 0 | UIP | RO | Update in progress |
| 1 | PF | W1C | Periodic flag |
| 2 | AF | W1C | Alarm flag |
| 3 | UF | W1C | Update flag |
| 4 | IRQF | RO | IRQ flag (PF |
| 7:5 | Reserved | RO | Reserved |

## BCD Format

Time/date values stored in BCD: - Seconds: 0x00-0x59 - Minutes: 0x00-0x59 - Hours (24hr): 0x00-0x23 - Hours (12hr): 0x01-0x12 + bit 7 for PM - Date: 0x01-0x31 - Month: 0x01-0x12 - Year: 0x00-0x99

# Retro Legacy Blocks - Product Requirements Document

**Component:** Retro Legacy Blocks (RLB) - Production-Quality Legacy Peripherals
**Version:** 1.0 **Status:** 🟢 Active Development - HPET Production Ready **Last Updated:** 2025-10-29

---

## 1. Overview

### 1.1 Purpose

The Retro Legacy Blocks (RLB) component provides production-quality implementations of legacy peripheral blocks based on proven peripheral designs. These blocks are designed to be reusable, well-tested, and suitable for both FPGA and ASIC implementation.

### 1.2 Design Philosophy

**"Retro" - Proven Architectures:** - Implements time-tested peripheral designs from successful platforms - Focuses on simplicity, reliability, and well-understood behavior - Prioritizes production-readiness over experimental features

**"Legacy" - Time-Tested Interfaces:** - Based on proven peripheral interface specifications - Suitable for systems requiring retro-compatible peripheral compatibility - APB-based interface for easy integration

**"Blocks" - Modular Collection:** - Each peripheral is independent and self-contained - Clear separation between different blocks (rtl/hpet/, rtl/gpio/, etc.) - Can be used individually or wrapped into integrated subsystem

### 1.3 Target Applications

- Retro-compatible platform compatibility layers
- Embedded systems requiring legacy peripheral interfaces
- FPGA-based system emulation
- Educational platforms demonstrating classic peripheral designs
- Mixed-vintage SoC integration (modern + legacy interfaces)

# 2. Implemented Blocks

## 2.1 HPET - High Precision Event Timer

**Status:** √ Production Ready (5/6 configurations 100% passing) **RTL Location:** rtl/hpet/ **Documentation:** docs/hpet_spec/
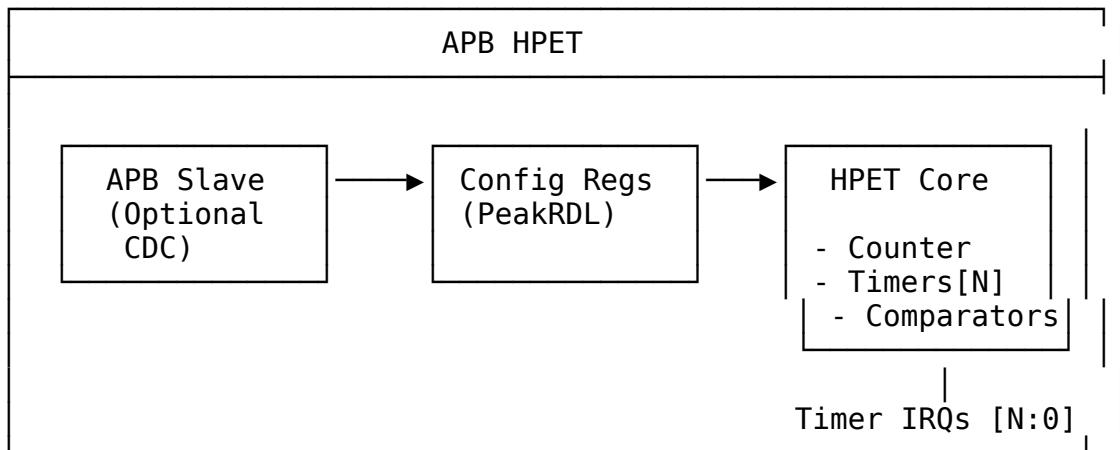
**Key Features:** - Configurable timer count: 2, 3, or 8 independent timers - 64-bit main counter for high-resolution timestamps - 64-bit comparators per timer - Operating modes: One-shot and periodic - Clock domain crossing: Optional CDC for timer/APB clock independence - APB4 interface: Standard AMBA APB protocol - PeakRDL integration: Register map generated from SystemRDL specification

**Applications:** - System tick generation - Real-time OS scheduling - Precise event timing - Performance profiling - Watchdog timers - Multi-rate timing domains

**Test Coverage:** - 6 configurations tested (2/3/8 timers, CDC on/off) - 5/6 configurations at 100% pass rate - 1 configuration at 92% (minor stress test timeout) - 12 test cases per configuration (basic/medium/full)

**Architecture:**

```
┌─────────────────────────────────────────────────────────────┐ |
│                          APB HPET                           │ |
├─────────────────────────────────────────────────────────────┤ |
│                                                                  |
│  ┌──────────────┐     ┌──────────────┐     ┌──────────────┐ |
│  │ APB Slave    │ ──> │ Config Regs  │ ──> │ HPET Core    │ |
│  │ (Optional    │     │ (PeakRDL)    │     │              │ |
│  │  CDC)        │     │              │     │ - Counter    │ |
│  └──────────────┘     └──────────────┘     │ - Timers[N]  │ |
│                                            │ ┌──────────────┐ |
│                                            │ │ - Comparators│ |
│                                            └─┴──────────────┘ |
│                                                   │            |
│                                            Timer IRQs [N:0]   |
└─────────────────────────────────────────────────────────────┘
```

**Design Highlights:** - Reset macro standardization (FPGA-friendly) - Per-timer data buses prevent corruption - Edge-triggered register write strobes (not level) - W1C status register for interrupt clearing - Optional asynchronous clock domains with handshake CDC

**See:** docs/hpet_spec/hpet_index.md for complete HPET specification

# 3. Planned Blocks

## 3.1 8259 - Programmable Interrupt Controller (PIC)

**Status:** 📋 Planned **Priority:** High **Effort:** 6-8 weeks **Address:** `0x4000_1000` - `0x4000_1FFF` (4KB window)

**Planned Features:** - Intel 8259A-compatible register interface - 8 interrupt request (IRQ) inputs - Cascadable (master/slave configuration) - Priority resolver (fixed and rotating priority) - Edge and level triggered modes - Interrupt mask register - End-of-Interrupt (EOI) handling - APB register interface

**Applications:** - Legacy interrupt management - PC-compatible systems - Hardware interrupt aggregation - Priority-based interrupt handling - Cascaded multi-level interrupt systems

## 3.2 8254 - Programmable Interval Timer (PIT)

**Status:** 📋 Planned **Priority:** High **Effort:** 4-5 weeks **Address:** `0x4000_2000` - `0x4000_2FFF` (4KB window)

**Planned Features:** - Intel 8254-compatible register interface - 3 independent 16-bit counters - 6 programmable counter modes - Binary and BCD counting - Read-back command - Configurable clock input - Interrupt/output generation per counter - APB register interface

**Counter Modes:** - Mode 0: Interrupt on terminal count - Mode 1: Hardware retriggerable one-shot - Mode 2: Rate generator - Mode 3: Square wave mode - Mode 4: Software triggered strobe - Mode 5: Hardware triggered strobe

**Applications:** - System tick generation - Periodic timer interrupts - Square wave generation - Event counting - Legacy PC timer compatibility

## 3.3 GPIO - General Purpose I/O

**Status:** 📋 Planned **Priority:** Medium **Effort:** 4-6 weeks **Address:** TBD (not in primary ILB address map)

**Planned Features:** - Configurable pin count (8, 16, 32 pins) - Per-pin direction control (input/output/bidirectional) - Input debouncing logic - Interrupt generation (rising/falling/both edges, level) - Output drive strength configuration - Pull-up/pull-down control - APB register interface

**Applications:** - LED control - Button inputs - Hardware control signals - Chip-select generation - Status monitoring

### 3.4 RTC - Real-Time Clock

**Status:** 📋 Planned **Priority:** Medium **Effort:** 3-4 weeks **Address:** `0x4000_3000` - `0x4000_3FFF` (4KB window)

**Planned Features:** - 32.768 kHz clock input (typical RTC crystal frequency) - Seconds, minutes, hours, day, month, year tracking - Alarm functionality - Battery backup support (power domain considerations) - 24-hour or 12-hour (AM/PM) mode - Leap year handling - APB register interface

**Applications:** - System time-of-day tracking - Wake-on-alarm functionality - Timestamp generation - Power-aware applications

### 3.5 SMBus Controller

**Status:** 📋 Planned **Priority:** Medium **Effort:** 6-8 weeks **Address:** `0x4000_4000` - `0x4000_4FFF` (4KB window)

**Planned Features:** - SMBus 2.0 compliance - Master and slave modes - Clock stretching support - Packet Error Checking (PEC) - Alert response address - Configurable clock speed - APB register interface

**Applications:** - System management bus communication - Sensor interfaces (temperature, voltage) - EEPROM access - Battery management - Fan control

### 3.6 UART - Universal Asynchronous Receiver/Transmitter

**Status:** 📋 Planned **Priority:** Medium **Effort:** 4-5 weeks **Address:** TBD (not in primary ILB address map)

**Planned Features:** - 16550-compatible register interface - Configurable baud rate generation - 5/6/7/8 data bits - Parity: none, even, odd, mark, space - Stop bits: 1, 1.5, 2 - Hardware flow control (RTS/CTS) - FIFO buffers (16-byte TX/RX) - Interrupt generation

**Applications:** - Debug console - Serial communication - Modem interfaces - Legacy peripheral communication

### 3.7 SPI Controller

**Status:** 📋 Planned **Priority:** Low **Effort:** 5-6 weeks **Address:** TBD (not in primary ILB address map)

**Planned Features:** - Master mode (initially; slave mode future) - Configurable clock polarity and phase (CPOL/CPHA) - Multiple chip selects - Configurable word size (8/16/32 bits) - TX/RX FIFOs - DMA support (future) - APB register interface

**Applications:** - Flash memory access - ADC/DAC interfaces - Display controllers - SD card communication

## 3.8 I2C Controller

**Status:** 📋 Planned **Priority:** Low **Effort:** 5-7 weeks **Address:** TBD (not in primary ILB address map)

**Planned Features:** - I2C standard (100 kHz), fast (400 kHz), fast-plus (1 MHz) modes - Multi-master arbitration - 7-bit and 10-bit addressing - Clock stretching - General call support - APB register interface

**Applications:** - Sensor interfaces - EEPROM access - Multi-chip communication - System configuration

## 3.9 Watchdog Timer

**Status:** 📋 Planned **Priority:** Low **Effort:** 2-3 weeks **Address:** TBD (not in primary ILB address map)

**Planned Features:** - Configurable timeout period - Countdown counter with reload - Reset generation on timeout - Lock mechanism to prevent accidental disable - Interrupt before reset (optional warning) - APB register interface

**Applications:** - System fault recovery - Software hang detection - Periodic system reset - Safety-critical applications

## 3.10 Power Management / ACPI Controller

**Status:** 📋 Planned **Priority:** Medium **Effort:** 8-10 weeks **Address:** `0x4000_5000 - 0x4000_5FFF` (4KB window)

**Planned Features:** - Clock gating control per block - Power domain sequencing - Reset generation and distribution - Wake event handling - Sleep/idle mode control - ACPI-compatible registers - APB register interface

**Applications:** - Low-power system design - Battery-powered devices - Dynamic power management - Thermal management - OS power management interface

### 3.11 IOAPIC - I/O Advanced Programmable Interrupt Controller

**Status:** 📋 Planned **Priority:** Medium **Effort:** 6-8 weeks **Address:** `0x4000_6000` - `0x4000_6FFF` (4KB window)

**Planned Features:** - I/O APIC CSR model (register-based interface) - Multiple interrupt inputs (24+) - Programmable interrupt routing - Edge and level triggered modes - Priority-based arbitration - Interrupt masking per input - APB register interface for configuration

**Applications:** - Advanced interrupt routing - Multi-processor interrupt distribution - Flexible interrupt mapping - Legacy IRQ redirection - PC-compatible systems

### 3.12 Interconnect ID / Version Registers

**Status:** 📋 Planned **Priority:** Low **Effort:** 1-2 weeks **Address:** `0x4000_F000` - `0x4000_FFFF` (4KB window)

**Planned Features:** - Vendor ID register - Device ID register - Revision ID register - Block presence/capability bits - Configuration status registers - Debug/diagnostic registers - APB register interface

**Applications:** - Software block discovery - Version checking - Feature detection - Debug and diagnostics - Platform identification

---

## 4. Integration and Wrapper Goals

### 4.1 Individual Block Integration

Each block is designed to be used standalone:

**Example - HPET Integration:**

```
apb_hpet #(
    .NUM_TIMERS(3),
    .VENDOR_ID(16'h8086),
    .REVISION_ID(16'h0001),
    .CDC_ENABLE(0)
) u_hpet (
    .pclk           (apb_clk),
    .presetn        (apb_rst_n),
    // APB interface
    .paddr          (paddr),
    .psel           (psel_hpet),
```
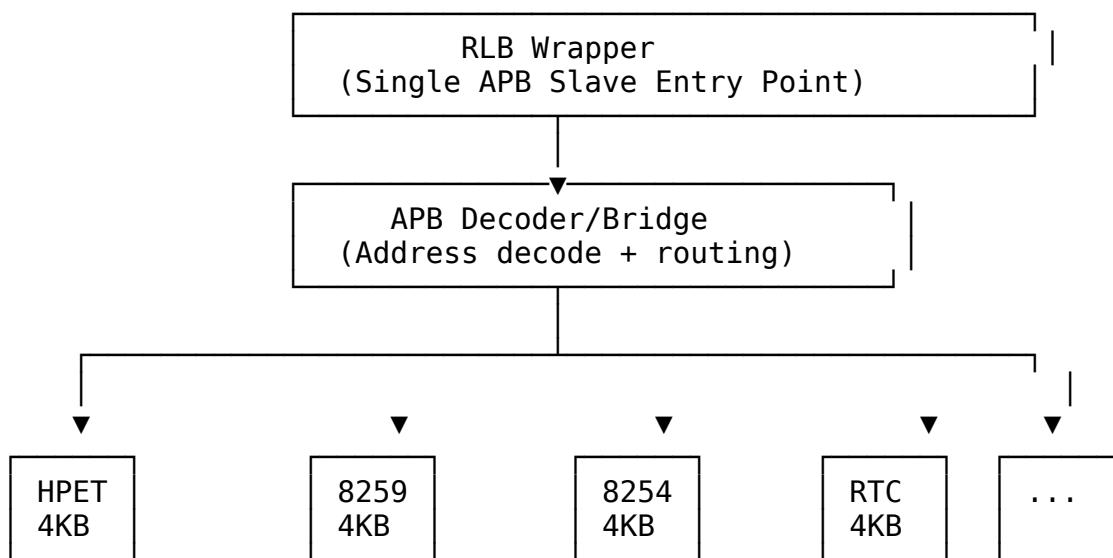
```
    .penable      (penable),
    .pwrite       (pwrite),
    .pwdata       (pwdata),
    .prdata       (prdata_hpet),
    .pready       (pready_hpet),
    .pslverr      (pslverr_hpet),
    // HPET-specific
    .hpet_clk     (timer_clk),
    .hpet_rst_n   (timer_rst_n),
    .timer_irq    (timer_irq[2:0])
);
```

## 4.2 RLB Wrapper Architecture

**Goal:** Create top-level wrapper combining multiple legacy blocks into unified retro-compatible subsystem.

**System Architecture:**

```
            ┌─────────────────────────────────────────┐ │
            │            RLB Wrapper                   │
            │   (Single APB Slave Entry Point)         │
            └─────────────────────────────────────────┘
                              │
            ┌─────────────────▼───────────────────────┐ │
            │         APB Decoder/Bridge               │
            │     (Address decode + routing)           │
            └──────────────────────────────────────────┘
                              │                        │
     ┌──────────┬──────────┬──────────┬──────────┐
     ▼          ▼          ▼          ▼          ▼
 ┌───────┐  ┌───────┐  ┌───────┐  ┌───────┐  ┌───────┐
 │ HPET  │  │ 8259  │  │ 8254  │  │ RTC   │  │  ...  │
 │ 4KB   │  │ 4KB   │  │ 4KB   │  │ 4KB   │  │       │
 └───────┘  └───────┘  └───────┘  └───────┘  └───────┘
```

**Address Map:**

Base address: 0x4000_0000 (1GB region in typical 32-bit system) Window size: 4KB per block (clean power-of-2 decode)

| Address Range | Block | Size | Function |
|---|---|---|---|
| 0x4000_0000 - 0x4000_0FFF | HPET | 4KB | High Precision Event Timer |
| 0x4000_1000 - 0x4000_1FFF | 8259 | 4KB | Programmable Interrupt Controller (PIC) |

| Address Range | Block | Size | Function |
|---|---|---|---|
| 0x4000_2000 - 0x4000_2FFF | 8254 | 4KB | Programmable Interval Timer (PIT) |
| 0x4000_3000 - 0x4000_3FFF | RTC | 4KB | Real-Time Clock |
| 0x4000_4000 - 0x4000_4FFF | SMBus | 4KB | SMBus Host Controller |
| 0x4000_5000 - 0x4000_5FFF | PM/ACPI | 4KB | Power Management / ACPI Registers |
| 0x4000_6000 - 0x4000_6FFF | IOAPIC | 4KB | I/O Advanced PIC (CSR model) |
| 0x4000_7000 - 0x4000_EFFF | *Reserved* | 32KB | Future expansion |
| 0x4000_F000 - 0x4000_FFFF | Interconnect | 4KB | ID/Version/Control registers |
| All other addresses | Error Slave | - | Returns DECERR/SLVERR |

**Decoder Implementation:**

```verilog
// Address decode logic (simplified)
localparam BASE_ADDR = 32'h4000_0000;
localparam BLOCK_SIZE = 12;  // 4KB = 2^12

logic [3:0] block_sel;
assign block_sel = paddr[15:12];  // Extract window number

always_comb begin
    psel_hpet    = (block_sel == 4'h0) & psel;  // 0x4000_0xxx
    psel_pic8259 = (block_sel == 4'h1) & psel;  // 0x4000_1xxx
    psel_pit8254 = (block_sel == 4'h2) & psel;  // 0x4000_2xxx
    psel_rtc     = (block_sel == 4'h3) & psel;  // 0x4000_3xxx
    psel_smbus   = (block_sel == 4'h4) & psel;  // 0x4000_4xxx
    psel_pm      = (block_sel == 4'h5) & psel;  // 0x4000_5xxx
    psel_ioapic  = (block_sel == 4'h6) & psel;  // 0x4000_6xxx
    psel_id      = (block_sel == 4'hF) & psel;  // 0x4000_Fxxx
    psel_error   = !(|{psel_hpet, psel_pic8259, psel_pit8254,
                    psel_rtc, psel_smbus, psel_pm,
                    psel_ioapic, psel_id}) & psel;
end
```

**Interface:** - **Single APB slave port** at base address 0x4000_0000 - **Aggregated interrupt output** combining all block IRQs - **Per-block clock/reset control** for power management - **External I/O signals** (GPIO, UART, I2C/SMBus, etc.) - **Error slave** returns SLVERR for unmapped addresses

**Benefits:** - Simplified system integration (single APB slave) - Consistent 4KB window addressing - Clean power-of-2 address decode - Easy expansion (32KB reserved space) - Single verification target - Drop-in retro-compatible peripheral subsystem

---

## 5. Design Standards

### 5.1 Reset Handling

**MANDATORY:** All blocks must use standardized reset macros from rtl/amba/includes/reset_defs.svh

**Pattern:**

```
`include "reset_defs.svh"

`ALWAYS_FF_RST(clk, rst_n,
    if (`RST_ASSERTED(rst_n)) begin
        r_state <= IDLE;
        r_counter <= '0;
    end else begin
        r_state <= w_next_state;
        r_counter <= r_counter + 1'b1;
    end
)
```

**Why:** - FPGA-friendly reset inference - Consistent synthesis behavior - Single-point reset polarity control - Better timing closure

### 5.2 Register Generation

**Preferred:** Use PeakRDL for register map generation

**Process:** 1. Define registers in SystemRDL (.rdl file) 2. Generate RTL using PeakRDL regblock 3. Create wrapper module connecting registers to core logic 4. Use edge detection for write strobes (not level)

**Benefits:** - Consistent register interface - Auto-generated documentation - Reduced manual RTL errors - Easy register map changes

## 5.3 Testbench Architecture

**MANDATORY:** Follow project testbench organization pattern

**Structure:**

```
dv/
├── tbclasses/{block}/          # Block-specific TB classes
│   ├── {block}_tb.py           # Main testbench
│   ├── {block}_tests_basic.py  # Basic test suite
│   ├── {block}_tests_medium.py # Medium test suite
│   └── {block}_tests_full.py   # Full test suite
└── tests/{block}/              # Test runners
    ├── test_apb_{block}.py     # Pytest wrapper
    └── conftest.py             # Pytest configuration
```

**Import Pattern:**

```python
# Always import from PROJECT AREA
from projects.components.retro_legacy_blocks.dv.tbclasses.{block}.{block}_tb import {Block}TB
```

**Test Levels:** - **Basic:** Core functionality (register access, basic operation) - **Medium:** Extended features (modes, configurations, edge cases) - **Full:** Stress testing, CDC variants, corner cases

**Target:** 100% pass rate at all levels

## 5.4 FPGA Synthesis Attributes

**MANDATORY:** Add FPGA synthesis hints for memory arrays

```systemverilog
`ifdef XILINX
    (* ram_style = "auto" *)
`elsif INTEL
    /* synthesis ramstyle = "AUTO" */
`endif
logic [DATA_WIDTH-1:0] mem [DEPTH];
```

## 5.5 Documentation Requirements

Each block must have: - RTL comments (inline) - Register map specification - Block-level specification in docs/{block}_spec/ - Integration guide - Test plan and results

# 6. Quality Metrics

## 6.1 Production Readiness Criteria

A block is considered "Production Ready" when:

- √ All basic tests pass 100%
- √ All medium tests pass 100%
- √ All full tests pass ≥95%
- √ Complete register map specification
- √ RTL lint clean (Verilator)
- √ Reset macros used throughout
- √ FPGA synthesis attributes applied
- √ Integration guide written
- √ Known issues documented

## 6.2 Current Status

| Block | Priority | Status | Test Pass Rate | Documentation | Production Ready |
|-------|----------|--------|----------------|---------------|------------------|
| HPET | High | √ Complete | 5/6 at 100%, 1/6 at 92% | √ Complete | √ Yes |
| 8259 PIC | High | 📋 Planned | N/A | N/A | ✗ No |
| 8254 PIT | High | 📋 Planned | N/A | N/A | ✗ No |
| GPIO | Medium | 📋 Planned | N/A | N/A | ✗ No |
| RTC | Medium | 📋 Planned | N/A | N/A | ✗ No |
| SMBus | Medium | 📋 Planned | N/A | N/A | ✗ No |
| PM/ | Medium | 📋 | N/A | N/A | ✗ No |

| Block | Priority | Status | Test Pass Rate | Documentation | Production Ready |
|---|---|---|---|---|---|
| ACPI | | Planned | | | |
| IOAPIC | Medium | 📋 Planned | N/A | N/A | ✗ No |
| UART | Medium | 📋 Planned | N/A | N/A | ✗ No |
| SPI | Low | 📋 Planned | N/A | N/A | ✗ No |
| I2C | Low | 📋 Planned | N/A | N/A | ✗ No |
| Watchdog | Low | 📋 Planned | N/A | N/A | ✗ No |
| Interconnect | Low | 📋 Planned | N/A | N/A | ✗ No |

## 7. Development Roadmap

### 7.1 Phase 1: Foundation (Complete ✓)

- √ HPET implementation
- √ Directory structure for multiple blocks
- √ Testbench architecture established
- √ Documentation templates
- √ Build and test infrastructure

### 7.2 Phase 2: Core Peripherals (Next 6-9 Months)

**Q1 2026 (High Priority):** - 8259 PIC (6-8 weeks) - Interrupt controller - 8254 PIT (4-5 weeks) - Interval timer - RTC (3-4 weeks) - Real-time clock

**Q2 2026 (Medium Priority):** - GPIO Controller (4-6 weeks) - SMBus Controller (6-8 weeks) - PM/ACPI Controller (8-10 weeks)

**Q3 2026:** - UART (4-5 weeks) - IOAPIC (6-8 weeks)

## 7.3 Phase 3: Advanced Peripherals (9-15 Months)

**Q4 2026:** - SPI Controller (5-6 weeks) - I2C Controller (5-7 weeks) - Watchdog Timer (2-3 weeks)

**Q1 2027:** - Interconnect ID/Version Registers (1-2 weeks) - ILB Wrapper integration starts

## 7.4 Phase 4: System Integration (15+ Months)

**Q2-Q4 2027:** - Complete ILB wrapper with all blocks - System-level integration examples - Performance characterization - FPGA reference designs - Application notes - Software driver examples

---

# 8. References

## 8.1 External Standards

**Peripheral Specifications:** - ACPI HPET Specification 1.0a - SMBus Specification Version 2.0 - 16550 UART Datasheet - I2C Specification (NXP) - SPI Protocol Specification

**Bus Protocols:** - AMBA APB Protocol Specification (ARM) - AMBA 3 APB Protocol v1.0

## 8.2 Internal Documentation

- `/CLAUDE.md` - Repository AI guide
- `/PRD.md` - Master repository requirements
- `projects/components/retro_legacy_blocks/CLAUDE.md` - Component AI guide
- `projects/components/retro_legacy_blocks/README.md` - Component overview
- `projects/components/retro_legacy_blocks/TASKS.md` - Task tracking

### 8.3 Block-Specific Documentation

**HPET:** - `docs/hpet_spec/hpet_index.md` - HPET specification - `docs/IMPLEMENTATION_STATUS.md` - HPET test results - `known_issues/` - HPET issue tracking

---

## 9. Success Criteria

### 9.1 Individual Block Success

Each block must: - Pass all basic/medium tests at 100% - Pass full tests at ≥95% - Have complete register map specification - Include integration guide with examples - Be lint-clean (Verilator) - Use reset macros throughout - Include FPGA synthesis attributes

### 9.2 Collection Success

The retro_legacy_blocks component is successful when: - At least 6 blocks production-ready (HPET + 5 high/medium priority blocks) - All blocks follow consistent architecture (reset macros, PeakRDL, APB interface) - RLB wrapper integrates all blocks seamlessly with clean 4KB addressing - System-level integration example provided - Complete documentation for all blocks - FPGA reference design available - Address map covers all essential retro-compatible peripherals

### 9.3 Long-Term Vision

Ultimate goal: - Production-quality retro-compatible peripheral subsystem - Complete peripheral coverage for legacy platform requirements - Used in production FPGA designs - Educational resource for classic peripheral design - Foundation for mixed-vintage SoC designs

---

**Version:** 1.0 **Last Review:** 2025-10-29 **Next Review:** After each new block completion **Maintained By:** RTL Design Sherpa Project