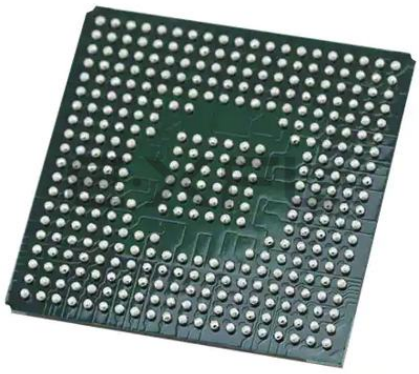


Welcome to E-XFL.COM

## Understanding Embedded - Microprocessors

Embedded microprocessors are specialized computing chips designed to perform specific tasks within an embedded system. Unlike general-purpose microprocessors found in personal computers, embedded microprocessors are tailored for dedicated functions within larger systems, offering optimized performance, efficiency, and reliability. These microprocessors are integral to the operation of countless electronic devices, providing the computational power necessary for controlling processes, handling data, and managing communications.

## Applications of Embedded - Microprocessors

Embedded microprocessors are utilized across a broad spectrum of applications, making them indispensable in

| Details | |
|---|---|
| Product Status | Obsolete |
| Core Processor | ARM920T |
| Number of Cores/Bus Width | 1 Core, 32-Bit |
| Speed | 200MHz |
| Co-Processors/DSP | Math Engine; MaverickCrunch™ |
| RAM Controllers | SDRAM |
| Graphics Acceleration | Yes |
| Display & Interface Controllers | Keypad, LCD, Touchscreen |
| Ethernet | 1/10/100Mbps (1) |
| SATA | - |
| USB | USB 2.0 (3) |
| Voltage - I/O | 1.8V, 3.3V |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Security Features | Hardware ID |
| Package / Case | 352-BBGA |
| Supplier Device Package | 352-PBGA (27x27) |
| Purchase URL | https://www.e-xfl.com/product-detail/cirrus-logic/ep9315-ib |

# OVERVIEW

The EP9315 is an ARM920T-based system-on-a-chip design with a large peripheral set targeted to a variety of applications:

- Thin Client Computers for Business and Home
- Internet Radio
- Internet Access Devices
- Industrial Computers
- Specialized Terminals
- Point-of-sale Terminals
- Test and Measurement Equipment

The ARM920T microprocessor core with separate 16-kbyte, 64-way set-associative instruction and data caches is augmented by the MaverickCrunch™ co-processor, enabling high-speed floating point calculations.

MaverickKey™ unique hardware programmed IDs are a solution to the growing concern over secure web content and commerce. With Internet security playing an important role in the delivery of digital media such as books or music, traditional software methods are quickly becoming unreliable. The MaverickKey unique IDs provide OEMs with a method of utilizing specific hardware IDs such as those assigned for SDMI (Secure Digital Music Initiative) or any other authentication mechanism.

A high-performance 1/10/100-Mbps Ethernet media access controller (EMAC) is included along with external interfaces to SPI, $I^2S$ audio, Raster/LCD, IDE storage peripherals, keypad, and touchscreen. A three-port USB 2.0 Full Speed Host (OHCI) (12 Mbits per second) and three UARTs are included as well.

The EP9315 is a high-performance, low-power, RISC-based, single-chip computer built around an ARM920T microprocessor core with a maximum operating clock rate of 200 MHz (184 MHz for industrial conditions). The ARM core operates from a 1.8 V supply, while the I/O operates at 3.3 V with power usage between 100 mW and 750 mW (dependent on speed).

**Table A. Change History**

| Revision | Date | Changes |
|---|---|---|
| PP1 | January 2004 | Initial Release. |
| PP2 | July 2004 | Update AC data.<br>Add ADC data. |
| PP3 | Febuary 2005 | Update electrical characteristics based upon more complete characterization data. |
| PP4 | March 2005 | Minor correction to block diagram on page 1. DD7 changed to pull down. |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

## List of Figures



## List of Tables

## Processor Core - ARM920T

The ARM920T is a Harvard architecture processor with separate 16-kbyte instruction and data caches with an 8-word line length but a unified memory. The processor utilizes a five-stage pipeline consisting of fetch, decode, execute, memory, and write stages. Key features include:

- ARM (32-bit) and Thumb (16-bit compressed) Instruction Sets
- 32-bit Advanced Micro-Controller Bus Architecture (AMBA)
- 16-kbyte Instruction Cache with Lockdown
- 16-kbyte Data Cache (programmable write-through or write-back) with Lockdown
- MMU for Linux®, Microsoft® Windows® CE and Other Operating Systems
- Translation Look Aside Buffers with 64 Data and 64 Instruction Entries
- Programmable Page Sizes of 1 Mbyte, 64 kbyte, 4 kbyte, and 1 kbyte
- Independent Lockdown of TLB Entries

## MaverickCrunch™ Math Engine

The MaverickCrunch Engine is a mixed-mode coprocessor designed primarily to accelerate the math processing required to rapidly encode digital audio formats. It accelerates single and double precision integer and floating point operations plus an integer multiply-accumulate (MAC) instruction that is considerably faster than the ARM920T's native MAC instruction. The ARM920T coprocessor interface is utilized thereby sharing its memory interface and instruction stream. Hardware forwarding and interlock allows the ARM to handle looping and addressing while MaverickCrunch handles computation. Features include:

- IEEE-754 single and double precision floating point
- 32 / 64-bit integer
- Add / multiply / compare
- Integer MAC 32-bit input with 72-bit accumulate
- Integer Shifts
- Floating point to/from integer conversion
- Sixteen 64-bit register files
- Four 72-bit accumulators

## MaverickKey™ Unique ID

MaverickKey unique hardware programmed IDs are a solution to the growing concern over secure web content and commerce. With Internet security playing an important role in the delivery of digital media such as books or music, traditional software methods are quickly becoming unreliable. The MaverickKey unique IDs provide OEMs with a method of utilizing specific hardware IDs such as those assigned for SDMI (Secure Digital Music Initiative) or any other authentication mechanism.

Both a specific 32-bit ID as well as a 128-bit random ID is programmed into the EP9315 through the use of laser probing technology. These IDs can then be used to match secure copyrighted content with the ID of the target device the EP9315 is powering, and then deliver the copyrighted information over a secure connection. In addition, secure transactions can benefit by also matching device IDs to server IDs. MaverickKey IDs provide a level of hardware security required for today's Internet appliances.

## General Purpose Memory Interface (SDRAM, SRAM, ROM, FLASH)

The EP9315 features a unified memory address model where all memory devices are accessed over a common address/data bus. A separate internal port is dedicated to the read-only Raster/LCD refresh engine, while the rest of the memory accesses are performed via the Processor bus. The SRAM memory controller supports 8, 16 and 32-bit devices and accommodates an internal boot ROM concurrently with 32-bit SDRAM memory.

- 1-4 banks of 32-bit 66 or 100 MHz SDRAM
- One internal port dedicated to the Raster/LCD Refresh Engine (Read Only)
- Address and data bus shared between SDRAM, SRAM, ROM, and FLASH memory
- NOR FLASH memory supported

**Table B. General Purpose Memory Interface Pin Assignments**

| Pin Mnemonic | Pin Description |
|---|---|
| SDCLK | SDRAM Clock |
| SDCLKEN | SDRAM Clock Enable |
| SDCSn[3:0] | SDRAM Chip Selects 3-0 |
| RASn | SDRAM RAS |
| CASn | SDRAM CAS |
| SDWEn | SDRAM Write Enable |
| CSn[7:6] and CSn[3:0] | Chip Selects 7, 6, 3, 2, 1, 0 |
| AD[25:0] | Address Bus 25-0 |
| DA[31:0] | Data Bus 31-0 |
| DQMn[3:0] | SDRAM Output Enables / Data Masks |
| WRn | SRAM Write Strobe |
| RDn | SRAM Read / OE Strobe |
| WAITn | SRAM Wait Input |

## Universal Asynchronous Receiver/Transmitters (UARTs)

Three 16550-compatible UARTs are supplied. Two provide asynchronous HDLC (High-level Data Link Control) protocol support for full-duplex transmit and receive. The HDLC receiver handles framing, address matching, CRC checking, control-octet transparency, and optionally passes the CRC to the host at the end of the packet. The HDLC transmitter handles framing, CRC generation, and control-octet transparency. The host must assemble the frame in memory before transmission. The HDLC receiver and transmitter use the UART FIFOs to buffer the data streams. A third IrDA®-compatible UART is also supplied.

- UART1 supports modem bit rates up to 115.2 Kbps, supports HDLC and includes a 16-byte FIFO for receive and a 16-byte FIFO for transmit. Interrupts are generated on Rx, Tx, and modem status change.
- UART2 contains an IrDA encoder operating at either the slow (up to 115 Kbps), medium (0.576 or 1.152 Mbps), or fast (4 Mbps) IR data rates. It also has a 16-byte FIFO for receive and a 16-byte FIFO for transmit.
- UART3 supports HDLC and includes a 16-byte FIFO for receive and a 16-byte FIFO for transmit. Interrupts are generated on Rx and Tx.

**Table I. Universal Asynchronous Receiver/Transmitters Pin Assignments**

| Pin Mnemonic | Pin Name - Description |
|---|---|
| TXD0 | UART1 Transmit |
| RXD0 | UART1 Receive |
| CTSn | UART1 Clear To Send / Transmit Enable |
| DSRn / DCDn | UART1 Data Set Ready / Data Carrier Detect |
| DTRn | UART1 Data Terminal Ready |
| RTSn | UART1 Ready To Send |
| EGPIO[0] / RI | UART1 Ring Indicator |
| TXD1 / SIROUT | UART2 Transmit / IrDA Output |
| RXD1 / SIRIN | UART2 Receive / IrDA Input |
| TXD2 | UART3 Transmit |
| RXD2 | UART3 Receive |
| EGPIO[3] / TENn | HDLC3 Transmit Enable |

## Triple Port USB Host

The USB Open Host Controller Interface (Open HCI) provides full speed serial communications ports at a baud rate of 12 Mbits/sec. Up to 127 USB devices (printer, mouse, camera, keyboard, etc.) and USB hubs can be connected to the USB host in the USB “tiered-start” topology.

This includes the following features:

- Compliance with the USB 2.0 specification
- Compliance with the Open HCI Rev 1.0 specification
- Supports both low speed (1.5 Mbps) and full speed (12 Mbps) USB device connections
- Root HUB integrated with 3 downstream USB ports
- Transceiver buffers integrated, over-current protection on ports
- Supports power management
- Operates as a master on the bus

The Open HCI host controller initializes the master DMA transfer with the AHB bus:

- Fetches endpoint descriptors and transfer descriptors
- Accesses endpoint data from system memory
- Accesses the HC communication area
- Writes status and retire transfer descriptor

**Table J. Triple Port USB Host Pin Assignments**

| Pin Mnemonic | Pin Name - Description |
|---|---|
| USBp[2:0] | USB Positive signals |
| USBm[2:0] | USB Negative Signals |

## Two-wire Interface

The two-wire interface provides communication and control for synchronous-serial-driven devices.

**Table K. Two-Wire Port with EEPROM Support Pin Assignments**

| Pin Mnemonic | Pin Name - Description | Alternative Usage |
|---|---|---|
| EECLK | Two-Wire Interface Clock | General Purpose I/O |
| EEDATA | Two-Wire Interface Data | General Purpose I/O |

## Real-Time Clock with Software Trim

The software trim feature on the real time clock (RTC) provides software controlled digital compensation of the 32.768 kHz input clock. This compensation is accurate to ± 1.24 sec/month.

*Note: A real time clock must be connected to RTCXTALI or the EP9315 device will not boot.*

**Table L. Real-Time Clock with Pin Assignments**

| Pin Mnemonic | Pin Name - Description |
|---|---|
| RTCXTALI | Real-Time Clock Oscillator Input |
| RTCXTALO | Real-Time Clock Oscillator Output |

## PLL and Clocking

The processor and the peripheral clocks operate from a single 14.7456 MHz crystal.

The real time clock operates from a 32.768 kHz external oscillator.

**Table M. PLL and Clocking Pin Assignments**

| Pin Mnemonic | Pin Name - Description |
|---|---|
| XTALI | Main Oscillator Input |
| XTALO | Main Oscillator Output |
| VDD_PLL | Main Oscillator Power |
| GND_PLL | Main Oscillator Ground |

## Timers

The Watchdog Timer insures proper operation by requiring periodic attention to prevent a reset-on-time-out.

Two 16-bit timers operate as free running down-counters or as periodic timers for fixed interval interrupts and have a range of 0.03 ms to 4.27 seconds.

One 32-bit timer, plus a 6-bit prescale counter, has a range of 0.03 µs to 73.3 hours.

One 40-bit debug timer, plus 6-bit prescale counter, has a range of 1.0 µs to 12.7 days.

## Interrupt Controller

The interrupt controller allows up to 64 interrupts to generate an Interrupt Request (IRQ) or Fast Interrupt Request (FIQ) signal to the processor core. Thirty-two hardware priority assignments are provided for assisting IRQ vectoring, and two levels are provided for FIQ vectoring. This allows time critical interrupts to be processed in the shortest time possible. Internal interrupts may be programmed as active-high or active-low, level-sensitive inputs. GPIO may be programmed as active-high level-sensitive, active-low level-sensitive, rising-edge-triggered, falling-edge-triggered, or combined rising/falling-edge-triggered.

- Supports 64 interrupts from a variety of sources (such as UARTs, GPIO, and key matrix)
- Routes interrupt sources to either the ARM920T's IRQ or FIQ (Fast IRQ) inputs
- Four dedicated off-chip interrupt lines INT[3:0] operate as active-high, level-sensitive interrupts
- Any of the 16 GPIO lines maybe configured to generate interrupts
- Software supported priority mask for all FIQs and IRQs

**Table N. External Interrupt Pin Assignment**

| Pin Mnemonic | Pin Name - Description |
|---|---|
| INT[3:0] | External Interrupt 3-0 |

## Dual LED Drivers

Two pins are assigned specifically to drive external LEDs.

**Table O. Dual LED Pin Assignments**

| Pin Mnemonic | Pin Name - Description | Alternative Usage |
|---|---|---|
| GRLED | Green LED | General Purpose I/O |
| REDLED | Red LED | General Purpose I/O |

## General Purpose Input/Output (GPIO)

The 16 EGPIO pins may each be configured individually as an output, an input, or an interrupt input. Port F may be configured as GPIO. Each Port F pin may be configured individually as an output, input or an interrupt input.

There are 23 pins that may be used as alternate inputs or outputs, but do not support interrupts. These pins are:

- Key Matrix ROW[7:0], COL[7:0]
- Ethernet MDIO
- Both LED Outputs
- Two-wire Clock and Data
- SLA [1:0]

6 pins may alternatively be used as inputs only:

- CTSn, DSRn / DCDn
- 4 Interrupt Lines

2 pins may alternatively be used as outputs only:

- RTSn
- ARSTn

### *Static Memory 32-bit Read on 8-bit External Bus*

| Parameter | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| AD setup to CSn assert time | $t_{ADs}$ | $t_{HCLK}$ | - | - | ns |
| CSn assert to Address transition time | $t_{AD1}$ | - | $t_{HCLK} \times (WST1 + 1)$ | - | ns |
| Address assert time | $t_{AD2}$ | - | $t_{HCLK} \times (WST1 + 1)$ | - | ns |
| AD transition to CSn deassert time | $t_{AD3}$ | - | $t_{HCLK} \times (WST1 + 2)$ | - | ns |
| AD hold from CSn deassert time | $t_{ADh}$ | $t_{HCLK}$ | - | - | ns |
| RDn assert time | $t_{RDpwL}$ | - | $t_{HCLK} \times (4 \times WST1 + 5)$ | - | ns |
| CSn to RDn delay time | $t_{RDd}$ | - | - | 3 | ns |
| CSn assert to DQMn assert delay time | $t_{DQMd}$ | - | - | 1 | ns |
| DA setup to AD transition time | $t_{DAs1}$ | 15 | - | - | ns |
| DA setup to RDn deassert time | $t_{DAs2}$ | $t_{HCLK} + 12$ | - | - | ns |
| DA hold from AD transition time | $t_{DAh1}$ | 0 | - | - | ns |
| DA hold from RDn deassert time | $t_{DAh2}$ | 0 | - | - | ns |

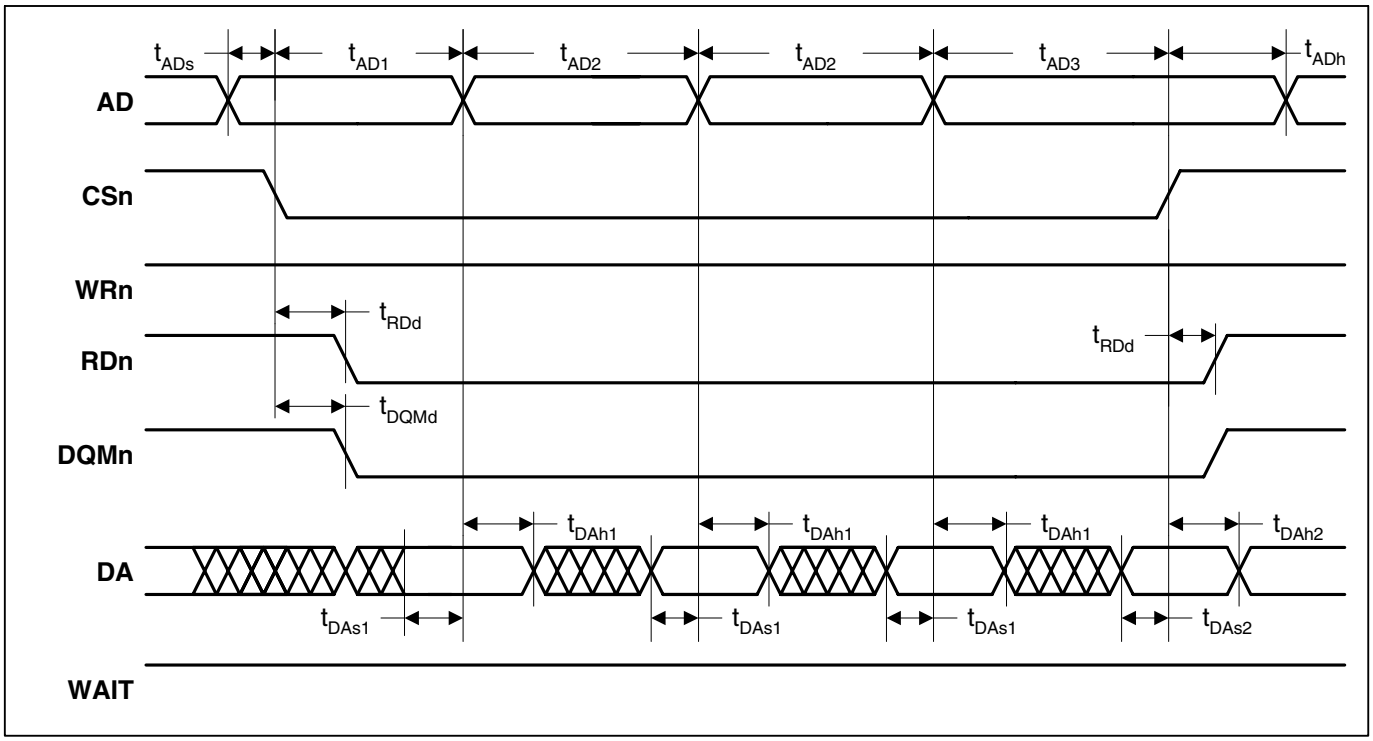



**Figure 8. Static Memory Multiple Word Read 8-bit Cycle Timing Measurement**

### *Static Memory 32-bit Read on 16-bit External Bus*

| Parameter | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| AD setup to CSn assert time | $t_{ADs}$ | $t_{HCLK}$ | - | - | ns |
| CSn assert to AD transition time | $t_{ADd1}$ | - | $t_{HCLK} \times (WST1 + 1)$ | - | ns |
| AD transition to CSn deassert time | $t_{ADd2}$ | - | $t_{HCLK} \times (WST1 + 2)$ | - | ns |
| AD hold from CSn deassert time | $t_{ADh}$ | $t_{HCLK}$ | - | - | ns |
| RDn assert time | $t_{RDpwL}$ | - | $t_{HCLK} \times ((2 \times WST1) + 3)$ | - | ns |
| CSn to RDn delay time | $t_{RDd}$ | - | - | 3 | ns |
| CSn assert to DQMn assert delay time | $t_{DQMd}$ | - | - | 1 | ns |
| DA setup to AD transition time | $t_{DAs1}$ | 15 | - | - | ns |
| DA to RDn deassert time | $t_{DAs2}$ | $t_{HCLK}$ + 12 | - | - | ns |
| DA hold from AD transition time | $t_{DAh1}$ | 0 | - | - | ns |
| DA hold from RDn deassert time | $t_{DAh2}$ | 0 | - | - | ns |

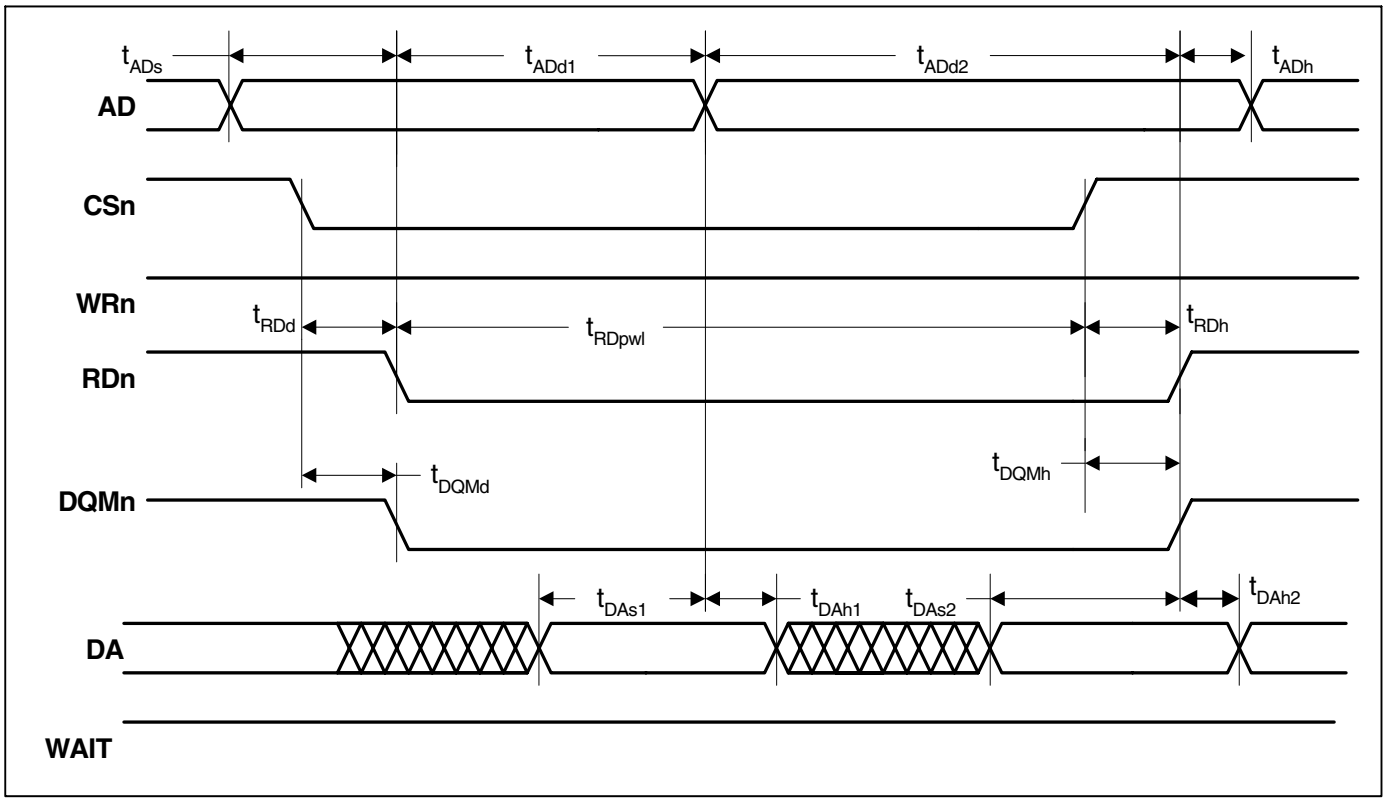



**Figure 10. Static Memory Multiple Word Read 16-bit Cycle Timing Measurement**

### *Static Memory Single Write Wait Cycle*

| Parameter | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| WAIT to WRn deassert delay time | $t_{WRd}$ | $t_{HCLK} \times 2$ | - | $t_{HCLK} \times 4$ | ns |
| CSn assert to WAIT time | $t_{WAITd}$ | - | - | $t_{HCLK} \times$ (WST1-2) | ns |
| WAIT assert time | $t_{WAITpw}$ | $t_{HCLK} \times 2$ | - | $t_{HCLK} \times 510$ | ns |
| WAIT to CSn deassert delay time | $t_{CSnd}$ | $t_{HCLK} \times 3$ | - | $t_{HCLK} \times 5$ | ns |

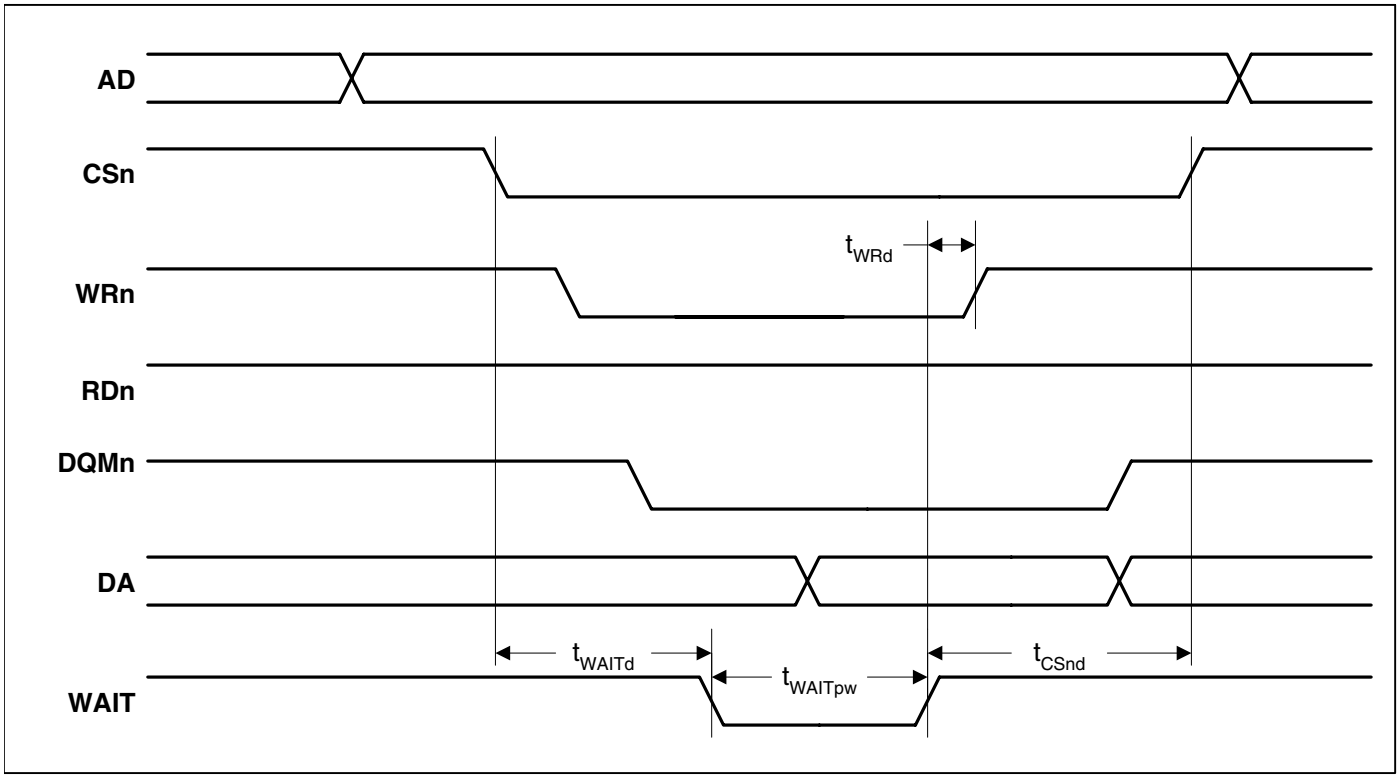



**Figure 15. Static Memory Single Write Wait Cycle Timing Measurement**

### *Static Memory Turnaround Cycle*

| Parameter | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| CSnX deassert to CSnY assert time | $t_{BTcyc}$ | - | $t_{HCLK} \times (IDCY+1)$ | - | ns |

*Notes: 1. X and Y represent any two chip select numbers.*
*2. IDCY occurs on read-to-write and write-to-read.*
*3. IDCY is honored when going from a asynchronous device (CSx) to a synchronous device (/SDCSy).*

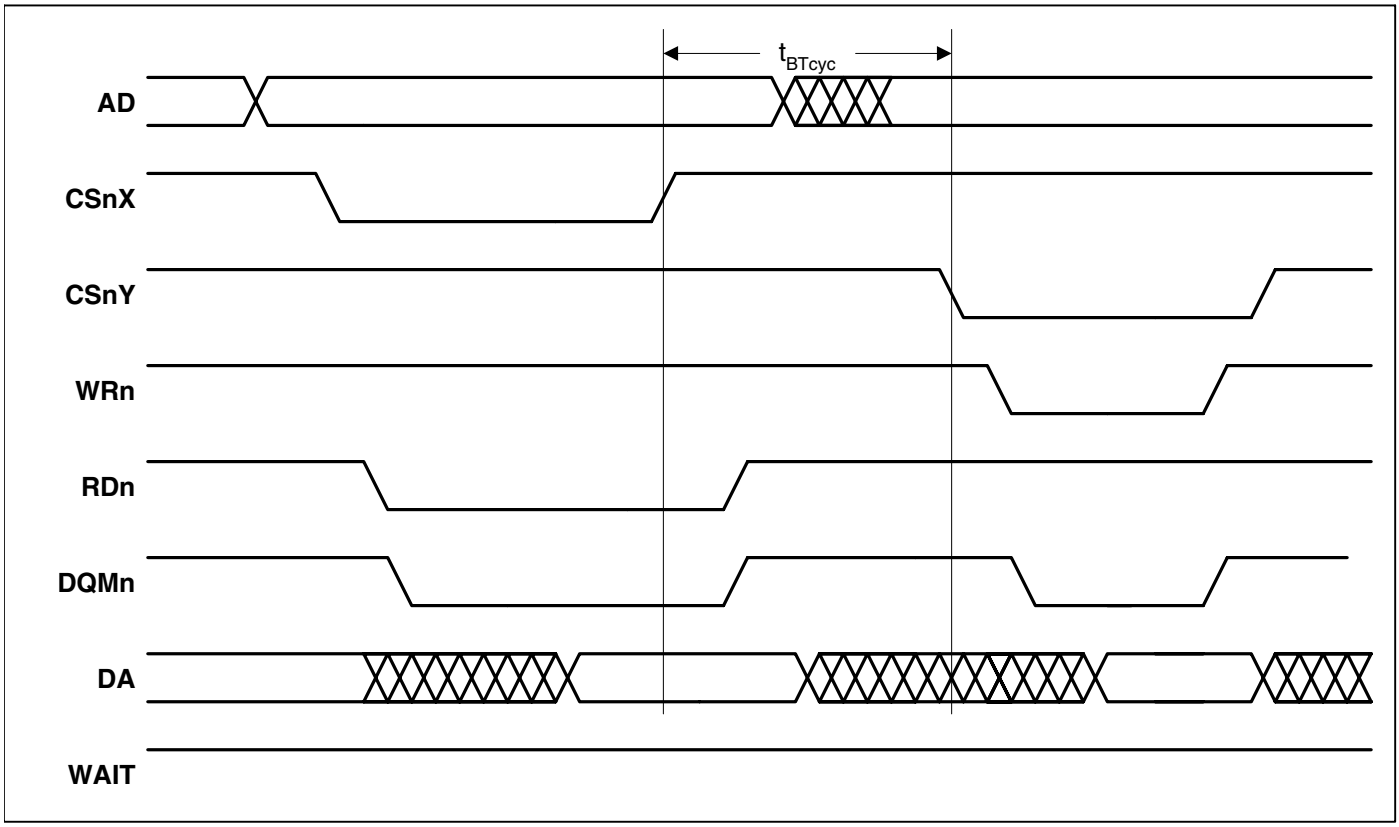



**Figure 16. Static Memory Turnaround Cycle Timing Measurement**

### *PCMCIA Write Cycle*

| Parameter | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| AD setup to signal transition time | $t_{ADs}$ | 0 | - | - | ns |
| Attribute access time | $t_A$ | $[(AA + 1) \times t_{HCLK}] - 14$ | $(AA + 1) \times t_{HCLK}$ | - | ns |
| Attribute hold time | $t_H$ | $[(HA + 1) \times t_{HCLK}] - 3$ | $(HA + 1) \times t_{HCLK}$ | - | ns |
| Attribute space pre-charge delay time | $t_p$ | $(PA + 1) \times t_{HCLK}$ | $(PA + 1) \times t_{HCLK}$ | - | ns |
| Common access time | $t_A$ | $[(AC + 1) \times t_{HCLK}] - 14$ | $(AC + 1) \times t_{HCLK}$ | - | ns |
| Common hold time | $t_H$ | $[(HC + 1) \times t_{HCLK}] - 3$ | $(HC + 1) \times t_{HCLK}$ | - | ns |
| Common space pre-charge delay time | $t_p$ | $(PC + 1) \times t_{HCLK}$ | $(PC + 1) \times t_{HCLK}$ | - | ns |
| I/O access time | $t_A$ | $[(AI + 1) \times t_{HCLK}] - 14$ | $(AI + 1) \times t_{HCLK}$ | - | ns |
| I/O hold time | $t_H$ | $[(HI + 1) \times t_{HCLK}] - 3$ | $(HI + 1) \times t_{HCLK}$ | - | ns |
| I/O space pre-charge delay time | $t_p$ | $(PI + 1) \times t_{HCLK}$ | $(PI + 1) \times t_{HCLK}$ | - | ns |
| MCDIR hold time | $t_{MCDh}$ | 0 | - | - | ns |
| DATA invalid delay time | $t_{DAfo}$ | 0 | - | - | ns |

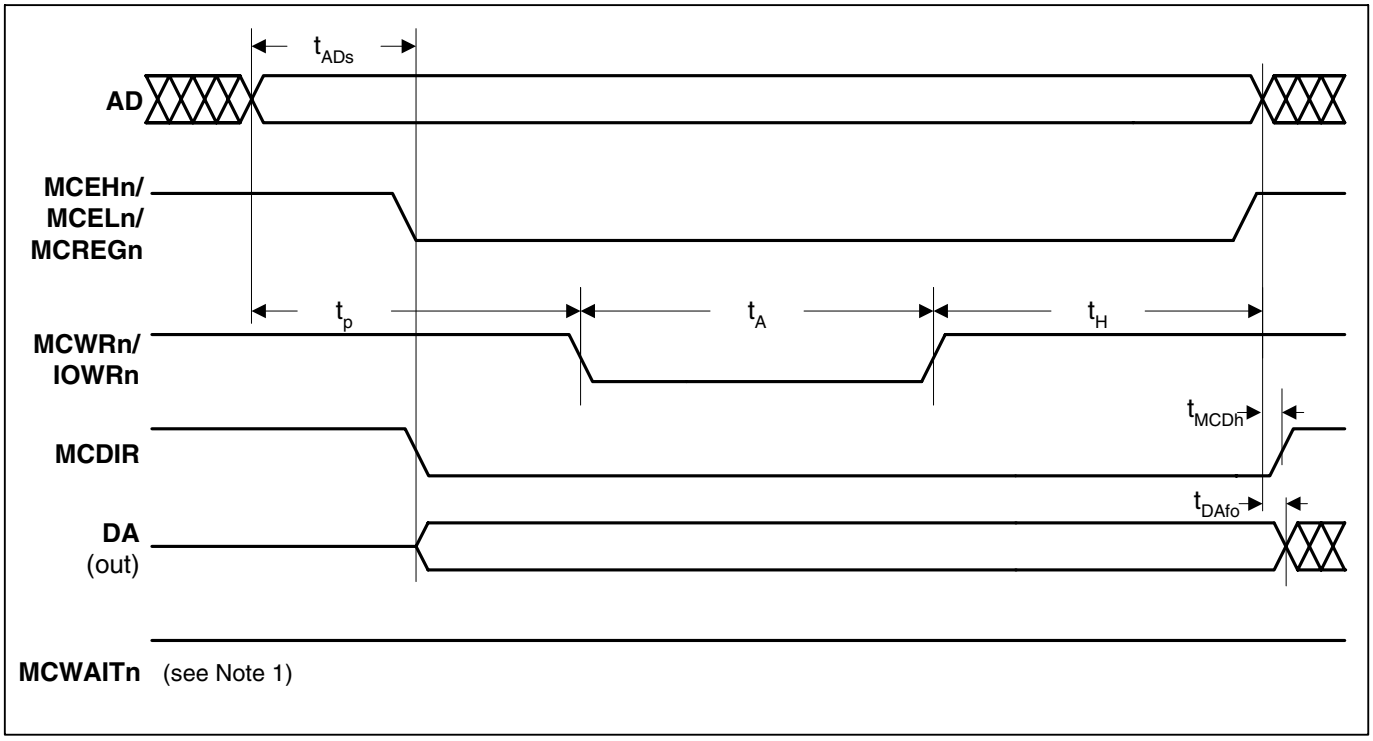
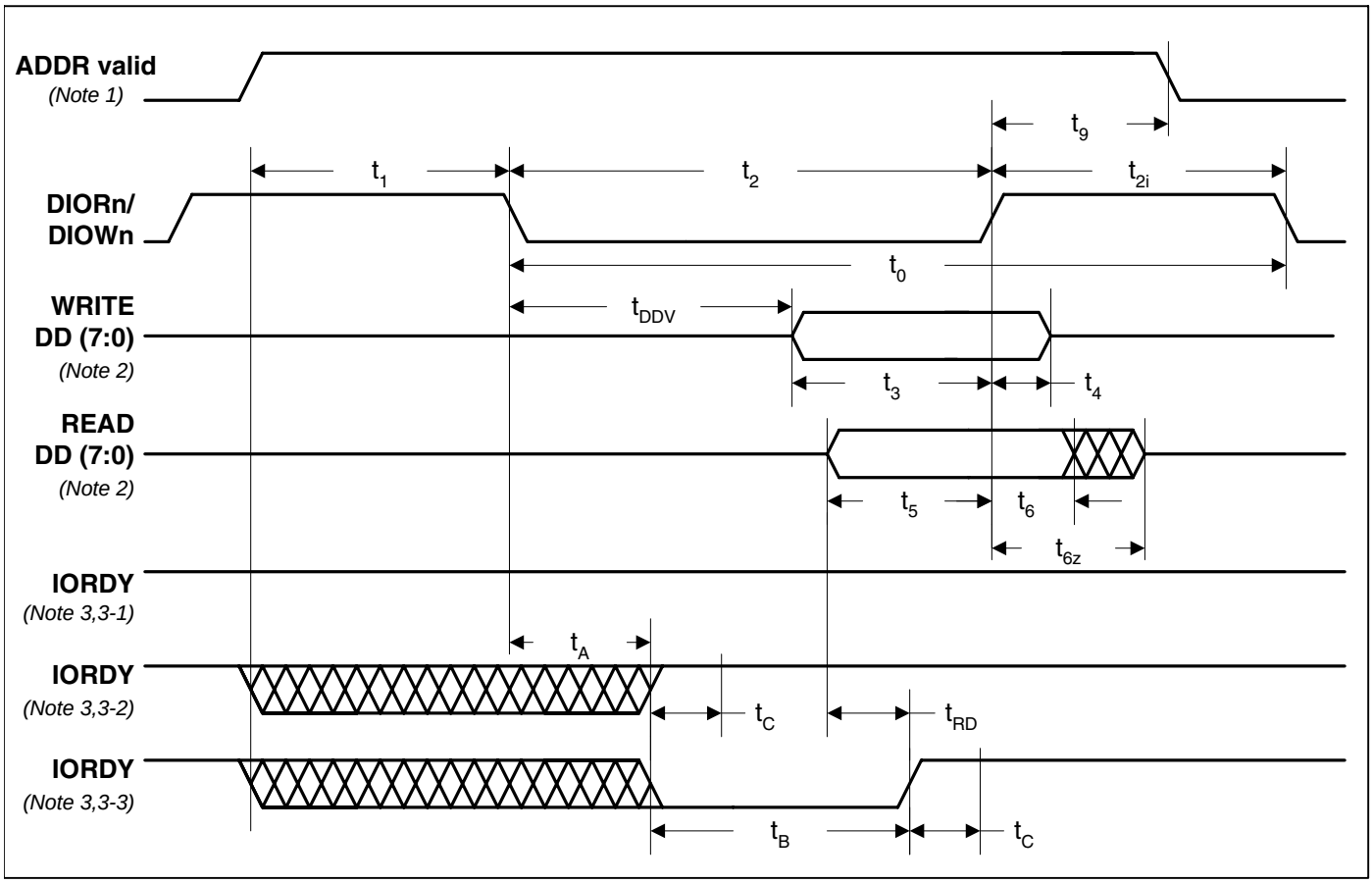



**Figure 18. PCMCIA Write Cycle Timing Measurement**

*Note: 1 - MCWAITn asserted will extend the MCWR / IOWR strobe time.*




*Note: 1. Device address consists of signals IDECS0n, IDECS1n and IDEDA (2:0)*

*2. Data consists of DD (7:0)*

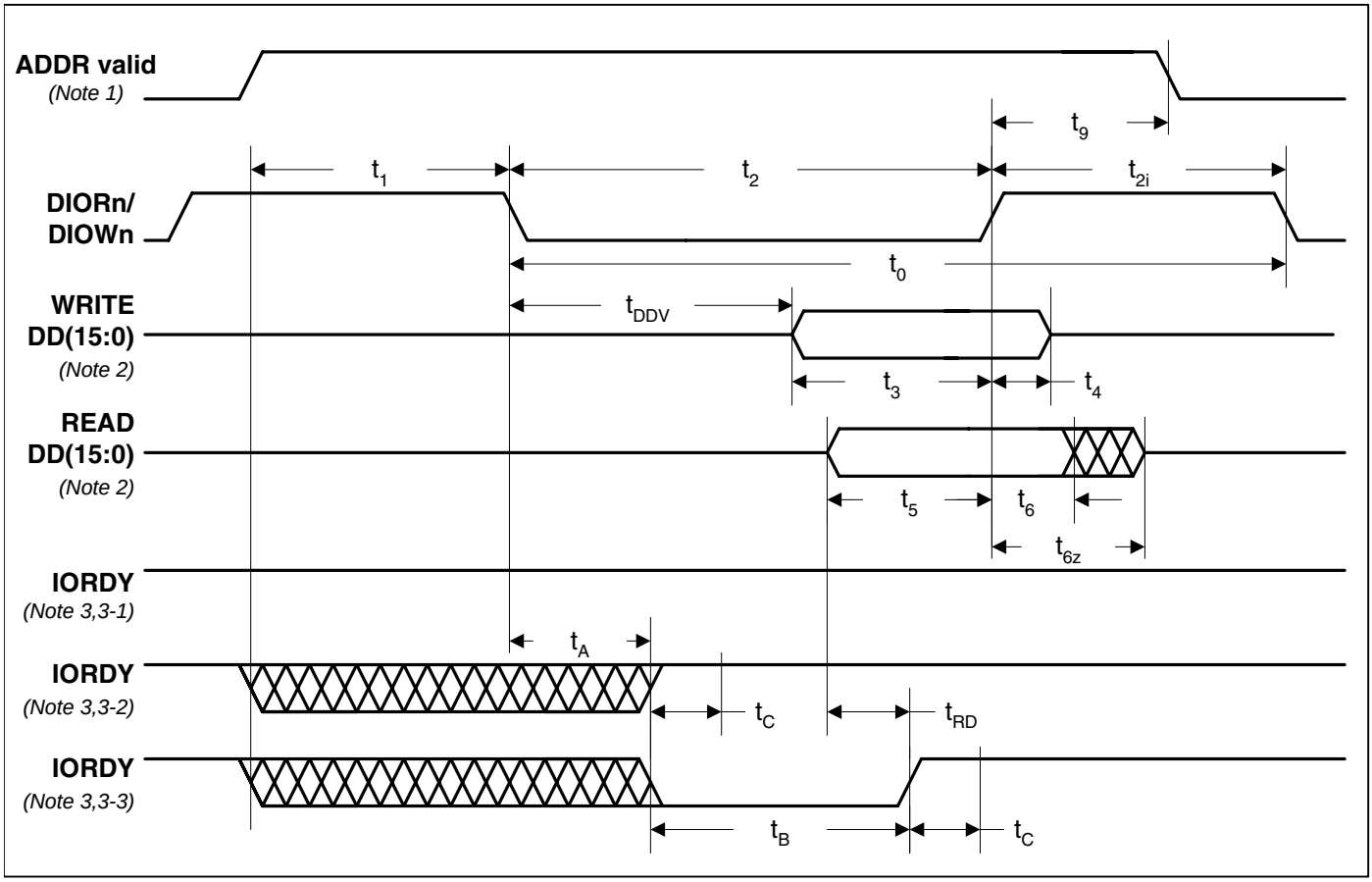
*3. The negation of IORDY by the device is used to extend the register transfer cycle. The determination of whether the cycle is to be extended is made by the host after $t_A$ from the assertion of DIORn or DIOWn. The assertion and negation or IORDY are described in the following three cases:*

*3-1 Device never negates IORDY, devices keeps IORDY released: no wait is generated.*

*3-2 Device negates IORDY before $t_A$, but causes IORDY to be asserted before $t_A$. IORDY is released prior to negation and may be asserted for no more than $t_C$ before release: no wait generated.*

*3-3 Device negates IORDY before $t_A$. IORDY is released prior to negation and may be asserted for no more than $t_C$ before release: wait generated. The cycle completes after IORDY is reasserted. For cycles where a wait is generated and DIORn is asserted, the device shall place read data on DD (7:0) for $t_{RD}$ before asserting IORDY.*

**Figure 19. Register Transfer to/from Device**




*Note: 1. Device address consists of signals IDECS0n, IDECS1n and IDEDA (2:0)*

*2. Data consists of DD (15:0)*

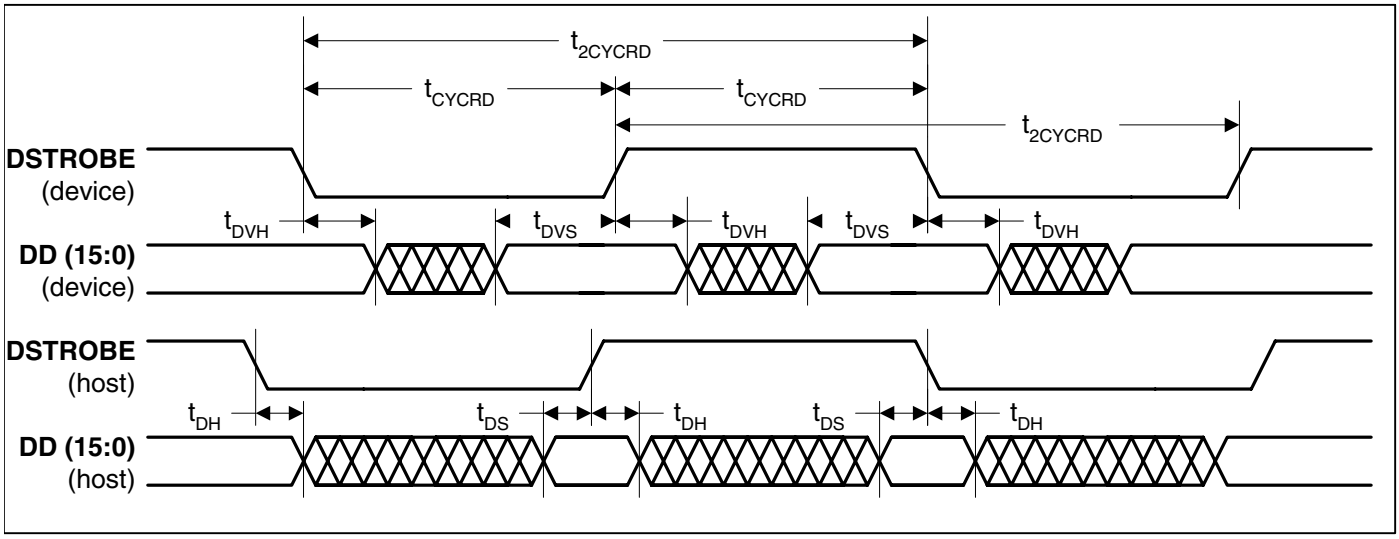
*3. The negation of IORDY by the device is used to extend the register transfer cycle. The determination of whether the cycle is to be extended is made by the host after $t_A$ from the assertion of DIORn or DIOWn. The assertion and negation or IORDY are described in the following three cases:*

*3-1 Device never negates IORDY, devices keeps IORDY released: no wait is generated.*

*3-2 Device negates IORDY before $t_A$, but causes IORDY to be asserted before $t_A$. IORDY is released prior to negation and may be asserted for no more than $t_C$ before release: no wait generated.*

*3-3 Device negates IORDY before $t_A$. IORDY is released prior to negation and may be asserted for no more than $t_C$ before release: wait generated. The cycle completes after IORDY is reasserted. For cycles where a wait is generated and DIORn is asserted, the device shall place read data on DD (15:0) for $t_{RD}$ before asserting IORDY.*

**Figure 20. PIO Data Transfer to/from Device**




*Note: DD (15:0) and DSTROBE signals are shown at both the host and the device to emphasize that cable settling time as well as cable propagation delay shall not allow the data signals to be considered stable at the host until some time after they are driven by the device.*

**Figure 22. Sustained Ultra DMA data-in Burst**

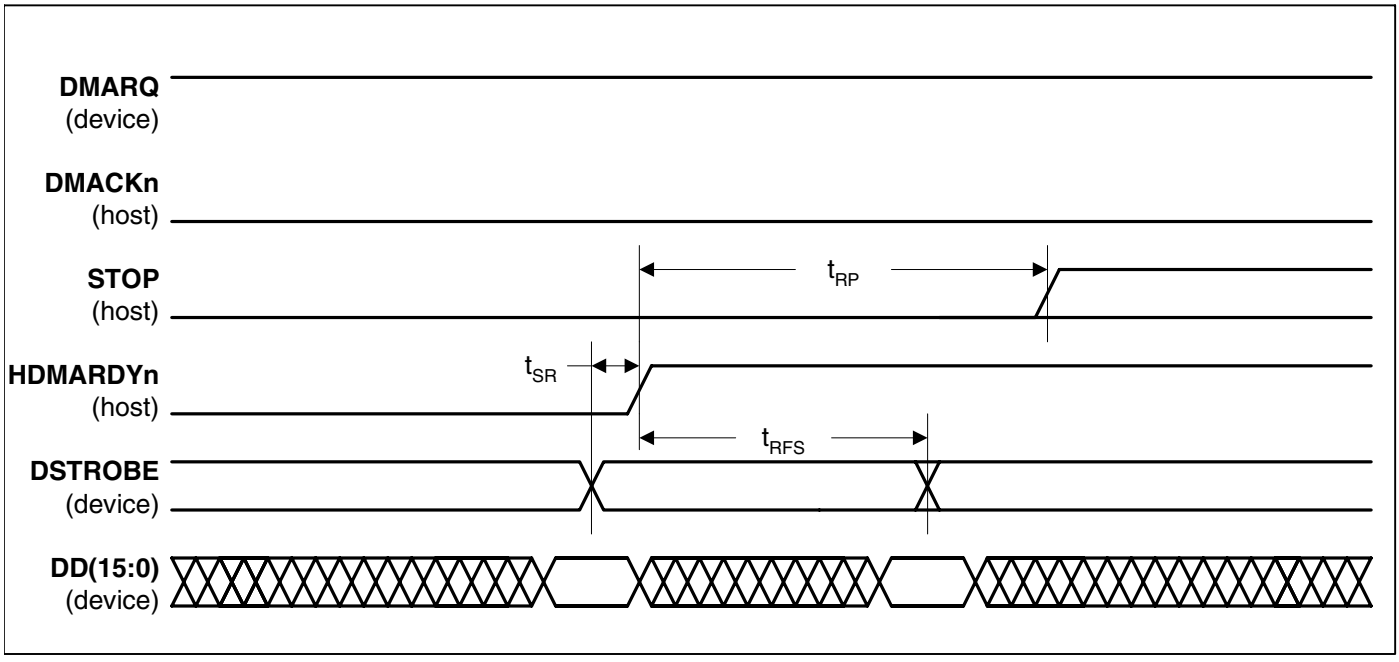
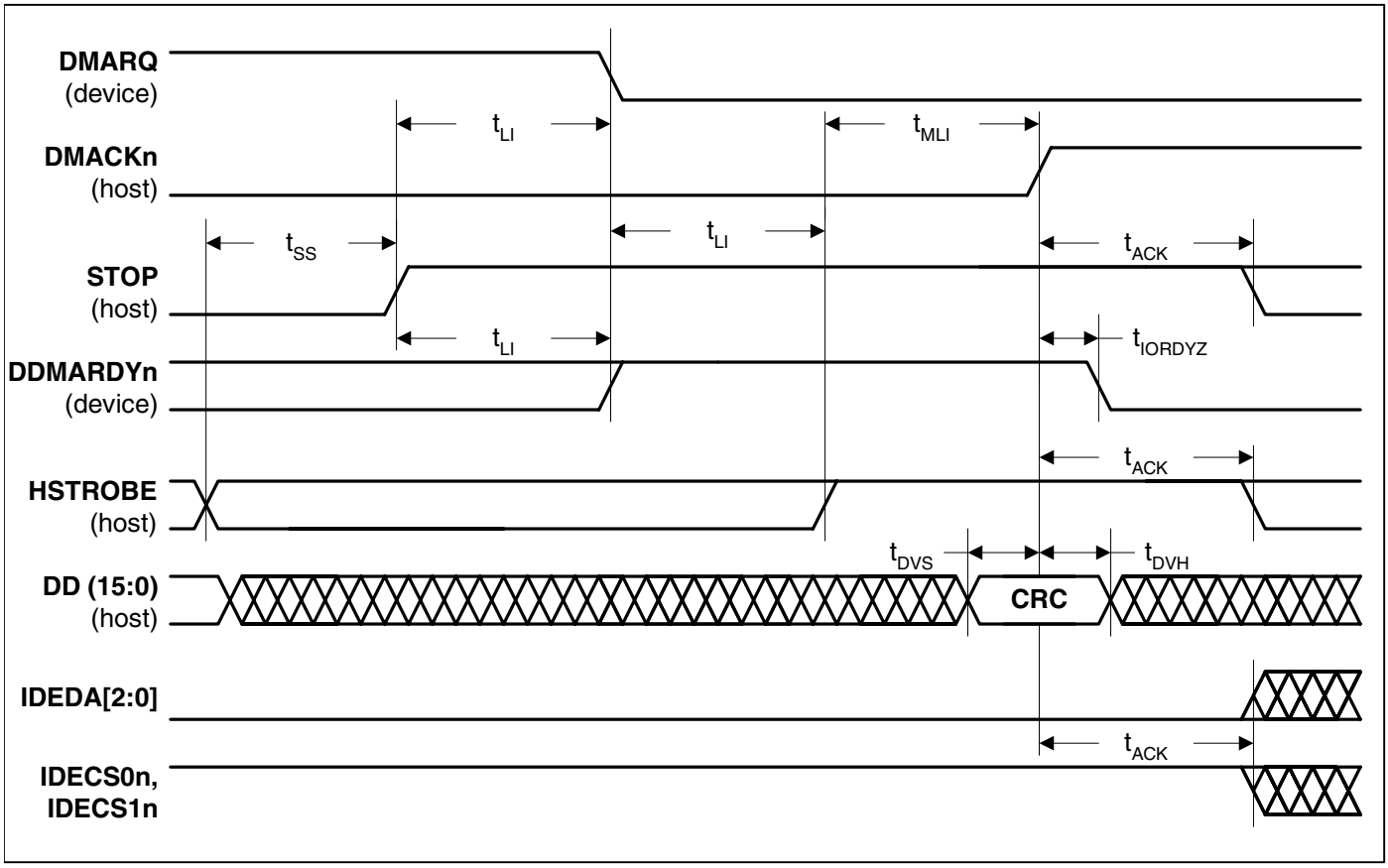



**Figure 23. Host Pausing an Ultra DMA data-in Burst**




*Note: The definitions for the DIOWn:STOP, IORDY:DDMARDYn:DSTROBE and DIORn:HDMARDYn:HSTROBE signal lines are no longer in effect after DMARQ and DMACKn are negated.*

**Figure 29. Host Terminating an Ultra DMA data-out Burst**

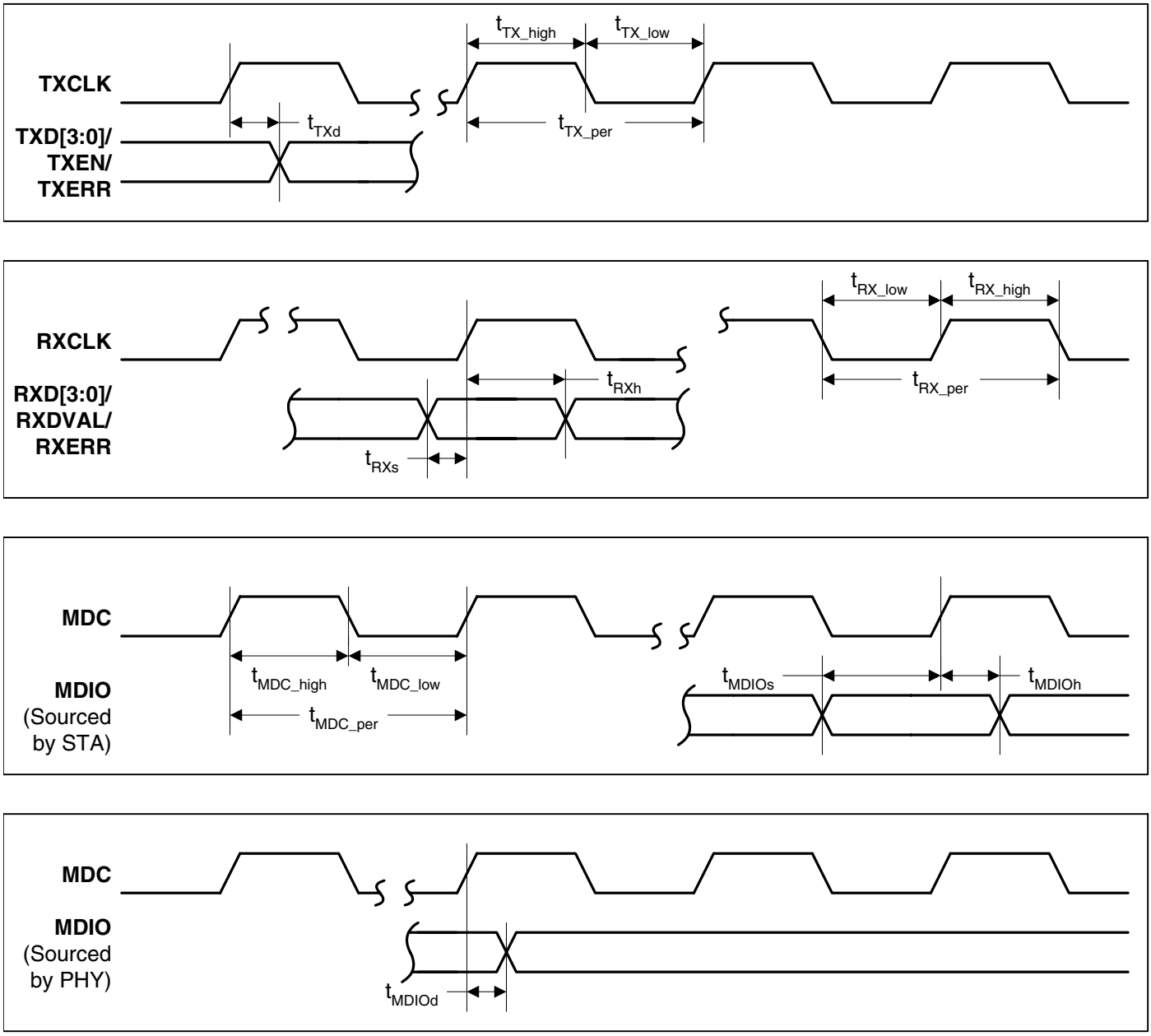



**Figure 31. Ethernet MAC Timing Measurement**

### *Motorola SPI*

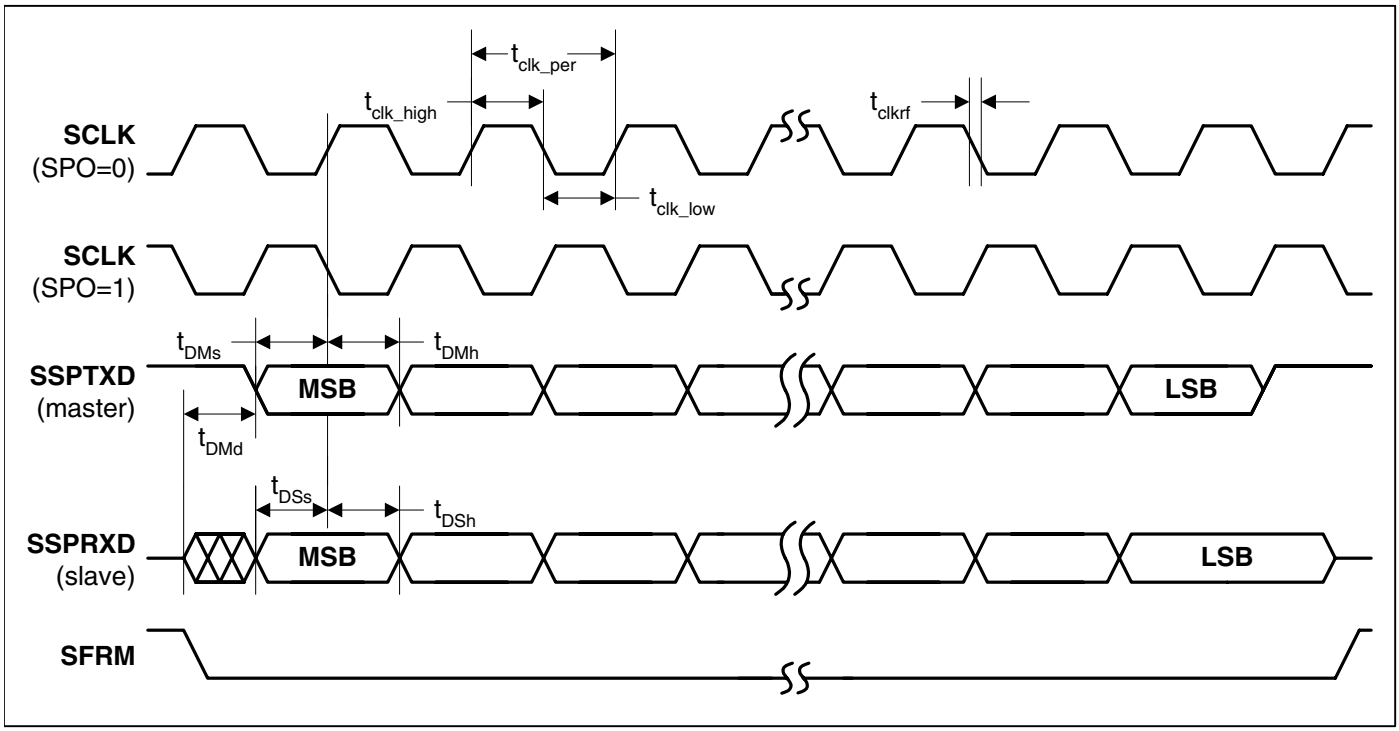



**Figure 34. SPI Format with SPH=1 Timing Measurement**

**Figure 40. 352 PIN BGA PINOUT**

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Y** | HSYNC | DD[1] | DD[12] | P[2] | AD[15] | DA[6] | DA[4] | AD[10] | DA[1] | AD[8] | IDEDA[0] | DTRN | TDO | BOOT[0] | EEDAT | ASDO | SFRM1 | RDLED | USBP[1] | ABITCLK | **Y** |
| **W** | P[12] | P[9] | DD[0] | P[5] | P[3] | DA[7] | DA[5] | AD[11] | AD[9] | IDECS1N | IDEDA[1] | TCK | TMS | EECLK | SCLK1 | GRLED | INT[3] | SLA[1] | SLA[0] | RXD[2] | **W** |
| **V** | P[16] | P[11] | P[8] | DD[15] | DD[13] | P[1] | AD[14] | AD[12] | DA[2] | IDECS0N | IDEDA[2] | TDI | GND | ASYNC | SSPTX1 | INT[2] | RTSN | USBP[0] | CTSN | TXD[0] | **V** |
| **U** | AD[0] | P[15] | P[10] | P[7] | P[6] | P[4] | P[0] | AD[13] | DA[3] | DA[0] | DSRN | BOOT[1] | NC | SSPRX1 | INT[1] | PWMOUT | USBM[0] | RXD[1] | TXD[1] | ROW[1] | **U** |
| **T** | DA[8] | BLANK | P[13] | SPCLK | V_CSYNC | DD[14] | GND | CVDD | RVDD | GND | GND | RVDD | CVDD | GND | INT[0] | USBM[1] | RXD[0] | TXD[2] | ROW[2] | ROW[4] | **T** |
| **R** | AD[2] | AD[1] | P[17] | P[14] | RVDD | RVDD | GND | CVDD | | | | | CVDD | GND | RVDD | RVDD | ROW[0] | ROW[3] | PLL_GND | ROW[5] | **R** |
| **P** | AD[4] | DA[10] | DA[9] | BRIGHT | RVDD | RVDD | | | | | | | | | RVDD | RVDD | XTALI | PLL_VDD | ROW[6] | ROW[7] | **P** |
| **N** | DA[13] | DA[12] | DA[11] | AD[3] | CVDD | CVDD | | GND | GND | GND | GND | GND | GND | | GND | GND | XTALO | COL[0] | COL[1] | COL[2] | **N** |
| **M** | AD[7] | DA[14] | AD[6] | AD[5] | CVDD | | | GND | GND | GND | GND | GND | GND | | | GND | COL[4] | COL[3] | COL[6] | CSN[0] | **M** |
| **L** | DA[18] | DA[17] | DA[16] | DA[15] | GND | | | GND | GND | GND | GND | GND | GND | | | CVDD | COL[5] | COL[7] | RSTON | PRSTN | **L** |
| **K** | AD[22] | DA[20] | AD[21] | DA[19] | RVDD | | | GND | GND | GND | GND | GND | GND | | | CVDD | SYM | SYP | SXM | SXP | **K** |
| **J** | DA[21] | DQMN[0] | DQMN[1] | DQMN[2] | GND | | | GND | GND | GND | GND | GND | GND | | | CVDD | RTCXTALI | XM | YP | YM | **J** |
| **H** | DQMN[3] | CASN | RASN | SDCSN[2] | CVDD | | | GND | GND | GND | GND | GND | GND | | | RVDD | RTCXTALO | ADC_VDD | ADC_GND | XP | **H** |
| **G** | SDCSN[0] | SDCSN[1] | SDWEN | SDCLK | RVDD | RVDD | | | | | | | | | RVDD | RVDD | EGPIO[7] | EGPIO[9] | EGPIO[10] | EGPIO[11] | **G** |
| **F** | SDCSN[3] | DA[22] | DA[24] | AD[25] | RVDD | GND | CVDD | | | | | | | CVDD | GND | GND | EGPIO[2] | EGPIO[4] | EGPIO[6] | EGPIO[8] | **F** |
| **E** | AD[23] | DA[23] | DA[26] | CSN[6] | GND | GND | CVDD | CVDD | RVDD | GND | GND | RVDD | CVDD | CVDD | GND | ASDI | DIOWN | EGPIO[0] | EGPIO[3] | EGPIO[5] | **E** |
| **D** | AD[24] | DA[25] | DD[11] | SDCLKEN | AD[19] | DD[9] | DD[5] | AD[16] | MIIRXD[2] | MIITXD[3] | TXEN | MCWAITN | MCDAENN | MCADENN | EGPIO[14] | WP | USBM[2] | ARSTN | DIORN | EGPIO[1] | **D** |
| **C** | CSN[1] | CSN[3] | AD[20] | DA[29] | DD[10] | DD[6] | DD[2] | MDC | MIIRXD[3] | TXCLK | MIITXD[0] | READY | MCD2 | MCDIR | MCELN | IORDN | MCWRN | USBP[2] | IORDY | DMACKN | **C** |
| **B** | CSN[2] | DA[31] | DA[30] | DA[27] | DD[7] | DD[3] | WRN | MDIO | MIIRXD[1] | RXERR | MIITXD[1] | CRS | VS1 | MCD1 | MCBVD2 | MCEHN | EGPIO[13] | MCRDN | WAITN | TRSTN | **B** |
| **A** | CSN[7] | DA[28] | AD[18] | DD[8] | DD[4] | AD[17] | RDN | RXCLK | MIIRXD[0] | RXDVAL | MIITXD[2] | TXERR | CLD | VS2 | MCBVD1 | MCREGN | EGPIO[12] | EGPIO[15] | IOWRN | MCRESETN | **A** |
| | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | |

| Ball | Signal | Ball | Signal | Ball | Signal | Ball | Signal |
|---|---|---|---|---|---|---|---|
| C3 | AD[20] | H5 | CVDD | N15 | GND | V15 | SSPTX1 |
| C4 | DA[29] | H8 | GND | N16 | GND | V16 | INT[2] |
| C5 | DD[10] | H9 | GND | N17 | XTALO | V17 | RTSN |
| C6 | DD[6] | H10 | GND | N18 | COL[0] | V18 | USBP[0] |
| C7 | DD[2] | H11 | GND | N19 | COL[1] | V19 | CTSN |
| C8 | MDC | H12 | GND | N20 | COL[2] | V20 | TXD[0] |
| C9 | MIIRXD[3] | H13 | GND | P1 | AD[4] | W1 | P[12] |
| C10 | TXCLK | H16 | RVDD | P2 | DA[10] | W2 | P[9] |
| C11 | MIITXD[0] | H17 | RTCXTALO | P3 | DA[9] | W3 | DD[0] |
| C12 | READY | H18 | ADC_VDD | P4 | BRIGHT | W4 | P[5] |
| C13 | MCD2 | H19 | ADC_GND | P5 | RVDD | W5 | P[3] |
| C14 | MCDIR | H20 | XP | P6 | RVDD | W6 | DA[7] |
| C15 | MCELN | J1 | DA[21] | P15 | RVDD | W7 | DA[5] |
| C16 | IORDN | J2 | DQMN[0] | P16 | RVDD | W8 | AD[11] |
| C17 | MCWRN | J3 | DQMN[1] | P17 | XTALI | W9 | AD[9] |
| C18 | USBP[2] | J4 | DQMN[2] | P18 | PLL_VDD | W10 | IDECS1N |
| C19 | IORDY | J5 | GND | P19 | ROW[6] | W11 | IDEDA[1] |
| C20 | DMACKN | J8 | GND | P20 | ROW[7] | W12 | TCK |
| D1 | AD[24] | J9 | GND | R1 | AD[2] | W13 | TMS |
| D2 | DA[25] | J10 | GND | R2 | AD[1] | W14 | EECLK |
| D3 | DD[11] | J11 | GND | R3 | P[17] | W15 | SCLK1 |
| D4 | SDCLKEN | J12 | GND | R4 | P[14] | W16 | GRLED |
| D5 | AD[19] | J13 | GND | R5 | RVDD | W17 | INT[3] |
| D6 | DD[9] | J16 | CVDD | R6 | RVDD | W18 | SLA[1] |
| D7 | DD[5] | J17 | RTCXTALI | R7 | GND | W19 | SLA[0] |
| D8 | AD[16] | J18 | XM | R8 | CVDD | W20 | RXD[2] |
| D9 | MIIRXD[2] | J19 | YP | R13 | CVDD | Y1 | HSYNC |
| D10 | MIITXD[3] | J20 | YM | R14 | GND | Y2 | DD[1] |
| D11 | TXEN | K1 | AD[22] | R15 | RVDD | Y3 | DD[12] |
| D12 | MCWAITN | K2 | DA[20] | R16 | RVDD | Y4 | P[2] |
| D13 | MCDAENN | K3 | AD[21] | R17 | ROW[0] | Y5 | AD[15] |
| D14 | MCADENN | K4 | DA[19] | R18 | ROW[3] | Y6 | DA[6] |
| D15 | EGPIO[14] | K5 | RVDD | R19 | PLL_GND | Y7 | DA[4] |
| D16 | WP | K8 | GND | R20 | ROW[5] | Y8 | AD[10] |
| D17 | USBM[2] | K9 | GND | T1 | DA[8] | Y9 | DA[1] |
| D18 | ARSTN | K10 | GND | T2 | BLANK | Y10 | AD[8] |
| D19 | DIORN | K11 | GND | T3 | P[13] | Y11 | IDEDA[0] |
| D20 | EGPIO[1] | K12 | GND | T4 | SPCLK | Y12 | DTRN |
| E1 | AD[23] | K13 | GND | T5 | V_CSYNC | Y13 | TDO |
| E2 | DA[23] | K16 | CVDD | T6 | DD[14] | Y14 | BOOT[0] |
| E3 | DA[26] | K17 | SYM | T7 | GND | Y15 | EEDAT |
| E4 | CSN[6] | K18 | SYP | T8 | CVDD | Y16 | ASDO |
| E5 | GND | K19 | SXM | T9 | RVDD | Y17 | SFRM1 |
| E6 | GND | K20 | SXP | T10 | GND | Y18 | RDLED |
| E7 | CVDD | L1 | DA[18] | T11 | GND | Y19 | USBP[1] |
| E8 | CVDD | L2 | DA[17] | T12 | RVDD | Y20 | ABITCLK |

## Acronyms and Abbreviations

The following tables list abbreviations and acronyms used in this data sheet.

| Term | Definition |
|---|---|
| ADC | Analog-to-Digital Converter |
| ALT | Alternative |
| AMBA | Advanced Micro-controller Bus Architecture |
| ATAPI | ATA Packet Interface |
| CODEC | COder / DECoder |
| CRC | Cyclic Redundancy Check |
| DAC | Digital-to-Analog Converter |
| DMA | Direct-Memory Access |
| EBUS | External Memory Bus |
| EEPROM | Electronically Erasable Programmable Read Only Memory |
| EMAC | Ethernet Media Access Controller |
| FIFO | First In / First Out |
| FIQ | Fast Interrupt Request |
| FLASH | Flash memory |
| GPIO | General Purpose I/O |
| HDLC | High-level Data Link Control |
| I/F | Interface |
| $I^2S$ | Inter-IC Sound |
| IC | Integrated Circuit |
| ICE | In-Circuit Emulator |
| IDE | Integrated Drive Electronics |
| IEEE | Institute of Electronics and Electrical Engineers |
| IrDA | Infrared Data Association |
| IRQ | Standard Interrupt Request |
| ISO | International Standards Organization |
| JTAG | Joint Test Action Group |
| LFSR | Linear Feedback Shift Register |
| MII | Media Independent Interface |
| MMU | Memory Management Unit |
| OHCI | Open Host Controller Interface |
| PHY | Ethernet PHYsical layer interface |
| PIO | Programmed I/O |
| RISC | Reduced Instruction Set Computer |
| SDMI | Secure Digital Music Initiative |
| SDRAM | Synchronous Dynamic RAM |
| SPI | Serial Peripheral Interface |
| SRAM | Static Random Access Memory |
| STA | Station - Any device that contains an IEEE 802.11 conforming Medium Access Control (MAC) and physical layer (PHY) interface to the wireless medium |
| TFT | Thin Film Transistor |
| TLB | Translation Lookaside Buffer |
| USB | Universal Serial Bus |

## Units of Measurement

| Symbol | Unit of Measure |
|---|---|
| $^{\circ}C$ | degree Celsius |
| Hz | Hertz = cycle per second |
| Kbps | Kilobits per second |
| kbyte | Kilobyte |
| kHz | KiloHertz = 1000 Hz |
| Mbps | Megabits per second |
| MHz | MegaHertz = 1,000 kHz |
| µA | microAmpere = $10^{-6}$ Ampere |
| µs | microsecond = 1,000 nanoseconds = $10^{-6}$ seconds |
| mA | milliAmpere = $10^{-3}$ Ampere |
| ms | millisecond = 1,000 microseconds = $10^{-3}$ seconds |
| mW | milliWatt = $10^{-3}$ Watts |
| ns | nanosecond = $10^{-9}$ seconds |
| pF | picoFarad = $10^{-12}$ Farads |
| V | Volt |
| W | Watt |