RMAP ISSUES

Sandi Habinc

9th SpaceWire Working Group Meeting

26-27 April 2007

NoordwijK

OUTLINE

RMAP issues related to Draft F:

– VHDL Source Code in Annex A

– CRC octet for null size data field

– Error Code 12

Logotype?

VHDL Source Code Example

The VHDL source code example in Annex A has the following drawbacks:

- Imprecise definition on how to map the VHDL data structure to SpaceWire octet (byte):
  - Unclear what bits of the input and output vectors correspond to what bits of the SpaceWire octet (byte)
- Inconsistent mapping between input data and output CRC to SpaceWire octet (byte):
  - Input vector is defined (”next serial bit”), but the CRC output vector has actually the reverse bit order
- Cumbersome description using multiple procedures and custom data types
- Lacks comments on functionality

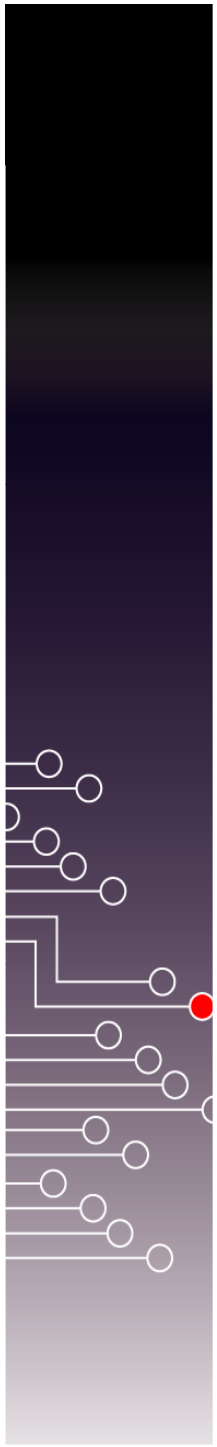
VHDL Source Code Example ...

Example proposed as a starting point:

```
------------------------------------------------------------------------
-- Cyclic Redundancy Code (CRC) for Remote Memory Access Protocol (RMAP)
------------------------------------------------------------------------
-- Polynomial: g(x) = g(x) = x**8 + x**2 + x**1 + x**0
------------------------------------------------------------------------
--         +---+   +---+   +---+   +---+   +---+   +---+    +---+    +---+
-- out <-+-| 7 |<--| 6 |<--| 5 |<--| 4 |<--| 3 |<--| 2 |<-X-| 1 |<-X-| 0 |<-+
--       | +---+   +---+   +---+   +---+   +---+   +---+  ^ +---+  ^ +---+  |
--       |                                                 |        |        |
--       v                                                 |        |        |
-- in -->X-------------------------------------------------+--------+--------+
--    x**8     x**7    x**6    x**5    x**4    x**3    x**2     x**1     x**0
------------------------------------------------------------------------
-- One-to-many implementation: Galois version of LFSR (reverse CRC)
------------------------------------------------------------------------
```

VHDL Source Code Example ...

```
-------------------------------------------------------------------------
-- Updates the next value of the SpaceWire RMAP CRC register.
--
-- Parameters:
-- CRC(7:0)    - The CRC byte value which is updated
-- INBYTE(7:0) - The next SpaceWire message data byte
--
-- Note: The CRC input must be all zeros for the first INBYTE of a message.
--
-- Note: SpaceWire data is sent/received Least Significant Bit (LSB) first.
-- INBYTE(0) corresponds to LSB of SpaceWire data byte (sent/received first)
-- CRC(0)    corresponds to LSB of SpaceWire data byte (sent/received first)
-------------------------------------------------------------------------
```



```
procedure RMAPCRC(
   variable CRC:     inout Std_Logic_Vector(7 downto 0);     -- bit 0 first
   variable INBYTE:  in    Std_Logic_Vector(7 downto 0)) is  -- bit 0 first

   -- internal Linear Feedback Shift Register (LFSR)
   variable LFSR:          Std_Logic_Vector(7 downto 0);

   -- Note: The vector indices of the LFSR variable correspond to the powers
   --       of the Galois field polynomial, which is not the same as the
   --       indices of the SpaceWire data byte.
begin
   -- external to internal bit-order reversal to match indices
   for i in 0 to 7 loop
      LFSR(7-i) := CRC(i);
   end loop;

   -- left-shift LFSR eight times, feed in INBYTE with bit 0 first (LSB)
   for j in 0 to 7 loop
      LFSR(7 downto 0) := (LFSR(6 downto 2)) &
                          (INBYTE(j) xor LFSR(7) xor LFSR(1)) &
                          (INBYTE(j) xor LFSR(7) xor LFSR(0)) &
                          (INBYTE(j) xor LFSR(7));
   end loop;

   -- internal to external bit-order reversal to match indices
   for i in 0 to 7 loop
      CRC(7-i) := LFSR(i);
   end loop;
end procedure RMAPCRC;
```

CRC octet for null size data fieldS

It has been recently clarified that a Data CRC byte should be present even if the corresponding Data Field has the size zero.

- Consistent with commands/responses that have Data Fields of varying sizes (0 to n).
- Inconsistent with commands/responses that have no Data Field at all:
  - Consider a Data Field without information not to be a Data Field, thus not requiring a Data CRC byte.
  - Data CRC byte is not part of Data Field

CRC octet for null size data fieldS ...

The null size Data Field CRC byte has an impact on the hardware implementation:

- Implementation differences in transmitter; sometimes it needs to send an all-zero CRC byte, sometimes not
- How to handle commands/responses without a CRC byte? => Early EOP?
- How to handle commands/responses with a non-all-zero CRC byte? => Data CRC Error?

Error Code 12

If an RMAP command with an *Invalid Destination Logical Address* is received (with a request for acknowledgement), the response shall be Error Code 12: *The header CRC was decoded correctly but the destination logical address was not the value expected by the destination.*

The SpaceWire standard states: *10.6.4.3 Node*

If a packet arrives at a node with an unexpected destination address then that packet shall be discarded.

NOTE 1 Whether a particular address is expected or not within a node depends upon the host system and its use of virtual channels.

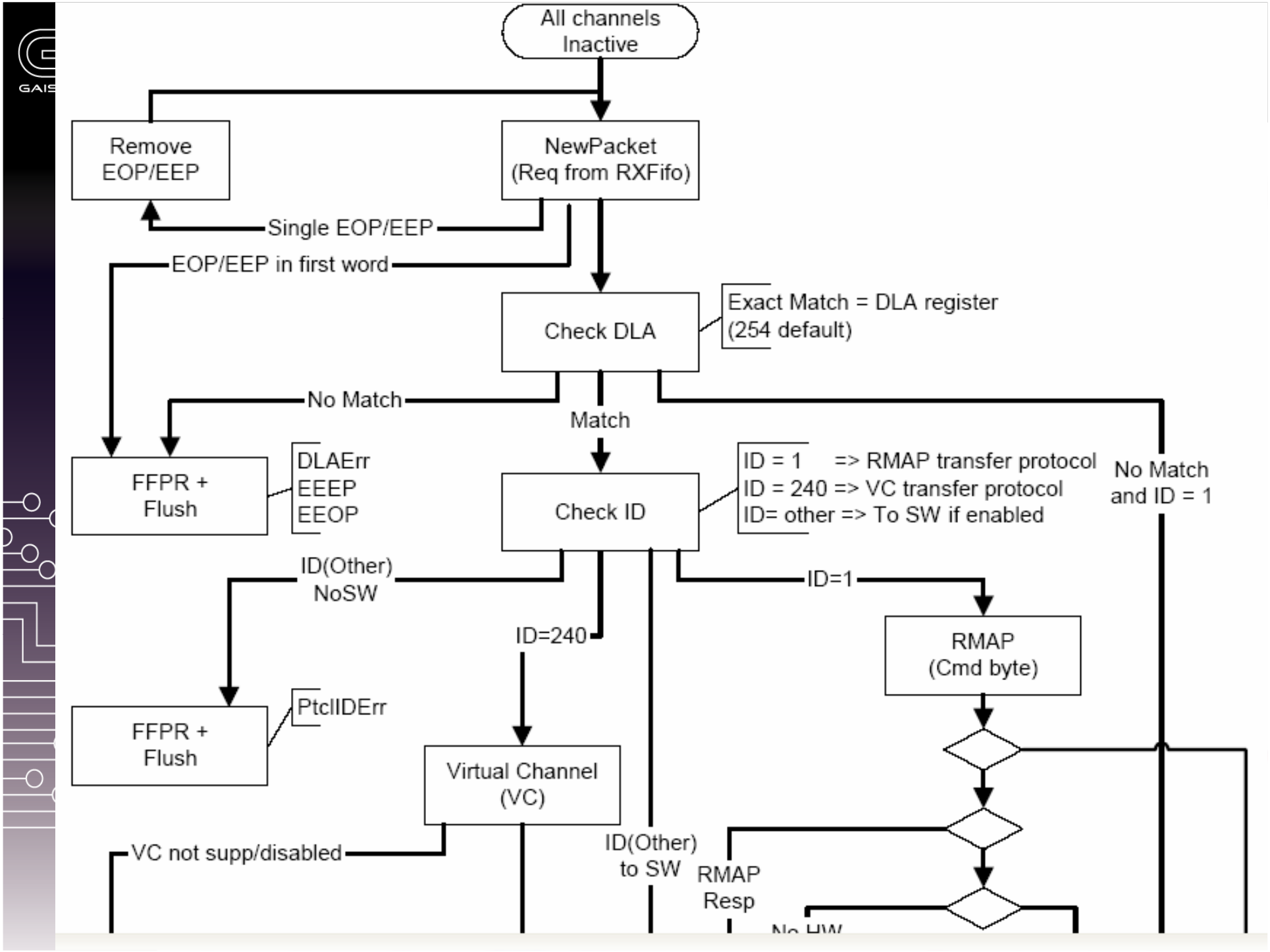
NOTE 2 The unexpected destination address may be flagged to the host system.

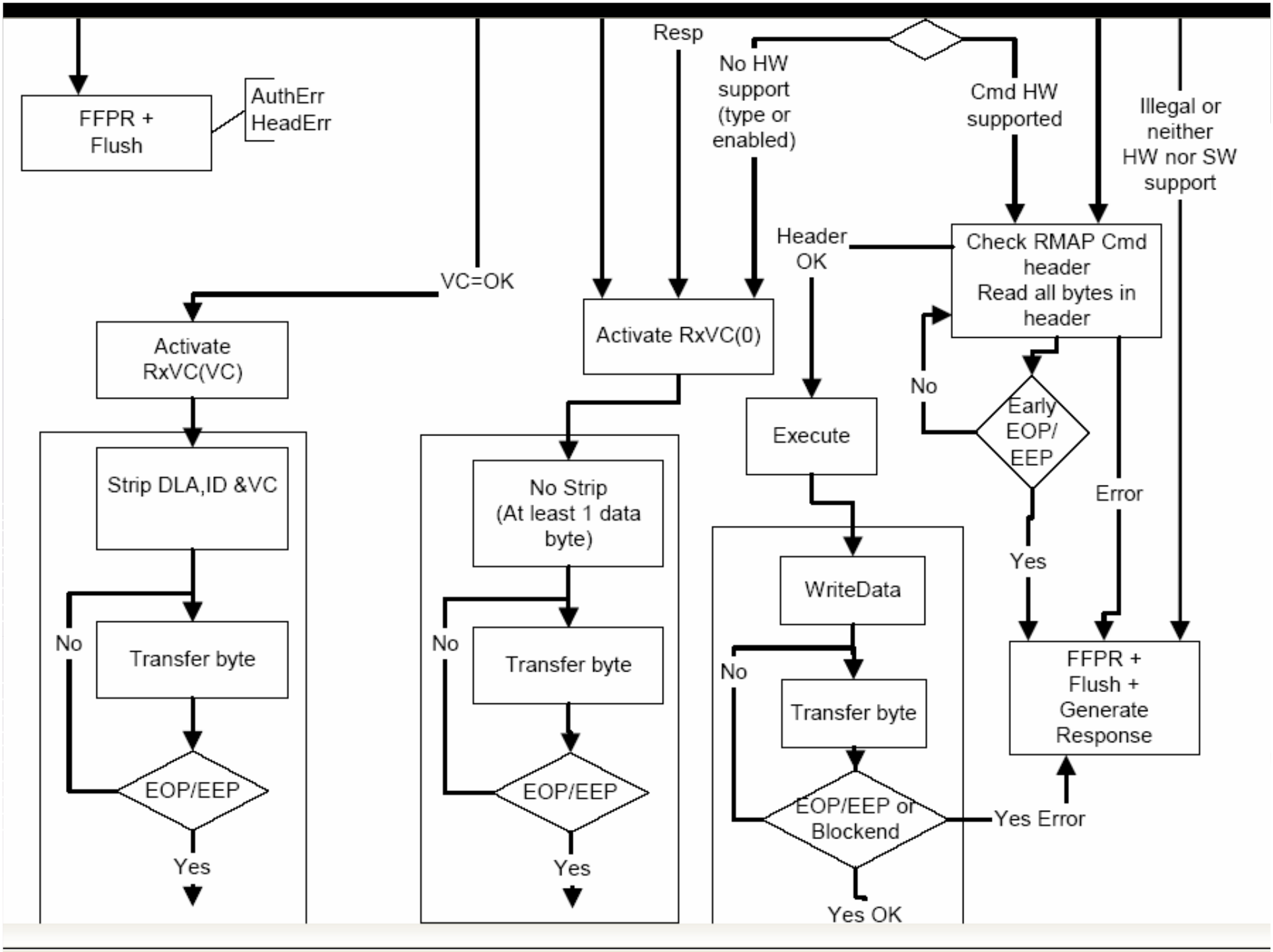
Error Code 12 ...

The SpaceWire standard is the underlying protocol upon which the RMAP standard is built, i.e. SpaceWire standard implements the layer under the RMAP protocol.

- In the lower SpaceWire layer, packets received with unexpected destination addresses shall be discarded.
- A discarded packet should thus not be sent to the higher layer RMAP protocol.

This potential violation of the SpaceWire standard has great implications on hardware complexity, since the underlying SpaceWire receiver needs to take care of a special case for RMAP packets.


All channels Inactive
Remove EOP/EEP
NewPacket (Req from RXFifo)
Single EOP/EEP
EOP/EEP in first word
Check DLA
Exact Match = DLA register (254 default)
No Match
Match
FFPR + Flush
DLAErr
EEEP
EEOP
Check ID
ID = 1 => RMAP transfer protocol
ID = 240 => VC transfer protocol
ID= other => To SW if enabled
No Match and ID = 1
ID(Other) NoSW
ID=240
ID=1
RMAP (Cmd byte)
FFPR + Flush
PtclIDErr
Virtual Channel (VC)
VC not supp/disabled
ID(Other) to SW
RMAP Resp


FFPR +
Flush
AuthErr
HeadErr
VC=OK
Activate
RxVC(VC)
Strip DLA,ID &VC
No
Transfer byte
EOP/EEP
Yes
Resp
Activate RxVC(0)
No Strip
(At least 1 data
byte)
No
Transfer byte
EOP/EEP
Yes
No HW
support
(type or
enabled)
Header
OK
Execute
WriteData
No
Transfer byte
EOP/EEP or
Blockend
Yes Error
Yes OK
Cmd HW
supported
Check RMAP Cmd
header
Read all bytes in
header
No
Early
EOP/
EEP
Yes
Error
Illegal or
neither
HW nor SW
support
FFPR +
Flush +
Generate
Response

Error Code 12 ...

Considering the hardware cost, is it really worthwhile to implement Error Code 12?

During earlier presentations it has been implied that a node could have multiple Destination Logical Addresses.

Which Destination Logical Address should be sent back as part of the reply (Error Code 12) when the received address does not match none of them?

LOGOTYPE

Is there (a need for) a logotype for SpaceWire compatibility or compliance?

For example, Sparc International that governs the SPARC RISC Architecture has several logotypes that can be used by members and compliant processors: