**ETHERNET PROTOCOL**

1. **What is Ethernet?**

Ethernet is a widely used technology for connecting devices in a local area network (LAN). It uses a wired connection to transmit data packets between devices over a network cable, typically using twisted pair wiring or fiber optic cables.

1. **Ethernet technologies**

* PoE (Power over Ethernet): This technology allows Ethernet devices to receive power over the same Ethernet cable that is used for data transmission. This can be useful for powering devices like IP cameras, access points, and VoIP phones, without requiring a separate power source.
* AVB (Audio Video Bridging): This is a set of standards that adds real-time audio and video capabilities to Ethernet networks. AVB allows for precise timing and synchronization of audio and video data, making it useful for applications like streaming media, conferencing, and live performances.
* TSN (Time-Sensitive Networking): This is another set of standards that adds real-time capabilities to Ethernet networks. TSN allows for precise timing and synchronization of data, making it useful for applications like industrial automation, robotics, and transportation systems.

1. **MAC address**

* A MAC address, or Media Access Control address, is a unique identifier assigned to a network interface controller (NIC) for use as a network address in communications within a network segment. The MAC address is a 48-bit address, usually represented as a series of 12 hexadecimal digits (0-9 and A-F) separated by colons or dashes.
* MAC addresses are used by the Data Link layer of the OSI model to identify individual devices on a network. When a device wants to communicate with another device on the same network, it uses the MAC address of the destination device to address the data packets.

1. **Ethernet Layers**
   1. **Physical layer**

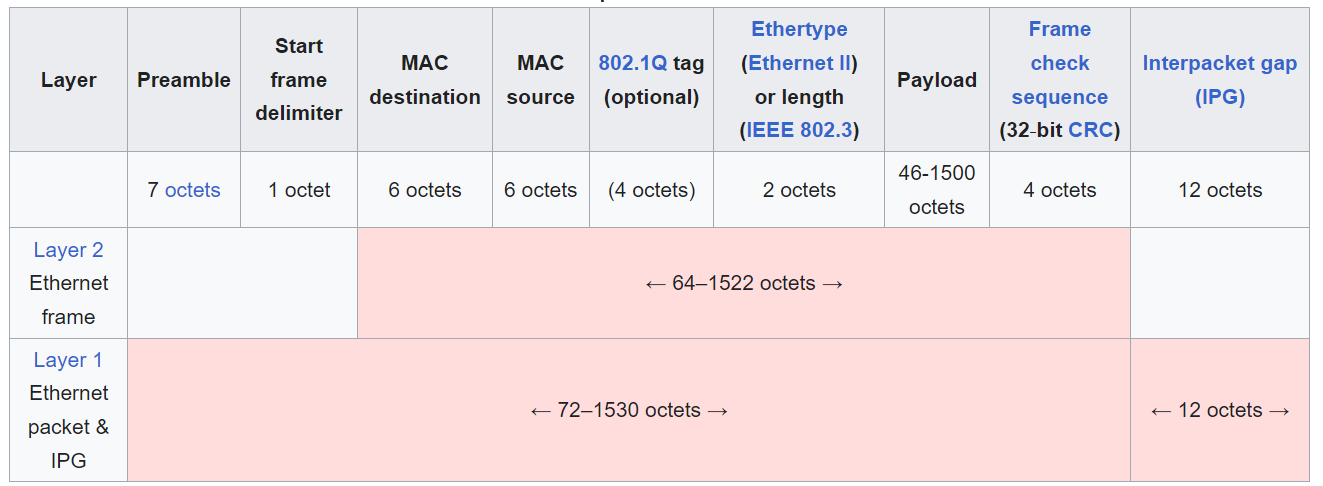
* Include: cable and devices
* Function: encoding, decoding, signal transmission, signal reception, collision detection, physical addressing
  1. **Data link layer**

Sublayer:

* Logical Link Control (LLC): The LLC sublayer is responsible for providing a uniform interface to the network layer, regardless of the specific physical medium being used. The LLC sublayer also handles issues such as error control and flow control.
* Media Access Control (MAC): The MAC sublayer is responsible for controlling access to the physical medium, and ensures that only one device can transmit on the network at any given time. The MAC sublayer also handles issues such as addressing and error detection.

Function: Framing, addressing, error detection and correction, flow control

1. **Ethernet packet and frame structure**



* 1. **Ethernet frame**

Include: Header, Payload, Frame check sequence

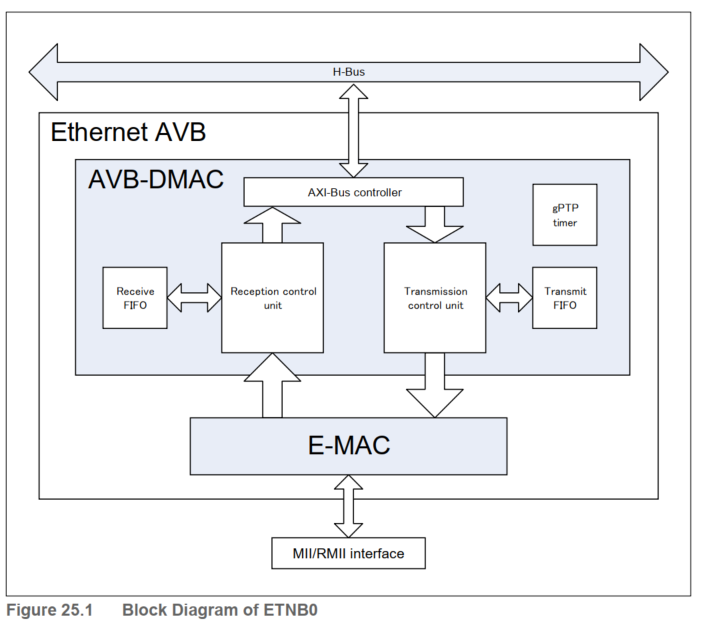
* The header features destination and source MAC addresses (each six octets in length), the EtherType field and, optionally, an IEEE 802.1Q tag or IEEE 802.1ad tag. The EtherType field is two octets long and it can be used for two different purposes. Values of 1500 and below mean that it is used to indicate the size of the payload in octets, while values of 1536 and above indicate that it is used as an EtherType, to indicate which protocol is encapsulated in the payload of the frame. When used as EtherType, the length of the frame is determined by the location of the interpacket gap and valid frame check sequence
* Payload: data
* Frame check sequence is a four-octet cyclic redundancy check (CRC) that allows detection of corrupted data within the entire frame as received on the receiver side
  1. **Ethernet packet**

Include: Preamble, SFD, Ethernet Frame and Interpacket gap

* Preamble: The preamble consists of a 56-bit (seven-byte) pattern of alternating 1 and 0 bits, allowing devices on the network to easily synchronize their receiver clocks, providing bit-level synchronization
* SFD is the eight-bit (one-byte) value that marks the end of the preamble, which is the first field of an Ethernet packet, and indicates the beginning of the Ethernet frame
* Interpacket gap (IPG) is idle time between packets. After a packet has been sent, transmitters are required to transmit a minimum of 96 bits (12 octets) of idle line state before transmitting the next packet.

**ETHERNET RCAR S4**

1. **Ethernet AVB-IF**
   1. **Block Diagram**

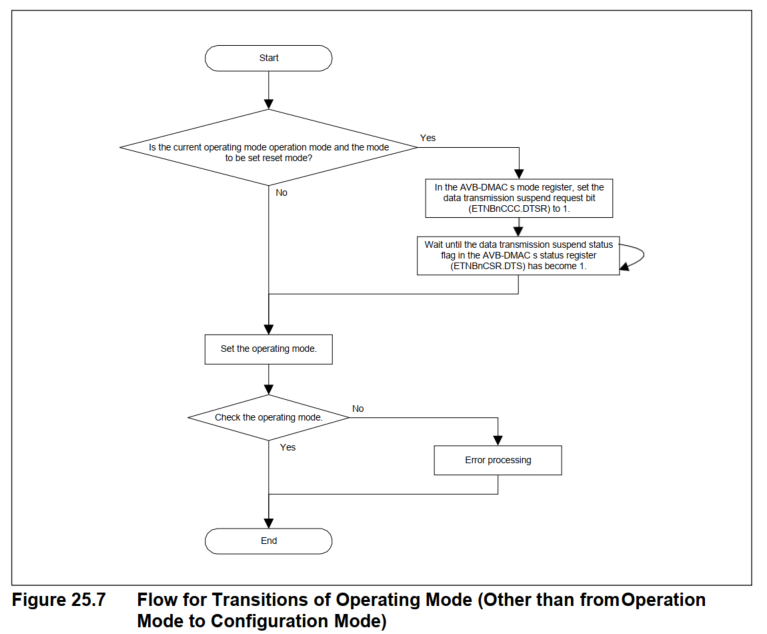


* AVB-DMAC (DMA transfer controller): Handles DMA transfer between the data storage areas for reception and transmission in the URAM and the reception and transmission FIFO buffers
* Receive block: Receive FIFO, Reception control unit - control the process of receiving data frames and storing frames temporarily on FIFO
* Transmit block: Transmit FIFO, Transmission control unit - control the process of Transmitting data frames and storing frames temporarily on FIFO.
* Timer block: The gPTP timer block captures the current time of eth,(capture the exact time to transmit/receive multiple frames)
* E-MAC (MAC controller): Handles transfer between the reception and transmission FIFO buffers and the MII
  1. **Operation**
     1. **Operations mode of AVB-DMAC**

Diagram

Description automatically generated

* Reset mode: in this mode, only the AVB-DMAC operating mode control function is controllable and other functions are all stopped
* Configuration mode: various settings for the AVB-DMAC can be made. The operating functions are stopped, and all status registers are initialized to their reset values
* Operation mode: all functions of the AVB-DMAC can operate. Ethernet communications can only proceed in this mode.
* Standby mode: the E-MAC can only be used to control the operating mode. Other functions cannot be used.
  + 1. **Set the operating mode.**



Diagram

Description automatically generated

* 1. **Common control for Transmission and Reception**
     1. **Initialization Procedure**

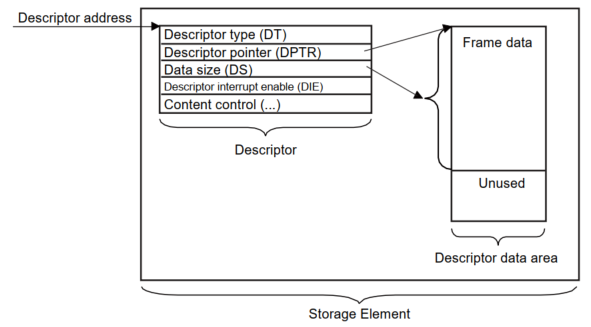
Diagram

Description automatically generated

|  |  |
| --- | --- |
|  |  |
| Procedure for Initializing the Receiver Section | Procedure for Initializing the Transmitter Section |
|  |  |
| Procedure for Configuration the E-MAC Block | Procedure for Configuration the Message Handler Section |
|  |  |

* 1. **Descriptor**
     1. **Data representation in URAM**

The memory in the URAM for use by the AVB-DMAC is configured with control structures referred to as descriptors and associated areas to which the frame data are allocated. Dividing the memory into a control area and data area allows the flexible allocation of frame data to the URAM. This enables sharing of the areas to which frame data are allocated and the use of non-contiguous areas. Frame data can be copied without using the CPU.



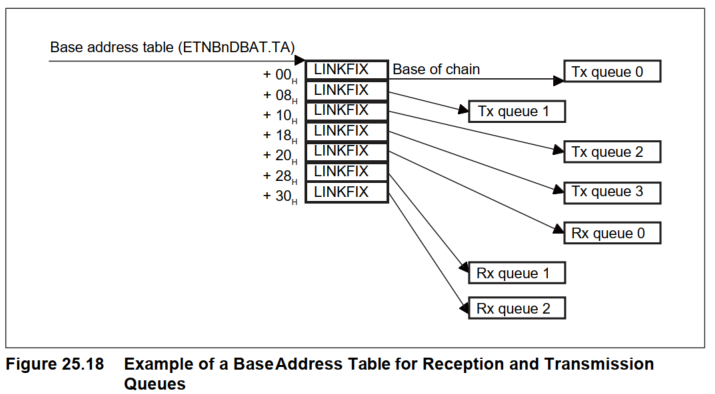
* + 1. **Descriptor chains in queues**

Transmission and reception descriptors in the URAM are grouped into queues. Each queue handles frames so that they are transmitted in order of priority and received separately. A queue can control one or more frames. Accordingly, multiple descriptors can be assigned to one queue. A combination of multiple descriptors is referred to as a descriptor chain.

Diagram, table

Description automatically generated

* + 1. **Descriptor Base Address Table**



* 1. **Control in Reception**
  2. **Transmission Control**
  3. **CBS (Credit-Based Shaping)**
  4. **gPTP Timer**
  5. **Interrupts**
  6. **Flow of Operation**
     1. **E-MAC Initialization**

Diagram

Description automatically generated

* + 1. **AVB-DMAC Initialization**

Diagram, schematic

Description automatically generated

* + 1. **AVB-DMAC in Reception**



* + 1. **AVB-DMAC in Transmission**

Diagram, schematic

Description automatically generated

* + 1. **Stopping AVB-DMAC Operation in Reception**

Diagram

Description automatically generated

* + 1. **Stopping AVB-DMAC Operation in Transmission**

Diagram

Description automatically generated

* + 1. **Stopping and Resetting the AVB-DMAC**

Diagram

Description automatically generated

* + 1. **Emergency Stopping the AVB-DMAC**

Diagram

Description automatically generated

* + 1. **gPTP Initialization**

Diagram

Description automatically generated

* + 1. **gPTP Timestamping in Transmission**

Graphical user interface, diagram

Description automatically generated

* + 1. **gPTP Timestamping and Synchronization in Reception**

Graphical user interface, diagram, application

Description automatically generated

* + 1. **Capturing gPTP presentation Times**

Diagram

Description automatically generated

* + 1. **AVTP presentation time comparison**

A picture containing diagram

Description automatically generated

* + 1. **Loopback Mode Operation**

Graphical user interface, application

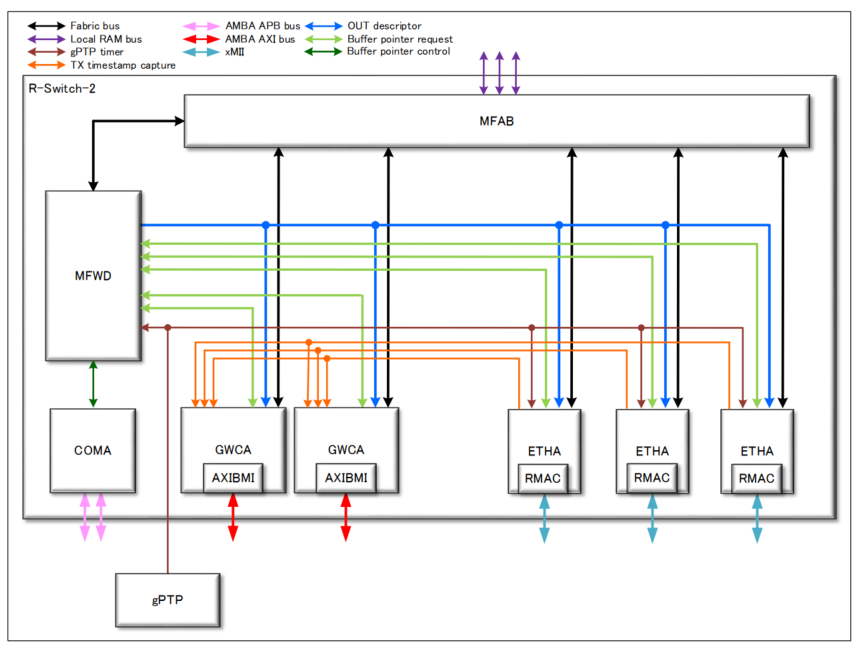
Description automatically generated

**ETHERNET RSWITCH**

1. **Ethernet RSwich2**
   1. **Overview**

The Ethernet Switch system consists on an Ethernet switch with higher level routing capability and multi-protocol interface support. It allows autonomous frame routing within same and between different network interfaces protocols (for now only Ethernet) for optimized gateway applications

* 1. **Block diagram**



* 1. **Modes**
* Hub
* Layer 2 switch
* VLAN aware layer 2 switch
* Layer 3 switch
  1. **Sub-module**
     1. **MFWD – Forwarding Engine**

Forwarding engine is part of R-Switch system and aims at filtering, forwarding and routing the switch frames. It snoops the frame information from fabric to redistribute them to the agent after filtering, forwarding and routing mechanisms.

* + - 1. **General functionalities**
* Exceptional path is used to forward faulty frames to CPU for diagnosis.
* Learning path is used to forward frames containing an unknown field to CPU for learning.
* CPU Mirroring path is used to copy frames to a CPU for monitoring purpose.
* Ethernet Mirroring path is used to copy frames to an ethernet agent for monitoring purpose.
* Mirroring path is used to copy frames to an Ethernet port or a CPU for monitoring purpose.
  + - 1. **Forwarding/Routing**
* Direct descriptor forwarding
* L3 forwarding/routing
* L2 forwarding
* Port based forwarding
  + - 1. **Filtering**

Forwarding engine allows frame filtering and shaping using the following functions:

* PSFP (Per Stream Filtering and Policing) [802.1Qci]
* ATS (Asynchronous Traffic Shaping) [802.1Qcr]
* FRER (Frame Replication and Elimination for Reliability) [802.1CB]
  + 1. **MFAB – Fabric**

Exchange data between the agents, local RAM and FWD

* + 1. **ETHA – Ethernet TSN Agent**

Exchange data between R-Switch and Ethernet PHY

Operation Mode

Diagram

Description automatically generated

Function:

* Data transmission

Diagram

Description automatically generated

+ Descriptor store: storing the descriptor received from FWD

+ L2/L3 update: fetching the L2/L3 update rules from the Forwarding Engine to update the frame while sending it

+ Frame time: This block aims at calculating how many ns a frame would take to be sent on the ethernet PHY for TAS arbitration

+ Frame size check: This block aims at releasing the pointer of the frames rejected by Frame size control block

+ CBS: shaping the data traffic per descriptor queue by controlling the data throughput per queue

+ TAS: shaping the data traffic per descriptor queue by controlling the data transmission with a schedule

+ Descriptor arbitration: Arbitrate between descriptors based on their descriptor queue using strict priority and/or WRR arbitration

+ TX data fetch: This block fetches the frame data from the local RAM, update it depending on the information obtained by L2/L3 update module and on VLAN control information and release the frame pointers

+ RMAC: exchange data with Ethernet PHY

* Data reception:

Diagram

Description automatically generated

+ RMAC: exchange data with Ethernet PHY

+ RX data store: extracts TAG information for frames, applies VLAN tagging and save the frames in the local RAM

* + 1. **GWCA – Gateway CPU Agent**

Exchange data between the R-Switch and GWCPU subsystem

Operation Mode

Diagram

Description automatically generated

Function:

* Data transmission

Diagram

Description automatically generated

+ TX queue arbitration: arbitrates between TX queues when several queues are requesting for transmission

+ AXI master interface: handles the data/AXI descriptor exchange with the CPU

+ TX data store: extracts TAG information for frames, applies VLAN tagging and save the frames in the local RAM

* Data reception

Diagram

Description automatically generated

+ Descriptor store: store the descriptor received from the FWD

+ L2/L3 update: fetching the L2/L3 update rules from the FWD to update the frame while sending it

+ Multicast control: checking the frame size and security level and decide to forward or discard frames and at copying the descriptors to be able to send the data at several CPU sub-destinations

+ Descriptor reject: releasing the pointer of the frames rejected

+ RX data fetch: fetches frames data from the local RAM, update frames depending on the information obtained by L2/L3 update module, on VLAN control information and the R-TAG information, can remove frame FCSs, and release frame pointers

+ AXI master interface: handles the data/AXI descriptor exchange with the CPU

* + 1. **COMA – Common Agent**

Aims at gathering common functionalities for other agents.

It handles the APB to SFR bus conversion, the buffer pointer release for rejected frames and the pointer handling for Local RAM. SW involvement required for common agent configuration

Functions:

* Watermark function: allows to selectively reject frames in forwarding engine [FWD] before a switch overflow occurs
* IPV based watermark
* Global level-based watermark
* Per-port level-based watermark
* Pause function: allows to selectively pause frame injection by agents [GWCA] [RMAC]
* Global function
* Per-port Pause function
* Per port memory allocation function: It is possible to control the maximum and the minimum number of pointers that can be used per port thanks to CABPULCi.MNNPNi and CABPULCi.MXNPNi
* Shared memory
* Split memory
* Reduced memory
* Hybrid memory
  + 1. **gPTP – gPTP timer**

The gPTP timer enables accurate synchronization of the clock in the control system.

* 1. **Operation**

External PHY initialization -> Ethernet SERDES initialization -> Wait linkup of RJ45 side of external PHY -> Ethernet Switch

* + 1. **Switch initialization flow**

|  |  |
| --- | --- |
| Diagram  Description automatically generated | Diagram  Description automatically generated |
| Switch initialization flow | Switch clocks enable flow [COMA] |
| Diagram  Description automatically generated | Diagram  Description automatically generated |
| Buffer pool initialization flow [COMA] | Full setting flow [FWD] |

|  |  |
| --- | --- |
|  |  |
| Initialization flow [GWCA] | Initialization flow [TSNA] |

* + 1. **Switch reset flow.**

|  |  |
| --- | --- |
| Diagram  Description automatically generated |  |
| Switch reset flow | R-Switch reset flow |

|  |  |
| --- | --- |
|  |  |
| Mode transition flow [TSNA] | Mode transition flow [GWCA] |

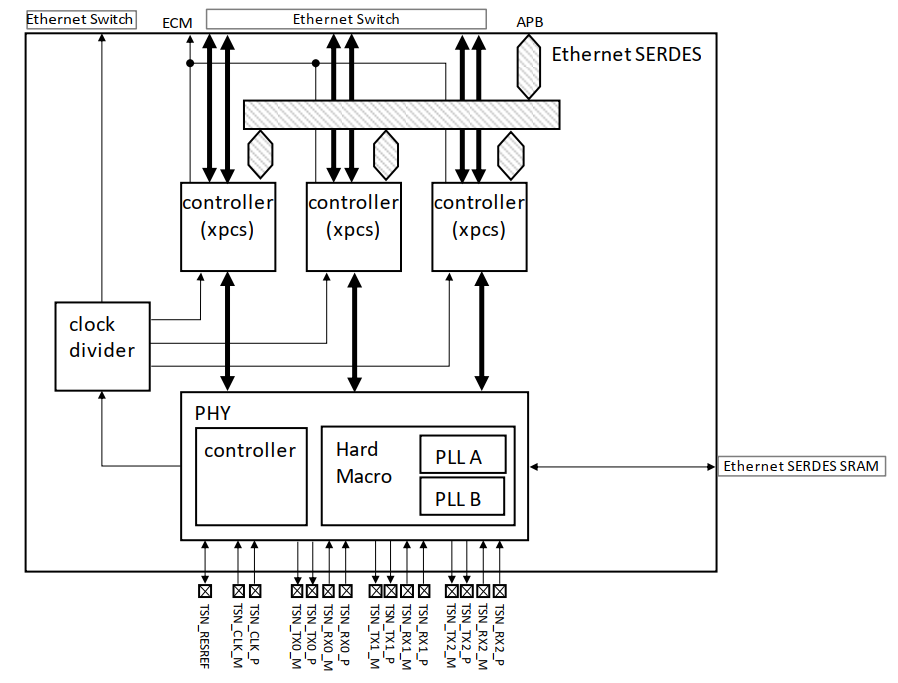
* + 1. **gPTP initialization flow**

Diagram

Description automatically generated

1. **ETHERNET SERDES**
   1. **Overview**

* SGMII (LinkSpeed 1GHz; DataRate 1Gbps, 100Mbps) / 5G USXGMII (LinkSpeed 5GHz; DataRate 1Gbps, 2.5Gbps) support
* Include 3 channel.
  1. **Block diagram**

****

* 1. **Operation Procedure**
     1. **Set common settings for USXGMII**
     2. **USXGMII setting for lane X for AN\_ON**
     3. **Set common settings for SGMII**
     4. **SGMII setting for lane X for AN\_ON**
     5. **SGMII setting for lane X for AN\_OFF**
     6. **Combination of USXGMII setting and SGMII setting**