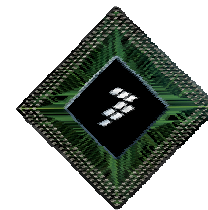


March 6, 2007

FlexRay Solutions and Concepts from Freescale

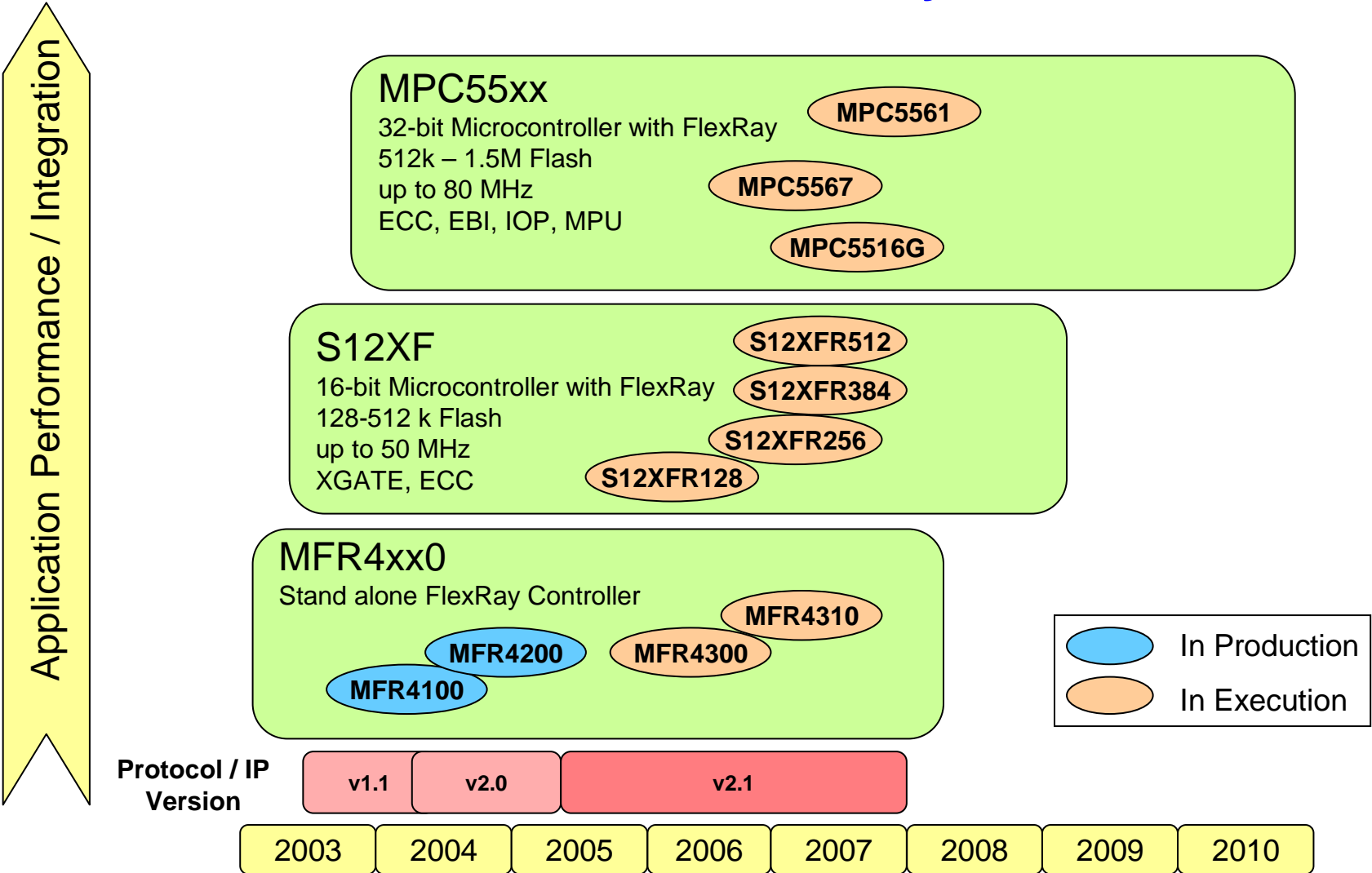


Mathias Rausch
Dr.-Ing.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners. © Freescale Semiconductor, Inc. 2006.



Freescale's FlexRay Product Roadmap



Freescal: MFR 4200A

- ▶ MFR 4200 introduced by Freescale in Q2/2003
- ▶ Redesign MFR 4200 Rev.A in 04/2003
- ▶ Based on protocol specification v1.9
- ▶ Available to lead customers May 2004
- ▶ General market June 2004
- ▶ Automotive qualified in Q2/2005
- ▶ First FlexRay controller used in a series car (BMW X5) in Q4/2006

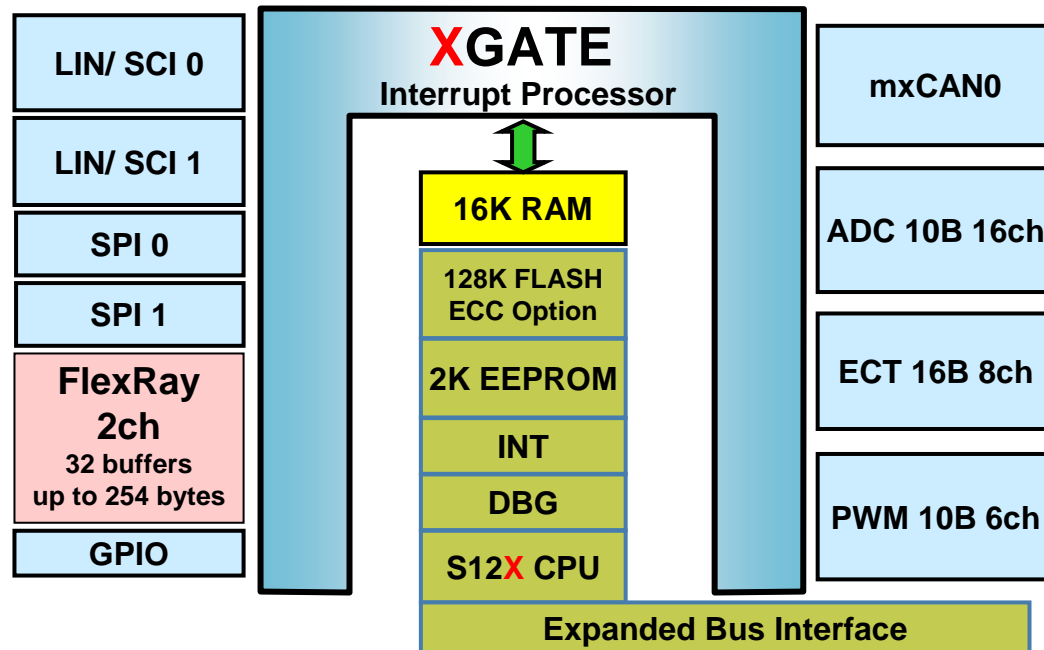


Freescal: MFR 4300

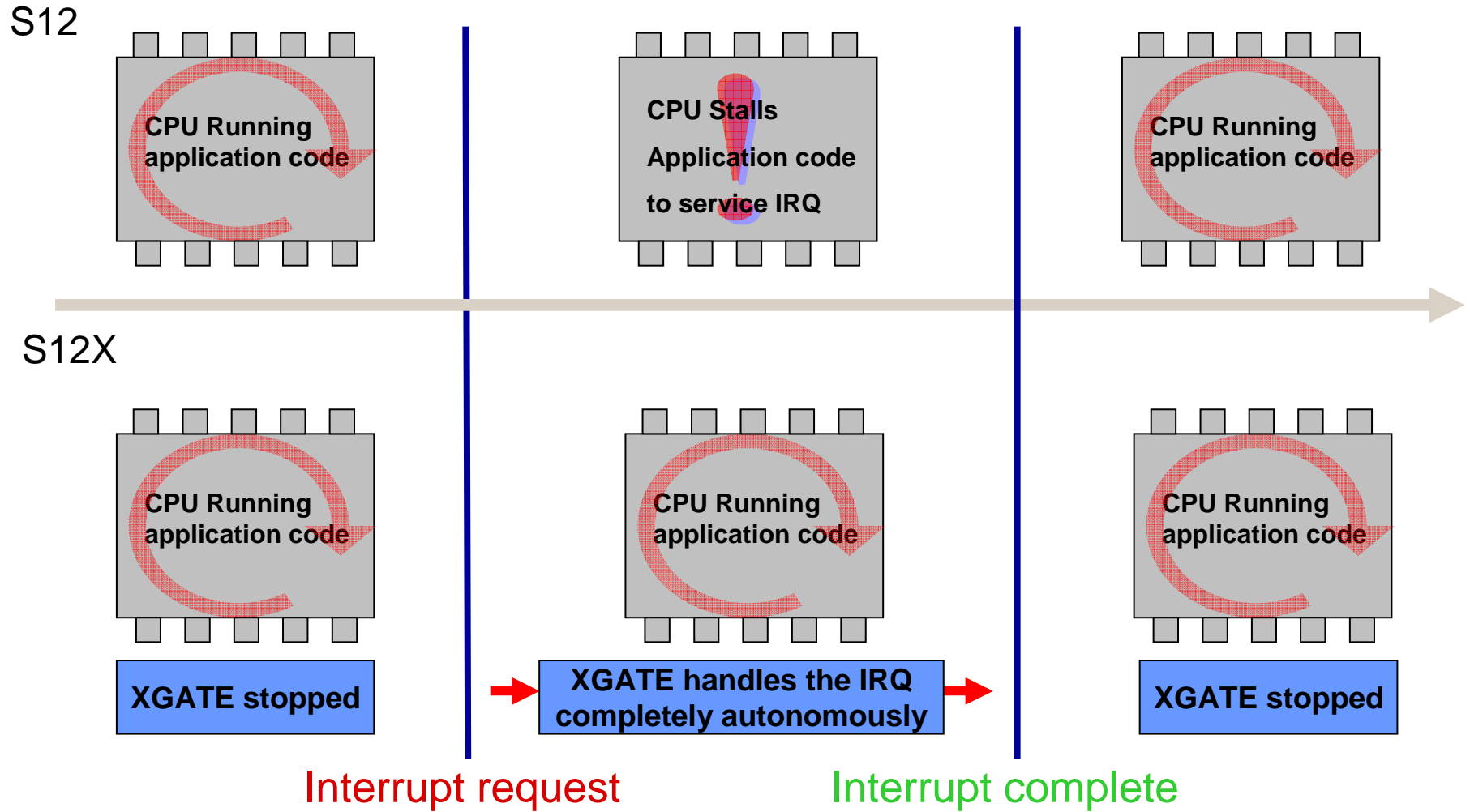
- ▶ introduced by Freescale Q4/2005
- ▶ based on protocol specification 2.1 but conformance test not passed
- ▶ supports 128 buffers with individual frame ID, channel ID, and cycle counter filtering
- ▶ two independent message data sections configurable from 0 to up to 254 bytes
- ▶ one FIFO per channel with up to 256 entries per FIFO
- ▶ transmit message buffers configurable with state/event semantics
- ▶ single and dual channel support
- ▶ zero padding for transmit message buffers in static segment
- ▶ individual message buffer reconfiguration supported
- ▶ one absolute timer
- ▶ one timer that can be configured to absolute or relative

Freescal: MC9S12XFR/E128 / 256 / 384 / 512

- Based on S12X-Family (16 bit Microcontroller), Integrated FlexRay Module
- introduced by Freescal in Q4/2005
- supports 32 buffers with up to 254 bytes, configurable in two length segments
- one FIFO per channel
- packages: 144 LQFP, 112 LQFP, 80 QFP, 64 LQFP
- based on protocol spec 2.1 but conformance test no passed

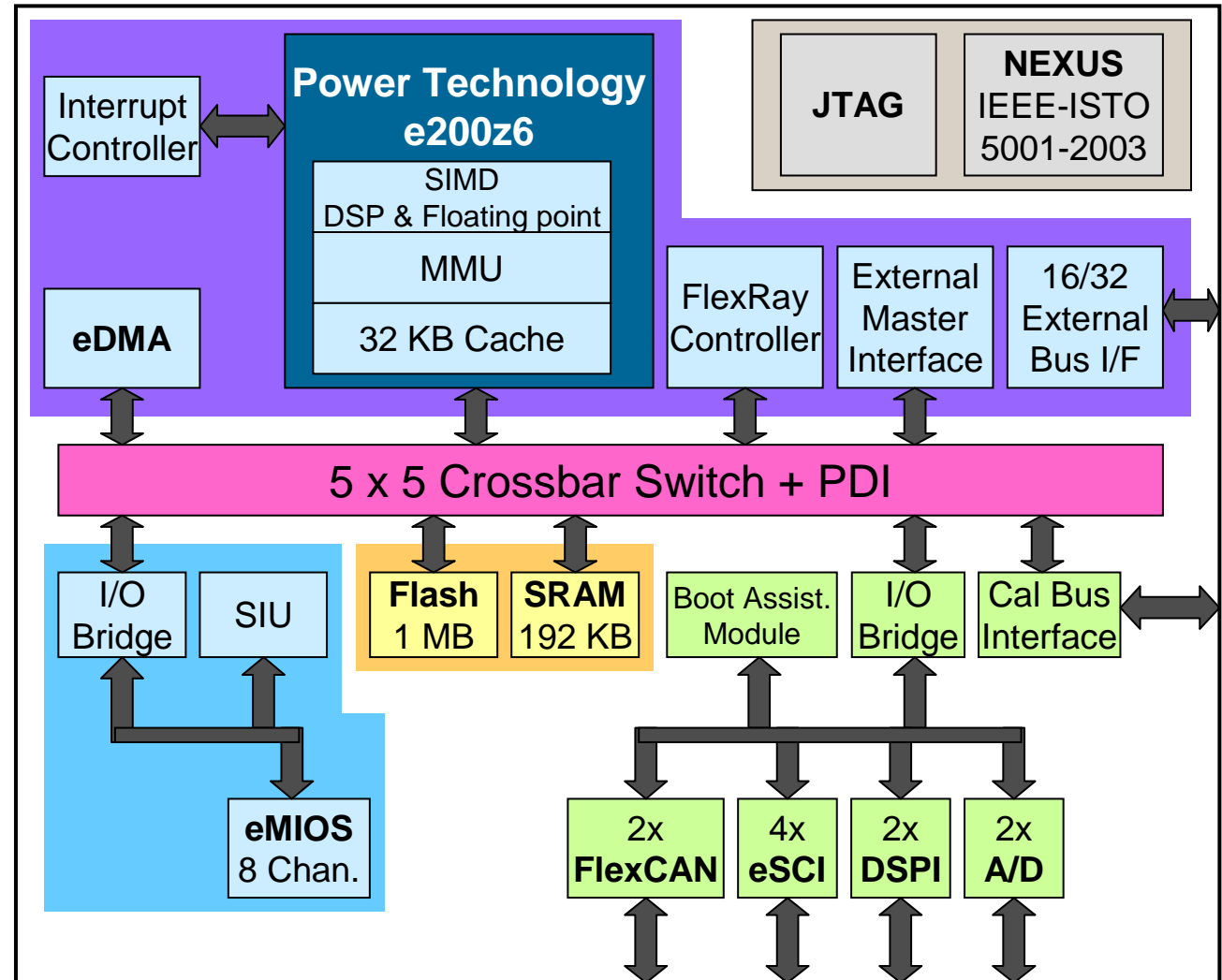


Application Level Support Example S12X



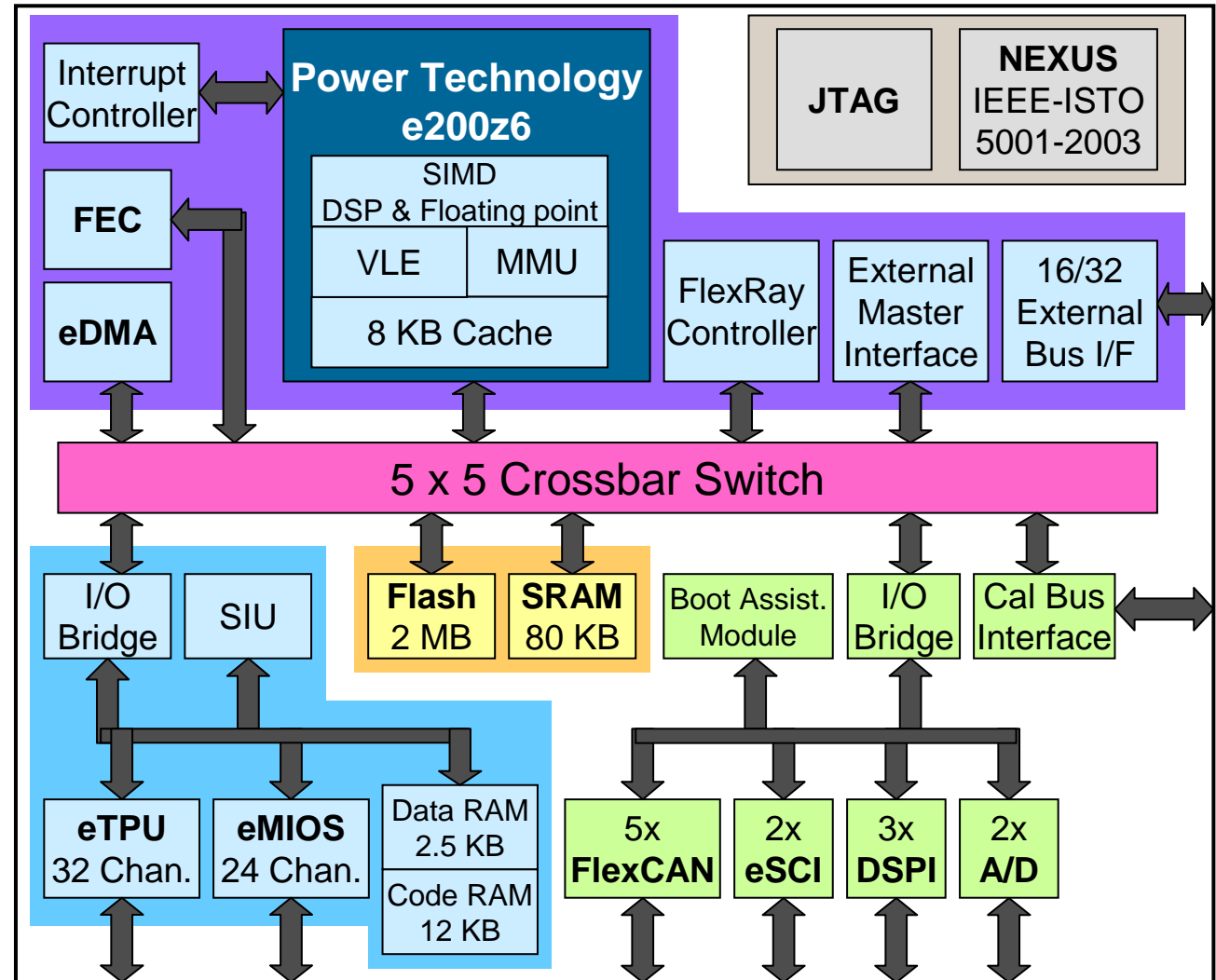
Freescale: MPC5561

- ▶ e200z6 Core
- ▶ 40–132 MHz
- ▶ Flash with ECC
- ▶ RAM with ECC
- ▶ 64 channel DMA
- ▶ FM-PLL
- ▶ 324 PBGA package
- ▶ 5/3.3V I/O



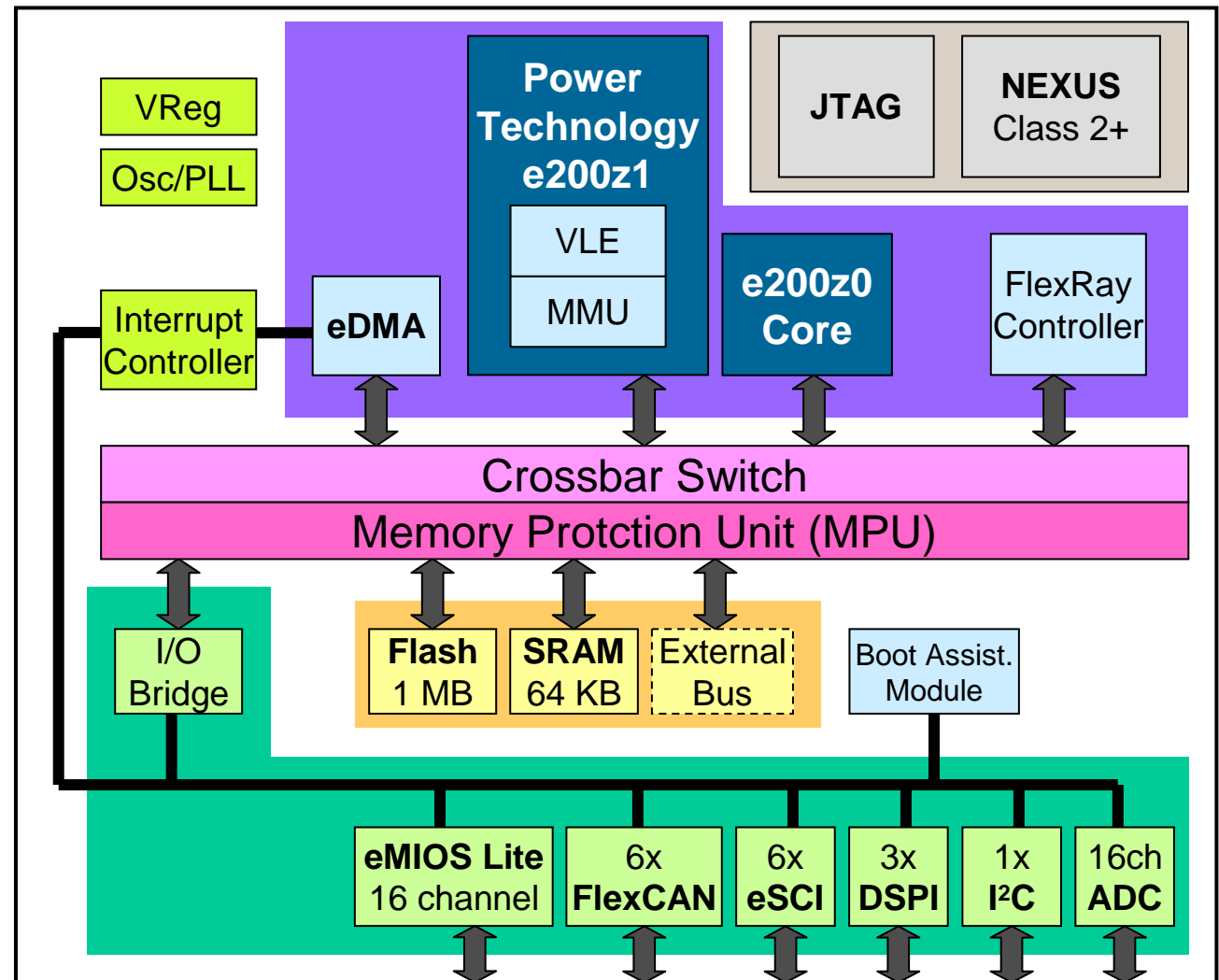
Freescale: MPC5567

- ▶ e200z6 Core
- ▶ 40–132 MHz
- ▶ Flash with ECC
- ▶ 32 channel DMA
- ▶ Fast Ethernet Contr.
- ▶ 281 Source Interrupt Controller
- ▶ FM-PLL
- ▶ 324 PBGA package
416 PBGA package
- ▶ 5/3.3V I/O
- ▶ 40 Ch. Dual ADC

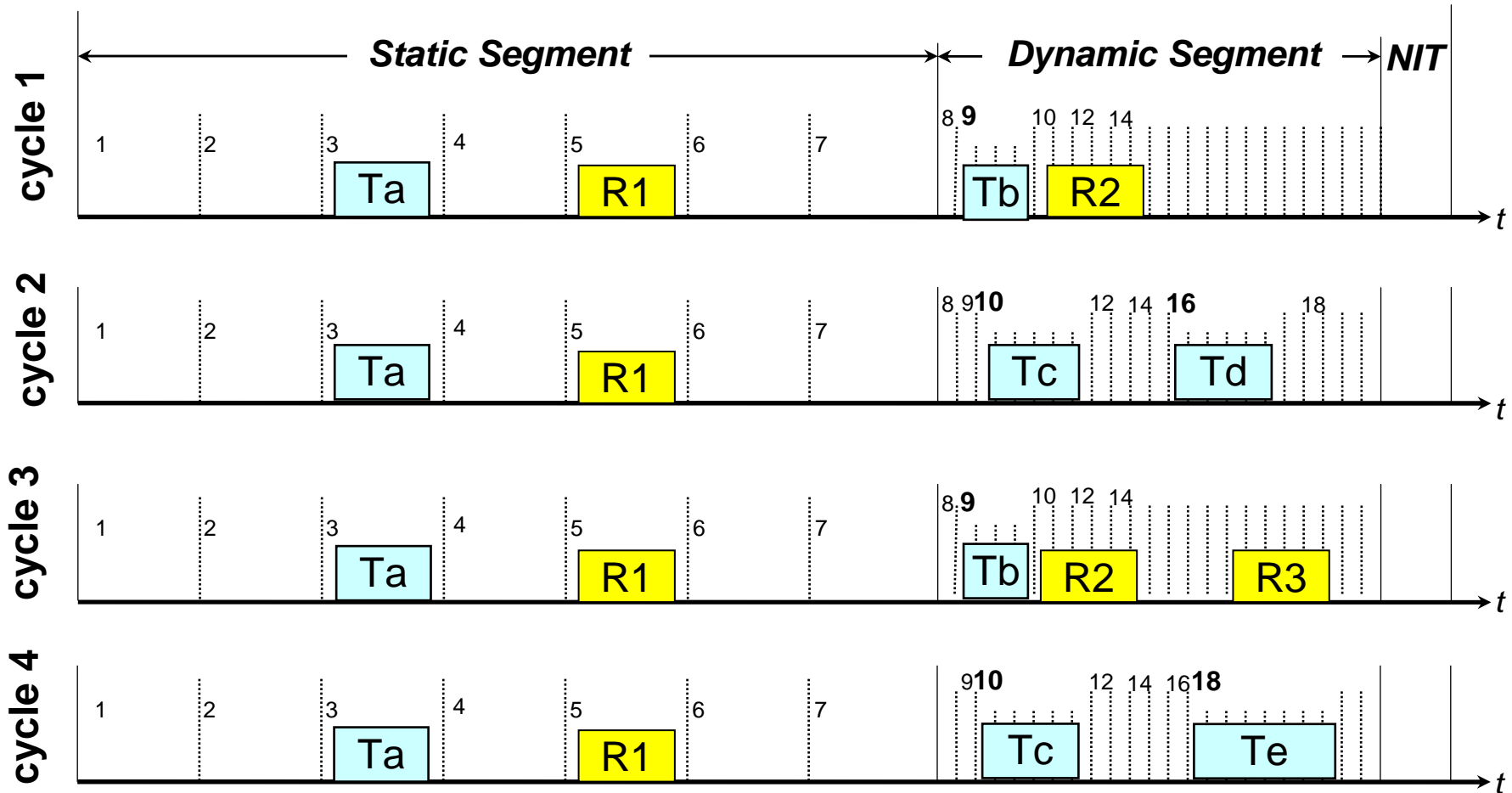


Freescale: MPC5516G

- ▶ e200z1 Core
- ▶ 66 MHz
- ▶ Flash with ECC
- ▶ RAM with ECC
- ▶ 144 LQFP package
208 MAPBGA package
- ▶ 5V I/O
- ▶ 5V ADC



Transmit and Receive Schedule



How many buffers are needed?

Buffer Configuration (1)

Static Segment

Ta	buffer 1
R1	buffer 6

Dynamic Segment

Tb	buffer 2 – cycle counter filter, every second cycle
Tc	buffer 3 – cycle counter filter, every second cycle
Td	buffer 4 – cycle counter filter, every fourth cycle
Te	buffer 5 – cycle counter filter, every fourth cycle
R2	buffer 7 – cycle counter filter, every second cycle
R3	buffer 8 – cycle counter filter, every fourth cycle

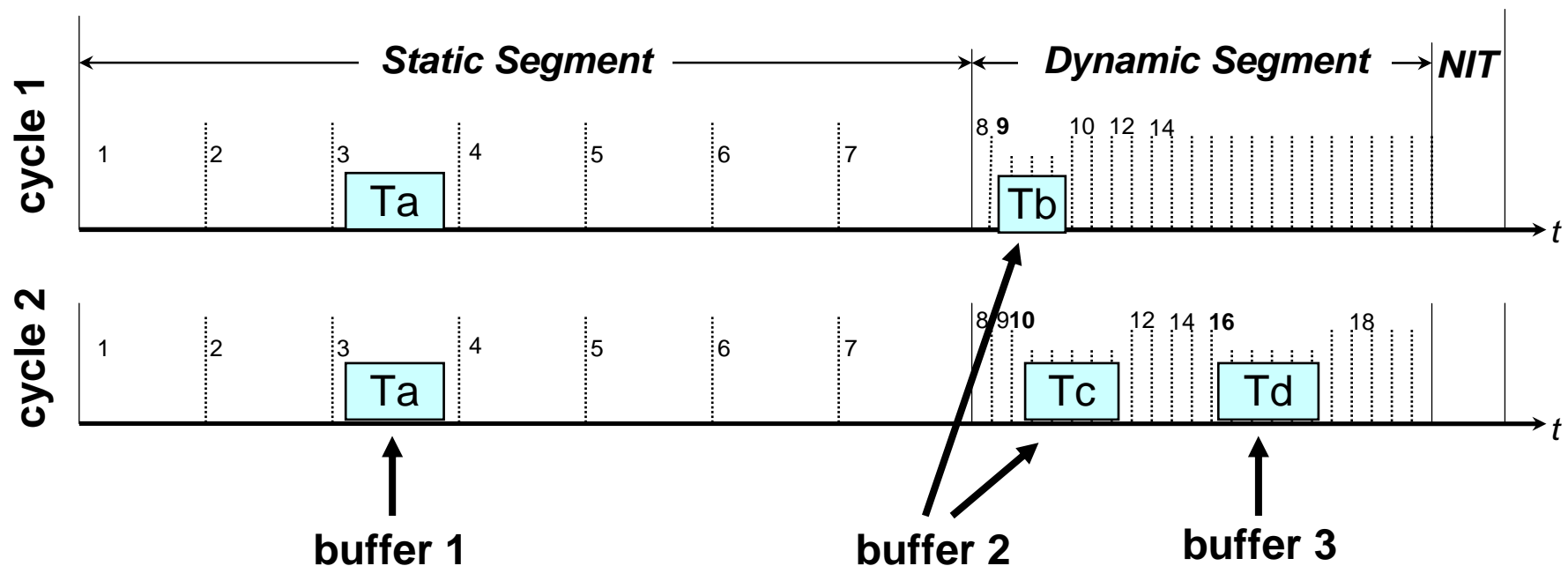
8 buffers are required

Buffer Reconfiguration (1)

How many buffers are needed?

- using buffer 2 in cycle $n+1$ for slot 9
- using buffer 2 in cycle $n+2$ for slot 10

saves one buffer but introduces some overhead in the software



Buffer Configuration (2)

Static Segment

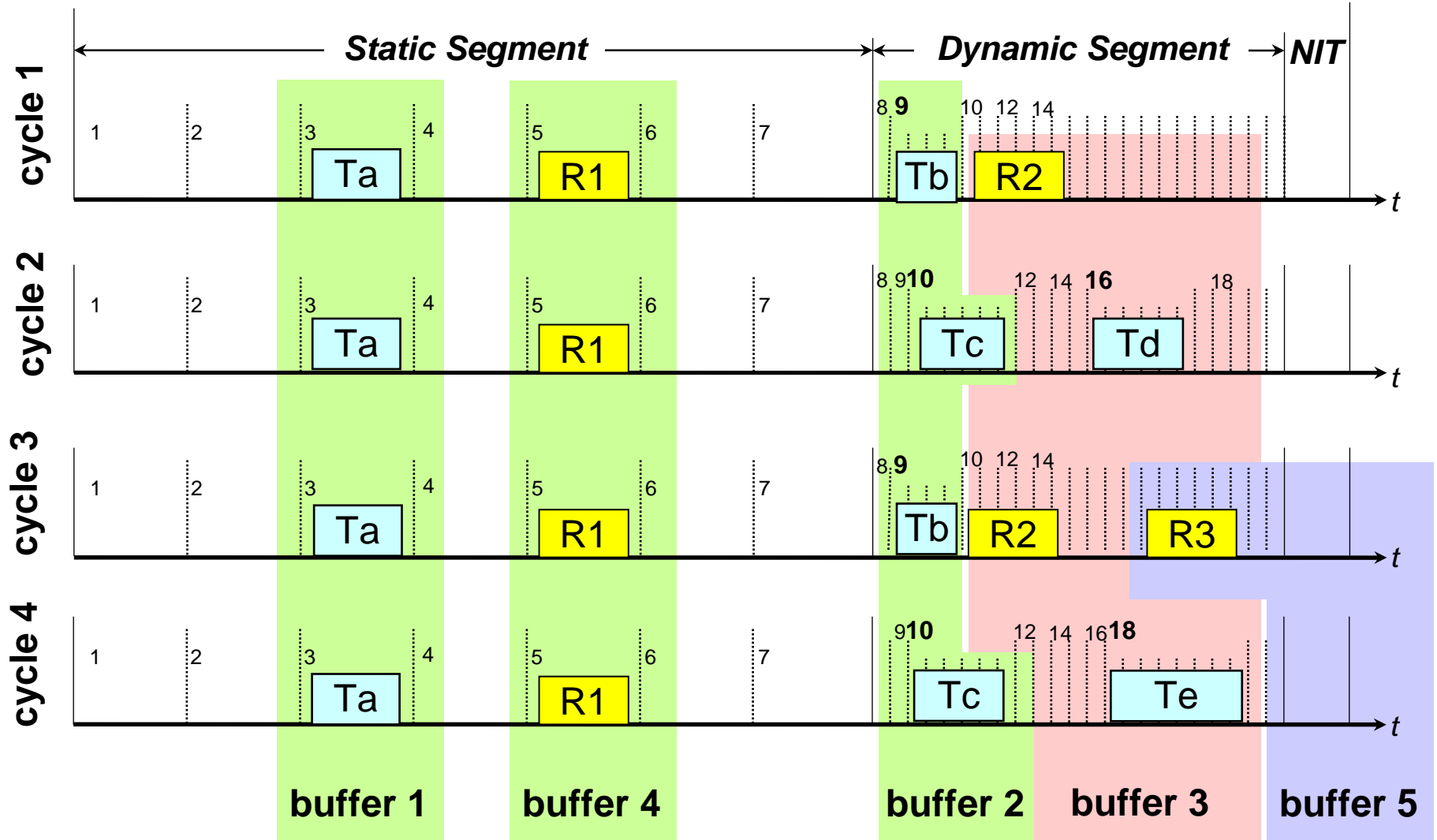
Ta	buffer 1
R1	buffer 4

Dynamic Segment

Tb	}	buffer 2 – reconfiguration after every cycle
Tc		
Td	}	buffer 3 – reconfiguration after every second cycle, cycle counter filter to even
Te		
R2	buffer 5 – cycle counter filter, every second cycle	
R3	buffer 6 – cycle counter filter, every fourth cycle	

6 buffers are required

Buffer Reconfiguration (2)



Buffer Configuration (3)

Static Segment

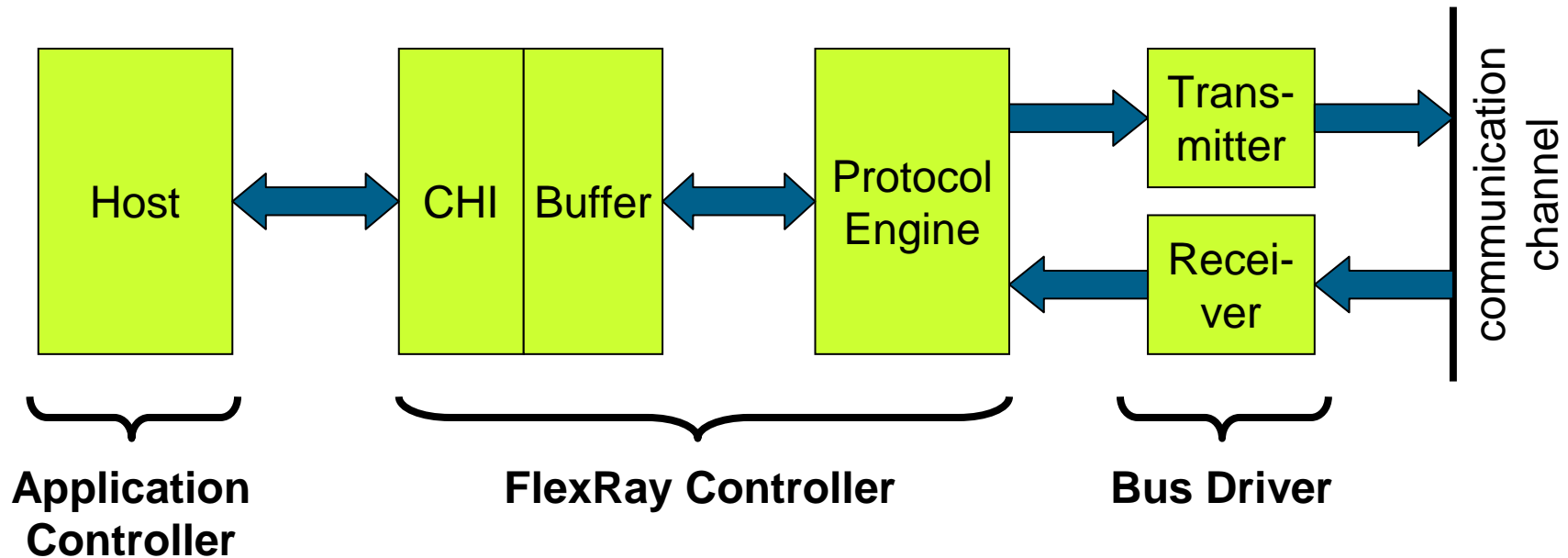
Ta	buffer 1
R1	buffer 4

Dynamic Segment

Tb	}	buffer 2 – reconfiguration after every cycle
Tc		
Td	}	buffer 3 – reconfiguration after every cycle changing between Tx and Rx
Te		
R2		
R3	buffer 5 – cycle counter filter, every fourth cycle	

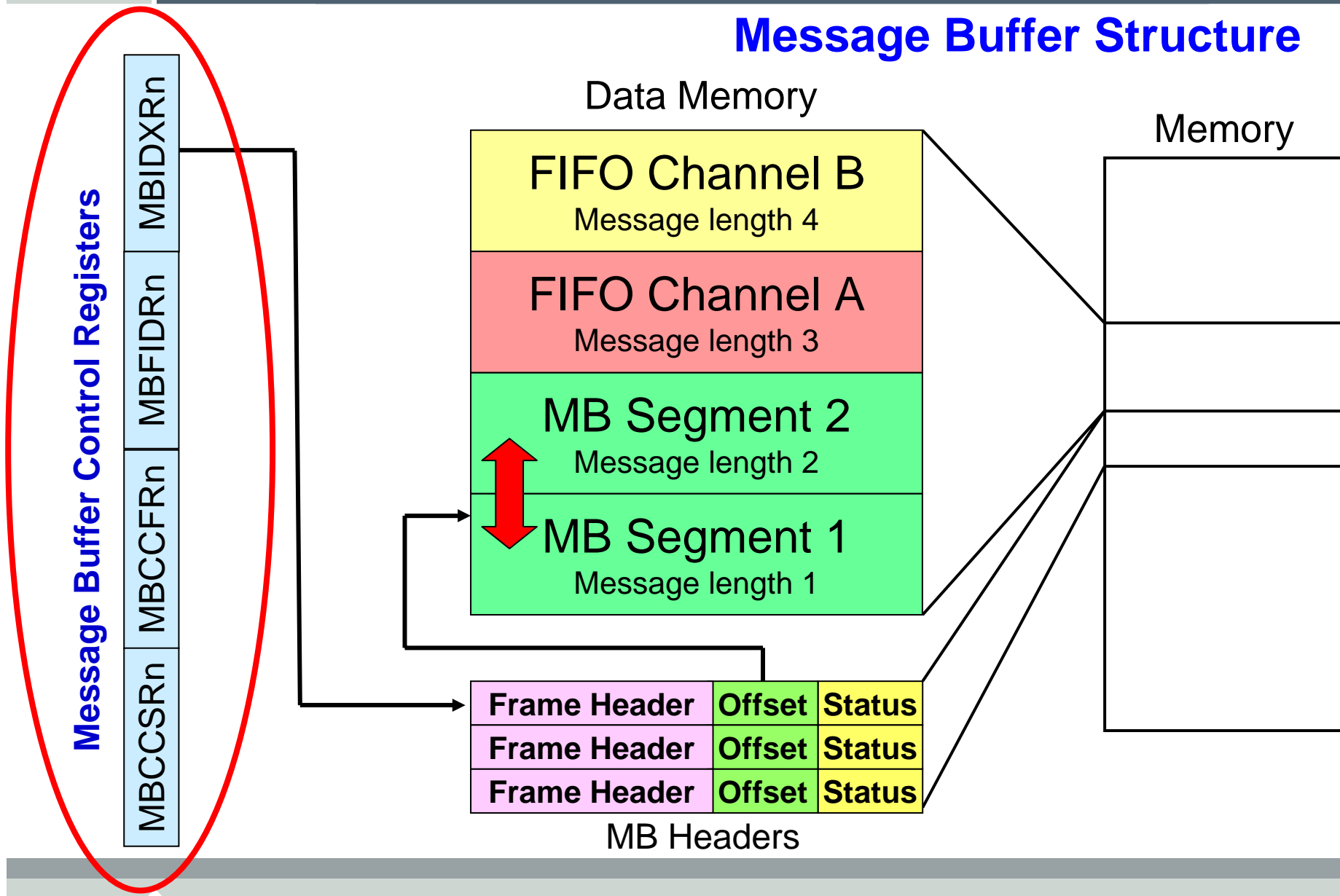
5 buffers are required

Message Flow in FlexRay



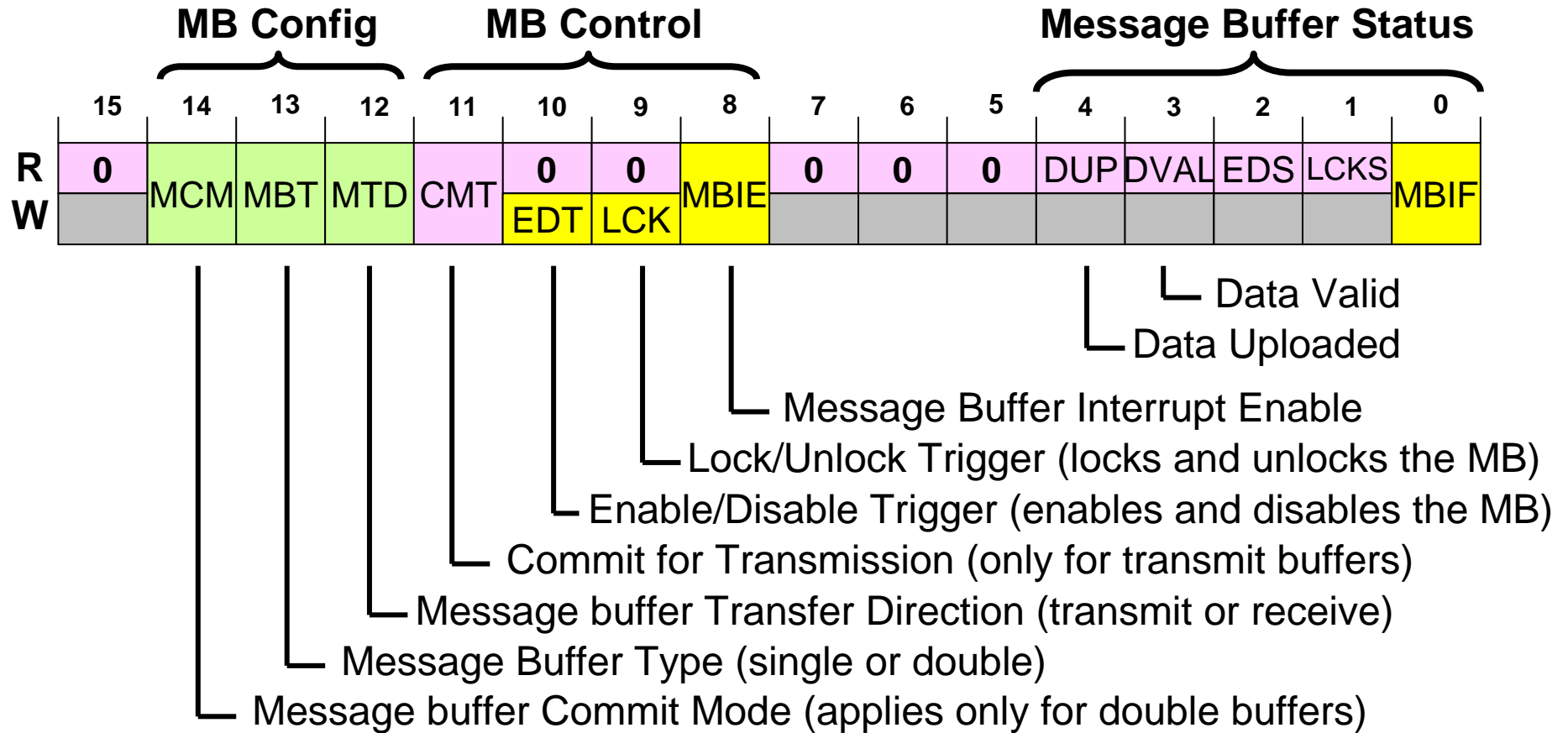
- The design of the Controller Host Interface (CHI) dominates the user interface of the FlexRay device.
- The CHI is **not** specified in the FlexRay protocol specification.
- The silicon manufacture has a high degree for freedom.

Message Buffer Structure



Message Buffer Configuration Registers

Message Buffer Configuration, Control, Status Registers (MBCCSRn)

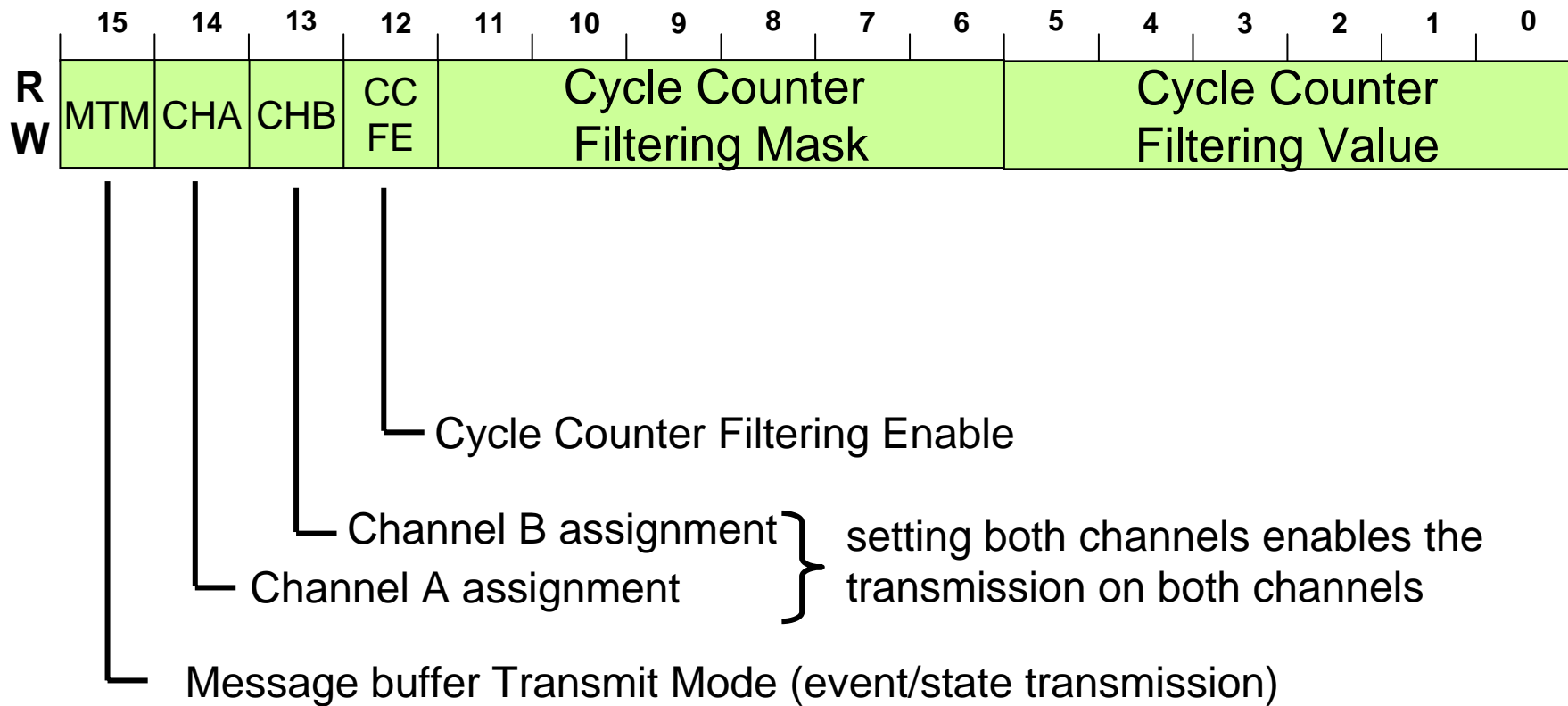


write in Normal Mode

write in POC:config or MB disabled

Message Buffer Configuration Registers

Message Buffer Cycle Counter Filtering Registers (MBCCFRn)



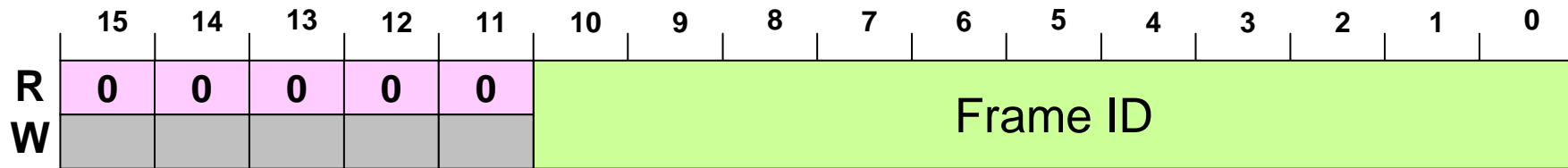
write in Normal Mode



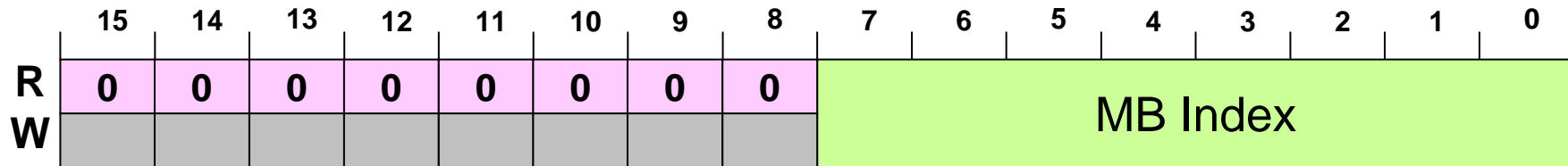
write in POC:config or MB disabled

Message Buffer Configuration Registers

Message Buffer Frame ID Registers (MBFIDRn)



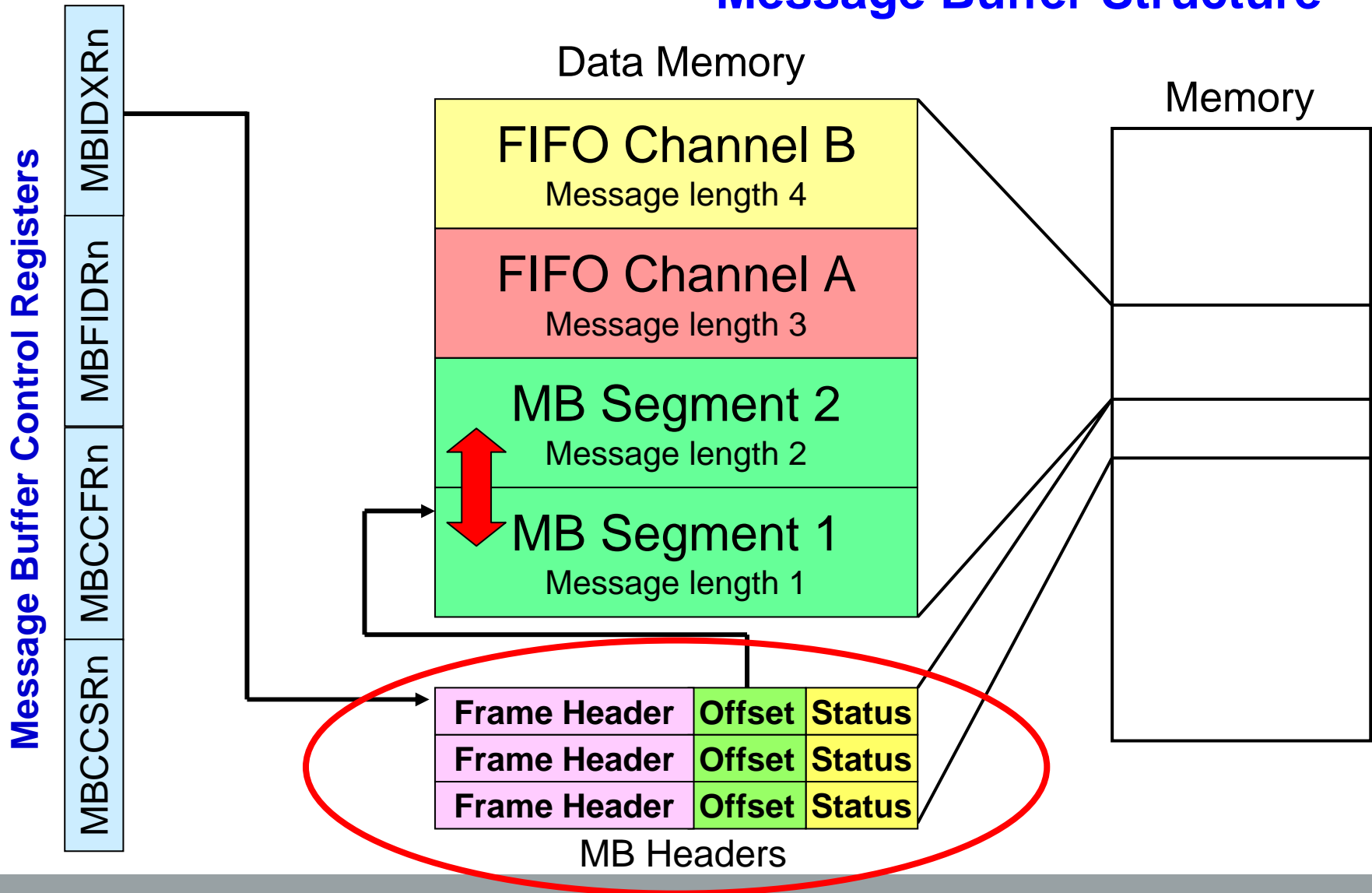
Message Buffer Frame ID Registers (MBIDXRn)



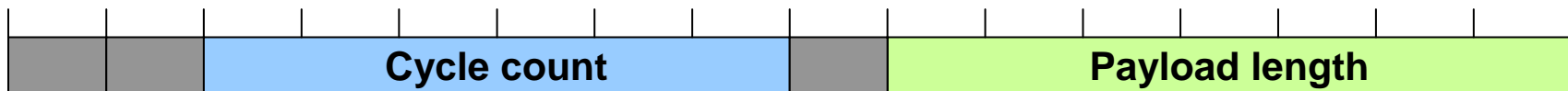
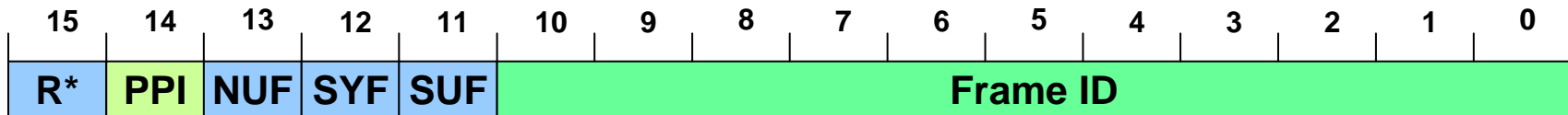
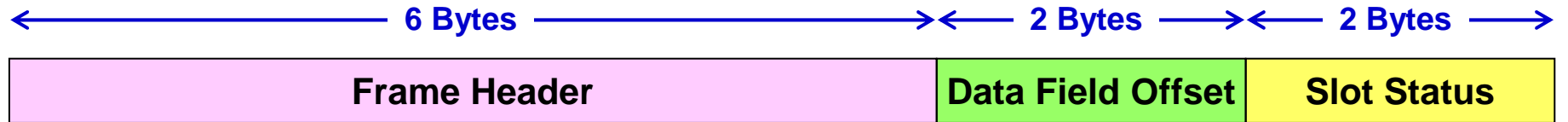
 write in Normal Mode

 write in POC:config or MB disabled

Message Buffer Structure

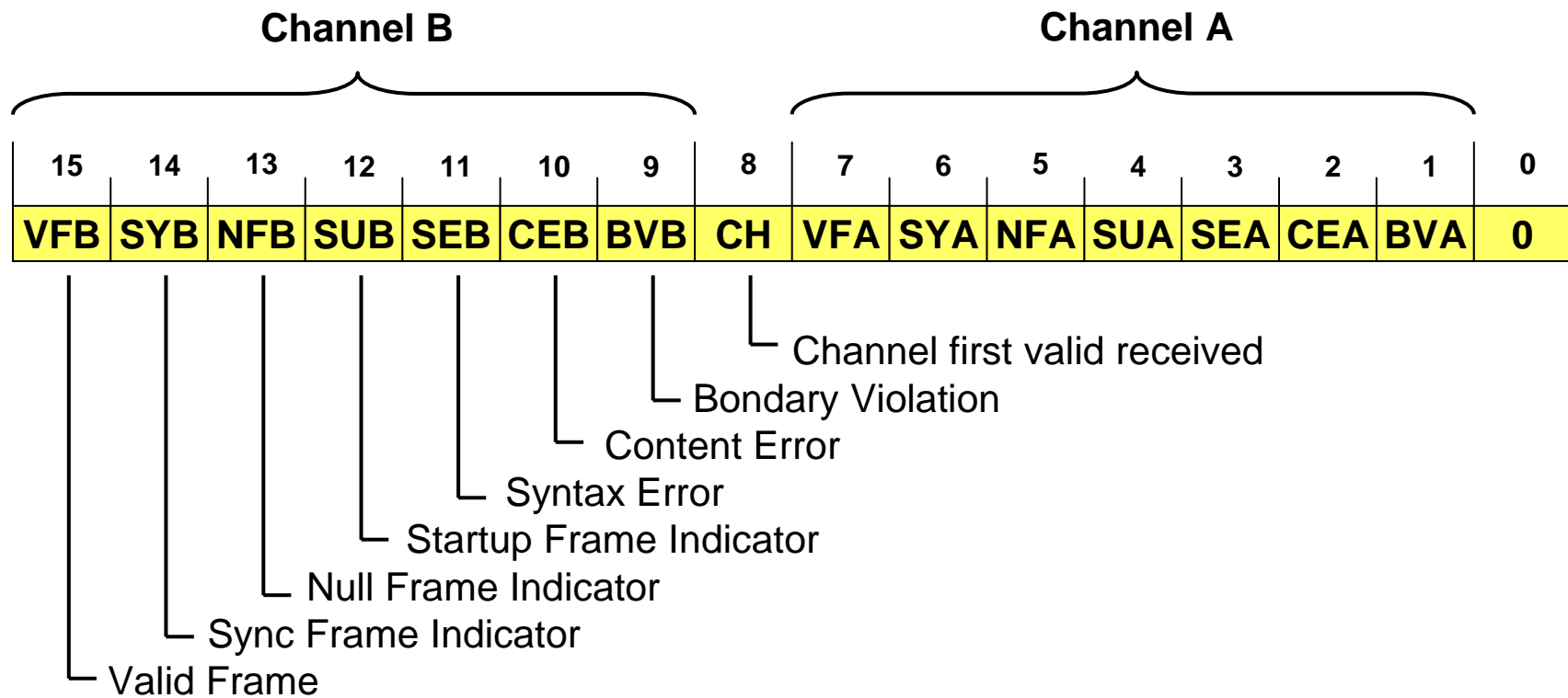


Message Buffer Structure

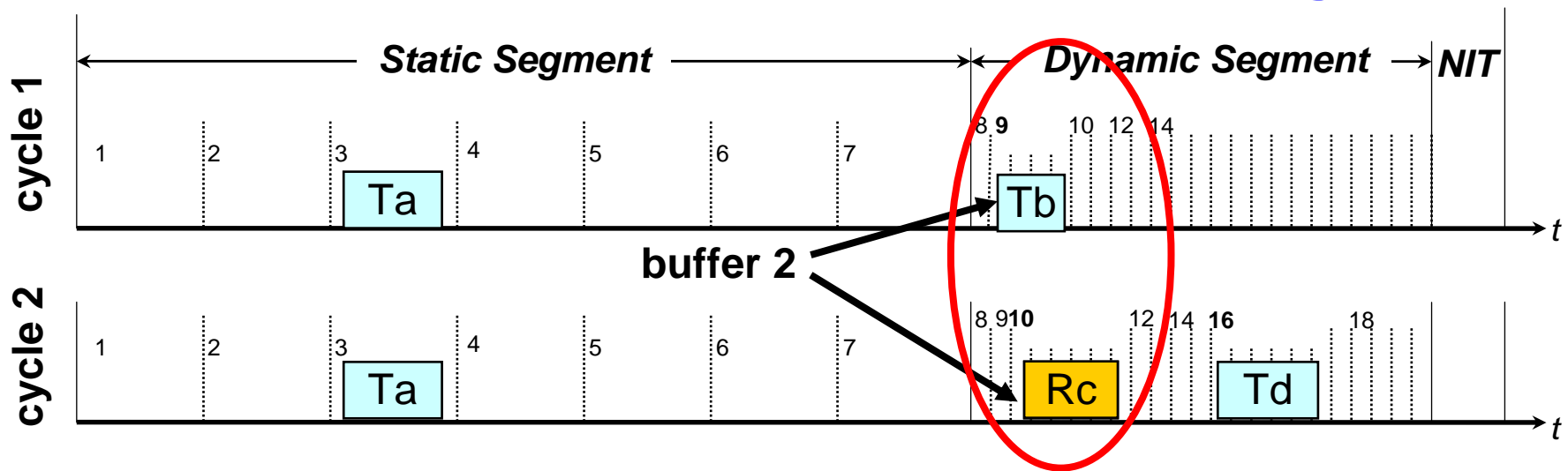


- not used for frame transmission
- write in POC:config or MB disabled (or MB locked in dynamic segment)

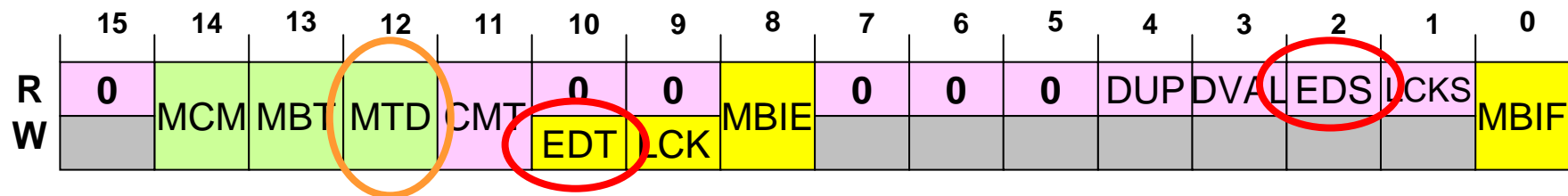
Message Buffer Structure



Buffer Reconfiguration



Message Buffer Configuration, Control, Status Registers (MBCCSRn)

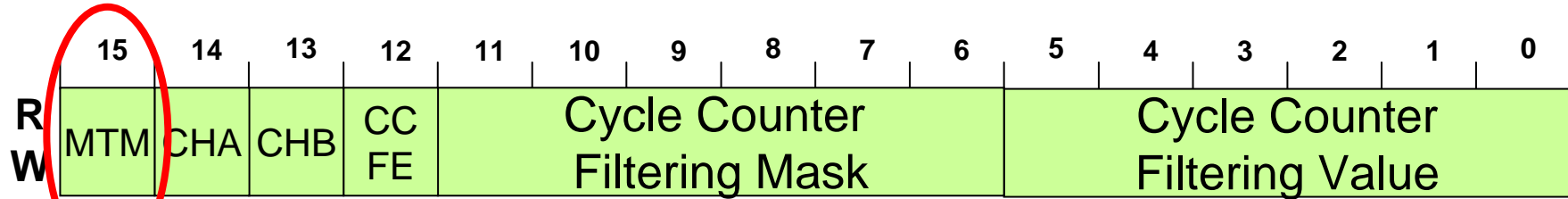


Message Buffer Frame ID Registers (MBFIDRn)



Event versus State Transmission

Message Buffer Cycle Counter Filtering Registers (MBCCFRn)



Message buffer Transmit Mode (event/state transmission)

Event Transmission (single shot mode):

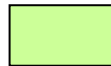
- The frame will be transmitted if the buffer was updated.

State Transmission (continous mode):

- The frame will be transmitted in every cycle.

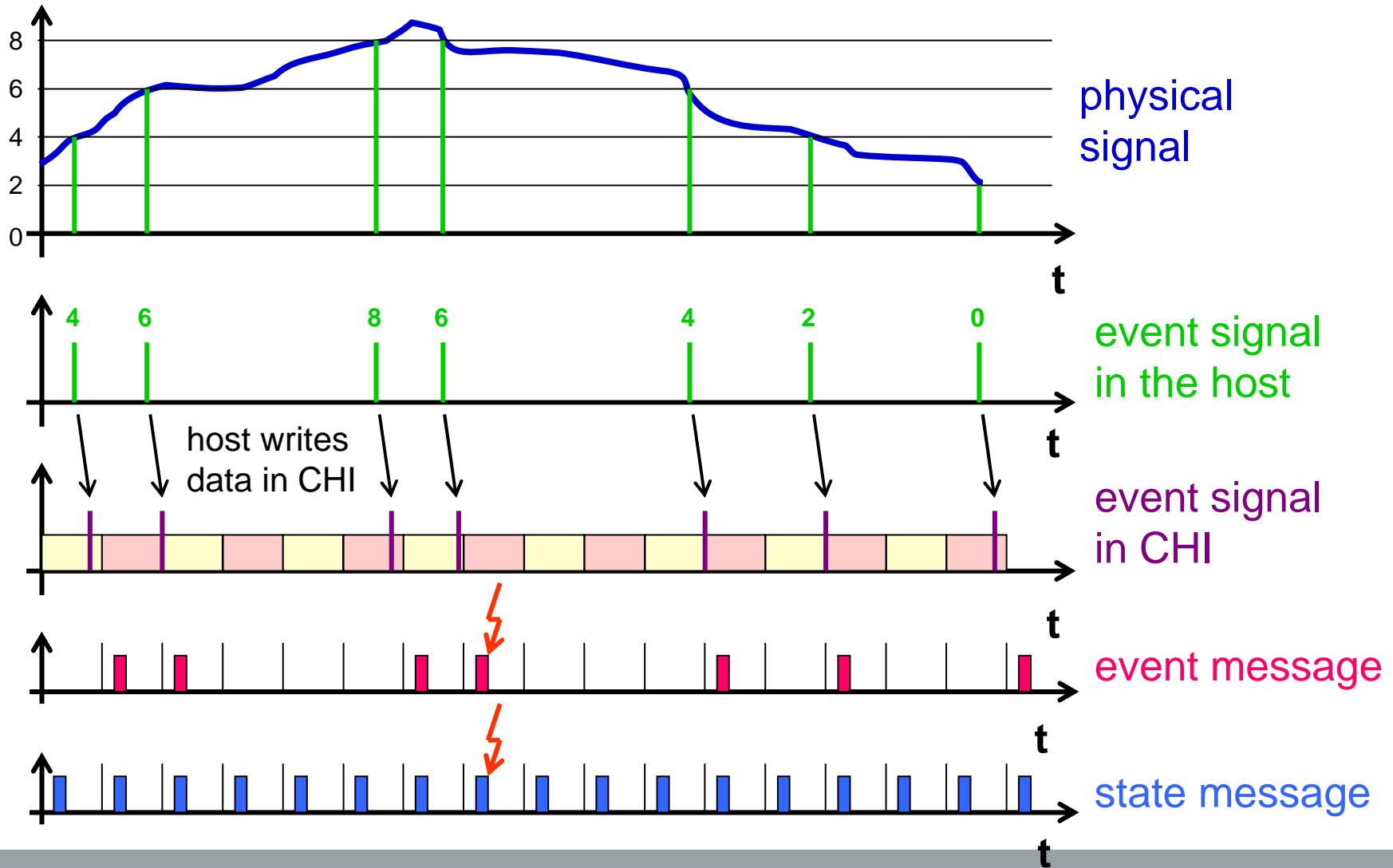


write in Normal Mode

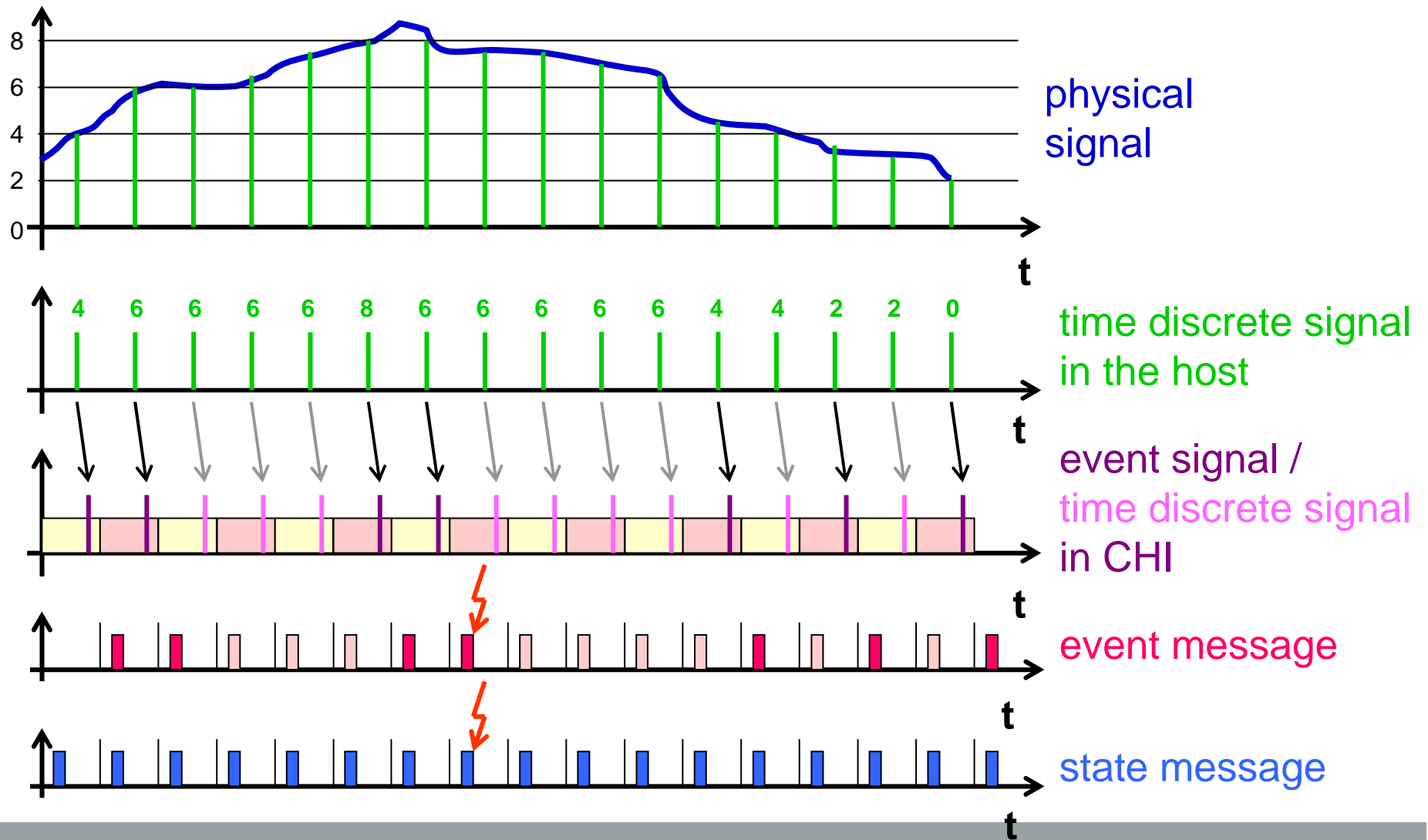


write in POC:config or MB disabled

Event versus State Transmission



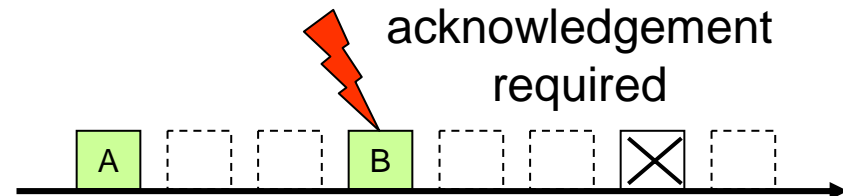
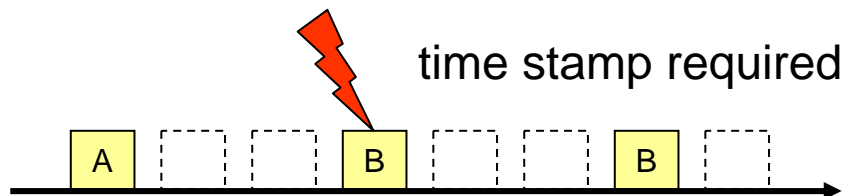
Event versus State Transmission



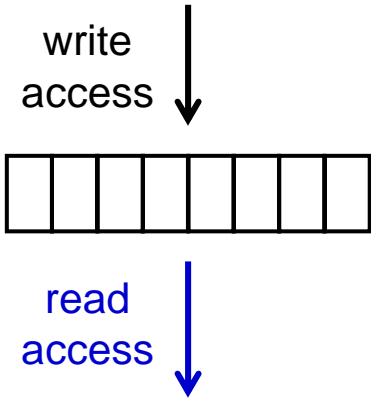
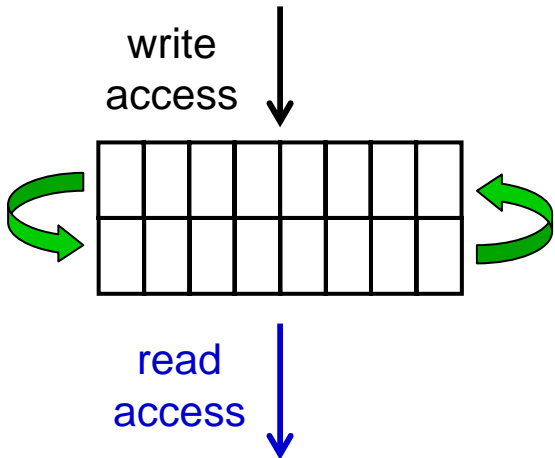
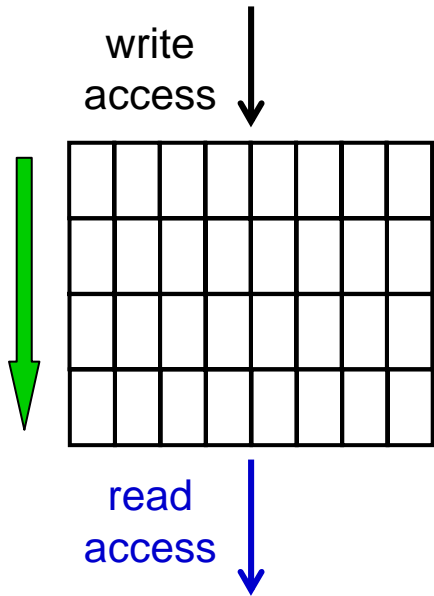
State Message versus Event Message Interface

State message interface
Message transmitted repeatedly
Lost message replaced by next copy
Resilient against loss / corruption of message on channel
Reception of message gives no indication of aliveness of sending task (autonomous communication)
→ Application level time stamping of message data required to eliminate state message data

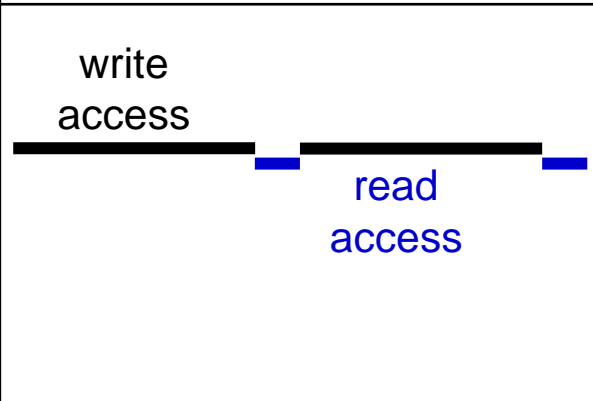
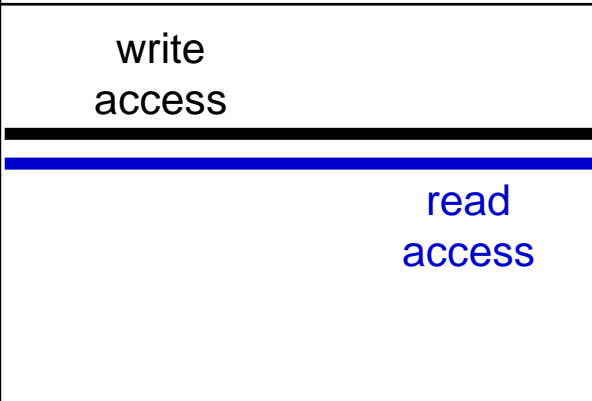
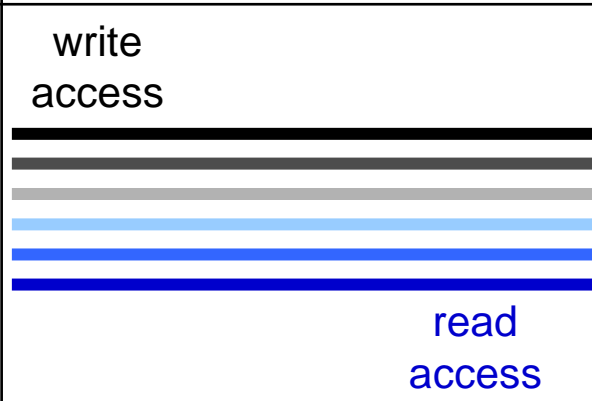
Event message interface
Message transmitted upon change
Lost message requires resend
Not resilient against loss / corruption of message on channel
Reception of message can be considered an aliveness indicator of sending task
→ Application level acknowledgement required to re-request lost message



Buffer Types

Single Buffer	Double Buffer	FIFO
		
exclusive read and write access	concurrent read and write access any time	decoupled read and write access
shared momory	queued memory	queued memory
messages not consumed	messages consumed	messages consumed

Buffer Types

Single Buffer	Double Buffer	FIFO
 <p>write access</p> <p>read access</p>	 <p>write access</p> <p>read access</p>	 <p>write access</p> <p>read access</p>
host must be synchronized with the FlexRay CC	host can run unsynchronized to the FlexRay CC	host can run unsynchronized to the FlexRay CC
direct access	doubled memory size required	very flexible and efficient but only queued read access
	Is a FIFO with a depth of two buffers	host has to „empty“ the FIFO buffer

Conclusion

- **Freescale has the largest portfolio of FlexRay communication controllers.**
- **An good buffer design leads to an efficient buffer usage.**
- **Powerful CHI features support the software design.**
- **The large number of CHI features require tool support that the user is able to find the best solution.**

