PLC Framing Structure

Marek Hajduczenia, PhD

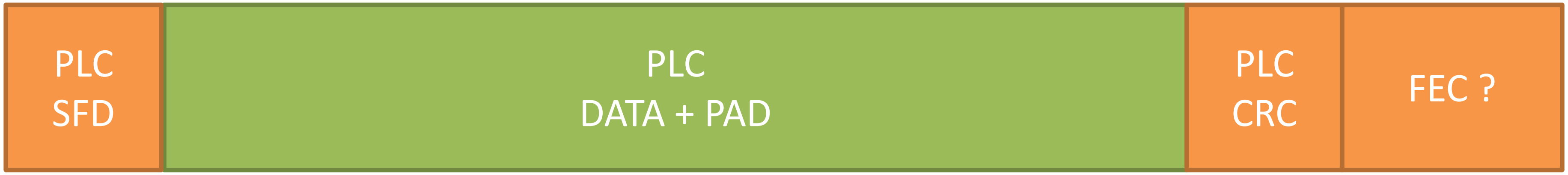
ZTE Corporation

marek.hajduczenia@zte.pt

Summary

- PLC needs well-designed framing structure, facilitating data recovery and supporting reliable exchange of data between peer stations
- Without PLC-based configuration capability for remote CNU PHY, there is no EPoC to start with.
- Reliability of PLC data transport should be the primary concern; PLC bandwidth comes second
- The following slides outline the proposed PLC frame structure, function of individual fields, and parsing process

PLC Frame Structure – DATA+PAD



PLC SFD

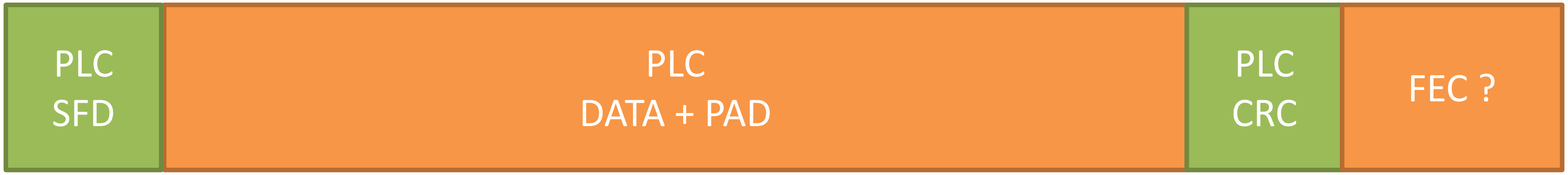
PLC DATA + PAD

PLC CRC

FEC ?

- We have to consider pros and cons of using a stream of fixed size “ATM-like data cells” versus on-demand “Ethernet-like” data frames with variable length
  - Fixed-size frame facilitates processing, but causes some bandwidth loss (due to padding and nop commands when there is nothing to transmit). Receiver side is always busy (nops are sent when there is no other data to transmit)
  - Variable-size frame is more bandwidth efficient, since frames are transmitted only when there is something to transmit. When there is nothing to send, PLC remains idle

PLC Frame Structure – SFD & CRC



PLC SFD

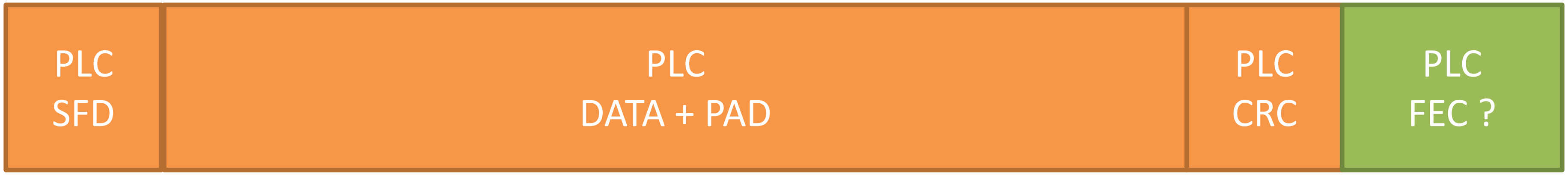
PLC DATA + PAD

PLC CRC

FEC ?

- Start of Frame Delimiter (SFD) is needed to facilitate searching for the start of the PLC frame
  - Otherwise, only FEC decoding and CRC checking can guarantee that we did indeed found the start of frame
  - PLC SFD sequence can be different than Ethernet SFD (longer?)
- CRC is needed to improve MTTFPA post-FEC
  - Ethernet CRC32 could be reused (readily available) and has known MTTFPA properties
  - Stronger CRC can be used if post-FEC bit error is high and MTTFPA with CRC32 is insufficient (analysis would be needed to prove CRC32 actually fails)

PLC Frame Structure – FEC



PLC SFD

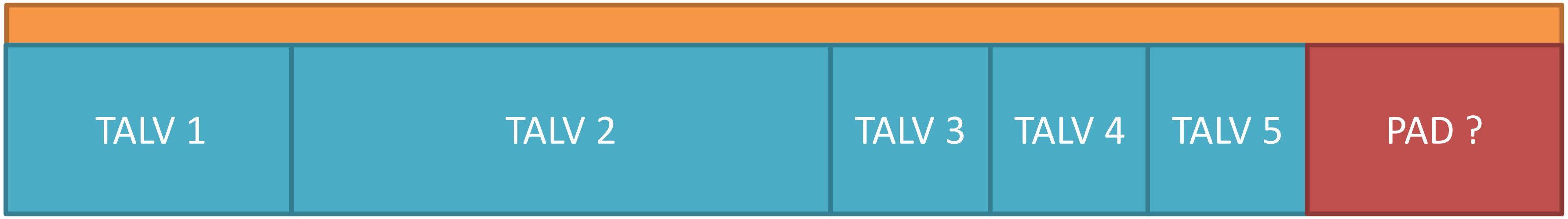
PLC DATA + PAD

PLC CRC

PLC FEC ?

- PLC link encoding guarantees resiliency to noise and bit errors. However, adding low-rate FEC should improve PLC's immunity to bit errors
- PLC frame error ratio out of FEC decoder needs to be defined in such a way that MTTFPA for PLC (PLC FEC + PLC CRC) is at least equal to the age of the Universe (actual calculations are TBD at this time, pending decision on PLC frame structure)

DATA+PAD Structure



- DATA+PAD portion of PLD Frame comprises a series of (Type/Address/Length/Value = TALV) fields and PAD, all with variable size
- PAD field is only present in a PLC frame when:
  - the PLC frame is fixed-size; and
  - the total size of the total size of the TALV fields is smaller than the size of the PLC frame
- PAD is a special case of a TALV (same structure)

TALV Structure

Type	Address	Length	Value

- Each TALV comprises of the following fields:
  - Type: encodes the target operation
  - Address: identifies the target (start) register for the operation; reading/writing into sequential registers can be also supported
  - Length: length of the Value field (excluding Type and Address fields)
  - Value: carries the actual value for the given (set of) registers and bits within registers

TALV Structure (T)

- Field: Type
  - Size: 1 octet
  - Function: encodes the operation (pad / nop / read / read sequence / write / write sequence / etc.) for the given TALV. 255 codes possible, some may be reserved for now.
    - 0x00: pad
    - 0x01: nop
    - 0x02: read
    - 0x03: read sequence
    - 0x04: read response
    - 0x05: write
    - 0x06: write sequence
    - 0x07: write response
    - 0xFF: CNU ID follows
  - This field is always present in a TALV

TALV Structure (A)

- Field: Address
  - Size: 4 octets
  - Function: encodes the address of the specific (start) register and (start) bit within the register in the format: MMD.Register.bit (1/2/1 octets long)
  - This field is always present in a TALV; for Type == 0x00 (pad) or Type = 0xFF (CNU ID), address is equal to 0xFF.0xFF-FF.0xFF
  - For sequential read/write operations, address has the format of MMD.Register.0 (sequential operations done always on complete registers)

TALV Structure (L)

- Field: Length
  - Size: 1 octet, LSB = 1 indicates that remaining 7 bits express length of Value field in bits; LSB = 0 indicates that remaining 7 bits express length of Value field in octets (for sequence read/write operations)
  - Function: encodes the length of the value carried in the Value field. When LSB = 1, the size of the Value field (in octets) is calculated as follows:

    ceil(Length/8)
  - For example, when accessing PMA/PMD reset bit, 1.0.15, Length field carries value of 0b10000001, Value field is 1 (ceil(1/8)) octet long and only the first bit is meaningful
  - Always present in a TALV; when Type == 0x00 (pad), Length LSB = 0 (length expressed in octets); when Type == 0xFF (CNU ID), Length = 0x06 (CNU MAC address)

TALV Structure (V)

- Field: Value
  - Size: variable
  - Function: carries the actual value to be stored in / read from the given (set of) register(s).
    When single bit is accessed, its value is stored starting from the LSB in Value field, and Length indicates how many bits in Value field are significant.
    When sequence read/write is performed, Value field contains value for multiple sequential registers.
    When Type == 0x00 (pad), Value field contains a sequence of 0x00 (all zeros) with length indicated by Length field
  - This field is present in a TALV apart from read / read sequence request messages

Return codes

- Carried in Value field of TALV message sent by CNU in response to write / read request from CLT
- Some values we need to define (examples):
  - 0xFF: write OK
  - 0xFE: write sequence OK
  - 0xFD: write failed
  - 0xFC: write sequence failed
  - 0xFB: read failed, no such register
  - 0xFA: read failed, register not accessible
  - … more to come if needed

Message context / address

- The message context (the target CNU or CNUs the following TALVs are intended for) is set once and remains valid until changed.
- This limits the number of CNU IDs that have to be sent over the PLC, while still allowing a compliant CLT send CNU ID before every message (if needed)
- CNU ID may be equal to CNU MAC address (or not). The currently reserved message space is 6 octets wide.
- CNU ID may identify a single CNU, or all CNUs (CNU ID of 0xFF-FF-FF-FF-FF-FF).

Examples (1)

- Read the value of a bit from address 1.0.15 (PMA/PMD reset bit):

Type	Address	Length
0x02	0x01.0x00-00.0x10	0x00

read request sent to CNU

Type	Address	Length	Value
0x04	0x01.0x00-00.0x10	0b10000001	0b1xxxxxxx

read response from CNU

- Write a value of 0x11-12-13-15 into a 64-bit register located at 8.9800.0

Type	Address	Length	Value
0x05	0x08.0x26-48.0x00	0x04	0x11-12-13-15

write request sent to CNU

Type	Address	Length	Value
0x07	0x08.0x26-48.0x00	0x01	0xFF

write response from CNU
(0xFF = write OK)

Examples (2)

- Read the value from register 1.8900.0 (in this example, such register does not exist):

Type	Address	Length
0x02	0x08.0x26-48.0x00	0x00

read request sent to CNU

Type	Address	Length	Value
0x04	0x08.0x26-48.0x00	0x01	0xFB

read response from CNU
(0xFB = such register does not exist)

- Read sequence of 128-bits starting from address 8.8900.0 (8 consecutive registers)

Type	Address	Length	Value
0x03	0x08.0x26-48.0x00	0x01	0x08

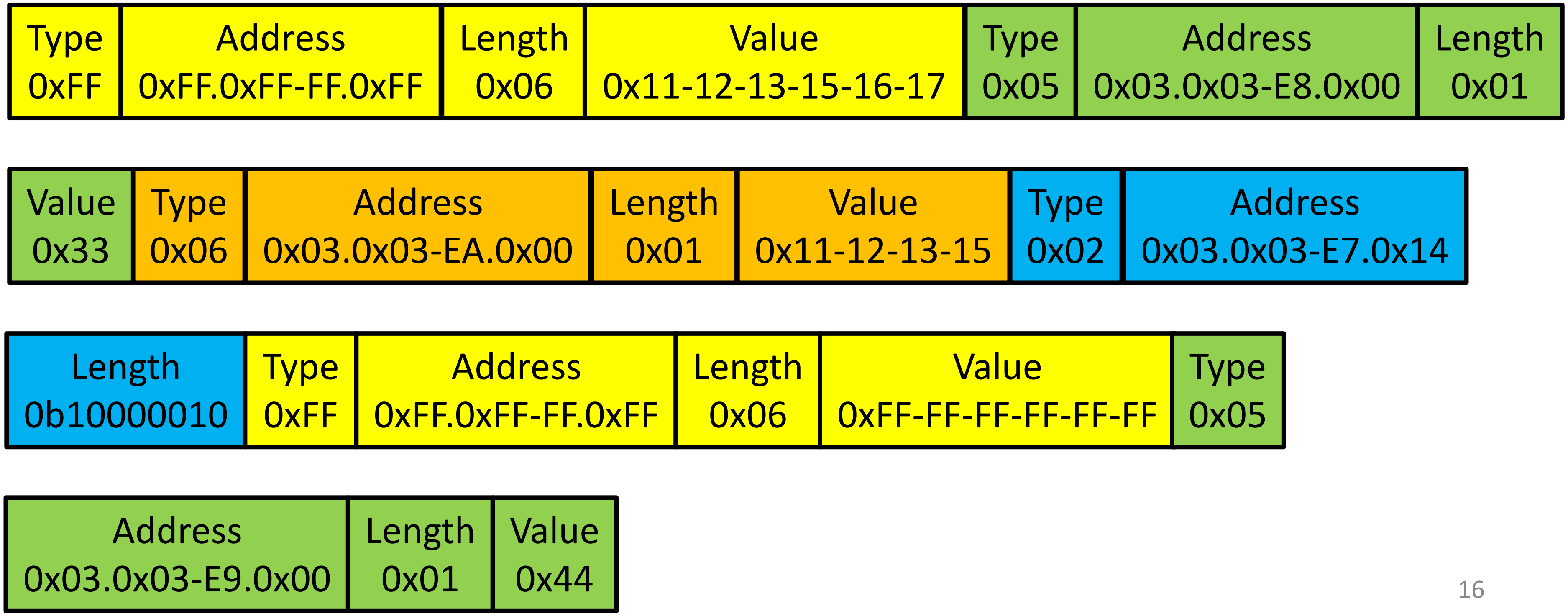
read sequence request sent to CNU
(give me 8 consecutive registers)

Type	Address	Length	Value
0x07	0x08.0x26-48.0x00	0x08	0x11-12-13-15-16-17-18

read sequence response from CNU

Examples (3a)

- A sequence of few operations to specific CNU: write of a value into register 3.1000.0 (whole register, 1 octet), write of a value into register 3.1002.0 (whole register, 4 octets), and read of value from register 3.999.14 (2 bits), followed by broadcast write request to all CNUs:



Type	Address	Length	Value	Type	Address	Length
0xFF	0xFF.0xFF-FF.0xFF	0x06	0x11-12-13-15-16-17	0x05	0x03.0x03-E8.0x00	0x01

Value	Type	Address	Length	Value	Type	Address
0x33	0x06	0x03.0x03-EA.0x00	0x01	0x11-12-13-15	0x02	0x03.0x03-E7.0x14

Length	Type	Address	Length	Value	Type
0b10000010	0xFF	0xFF.0xFF-FF.0xFF	0x06	0xFF-FF-FF-FF-FF-FF	0x05

Address	Length	Value
0x03.0x03-E9.0x00	0x01	0x44

Examples (3b)

- CNU (ID: 0x11-12-13-15-16-17) confirms write of requested values into registers: 3.1000.0, 3.1002.0 (sequence of 4), and 3.1001.0; also return value from register 3.999.14 (2 bits):

Type	Address	Length	Value	Type	Address	Length
0xFF	0xFF.0xFF-FF.0xFF	0x06	0x11-12-13-15-16-17	0x07	0x03.0x03-E8.0x00	0x01

Value	Type	Address	Length	Value	Type	Address
0xFF	0x07	0x03.0x03-EA.0x00	0x04	0xFE	0x02	0x03.0x03-E7.0x14

Length	Value	Type	Address	Length	Value
0b10000010	0b11000000	0x07	0x03.0x03-E9.0x00	0x01	0xFF

- PAD encoding (when and if needed): 14 octets long

Type	**Address**	**Length**	**Value**
0x00	**0xFF.0xFF-FF.0xFF**	**0x08**	**0x00-00-00-00-00-00-00**

Conclusions

- The main decision points for PLC frame design have been outlined and need to be discussed by the ad-hoc in a timely fashion
- Proposal for PLC frame structure was made, including individual fields, TALVs, interpretation of the fields and their meaning
- Operational examples were shown, for read / write operations on bits and whole registers, as well as register sequences.