Physical Layer Design and Analysis of Network Centric Cognitive Radio (WiNC2R)

By Tejaswy Hari

Thesis Director Prof. Narayan Mandayam

Committee Members Prof. Predrag Spasojevic

Prof. Yangyong Zhang

Prof. Zoran Miljanic

RUTGERS

WINLAB

Outline

1. WiNC2R
   Virtual Flow Pipelining
2. Top Level Architecture Design
   Transmit and Receive Command flow for 802.11a-Lite
3. Processing Engines
   I. Modulator
   II. Demodulator
   III. Checker
   IV. Verification
4. Software on WiNC2R
5. Timing Analysis
   Parallel Implementation Design and Analysis
6. Conclusion and Future Work

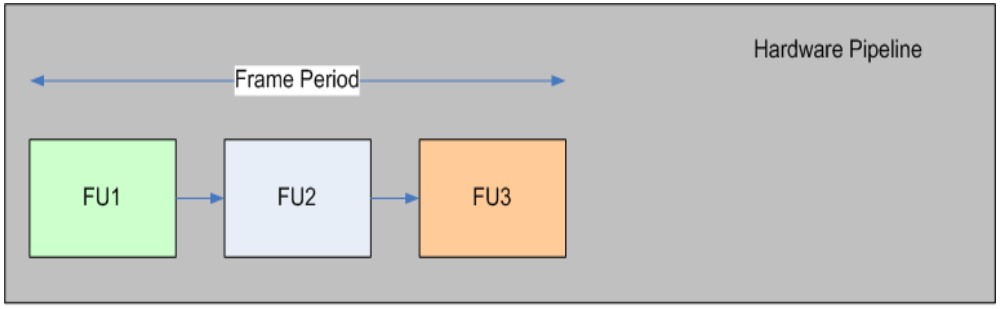
Outline

1. **WiNC2R**
   Virtual Flow Pipelining
2. Top Level Architecture Design
   Transmit and Receive Command flow for 802.11a-Lite
3. Processing Engines
   I. Modulator
   II. Demodulator
   III. Checker
   IV. Verification
4. Software on WiNC2R
5. Timing Analysis
   Parallel Implementation Design and Analysis
6. Conclusion and Future Work

WiNC2R – A Cognitive Radio Platform

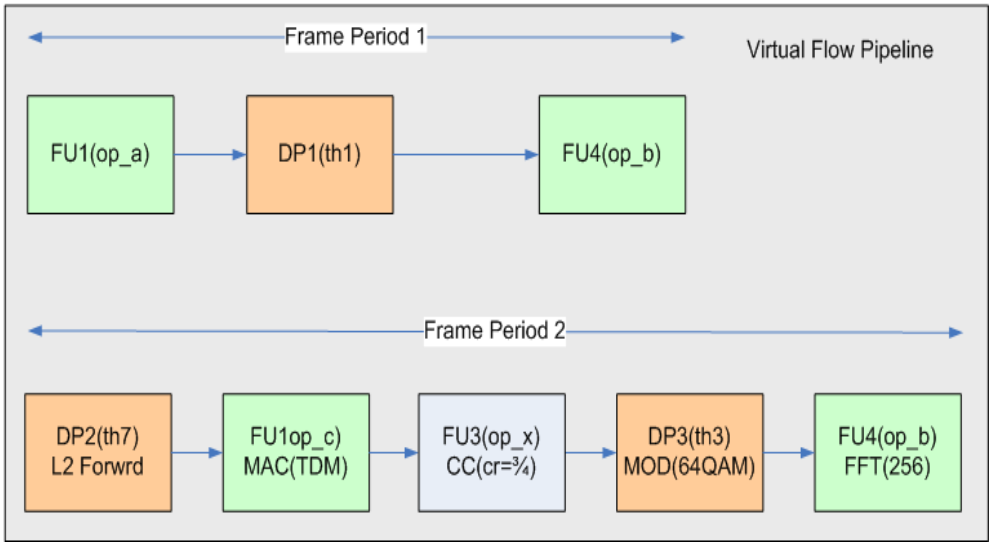
- Programmability and Reconfigurability
  - WiNC2R implements the radio functionality using various hardware accelerators, which are programmable by software
- Interoperability
  - Ability to shift between protocols quickly
  - Independent Signal processing applications
- Scalability
  - Add or Remove data processing modules dynamically
  - Functional Units with Plug and Play feature
- Implementation of complex algorithms
  - Capabilities of the cognitive radio platform algorithms – spectrum allocation, SNR based adaptation, per packet protocol adaptation

Virtual Flow Pipelining (VFP)



Hardware Pipeline

- Fixed order of FU activation
- Fixed operation each stage
- Single flow with fixed size time frame
- No software based processing in pipeline



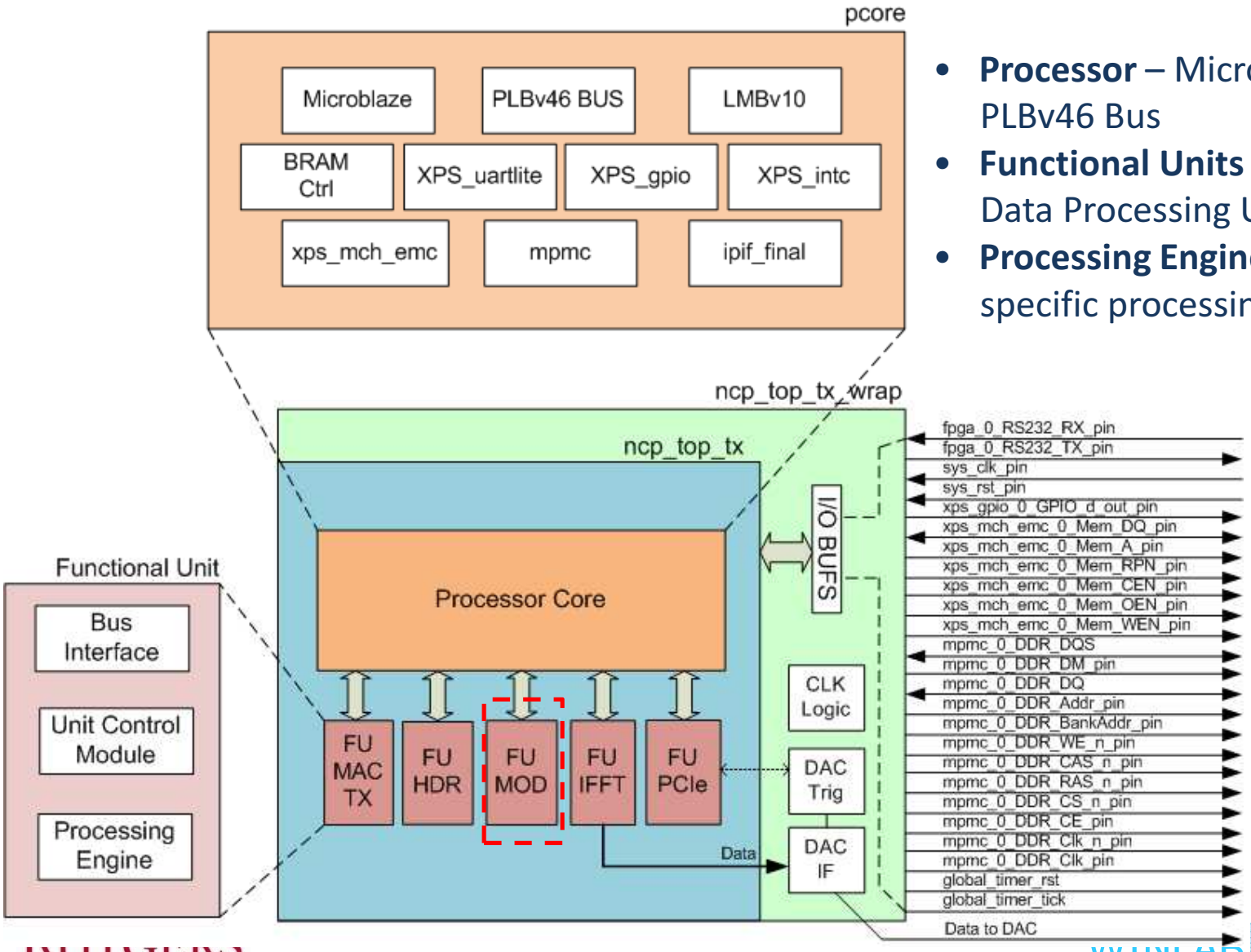
Virtual Flow Pipelining

- Programmable order of FU activations in each flow
- Programmable selection of FU operation in each stage
- Coexistence of multiple flows
- Programmable time frame for each flow
- Any pipeline stage can be software based processing

Outline

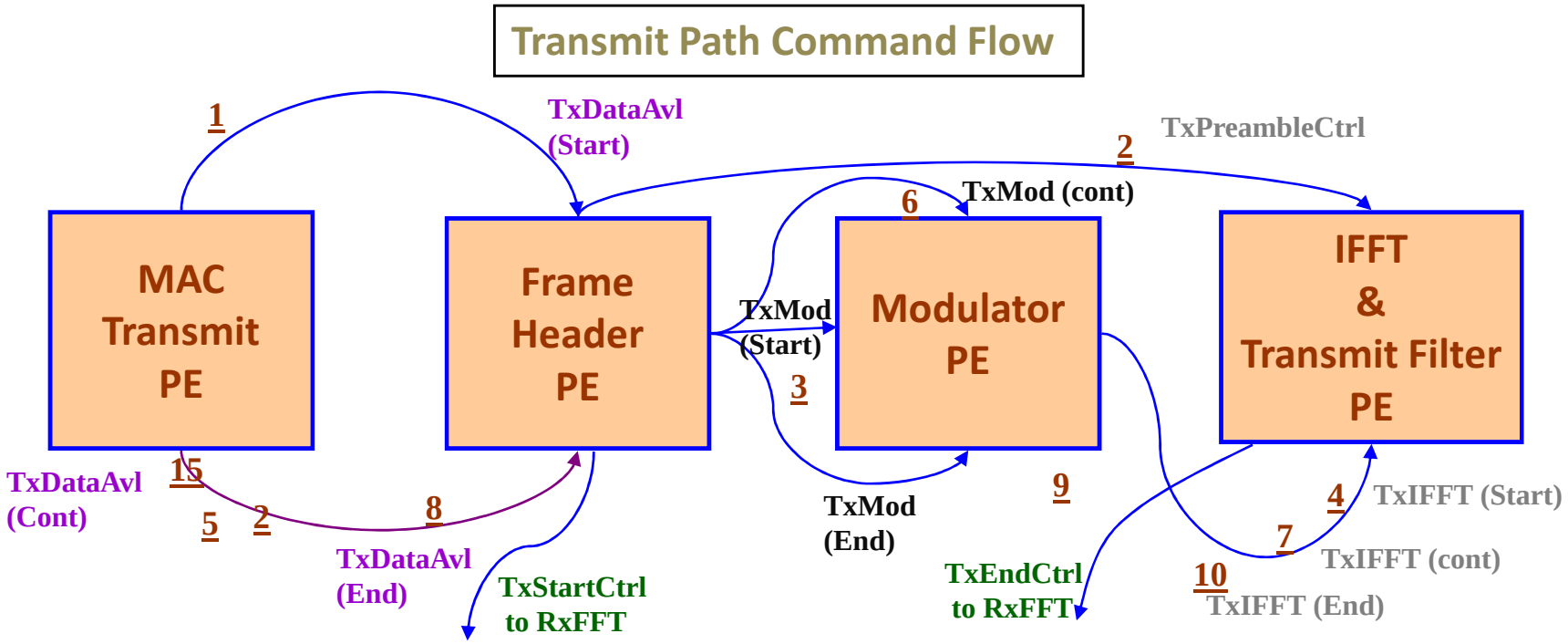
1. WiNC2R
   Virtual Flow Pipelining
2. **Top Level Architecture Design**
   Transmit and Receive Command flow for 802.11a-Lite
3. Processing Engines
   I. Modulator
   II. Demodulator
   III. Checker
   IV. Verification
4. Software on WiNC2R
5. Timing Analysis
   Parallel Implementation Design and Analysis
6. Conclusion and Future Work

Top Level Architecture - Tx



- **Processor** – Microblaze and PLBv46 Bus
- **Functional Units** – Low level Data Processing Units
- **Processing Engine** – Application specific processing

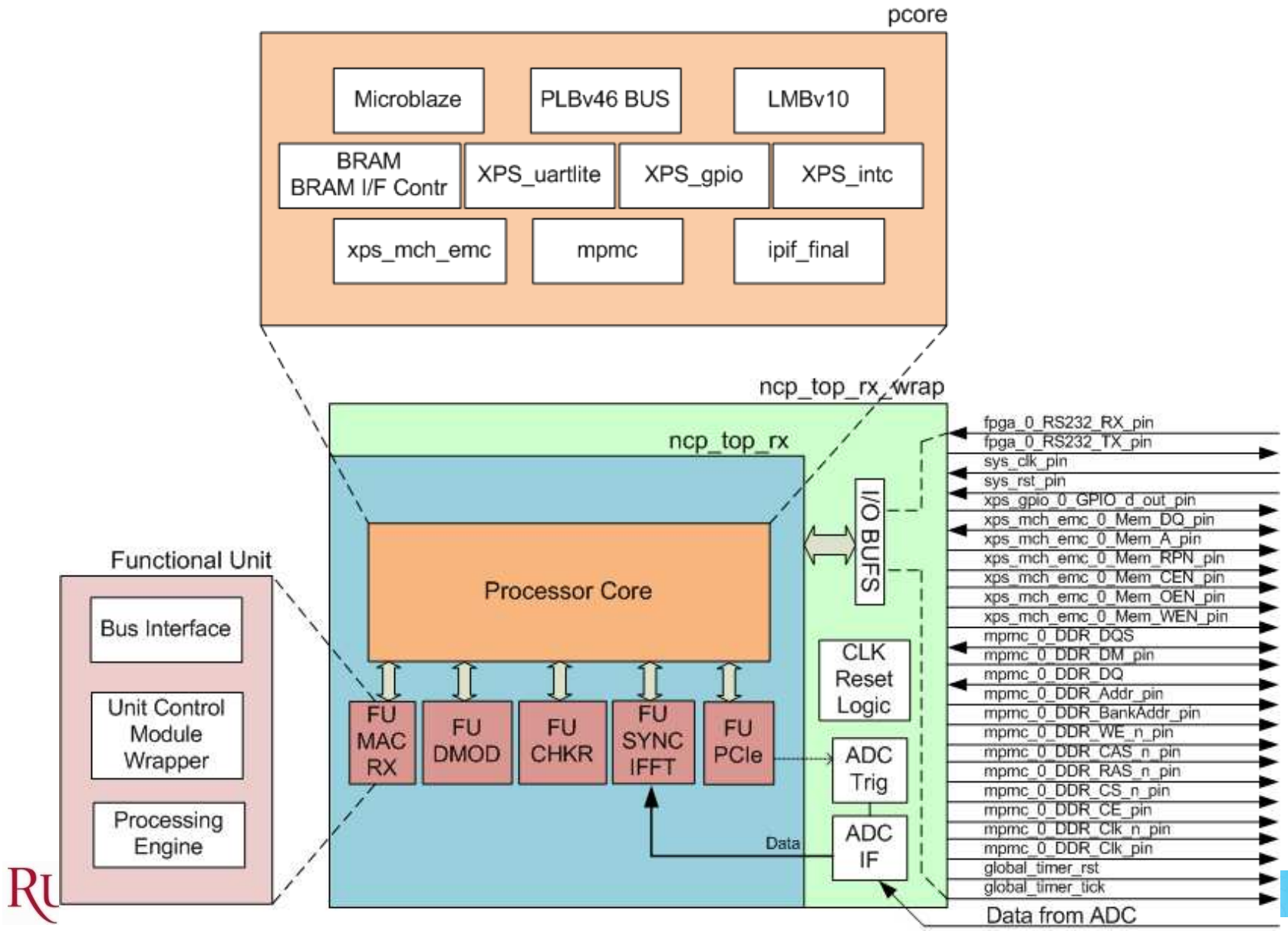
VFP for 802.11a-Lite Transmitter



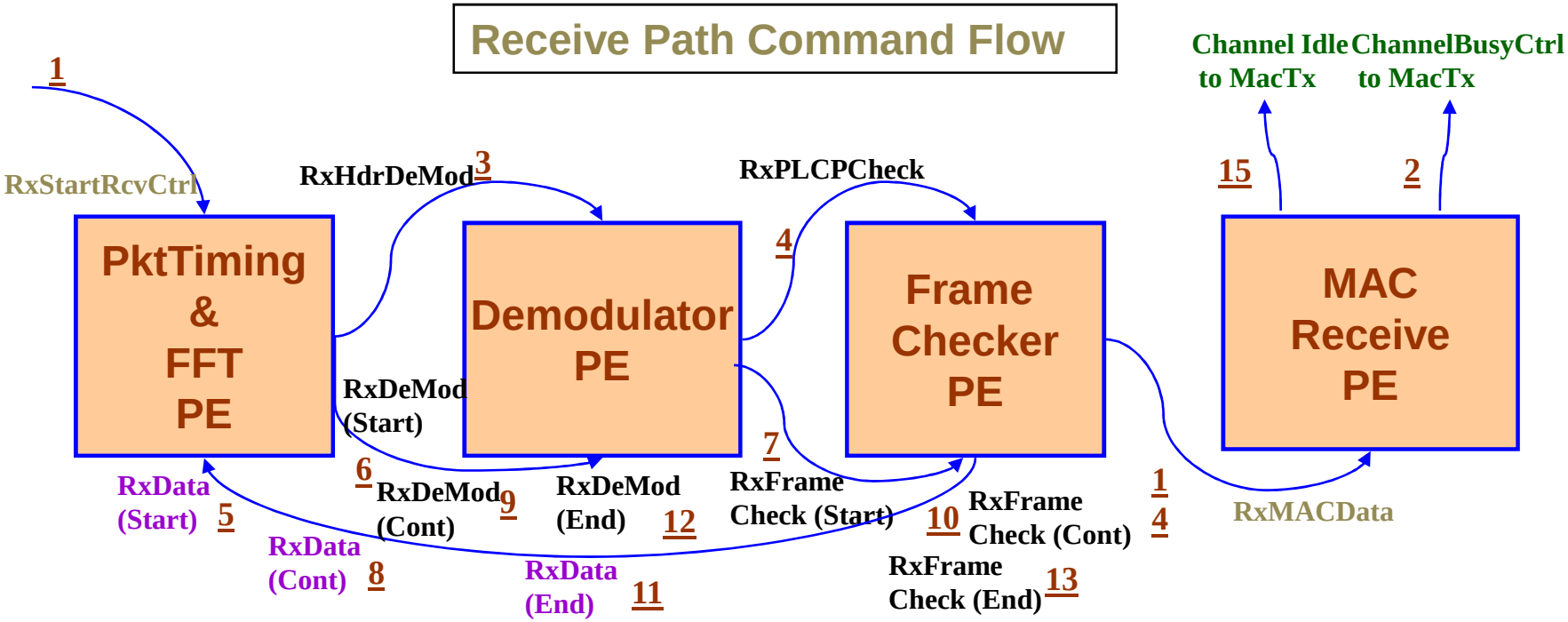
Transmitter Flow

- MAC Tx triggers data flow
- Frame Header attaches header information and forwards data to Modulator
- Modulated output goes into IFFT and FILTER blocks
- Output of FFT is sent to DAC

Top Level Architecture - Rx



VFP for 802.11a-Lite Receiver



Receiver Flow

- ADC output goes to Synchronizer and FFT
- Header output is Demodulated and CHECKER processes header information and sends it back
- FFT triggers data flow to DEMODULATOR
- CHECKER monitors CRC for each frame
- MAC Rx extracts data from the frame

Tasks for Modulator and Demodulator

TxMod

Modulate the buffer. Three different sub-tasks - first chunk, last chunk and the intermediate chunk

RxHdrDmod

Demodulate the PLCP header (802.11a header).

RxPLCPChk

The frame checker checks the parity of the header and extracts the frame parameters from it.

RxDeMod

Demodulate the frame in the input buffer.

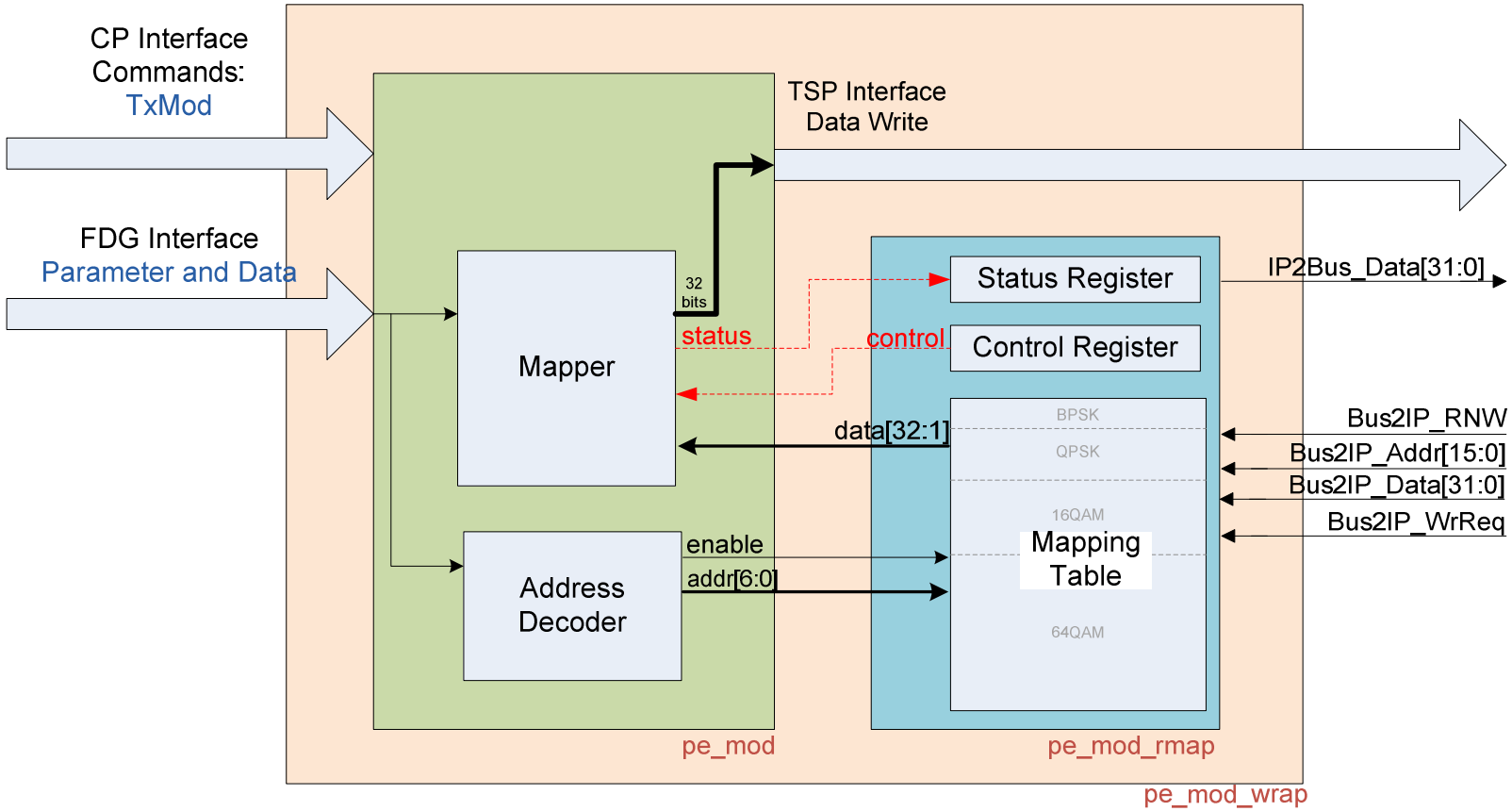
RxFrameChk

Calculate the CRC of the frame and forward to the MAC. Three different sub-tasks - first chunk, last chunk and the intermediate chunk

Outline

1. WiNC2R
   Virtual Flow Pipelining
2. Top Level Architecture Design
   Transmit and Receive Command flow for 802.11a-Lite
3. **Processing Engines**
   I. **Modulator**
   II. **Demodulator**
   III. **Checker**
   IV. **Verification**
4. Software on WiNC2R
5. Timing Analysis
   Parallel Implementation Design and Analysis
6. Conclusion and Future Work

PE - Modulator



- **Address Decoder** – Interprets input bits and generates address for mapping table
- **Mapping Table** – Contains constellation configuration
- **Mapper** – Forwards Modulated bit to the output buffer
- **Status and Control Register** – Setup and Debugging

Command Parameters

- Command Parameters – Contain frame properties
- Shared by all processing engines
- Every data chunk processed independently

b_{31}	b_{30}	b_{29}-b_{28}	b_{27} - b_{23}	b_{22}	b_{21}- b_{20}	b_{19}	b_{18}
Preamble Present	Not End of Burst	Midamble Interval	Subchannelization	Short/Long Preamble	Channel ID	No Coder	Uplink/ Downlink

b_{17}-b_{16}	b_{15}-b_{11}	b_{10}-b_{9}	b_{8}-b_{7}	b_{6}	b_{5}-b_{4}	b_{3}-b_{2}	b_{1}-b_{0}
Standard ID	Header Bytes	Header Code Rate	Header Frame Modln	Header Present	Code Rate	Frame Modln	Frame Status Bits

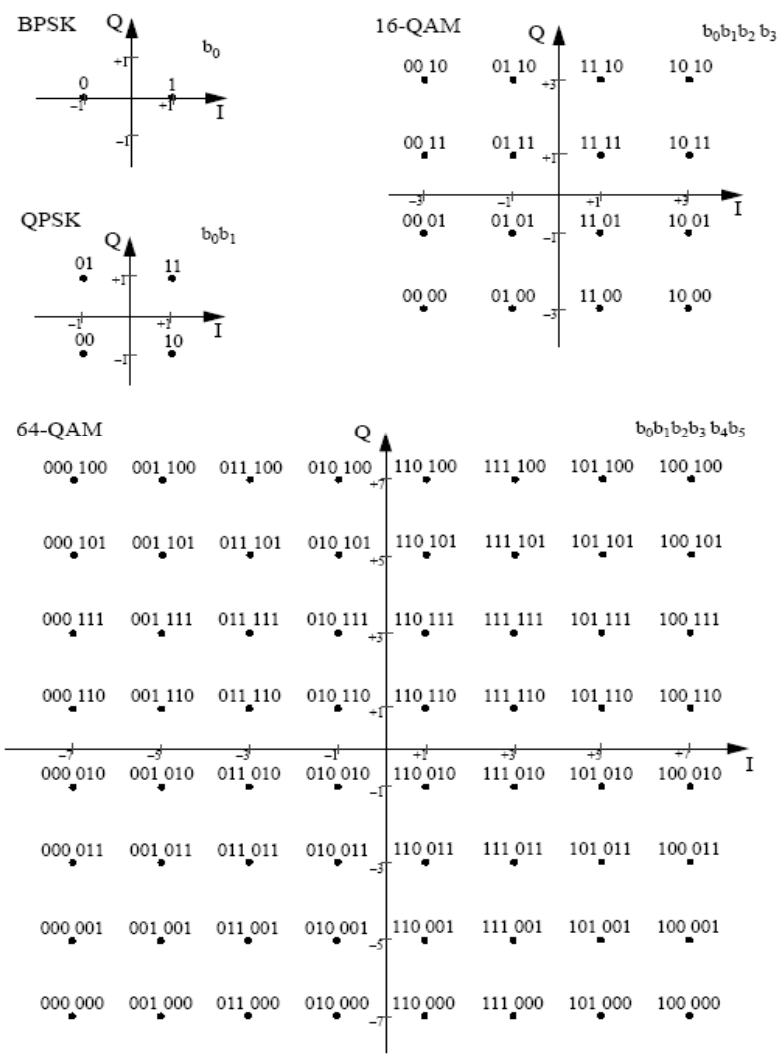
Standard ID	Description
00	WiFi
01	WiMax

Code Rate	Code
1/2	00
3/4	01
2/3	10
5/6	11

Modulation	Code
BPSK	00
QPSK	01
16-QAM	10
64-QAM	11

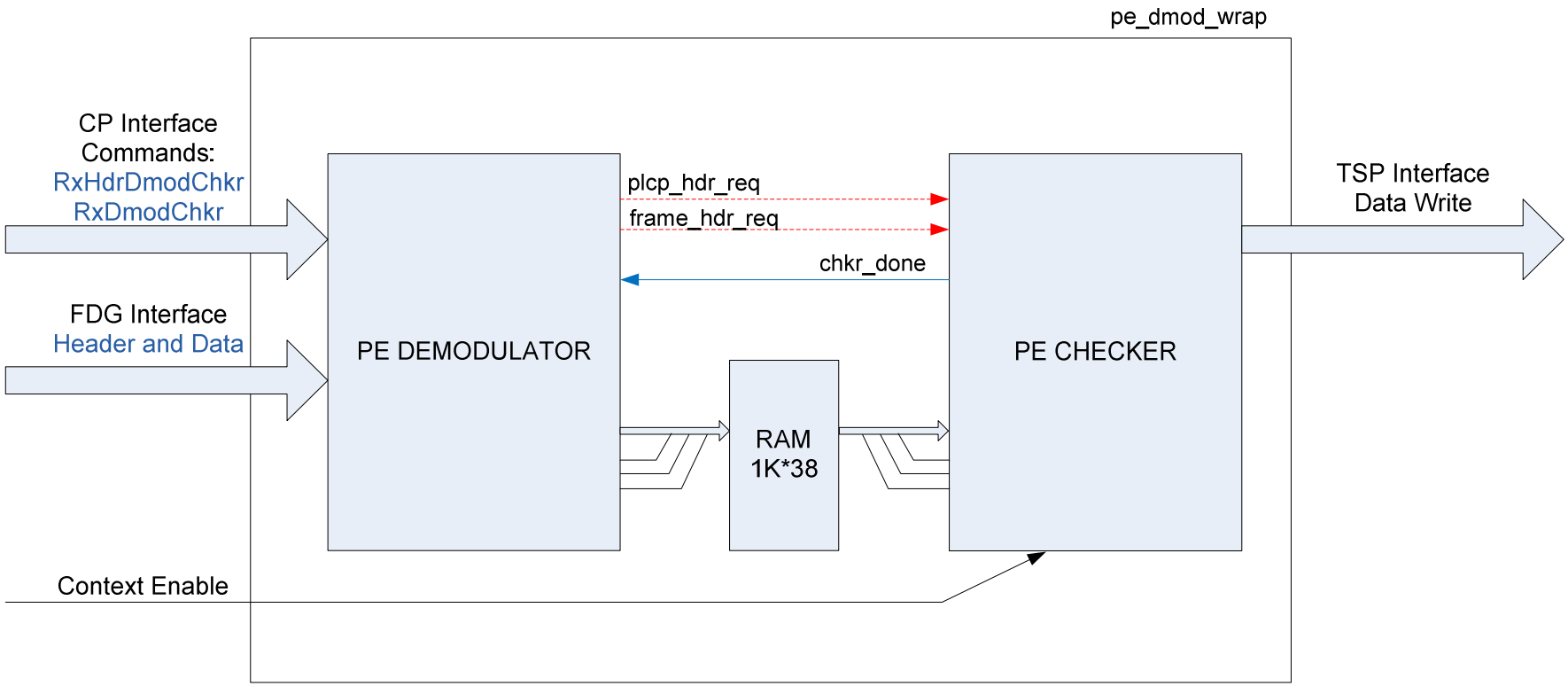
Frame Status	Part of Frame
00	Start and End
01	Start
10	Continue
11	End

Register Maps & Mapping Table



Addr Offset	WiFi	Addr Offset	WiMAX
00	2 points BPSK	86	2 points BPSK
02	4 points QPSK	88	4 points QPSK
06	16 points 16QAM	92	16 points 16QAM
22	64 points 64QAM	108	64 points 64QAM

PE – Demodulator and Checker



- Demodulator – Demodulates the input data
- Checker – Header Parsing and CRC check

Command Parameters

RxDeModChk Command Parameters

Command Code: 0x02

b_{31}-b_{28}	b_{27}-b_{23}	b_{22}	b_{21}-b_{20}	b_{19}	b_{18}
0x0	Subchannelization	Short/Long Preamble	Channel ID	No Coder	Uplink/ Downlink

b_{17}-b_{16}	b_{15}-b_{6}	b_{5}-b_{4}	b_{3}-b_{2}	b_{1}-b_{0}
Standard ID	0x0	Code Rate	Frame Modln	Frame Status Bits

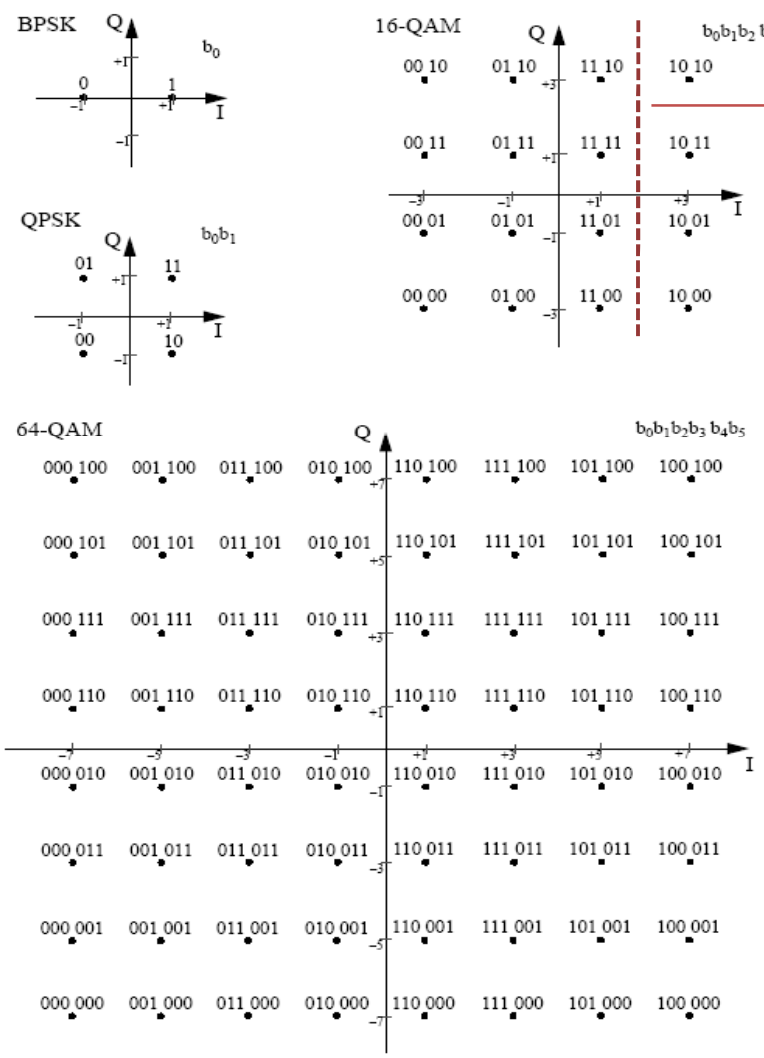
Standard ID	Description
00	WiFi
01	WiMax

Code Rate	Code
½	00
3/4	01
2/3	10
5/6	11

Modulation	Code
BPSK	00
QPSK	01
16-QAM	10
64-QAM	11

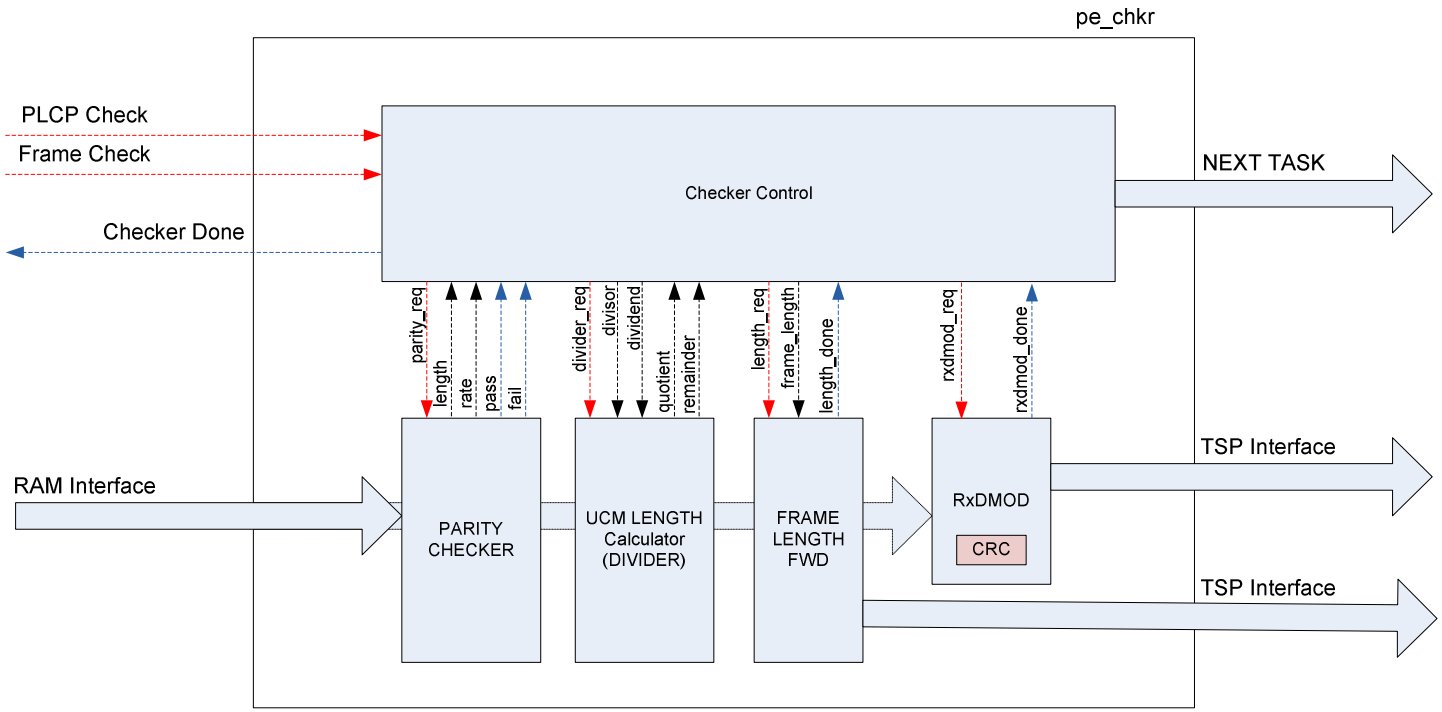
Frame Status Bits	Description
00	Start and End
01	Start
10	Continue
11	End

Decision Table



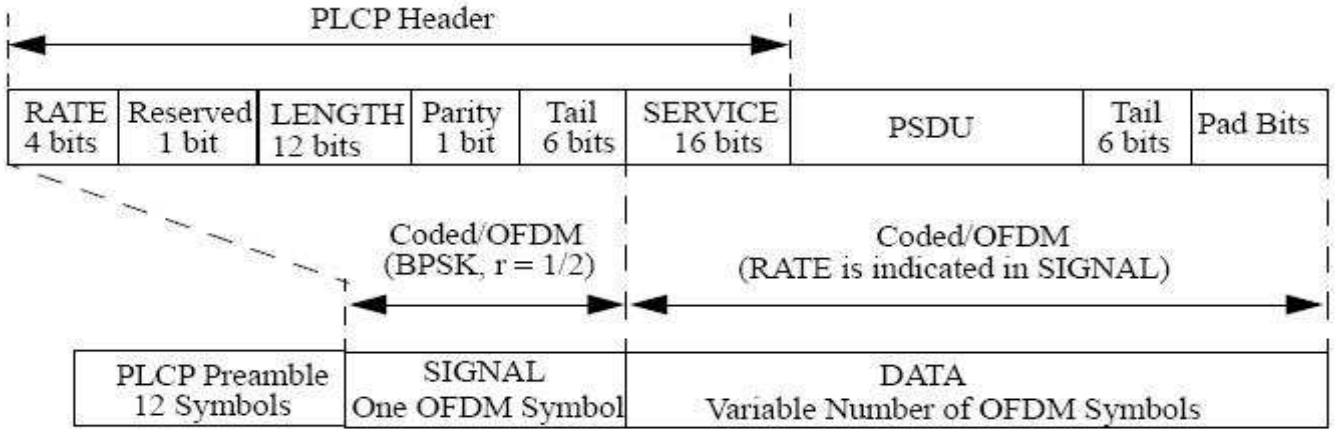
Depth	Boundary
2	2 points BPSK
4	4 points QPSK
8	16 points 16QAM
16	64 points 64QAM

PE Checker



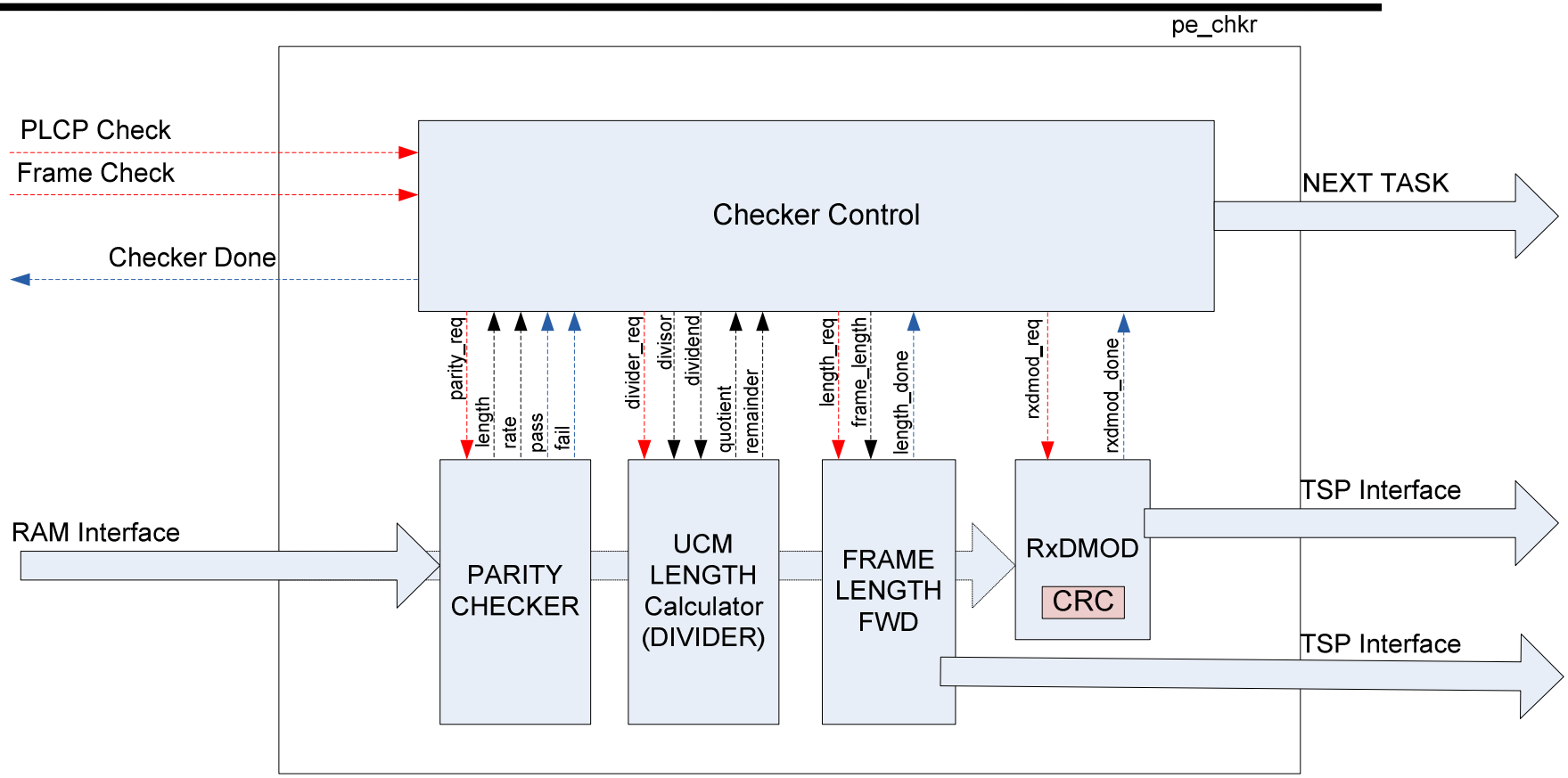
- **Parity Checker** – Checks parity of the header and parses it.

PLCP Header



- Sync triggers Demodulator after Preamble
- After demodulation, Header Contents are extracted
- Rate and Length obtained

PE Checker

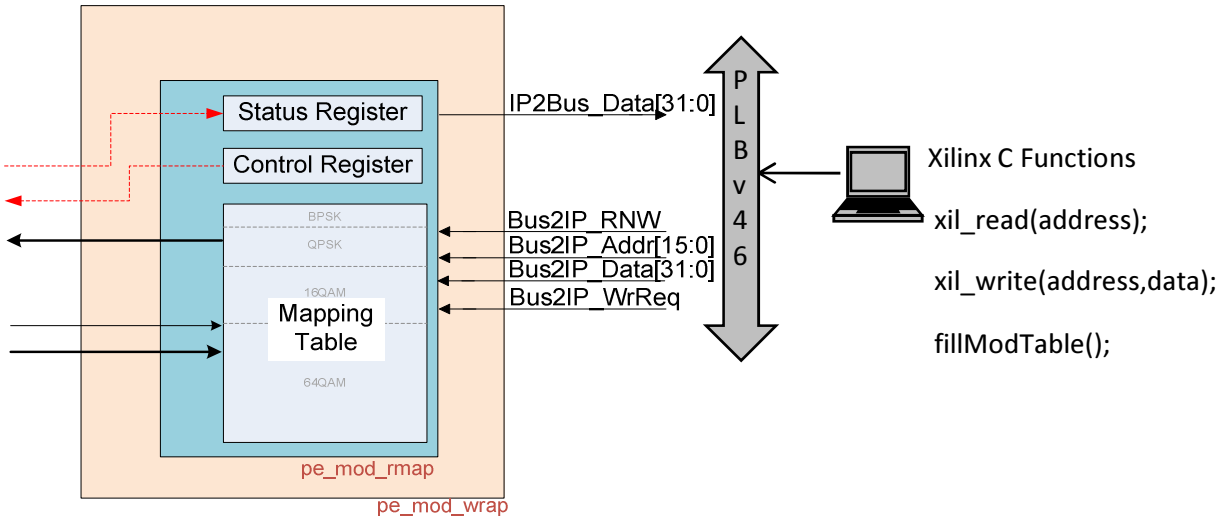


- **Parity Checker** – Checks parity of the header and parses it.
- **Length Calculator** – Calculates no. of OFDM symbols
- **Frame Length Fwd** – Forwards frame length and no. OFDM symbols to sync
- **RxDMOD** – Calculates CRC and writes to output buffer
- Ready for next module

Outline

1. WiNC2R
   Virtual Flow Pipelining
2. Top Level Architecture Design
   Transmit and Receive Command flow for 802.11a-Lite
3. Processing Engines
   I. Modulator
   II. Demodulator
   III. Checker
   IV. Verification
4. **Software on WiNC2R**
5. Timing Analysis
   Parallel Implementation Design and Analysis
6. Conclusion and Future Work

Software Setup of WiNC2R

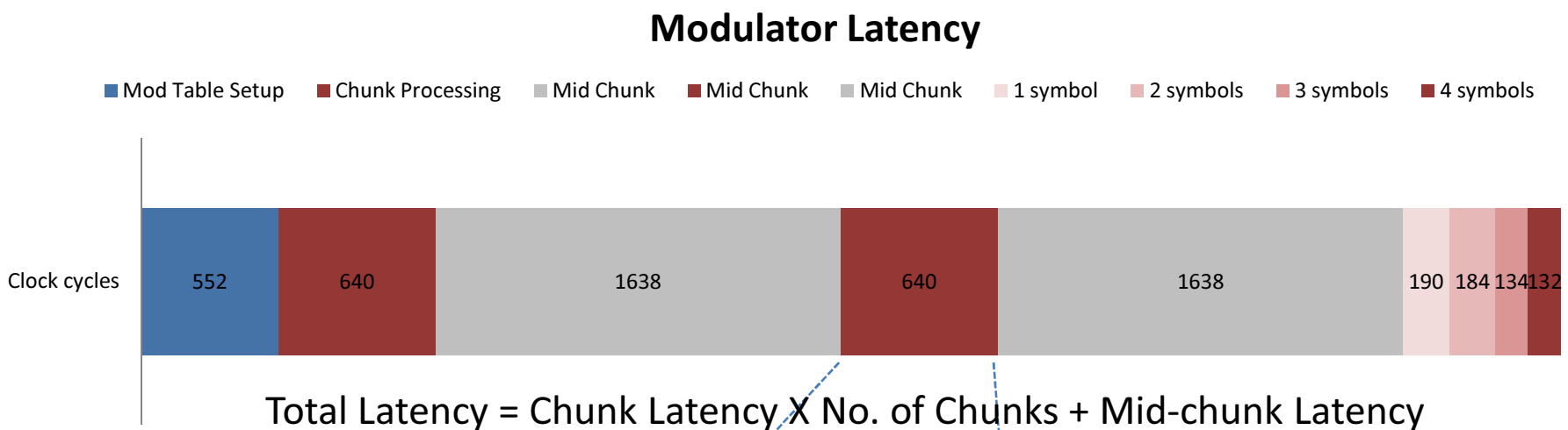


- Sets up the VFP – Task flows
- Writes data to control regions – to configure processing engines
- Calculates constellation points and decision regions
- Sets up DAC and ADC
- Cognitive Algorithms implemented here

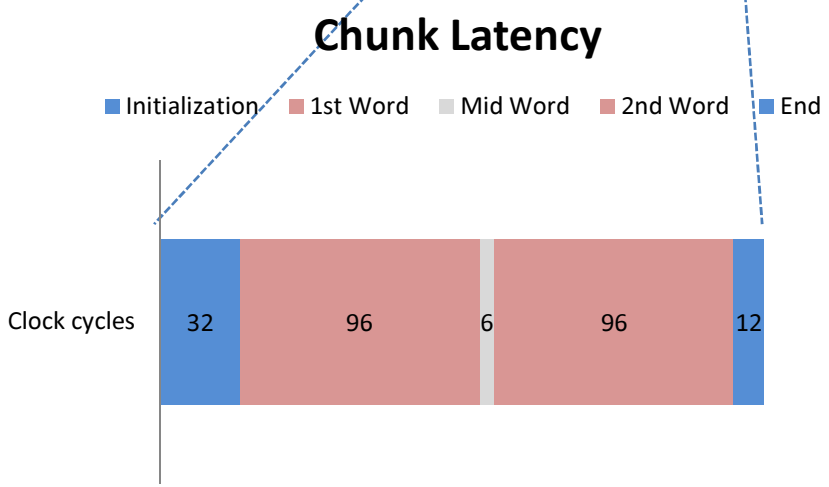
Outline

1. WiNC2R
   Virtual Flow Pipelining
2. Top Level Architecture Design
   Transmit and Receive Command flow for 802.11a-Lite
3. Processing Engines
   I. Modulator
   II. Demodulator
   III. Checker
   IV. Verification
4. Software on WiNC2R
5. **Timing Analysis**
   Parallel Implementation Design and Analysis
6. Conclusion and Future Work

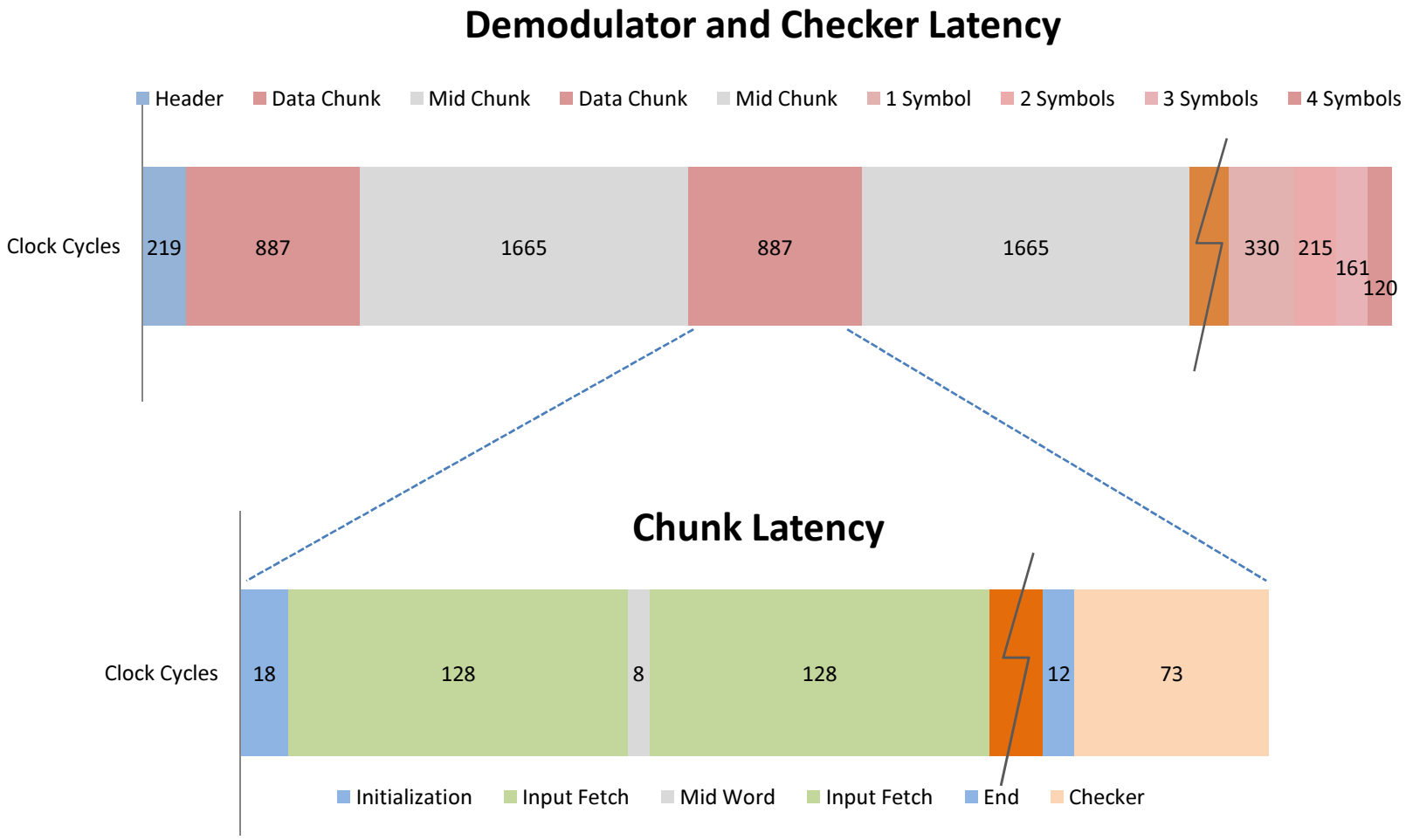
Modulator Latency



Total Latency = Chunk Latency X No. of Chunks + Mid-chunk Latency



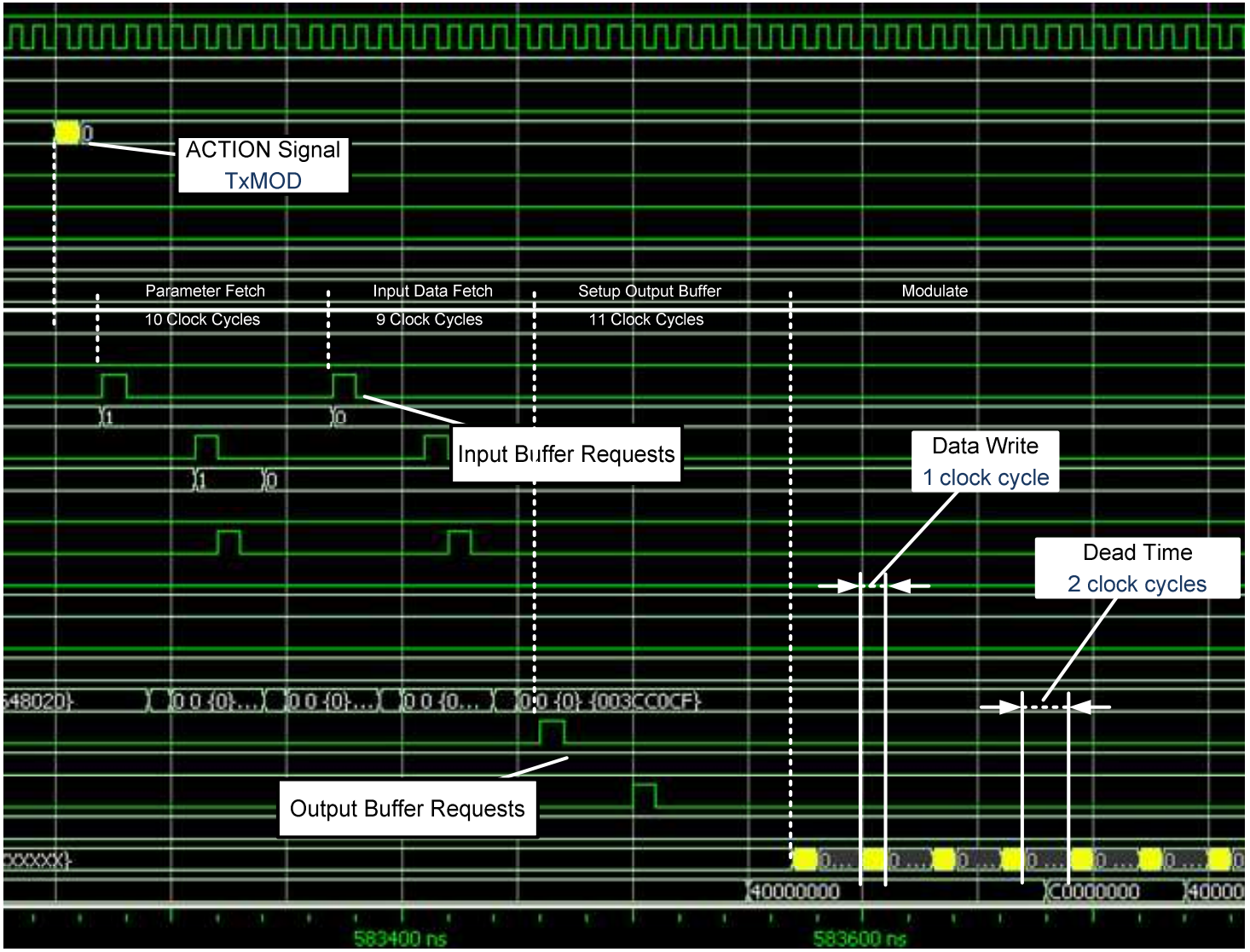
Demodulator Latency



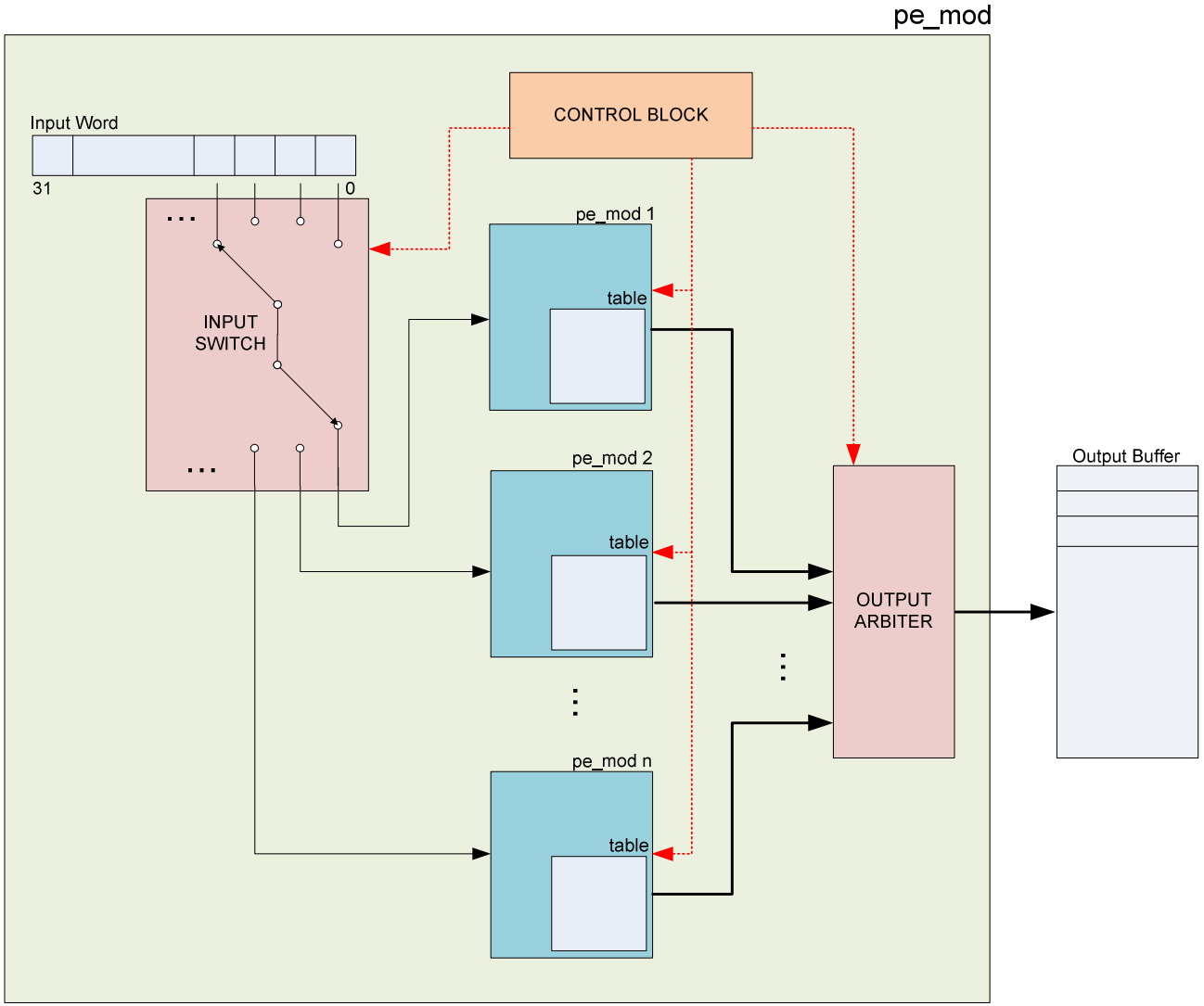
Latency Analysis

- Helpful for System latency calculation
- Estimate performance in time-restricting environments
- Bandwidth Analysis and Improvement
- Room for improvement

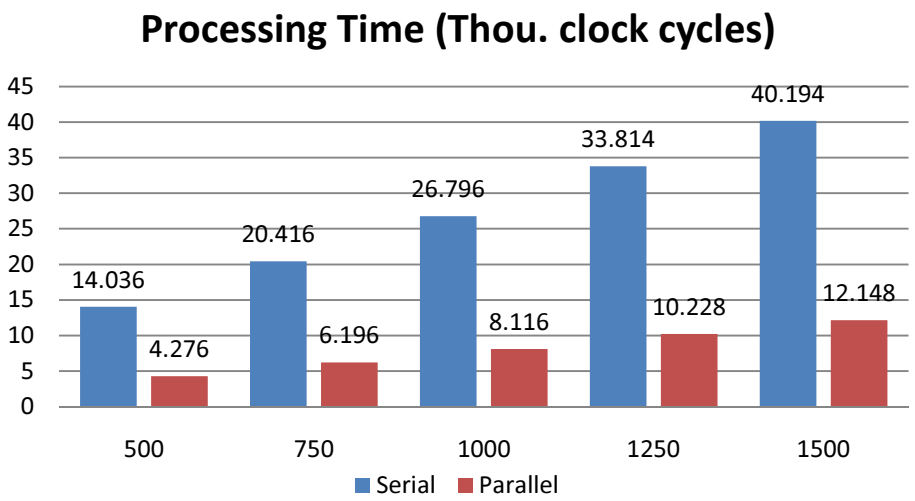
Modulator Observed Waveform

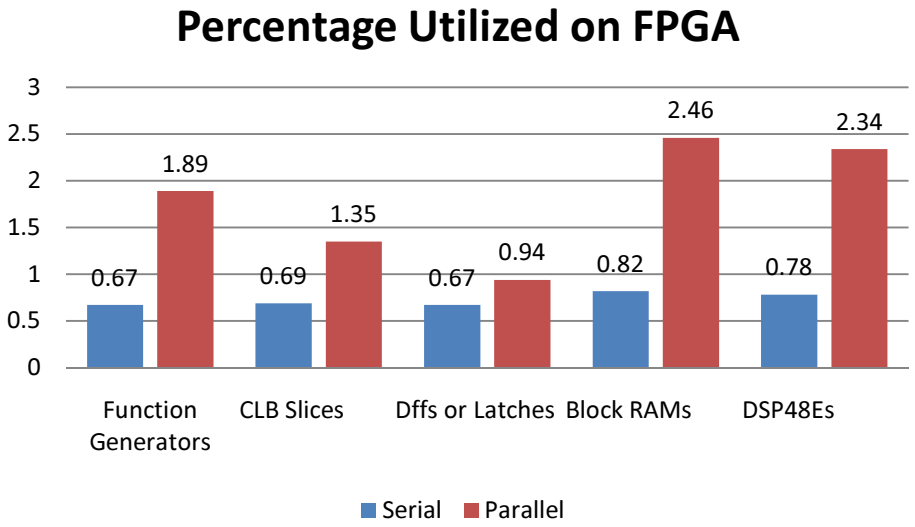


Parallel Implementation



Timing and Resource Analysis

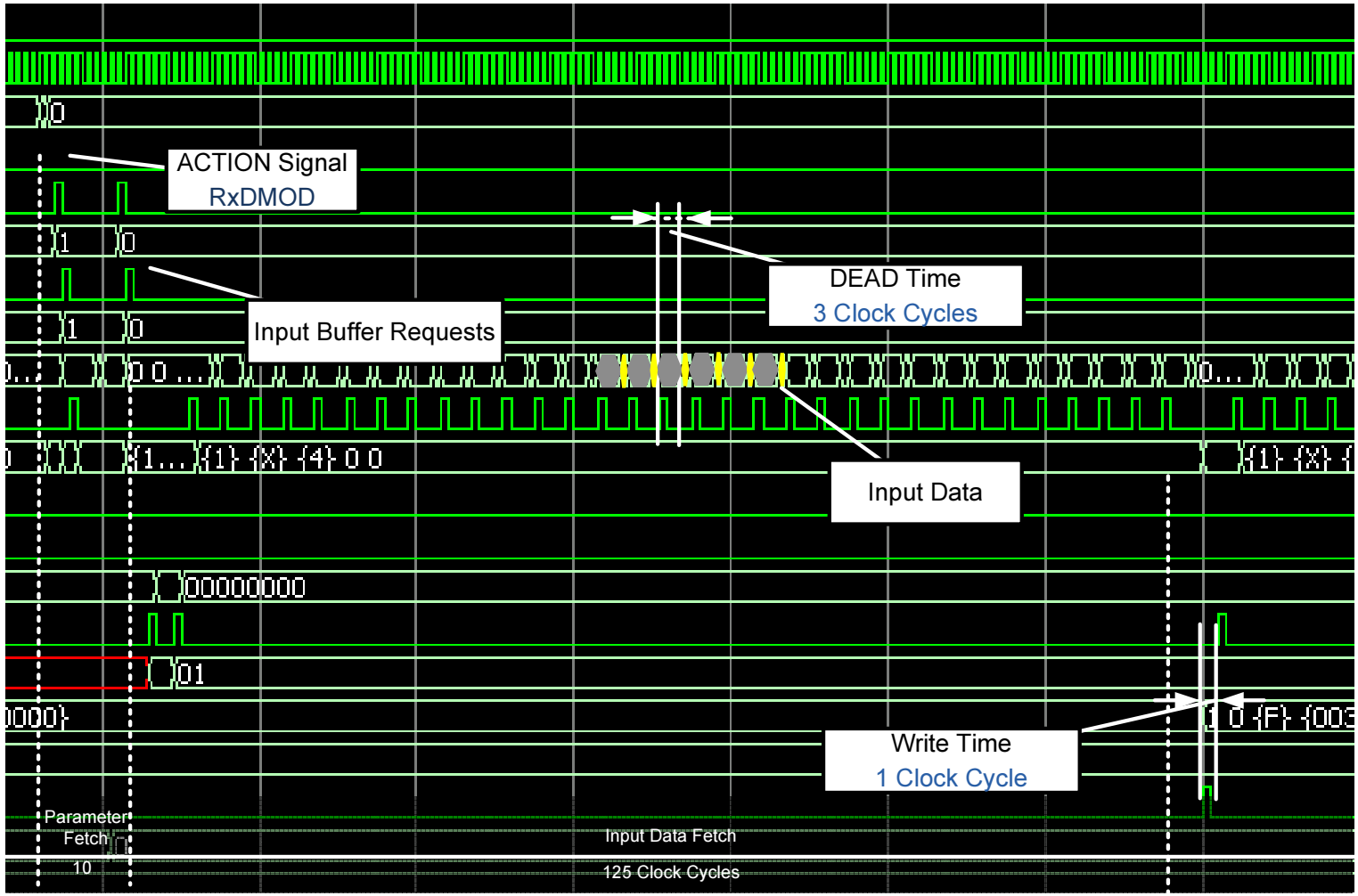




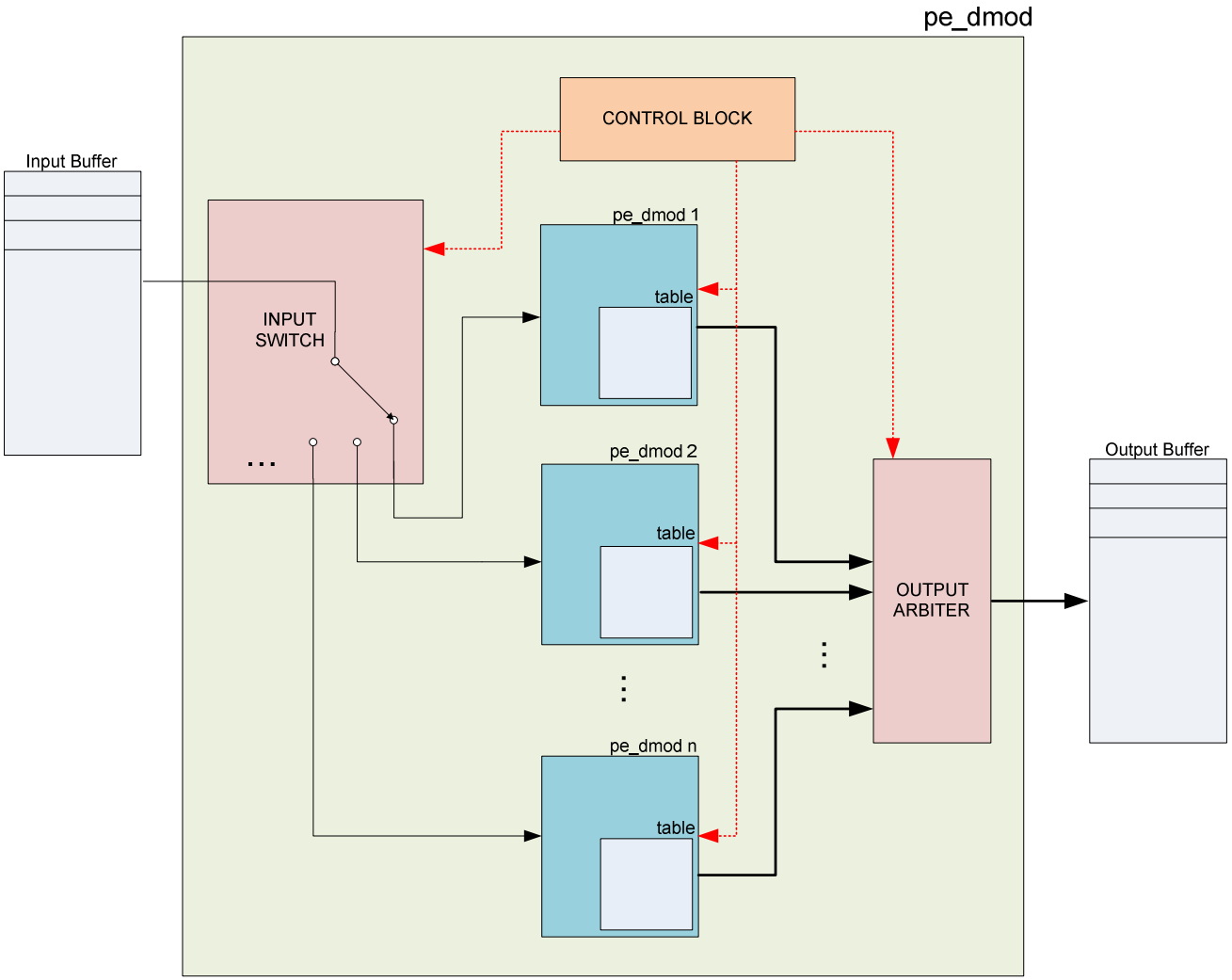
Latency Reduction – 69%

Area Utilization Increase 3 times

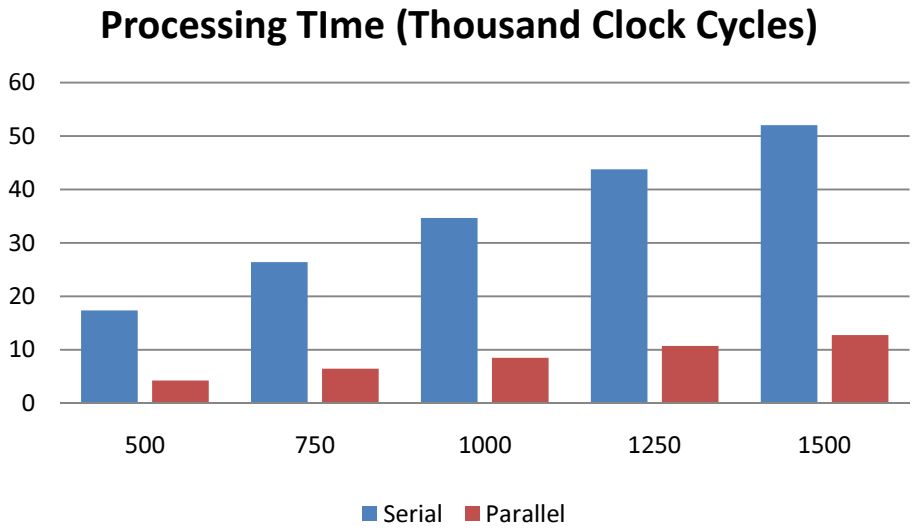
Demodulator Waveform

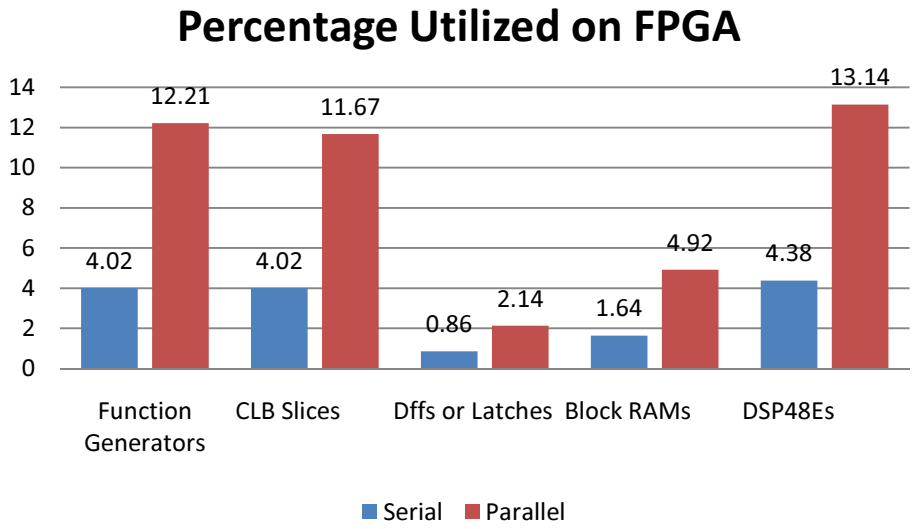


Parallel Implementation



Timing and Resource Analysis



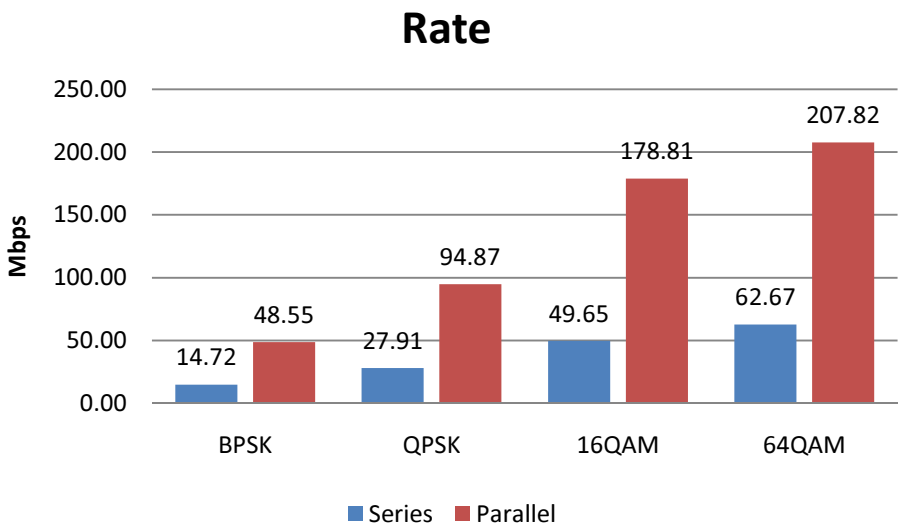


Latency Reduction – 76%

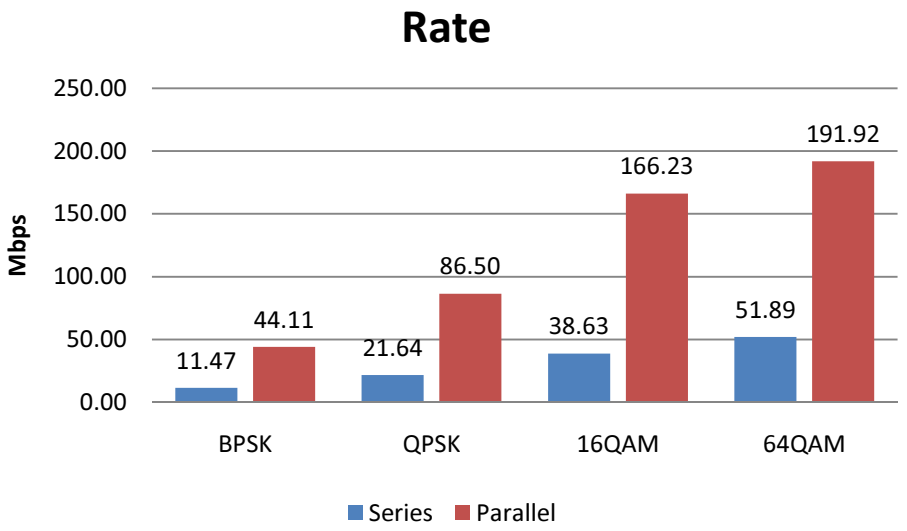
Area Utilization Increase 4 times

Bandwidth

Modulator rate



Demodulator Rate



Outline

1. WiNC2R
   Virtual Flow Pipelining
2. Top Level Architecture Design
   Transmit and Receive Command flow for 802.11a-Lite
3. Processing Engines
   I. Modulator
   II. Demodulator
   III. Checker
   IV. Verification
4. Software on WiNC2R
5. Timing Analysis
   Parallel Implementation Design and Analysis
6. **Conclusion and Future Work**

Conclusion

- WiNC2R Top Level Design
  - Hardware realization of Virtual Flow Pipelining
- 802.11a - Lite
  - VFP implemented for 802.11a Lite protocol
- Processing Engine Design
  - Modulator , Demodulator and Checker
- Software on WiNC2R
  - Setup of Data flows, PE configuration, Populating the Mapping table and Decision Table
- Latency Analysis
  - Processing times for modulator and demodulator
- Parallel Implementation
  - Improved design for reducing Latency

Future Work

- Support better Modulation Schemes
  - Design modulator compatible for circular modulation schemes
- Parallel Implementation
  - Full Functional Verification of parallel design
- Reusing Processing Engine in Hardware
  - Look-up table implementation utilized for other data processing – Coding, Encryption etc
- Software Development
  - Include traffic generators, feedback paths, user-friendliness
- Area Reduction
  - Improvement in design, clustering multiple processing engines in a single functional units

References

1. Z. Miljanic, I. Seskar, K. Le, and D. Raychaudhuri, WINLAB Network Centric Cognitive Radio Hardware Platform - WiNC2R, in Proceedings of International Conference on *Cognitive Radio Oriented Wireless Networks and Communications*, Orlando, Florida, 2007. CrowCom 2007.
2. Zoran Miljanic and Predrag Spasojevic, *Resource Virtualization with Programmable Radio Processing Platform* WICON'08, Maui Hawaii, USA.
3. Network Centric Cognitive Radio (WiNC2R) http://www.winlab.rutgers.edu/docs/focus/WiNC2R.html
4. GNURadio wiki page : http://gnuradio.org/trac
5. WARP wiki page : http://warp.rice.edu/trac
6. Xilinx Documentation : http://www.xilinx.com/support/documentation/
7. Innovative Integration's X5-400M Manual : http://www.innovative-dsp.com/support/manuals/X5-400M.pdf
8. Innovative Integration's X5-400M Logic User Guide
9. WiNC2R wiki page
10. WiNC2R – Processing Engine Documents
11. WiNC2R – Functional Unit Documents
12. WiNC2R – Network Centric Protocol Documents.
13. S.Jain. Hardware and software for WiNC2R. Masters Thesis, Rutgers University, October 2008.
14. Vijayant Bhatnagar. Performance and Hardware Complexity Analysis of Programmable Radio Platform for MIMO OFDM Based WLAN Systems. Masters Thesis, Rutgers University, May 2008.
15. "IEEE 802.11a Physical and MAC Layer Specifications," IEEE 802.11a, 1999.
16. S.Satarkar. Performance Analysis of WiNC2R Platform. Masters Thesis, Rutgers University, August 2009

Thanks