Hardware Complexity of Store-and-Forward, Cut-Through, and Preemption

Wolfram Lemppenau

University of Applied Science of Bocholt
Germany

Prof. Dr. Wolfram Lemppenau
email: **wle@bocholt.fh-ge.de**
phone: **+49-2871-2155 814**

Objective:

- **Better understanding how**

  store and forward - cut through - preemption

  influences hardware complexity

Strategy

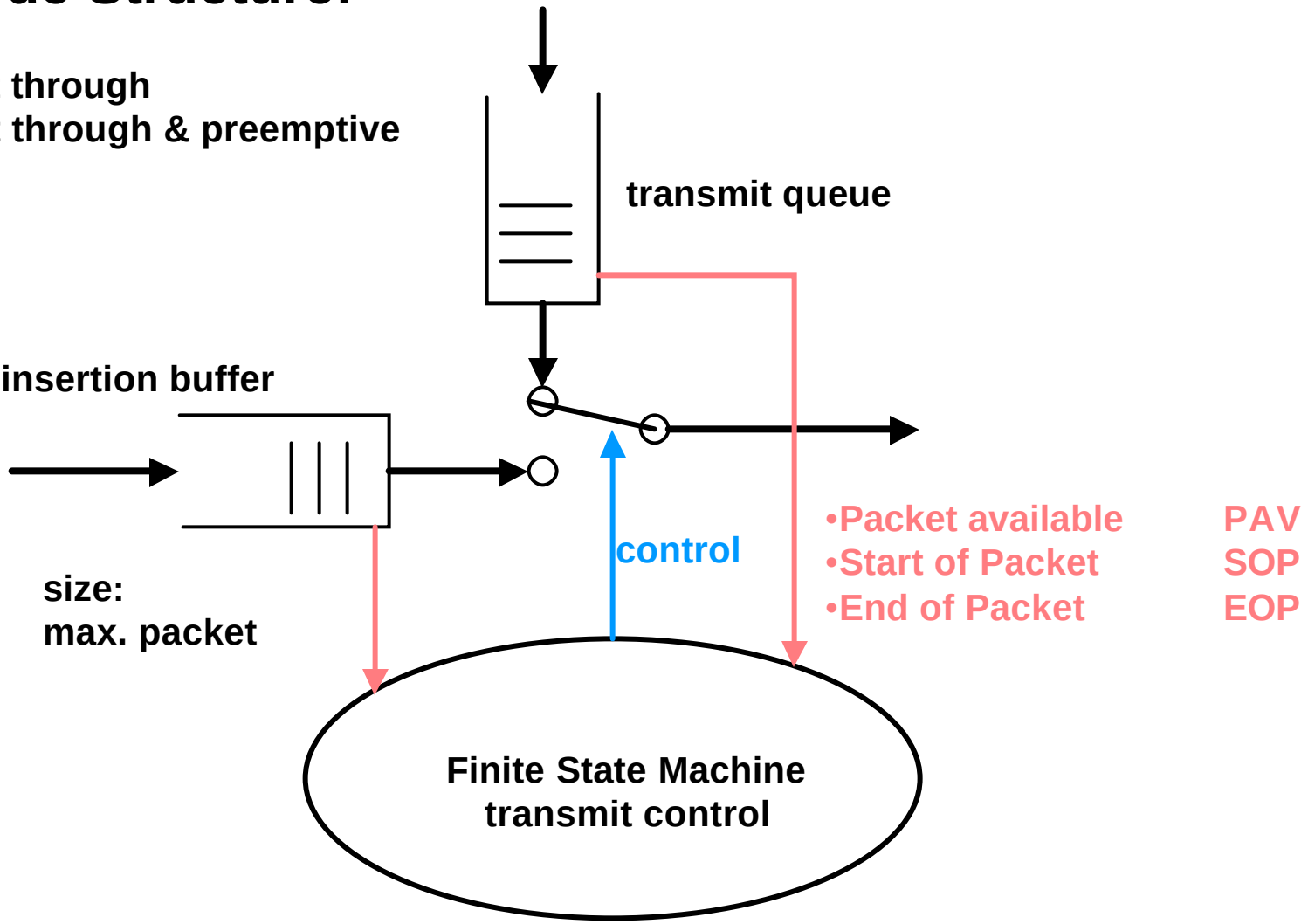
- **Examine hardware complexity based on real implementation**
- **Generic enough: parameters of interest dominate**
- **Detailed enough: realistic results**

It’s not

- **a product specific implementation**
- **MAC protocol dependent**

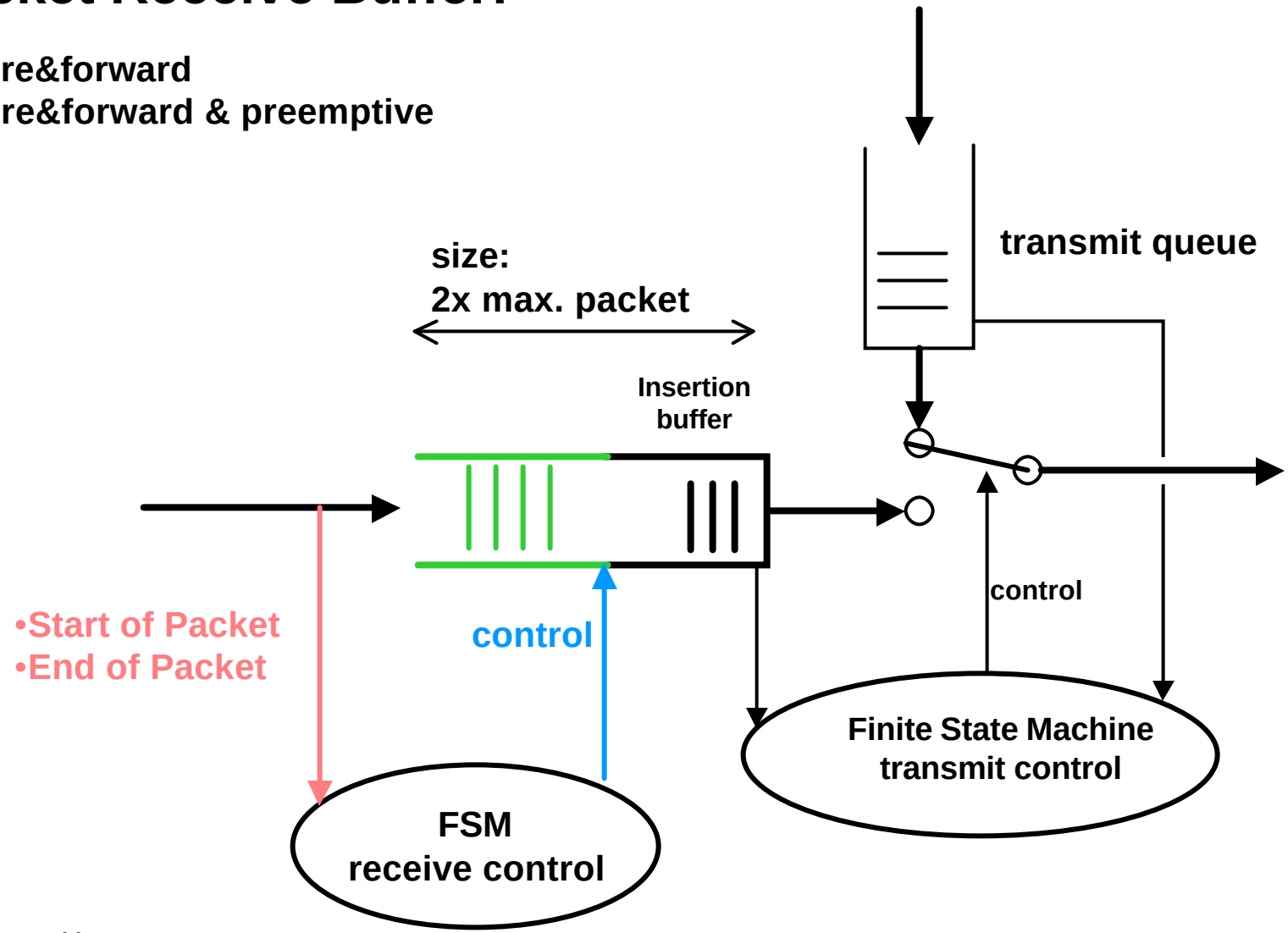
Node Structure:

- cut through
- cut through & preemptive



Packet Receive Buffer:

- store&forward
- store&forward & preemptive



FSM:
Finite State Machine

Considered Case Studies:

Case	Type	Priorities	Store&forward	Cut through	Preemptive	Error condition
A1/3	1/2	1	X	--	--	--
A2/4	1/2	1	X	--	--	X
B1/3	1/2	1	--	X	--	--
B2/4	1/2	1	--	X	--	X
C1/3	1/2	2	P1,P2	--	--	--
C2/4	1/2	2	P1,P2	--	--	X
C5/7	1/2	2	P1,P2	--	P1	--
C6/8	1/2	2	P1,P2	--	P1	X
D1/3	1/2	2	--	P1,P2	--	--
D2/4	1/2	2	--	P1,P2	--	X
D5/7	1/2	2	--	P1,P2	P1	--
D6/8	1/2	2	--	P1,P2	P1	X
E1	1	2	P2	P1	P1	--
E2	1	2	P2	P1	P1	X

Considered Case Studies:

Case	Type	Priorities	Store&forward	Cut through	Preemptive	Error condition
F1	1	3	P1,P2,P3	--	--	--
F2	1	3	P1,P2,P3	--	--	X
G1	1	3	--	P1,P2,P3	--	--
G2	1	3	--	P1,P2,P3	--	X
H1	1	3	P1,P2,P3	--	P1	--
H2	1	3	P1,P2,P3	--	P1	X
J1	1	3	P2,P3	P1	--	--
J2	1	3	P2,P3	P1	--	X
K1	1	3	P2,P3	P1	P1	--
K2	1	3	P2,P3	P1	P1	X

P1: Priority 1 (highest)

P2: Priority 2

P3: Priority 3 (lowest)

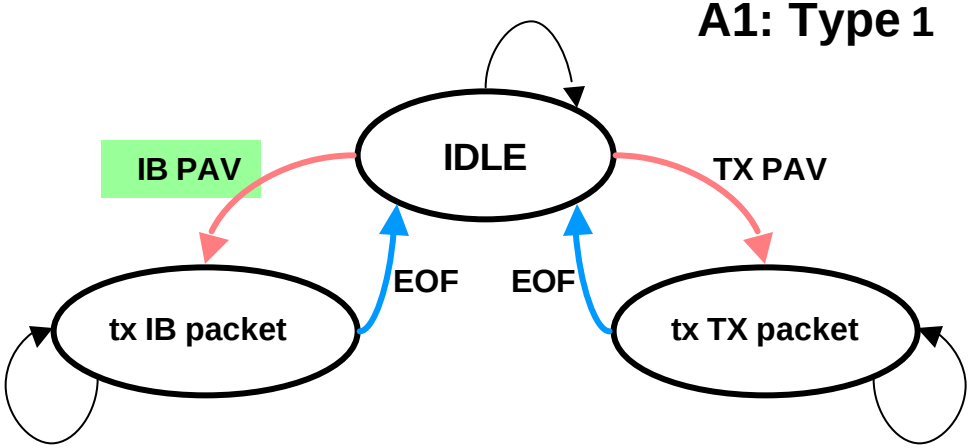
FSM Implementation Details:

- **Type 1 / 2 difference: only different FSM implementation behavior is the same**
- **All event combinations per state decoded**
- **Error condition: all unexpected (forbidden) events are treated as error, error-state entry enforced (ignored otherwise, remain in actual state)**
- **All unused “rest”-states (because of used state-encoding) enforce an error-state entry with error indication**
- **Binary state encoding, no optimization**
- **Synthesized and mapped (located) to FPGA (Xilinx) and CPLD (Lattice) architectures**
- **Optimized for speed/area unless otherwise noted**

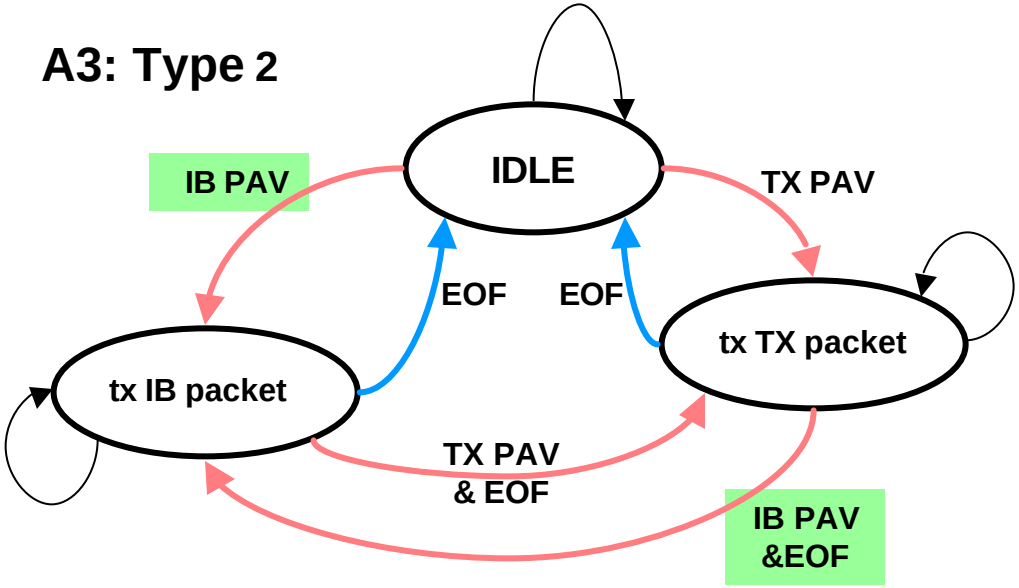
Case Studies A1, A3:

- one priority
- store&forward (non-preemptive)

A1: Type 1



A3: Type 2

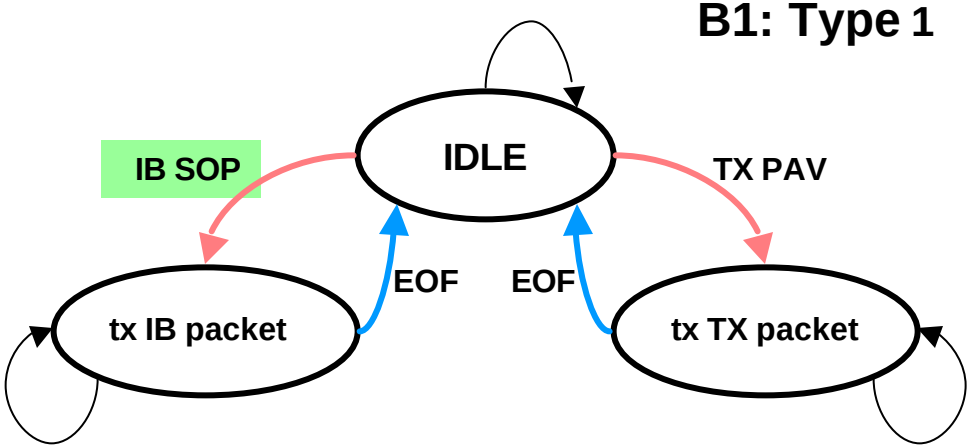


PAV: packet available
EOF: end of packet
TX: transmit buffer
IB: insertion buffer

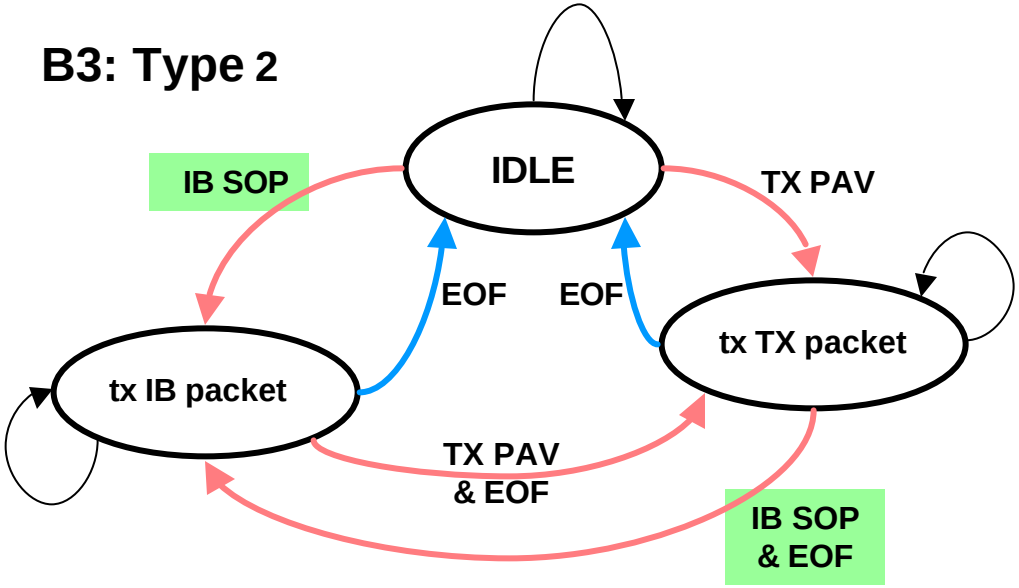
Case Studies B1, B3:

- one priority
- cut through (non-preemptive)

B1: Type 1



B3: Type 2

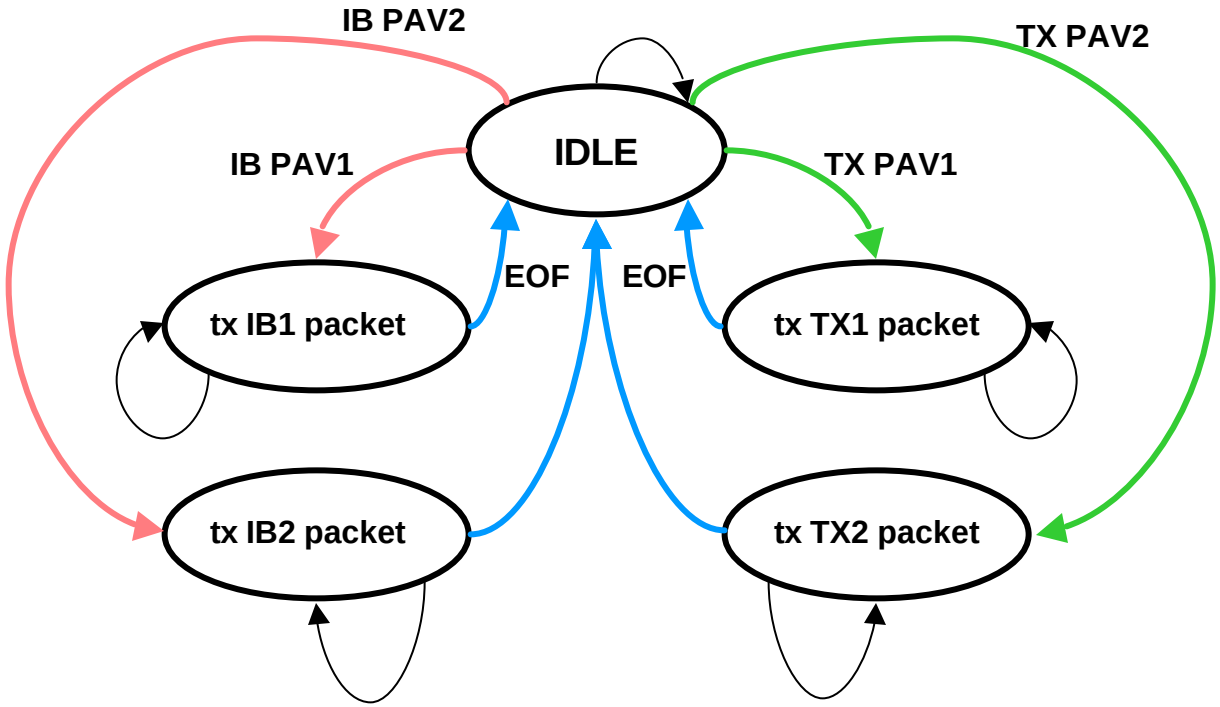


SOF: start of packet
EOF: end of packet
TX: transmit buffer
IB: insertion buffer
tx: transmit

Case Studies C1, C3:

- two priorities
- store&forward
- non-preemptive

C1: Type 1



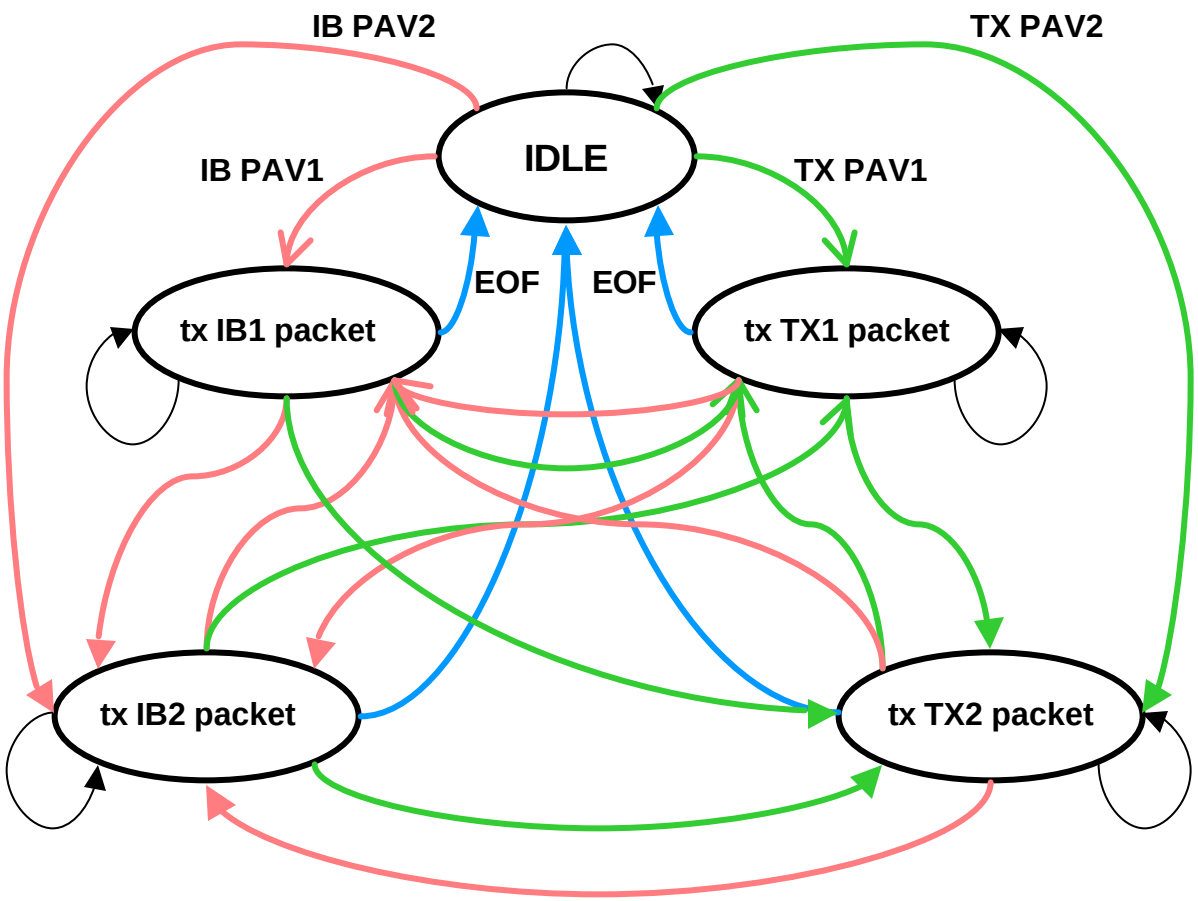
PAV: start of packet
EOF: end of packet
TX: transmit buffer
IB: insertion buffer

...1: priority 1
...2: priority 2
tx: transmit

Case Studies C1, C3:

C3: Type 2

- two priorities
- store&forward
- non-preemptive

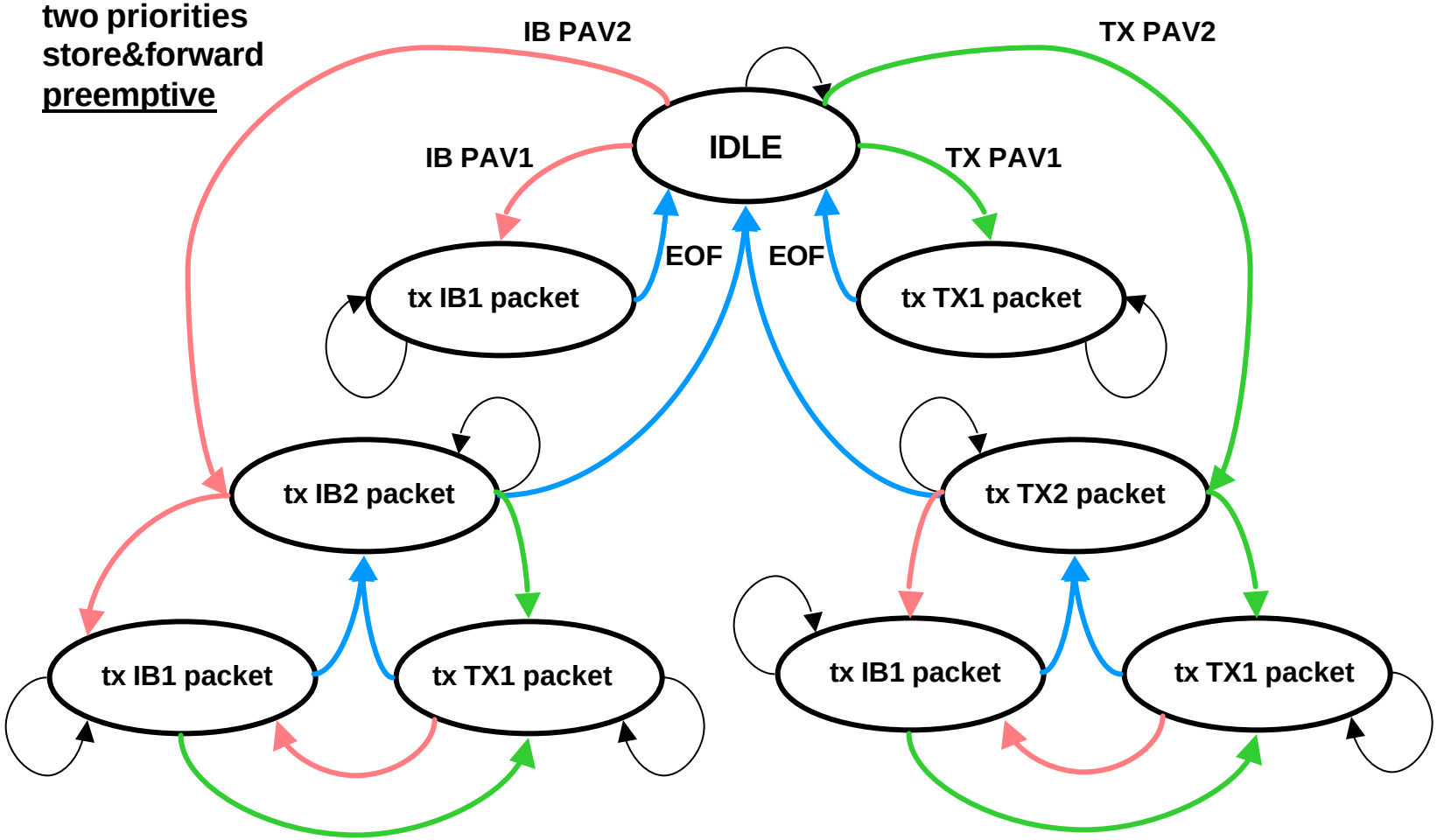


C1, C3:
same behavior in controlling outgoing packets

Case Study C5:

C5: Type 1

- two priorities
- store&forward
- preemptive



- D5: two priorities, cut through, preemtive almost identical

Results:

Case	Type	Priorities	S&f	Cut through	Preemptive	Error condition	FPGA
A1	1	1	X	--	--	--	4
A3	1	1	X	--	--	X	10
A2	2	1	X	--	--	--	6
B1	1	1	--	X	--	--	4
B2	2	1	--	X	--	--	6
C1	1	2	P1,P2	--	--	--	11
C3	2	2	P1,P2	--	--	--	22
C5	1	2	P1,P2	--	P1	--	29
D1	1	2	--	P1,P2	--	--	13
D3	2	2	--	P1,P2	--	--	22
D5	1	2	--	P1,P2	P1	--	31
D7	2	2	--	P1,P2	P1	--	37
E1	1	2	P2	P1	P1	--	31

to be completed