Advanced Communication Devices

Data Sheet: ACD82112

INTRODUCTORY

Data Sheet: ACD82112

12 Ports 10/100 Fast Ethernet Switch

Rev.1.3.4.I
Last Update: March 20, 1999
Subject to Chnage

Please check ACD's website for update information before starting a design
Web site: http://www.acdcorp.com

or Contact ACD at:
Email: support@acdcorp.com
Tel: 408-433-9898
Fax: 408-545-0930

ACD Confidential Material

For ACD authorized customer use only. No reproduction or redistribution without ACD's prior permission.

ACD Confidential. Do Not Reproduce. Use under Non-Disclosure agreement only.

Table of Contents

Section **Page**

Appendix

INTRODUCTORY

ACD Confidential. Do Not Reproduce. Use under Non-Disclosure agreement only.

1. GENERAL DESCRIPTION

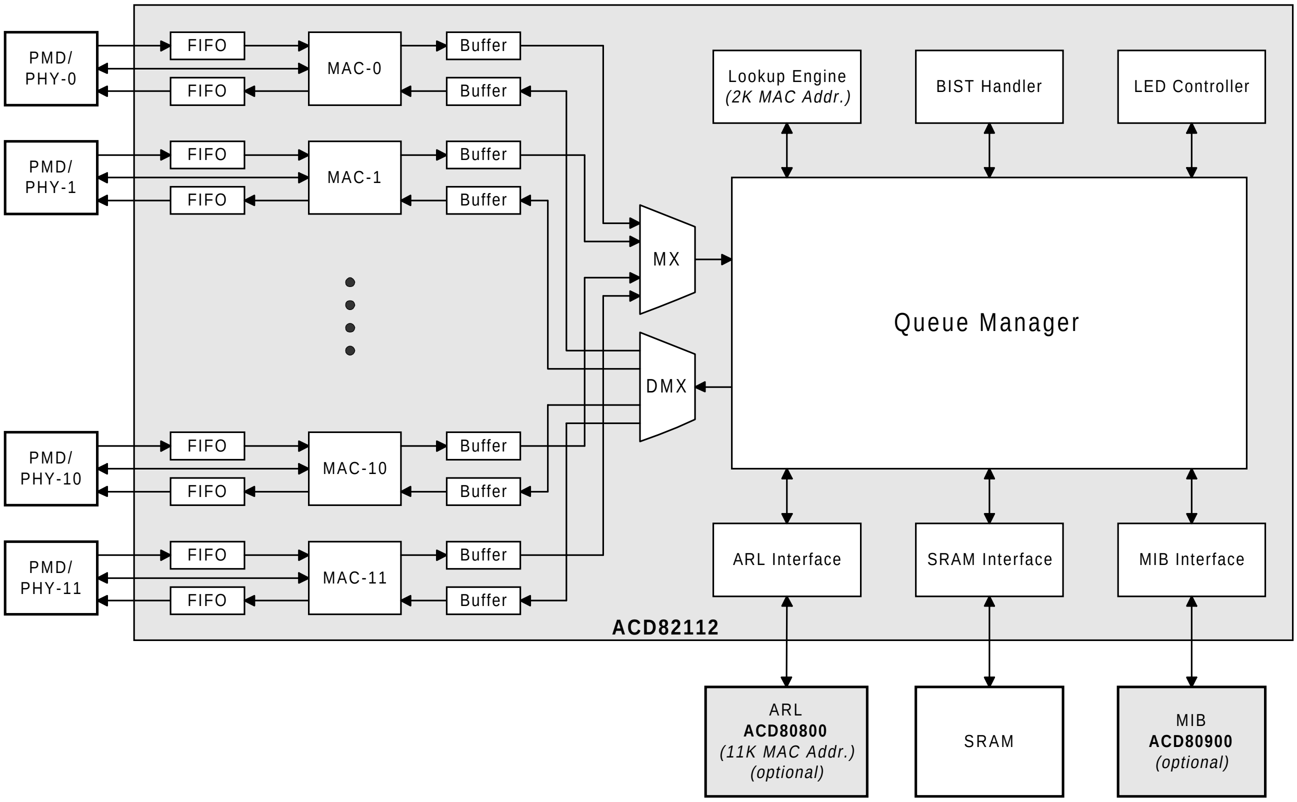
The ACD82112 is a single chip implementation of a 12 port 10/100 Ethernet switch system intended for IEEE 802.3 and 802.3u compatible networks. The device includes 12 independent 10/100 MACs. Each MAC interfaces with an external PMD/PHY device through a standard MII interface. Speed can be automatically configured through the MDIO or the optional CPU UART. Each port can operate at either 10Mbps or 100Mbps, in half-duplex or full-duplex mode. The core logic of the ACD82112, implemented with patent pending BASIQ *(Bandwidth Assured Switching with Intelligent Queuing)* technology, can simultaneously process 12 asynchronous 10/100Mbps traffics. The Queue Manager inside the ACD82112 provides the capability of routing traffic with the same order of sequence, without any packet loss.

A complete 12 port 10/100 switch can be built with the use of the ACD82112, 10/100 PHY and SRAM. The MAC addresses space can be expanded from the built-in 2K to 11K by using ACD's external ARL (*Address Resolution Logic)*, the ACD80800. Advanced network management features can be supported with the use of ACD's *MIB (Management Information Base, ACD80900)* chip.

2. MAIN FEATURES

- 12 - 10/100 Mbps, auto-sensing ports with MII interface
- Full / half duplex operation
- 2.4 Gbps aggregated throughput
- True non-blocking switch architecture
- Shared buffer with starvation control algorithm
- Built-in storage of 2,048 MAC addresses
- Automatic source address learning
- Optional back-pressure (half duplex) flow control
- 802.3x pause frame (full duplex) flow control
- Store-and-forward switch mode
- Port based VLAN support
- UART type CPU management interface
- Supports up to 11K addresses with External ARL controller, the ACD80800
- RMON and SNMP support with External MIB controller, the ACD80900
- Status LEDs: Link, Speed, Full/Half Duplex, Transmit, Receive, Collision and Frame Error
- Reversible MII option for CPU and expansion port interface, with hardware based flow control
- Wire speed forwarding rate
- 388-pin PBGA package (including 36 Thermal Ground pins at the center)
- 3.3V power, 3.3V I/O with 5V tolerance

3. SYSTEM BLOCK DIAGRAM



Data Sheet: ACD82112

INTRODUCTORY

ACD Confidential. Do Not Reproduce. Use under Non-Disclosure agreement only.

4. SYSTEM DESCRIPTION

The ACD82112 is a single chip implementation of a 12-port Fast Ethernet switch. Together with external SRAM devices and transceiver devices, it can be used to build a complete 10/100 Mbps Fast Ethernet switch. Each individual port can be either auto-sensing or manually selected to run at 10 Mbps or 100 Mbps speed rates and under Full or Half-duplex mode.

The ACD82112 Ethernet switch contains three major functional blocks: the Media Access Controller (MAC), the Queue Manager, and the Lookup Engine.

There are 12 independent MACs within the ACD82112. The MAC controls the receiving, transmitting, and deferring process of each individual port, in accordance to the IEEE 802.3 and 802.3u standards. The MAC logic also provides framing, FCS checking, error handling, status indication and flow control functions. Each MAC interfaces with an external transceiver through a standard MII interface.

The device utilizes ACD's proprietary BASIQ (Bandwidth Assured Switching with Intelligent Queuing) technology. It is a technology to efficiently enforce the first-in-first-out rule of Ethernet Bridge-type devices. The technology enables a true non-blocking frame switching operation at wire speeds for a high throughput and high port density Ethernet switch.

The on-chip Lookup Engine implements a 2,048 entries MAC address lookup table. It maps each destination address into a corresponding port. Each MAC address is automatically learned by the LOOKUP ENGINE after an error-free frame is received. Therefore, the ACD82112 alone can be used to build a complete Fast Ethernet switch with up to 2,048 host connections. *(For detailed information about the built-in ARL, please refer to the ACD80800 data sheet.)*

For workgroup or backbone switches, the ACD82112 can support more MAC addresses per port through the use of an external ARL chip, the ACD80800. The ACD82112 has a glueless ARL interface that allows a supporting chip (ACD80800) to provide up to 11K MAC addresses per switch. System designers can also use this ARL interface to implement a vendor-specific address resolution algorithm.

The ACD82112 provides management support through its MIB (Management Information Base) interface. The MIB interface can be used to monitor all traffic activities of the switch system. The supporting chip (the ACD80900) provides a full set of statistical counters to support both SNMP and RMON network management functions. System designers can also use the MIB interface to implement vendor-specific network management functionality.

Among the 12 MII interfaces, 5 of them can be configured as reversed MII, to connect directly with stand-alone MAC controller devices. A MAC in the ACD82112 can be viewed logically as a PHY device if it is configured as the reversed MII interface. Reversed MII is intended for a CPU network interface, or an expansion port interface.

The System CPU can access various registers inside the ACD82112 through a serial CPU management interface. The CPU can configure the switch by writing into the appropriate registers, or retrieve the status of the switch by reading the corresponding registers. The CPU can also access the registers of external transceiver (PHY) devices through the CPU management interface.

5. FUNCTIONAL DESCRIPTION

The MAC controller performs transmit, receive, and defer functions, in accordance to the 802.3 and 802.3u specification. The MAC logic also handles frame detection, frame generation, error detection, error handling, status indication and flow control functions. Under full-duplex mode, the flow control is implemented in compliance with IEEE 802.3x standard.

Frame Format

The ACD82112 assumes that the received data packet will have the following format:

Preamble	SFD	DA	SA	Type/Len	Data	FCS

Where,

Preamble is a repetitive pattern of '1010....' of any length with nibble alignment.

The *SFD* (Start Frame Delimiter) is defined as an octet pattern of 10101011.

The *DA* (Destination Address) is a 48-bit field that specifies the MAC address of the destined DTE. If the first bit of DA is 1, the ACD82112 will treat the frame as a broadcast/multicast frame and will forward the frame to all ports within the source port's VLAN except the source port itself or BPDU address.

The *SA* (Source Address) is a 48-bit field that contains the MAC address of the source DTE that is transmitting the frame to the ACD82112. After a frame is received with no error, the SA is learned as the port's MAC address.

The *Type/Len* field is a 2-byte field that specifies the type (DIX Ethernet frame) or length (IEEE 802.3 frame) of the frame. The ACD82112 does not process this information.

The *Data* is the encapsulated information within the Ethernet Packet. The ACD82112 does not process any of the data information in this field.

The *FCS* (Frame Check Sequence) is a 32-bit field of CRC (Cyclic Redundancy Check) value based on the destination address, the source address, the type/length and the data field. The ACD82112 will verify the FCS field for each frame. The procedure for computing FCS is described in the section "FCS Calculation."

Start of Frame Detection

When a port's MAC is idle, assertion of the RXDV in the MII interface will cause the port to go into the receive state. The MII presents the received data in 4-bit nibbles that are synchronous to the receive clock (25Mhz or 2.5MHz). The ACD82112 will convert this data into a serial bit stream, and attempt to detect the occurrence of the SFD (10101011) pattern. All data prior to the detection of SFD are discarded. Once SFD is detected, the following frame data are forwarded and stored in the buffer of the switch.

Frame Reception

Under normal operating conditions, the ACD82112 expects a received frame to have a minimum inter frame gap (IFG). The minimum IFG required by the device is 64 BT.

In the event the ACD82112 receives a packet with IFG less than 64 BT, the ACD82112 does not guarantee to be able to receive the frame. The packet will be dropped if the ACD82112 cannot receive the frame.

The device will check all received frames for errors such as symbol error, FCS error, short event, runt, long event, jabber, etc. Frames with any kind of error will not be forwarded to any port.

Preamble Bit Processing

The preamble bit in the header of each frame will be used to synchronize the MAC logic with the incoming bit stream. The minimum length of the preamble is 0 bits and there is no limitation on the maximum length of preamble. After the receive data valid signal RXDV is asserted by the external PHY device, the port will wait for the occurrence of the SFD pattern (10101011) and then start a frame receiving process.

Source Address and Destination Address

After a frame is received by the ACD82112, the embedded destination address and source address are retrieved. The destination address is passed to the lookup table to find the destination port. The source address is automatically stored into the address lookup table. For applications that uses an external ARL, the ACD82112 will disable the internal lookup table and pass the DA and SA to the external ARL for address lookup and learning.

During the receive process, the Lookup Engine will attempt to match the destination address with the addresses stored in the address table. If there is a match found, a link between the source port and the destination port is then established. If an external ARL is used, the ACD82112 indicates the presence of 48-bit DA through the status line of the ARL interface. The external ARL will use the value of DA for address comparison and return a result of the lookup to the ACD82112.

Frame Data

Frame data are transparent to the ACD82112. The ACD82112 will forward the data to destination port(s) without interpreting the content of the frame data field.

FCS Calculation

Each port of the ACD82112 has a CRC checking logic to verify if the received frame has a correct FCS value. A wrong FCS value is an indication of a fragmented frame or a frame with frame bit error. The method of calculating the CRC value is using the following polynomial,

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1$$

as a divider to divide the bit sequence of the incoming frame, beginning with the first bit of the destination address field, to the end of the data field. The result of the calculation, which is the residue after the polynomial division, is the value of the frame check sequence. This value should be equal to the FCS field appended at the end of the frame. If the value does not match the FCS field of the frame, the Frame Bit Error LED of the port will be turned on once and the packet will be dropped.

Illegal Frame Length

During the receiving process, the MAC will monitor the length of the received frame. Legal Ethernet frames should have a length of not less than 64 bytes and no more than 1518 bytes. If the carrier-sense signal of a frame is asserted for less than 76 BT, the frame is flagged with short event error. If the length of a frame is less then 512 BT, the frame is flagged with runt error.

In order to support an application where extra byte length is required, an Extra long frame option is provided. When the Extra long frame option is enabled *(Table-7.24, bit-7)*, only frames longer than 1530 bytes are marked with a long event error. Frame length is measured from the first byte of DA to the last byte of FCS.

Frame Filtering

Frames with any kind of error will be filtered. Types of error include code error (indicated by assertion of RXER signal), FCS error, alignment error, short event, runt, and long event.

Any frame heading to its own source port will be filtered.

When external ARL is used, the filtering decision will be made by the ARL. The ACD82112 will act in accordance with the ARL's direction.

If the *Spanning Tree Support* option is enabled, frames containing DA equal to any reserved Bridge Management Group Address specified in Table 3.5 of the IEEE 802.1D will not be forwarded to any ports, except Port-11, which may receive BPDU frames. If spanning tree support is not enabled, frames with DA equal to the reserved Group Address for PBDU will be broadcasted to all ports in the same VLAN of the source port.

Jabber Lockup Protection

If a receiving port is active continuously for more than 50,000 bit times, the port is considered to be jabbering. A jabbering port will automatically be partitioned from the switch system in order to prevent it from impairing the performance of the network. The partitioned port will be re-activated as soon as the offending signal discontinues.

Excessive Collision

In the event that there are more than 16 consecutive collisions, the ACD82112 will reset the counter to zero and retransmit the packet. This implementation insures there is no packet loss even under channel capture situation. However, the ACD82112 has an option to drop the packet on excessive collisions. When this option is enabled *(Table-7.24, bit-11)*, the frame will be dropped after 16 consecutive collisions.

If a port has encountered 256 consecutive collisions, it is assumed to be non-functional and will be partitioned. The partitioned port will not receive any frame. It will still transmit new frames, but without retry after encountering a collision. The partition port will be released once a new frame is transmitted without incurring a collision, which indicates that, the port has regained normal functions.

False Carrier Events

If the signal in the MII interface is asserted but the receive data valid (RXDV) signal is not, and the RXD shows 1110 at the same time, the port is considered to have a false carrier event. If a port has more than two consecutive false carrier events, the port will automatically be partitioned from the switch system. The partitioned port will be re-activated if it has been idling for 33,000 BT or it has received 500 bits of valid data after a minimum 64 BT idle period.

Frame Forwarding

If the first bit of the destination address is 0, the frame is handled as a unicast frame. The destination address is passed to the Address Resolution Logic; which returns a destination port number to identify which port the frame should be forwarded to. If the Address Resolution Logic cannot find any match for the destination address, the frame will be treated as a frame with unknown DA. The frame will be processed in one of two ways. If the option flood-to-all-port is enabled, the switch will forward the frame to all ports within the same VLAN of the source port, except the source port itself. If the option is not enabled, the frame will be forwarded to the 'dumping port' of the source port VLAN only. The dumping port is determined by the VLAN ID of the source port. If the source port belongs to multiple VLANs, a frame with unknown DA will then be forwarded to multiple dumping ports of the VLANs.

If the first bit of the destination address is a 1, the frame is handled as a multicast or broadcast frame. The ACD82112 does not differentiate a multicast packet from a broadcast packet except for the reserved bridge management group address, as specified in Table 3.5 of IEEE 802.1D standard. The destination ports of a broadcast frame are all ports within the same VLAN except the source port itself.

The order of all broadcast frames with respect to the unicast frames is strictly enforced by the ACD82112.

Frame Transmission

The ACD82112 transmits all frames in accordance to IEEE 802.3 standard. The ACD82112 will send the frames with a guaranteed minimum interframe gap of 96 BT, even if the received frames have an IFG less than the minimum requirement. Before the transmit process is started, the MAC logic will check if the channel has been silent for more than 64 BT. Within the 64 BT silent window, the transmission process will defer on any receiving process. If the channel has been silent for more than 64 BT, the MAC will wait an additional 32 BT before starting the transmit process. In the event that the carrier sense signal is asserted by the MII during the wait period, the MAC logic will generate a JAM signal to cause a forced collision.

The MAC logic will abort the transmit process if a collision is detected through the assertion of the Col signal of the MII. Re-transmission of the frame is scheduled in accordance to the IEEE 802.3's truncated binary exponential backoff algorithm. If the transmit process has encountered 16 consecutive collisions, an excessive collision error is reported, and the ACD82112 will try to re-transmit the frame, unless the drop-on-excessive-collision option of the port is enabled. It will first reset the number of collisions to zero and then start the transmission after a 96 bit time of inter frame gap. If drop-on-excessive-collision is enabled, the ACD82112 will not try to re-transmit the frame after 16 consecutive collisions. If collision is detected after 512 BT of the transmission, a late collision error will be reported, but the frame will still be retransmitted after proper backoff time.

Frame Generation

During a transmit process, frame data is read out from the memory buffer and is forwarded to the destination port's PHY device in nibbles. 7 bytes of a preamble signal (10101010) will be generated first followed by the SFD (10101011), and then the frame data and 4 bytes of FCS are sent out at last.

Shared Buffer

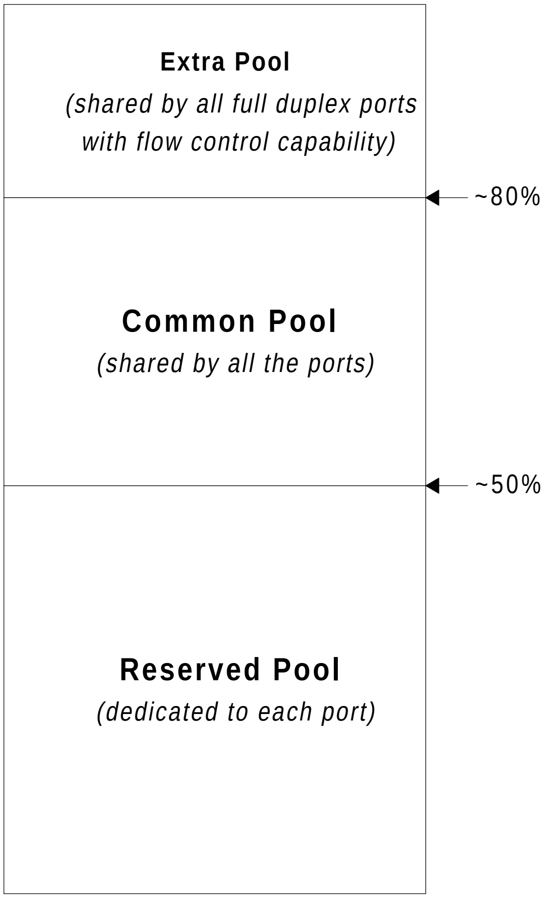
All ports of the ACD82112 work in Store-And-Forward mode so that all ports can support both 10Mbps and 100Mbps data speeds. The ACD82112 utilizes a global memory buffer pool, which is shared by all ports. The device has a unique architecture that inherits the advantage of both output buffer-based and input buffer-based switches. The output buffer-based switches store the received data only once into the memory, and hence has a short latency. Whereas input buffer based switches typically have more efficient flow control.

Starvation Control Scheme

All frames received by the ACD82112 will be stored into a common physical frame buffer pool. In order to make sure all ports have fair access to the network, a buffer allocation scheme (starvation control) is used to prevent active ports from occupying all the buffers and starve off the less active ports.

The frame buffer pool is divided into 3 portions: the *reserved poo*l, the *common pool* and the *extra pool*, as shown in *Figure-5.1:*

Figure-5.1: Buffer Partition


Extra Pool
(shared by all full duplex ports
with flow control capability)
~80%
Common Pool
(shared by all the ports)
~50%
Reserved Pool
(dedicated to each port)

The *reserved pool* guarantees each port will have a fair network access possibility, even under the worst traffic congestion situation. It takes about 50% of the total buffer and is evenly allocated to each port as its dedicated buffer slot. The dedicated slot is not shared with other ports.

The *common pool* provides a deep buffer for the busy ports (e.g. server port) to serve multiple low speed ports (e.g. client port) simultaneously. It helps to avoid head-of-line blocking. It takes about 30% of the total buffer and is shared by all ports. It stores the congested traffics before the flow control mechanism is triggered.

The *extra pool* is reserved only for ports with pause frame based flow control capacity. It takes the remaining 20% of the total buffer. It is used to minimize the chance of frame dropping by buffering for the latency of the pause frame based flow control scheme. It is used only after a flow control mechanism is triggered.

Flow Control Scheme

Flow control activity is triggered when the buffer utilization exceeds certain thresholds specified by the dedicated registers. *Register-10* is used to specify the Upper and the Lower Thresholds of the reserved buffer slot for each port. *Register-11* is used to specify the Upper and the Lower thresholds of the broadcast queue.

Under full duplex operation, if the buffer utilization of a port has exceeded the upper threshold of the reserved buffer slot, and the common pool has been used up, *a max-pause-frame* (a pause frame with a maximum time interval of FFFFh) will be sent to the sending party to stop it from sending new frames. If pause-frame based flow control is not enabled at that port, the frame will be dropped. Once a *max-pause-frame* is sent, if the utilization of the reserved buffer slot of the port drops below the lower threshold, a *mini-pause-frame* (a pause frame with minimum time interval of 0) will be sent to the linking party to enable new frame transmission.

Under half duplex operation, if the buffer utilization of a port has exceeded the upper threshold of the reserved buffer slot, and the common pool has been used up, the port will execute back-pressure based flow control by sending a jam pattern on each incoming frame. If backpressure flow control of the port is not enabled, the frame will be dropped.

If the broadcast flow control is enabled (when bit-13 of register-25 is set), flow control will be triggered when the broadcast queue is longer than the upper threshold specified by *Register-11*. All full duplex ports with pause-frame capability will send a *max-pause-frame* to its linking party. All half-duplex ports with backpressure capability will jam incoming frames. After a max-pause-frame is sent, and if the broadcast queue is shorten below the lower threshold specified by *Register-11*, a mini-pause-frame will be sent to release the hold on transmission.

VLAN Support *(Registers 23 & 24)*

The ACD82112 can support up to 4 port-based security VLANs. Each port of the ACD82112 can be assigned to up to four VLAN. On power up, every port is assigned to VLAN-I as the default VLAN. Frames from the source port will only be forwarded to destination ports within the same VLAN domain. A broadcast/multicast frame will be forwarded to all ports within the VLAN(s) of the source port. A unicast frame will be forwarded to the destination port only if the destination port is in the same VLAN as the source port. Otherwise, the frame will be treated as a frame with unknown DA. Each VLAN can be assigned with a dedicated dumping port. Multiple VLANs can also share a dumping port. Unicast frames with unknown destination addresses will be forwarded to the dumping port of the source port VLAN.

Security VLAN can be disabled by setting the corresponding bit in the system configuration register (bit 8 of *Register 16, see Table 7.15*). When security VLAN is disabled, each VLAN becomes a Leaky VLAN and is equivalent to a broadcast domain. Four dumping ports of four different Leaky VLANs can be grouped together to form a fat pipe uplink (for example, port 0, port 1, port 2, and port 3 can be grouped to form an 800 Mbps uplink port). When multiple dumping ports are grouped as a single pipe, each port has to be assigned to one and only one VLAN. A unicast frame with a matched DA will be forwarded to any destination port, even if the VLAN ID is different. All unmatched DA packets will be forwarded to the designated dumping port of the source port VLAN. The broadcast and multicast packets will only be forwarded to the ports in the same VLAN of the source port. Therefore, a 200 to 800 Mbit/s pipe can be established by carefully grouping the dumping ports, and directly connecting with any segmentation switches.

Dumping Port

Each VLAN can be assigned with a dedicated dumping port. Multiple VLANs can share a dumping port. Each dumping port can be used for an up-link connection or for a DTE connection. That is, the dumping port can be used to connect the switch with a computer repeater hub, a workgroup switch, a router, or any type of interconnection device compliant with IEEE 802.3 standard. ACD82112 will direct the following frames to the dumping port:

- A frame with a unicast destination address that does not match with any port's source address within the VLAN of the source port.
- A frame with a broadcast/multicast destination address*.

*See *Spanning Tree Support*

If the device is configured to work under Flood-to-All-Port mode *(Register 25, bit 8)*, the frames with unknown DA will be forwarded to all the ports in the VLAN(s) of the source port except the source port itself.

Mode of Operation *(Register 18)*

All ports of the ACD82112 can work in half duplex mode or full duplex mode. If auto-negotiation is enabled, the mode is determined by the PHY device. Otherwise, the mode is assigned by the Full Duplex mode indication/ assignment register.

Spanning Tree Support

The ACD82112 supports the Spanning Tree protocol. When Spanning Tree Support is enabled *(Register-16 bit 1, see Table 7.15)*, frames from the CPU port (port 11) having a DA value equal to the reserved Bridge Management Group Address for BPDU will be forwarded to the port specified by the CPU. Frames from all other ports with a DA value equal to the Reserved Group Address for BPDU will be forwarded to the CPU port if the port is in the same VLAN of the CPU port. Port 11 is designed as the default CPU port. When Spanning Tree Support is disabled, all reserved group addresses for Bridge Management is treated as broadcast address, with the exception of the reserved multicast addresses for pause frame specified by IEEE802.3x.

Every port of the ACD82112 can be set to block-and-listen mode through the CPU interface. In this mode, incoming frames with a DA value equal to the reserved Group Address for BPDU will be forwarded to the CPU port. Incoming frames with all other DA values will be dropped. Outgoing frames with a DA value equal to the Group Address for BPDU will be forwarded to the attached PHY device; all other outgoing frames will be filtered.

Queue Management

Each port of the ACD82112 has its own individual transmission queue. All frames coming into the ACD82112 are stored into the shared memory buffer, and are lined up in the transmission queues of the corresponding destination port. The order of all frames, unicast or broadcast, is strictly enforced by the ACD82112. The ACD82112 is designed with a non-blocking switching architecture. It is capable of achieving wire speed forwarding rates and can handle maximum traffic loads.

MII Interface

The MAC of each port of the ACD82112 interfaces with the port's PHY device through the standard MII interface. For reception, the received data (RXD) is sampled by the rising edge of the receive clock (RXCLK). Assertion of the receive data valid (RXDV) signal will cause the MAC to look for the start of SFD. For transmission, the transmit data enable (TXEN) signal is asserted when the first preamble nibble is sent on the transmit data (TXD) lines. The transmit data are clocked out by the falling edge of the transmit clock (TXCLK).

PHY Management

The ACD82112 supports PHY device management through the serial MDIO and MDC signal lines. The ACD82112 can continuously poll the status of the PHY devices through the serial management interface, without CPU intervention. The ACD82112 will also configure the PHY capability field to ensure proper operation of the link. The ACD82112 also enables the CPU to access any registers in the PHY devices through the CPU interface. The ID of the PHY device can start from either "1" or "4", depending on the setting of bit-10 of register-25.

Reversed MII Interface

Five of the ACD82112's 12 ports can be configured with reversed MII interface. Reversed MII behaves as a PHY MII: the TXCLK, COL, RXD<3:0>, RXCLK, RXDV, CRS signals (names specified by IEEE 802.3u) become output signals of the ACD82112, and the TXER, TXD<3:0>, TXEN, RXER signals (names specified by IEEE 802.3u) become input signals of the ACD82112. The Reversed MII interface enables an external MAC device to be connected directly with the ACD82112.

SRAM Interface

The ACD82112 requires the use of ASRAM, or Flow-through SSRAM as a memory buffer. Each read or write cycle takes up to 20 ns. For ASRAM, the access time should be at or less than 12 ns. For SSRAM, the speed should be at or higher than 100MHz. The SRAM interface contains a 52-bit data bus, a 17-bit address bus and 4 chip-select signals.

CPU Interface

The ACD82112 does not require any microprocessor for operation. Initialization and most configurations can be done with the pull-up or pull-down of designated hardware pins. A CPU interface is provided for a microprocessor to access the control registers and status registers. The microprocessor can send a read command to retrieve the status of the switch, or send a write command to configure the switch through the interface. The interface is a commonly used UART type interface. The CPU interface can also be used to access the registers inside each PHY device connected with the ACD82112.

ARL Interface

The ACD82112 has a built-in MAC address storage for up to 2,048 source addresses. If more than 2,048 addresses are needed, an external ARL (e.g. ACD80800) can be used to expand the address space to 11K entries.

The external ARL is connected through the ARL interface *(Table-7.24, bit-9)*. It can tap the value of DA out of the memory interface bus, and execute a lookup process to map the value of the DA into a port number. It can also learn the SA values embedded in the received frames. The value of SA is used to build the address lookup table inside the ACD80800.

MIB Interface

Traffic activities on all ports of the ACD82112 can be monitored through the MIB interface. Through the MIB interface, a MIB device can view the frames transmitted from or received by any port. Therefore, the MIB device can maintain a record of traffic statistics for each port to support network management. Since all received data are stored into the memory buffer, and all transmitted data are retrieved from the memory buffer, the data of the activities can also be captured from the memory interface data bus. The status of each data transaction between the ACD82112 and the SRAM is displayed by dedicated status signal pins of the ACD82112.

LED Interface

The ACD82112 provides a wide variety of LED indicators for simple system management. The update of the LED is completely autonomous and merely requires low speed TTL or CMOS devices as LED drivers. The status display is designed to be flexible to allow the system designer to choose those indicators appropriate for the specification of the equipment.

There are two LED control signals, LEDVLD0 and LEDVLD1. They are used to indicate the start and end of the LED data signal presented on nLED0-nLED3. The LEDCLK signal is a 2.5MHz clock signal. The rising edge of LEDCLK should be used to latch the LED data signal into the LED driver circuitry.

The LED data signals contain Lnk, Xmt, Rcv, Col, Err, Fdx and Spd, which represent Link status, Transmit status, Receive status, Collision indication, Frame error indication, Full duplex operation and Operational Speed status respectively. These status signals are sent out sequentially from port 11 to port 0, once every 50ms. For details about the timing diagrams of the LED signals, refer to the chapter of “Timing Description ”

Life Pulse

The ACD82112 continuously sends out life pulses to the WCHDOG pin when it is operating properly. In a catastrophic event, the ACD82112 will not send the life pulse to cause the external watchdog circuitry to timeout and reset the switch system.

6. INTERFACE DESCRIPTION

MII Interface (MII)

The ACD82112 communicates with the external 10/100 Ethernet transceivers through standard MII interface. The signals of MII interface are described in *Table-6.1*:

Table-6.1: MII Interface Signals

Name	Type	Description
PxCRS	I	Carrier sense
PxRXDV	I	Receive data valid
PxRXCLK	I	Receive clock (25/2.5 MHz)
PxRXERR	I	Receive error
PxRXD0	I	Receive data bit 0
PxRXD1	I	Receive data bit 1
PxRXD2	I	Receive data bit 2
PxRXD3	I	Receive data bit 3
PxCOL	I	Collision indication
PxTXEN	O	Transmit data valid
PxTXCLK	I	Transmit clock (25/2.5 MHz)
PxTXD0	O	Transmit data bit 0
PxTXD1	O	Transmit data bit 1
PxTXD2	O	Transmit data bit 2
PxTXD3	O	Transmit data bit 3

For MII interface, signal PxRXDV, PxRXER and PxRXD0 through PxRXD3 are sampled by the rising edge of PxRXCLK. Signal PxTXEN, and PxTXD0 through PxTXD3 are clocked out by the falling edge of PxTXCLK. The detailed timing requirement is described in the chapter of "Timing Description"

Ports 0, 1, 2, 3 and 11 can be configured as reversed MII ports (*Register 28*, the Reversed MII Enable register). These ports, when configured as "normal" MII, have the same characteristics as all other MII ports. However, when configured as reversed MII interface, they will behave logically like a PHY device, and can interface directly with a MAC device. The signal of reversed MII interface are described by *Table-6.2*:

Table-6.2: Reversed MII Interface Signals

Name	Type	Description
PxCRSR	O	Carrier sense
PxRXDVR	I	Transmit data valid
PxRXCLKR	O	Transmit clock (25/2.5 MHz)
PxRXERR	I	Transmit-Not-Ready
PxRXD0R	I	Transmit data bit 0
PxRXD1R	I	Transmit data bit 1
PxRXD2R	I	Transmit data bit 2
PxRXD3R	I	Transmit data bit 3
PxCOLR*	O	Collision Indication/ Receive-Not-Ready*
PxTXENR	O	Receive data valid
PxTXCLKR	O	Receive clock (25/2.5 MHz)
PxTXD0R	O	Receive data bit 0
PxTXD1R	O	Receive data bit 1
PxTXD2R	O	Receive data bit 2
PxTXD3R	O	Receive data bit 3

**Collision Indication for half-duplex, Receive-Not-Ready for full duplex only.*

For reversed MII interface, signal PxRXDVR, and PxRXD0R through PxRXD3R are clocked out by the falling edge of PxRXCLKR. Signal PxTXENR, and PxTXD0R through PxTXD3R can be sampled by the falling edge or rising edge of PxTXCLKR, depends on the setting of bit 9 of *Register-16*. The timing behavior is described in the chapter of "Timing Description."

PHY Management Interface

All control and status registers of the PHY devices are accessible through the PHY management interface. The interface consists of two signals: MDC and MDIO, which are described in *table-6.3*.

Table-6.3: PHY Management Interface Signals

Name	Type	Description
MDC	O	PHY management clock (1.25MHz)
MDIO	I/O	PHY management data

Frames transmitted on MDIO has the following format (*Table-6.4*):

Table-6.4: MDIO Format

Operation	PRE	ST	OP	PHY-ID	REG-AD	TA	DATA	IDLE
Write	1…1	01	01	aaaaa	rrrrr	10	d…d	Z
Read	1…1	01	10	aaaaa	rrrrr	Z0	d…d	Z

Data Sheet: ACD82112

INTRODUCTORY

Table-6.6: CPUDI Format

Operation	Command	Address	Index	Data	Checksum
Write	0010XX11	8-bit	8-bit	24-bit	8-bit
Read	0010XX01	8-bit	8-bit	24-bit	8-bit

Prior to any transaction, the ACD82112 will output thirty-two bits of '1' as preamble signal. After the preamble, a 01 signal is used to indicate the start of the frame.

For a write operation, the device will send a '01' to signal a write operation. Following the '01' write signal will be the 5 bit ID address of the PHY device and the 5 bit register address. A '10' turn around signal is then followed. After the turn around, the 16 bit of data will be written into the register. After the completion of the write transaction, the line will be left in a high impedance state.

For a read operation, the ACD82112 will output a '10' to indicate read operation after the start of frame indicator. Following the '10' read signal will be the 5-bit ID address of the PHY device and the 5-bit register address. Then, the ACD82112 will cease driving the MDIO line, and wait for one bit time. During this time, the MDIO should be in a high impedance state. The ACD82112 will then synchronize with the next bit of '0" driven by the PHY device, and continue on to read 16 bits of data from the PHY device.

The system designer can set the ID of the PHY devices as 1 for port 0, 2 for port 1, … and 12 for port 11, when the PHYID option (Bit-10 of Register-25) is set to "0". If the PHYID option is set to "1", the corresponding PHY ID should set to 4 through 15. The detailed timing requirements on PHY management signals are described in the chapter of "Timing Description."

CPU Interface

The ACD82112 includes a CPU interface to enable an external CPU to access the internal registers of the ACD82112. The signal used in the CPU is UART. The baud rate can be from 1200 bps to 76800 bps. The ACD82112 automatically detects the baud rate for each command, and returns the result at the same baud rate. The signals in CPU interface are described in *Table-6.5*.

Table-6.5: CPU Interface Signals

Name	Type	Description
CPUDI	I	CPU data input
CPUDO	O	CPU data output
CPUIRQ	O	CPU interrupt request

A command sent by CPU comes through the CPUDI line. The command consists of 8 octets. Command frames transmitted on CPUDI have the following format (*Table-6.6*):

The byte order of data in all fields follows the big-endian convention, i.e. most significant octet first. The bit order is the least significant order first. The Command octet specifies the type of the operation. Bit 2 and bit 3 of the command octet is used to specify the device ID of the chip. They are set by bit 16 and bit 17 of the *Register 25* at power on strobing. The address octet specifies the type of the register. The index octet specifies the index of the register in a register array. For write operation, the Data field is a 3-octet value to specify what to write into the register. For read operation, the Data field is a 3-octet 0 as padded data. The checksum value is an 8-bit value of exclusive-OR of all octets in the frame, starting from the Command octet.

For each valid command heading to it, the ACD82112 will always send a response. Response from the ACD82112 is sent through the CPUDO line. Response frames sent by the ACD82112 has the following format (*Table-6.7*):

Table-6.7: Switch Response Format

Response	Command	Result	Data	Checksum
Write	00100011	8-bit	24-bit	8-bit
Read	00100001	8-bit	24-bit	8-bit

The command octet specifies the type of the response. The result octet specifies the result of the execution.

The Result field in a response frame is defined as:

- 00 for no error
- 01 for Checksum
- 10 for address incorrect
- 11 for MDIO waiting timeout

For response to a read operation, the Data field is a 4-octet value to indicate the content of the register. For response to a write operation, the Data field is 32 bits of 0. The checksum value is an 8-bit value of exclusive-OR of all octets in the response frame, starting from the Command octet.

CPUIRQ is used to notify the CPU that some special status has been encountered by the ACD82112, like port partition, fatal system error, etc. By clearing the appropriate bit in the interrupt mask register, one can stop the specific source from generating an interrupt request. Reading the interrupt source register retrieves the source of the interrupt request and clears the interrupt source register.

SRAM Interface

All received frames are stored into the shared memory buffer through the SRAM interface. When the destination port is ready to transmit the frame, data is read from the shared memory buffer through the SRAM interface. The signals in SRAM interface are described in *Table-6.8*.

Table-6.8: SRAM Interface

Name	Type	Description
DATA0 - DATA51	I/O	memory data bus
ADDR0 - ADDR16	O	memory address bus
nOE	O	output enable, low active
nWE	O	write enable, low active
nCS0 - nCS3	O	chip select signals, low active.

Data is written into the SRAM or read from the SRAM in 52-bit wide words. The data is a 48 bit wide value and the control is a 4 bit wide value. ADDR specifies the address of the word, and DATA contains the content of the word. Bit 0 ~ 47 of DATA bus are used to pass 48-bit frame data. Bit 48 are used to indicate the start and end of a frame. Bit 49~51 are used to indicate the length of the data shown on first 48 bit of DATA bus.

nOE and nWE are used to control the timing of read or write operation respectively. nCSx selects the SRAM chip corresponding to the word address. The timing requirement on SRAM access is described in "Timing Description" (Chapter-9).

ARL Interface

The ARL interface provides a communication path between the ACD82112 and an ARL device, which can provide up to 11K of address lookup. As the ACD82112 receives a frame, the destination address and source address of the frame are displayed on the ARLDO data lines for the external ARL device. After the external ARL finds the corresponding destination port, it returns the result through the ARLDIx lines to the ACD82112. The timing requirement on ARL signals is described in Chapter 9, "Timing Description." *Table-6.9* shows the associated signals in ARL interface.

Table-6.9: ARL Interface Signals

Name	Type	Description
ARLDO0-RLDO51	O	ARL data output, shared with DATA 0 - DATA 51
ARLDIR1-ARLDIR0	O	ARL data direction indicator 00 for idle 01 for receive 10 for transmit 11 for control
ARLSYNC	O	ARL port synchronization
ARLSTAT0-ARLSTAT3	O	ARL data state indicator
ARLCLK	O	ARL clock
ARLDI0 - ARLDI3	I	ARL data input
ARLDIV	I	ARL input data valid

The data signal is tapped from the DATA bus of the SRAM interface. Since all data of the received frames will be written into the shared memory through the DATA bus, the bus can be used to monitor occurrences of DA and SA values, indicated by the status signal of ARLSTAT. Therefore, ARLD0 through ARLD51 are the same signals of DATA0 through DATA51.

The ARLDIR1 and ARLDIR0 are used to indicate the direction of data on the ARLDO bus:

- 00: Idle
- 01: For receiving data
- 10: For transmitting data
- 11: Header

The ARLSYNC is used to indicate port 0 is driving the DATA bus. Since the bus is pre-allocated in time division multiplexing manner, the ARL device can determine which port is driving the DATA bus.

The ARLSTAT are used to indicate the status of the data shown on the first 48 bit of DATA bus. The 4-bit status is defined as:

- 0000 - Idle
- 0001 - First word (DA)
- 0010 - Second word (SA)
- 0011 - Third through last word
- 0100 - Filter Event
- 0101 - Drop Event
- 0110 - Jabber
- 0111 - False Carrier/Deferred Transmission*
- 1000 - Alignment error/Single Collision*
- 1001 - Flow Control/Multiple Collision*
- 1010 - Short Event/Excessive Collision*
- 1011 - Runt/Late Collision*
- 1100 - Symbol Error
- 1101 - FCS Error

Table-6.10: LED Interface Signals

Name	Type	Description	Signal Group 1	Signal Group 2
LEDVLD0	O	LED signal valid #0	1	0
LEDVLD1	O	LED signal valid #1	0	1
nLEDCLK	O	2.5 MHz LED clock	-	-
nLED0	O	Dual purpose indicator	address learning status	frame error indicator
nLED1	O	Dual purpose indicator	full duplex indication	collision indication
nLED2	O	Dual purpose indicator	port speed (1=10Mbps,0=100Mbps)	receiving activity
nLED3	O	Dual purpose indicator	Link status	transmit activity

- 1110 - Long Event
- 1111 - Reserved

*Note: error type depends on the port is receiving or transmitting.

ARLDIx is used to receive the lookup result from the external ARL. The result is returned by external ARL device through the ARLDIx lines. Returned data can be sampled by the rising or the falling edges of ARLCLK, depending on the setting of Bit-18 of Register-25. The ARL results have the following format:

SID	RSLT	DID

Where

- SID is a 5-bit ID of the source port (0 - 11)
- RSLT is a 2-bit result, defined as:
  - 00 - reserved
  - 01 - matched
  - 10 - not matched
  - 11 - forced discard
- DID is a 5-bit ID of the destination port (0 - 11)

The start of each ARL result is indicated by assertion of ARLDIV signal.

LED Interface

The signals in the LED interface is described in *Table-6.10*:

The status of each port is displayed on the LED interface for every 50ms. LEDVLD0 and LEDVLD1 are used to indicate the start and end of the LED data. LED data is clocked out by the falling edge of LEDCLK, and should be sampled by the rising edge of LEDCLK. LED data of port-11 are clocked out first, followed by port-10 down to port-0. All LED signals are low active.

Configuration Interface

The default values of certain register bits are set by internal pull-high/pull-low with 75K Ohm resistors. These default values can be overwritten by external pull-high/pull-low of certain designated pins with 4.7K Ohm external resistors. *Table-6.11* lists all the available pins. The meanings of the register bits are described in the chapter of "Register Description."

Table-6.11: Configuration Interface

Pin Name	Register #	Bit #	Default
P7TXD0	25	0	See Table-7.25
P7TXD1		1	
P7TXD2		2	
P7TXD3		3	
P6TXD0		4	
P6TXD1		5	
P6TXD2		6	
P6TXD3		7	
LEDCLK		8	
LEDVLD0		9	
LEDVLD1		10	
nLED3		11	
nLED2		12	
nLED1		13	
nLED0		14	
P5TXD0		15	
P5TXD1		16	
P5TXD2		17	
P5TXD3		18	
P0TXD0	30	0	See Table-7.30
P0TXD1		1	
P0TXD2		2	
P0TXD3		3	
P1TXD0		4	
P1TXD1		5	
P1TXD2		6	
P1TXD3		7	
P2TXD3	20, inside the *Internal* ARL	3	0
P2TXD2		1	1

Table-6.12: Other Interface

Name	Type	Description
CLK50	I	50 MHz clock input
nRESET	I	hardware reset
WCHDOG	O	watch dog life pulse signal
VDD	-	3.3 V power
VSS	-	ground

Other Interface (*Table-6.12*)

The CLK50 should come from a clock oscillator, with 0.01% (100 ppm) accuracy.

The nRESET is a low-active hardware reset pin. Assertion of this pin will cause the ACD82112 to go through the power-up initialization process. All registers are set to their default value after reset.

The WCHDOG pin is used to handle exceptional cases. A normal working ACD82112 sends out continuous life pulses from the WCHDOG pin, which can be monitored by an external watchdog circuit. If no life pulse is detected , the external watchdog circuit may force reset of the switch system. It is a safeguard against unforeseeable situations.

The VDD is 3.3V power supply.

The VSS is power ground.

7. REGISTER DESCRIPTION

Registers in the ACD82112 are used to define the operation mode of various function modules of the switch controller and the peripheral devices. Default values at power-on are defined by the factory. The management CPU (optional) can read the content of all registers and modify some of the registers to change the operation mode. Table-7.0 lists all the registers inside the switch controller.

INTSRC register (register 1)

The INTSRC register indicates the source of the interrupt request. Before the CPU starts to respond to an interrupt request, it should read this register to find out the interrupt source. This register is automatically cleared after each read. Table-7.1 lists all the bits of this register.

Table-7.1: INTSRC Register

Bit	Description	Default
0	System initialization completed	0
1	System error occurred	
2	Port partition occurred	
3	ARL Interrupt	
4	Reserved	
5		
6		
7		

SYSERR register (register 2)

The SYSERR register indicates the presence of system errors. It is automatically cleared after each read. Table-7.2 lists all kind of system error.

Table-7.0: Register List

Address	Name	Type	Size	Depth	Description
0	Reserved	R	-	-	-
1	INTSRC	R	8 Bit	1	Interrupt Source
2	SYSERR	R	12 Bit	1	System Error
3	PAR	R	12 Bit	1	Port Partition Indication
4	PMERR	R	12 Bit	1	PHY Management Error
5	ACT	R	12 Bit	1	Port Avtivity
6	Reserved	R	-	-	-
7	Reserved	R	-	-	-
8	SAL	R/W	24 Bit	1	Source Address, bit 23:0
9	SAH	R/W	24 Bit	1	Source Address, bit 47:24
10	UTH	R/W	16 Bit	1	Unicast Threshold
11	BTH	R/W	16 Bit	1	Broadcast Threshold
12	MAXL	R/W	16 Bit	1	FCS of Max-Pause-Frame, bit 15:0
13	MAXH	R/W	16 Bit	1	FCS of Max-Pause-Frame, bit 31:16
14	MINL	R/W	16 Bit	1	FCS of Min-Pause-Frame, bit 15:0
15	MINH	R/W	16 Bit	1	FCS of Min-Pause-Frame, bit 31:16
16	SYSCFG	R/W	16 Bit	1	System Configuration
17	INTMSK	R/W	8 Bit	1	Interrupt Mask
18	SPEED	R/W	12 Bit	1	Port Speed
19	LINK	R/W	12 Bit	1	Port Link
20	nFWD	R/W	12 Bit	1	Port Forward Disable
21	nBP	R/W	12 Bit	1	Port Back Pressure Disable
22	nPORT	R/W	12 Bit	1	Port Disable
23	PVID	R/W	4 Bit	12	Port VLAN ID
24	VPID	R/W	5 Bit	4	VLAN Dumping Port
25	POSCFG	R/W	20 Bit	1	Power-On-Strobe Configuration
26	PAUSE	R/W	12 Bit	1	Port Pause Frame Disable
27	DPLX	R/W	12 Bit	1	Port Duplex Mode
28	RVSMII	R/W	5 Bit	1	Reversed MII Selection
29	nPM	R/W	12 Bit	1	Port PHY Management Disable
30	ERRMSK	R/W	8 Bit	1	Error Mask
31	CLKADJ	R/W	4 Bit	1	ARL Clock Delay Adjustment
32-43	PHYREG	R/W	16 Bit	*	Pointer to Registers in PHY devices
44-63	Reserved	R/W	-	-	-

Table-7.2: SYSERR Register

Bit	Description	Default
0	BIST failure indication	0
1	Reserved	
2		
3		
4		
5		
6		
7		
8		

PAR register (register 3)

The PAR register indicates the presence of the partitioned ports and the port ID. A port can be automatically partitioned if there is a consecutive false carrier event, an excessive collision or a jabber. This register is automatically cleared after each read. Table-7.3 lists all the bits of this register.

Table-7.3: PAR Register

Bit	Description	Default
0	0 - Port 0 is not partitioned. 1 - Port 0 is partitioned.	0
1	0 - Port 1 is not partitioned. 1 - Port 1 is partitioned.	
2	0 - Port 2 is not partitioned. 1 - Port 2 is partitioned.	
3	0 - Port 3 is not partitioned. 1 - Port 3 is partitioned.	
4	0 - Port 4 is not partitioned. 1 - Port 4 is partitioned.	
5	0 - Port 5 is not partitioned. 1 - Port 5 is partitioned.	
6	0 - Port 6 is not partitioned. 1 - Port 6 is partitioned.	
7	0 - Port 7 is not partitioned. 1 - Port 7 is partitioned.	
8	0 - Port 8 is not partitioned. 1 - Port 8 is partitioned.	
9	0 - Port 9 is not partitioned. 1 - Port 9 is partitioned.	
10	0 - Port 10 is not partitioned. 1 - Port 10 is partitioned.	
11	0 - Port 11 is not partitioned. 1 - Port 11 is partitioned.	

PMERR register (register 4)

The PMERR register indicates the presence of PHYs that have failed to respond to the PHY Management command issued through the MDIO line. This register is automatically cleared after each read. Table-7.4 describes all the bit of this register.

Table-7.4: PMERR Register

Bit	Description	Default
0	0 - Port 0's PHY responded 1 - Port 0’s PHY failed to respond	0
1	0 - Port 1’s PHY responded 1 - Port 1’s PHY failed to respond	
2	0 - Port 2’s PHY responded 1 - Port 2’s PHY failed to respond	
3	0 - Port 3’s PHY responded 1 - Port 3’s PHY failed to respond	
4	0 - Port 4’s PHY responded 1 - Port 4’s PHY failed to respond	
5	0 - Port 5’s PHY responded 1 - Port 5’s PHY failed to respond	
6	0 - Port 6’s PHY responded 1 - Port 6’s PHY failed to respond	
7	0 - Port 7’s PHY responded 1 - Port 7’s PHY failed to respond	
8	0 - Port 8’s PHY responded 1 - Port 8’s PHY failed to respond	
9	0 - Port 9’s PHY responded 1 - Port 9’s PHY failed to respond	
10	0 - Port 10’s PHY responded 1 - Port 10’s PHY failed to respond	
11	0 - Port 11’s PHY responded 1 - Port 11’s PHY failed to respond	

ACT register (register 5)

The ACT register indicates the presence of transmit or receive activities of each port since the register was last read. This register is automatically cleared after each read. Table-7.5 describes all the bits of this register.

Table-7.5: ACT register

Bit	Description	Default
0	0 - Port 0 no activity 1 - Port 0 has activity	0
1	0 - Port 1 no activity 1 - Port 1 has activity	
2	0 - Port 2 no activity 1 - Port 2 has activity	
3	0 - Port 3 no activity 1 - Port 3 has activity	
4	0 - Port 4 no activity 1 - Port 4 has activity	
5	0 - Port 5 no activity 1 - Port 5 has activity	
6	0 - Port 6 no activity 1 - Port 6 has activity	
7	0 - Port 7 no activity 1 - Port 7 has activity	
8	0 - Port 8 no activity 1 - Port 8 has activity	
9	0 - Port 9 no activity 1 - Port 9 has activity	
10	0 - Port 10 no activity 1 - Port 10 has activity	
11	0 - Port 11 no activity 1 - Port 11 has activity	

SAL & SAH register (register 8,9)

The SAL and SAH registers together contain the complete Source Address for pause frame generation. SAL contains the least significant 24 bit of the MAC address. SAH contains the most significant 24 bit of the MAC address. The default locally managed source address for pause frame generation is FEh-FFh-FFh-FFh-FFh-FFh a. Table-7.8 and table-7.9 describes all the bits of these two registers.

Table-7.8: SAL Register

Bit	Description	Default
7:0	Bit 47:40 of the switch's MAC address.	FEh
15:8	Bit 39:32 of the switch's MAC address.	FFh
23:16	Bit 31:24 of the switch's MAC address.	

Table-7.9: SAH Register

Bit	Description	Default
7:0	Bit 23:16 of the switch's MAC address.	FFh
15:8	Bit 15:8 of the switch's MAC address.	
23:16	Bit 7:0 of the switch's MAC address.	

UTH register (register 10)

The UTH register contains the unicast buffer thresholds for each port. When the upper threshold is exceeded, the MAC may generate a max-pause-frame. When the lower threshold is crossed, the MAC may generate a mini-pause-frame. Table-7.10 describes each bit in this register.

Table-7.10: UTH Register

Bit	Description	Default*
7:0	Lower threshold of unicast utilization.	4,8,16,32
15:8	Higher threshold of unicast utilization.	8,24,64,144

*Note: The default value is related to the memory size specified by bit[6:5] of register 25.

BTH register (register 11)

The BTH register contains the broadcast queue buffer threshold for each port. When the upper threshold is exceeded, the MAC may generate a max-pause-frame. When the lower threshold is crossed, the MAC may generate a mini-pause-frame. Table-7.11 describes each bit in this register.

Table-7.11: BTH Register

Bit	Description	Default
7:0	Lower threshold of broadcast queue	16
15:8	Higher threshold of broadcast queue	48

MINL & MINH register (register 12,13)

The MINL and MINH registers together contain the 32-bit Frame Check Sequence (FCS) of the mini-pause-frame. MINL contains the least significant 16 bit of the FCS. MINH contains the most significant 16 bit of the FCS. The default FCS value assumes the default source address for the mini-pause-frame. Table-7.12 and table-7.13 describe all the bits of these two registers.

Table-7.12: MINL Register

Bit	Description	Default
7:0	Bit 31:24 of the mini-pause-frame's FCS	89
15:8	Bit 23:16 of the mini-pause-frame's FCS	O3

Table-7.13: MINH Register

Bit	Description	Default
7:0	Bit 15:18 of the mini-pause-frame's FCS	D7
15:8	Bit 7:0 of the mini-pause-frame's FCS	A9

MAXL & MAXH register (register 14,15)

The MAXL and MAXH registers together contain the 32-bit Frame Check Sequence (FCS) of the max-pause-frame. MAXL contains the least significant 16 bit of the FCS. MAXH contains the most significant 16 bit of the FCS. The default FCS value assumes the default source address for the max-pause-frame. Table-7.14 and table-7.15 describe all the bits of these two registers.

Table-7.14: MAXL Register

Bit	Description	Default
7:0	Bit 31:24 of the max-pause-frame's FCS	0D
15:8	Bit 23:16 of the max-pause-frame's FCS	68

Table-7.15: MAXH Register

Bit	Description	Default
7:0	Bit 15:8 of the max-pause-frame's FCS	D8
15:8	Bit 7:0 of the max-pause-frame's FCS	D0

SYSCFG register (register 16)

The SYSCFG register specifies certain system configurations. The system options are described in the chapter of "Function Description." Table-7.16 describes all the bit of this register.

INTMSK register (register 17)

The INTMSK register defines the valid interrupt sources allowed to assert interrupt request pin. Table-7.17 lists all the bits of this register.

Table-7.17: INTMSK Register

Bit	Description	Default
0	Enable "system initialization completion" to interrupt	1
1	Enable "internal system error" to interrupt	
2	Enable "port partition event" to interrupt	
3	Enable "Internal ARL" to interrupt	
4	ARL Interupt Enable	
5	Reserved	
6		
7		

Table-7.16: SYSCFG Register

Bit	Description	Default
0	0 - BIST enabled; 1 - BIST disabled.	0
1	0 - Spanning Tree support disabled; 1 - Spanning Tree support enabled	
2	0 - rising edge of RXCLK to latch RXD for MII 1 - falling edge of RXCLK to latch RXD for MII	
3	Reserved.	
4	Reserved.	
5	0 - wait for CPU. 1 - system ready to start **This bit is used by the CPU when bit-15 of register-25 is set as "0" (for system with control CPU). The system will wait for CPU to set this bit.*	
6	0 - PHY Management not completed 1 - PHY Management completed. **This bit is used by the CPU when bit-15 of register-25 is set as "0" (for system with a control CPU). The MAC will not start until this bit is set sy the CPU.*	
7	0 - Watchdog function enabled. 1 - Watchdog function disabled.	
8	0 - Secure VLAN checking rule enforced. 1 - Leaky VLAN checking rule enforced.	
9	0 - Rising edge of RXCLK to latch data. 1 - Falling edge of RXCLK to latch data. **For Reversed MII port only.*	
10	0 - Late Back-Pressure scheme disabled 1 - Late Back-Pressure scheme enabled **When enabled, the MAC will generate back-pressure only after reading the first bit of DA*	
11	0 - special handling of broadcast frames disabled 1 - special handling of broadcast frames enabled **When enabled, all broadcast frames from non-CPU port are forwarded to the CPU port only, and all broadcast frames from the CPU port are forwarded to all other ports.*	
12	Software Reset: "1" to start a system reset to innitialize all state machines.	
13	Hardware Reset: "1" to stop the life pulse on the watchdog pin, which in turn will trigger the external watchdog circuitry to reset the whole system.	
14	Reserved	
15	Reserved	

Data Sheet: ACD82112

INTRODUCTORY

SPEED register (register 18)

The SPEED register specifies or indicates the speed rate of each port. It is read-only, unless the bit-12 of register-25 is set (through POS to disable automatic PHY management). At read-only mode, it indicates the speed achieved through PHY management. At the write-able mode, the control CPU will be able to assign speed rate for each port. Table-7.18 describes all the bit of this register.

Table-7.18: SPEED Register

Bit	Description	Default
0	0 - Port 0 at 10Mbps 1 - Port 0 at 100Mbps	0
1	0 - Port 1 at 10Mbps 1 - Port 1 at 100Mbps	
2	0 - Port 2 at 10Mbps 1 - Port 2 at 100Mbps	
3	0 - Port 3 at 10Mbps 1 - Port 3 at 100Mbps	
4	0 - Port 4 at 10Mbps 1 - Port 4 at 100Mbps	
5	0 - Port 5 at 10Mbps 1 - Port 5 at 100Mbps	
6	0 - Port 6 at 10Mbps 1 - Port 6 at 100Mbps	
7	0 - Port 7 at 10Mbps 1 - Port 7 at 100Mbps	
8	0 - Port 8 at 10Mbps 1 - Port 8 at 100Mbps	
9	0 - Port 9 at 10Mbps 1 - Port 9 at 100Mbps	
10	0 - Port 10 at 10Mbps 1 - Port 10 at 100Mbps	
11	0 - Port 11 at 10Mbps 1 - Port 11 at 100Mbps	

LINK register (register 19)

The LINK register specifies or indicates the link status of each port. It is read-only, unless bit-12 of register-25 is set (through POS, to disable automatic PHY management). At read-only mode, it indicates the result achieved by PHY management. At write-able mode, the control CPU can assign link status for each port. Table-7.19 describes all the bit of this register.

Table-7.19: LINK Register

Bit	Description	Default
0	0 - Port 0 link not established 1 - Port 0 link established	0
1	0 - Port 1 link not established 1 - Port 1 link established	
2	0 - Port 2 link not established 1 - Port 2 link established	
3	0 - Port 3 link not established 1 - Port 3 link established	
4	0 - Port 4 link not established 1 - Port 4 link established	
5	0 - Port 5 link not established 1 - Port 5 link established	
6	0 - Port 6 link not established 1 - Port 6 link established	
7	0 - Port 7 link not established 1 - Port 7 link established	
8	0 - Port 8 link not established 1 - Port 8 link established	
9	0 - Port 9 link not established 1 - Port 9 link established	
10	0 - Port 10 link not established 1 - Port 10 link established	
11	0 - Port 11 link not established 1 - Port 11 link established	

nFWD register (register 20)

The nFWD register defines the forwarding mode of each port. Under *forwarding* mode, a port can forward all frames. Under *block-and-listen* mode, a port will not forward regular frames, except BPDU frames. If the spanning tree algorithm discovers redundant links, the control CPU will allow only one link remaining in *forwarding* mode and force all other links into *block-and-listen* mode. Setting the associated bit in this register will put the port into *block-and-listen* mode. Table-7.20

Table-7.20: nFWD Register

Bit	Description	Default
0	0 - Port 0 in forwarding state. 1 - Port 0 in block-and-listen state.	0
1	0 - Port 1 in forwarding state. 1 - Port 1 in block-and-listen state.	
2	0 - Port 2 in forwarding state. 1 - Port 2 in block-and-listen state.	
3	0 - Port 3 in forwarding state. 1 - Port 3 in block-and-listen state.	
4	0 - Port 4 in forwarding state. 1 - Port 4 in block-and-listen state.	
5	0 - Port 5 in forwarding state. 1 - Port 5 in block-and-listen state.	
6	0 - Port 6 in forwarding state. 1 - Port 6 in block-and-listen state.	
7	0 - Port 7 in forwarding state. 1 - Port 7 in block-and-listen state.	
8	0 - Port 8 in forwarding state. 1 - Port 8 in block-and-listen state.	
9	0 - Port 9 in forwarding state. 1 - Port 9 in block-and-listen state.	
10	0 - Port 10 in forwarding state. 1 - Port 10 in block-and-listen state.	
11	0 - Port 11 in forwarding state. 1 - Port 11 in block-and-listen state.	

describes all the bit of this register.

nBP register (register 21)

The nBP register defines back-pressure flow control capability for each port. Table-7.21 describes all the bit of this register.

Table-7.21: nBP Register

Bit	Description	Default
0	0 - Port 0 back-pressure scheme enabled 1 - Port 0 back pressure scheme disabled	0
1	0 - Port 1 back-pressure scheme enabled 1 - Port 1 back pressure scheme disabled	
2	0 - Port 2 back-pressure scheme enabled 1 - Port 2 back pressure scheme disabled	
3	0 - Port 3 back-pressure scheme enabled 1 - Port 3 back pressure scheme disabled	
4	0 - Port 4 back-pressure scheme enabled 1 - Port 4 back pressure scheme disabled	
5	0 - Port 5 back-pressure scheme enabled 1 - Port 5 back pressure scheme disabled	
6	0 - Port 6 back-pressure scheme enabled 1 - Port 6 back pressure scheme disabled	
7	0 - Port 7 back-pressure scheme enabled 1 - Port 7 back pressure scheme disabled	
8	0 - Port 8 back-pressure scheme enabled 1 - Port 8 back pressure scheme disabled	
9	0 - Port 9 back-pressure scheme enabled 1 - Port 9 back pressure scheme disabled	
10	0 - Port 10 back-pressure scheme enabled 1 - Port 10 back pressure scheme disabled	
11	0 - Port 11 back-pressure scheme enabled 1 - Port 11 back pressure scheme disabled	

nPORT register (register 22)

The nPORT register is used to isolate ports from the network. Setting the associated bit in this register will stop a port from receiving or transmitting any frame. Table-7.22 describes all the bits of this register.

PVID register (register 23)

The PVID registers assign VLAN IDs for each port. There are 12 PVID registers, one for each port. A PVID consists of 4 bits, each corresponding to one of the 4 VLANs. A port can belong to more than one VLAN at the same time. Table-7.23 describes all the bits of this register.

Table-7.22: nPORT Register

Bit	Description	Default
0	0 - Port 0 enabled 1 - Port 0 disabled	0
1	0 - Port 1 enabled 1 - Port 1 disabled	
2	0 - Port 2 enabled 1 - Port 2 disabled	
3	0 - Port 3 enabled 1 - Port 3 disabled	
4	0 - Port 4 enabled 1 - Port 4 disabled	
5	0 - Port 5 enabled 1 - Port 5 disabled	
6	0 - Port 6 enabled 1 - Port 6 disabled	
7	0 - Port 7 enabled 1 - Port 7 disabled	
8	0 - Port 8 enabled 1 - Port 8 disabled	
9	0 - Port 9 enabled 1 - Port 9 disabled	
10	0 - Port 10 enabled 1 - Port 10 disabled	
11	0 - Port 11 enabled 1 - Port 11 disabled	

Table-7.23: PVID Register

Bit	Description	Default
0	0 - port not in VLAN-I. 1 - port in VLAN-I.	1
1	0 - port not in VLAN-II. 1 - port in VLAN-II.	0
2	0 - port not in VLAN-III. 1 - port in VLAN-III.	
3	0 - port not in VLAN-IV. 1 - port in VLAN-IV.	

VPID register (register 24)

The VPID registers specify the dumping port for each VLAN. There are 4 VPID 5-bit registers, one for each VLAN. A valid VPID are “0” through “11” (other values are reserved and should not used). Table-7.24 describes all the bits of this register.

Table-7.24: VPID Registers *(4 registers)*

Bit	Description	Default
4:0	Dumping port ID for VLAN-1	"00000"
4:0	Dumping port ID for VLAN-2	"11111" dumping port not defined
4:0	Dumping port ID for VLAN-3	
4:0	Dumping port ID for VLAN-4	

Table-7.25: POSCFG Register

Bit	Description	Default
3:0	8 timing adjustment levels for SRAM Read data latching: 0000 - no delay 0001 - level 1 delay 0011 - level 2 delay 0101 - level 3 delay 0111 - level 4 delay 1001 - level 5 delay 1011 - level 6 delay 1101 - level 7 delay 1111 - level 8 delay	0000
4	0 - Absolute address mode: 1 row of 512K words, nCS2=ADDR17, nCS3=ADDR18 1 - Chip-Select address mode: 4 rows of 128K words, nCS[3:0] to select 4 rows of memory	0
6:5	SRAM size selection: 00 - 64K words 01 - 128K words 10 - 256k words 11 - 512K words	01
7	0 - Long Event defined as frame longer than 1518 byte. 1 - Long Event defined as frame longer than 1530 byte.	1
8	0 - Frames with unknown DA forwarded to the dumping port. 1 - Frames with unknown DA forwarded to all ports.	1
9	0 - Internal ARL selected (2K MAC address entry). 1 - External ARL selected (11K MAC address entry).	0
10	0 - PHY IDs start from 1, range from 1 to 12. 1 - PHY IDs start from 4, range from 4 to 15.	1
11	0 - Re-transmit after excessive collision. 1 - Drop after excessive collision.	0
12	0 - Automatic PHY Management enabled 1 - Automatic PHY Management disabled: the control CPU need to update the SPEED, LINK, DPLX and nPAUSE registers	0
13	Reserved	0
14	0 - Sysem errors will trigger software reset 1 - Sysem errors will trigger hardware reset	0
15	0 - System start itself without a control CPU 1 - System start after system-ready bit in register-16 is set by the control CPU	0
17:16	2-bit device ID for UART communication. The device responses only to UART commands with matching ID	00
18	0 - Rising edge of ARLCLK to latch ARLDI. 1 - Falling edge of ARLCLK to latch ARLDI.	1

<u>POSCFG register (register 25)</u>

The POSCFG register specifies a certain configuration setting for the switch system. The default values of this register can be changed through pull-up/pull-down of specific pins, as described in the “Configuration Interface” section of the “Interface Description” chapter. Table-7.25 describes all the bit of this register.

PAUSE register (register 26)

The PAUSE register defines the pause-frame based flow control capability of each port. Table-7.26 describes all the bits of this register.

Table-7.26: PAUSE Register

Bit	Description	Default
0	0 - Port 0 Pause-Frame disabled 1 - Port 0 Pause-Frame enabled	1
1	0 - Port 1 Pause-Frame disabled 1 - Port 1 Pause-Frame enabled	
2	0 - Port 2 Pause-Frame disabled 1 - Port 2 Pause-Frame enabled	
3	0 - Port 3 Pause-Frame disabled 1 - Port 3 Pause-Frame enabled	
4	0 - Port 4 Pause-Frame disabled 1 - Port 4 Pause-Frame enabled	
5	0 - Port 5 Pause-Frame disabled 1 - Port 5 Pause-Frame enabled	
6	0 - Port 6 Pause-Frame disabled 1 - Port 6 Pause-Frame enabled	
7	0 - Port 7 Pause-Frame disabled 1 - Port 7 Pause-Frame enabled	
8	0 - Port 8 Pause-Frame disabled 1 - Port 8 Pause-Frame enabled	
9	0 - Port 9 Pause-Frame disabled 1 - Port 9 Pause-Frame enabled	
10	0 - Port 10 Pause-Frame disabled 1 - Port 10 Pause-Frame enabled	
11	0 - Port 11 Pause-Frame disabled 1 - Port 11 Pause-Frame enabled	

DPLX register (register 27)

The DPLX register specifies or indicates the half/full-duplex mode of each port. It is read-only, unless bit-12 of register-25 is set (through POS, to disable automatic PHY management). At read-only mode, it indicates the result achieved by the PHY management. At write-able mode, the control CPU can assign a half-duplex or full-duplex mode for each port. Table-7.27 describes all the bits of this register.

Table-7.27: DPLX Register

Bit	Description	Default
0	0 - Port 0 under half duplex mode 1 - Port 0 under full duplex mode	0
1	0 - Port 1 under half duplex mode 1 - Port 1 under full duplex mode	
2	0 - Port 2 under half duplex mode 1 - Port 2 under full duplex mode	
3	0 - Port 3 under half duplex mode 1 - Port 3 under full duplex mode	
4	0 - Port 4 under half duplex mode 1 - Port 4 under full duplex mode	
5	0 - Port 5 under half duplex mode 1 - Port 5 under full duplex mode	
6	0 - Port 6 under half duplex mode 1 - Port 6 under full duplex mode	
7	0 - Port 7 under half duplex mode 1 - Port 7 under full duplex mode	
8	0 - Port 8 under half duplex mode 1 - Port 8 under full duplex mode	
9	0 - Port 9 under half duplex mode 1 - Port 9 under full duplex mode	
10	0 - Port 10 under half duplex mode 1 - Port 10 under full duplex mode	
11	0 - Port 11 under half duplex mode 1 - Port 11 under full duplex mode	

RVSMII register (register 28)

The RVSMII register defines the *reversed MII* mode for each port. Table-7.28 describes all the bits of this register.

Table-7.28: RVSMII register

Bit	Description	Default
0	0 - Port 0 under normal MII mode 1 - Port 0 under reversed MII mode	0
1	0 - Port 1 under normal MII mode 1 - Port 1 under reversed MII mode	
2	0 - Port 2 under normal MII mode 1 - Port 2 under reversed MII mode	
3	0 - Port 3 under normal MII mode 1 - Port 3 under reversed MII mode	
4	0 - Port 11 under normal MII mode 1 - Port 11 under reversed MII mode	

nPM register (register 29)

The nPM register indicates the automatic PHY management capability of each port. If a bit is set in this register, the corresponding SPEED, LINK, DPLX, and PAUSE status registers of a port will remain unchanged. Table-7.29 describes all the bits of this register.

Table-7.29: nPM Register

Bit	Description	Default
0	0 - Port 0's status update enabled 1 - Port 0's status update disabled	0
1	0 - Port 1's status update enabled 1 - Port 1's status update disabled	
2	0 - Port 2's status update enabled 1 - Port 2's status update disabled	
3	0 - Port 3's status update enabled 1 - Port 3's status update disabled	
4	0 - Port 4's status update enabled 1 - Port 4's status update disabled	
5	0 - Port 5's status update enabled 1 - Port 5's status update disabled	
6	0 - Port 6's status update enabled 1 - Port 6's status update disabled	
7	0 - Port 7's status update enabled 1 - Port 7's status update disabled	
8	0 - Port 8's status update enabled 1 - Port 8's status update disabled	
9	0 - Port 9's status update enabled 1 - Port 9's status update disabled	
10	0 - Port 10's status update enabled 1 - Port 10's status update disabled	
11	0 - Port 11's status update enabled 1 - Port 11's status update disabled	

ERRMSK register (register 30)

The ERRMSK register defines certain errors as *system errors*. It is reserved for factory use only. Table-7.30 lists all the error masks specified by this register.

Table-7.30: ERRMSK register

Bit	Description	Default
0	Reserved	1
1		
2		
3		
4		
5		
6		
7		

CLKADJ register (register 31)

The CLKADJ register defines the delay time of the ARLCLK relative to the transition edge of the data signals. The ARLCLK provides reference timing for supporting chips, such as the ACD80800 and the ACD80900, which need to snoop the data bus for certain activities. Table-7.31 describes all the bits of this register.

Table-7.31: CLKADJ Register

Bit	Description	Default
0	0 - ARLCLK not inverted 1 - ARLCLK inverted	0
3:1	ARLCLK delay levels: 000 - level 0 delay 001 - level 1 delay 010 - level 2 delay 011 - level 3 delay 100 - level 4 delay 101 - level 5 delay 110 - level 6 delay 111 - level 7 delay	000

PHYREG register (register 32-44)

The PHYREG refers to the registers residing on the PHY devices. The ACD82112 merely provides an access path for the control CPU to access the registers on the PHYs. For detailed information about these registers, please refer to the PHY data sheet.

Register-32 through Register-44 are assigned to PHY-0 through PHY-11 respectively. The contents of these registers are the register IDs inside the corresponding PHYs. For example, a "4" in Register-44 will point to the Control Register-4 inside PHY-11.

Data Sheet: ACD82112

INTRODUCTORY

8. PIN DESCRIPTIONS

Figure-8.1: Pin Diagram/Bottom View

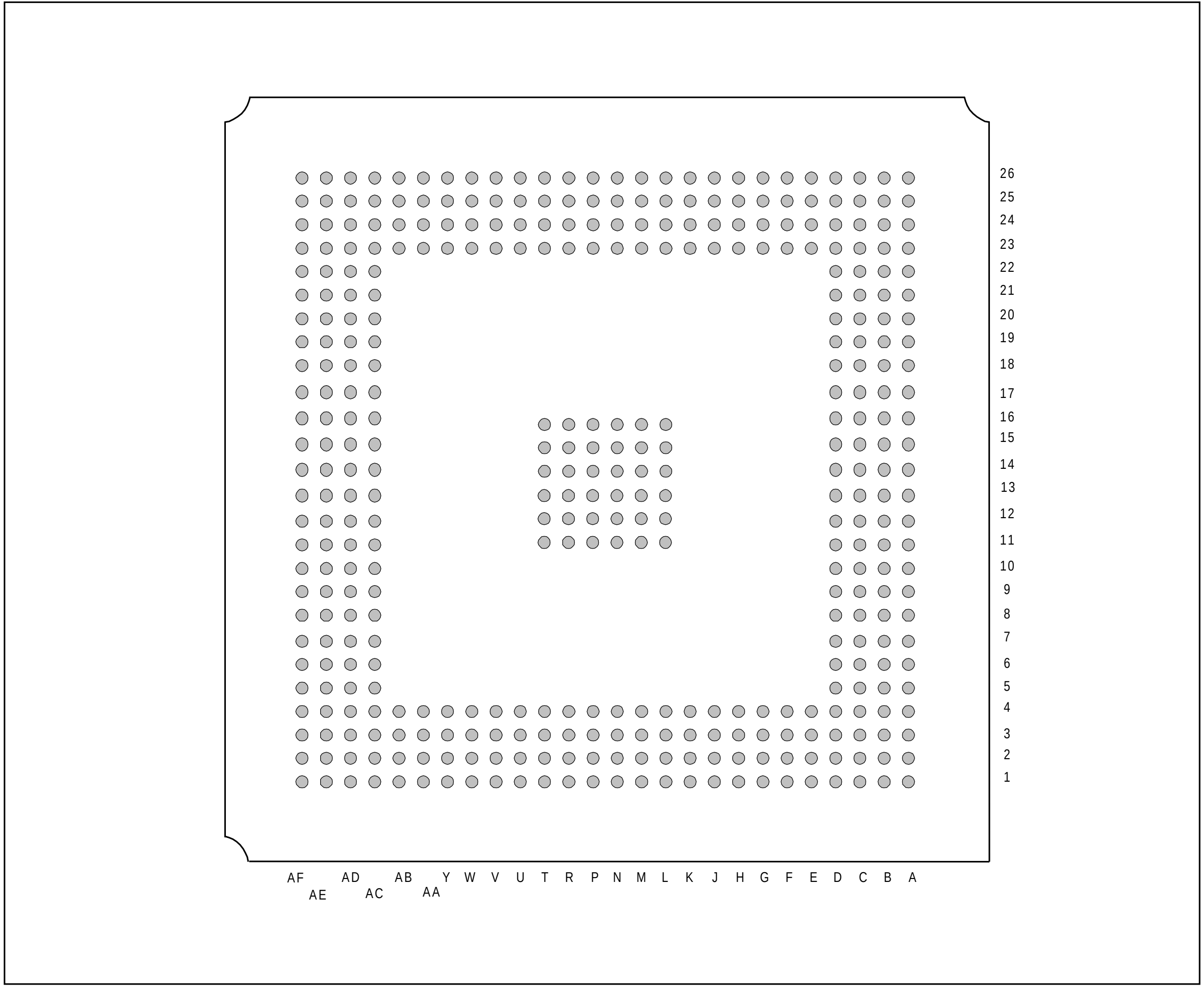


Table-8.1: Thermal Ground Pins

Pin	Signal	I/O Type
L[11:16]	VSS	Ground/Thermal
M[11:16]		
N[11:16]		
P[11:16]		
R[11:16]		
T[11:16]		

Table-8.2: Pin List By Location

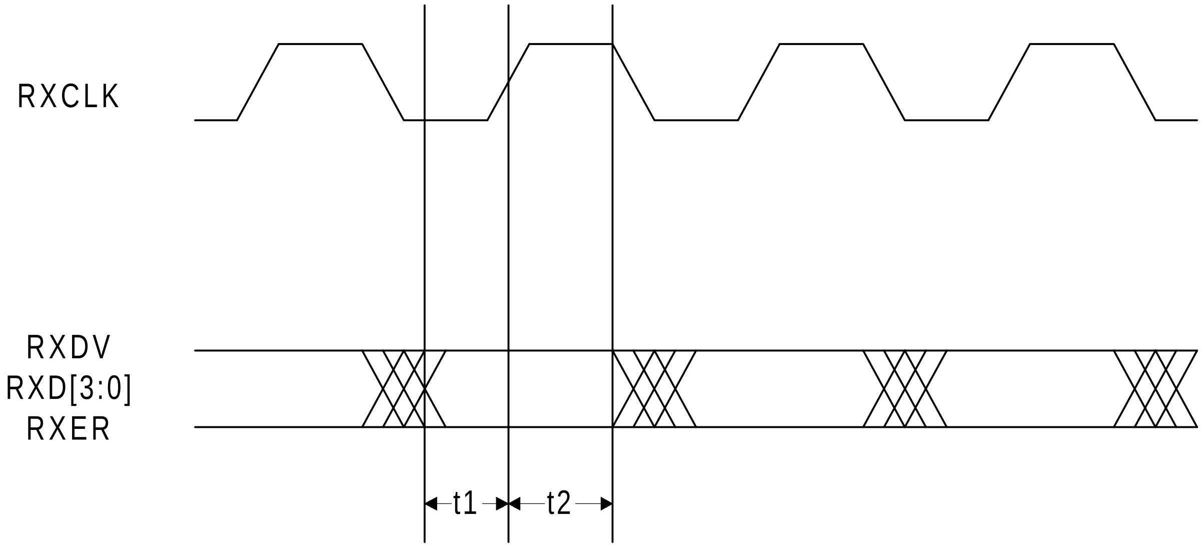
Pin	Signal Name	I/O	Type	Pin	Signal Name	I/O	Type	Pin	Signal Name	I/O	Type	Pin	Signal Name	I/O	Type
A01	DATA51	3.3V	I/O	D11	ARLSYNC	3.3V	O	P01	DATA25	3.3V	I/O	AC17	VSS		
A02	DATA50	3.3V	I/O	D12	VSS			P02	DATA24	3.3V	I/O	AC18	P3TXENR	3.3V	O
A03	STAT1	3.3V	O	D13	VSS			P03	VDD			AC19	P3RXD0R	3.3V	I
A04	STAT3	3.3V	O	D14	P11RXD3R	3.3V	I	P04	VSS			AC20	VSS		
A05	ADDR9	3.3V	O	D15	VSS			P23	P7TXD0	3.3V	I/O	AC21	P4TXD1	3.3V	O
A06	ADDR8	3.3V	O	D16	P11RXD1R	3.3V	I	P24	P7TXD1	3.3V	I/O	AC22	P4RXCLK	3.3V	I
A07	ADDR7	3.3V	O	D17	VSS			P25	P7TXD2	3.3V	I/O	AC23	VSS		
A08	ADDR6	3.3V	O	D18	P11TXENR	3.3V	O	P26	P7TXD3	3.3V	I/O	AC24	P5COL	3.3V	I
A09	ADDR5	3.3V	O	D19	P11COLR	3.3V	I/O	R01	DATA23	3.3V	I/O	AC25	P5TXD2	3.3V	I/O
A10	nWE	3.3V	O	D20	VSS			R02	DATA22	3.3V	I/O	AC26	P5TXD1	3.3V	I/O
A11	nCS0	3.3V	O	D21	P10RXER	3.3V	I	R03	VDD			AD01	DATA5	3.3V	I/O
A12	ADDR4	3.3V	O	D22	P10TXD2	3.3V	O	R04	VSS			AD02	DATA4	3.3V	I/O
A13	ADDR3	3.3V	O	D23	VSS			R23	P7COL	3.3V	I	AD03	VDD		
A14	ADDR2	3.3V	O	D24	P9RXD0	3.3V	I	R24	P7CRS	3.3V	I	AD04	P0CRSR	3.3V	I/O
A15	ADDR1	3.3V	O	D25	P9TXD1	3.3V	O	R25	P6RXD3	3.3V	I	AD05	P0TXD1R	3.3V	I/O
A16	ADDR0	3.3V	O	D26	P9TXD2	3.3V	O	R26	P6RXD2	3.3V	I	AD06	P0TXCLKR	3.3V	I/O
A17	P11RXERR	3.3V	I	E01	DATA43	3.3V	I/O	T01	DATA21	3.3V	I/O	AD07	VDD		
A18	P11TXD1R	3.3V	O	E02	DATA42	3.3V	I/O	T02	DATA20	3.3V	I/O	AD08	P1CRSR	3.3V	I/O
A19	P11CRSR	3.3V	I/O	E03	ARLDIR0	3.3V	O	T03	LEDCLK	3.3V	I/O	AD09	P1TXD0R	3.3V	I/O
A20	P10RXD3	3.3V	I	E04	ARLDIR1	3.3V	O	T04	CPUIRQ	3.3V	O	AD10	VDD		
A21	P10RXD1	3.3V	I	E23	P9RXDV	3.3V	I	T23	P6RXD1	3.3V	I	AD11	P1RXDVR	3.3V	I
A22	P10RXDV	3.3V	I	E24	P9TXEN	3.3V	O	T24	P6RXD0	3.3V	I	AD12	P1RXD3R	3.3V	I
A23	P10TXD0	3.3V	O	E25	P9COL	3.3V	I	T25	P6RXDV	3.3V	I	AD13	P2TXD2R	3.3V	O
A24	P10COL	3.3V	I	E26	P9CRS	3.3V	I	T26	P6RXCLK	3.3V	I	AD14	VDD		
A25	P9RXD3	3.3V	I	F01	DATA41	3.3V	I/O	U01	DATA19	3.3V	I/O	AD15	P2RXCLKR	3.3V	I/O
A26	P9RXD1	3.3V	I	F02	DATA40	3.3V	I/O	U02	DATA18	3.3V	I/O	AD16	P2RXD2R	3.3V	I
B01	DATA49	3.3V	I/O	F03	VDD			U03	VDD			AD17	VDD		
B02	DATA48	3.3V	I/O	F04	VSS			U04	VSS			AD18	P3TXD0R	3.3V	O
B03	STAT0	3.3V	O	F23	P9TXD0	3.3V	O	U23	VSS			AD19	P3RXDVR	3.3V	I
B04	STAT2	3.3V	O	F24	P9TXD3	3.3V	O	U24	VDD			AD20	VDD		
B05	ADDR10	3.3V	O	F25	P8RXD3	3.3V	I	U25	P6RXER	3.3V	I	AD21	P4COL	3.3V	I
B06	ADDR11	3.3V	O	F26	P8RXD2	3.3V	I	U26	P6TXCLK	3.3V	I	AD22	P4TXD0	3.3V	O
B07	ADDR12	3.3V	O	G01	DATA39	3.3V	I/O	V01	DATA17	3.3V	I/O	AD23	P4RXDV	3.3V	I
B08	ADDR13	3.3V	O	G02	DATA38	3.3V	I/O	V02	DATA16	3.3V	I/O	AD24	VDD		
B09	ADDR14	3.3V	O	G03	VDD			V03	VDD			AD25	P4RXD3	3.3V	I
B10	nOE	3.3V	I/O	G04	VSS			V04	VSS			AD26	P5CRS	3.3V	I
B11	ADDR15	3.3V	O	G23	VSS			V23	P6TXD0	3.3V	I/O	AE01	DATA3	3.3V	I/O
B12	ADDR16	3.3V	O	G24	VDD			V24	P6TXEN	3.3V	O	AE02	DATA2	3.3V	I/O
B13	nCS2	3.3V	O	G25	P8RXDV	3.3V	I	V25	P6TXD1	3.3V	I/O	AE03	VSS		
B14	nCS3	3.3V	O	G26	P8RXCLK	3.3V	I	V26	P6TXD2	3.3V	I/O	AE04	P0TXD3R	3.3V	I/O
B15	nCS1	3.3V	O	H01	DATA37	3.3V	I/O	W01	DATA15	3.3V	I/O	AE05	P0TXD0R	3.3V	I/O
B16	P11RXDVR	3.3V	I	H02	DATA36	3.3V	I/O	W02	DATA14	3.3V	I/O	AE06	P0RXCLKR	3.3V	I/O
B17	P11RXCLKR	3.3V	I/O	H03	LED2	3.3V	I/O	W03	CPUDO	3.3V	I/O	AE07	P0RXD0R	3.3V	I
B18	P11TXD0R	3.3V	O	H04	LED3	3.3V	I/O	W04	CPUDI	3.3V	I	AE08	P1COLR	3.3V	I/O
B19	P11TXD2R	3.3V	O	H23	P8RXD1	3.3V	I	W23	P6CRS	3.3V	I	AE09	P1TXD2R	3.3V	I/O
B20	P10RXD2	3.3V	I	H24	P8RXD0	3.3V	I	W24	P6COL	3.3V	I	AE10	P1TXCLKR	3.3V	I/O
B21	P10RXD0	3.3V	I	H25	P8RXER	3.3V	I	W25	P6TXD3	3.3V	I/O	AE11	P1RXCLKR	3.3V	I/O
B22	P10TXCLK	3.3V	I	H26	P8TXD0	3.3V	O	W26	P5RXD3	3.3V	I	AE12	P1RXD2R	3.3V	I
B23	P10TXD1	3.3V	O	J01	DATA35	3.3V	I/O	Y01	DATA13	3.3V	I/O	AE13	P2TXD3R	3.3V	O
B24	P10CRS	3.3V	I	J02	DATA34	3.3V	I/O	Y02	DATA12	3.3V	I/O	AE14	P2TXENR	3.3V	O
B25	P9RXD2	3.3V	I	J03	VDD			Y03	VDD			AE15	P2RXERR	3.3V	I
B26	P9RXER	3.3V	I	J04	VSS			Y04	VSS			AE16	P2RXD1R	3.3V	I
C01	DATA47	3.3V	I/O	J23	P8TXCLK	3.3V	I	Y23	VSS			AE17	P3COLR	3.3V	I/O
C02	DATA46	3.3V	I/O	J24	P8TXEN	3.3V	O	Y24	VDD			AE18	P3TXD2R	3.3V	O
C03	VDD			J25	P8TXD1	3.3V	O	Y25	P5RXD1	3.3V	I	AE19	P3TXCLKR	3.3V	I/O
C04	ARLDIV	3.3V	I	J26	P8TXD2	3.3V	O	Y26	P5RXD2	3.3V	I	AE20	P3RXCLKR	3.3V	I/O
C05	VDD			K01	DATA33	3.3V	I/O	AA01	DATA11	3.3V	I/O	AE21	P3RXD2R	3.3V	I
C06	ARLDI2	3.3V	I	K02	DATA32	3.3V	I/O	AA02	DATA10	3.3V	I/O	AE22	P4CRS	3.3V	I
C07	VDD			K03	VDD			AA03	MDIO	3.3V	I/O	AE23	P4TXD2	3.3V	O
C08	ARLDI0	3.3V	I	K04	VSS			AA04	WCHDOG	3.3V	O	AE24	P4RXER	3.3V	I
C09	VDD			K23	VSS			AA23	P5TXCLK	3.3V	I	AE25	P4RXD1	3.3V	I
C10	VDD			K24	VDD			AA24	P5RXER	3.3V	I	AE26	P4RXD2	3.3V	I
C11	ARLCLK	3.3V	O	K25	P8TXD3	3.3V	O	AA25	P5RXDV	3.3V	I	AF01	DATA1	3.3V	I/O
C12	VDD			K26	P8COL	3.3V	I	AA26	P5RXD0	3.3V	I	AF02	DATA0	3.3V	I/O
C13	VDD			L01	DATA31	3.3V	I/O	AB01	DATA9	3.3V	I/O	AF03	CLK50	3.3V	I
C14	P11RXD2R	3.3V	I	L02	DATA30	3.3V	I/O	AB02	MDC	3.3V	O	AF04	P0TXD2R	3.3V	I/O
C15	VDD			L03	LED1	3.3V	I/O	AB03	DATA8	3.3V	I/O	AF05	P0RXERR	3.3V	I
C16	P11RXD0R	3.3V	I	L04	LED0	3.3V	I/O	AB04	VSS	3.3V	I	AF06	P0RXDVR	3.3V	I
C17	VDD			L23	P8CRS	3.3V	I	AB23	P5TXD3	3.3V	I/O	AF07	P0RXD1R	3.3V	I
C18	P11TXCLKR	3.3V	I/O	L24	P7RXD3	3.3V	I	AB24	P5TXD0	3.3V	I/O	AF08	P0RXD2R	3.3V	I
C19	P11TXD3R	3.3V	O	L25	P7RXD2	3.3V	I	AB25	P5TXEN	3.3V	O	AF09	P1TXD3R	3.3V	I/O
C20	VDD			L26	P7RXD1	3.3V	I	AB26	P5RXCLK	3.3V	I	AF10	P1TXENR	3.3V	O
C21	P10RXCLK	3.3V	I	M01	DATA29	3.3V	I/O	AC01	DATA7	3.3V	I/O	AF11	P1RXERR	3.3V	I
C22	P10TXEN	3.3V	O	M02	DATA28	3.3V	I/O	AC02	DATA6	3.3V	I/O	AF12	P1RXD1R	3.3V	I
C23	P10TXD3	3.3V	O	M03	VDD			AC03	nRESET	3.3V	I	AF13	P2COLR	3.3V	I/O
C24	VDD			M04	VSS			AC04	VSS			AF14	P2TXD0R	3.3V	O
C25	P9RXCLK	3.3V	I	M23	P7RXD0	3.3V	I	AC05	P0COLR	3.3V	I/O	AF15	P2TXCLKR	3.3V	I/O
C26	P9TXCLK	3.3V	I	M24	P7RXDV	3.3V	I	AC06	P0TXENR	3.3V	O	AF16	P2RXD0R	3.3V	I
D01	DATA45	3.3V	I/O	M25	P7RXCLK	3.3V	I	AC07	VSS			AF17	P3CRSR	3.3V	I/O
D02	DATA44	3.3V	I/O	M26	P7RXER	3.3V	I	AC08	P0RXD3R	3.3V	I	AF18	P3TXD3R	3.3V	O
D03	VSS	3.3V	I	N01	DATA27	3.3V	I/O	AC09	P1TXD1R	3.3V	I/O	AF19	P3TXD1R	3.3V	O
D04	VSS			N02	DATA26	3.3V	I/O	AC10	VSS			AF20	P3RXERR	3.3V	I
D05	ARLDI3	3.3V	I	N03	LEDVLD1	3.3V	I/O	AC11	P1RXD0R	3.3V	I	AF21	P3RXD1R	3.3V	I
D06	VSS			N04	LEDVLD0	3.3V	I/O	AC12	P2CRSR	3.3V	I/O	AF22	P3RXD3R	3.3V	I
D07	VSS			N23	VSS			AC13	P2TXD1R	3.3V	O	AF23	P4TXD3	3.3V	O
D08	ARLDI1	3.3V	I	N24	VDD			AC14	VSS			AF24	P4TXEN	3.3V	O
D09	VSS			N25	P7TXCLK	3.3V	I	AC15	P2RXDVR	3.3V	I	AF25	P4TXCLK	3.3V	I
D10	VSS			N26	P7TXEN	3.3V	O	AC16	P2RXD3R	3.3V	I	AF26	P4RXD0	3.3V	I

Table-8.3: Pin List By Name

Signal Name	I/O	Type	Pin	Signal Name	I/O	Type	Pin	Signal Name	I/O	Type	Pin	Signal Name	I/O	Type	Pin
ADDR0	3.3V	O	A16	LEDVLD1	3.3V	I/O	N03	P5RXD0	3.3V	I	AA26	P11CRSR	3.3V	I/O	A19
ADDR1	3.3V	O	A15	MDC	3.3V	O	AB02	P5RXD1	3.3V	I	Y25	P11RXCLKR	3.3V	I/O	B17
ADDR2	3.3V	O	A14	MDIO	3.3V	I/O	AA03	P5RXD2	3.3V	I	Y26	P11RXD0R	3.3V	I	C16
ADDR3	3.3V	O	A13	nCS0	3.3V	O	A11	P5RXD3	3.3V	I	W26	P11RXD1R	3.3V	I	D16
ADDR4	3.3V	O	A12	nCS1	3.3V	O	B15	P5RXDV	3.3V	I	AA25	P11RXD2R	3.3V	I	C14
ADDR5	3.3V	O	A09	nCS2	3.3V	O	B13	P5RXER	3.3V	I	AA24	P11RXD3R	3.3V	I	D14
ADDR6	3.3V	O	A08	nCS3	3.3V	O	B14	P5TXCLK	3.3V	I	AA23	P11RXDVR	3.3V	I	B16
ADDR7	3.3V	O	A07	nOE	3.3V	I/O	B10	P5TXD0	3.3V	I/O	AB24	P11RXERR	3.3V	I	A17
ADDR8	3.3V	O	A06	nRESET	3.3V	I	AC03	P5TXD1	3.3V	I/O	AC26	P11TXCLKR	3.3V	I/O	C18
ADDR9	3.3V	O	A05	nWE	3.3V	O	A10	P5TXD2	3.3V	I/O	AC25	P11TXD0R	3.3V	O	B18
ADDR10	3.3V	O	B05	P0COLR	3.3V	I/O	AC05	P5TXD3	3.3V	I/O	AB23	P11TXD1R	3.3V	O	A18
ADDR11	3.3V	O	B06	P0CRSR	3.3V	I/O	AD04	P5TXEN	3.3V	O	AB25	P11TXD2R	3.3V	O	B19
ADDR12	3.3V	O	B07	P0RXCLKR	3.3V	I/O	AE06	P6COL	3.3V	I	W24	P11TXD3R	3.3V	O	C19
ADDR13	3.3V	O	B08	P0RXD0R	3.3V	I	AE07	P6CRS	3.3V	I	W23	P11TXENR	3.3V	O	D18
ADDR14	3.3V	O	B09	P0RXD1R	3.3V	I	AF07	P6RXCLK	3.3V	I	T26	STAT0	3.3V	O	B03
ADDR15	3.3V	O	B11	P0RXD2R	3.3V	I	AF08	P6RXD0	3.3V	I	T24	STAT1	3.3V	O	A03
ADDR16	3.3V	O	B12	P0RXD3R	3.3V	I	AC08	P6RXD1	3.3V	I	T23	STAT2	3.3V	O	B04
ARLCLK	3.3V	O	C11	P0RXDVR	3.3V	I	AF06	P6RXD2	3.3V	I	R26	STAT3	3.3V	O	A04
ARLDI0	3.3V	I	C08	P0RXERR	3.3V	I	AF05	P6RXD3	3.3V	I	R25	VDD	Power	3.3V	AD03
ARLDI1	3.3V	I	D08	P0TXCLKR	3.3V	I/O	AD06	P6RXDV	3.3V	I	T25	VDD	Power	3.3V	AD07
ARLDI2	3.3V	I	C06	P0TXD0R	3.3V	I/O	AE05	P6RXER	3.3V	I	U25	VDD	Power	3.3V	AD14
ARLDI3	3.3V	I	D05	P0TXD1R	3.3V	I/O	AD05	P6TXCLK	3.3V	I	U26	VDD	Power	3.3V	AD20
ARLDIR0	3.3V	O	E03	P0TXD2R	3.3V	I/O	AF04	P6TXD0	3.3V	I/O	V23	VDD	Power	3.3V	AD24
ARLDIR1	3.3V	O	E04	P0TXD3R	3.3V	I/O	AE04	P6TXD1	3.3V	I/O	V25	VDD	Power	3.3V	C03
ARLDIV	3.3V	I	C04	P0TXENR	3.3V	O	AC06	P6TXD2	3.3V	I/O	V26	VDD	Power	3.3V	C05
ARLSYNC	3.3V	O	D11	P1COLR	3.3V	I/O	AE08	P6TXD3	3.3V	I/O	W25	VDD	Power	3.3V	C09
CLK50	3.3V	I	AF03	P1CRSR	3.3V	I/O	AD08	P6TXEN	3.3V	O	V24	VDD	Power	3.3V	C12
CPUDI	3.3V	I	W04	P1RXCLKR	3.3V	I/O	AE11	P7COL	3.3V	I	R23	VDD	Power	3.3V	C13
CPUDO	3.3V	I/O	W03	P1RXD0R	3.3V	I	AC11	P7CRS	3.3V	I	R24	VDD	Power	3.3V	C15
CPUIRQ	3.3V	O	T04	P1RXD1R	3.3V	I	AF12	P7RXCLK	3.3V	I	M25	VDD	Power	3.3V	C20
DATA0	3.3V	I/O	AF02	P1RXD2R	3.3V	I	AE12	P7RXD0	3.3V	I	M23	VDD	Power	3.3V	C24
DATA1	3.3V	I/O	AF01	P1RXD3R	3.3V	I	AD12	P7RXD1	3.3V	I	L26	VDD	Power	3.3V	G03
DATA2	3.3V	I/O	AE02	P1RXDVR	3.3V	I	AD11	P7RXD2	3.3V	I	L25	VDD	Power	3.3V	J03
DATA3	3.3V	I/O	AE01	P1RXERR	3.3V	I	AF11	P7RXD3	3.3V	I	L24	VDD	Power	3.3V	N24
DATA4	3.3V	I/O	AD02	P1TXCLKR	3.3V	I/O	AE10	P7RXDV	3.3V	I	M24	VDD	Power	3.3V	P03
DATA5	3.3V	I/O	AD01	P1TXD0R	3.3V	I/O	AD09	P7RXER	3.3V	I	M26	VDD	Power	3.3V	R03
DATA6	3.3V	I/O	AC02	P1TXD1R	3.3V	I/O	AC09	P7TXCLK	3.3V	I	N25	VDD	Power	3.3V	Y24
DATA7	3.3V	I/O	AC01	P1TXD2R	3.3V	I/O	AE09	P7TXD0	3.3V	I/O	P23	VDD	Power	3.3V	AD17
DATA8	3.3V	I/O	AB03	P1TXD3R	3.3V	I/O	AF09	P7TXD1	3.3V	I/O	P24	VDD	Power	3.3V	C10
DATA9	3.3V	I/O	AB01	P1TXENR	3.3V	O	AF10	P7TXD2	3.3V	I/O	P25	VDD	Power	3.3V	K03
DATA10	3.3V	I/O	AA02	P2COLR	3.3V	I/O	AF13	P7TXD3	3.3V	I/O	P26	VDD	Power	3.3V	K24
DATA11	3.3V	I/O	AA01	P2CRSR	3.3V	I/O	AC12	P7TXEN	3.3V	O	N26	VDD	Power	3.3V	U03
DATA12	3.3V	I/O	Y02	P2RXCLKR	3.3V	I/O	AD15	P8COL	3.3V	I	K26	VDD	Power	3.3V	U24
DATA13	3.3V	I/O	Y01	P2RXD0R	3.3V	I	AF16	P8CRS	3.3V	I	L23	VDD	Power	3.3V	Y03
DATA14	3.3V	I/O	W02	P2RXD1R	3.3V	I	AE16	P8RXCLK	3.3V	I	G26	VDD	Power	3.3V	AD10
DATA15	3.3V	I/O	W01	P2RXD2R	3.3V	I	AD16	P8RXD0	3.3V	I	H24	VDD	Power	3.3V	C07
DATA16	3.3V	I/O	V02	P2RXD3R	3.3V	I	AC16	P8RXD1	3.3V	I	H23	VDD	Power	3.3V	C17
DATA17	3.3V	I/O	V01	P2RXDVR	3.3V	I	AC15	P8RXD2	3.3V	I	F26	VDD	Power	3.3V	F03
DATA18	3.3V	I/O	U02	P2RXERR	3.3V	I	AE15	P8RXD3	3.3V	I	F25	VDD	Power	3.3V	G24
DATA19	3.3V	I/O	U01	P2TXCLKR	3.3V	I/O	AF15	P8RXDV	3.3V	I	G25	VDD	Power	3.3V	M03
DATA20	3.3V	I/O	T02	P2TXD0R	3.3V	O	AF14	P8RXER	3.3V	I	H25	VDD	Power	3.3V	V03
DATA21	3.3V	I/O	T01	P2TXD1R	3.3V	O	AC13	P8TXCLK	3.3V	I	J23	VSS	3.3V	I	AB04
DATA22	3.3V	I/O	R02	P2TXD2R	3.3V	O	AD13	P8TXD0	3.3V	O	H26	VSS	3.3V	I	D03
DATA23	3.3V	I/O	R01	P2TXD3R	3.3V	O	AE13	P8TXD1	3.3V	O	J25	VSS	Ground		D09
DATA24	3.3V	I/O	P02	P2TXENR	3.3V	O	AE14	P8TXD2	3.3V	O	J26	VSS	Ground		D12
DATA25	3.3V	I/O	P01	P3COLR	3.3V	I/O	AE17	P8TXD3	3.3V	O	K25	VSS	Ground		D15
DATA26	3.3V	I/O	N02	P3CRSR	3.3V	I/O	AF17	P8TXEN	3.3V	O	J24	VSS	Ground		D20
DATA27	3.3V	I/O	N01	P3RXCLKR	3.3V	I/O	AE20	P9COL	3.3V	I	E25	VSS	Ground		G04
DATA28	3.3V	I/O	M02	P3RXD0R	3.3V	I	AC19	P9CRS	3.3V	I	E26	VSS	Ground		AC04
DATA29	3.3V	I/O	M01	P3RXD1R	3.3V	I	AF21	P9RXCLK	3.3V	I	C25	VSS	Ground		AC14
DATA30	3.3V	I/O	L02	P3RXD2R	3.3V	I	AE21	P9RXD0	3.3V	I	D24	VSS	Ground		AC23
DATA31	3.3V	I/O	L01	P3RXD3R	3.3V	I	AF22	P9RXD1	3.3V	I	A26	VSS	Ground		D04
DATA32	3.3V	I/O	K02	P3RXDVR	3.3V	I	AD19	P9RXD2	3.3V	I	B25	VSS	Ground		D10
DATA33	3.3V	I/O	K01	P3RXERR	3.3V	I	AF20	P9RXD3	3.3V	I	A25	VSS	Ground		D13
DATA34	3.3V	I/O	J02	P3TXCLKR	3.3V	I/O	AE19	P9RXDV	3.3V	I	E23	VSS	Ground		K04
DATA35	3.3V	I/O	J01	P3TXD0R	3.3V	O	AD18	P9RXER	3.3V	I	B26	VSS	Ground		K23
DATA36	3.3V	I/O	H02	P3TXD1R	3.3V	O	AF19	P9TXCLK	3.3V	I	C26	VSS	Ground		P04
DATA37	3.3V	I/O	H01	P3TXD2R	3.3V	O	AE18	P9TXD0	3.3V	O	F23	VSS	Ground		R04
DATA38	3.3V	I/O	G02	P3TXD3R	3.3V	O	AF18	P9TXD1	3.3V	O	D25	VSS	Ground		Y04
DATA39	3.3V	I/O	G01	P3TXENR	3.3V	O	AC18	P9TXD2	3.3V	O	D26	VSS	Ground		AC07
DATA40	3.3V	I/O	F02	P4COL	3.3V	I	AD21	P9TXD3	3.3V	O	F24	VSS	Ground		AC10
DATA41	3.3V	I/O	F01	P4CRS	3.3V	I	AE22	P9TXEN	3.3V	O	E24	VSS	Ground		AC17
DATA42	3.3V	I/O	E02	P4RXCLK	3.3V	I	AC22	P10COL	3.3V	I	A24	VSS	Ground		AC20
DATA43	3.3V	I/O	E01	P4RXD0	3.3V	I	AF26	P10CRS	3.3V	I	B24	VSS	Ground		AE03
DATA44	3.3V	I/O	D02	P4RXD1	3.3V	I	AE25	P10RXCLK	3.3V	I	C21	VSS	Ground		D06
DATA45	3.3V	I/O	D01	P4RXD2	3.3V	I	AE26	P10RXD0	3.3V	I	B21	VSS	Ground		D07
DATA46	3.3V	I/O	C02	P4RXD3	3.3V	I	AD25	P10RXD1	3.3V	I	A21	VSS	Ground		D17
DATA47	3.3V	I/O	C01	P4RXDV	3.3V	I	AD23	P10RXD2	3.3V	I	B20	VSS	Ground		D23
DATA48	3.3V	I/O	B02	P4RXER	3.3V	I	AE24	P10RXD3	3.3V	I	A20	VSS	Ground		F04
DATA49	3.3V	I/O	B01	P4TXCLK	3.3V	I	AF25	P10RXDV	3.3V	I	A22	VSS	Ground		G23
DATA50	3.3V	I/O	A02	P4TXD0	3.3V	O	AD22	P10RXER	3.3V	I	D21	VSS	Ground		J04
DATA51	3.3V	I/O	A01	P4TXD1	3.3V	O	AC21	P10TXCLK	3.3V	I	B22	VSS	Ground		M04
LED0	3.3V	I/O	L04	P4TXD2	3.3V	O	AE23	P10TXD0	3.3V	O	A23	VSS	Ground		N23
LED1	3.3V	I/O	L03	P4TXD3	3.3V	O	AF23	P10TXD1	3.3V	O	B23	VSS	Ground		U04
LED2	3.3V	I/O	H03	P4TXEN	3.3V	O	AF24	P10TXD2	3.3V	O	D22	VSS	Ground		U23
LED3	3.3V	I/O	H04	P5COL	3.3V	I	AC24	P10TXD3	3.3V	O	C23	VSS	Ground		V04
LEDCLK	3.3V	I/O	T03	P5CRS	3.3V	I	AD26	P10TXEN	3.3V	O	C22	VSS	Ground		Y23
LEDVLD0	3.3V	I/O	N04	P5RXCLK	3.3V	I	AB26	P11COLR	3.3V	I/O	D19	WCHDOG	3.3 V	O	AA04

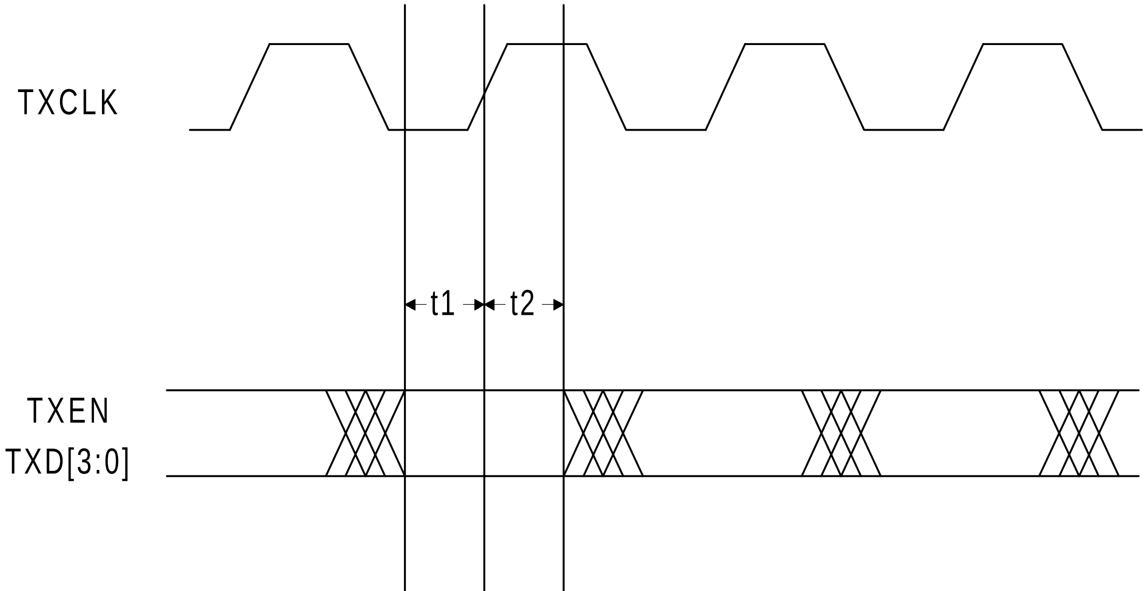
9. TIMING DESCRIPTION

MII Receive Timing



T#	Description:	MIN	TYP	MAX	UNIT
t1	RX_DV, RXD, RX_ER setup time	5	-	-	ns
t2	RX_DV, RXD, RX_ER hold time	5	-	-	ns

MII Transmit Timing

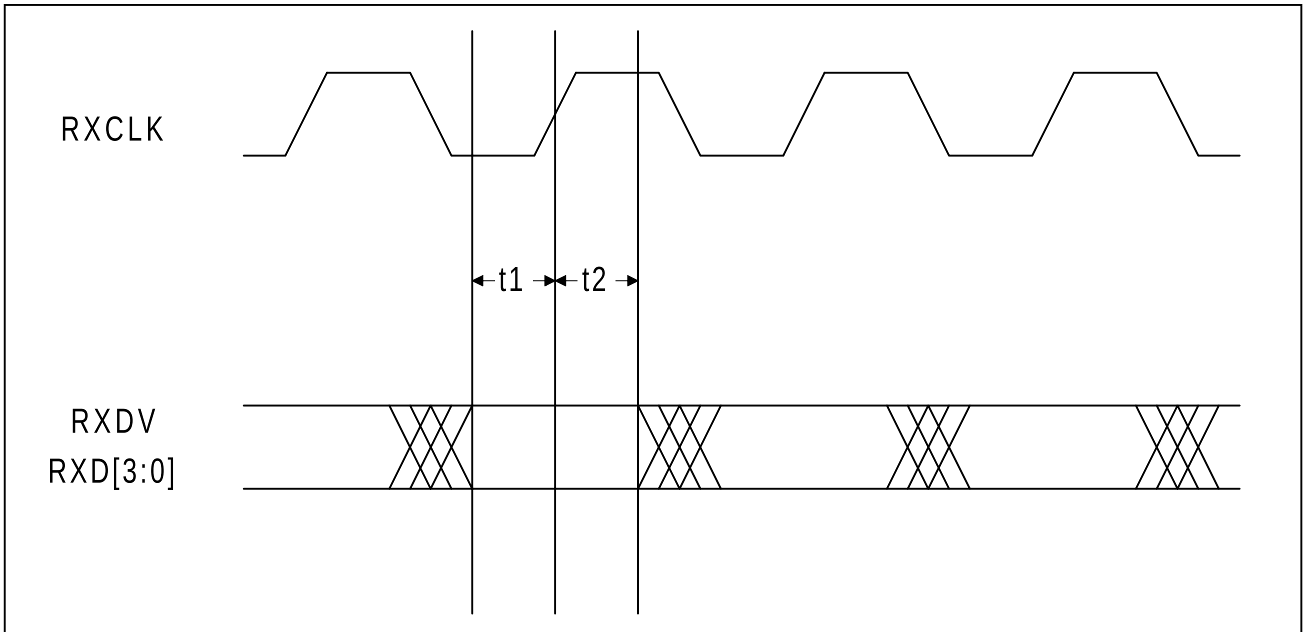


T#	Desciption	Min	Typ	Max	Unit
t1	TXEN, TXD setup time	10	-	-	ns
t2	TXEN, TXD hold time	10	-	-	ns

Data Sheet: ACD82112

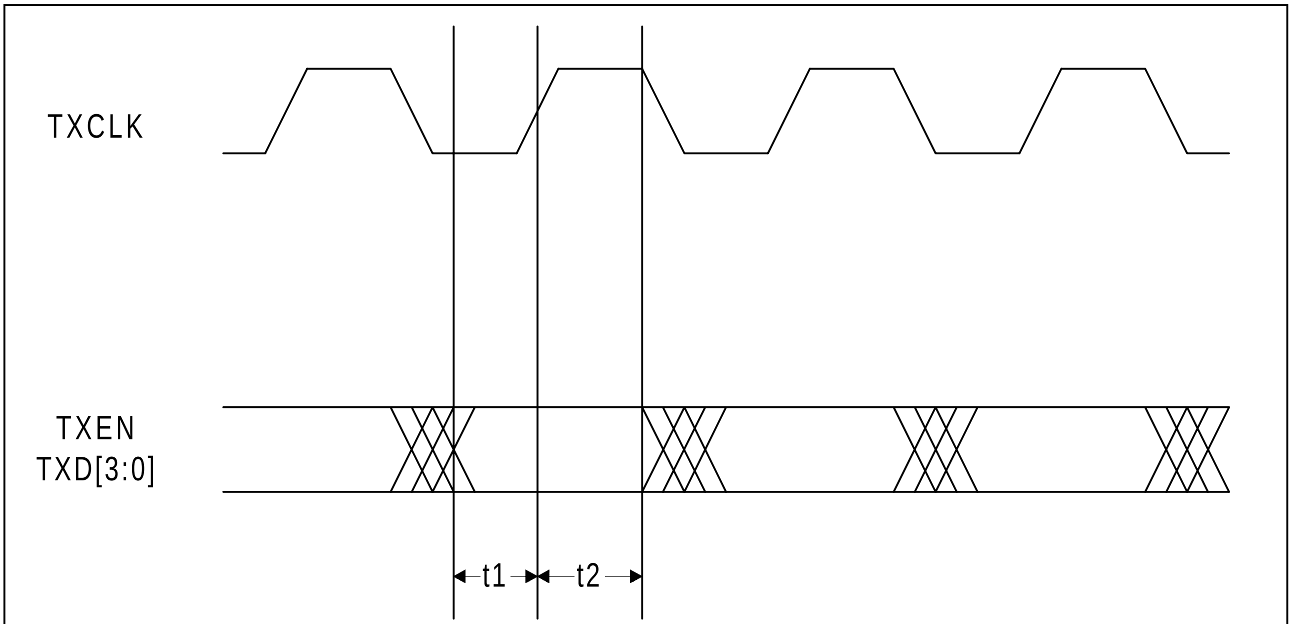
INTRODUCTORY

Reversed MII Receive Timing



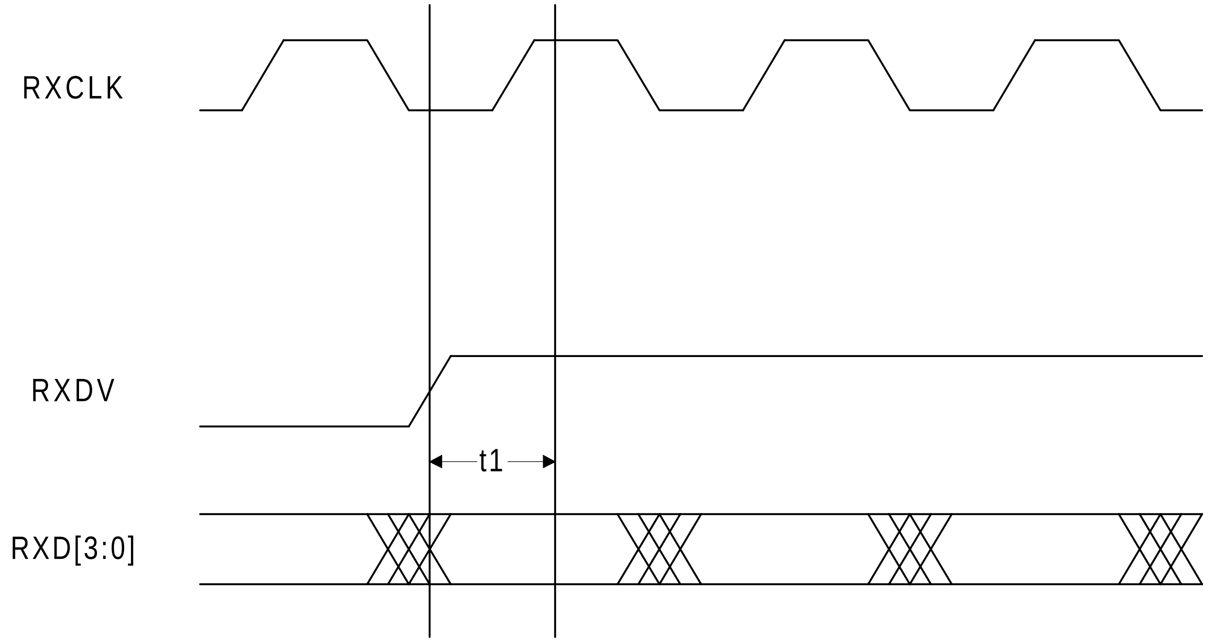
T#	Description:	MIN	TYP	MAX	UNIT
t1	RXDV, RXD setup time	10	-		ns
T2	RXDV, RXD hold time	10			ns

Reversed MII Transmit Timing



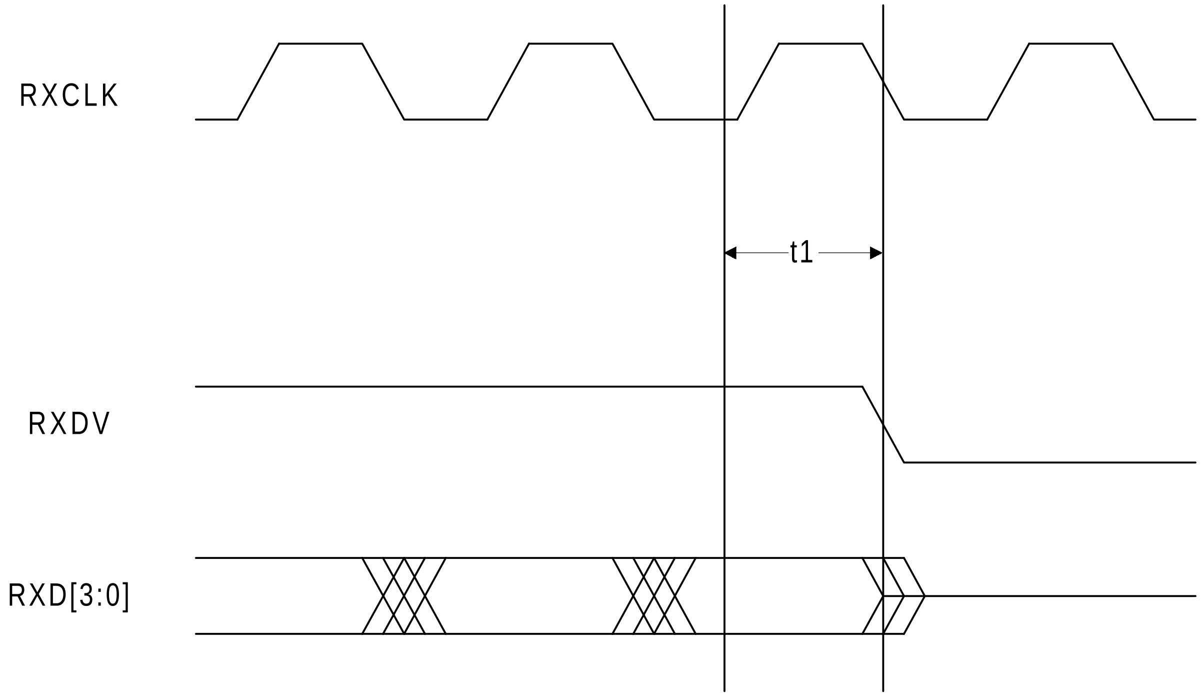
T#	Description:	MIN	TYP	MAX	UNIT
t1	RXDV, RXD setup time	5	-		ns
T2	RXDV, RXD hold time	5			ns

Reversed MII Packet Timing (Start of Packet)



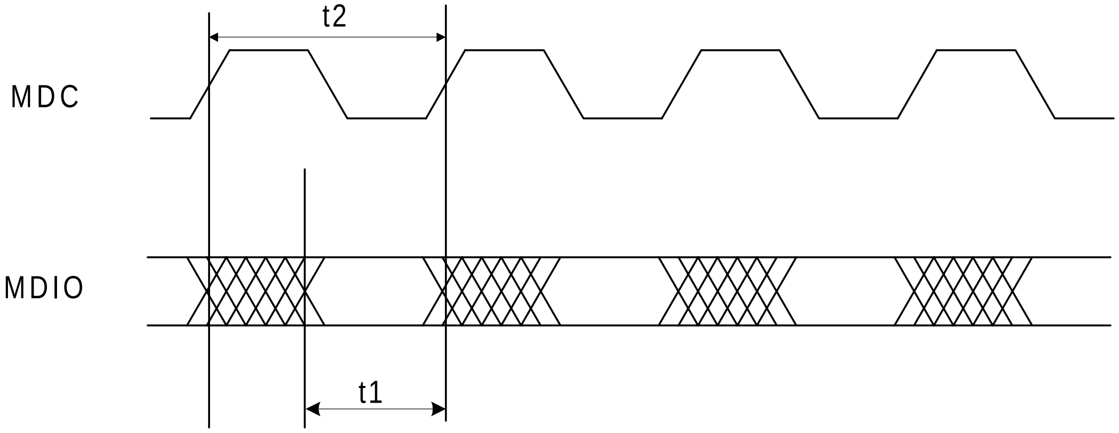
T#	Desciption	Min	Typ	Max	Unit
t1	RXD to RXDV	0	-	-	ns

Reversed MII Packet Timing (End of Packet)



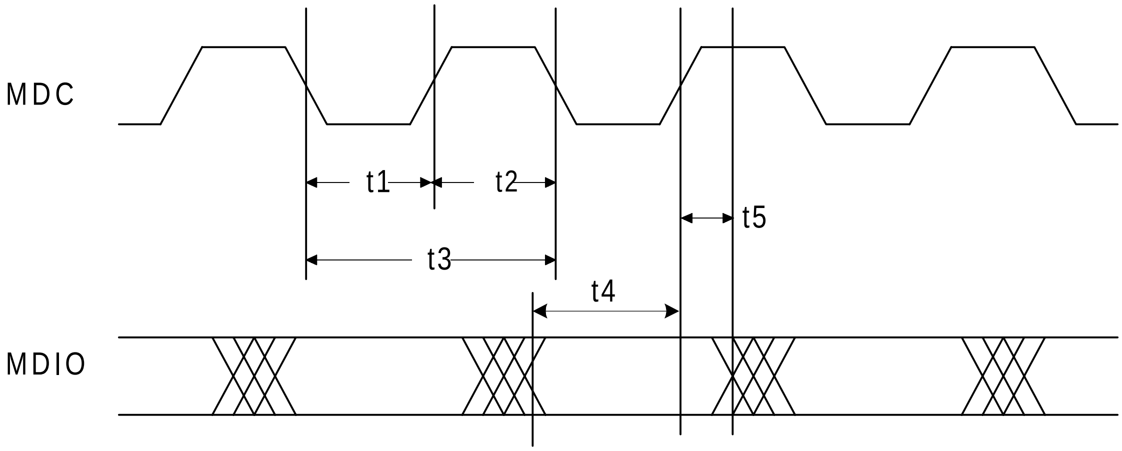
T#	Desciption	Min	Typ	Max	Unit
t1	PXD to RXDV delay time	0	-	-	ns

PHY Management Read Timing



T#	Description	MIN	TYP	MAX	UNIT
t1	MDIO setup time	0	-	300	ns
t2	MDC cycle	-	800	-	ns

PHY Management Write Timing



T#	Description	MIN	TYP	MAX	UNIT
t1	MDC High time	300	-	500	ns
t2	MDC Low time	300	-	500	ns
t3	MDC period	-	800	-	ns
t4	MDIO set up time	10	-	-	ns
t5	MDIO hold time	10	-	-	ns

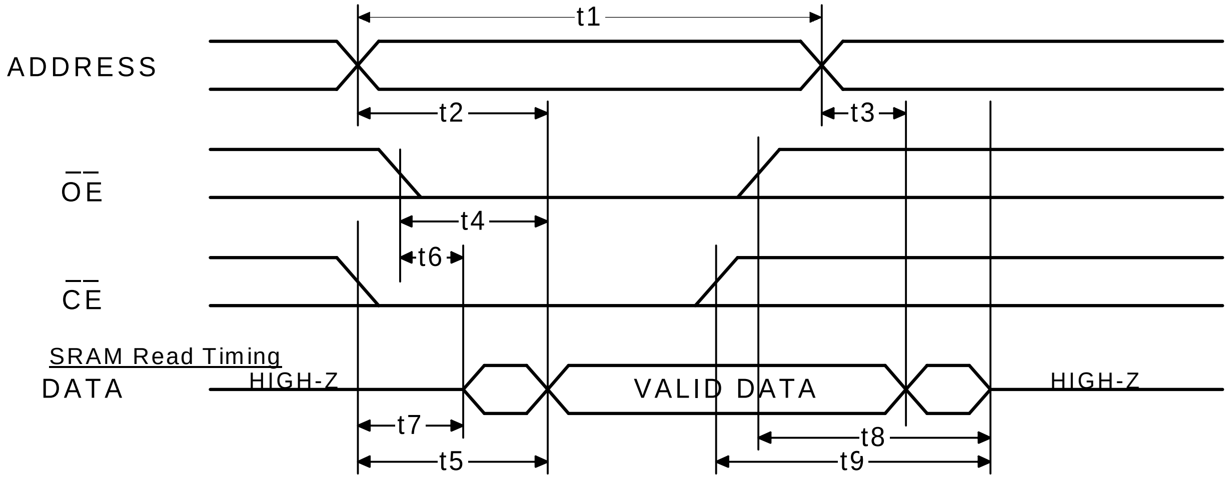
Data Sheet: ACD82112

INTRODUCTORY

Data Sheet: ACD82112

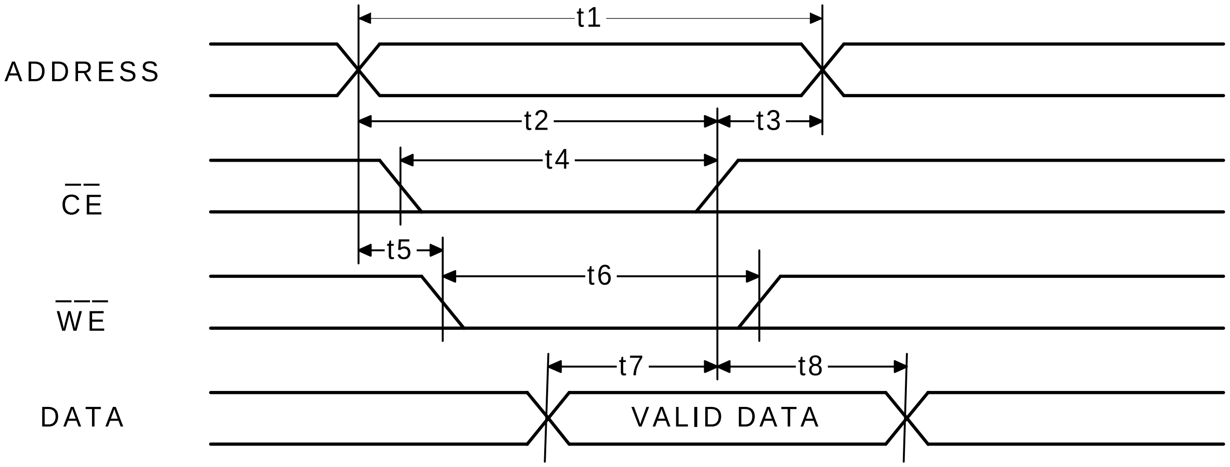
INTRODUCTORY

SRAM Read Timing



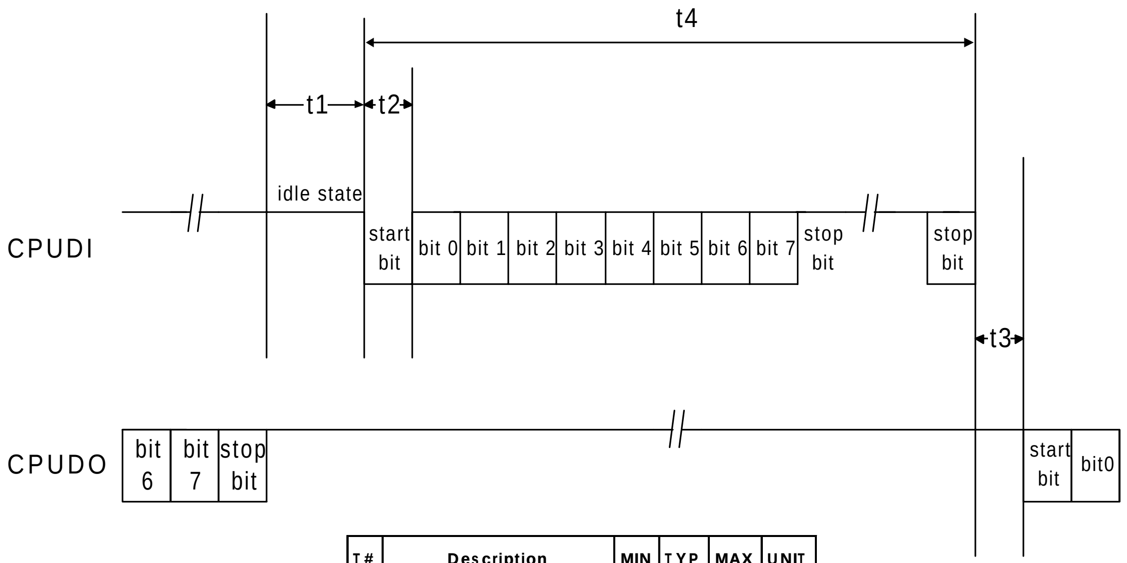
T#	Description	MIN	TYP	MAX	UNIT
t1	Read cycle time	-	20	-	ns
t2	Address access time	-	-	12	ns
t3	Output hold time	0	-	-	ns
t4	OE access time	-	-	12	ns
t5	CE access time	-	-	12	ns
t6	OE to Low-Z output	0	-	-	ns
t7	CE to Low-Z output	0	-	-	ns
t8	OE to High-Z output	-	-	6	ns
t9	CE to High-Z output	-	-	6	ns

SRAM Write Timing



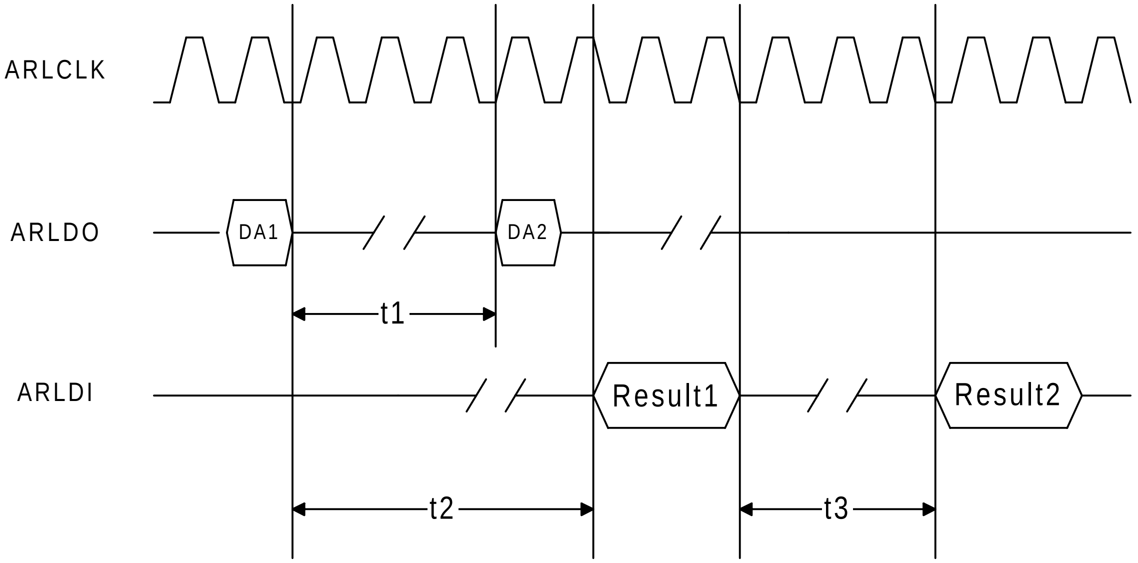
T#	Description	MIN	TYP	MAX	UNIT
t1	Write cycle time	-	20	-	ns
t2	Address Setup to Write End time	12	-	-	ns
t3	Address hold for Write End time	0	-	-	ns
t4	CE to Write End time	12	-	-	ns
t5	Address Setup time	4	-	-	ns
t6	WE pulse width	8	-	-	ns
t7	Data Setup to Write End	8	-	-	ns
t8	Data Hold for Write End	0	-	-	ns

CPU Command Timing



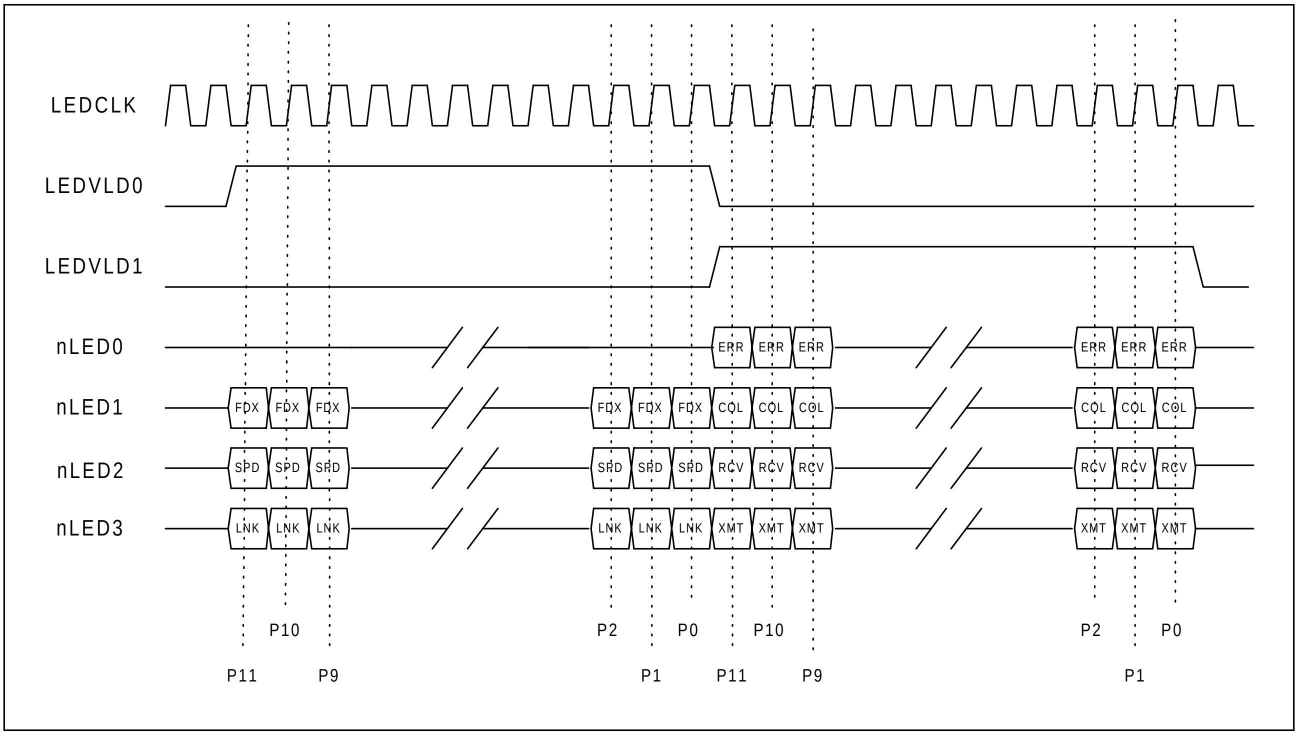
T #	Description	MIN	TYP	MAX	UNIT
t1	CPU idle time	0	-	-	us
t2	CPU command bit time	10	-	1000	us
t3	Response time	0	-	20	ms
t4	Command time	-	-	20	ms

ARL Result Timing



T #	Description	MIN	TYP	MAX	UNIT
t1	time between DAs	0	-	-	ns
t2	time for ARL result	0	-	200	ns
t3	time between results	0	-	-	ns

LED Signal Timing



10. ELECTRICAL SPECIFICATION

Absolute Maximum Ratings

Operation at absolute maximum ratings is not implied exposure to stresses outside those listed could cause permanent damage to the device.

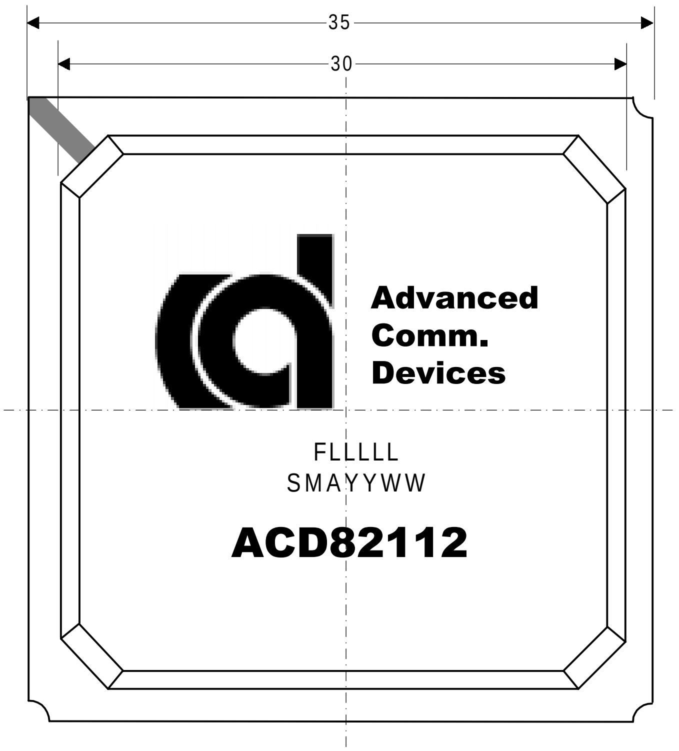
DC Supply voltage : VDD	-0.3V ~ +5.0V
DC input current: Iin	+/-10 mA
DC input voltage: Vin	-0.3 ~ VDD + 0.3V
DC output voltage: Vout	-0.3 ~ VDD + 0.3V
Storage temperature: Tstg	-40 to +125^{o}C

Recommended Operation Conditions

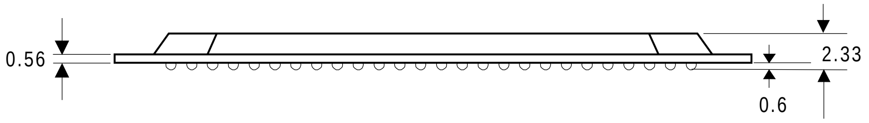
Supply voltage: VDD	3.3V, +/-0.3V
Operating temperature: Ta	0^{o}C -70 ^{o}C
Maximum power consumption	3.3W

11. PACKAGING

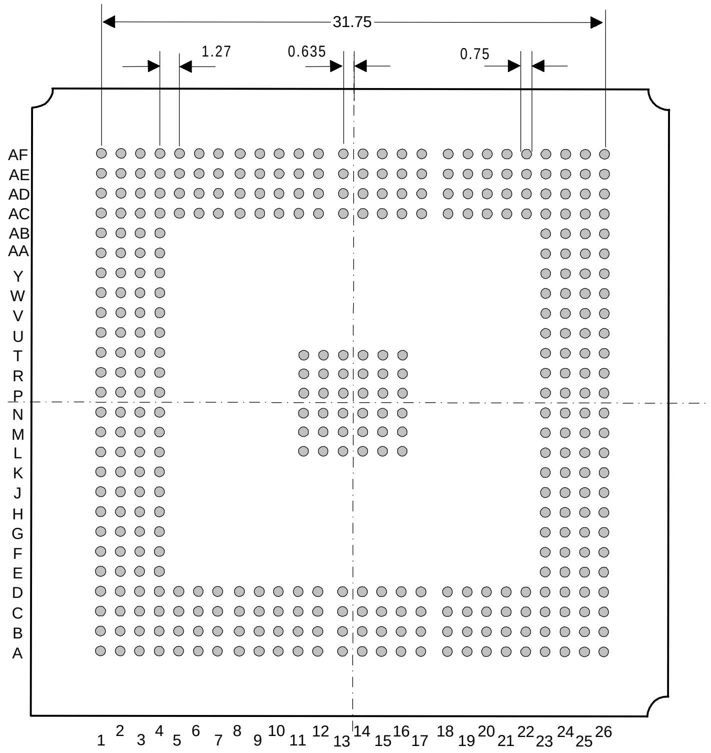
Top View



Side View



Bottom View



Appendix-A1

Address Resolution Logic

(The built-in ARL with 2048 MAC Addresses)

1. SUMMARY

The internal Address Resolution Logic (ARL) of ACD's switch controllers automatically builds up an address table and maps up to 2,048 MAC addresses into their associated port. It can work by itself without any CPU intervention in an UN-managed system.

For a managed system, the management CPU can configure the operation mode of the ARL, learn all the address in the address table, add new address into the table, control security or filtering feature of each address entry etc.

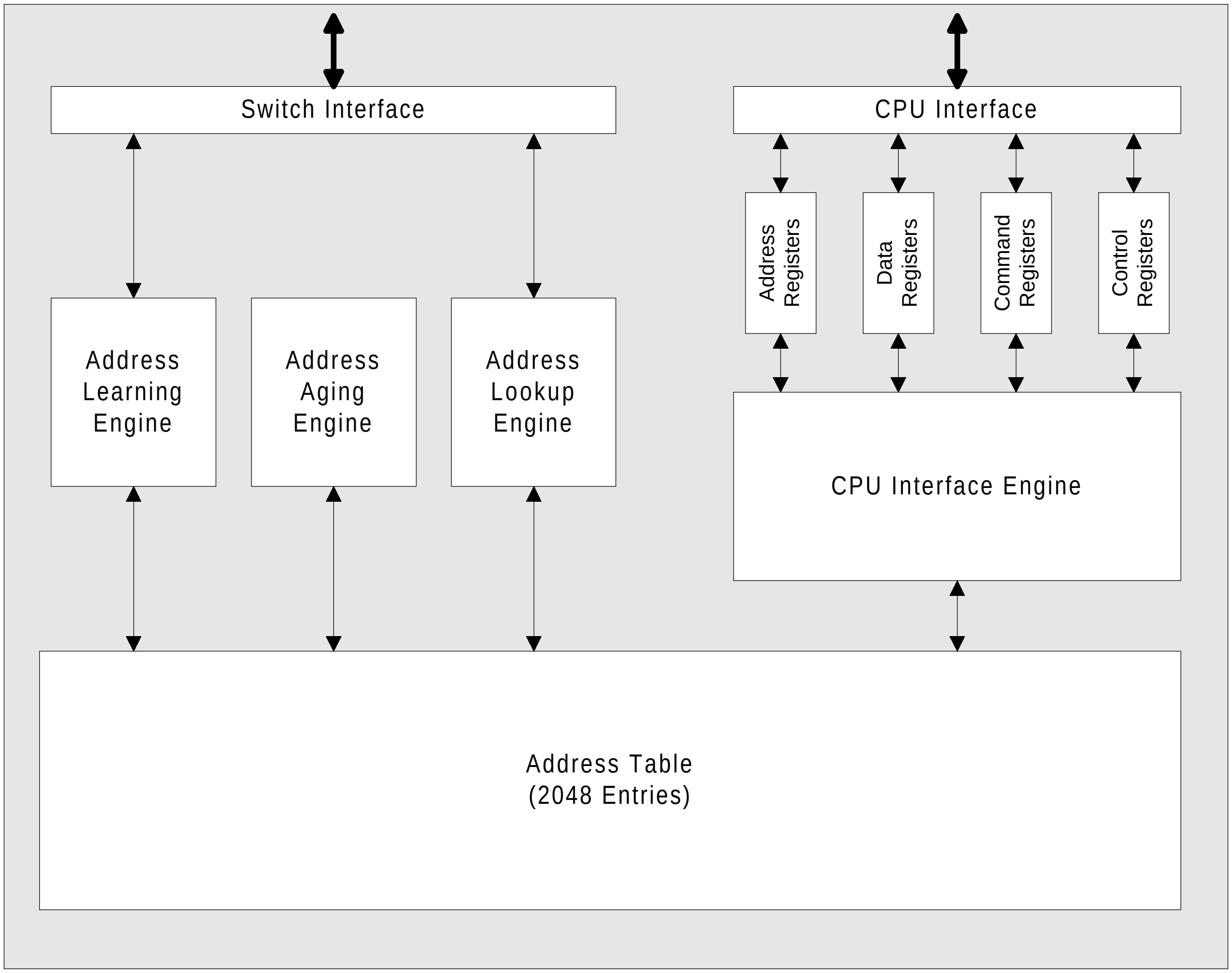
The ARL is designed with such a high performance that it will never slow down the frame switching operation. It helps the switch controllers to reach wire speed forwarding rate under any type of traffic load.

The address space can be expanded to 11K entries by using the external ARL, the ACD80800.

2. FEATURES

- Supports up to 2,048 MAC address lookup
- Provides UART type of interface for the management CPU
- Wire speed address lookup time.
- Wire speed address learning time.
- Address can be automatically learned from switch without the CPU intervention
- Address can be manually added by the CPU through the CPU interface
- Each MAC address can be secured by the CPU from being changed or aged out
- Each MAC address can be marked by the CPU from receiving any frame
- Each newly learned MAC address is notified to the CPU
- Each aged out MAC address is notified to the CPU
- Automatic address aging control, with configurable aging period

Figure-1. ARL Block Diagram



3. FUNCTIONAL DESCRIPTION

The ARL provides Address Resolution service for ACD's switch controllers. *Figure 2* is a block diagram of the ARL.

Traffic Snooping

All Ethernet frames received by ACD's switch controller have to be stored into memory buffer. As the frame data are written into memory, the status of the data shown on the data bus are displayed by ACD's switch controller through a state bus. The ARL's Switch Controller Interface contains the signals of the data bus and the state bus. By snooping the data bus and the state bus of ACD's switch controller, the ARL can detect the occurrence of any destination MAC address and source MAC address embedded inside each frame.

Address Learning

Each source address caught from the data bus, together with the ID of the ingress port, is passed to the Address Learning Engine of the ARL. The Address Learning Engine will first determine whether the frame is a valid frame. For a valid frame, it will first try to find the source address from the current address table. If that address doesn't exist, or if it does exist but the port ID associated with the MAC address is not the ingress port, the address will be learned into the address table. After an address is learned by the address learning engine, the CPU will be notified to read this newly learned address so that it can add it into the CPU's address table.

Address Aging

After each source address is learned into the address table, it has to be refreshed at least once within each address aging period. Refresh means it is caught again from the switch interface. If it has not occurred for a pre-set aging period, the address aging engine will remove the address from the address table. After an address is removed by the address aging engine, the CPU will be notified through interrupt request that it needs to read this aged out address so that it can remove this address from the CPU's address table.

Address Lookup

Each destination address is passed to the Address Lookup Engine of the ARL. The Address Lookup Engine checks if the destination address matches with any existing address in the address table. If it does, the ARL returns the associated Port ID to ACD's switch controller through the output data bus. Otherwise, a no match result is passed to ACD's switch controller through the output data bus.

CPU Interface

The CPU can access the registers of the ARL by sending commands to the UART data input line. Each command is consisted by action (read or write), register type, register index, and data. Each result of command execution is returned to the CPU through the UART data output line.

CPU Interface Registers

The ARL provides a bunch of registers for the control CPU. Through the registers, the CPU can read all address entries of the address table, delete particular addresses from the table, add particular addresses into the table, secure an address from being changed, set filtering on some addresses, change the hashing algorithm etc. Through a proper interrupt request signal, the CPU can be notified whenever it needs to retrieve data for a newly-learned address or an aged-out address so that the CPU can build an exact same address table learned by the ARL.

CPU Interface Engine

The command sent by the control CPU is executed by the CPU Interface Engine. For example, the CPU may send a command to learn the first newly-learned address. The CPU Interface Engine is responsible to find the newly-learned address from the address table, and passes it to CPU. The CPU may request to learn next newly-learned address. Then, it is again the responsibility of the CPU Interface Engine to search for next newly-learned address from the address table.

Address Table

The address table can hold up to 2,048 MAC addresses, together with the associated port ID, security flag, filtering flag, new flag, aging information etc. The address table resides in the embedded SRAM inside the ARL.

4. INTERFACE DESCRIPTION

CPU Interface

The CPU can communicate with the ARL through the UART interface of the switch IC. The management CPU can send command to the ARL by writing into associated registers, and retrieve result from ARL by reading corresponding registers. The registers are described in the section of “Register Description.” The CPU interface signals are described by *table-1*:

Table-1: CPU Interface

Name	I/O	Description
UARTDI	I	UART input data line.
UARTDO	O	UART output data line.

UARTDI is used by the control CPU to send command into the ARL. The baud rate will be automatically detected by the ARL. The result will be returned through the UARTDO line with the detected baud rate. The format of the command packet is shown as follows:

Header	Address	Data	Checksum

where:

- Header is further defined as:
  b1:b0 - read or write, 01 for read, 11 for write
  b4:b2 - device number, 000 to 111 (0 to 7, same as the host switch controller)
  b7:b5 - device type, 010 for ARL
- Address - 8-bit value used to select the register to access
- Data - 32-bit value, only the LSB is used for write operation, all 0 for read operation
- Checksum - 8-bit value of XOR of all bytes

UARTDO is used to return the result of command execution to the CPU. The format of the result packet is shown as follows:

Header	Address	Data	Checksum

where:

- Header is further defined as:
  b1:b0 - read or write, 01 for read, 11 for write
  b4:b2 - device number, 000 to 111 (0 to 7)
  b7:b5 - device type, 010 for ARL
- Address - 8-bit value for address of the selected register
- Data - 32-bit value, only the LSB is used for read operation, all 0 for write operation
- Checksum - 8-bit value of XOR of all bytes

The ARL will always check the CMD header to see if both the device type and the device number matches with its setting. If not, it ignores the command and will not generate any response to this command.

5. REGISTER DESCRIPTION

ACD80800 provides a bunch of registers for the CPU to access the address table inside it. Command is sent to ACD80800 by writing into the associated registers. Before the CPU can pass a command to ACD80800, it must check the result register *(register 11)* to see if the command has been done. When the Result register indicates the command has been done, the CPU may need to retrieve the result of previous command first. After that, the CPU has to write the associated parameter of the command into the Data registers. Then, the CPU can write the command type into the command register. When a new command is written into the command register, ACD80800 will change the status of the Result register to 0. The Result register will indicate the completion of the command at the end of the execution. Before the completion of the execution, any command written into the command register is ignored by ACD80800.

The registers accessible to the CPU are described by *table-2*:

Table-2: Register Description

Reg.	Name	Description
0	DataReg0	Byte 0 of data
1	DataReg1	Byte 1 of data
2	DataReg2	Byte 2 of data
3	DataReg3	Byte 3 of data
4	DataReg4	Byte 4 of data
5	DataReg5	Byte 5 of data
6	DataReg6	Byte 6 of data
7	DataReg7	Byte 7 of data
8	AddrReg0	LSB of address value
9	AddrReg1	MSB of address value
10	CmdReg	Command register
11	RsltReg	Result register
12	CfgReg	Configuration register
13	IntSrcReg	Interrupt source register
14	IntMskReg	Interrupt mask register
15	nLearnReg0	Address learning disable register for port 0 - 7
16	nLearnReg1	Address learning disable register for port 8 - 15
17	nLearnReg2	Address learning disable register for port 16 - 23
18	AgeTimeReg0	LSB of aging period register
19	AgeTimeReg1	MSB of aging period register
20	PosCfg	Power On Strobe configuration register 0

The *DataRegX* are registers used to pass the parameter of the command to the ACD80800, and the result of the command to the CPU.

The *AddrRegX* are registers used to specify the address associated with the command.

The *CmdReg* is used to pass the type of command to the ACD80800. The command types are listed in *table-3*. The details of each command is described in the chapter of “Command Description.”

Table-3: Command List

Command	Description
0x09	Add the specified MAC address into the address table
0x0A	Set a lock for the specified MAC address
0x0B	Set a filtering flag for the specified MAC address
0x0C	Delete the specified MAC address from the address table
0x0D	Assign a port ID to the specified MAC address
0x10	Read the first entry of the address table
0x11	Read next entry of address book
0x20	Read first valid entry
0x21	Read next valid entry
0x30	Read first new page
0x31	Read next new page
0x40	Read first aged page
0x41	Read next aged page
0x50	Read first locked page
0x51	Read next locked page
0x60	Read first filtered page
0x61	Read next filtered page
0x80	Read first page with specified PID
0x81	Read next page with specified PID
0xFF	System reset

The *RstReg* is used to indicate the status of command execution. The result code is listed as follows:

- *01 - command is being executed and is not done yet*
- *10 - command is done with no error*
- *1x - command is done, with error indicated by x, where x is a 4-bit error code: 0001 for cannot find the entry as specified*

The *CfgReg* is used to configure the way the ACD80800 works. The bit definition of CfgReg is described as:

- *bit 0 - disable address aging*
- *bit 1 - disable address lookup*
- *bit 2 - disable DA cache*
- *bit 3 - disable SA cache*
- *bit 7:4 - hashing algorithm selection, default is 0000*

The *IntSrcReg* is used to indicate what can cause interrupt request to CPU. The source of interrupt is listed as:

- *bit 0 - aged address exists*
- *bit 1 - new address exists*
- *bit 2 - reserved*
- *bit 3 - reserved*
- *bit 4 - bucket overflowed*
- *bit 5 - command is done*
- *bit 6 - system initialization is completed*
- *bit 7 - self test failure*

The *IntMskReg* is used to enable an interrupt source to generate an interrupt request. The bit definition is the same as IntSrcReg. A 1 in a bit enables the corresponding interrupt source to generate an interrupt request once it is set.

The *nLearnReg[2:0]* are used to disable address learning activity from a particular port. If the bit corresponding to a port is set, ACD80800 will not try to learn new addresses from that port.

The *AgeTimeReg[1:0]* are used to specify the period of address aging control. The aging period can be from 0 to 65535 units, with each unit counted as 2.684 second.

The *PosCfgReg* is a configuration register whose default value is determined by the pull-up or pull-down status of the associated hardware pin. The bits of PosCfgReg0 is listed as follows:

- *bit 1* - NOCPU*, "0" = presence of control CPU, "1" = no control CPU;*
- *bit 0 - CPUGO, "0" = wait for System Start command from CPU before starting self initialization, "1" = CPU ready. Only effective when bit-1 (NOCPU) is set to 0;*

Note: When *NOCPU* is set as 0, ACD80800 will not start the initialization process until a System Start command is sent to the command register.

6. COMMAND DESCRIPTION

Command 09H

Description: Add the specified MAC address into the address table.

Parameter: Store the MAC address into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB. Store the associated port number into DataReg6.

Result: the MAC address will be stored into the address table if there is space available. The result is indicated by the Result register.

Command 0AH

Description: Set the Lock bit for the specified MAC address.

Parameter: Store the MAC address into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB.

Result: the state machine will seek for an entry with matched MAC address, and set the Lock bit of the entry. The result is indicated by the Result register.

Command 0BH

Description: Set the Filter flag for the specified MAC address.

Parameter: Store the MAC address into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB.

Result: the state machine will seek for an entry with matched MAC address, and set the Filter bit of the entry. The result is indicated by the Result register.

Command 0CH

Description: Delete the specified MAC address from the address table.

Parameter: Store the MAC address into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB.

Result: the MAC address will be removed from the address table. The result is indicated by the Result register.

Command 0DH

Description: Assign the associated port number to the specified MAC address.

Parameter: Store the MAC address into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB. Store the port number into DataReg6.

Result: the port ID field of the entry containing the specified MAC address will be changed accordingly. The result is indicated by the Result register.

Command 10H

Description: Read the first entry of the address table.

Parameter: None

Result: The result is indicated by the Result register. If the command is completed with no error, the content of the first entry of the address book will be stored into the Data registers. The MAC address will be stored into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB. The port number is stored in DataReg6, and the Flag[*] bits are stored in DataReg7.The Read Pointer will be set to point to second entry of the address book.

Note - the Flag bits are defined as:

b7	b6	b5	b4	b3	b2	b1	b0
Rsvd	Rsvd	Filter	Lock	New	Old	Age	Valid

where:

- Filter - 1 indicates the frame heading to this address should be dropped.
- Lock - 1 indicates the entry should never be changed or aged out.
- New - 1 indicates the entry is a newly learned address.
- Old - 1 indicates the address has been aged out.
- Age - 1 indicates the address has not been visited for current age cycle.
- Valid - 1 indicates the entry is a valid one.
- Rsvd - Reserved bits.

Command 11H

Description: Read next entry of address book.

Parameter: None

Result: The result is indicated by the Result register. If the command is completed with no error, the content of the address book entry pointed by Read Pointer will be stored into the Data registers. The MAC address will be stored into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB. The port number is stored in DataReg6, and the Flag bits are stored in DataReg7. The Read Pointer will be increased by one.

Command 20H

Description: Read first valid entry.

Parameter: None

Result: The result is indicated by the Result register. If the command is completed with no error, the content of first valid entry of the address book will be stored into the Data registers. The MAC address will be stored into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB. The port number is stored in DataReg6, and the Flag bits are stored in DataReg7. The Read Pointer is set to point to this entry.

Command 21H

Description: Read next valid entry.

Parameter: None

Result: The result is indicated by the Result register. If the command is completed with no error, the content of next valid entry from the Read Pointer of the address book will be stored into the Data registers. The MAC address will be stored into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB. The port number is stored in DataReg6, and the Flag bits are stored in DataReg7. The Read Pointer is set to point to this entry.

Command 30H

Description: Read first new page.

Parameter: None

Result: The result is indicated by the Result register. If the command is completed with no error, the content of first new entry of the address book will be stored into the Data registers. The MAC address will be stored into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB. The port number is stored in DataReg6, and the Flag bits are stored in DataReg7. The Read Pointer is set to point to this entry.

Command 31H

Description: Read next new entry.

Parameter: None

Result: The result is indicated by the Result register. If the command is completed with no error, the content of next new entry from the Read Pointer of the address book will be stored into the Data registers. The MAC address will be stored into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB. The port number is stored in DataReg6, and the Flag bits are stored in DataReg7. The Read Pointer is set to point to this entry.

Command 40H

Description: Read first aged entry.

Parameter: None

Result: The result is indicated by the Result register. If the command is completed with no error, the content of first aged entry of the address book will be stored into the Data registers. The MAC address will be stored into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB. The port number is stored in DataReg6, and the Flag bits are stored in DataReg7. The Read Pointer is set to point to this entry.

Command 41H

Description: Read next aged entry.

Parameter: None

Result: The result is indicated by the Result register. If the command is completed with no error, the content of next aged entry from the Read Pointer of the address book will be stored into the Data registers. The MAC address will be stored into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB. The port number is stored in DataReg6, and the Flag bits are stored in DataReg7. The Read Pointer is set to point to this entry.

Command 50H

Description: Read first locked entry.

Parameter: None

Result: The result is indicated by the Result register. If the command is completed with no error, the content of first locked entry of the address book will be stored into the Data registers. The MAC address will be stored into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB. The port number is stored in DataReg6, and the Flag bits are stored in DataReg7. The Read Pointer is set to point to this entry.

Command 51H

Description: Read next locked entry.

Parameter: None

Result: The result is indicated by the Result register. If the command is completed with no error, the content of next locked entry from the Read Pointer of the address book will be stored into the Data registers. The MAC address will be stored into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB. The port number is stored in DataReg6, and the Flag bits are stored in DataReg7. The Read Pointer is set to point to this entry.

Command 60H

Description: Read first filtered page.

Parameter: None

Result: The result is indicated by the Result register. If the command is completed with no error, the content of first filtered entry of the address book will be stored into the Data registers. The MAC address will be stored into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB. The port number is stored in DataReg6, and the Flag bits are stored in DataReg7. The Read Pointer is set to point to this entry.

Command 61H

Description: Read next valid entry.

Parameter: None

Result: The result is indicated by the Result register. If the command is completed with no error, the content of next filtered entry from the Read Pointer of the address book will be stored into the Data registers. The MAC address will be stored into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB. The port number is stored in DataReg6, and the Flag bits are stored in DataReg7. The Read Pointer is set to point to this entry.

Command 80H

Description: Read first entry with specified port number.

Parameter: Store port number into DataReg6.

Result: The result is indicated by the Result register. If the command is completed with no error, the content of first entry of the address book with the said port number will be stored into the Data registers. The MAC address will be stored into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB. The port number is stored in DataReg6, and the Flag bits are stored in DataReg7. The Read Pointer is set to point to this entry.

Command 81H

Description: Read next valid entry.

Parameter: Store port number into DataReg6.

Result: The result is indicated by the Result register. If the command is completed with no error, the content of next entry from the Read Pointer of the address book with the said port number will be stored into the Data registers. The MAC address will be stored into DataReg5 - DataReg0, with DataReg5 contains the MSB of the MAC address and DataReg0 contains the LSB. The port number is stored in DataReg6, and the Flag bits are stored in DataReg7. The Read Pointer is set to point to this entry.

Command FFH

Description: System reset.

Parameter: None

Result: This command will reset the ARL system. All entries of the address book will be cleared.