MCC-I2.1 Specifications

MCC Design Group

Data Format

Revision:	1.0
Reference:	ATLAS ID-xx
Created:	10 December 2003
Last modified:	15 September 2004 11:13
Edited By:	R. Beccherle and G. Darbo
	ATLAS/Pixel Collaboration

Abstract

This document describes the Output Data Format of the MCC-I2.1.

Table Of Contents

List of Figures

List of Tables

6 Data Format

This chapter deals with all data formats provided by the *MCC*.

The *MCC* has to provide communication with 16 *FE* chips hosted on a module and with the ROD both for configuration and Data Taking. Both the communications between *MCC* and the *FE*'s and the *ROD* use a very simple serial command protocol that is explained in the following chapters.

The first two sections deal with the configuration protocol of the *MCC* and the *FE* chips, Section 6.3 explains the event format between the *FE* chips and the *MCC* while Section 6.4 deals with the *MCC* to *ROD* event format. The last section describes the Physical Layer Protocol, i.e. how data is splitted on both DTO lines depending on the *MCC* Output Speed settings.

6.1 ROD to MCC Configuration Data Format

As described in the "Command Decoder" chapter all configuration data between the *ROD* and the *MCC* follow a simple serial command protocol. This protocol is subdivided in three major command classes, Trigger, Fast and Slow commands.

⚠ All *MCC* configuration commands are Slow commands and will take the *MCC* out of RunMode. If the *MCC* is not in Run-Mode no Fast and Trigger commands are accepted by the *MCC*.

GlobalResetMCC is the reset command that initializes all default values of the *MCC* chip and is the first command to be issued in order to start the configuration phase from a well known state of all intrnal registers. This command has no effect on the *FE* chips hosted on a module. In order to reset the *FE* chips one has to use the SYNC or the GlobalResetFE commnads as described in the "Command Decoder" chapter.

⚠ The reset of the *MCC* is synchronous and therefore this command only has an effect on the chip if the system clock is turned on.

The WrRegister command is used to initialize all internal registers to their correct state. This command does not produce any output at the *MCC* DTO-0 and DTO-1 pins.

There are four additional configuration commands that are provided mainly for testing purposes. These commands are RdRegister, WrFifo, RdFifo and WrReceiver.

The WrFifo command allows to completely write any of the 16 data FIFO's present in the *MCC*, while RdRegister and RdFifo allow to check the contents of any register or FIFO respectively. In order to access a specific FIFO one has to set accordingli the FEEN register. These two commands are the only *MCC* configuration commands that produce an output.

Figure 6-1 shows an example of a RdRegister command. As can be seen in response to a RdRegister (RdFifo) command, after a certain delay the *MCC* presents on its output the 5 bit header (11101) immediately followed by the 16 (26) bits of the contents of the read register (fifo location).

⚠ All output signals generated by the *MCC* in response to a configuration command are sent to the *ROD* at a 40 Mbit/s rate on both links, independent of the Output Data Speed settings of te *MCC*.

The last configuration command has to be EnDataTake which is used to set the *MCC* in RunMode.

Once in RunMode the *MCC* is able to decode Fast and Trigger commands that are used to perform Event building.

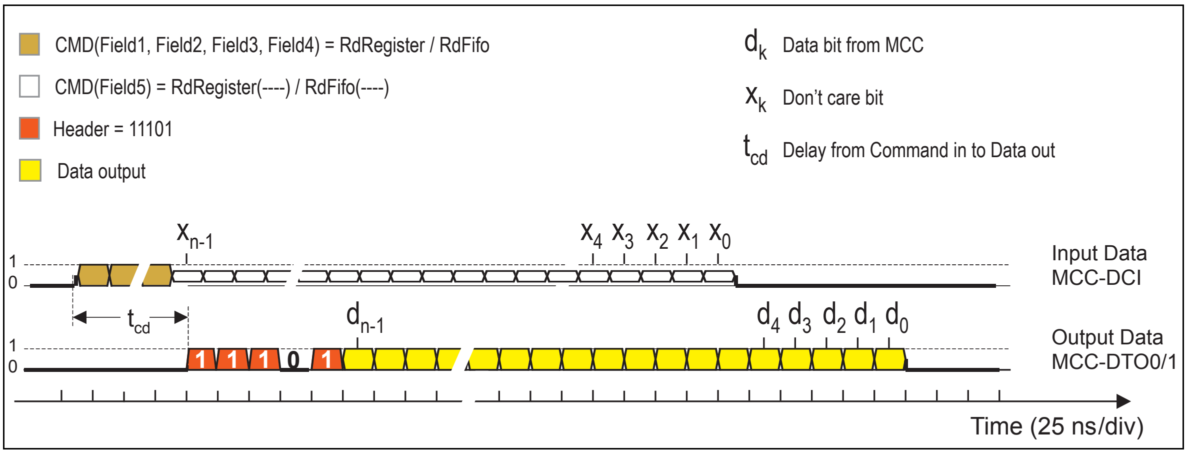

Figure 6-1 MCC to ROD data format in case of RdRegister and RdFifo commands.

For a complete reference on Trigger, Fast and Slow commands refer to Chapter 1.2, "Command Set".

6.1.1 Example of MCC configuration

Lets now assume that one wants to configure a module equipped with 16 working, and configured, *FE* chips, selecting a 160 Mbit/s output data transfer without consecutive Triggers, setting the RECD bit.

The correct sequence to perform this operation is:

1. GlobalResetMCC: Resets the *MCC*.
2. WrRegister FEEN `0xffff`: Enables all 16 *FE* chips.
3. RdRegister FEEN: Checks that we wrote the correct value. This command returns the header followed by `0xffff` on both DTO-0 and DTO-1 lines.
4. WrRegister CSR `0x0007`: Sets the RECD bit (`0004`) and sets 160 Mbit/s output speed (`0003`).
5. RdRegister CSR: Checks that we wrote the correct value. This command returns the header followed by `0x0007` on both DTO-0 and DTO-1 lines.
6. EnDataTake: Sets the *MCC* in RunMode.

We do not need to set the LV1 register as by default the *MCC* sends just one LV1 signal to all *FE*'s in response to a Trigger command.

6.2 MCC to FE Configuration Data Format

The interconnect topology between the *MCC* and the 16 *FE* chips in a module is a star topology which uses unidirectional serial links.

The *MCC* to *FE* protocol used to write and read *FE* chip configuration data uses three dedicated CMOS lines that are shared between all 16 *FE* chips hosted on a module. These lines are MCC-DAO, MCC-CCK and MCC-LD.

The *FE* configuration commands are logically subdivided in three parts. The first part of the command is an *MCC* Slow command followed by a data part. This part is subdivided in a *FE* address plus control word and data word that are dealt with inside the *FE* chips.

To upload data to the *FE*'s, the *MCC* provides data (MCC-DAO) together with a clock to latch them (MCM-CCK) on the positive edge. The MCC-LD signal is used to distinguish, in the bit stream from the MCC-DAO pin, the address plus control words from the subsequent data words. The CCK is generated by the *MCC* only when serial data present at the DAO output must be latched in the *FE*. This clock has a lower frequency (5 MHz) than the XCK to reduce the timing requirements on the internal registers which can be loaded or read back using this protocol.

FE chips are selected for parameter read / write by serial geographical addressing.

Since some of the internal registers in the *FE* chip are organized as long chains of shift registers their readout will be destructive. Furthermore, no partial data loading is possible within each memory structure. These structures can be loaded with new data values while the old ones are shifted out and read back into the *MCC* through the FE-DO lines.

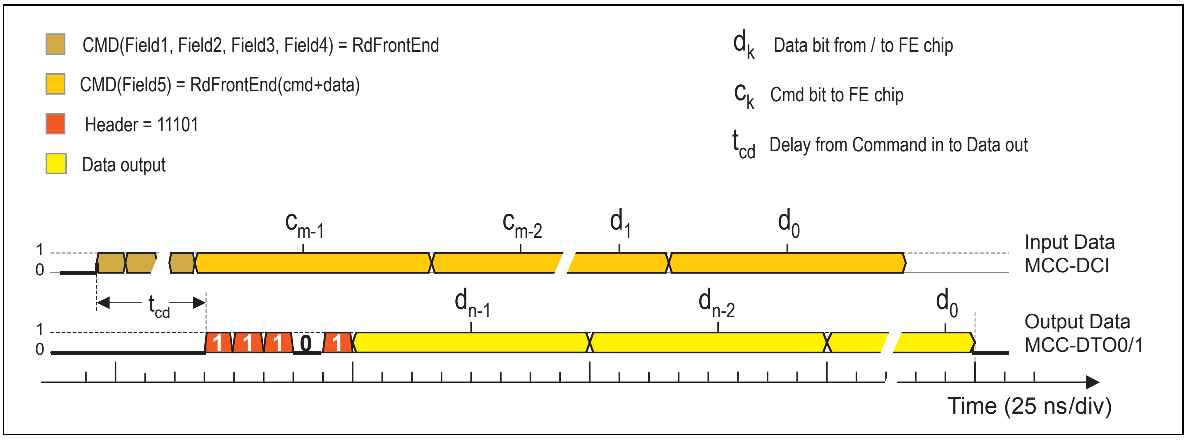

Figure 6-2 MCC to ROD data format in case of RdFrontEnd command.

Figure 6-2 shows an example of a RdFrontEnd configuration command in which an internal *FE* register is loaded. As can be seen, in response to a serial command, composed by the bits which identify the command to the *MCC* followed by the command and data to be written in the selected *FE* Register, the *MCC* responds with the 5-bit header (11101) followed by the contents of the *FE* Register with a 5 MHz clock.

Before transmitting the data train, the geographical address together with the command are shifted in (see Figure 6-3). Geographical address and command are transferred when LD is low. Then LD is raised and the transfer of the data starts. Data transmission is terminated by the falling edge of LD and the absence of CCK. CCK is generated by the *MCC* dividing its input clock (CK) by a factor 8 (i.e. CCK = 5 MHz). CCK is only generated when a bit carrying address, command or data information is present at MCC-DAO pin. The waveforms used by this protocol are shown in Figure 6-3.

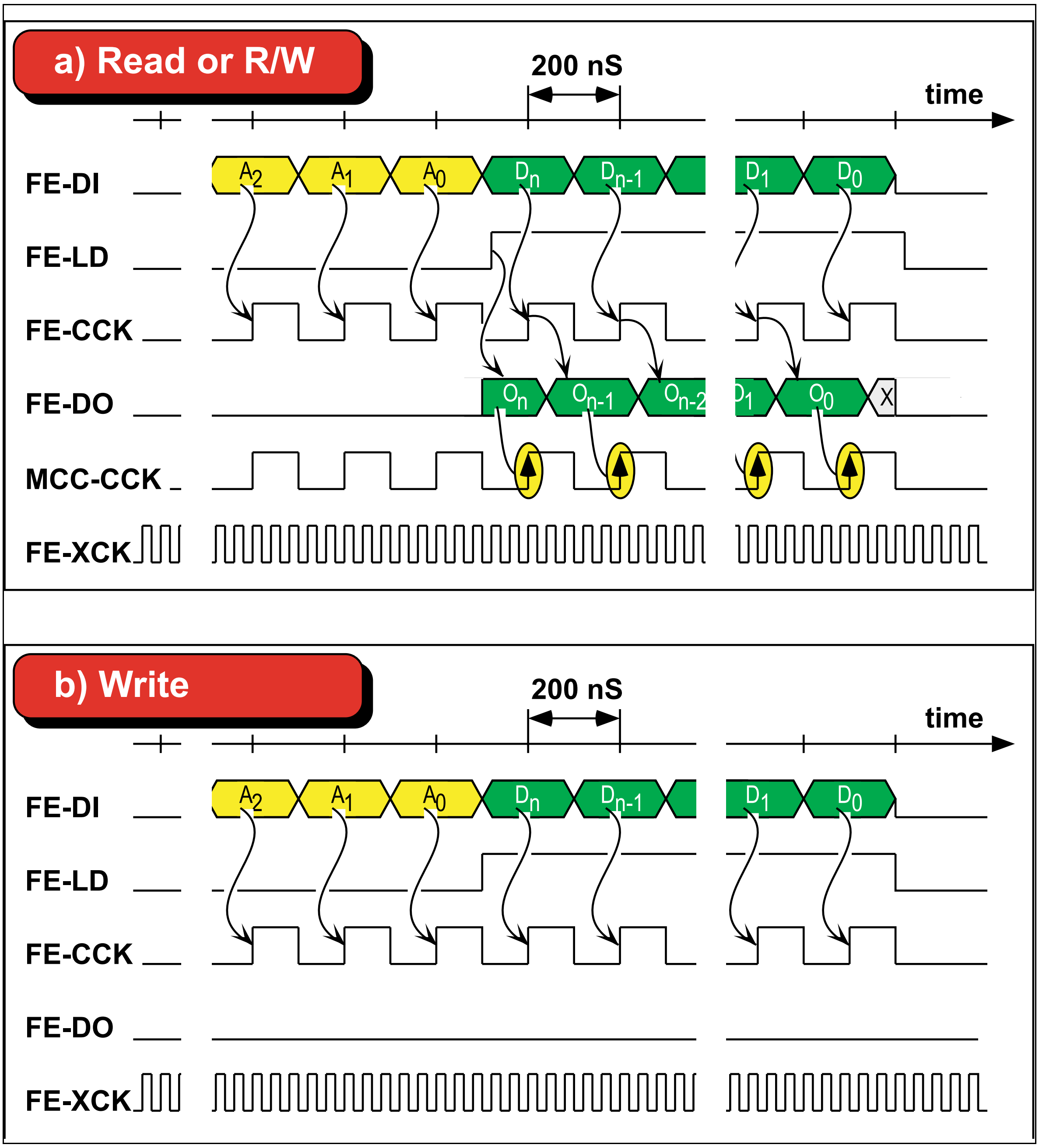

Figure 6-3 Protocol between MCC and FE chips to load initial configuration. Read/Write operations can be executed at the same time. In this case new values are loaded into the FE while present values are readout. Data bits are valid only on the low-to-high transition of FE-LD signal and only FE chips which recognize the transmitted address operate on data.

In this figure we show only the *FE* part of the serial command issued to the *MCC*.

Figure 6-4 shows the waveforms of the signals in response to a *FE* configuration command depending on the settings of the CNT register. CNT<15:3> sets the length of the Data part of the *FE* command in CCK units, while CNT<2:0> x 8 sets the length of the Command part of the *FE* command in CCK units. As LD is used to distinguish between Command and Data part we have to ensure that it is always generated. This figure shows how the LD signal is generated, in CCK units, in response to all possible combinations of Command and Data parts also when one of the two is zero.

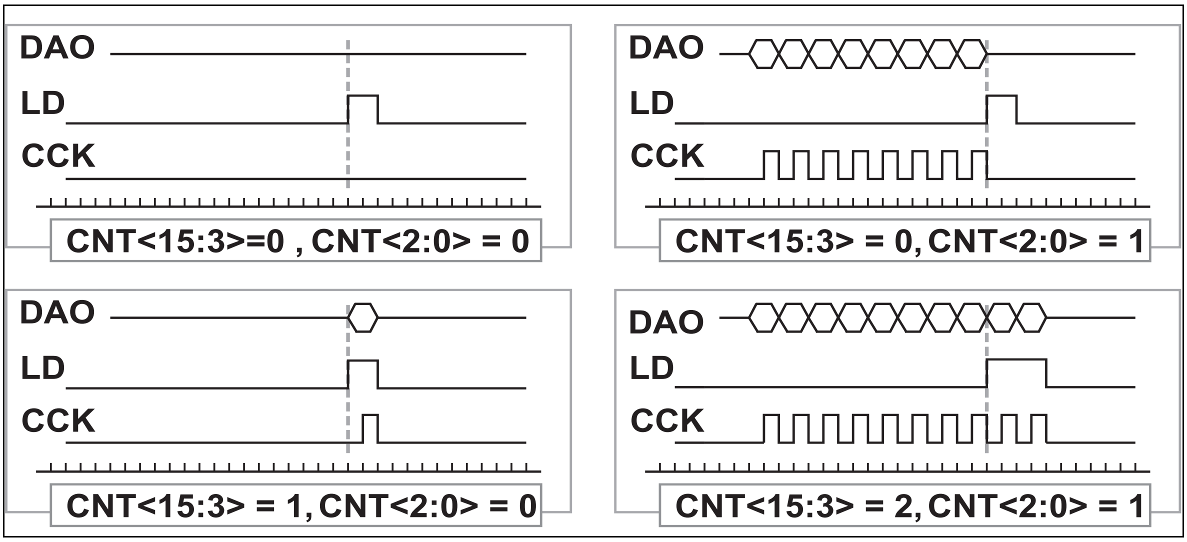
DAO
LD
CCK
CNT<15:3>=0 , CNT<2:0> = 0
DAO
LD
CCK
CNT<15:3> = 0, CNT<2:0> = 1
DAO
LD
CCK
CNT<15:3> = 1, CNT<2:0> = 0
DAO
LD
CCK
CNT<15:3> = 2, CNT<2:0> = 1

Figure 6-4 Waveforms of the signals to configure FE chips for different settings of CNT register.

6.3 FE to MCC Event Format

When the *MCC* is in RunMode, in response to a Trigger command, data is transmitted from the *FE* chips to the *MCC* via 16 dedicated LVDS lines (MCC-DTI<15:0>).

The *FE* chip encodes and transmits hits from the corresponding event using fixed length packets. Each hit contains a 1-bit header, followed by a 4-bit LV1, an 8-bit row number, a 5-bit column number, and an 8-bit ToT value. The encoding uses a non return to zero coding synchronous with the 40 MHz clock coming from the *MCC* (MCC-XCK). The *FE* encodes EoE (End-of-Event) and Warning messages. If there are no Hits only the Header and one EoE word (with or without Warning) are issued.
A Warning word is issued by the *FE* in case of hit overflow, parity error or LV1 inconsistency and is transmitted at the end of the event and replaces the EoE message, which is suppressed in such cases.

Table 6-1 summarizes the format for data hits and messages transmitted from *FE* chips to the *MCC*, while Figure 6-5 shows the transmission format in the time domain

Table 6-1 Packet format for event read-out used in transmission from FE to MCC.

	Header (Bin)	BCID# (Bin)	Row# (Bin)	Column# (Bin)	ToT
Data hit	**1**	0000:BBBB	0000 0000:1101 1111	0 0000:1 0111	0000 0000:1111 1111
EoE/WNG#n	**1**	bbbb	111**0** ffff	P LLLL	BBBB FFFF
EoE	**1**	bbbb	111**1** ffff	x xxxx	xxxx xxxx

Notes:

(1) x = don’t care (Not used by the *MCC* if it is an EoE word without Warning)

(2) P = Parity Error: will always produce a FE-Flag word in the *MCC*.

(3) F = FE-Flag[7:4] (4) f = FE-Flag[3:0]

(5) L = LV1ID[3:0]

(6) B = BCID[7:4] (7) b = BCID[3:0]

As can be seen the 4 least significant bits of the LV1ID are transmitted both in the Data and EoE words. EoE words containing Warning bits are detected by checking bits EoE[20:17]. If EoE[17] = 0 there is a Warning, and eventually the *MCC* will generate a FE-Flag warning word.

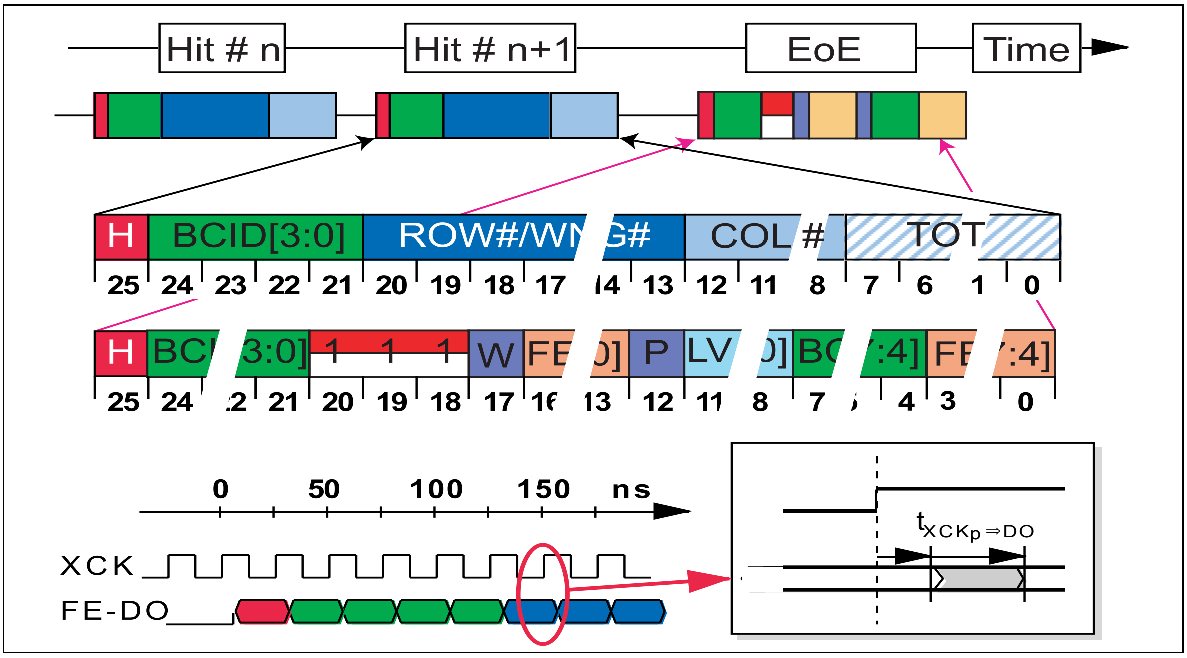

Figure 6-5 Data format and timing for event transmission from FE to MCC. (a) Hit frame is 26-bit long, (b) EoE frame is also 26-bit long.

Due to a misunderstanding in the specifications the Parity bit, EoE[12], is treated by the *MCC* as an Error and therefore if this bit is set the *MCC* will always generate a FE-Flag warning word. This behaviour unfortunately cannot be disabled and therefore one has to ensura that the *FE* chip will never genrate data on this bit of the EoE word.

6.4 MCC to ROD Event Format

Figure 6-6 is a graphic example of event encoding from the *MCC* output.
Example 1 of the figure shows an Event that contains two Hits belonging to the same *FE* chip, while the second one shows an Event that contains two Hits belonging to two different *FE* chips. The last example shows an Event containing one hit and a FE-Flag word (with both the MCC-Flag and the FE-Flag bytes).

The format of the event generated by the *MCC* has been defined considering that:

- Events are ordered by LV1 arrival time.

  Event building is done by grouping all hits belonging to the same LV1 number. The first event which has been triggered is the first to be sent out. Every event is entirely transmitted before the next event is considered for transmission: no event interleaving is allowed.

- Data are sorted and grouped by *FE* chips.
  The event builder sorts pixel hits by *FE* chip order. This permits data compression by “clustering” for either the case of a non-uniform hit distribution (jets) or a large pixel occupancy (B-layer) since every hit does not need to extend the address information to include the *FE* chip number with its row and column address.

- Event length is not known until the whole event is transmitted.
  The event builder does not know until the event is completely built up the total number of hits, or the number of hits per *FE* chip. This makes the event builder simpler, but forbids the usage of forward word counters in the event frame. Event data fields must be recognized by their content.

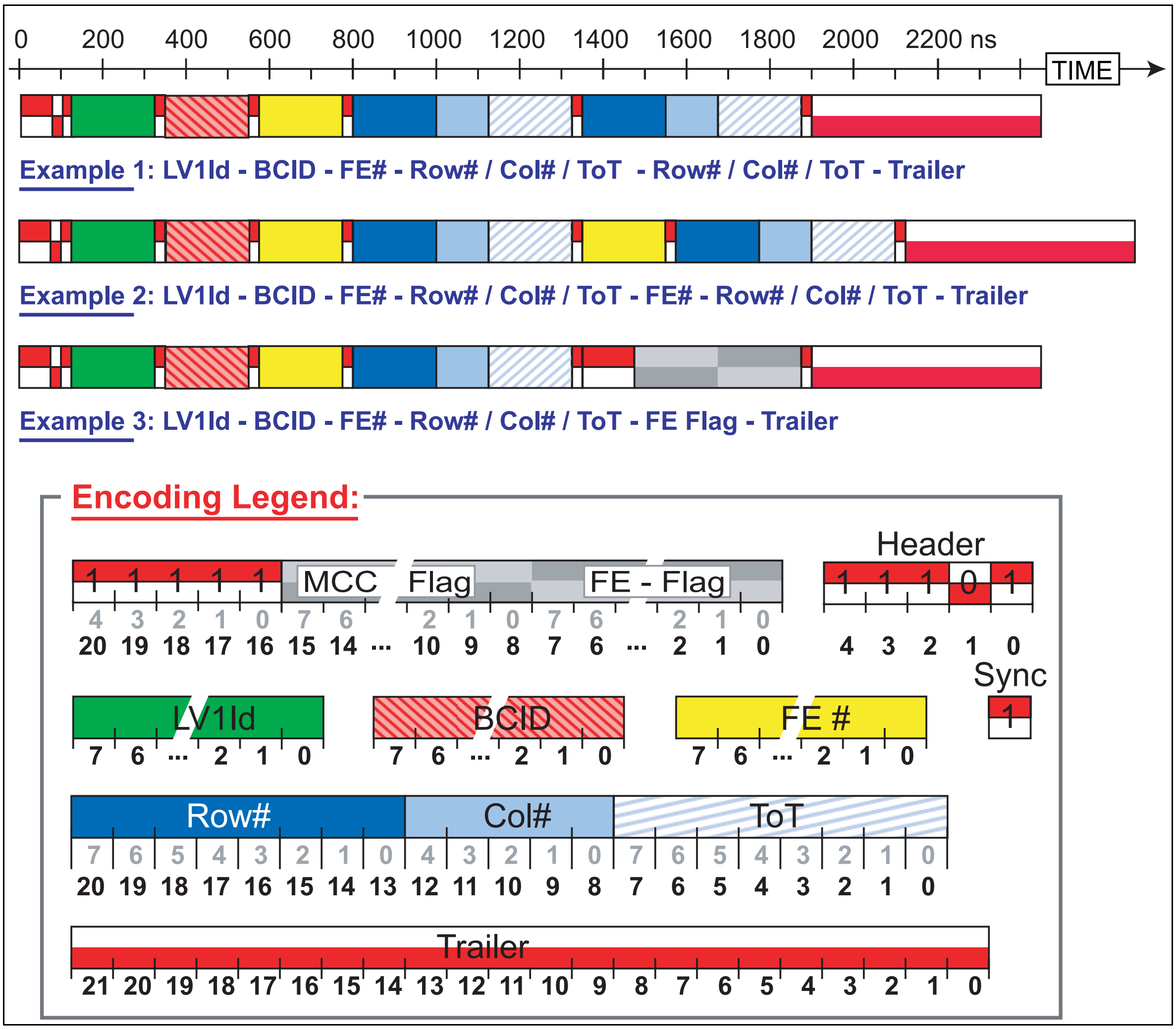

Figure 6-6 Event Data format at the MCC output. Example 3 illustrates the case of an event with a FE-Flag.

- Data must be compressed due to bandwidth limit.
  Since we need to keep the number of transmission cables as small as possible, and since a rather large amount of data is transmitted from the *MCC*, event coding which compresses data is specially important.
- Recovery from transmission errors.
  Any transmission error must be recovered by the next event data. The chosen mechanism is to use a Trailer whose value can never occur in the data. Since data fields are allowed to take any value we decided to add a Sync bit (bit set to ‘1’) after every data field and a Trailer containing 22 zeroes.

An event frame at the output of the *MCC* is an ordered stream of data fields separated by synchronization bits.

The syntax for the event data structure is as follows:

```
<Event>
    ::= <Header> L1ID <Sync> BCID <MccFlag>? <FrontEnd>* <Trailer>
<Header>
    ::= 11101
<Sync>
    ::= <1>
<FrontEnd>
    ::= <Sync> MCC-FE# <Hit>+ <FeFlag>?
    ||= <Sync> MCC-FE# <Hit>* <FeFlag>
<Hit>
    ::= <Sync> ROW# COL# TOT
<FeFlag>
    ::= <Sync> FE-FLAG
<Trailer>
    ::= <Sync> 00 0000 0000 0000 0000 0000
```

The following items summarize the format of the formal syntax description:

<Name> name in lower case is a syntax construct defined by other syntax constructs or by a bit field.

NAME in upper case is a bit field. It's definition is stated in Table 6-2.

<Name>? is an optional field, 0 or 1 items occurrence.

<Name>* is 0, 1 or more items.

<Name>+ is 1 or more items.

::= gives a syntax definition to an item.

||= introduces an alternative syntax definition.

The values of the keywords which appear in the event syntax are in Table 6-2.

Table 6-2 Value for the keywords in the event syntax used by the MCC.

Keyword	Low Value	High Value	Description
BCID	0000 0000	1111 1111	Bunch Crossing ID (0÷255)
COL#	0 0000	1 0111	Column number
FE-FLAG	1 1111 MMMMMMMM FFFFFFFF		Error / Warning: M=*MCC*, F=*FE*
LVID	0000 0000	1111 1111	Skipped Level 1 (0÷15) and Level 1Trigger ID (0÷15)
MCC-FE#	1110 0000	1110 1111	*FE* number in the module from 0 to 15
ROW#	0000 0000	1101 1111	Row number from 0 to 224
TOT	0000 0000	1111 1111	Time over Threshold

Table 6-3 shows the definition of all bits possibly generated by the *MCC* in the FE-Flag word.

Table 6-3 Definition of the MCC generated Warnings. Only 5 bits are used by the MCC-I2.1.

Bit position	Name	Description
FE-FLAG[8]	HitOverflow	There has been a Hit overflow inside the corresponding Receiver block.
FE-FLAG[9]	EoEOverflow	There has been an EoE overflow inside the corresponding Receiver block.
FE-FLAG[10]	Lv1ChkFail	The LV1ID check inside the corresponding Receiver has failed.
FE-FLAG[11]	BCIdChkFail	The BCID check between EoE words belonging to different *FE*'s has failed.
FE-FLAG[12]	Lv1ChkFail	The LV1ID check between EoE words belonging to different *FE*'s has failed.
FE-FLAG[15:13]	Empty	This three bits are always `000`.

For a complete description of the single bits please refer to the "Event Builder" chapter and the "Receiver" chapter, while for FE-Flag[7:0] refer to the *FE* specification document.

Figure 6-7 shows the flow diagram of the encoding/decoding algorithm that follows the grammar rules for the event data format. Please note that not all possible paths of the picture are possible.

⚠ As explained by the syntax for the event data structure, described before, one *MCC* event may never contain two consecutive FE-Flag words, but between two FE-Flag words there has always to be at least one synchronization bit followed by the FE# field.

In addition there can be a maximum of one FE-Flag word per *FE* chip, and therefore a maximum of 16 FE-Flag words per event.

The maximum number of Hits allowed per *FE* chip is 112 due to the fact that the *MCC* has 128 word FIFO's inside each Receiver and 16 words are reserved for EoE words.

As can be seen from Figure 6-7 the minimum size of an *MCC* Event is a Header followed by the L1ID and BCID information followed by a Trailer. The minimum event size totals 43 bits, that in turn, depending on the selected output speed, means a total number of 43, 22 or 11 clock cycles.

⚠ The 8 bit of the LV1Id field are splitted in two distinct fields: LV1Id<7:4> contains the number of skipped events, while LV1Id<3:0> contain a 4 bit LV1Id number. Both numbers are generated by the TTC (see "Trigger Timing & Control" chapter).

In case of skipped events by the *MCC* (this means that the *MCC* could not process all incoming Trigger commands due to the fact that by construction there can be a maximum number of 16 pending events inside the *MCC*) the last event processed by the *MCC* before the skipped ones will contain the number of skipped events in the LV1Id<7:4> field.

In order to allow the *ROD* to deal with skipped events (eventually adding missing empty events) additional 48 empty clock cycles are introduced by the *MCC* in the data stream before starting the next event.

6.5 MCC to ROD Physical Layer Protocol

The *MCC* has two differential pins which can be used to drive the *VDC* chip which in turn converts the LVDS signal to the suitable currents to drive the opto fibre going to the *ROD*.

The *MCC* has therefore four different ways to transmit data to the *ROD* through the opto-links. Each link can be used as a single 40 Mb/s or 80 Mb/s link. In addition the two links can operate as a single or a dual link. This allows for a peak bandwidth going from 40 Mb/s to 160 Mb/s.

The four operation modes are listed in the Table 6-4 and are graphically represented in Figure 6-8.

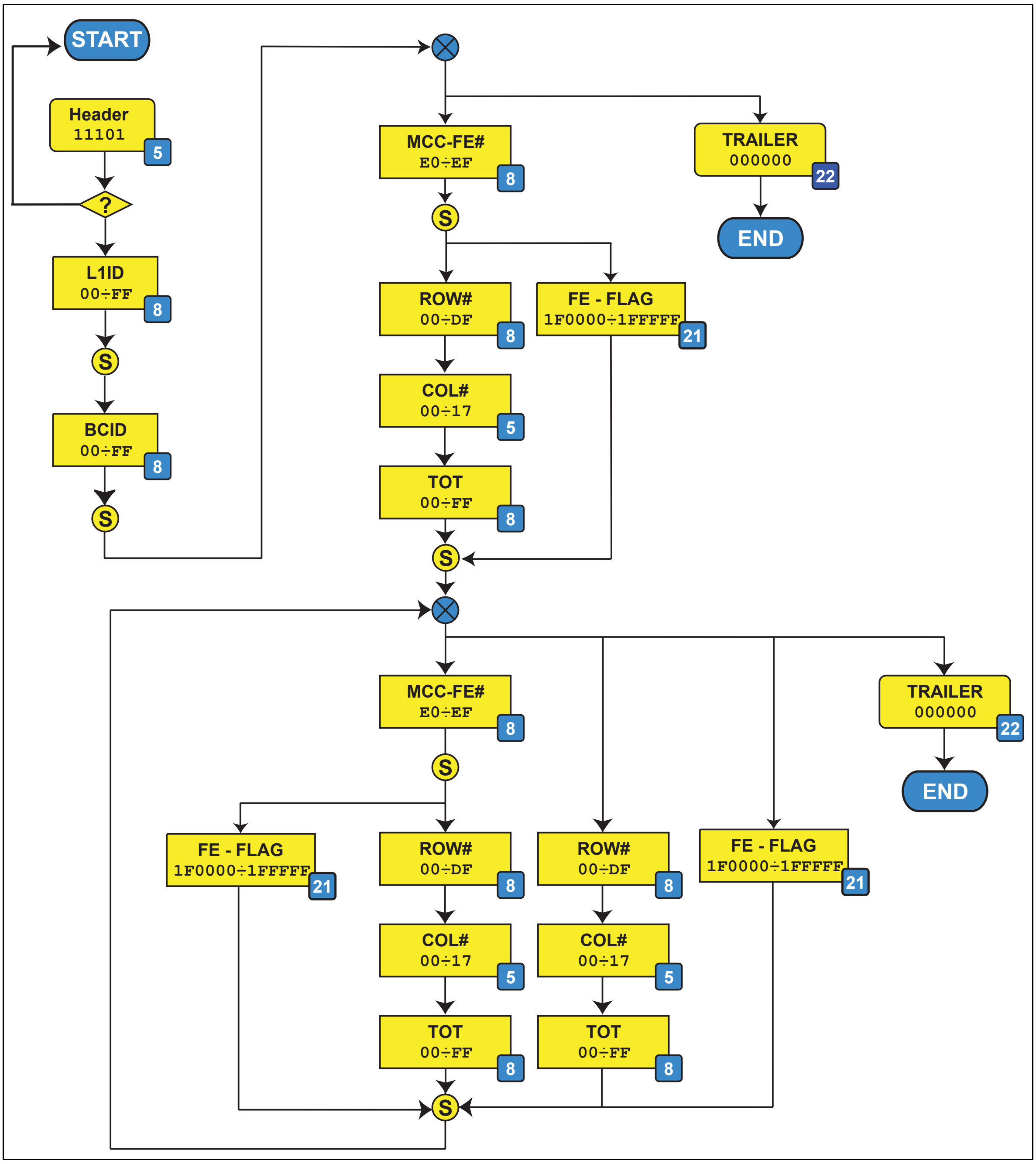

Figure 6-7 MCC encoding/decoding algorithm. Not all possible depicted paths are allowed.

Table 6-4 CSR bits setting for different link operation modes.

Mode	CSR<1:0>	Event Data	Event BW	Config. Data	Config. BW
40 x 1	0 0	2 links @ 40 Mb/s	40 Mb/s	2 links @ 40 Mb/s	40 Mb/s
40 x 2	0 1	2 links @ 40 Mb/s	80 Mb/s	2 links @ 40 Mb/s	40 Mb/s
80 x 1	1 0	2 links @ 80 Mb/s	80 Mb/s	2 links @ 40 Mb/s	40 Mb/s
80 x 2	1 1	2 links @ 80 Mb/s	160 Mb/s	2 links @ 40 Mb/s	40 Mb/s

The four link modes are selected by the two bits in CSR. In case of mode 40 x 1 and 80 x 1 the two links always transmit the same information and either one can be used. In case of dual link usage (40 x 2 and

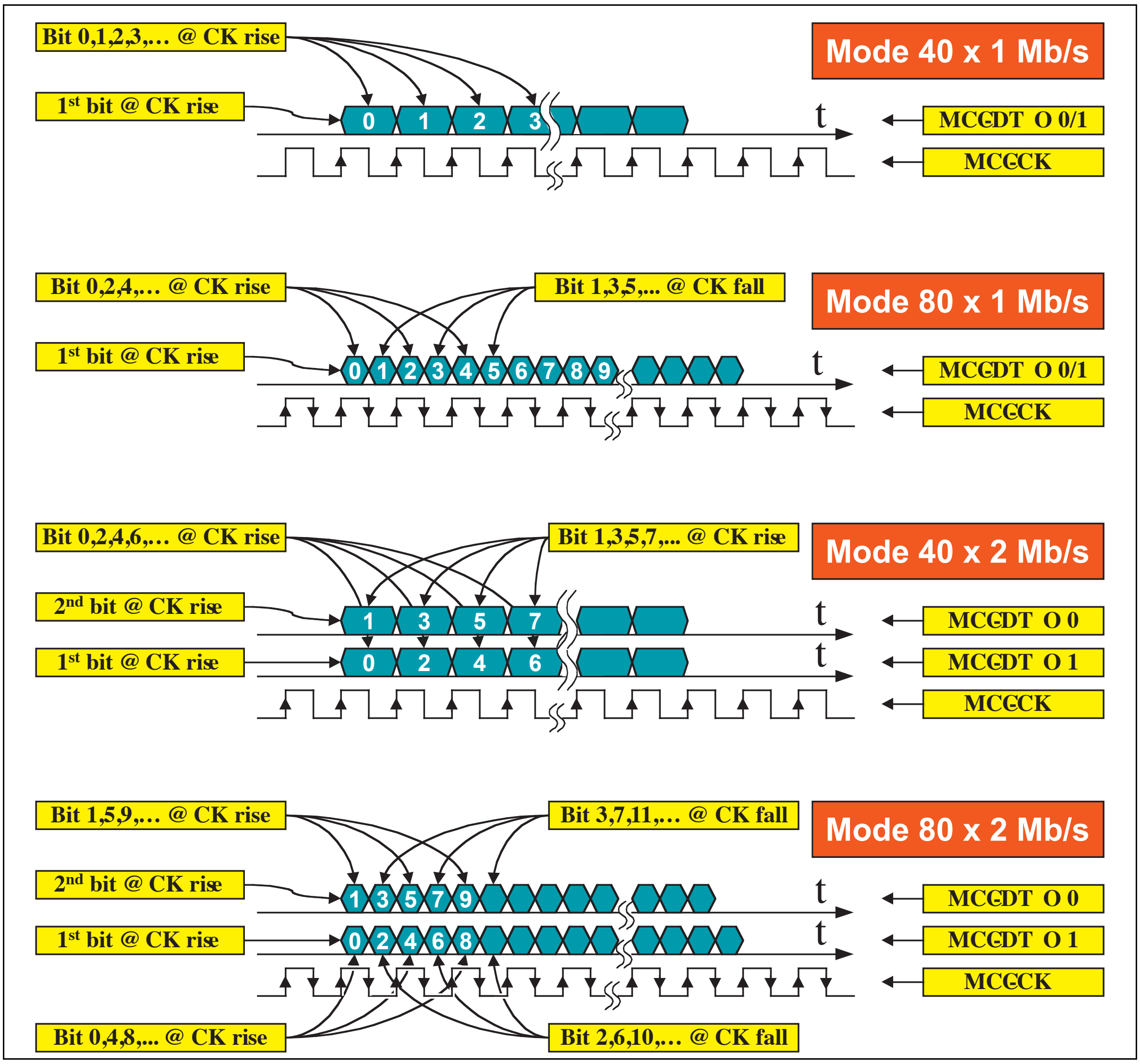

Figure 6-8 Data transmission on the two MCC outputs to the ROD for different link operation modes.

80 x 2) the first bit of every event record always starts on DTO-1 output, while the last bit may be either on DTO-1 or on DTO-0.

⚠ All configuration data, instead, is always transmitted at 40 Mb/s on both links independently from the link mode setting.