64b/66b PCS

Rick Walker Richard Dugan Birdy Amrutur John Ewen Rich Taborek Don Alderrou

Agilent Agilent Agilent IBM nSerial nSerial

May 23, 2000

Howard FrazierCiscoPaul BottorffNortelBrad BoothIntelKevin DainesWorld Wide PacketsOsamu IshidaNTT

IEEE 802.3ae Task Force

64b/66b Coding Update

Topics

- Code review and update
- Test vectors
- Bit ordering sequence
- Frame sync algorithm and state machine
- TX,RX error detection state machines
- Optional code features
- Summary



64b/66b Coding Update

Building frames from 10GbE RS symbols



Code Overview

Data Codewords have "01" sync preamble

0 1

Ottawa, ON

64 bit data field (scrambled)

Mixed Data/Control frames are identified with a "10" sync preamble. Both the coded 56-bit payload and TYPE field are scrambled

1 0 8-bit TYPE combined 56 bit data/control field (scrambled)

00,11 preambles are considered code errors and cause the packet to be invalidated by forcing an error (E) symbol on coder output



Code Summary

Input Data	Sy	nc		Bit fields 0-63						
$\mathbf{D}_0\mathbf{D}_1\mathbf{D}_2\mathbf{D}_3/\mathbf{D}_4\mathbf{D}_5\mathbf{D}_6\mathbf{D}_7$	0	1	D ₀	D ₁	D ₂	D ₃	D ₄	D ₅	D ₆	D ₇
$\mathbf{z}_0 \mathbf{z}_1 \mathbf{z}_2 \mathbf{z}_3 / \mathbf{z}_4 \mathbf{z}_5 \mathbf{z}_6 \mathbf{z}_7$	1	0	0x1e	z ₀	z ₁	z ₂	Z ₃	z ₄ z	5 Z 6	Z ₇
$\mathbf{z}_0 \mathbf{z}_1 \mathbf{z}_2 \mathbf{z}_3 / \mathbf{s}_4 \mathbf{D}_5 \mathbf{D}_6 \mathbf{D}_7$	1	0	0x33	z ₀	z ₁	z ₂	z 3	D ₅	D ₆	D ₇
$s_0 D_1 D_2 D_3 / D_4 D_5 D_6 D_7$	1	0	0x78	D ₁	D ₂	D ₃	D ₄	D ₅	D ₆	D ₇
$\mathbf{T}_0\mathbf{Z}_1\mathbf{Z}_2\mathbf{Z}_3/\mathbf{Z}_4\mathbf{Z}_5\mathbf{Z}_6\mathbf{Z}_7$	1	0	0x87		z ₁	z ₂	z ₃ 2	z ₄ z	5 Z 6	Z ₇
$\mathbf{D}_0\mathbf{T}_1\mathbf{Z}_2\mathbf{Z}_3/\mathbf{Z}_4\mathbf{Z}_5\mathbf{Z}_6\mathbf{Z}_7$	1	0	0x99	D ₀		z ₂	z ₃ 2	z ₄ z	5 Z 6	Z ₇
$\mathbf{D}_0 \mathbf{D}_1 \mathbf{T}_2 \mathbf{Z}_3 / \mathbf{Z}_4 \mathbf{Z}_5 \mathbf{Z}_6 \mathbf{Z}_7$	1	0	0xaa	D ₀	D ₁		z ₃ 2	z ₄ z	5 Z 6	Z ₇
$\mathbf{D}_0 \mathbf{D}_1 \mathbf{D}_2 \mathbf{T}_3 / \mathbf{Z}_4 \mathbf{Z}_5 \mathbf{Z}_6 \mathbf{Z}_7$	1	0	0xb4	D ₀	D ₁	D ₂		z ₄ z	5 Z 6	Z ₇
$\mathbf{D}_0 \mathbf{D}_1 \mathbf{D}_2 \mathbf{D}_3 / \mathbf{T}_4 \mathbf{Z}_5 \mathbf{Z}_6 \mathbf{Z}_7$	1	0	0xcc	D ₀	D ₁	D ₂	D ₃	Z	₅ Z ₆	Z ₇
$\mathbf{D}_0 \mathbf{D}_1 \mathbf{D}_2 \mathbf{D}_3 / \mathbf{D}_4 \mathbf{T}_5 \mathbf{Z}_6 \mathbf{Z}_7$	1	0	0xd2	D ₀	D ₁	D ₂	D ₃	D ₄	Z ₆	Z ₇
$\mathbf{D}_0 \mathbf{D}_1 \mathbf{D}_2 \mathbf{D}_3 / \mathbf{D}_4 \mathbf{D}_5 \mathbf{T}_6 \mathbf{Z}_7$	1	0	0xe1	D ₀	D ₁	D ₂	D ₃	D ₄	D ₅	Z ₇
$\mathbf{D}_0 \mathbf{D}_1 \mathbf{D}_2 \mathbf{D}_3 / \mathbf{D}_4 \mathbf{D}_5 \mathbf{D}_6 \mathbf{T}_7$	1	0	0xff	D ₀	D ₁	D ₂	D ₃	D ₄	D ₅	D ₆



IEEE 802.3ae Task Force

64b/66b Coding Update

7-bit control code mapping

rs value	name	shorthand	7-bit line code
0x07,1	idle	[1]	0x00
0xfb,1	start	[S]	encoded by TYPE byte
0xfd,1	terminate	[T]	encoded by TYPE byte
0xfe,1	error	[E]	0x1e
0x1c,1	reserved0	-	0x2d
0x3c,1	reserved1	-	0x33
0x7c,1	reserved2	-	0x4b
0xbc,1	reserved3	-	0x55
0xdc,1	reserved4	-	0x66
0xf7,1	reserved5	-	0x78



Ottawa, ON

Bit ordering sequence



Scrambler definition

Serial form of the Scrambler:

May 23, 2000

Ottawa, ON



The serial form of the scrambler is shown here for bit ordering purposes. Parallel implementations could also be used. For details see:

http://grouper.ieee.org/groups/802/3/ae/public/mar00/walker_1_0300.pdf



Sample 64b/66b Test Vector

Start with a minimum length (64 byte) Ethernet packet with CRC
 08 00 20 77 05 38 0e 8b 00 00 00 08 00 45 00 00 28 1c 66 00 00 1b 06 9e d7 00 00 59 4d 00 00
 68 d1 39 28 4a eb 00 00 30 77 00 00 7a 0c 50 12 1e d2 62 84 00 00 00 00 00 00 00 93 eb f7 79

Add SOP, EOP, Idles and convert to RS indications

 07,1
 07,1
 07,1
 07,1
 07,1
 07,1
 07,1
 1
 1
 1
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0</td

Arrange bytes into frames with type indicators and sync bits

May 23, 2000

 "10"
 1e
 00
 00
 00
 00
 00
 "10"
 78
 08
 00
 20
 77
 05
 38
 0e

 "01"
 8b
 00
 00
 00
 00
 45
 "01"
 00
 00
 20
 77
 05
 38
 0e

 "01"
 8b
 00
 00
 00
 45
 "01"
 00
 02
 1c
 66
 00
 00
 1b

 "01"
 06
 9e
 d7
 00
 00
 59
 4d
 00
 "01"
 00
 68
 d1
 39
 28
 4a
 eb
 00

 "01"
 00
 30
 77
 00
 00
 7a
 0c
 50
 "01"
 12
 1e
 d2
 62
 84
 00
 00
 00

 "01"
 00
 00
 00
 00
 93
 eb
 f7
 "10"
 99
 79
 00
 00
 00
 00
 00

Scramble and transmit left-to-right, lsb first, (scrambler initial state is set to all ones)

"10" 1e 00 00 00 80 f0 ff 7b "10" 78 48 f8 df 88 c6 f3 13 "01" c5 a5 9c da 93 25 1d 1f "01" 7a bb d0 dd ff c9 21 1b "01" 05 23 44 6f aa 5c 30 b6 "01" bb 2d 42 88 81 56 25 cf "01" 72 78 fd 94 3d d6 8c e5 "01" 39 f5 ea 52 2e af ec 92 "01" fd 3c 1c cf 75 5f be 8f "10" 8d b3 13 08 ba 3f e7 3f

> IEEE 802.3ae Task Force

Frame alignment algorithm

Look for presence of "01" or "10" sync patterns every 66 bits

This can be done either in parallel, by looking at all possible locations, or in serial by looking at only one potential location.

In either case, a frame sync detector is used to statistically qualify a valid sync alignment.

In the parallel case, a barrel shifter can immediately make the phase shift adjustment. In the serial case, a sync error is used to cycle-slip the demultiplexor to hunt for a valid sync phase.

So what algorithm should be used for reliable and rapid frame sync detection?



Frame sync criteria

If misaligned, then sync error rate will be 50%. We must quickly assert loss of sync and "slip" our alignment to another candidate location

If already aligned with good BER (<10e-9), then we want to stay in sync with very high reliability

If BER is worse than 10e-4 we should suppress sync, to avoid likelyhood of False Packet Acceptance due to CRC failures

BER	current sync state	next sync state	notes
~50%	in	out	should be fast
>10e-4	in	out	prevents MTTFPA events, can be relatively slow to trigger
<10e-9	out	in	should be fast



Frame sync algorithm

- frame sync is acquired after 64 contiguous frames have been received with valid "01" or "10" sync headers
- frame sync is declared lost after 32 "11" or "00" sync patterns have been declared in any block of 64 frames
- In addition, if there are 16 or more errors within any 125us time interval (~10e-4 BER), then frame sync is inhibited



64/66 frame sync performance



Synchronization state machine



IEEE 802.3ae Task Force

May 23, 2000

64b/66b Coding Update

Packet boundary protection

 A 2 bit error in the sync preamble can convert a packet boundary (S,T) into a Data frame (D) and vice-versa. However, all such errors violate frame sequencing rules unless another 4 errors recreate a false S,T packet (a total of six errors). Frame sequence errors invalidate the packet by forcing an (E) on the coder output.



TX state machine





RX state machine



IEEE 802.3ae Task Force

May 23, 2000

64b/66b Coding Update

Optional Code Features

- Special frames are reserved to support ordered sets for both Fiber Channel and 10GbE Link Signalling Sublayer (LSS)
- x,y ordered-set IDs are "1111" for FC and "0000" for 10GbE LSS

XGMII Pattern	Sync		Bit fields 0-63									
ZZZZ/ODDD	1	0	0x2d	Z0	Z1	Z2	Z	3	У	D5	D6	D7
ODDD/ZZZZ	1	0	0x4b	D1	D2	D3	,	х	Z4	l Z5	Z6	Z7
ODDD/ODDD	1	0	0x55	D1	D2	D3		х	У	D5	D6	D7
ODDD/SDDD	1	0	0x66	D1	D2	D3		х	У	D5	D6	D7
SDDD/DDDD	1	0	0x78	D1	D2	D3		D	4	D5	D6	D7
undefined	1	0	0x00		reser	rved f	or	fu	itur	e expa	ansion	

rs value	name	shorthand	7-bit line code
0x5c,1	FC ordered-set	[Of]	encoded by TYPE byte
0x9c,1	10 GbE Link Signalling	[LS]	encoded by TYPE byte



Summary

- We've shown a simple and reliable algorithm for 64b/66b frame sync detection
- Bit ordering has been clarified to be compatible with Ethernet CRC definition
- The TX and RX error control state machines have been presented
- A simple test vector has been produced to help to verify new implementations
- Optional 64b/66b extensions exist to support FC ordered sets and LS signalling



Supplementary slides

IEEE 802.3ae Task Force

May 23, 2000

64b/66b Coding Update

State machine notation conventions

Variables

TXD<35:0>TXD signal of GMII
RXD<35:0>RXD signal of GMII
tx_tobe_coded<71:0>
tx_tobe_xmitted<65:0>A 66 bit vector which is the result of a PCS ENCODE operation and is to be transmitted to the PMA.
rx_tobe_decoded<65:0>A 66 bit vector containing the most recently received code word from the PMA.
rx_decoded<71:0>72 bit vector which is the result of the PCS DECODE operation on the received bit vector, rx_tobe_decoded
rx_to_gmii<71:0>72 bit vector which is a pipelined delayed copy of rx_decoded. This is sent to GMII in two steps of 36 bits each. Bits 71:36 are sent first to RXD, followed by bits 35:0.
rx_err<71:0>This holds either a pipeline delayed copy of rx_decoded or the error frame EFRAME_G
stateHolds the current state of the transmit or the receive process.
sync_doneBoolean variable is set true when receiver is synchronized and set to false when receiver looses frame lock.
mt_validboolean variable is set true if received frame rx_tobe_decoded has valid frame prefix bits. I.e, mt_valid = rx_tobe_decoded[65] ^ rx_tobe_decoded[64]
mt_valid_cntHolds the number of frames within a window of 64 frames, with valid prefix bits
mt_invalid_cntHolds the number of frames within a window of 64 frames with invalid prefix bits
hi_ber_cntHolds the number of with invalid prefix bits, within a 125us period
hi_berBoolean is asserted true when the hi_ber_cnt exceeds 16 indicating a bit error rate >=10-4



State machine notation conventions

Constants

	[Z, S, T, D]Each 72 bit vector, tx_tobe_coded and the 66 bit vector, rx_tobe_decoded, can be classified to belong to one of the four types depending on its contents. The frame types Z,S, T, D are defined in TBD.
EFRAME_G<71:0>	72 bit vector to be sent to the GMII interface and represents a error octet in all the eight octet locations
EFRAME_P<65:0>	.66 bit vector to be sent to the PMA and represents a error octet in all the eight octet locations,

Functions

ENCODE(tx_tobe_coded<71:0>)Encodes the 72 bit vector into a 66 bit vector to be transmitted to the PMA DECODE(rx_tobe_decoded<65:0>)Decodes the 66 bit vector into a 72 bit vector to be sent to the GMII TYPE(tx_tobe_coded<71:0>) TYPE(rx_tobe_decoded<65:0>).....Decodes the FRAME_TYPE of the tx_tobe_coded<71:0> bit vector or the rx_tobe_decoded<65:0>

Timers

64frames_timer_doneTimer which is triggered once every 64 of the 66-bit frames in the receive process

125us_timer_done......Timer which is triggered once every 125us (is approximately 2¹⁴ 66-bit frames in the receive process).



64b/66b Coding Update

May 23, 2000