Hardware Implementation of an 802.11a Transmitter

Elizabeth Basha, Steve Gerding, Rose Liu

May 9, 2005

802.11a

- IEEE Standard for wireless communication

- Frequency of Operation: 5Ghz band

- Modulation: Orthogonal Frequency Division Multiplexing

  - OFDM symbol - unit of data transmission

    - One symbol consists of 48 data-encoded complex pairs and 4 pilot complex pairs which protect against noise
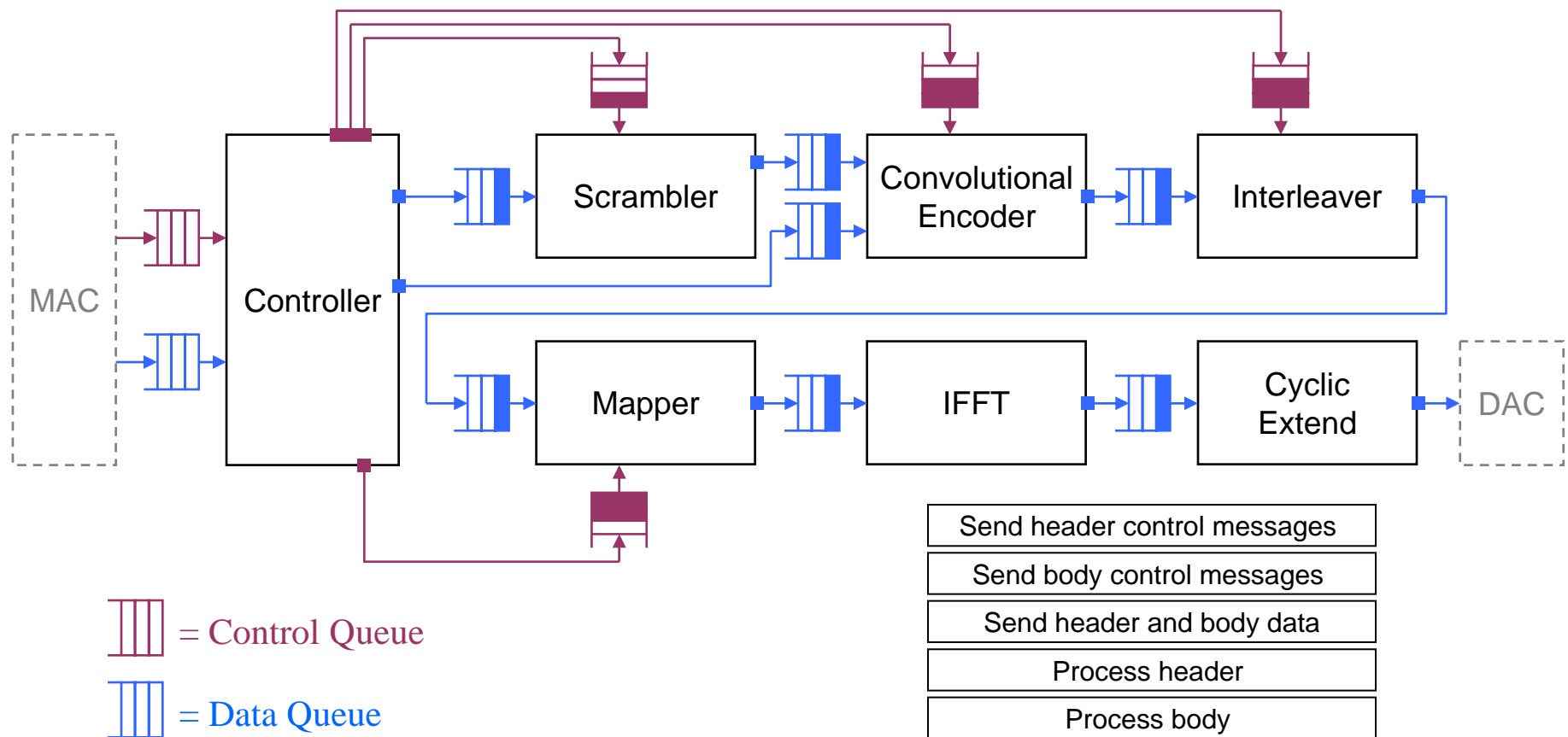
Transmitter Overview

- Tasks:
  - Encodes data for forward error correction
  - Maps data into complex pairs & distributes them among the different frequency indices
  - Transform frequency data into time domain
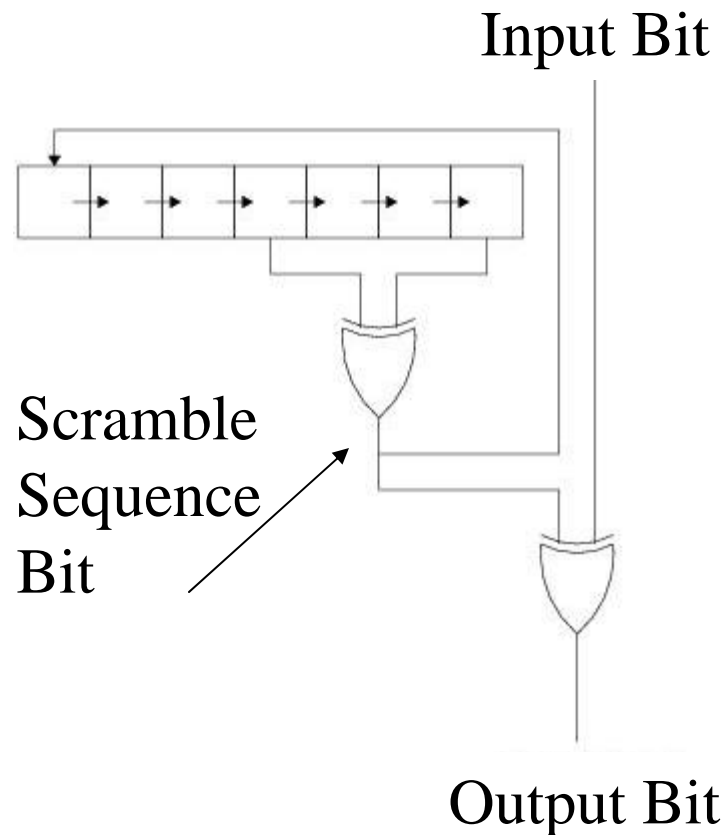
- Packet Format:
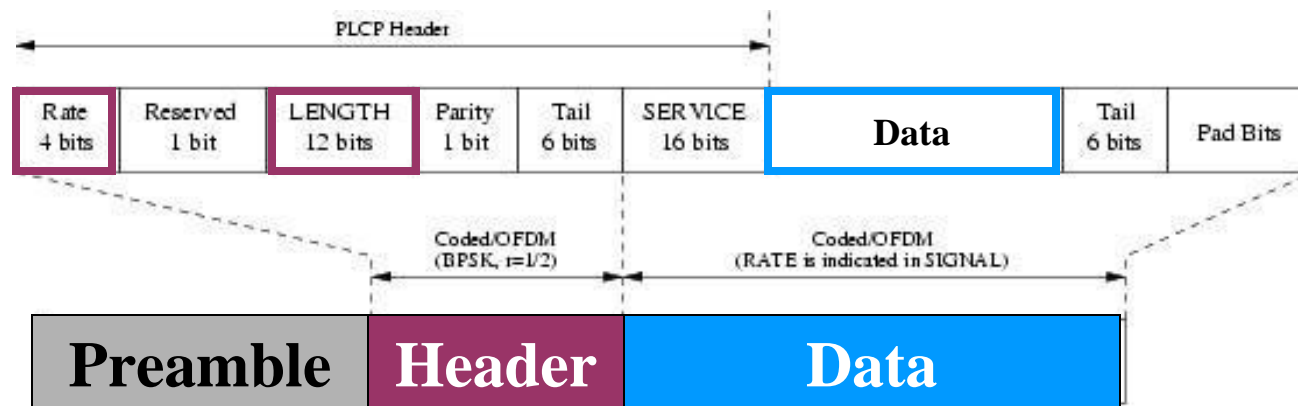
Design Specifications

- Transmit at 3 datarates:

    - 6Mb/s   – 1 24-bit input data frame per OFDM symbol

    - 12Mb/s – 2 24-bit input data frames per OFDM symbol

    - 24Mb/s – 4 24-bit input data frames per OFDM symbol

- Design Goals:

    - Minimize Area

    - Minimize Power by reducing frequency and lowering V_{DD}

    - Just-in-time performance to meet the required datarates

Top Level Model

= Control Queue

= Data Queue

| Send header control messages |
| Send body control messages |
| Send header and body data |
| Process header |
| Process body |

Basic Serial Scrambler Design

Input Bit

Scramble
Sequence
Bit

Output Bit

- Processes 1 input bit per cycle

- Simultaneously generates
1 scramble sequence bit and
computes 1 output bit

- Repeatedly generates a
127-bit scramble sequence

Initial Scrambler Design

For Each Message:
- Generates the entire 127-bit scramble sequence – 127 cycles
- Stores the scrambler sequence to be used throughout the message
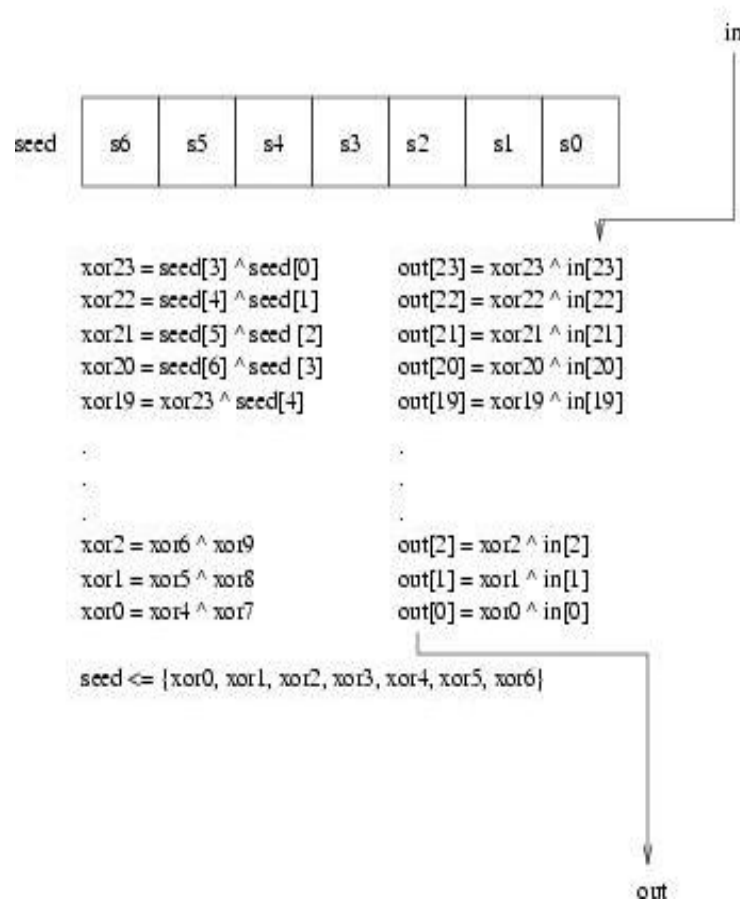
Advantage:
- Processes 1 24-bit input frame per cycle

Disadvantage:
- Large initialization overhead is especially apparent for a
 series of very short messages

Unrolled Scrambler Design

seed | s6 | s5 | s4 | s3 | s2 | s1 | s0

$xor23 = seed[3] \wedge seed[0]$
$xor22 = seed[4] \wedge seed[1]$
$xor21 = seed[5] \wedge seed[2]$
$xor20 = seed[6] \wedge seed[3]$
$xor19 = xor23 \wedge seed[4]$

.
.
.

$xor2 = xor6 \wedge xor9$
$xor1 = xor5 \wedge xor8$
$xor0 = xor4 \wedge xor7$

$out[23] = xor23 \wedge in[23]$
$out[22] = xor22 \wedge in[22]$
$out[21] = xor21 \wedge in[21]$
$out[20] = xor20 \wedge in[20]$
$out[19] = xor19 \wedge in[19]$

.
.
.

$out[2] = xor2 \wedge in[2]$
$out[1] = xor1 \wedge in[1]$
$out[0] = xor0 \wedge in[0]$

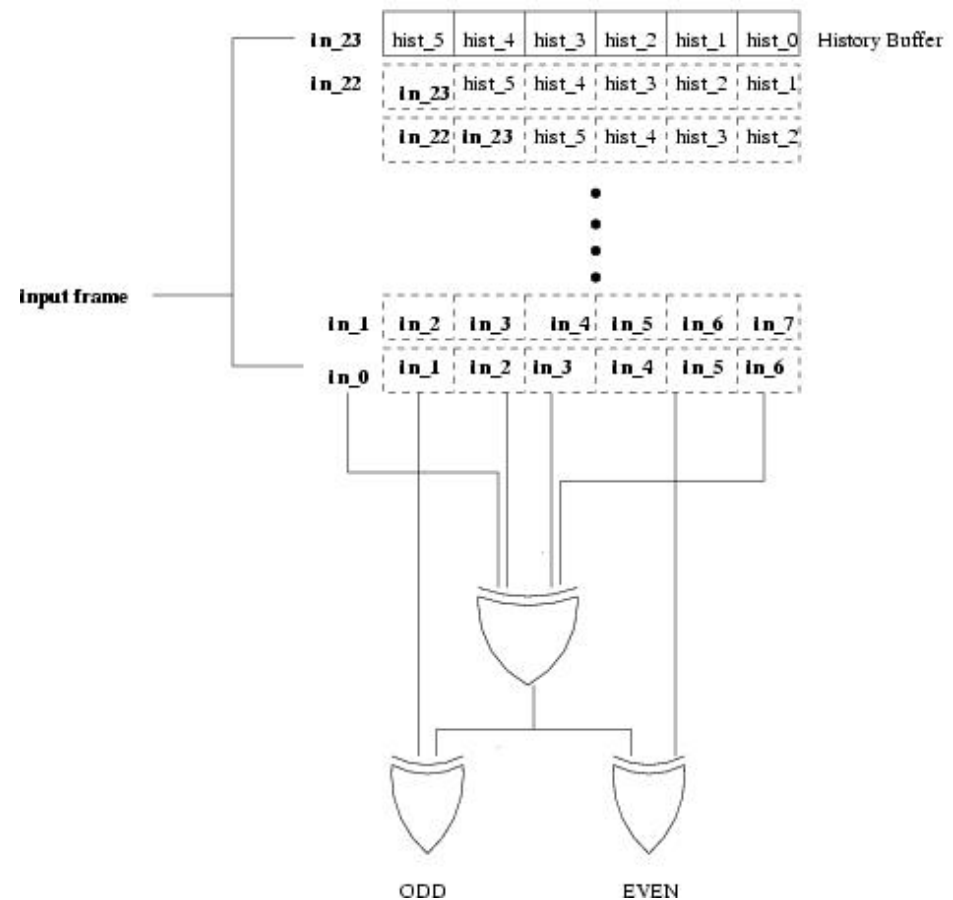$seed <= \{xor0, xor1, xor2, xor3, xor4, xor5, xor6\}$

in

out

- Simultaneously generates 1 frame of the scrambler sequence and processes 1 frame of input data per cycle

- Updates the state of the seed register at end of each cycle

- Advantages:
  - 1 cycle initialization
  - Processes 1 24-bit frame per cycle
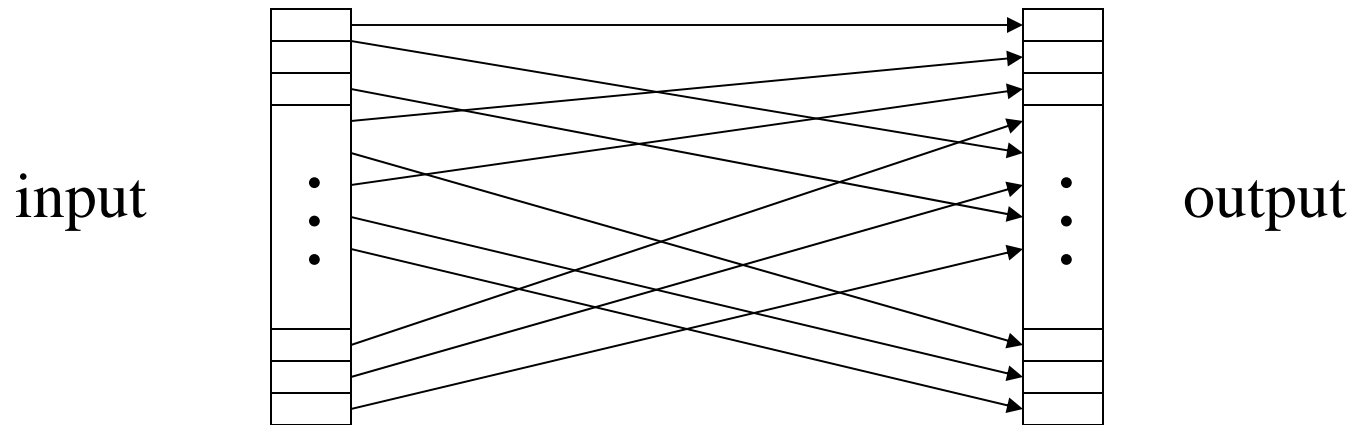
Convolutional Encoder Design
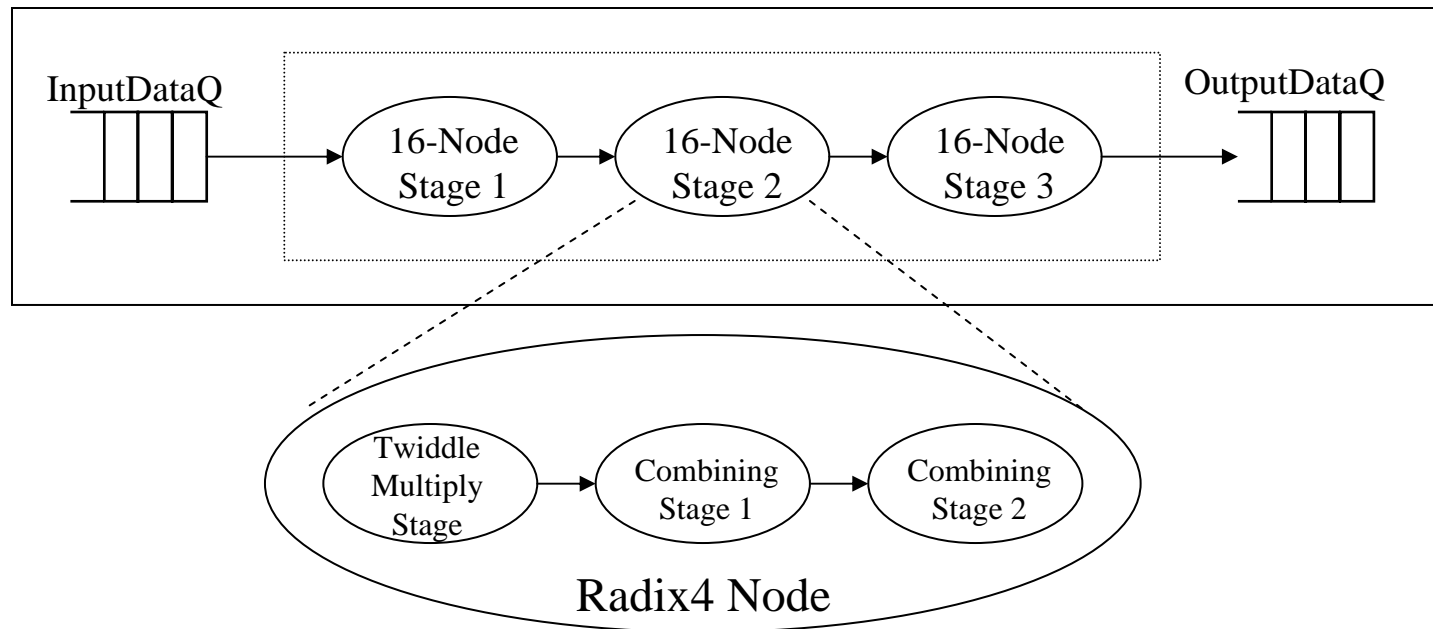
Serial Design
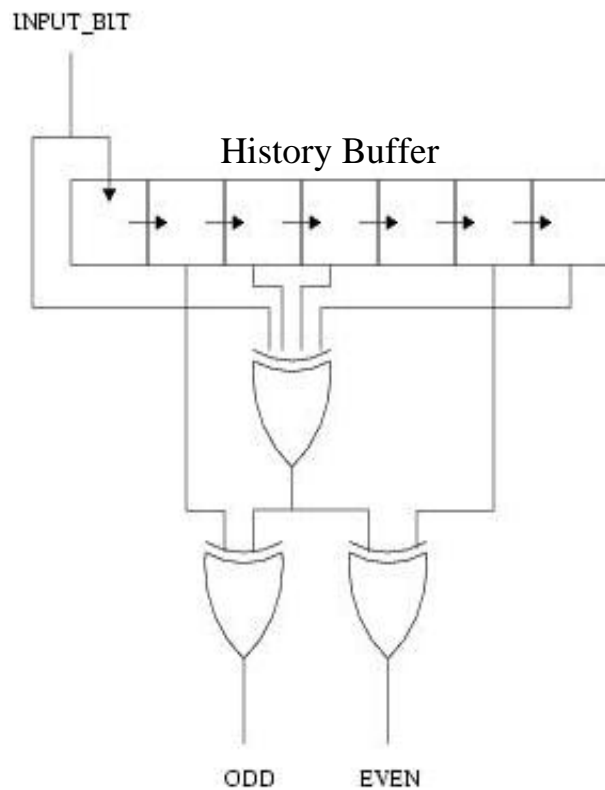
Unrolled Design

Interleaver Algorithm

input output

- Reorders input data bits

- Datarate dependent:

  - Interleaving Pattern

  - # of bits interleaved together

Mapper Algorithm

Input bits

Lookup tables
(1 for each rate)

Frequency domain
OFDM symbol

	I	Q
0	0x8000	0x0000
1	0x7FFF	0x0000
	.	.
	.	.
	.	.

	I	Q
0	0xA57E	0xA57E
1	0x5A82	0xA57E
	.	.
	.	.
	.	.

	I	Q
0	0x8692	0x8692
1	0x796E	0x8692
	.	.
	.	.
	.	.

0xA57E	0xA57E
0xA57E	0x5A7E

Rate

IFFT Initial Design

- Area = 29.12mm^2
- Cycle Time = 63.18ns

Radix4 Nodes	*	+
1	16	24
48	768	1152

IFFT Design Exploration 1
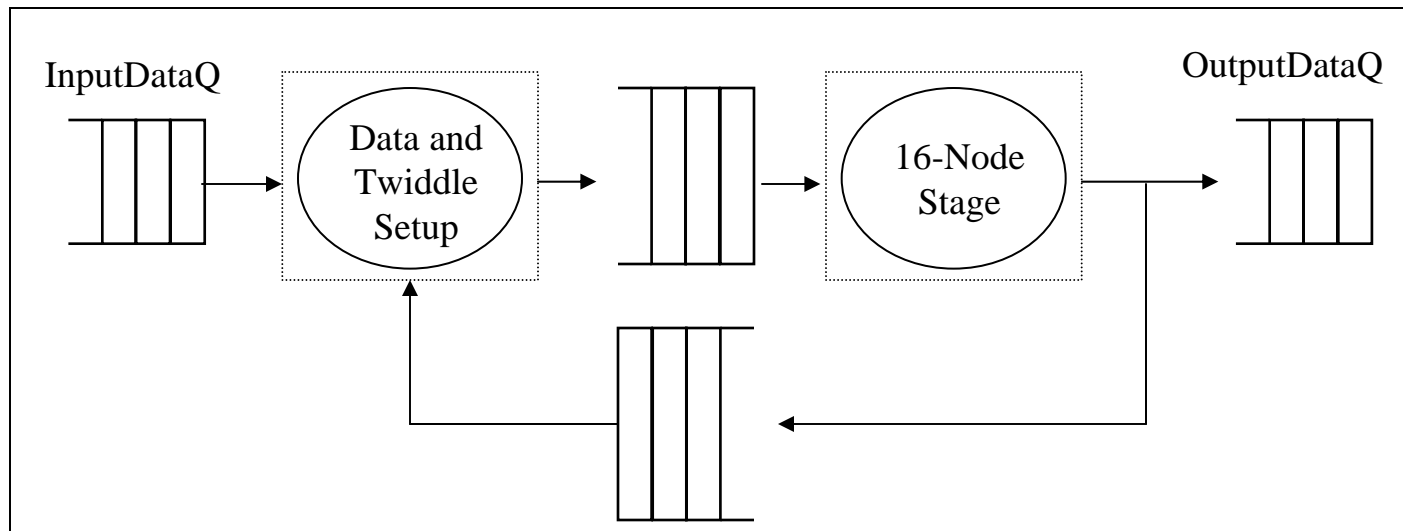
- Area = 5.19mm^2
- Cycle Time = 30.50ns

IFFT Design Exploration 2
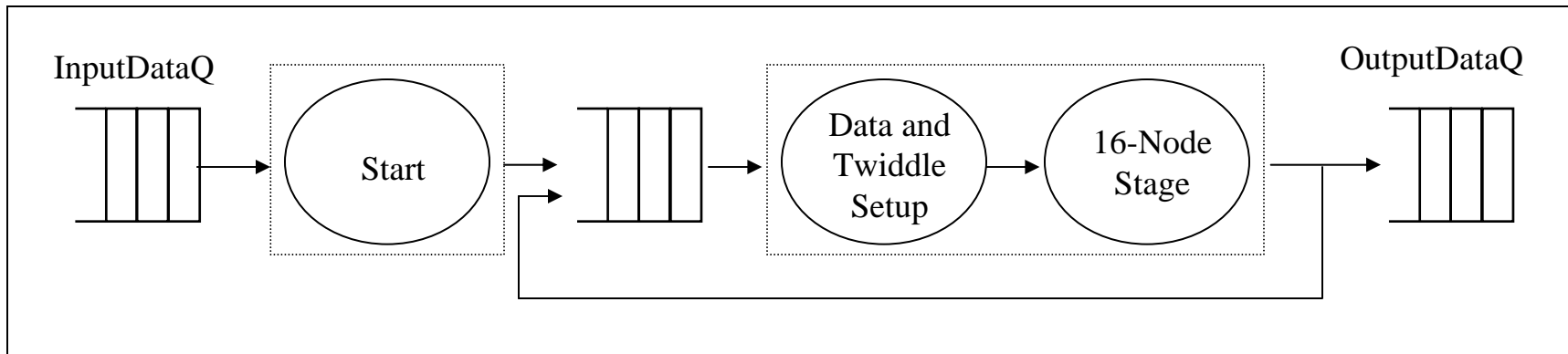
- Area = 4.57mm^2
- Cycle Time = 32.89ns

Cyclic Extender

64 Complex Pairs of Data

First Complex Pair

Last 16 Complex Pairs

64 Complex Pairs of Data
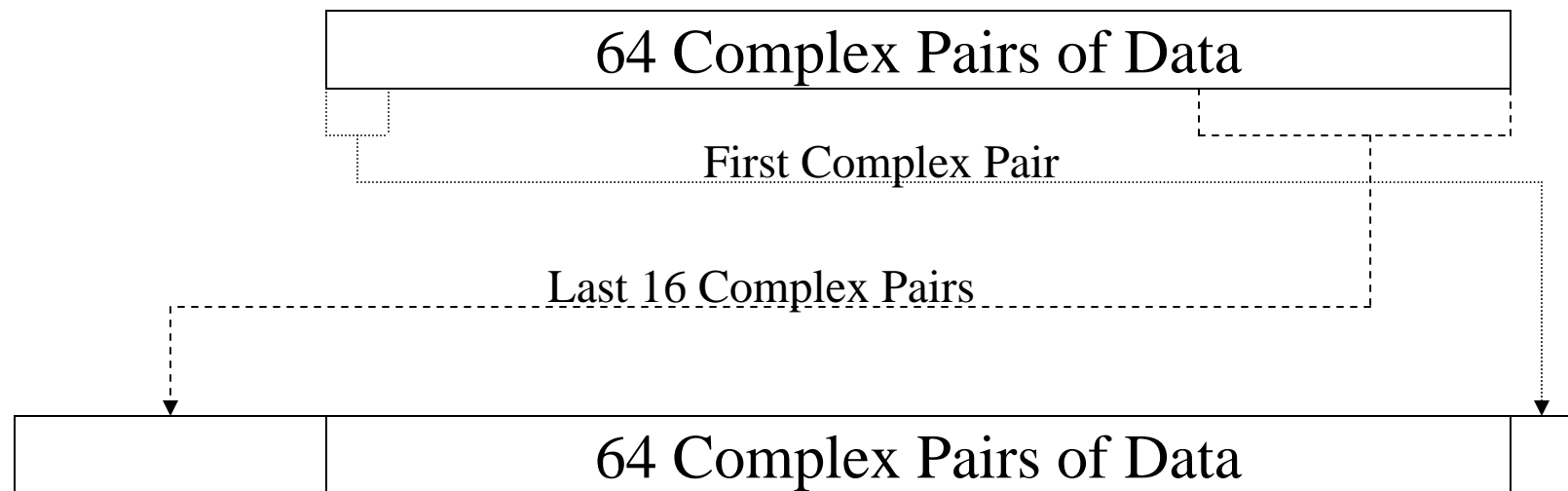
Test Strategy

- Our test structure must enable us to:
  - Debug each module separately
  - Quickly verify new version of modules
  - Verify correctness of entire system
  - Measure throughput of individual modules and system as a whole
- To do this, we leveraged the framework of the Extreme Benchmark Suite (XBS)
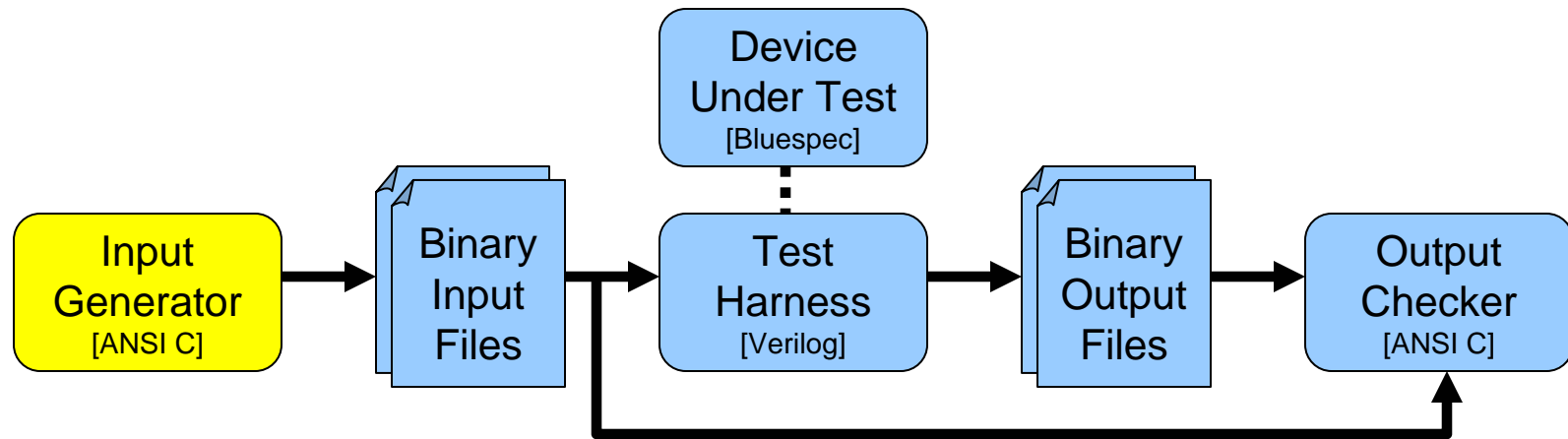
XBS Overview

XBS is a benchmark suite designed to measure the performance of highly parallel processors and custom hardware implementations.
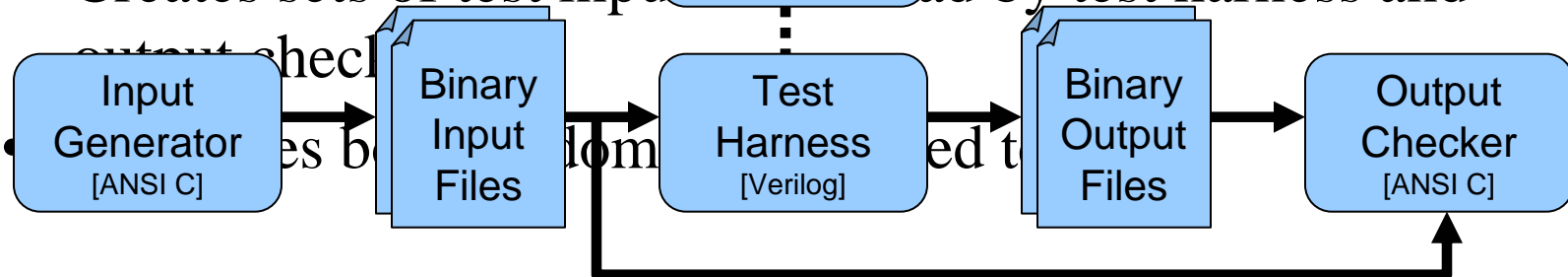
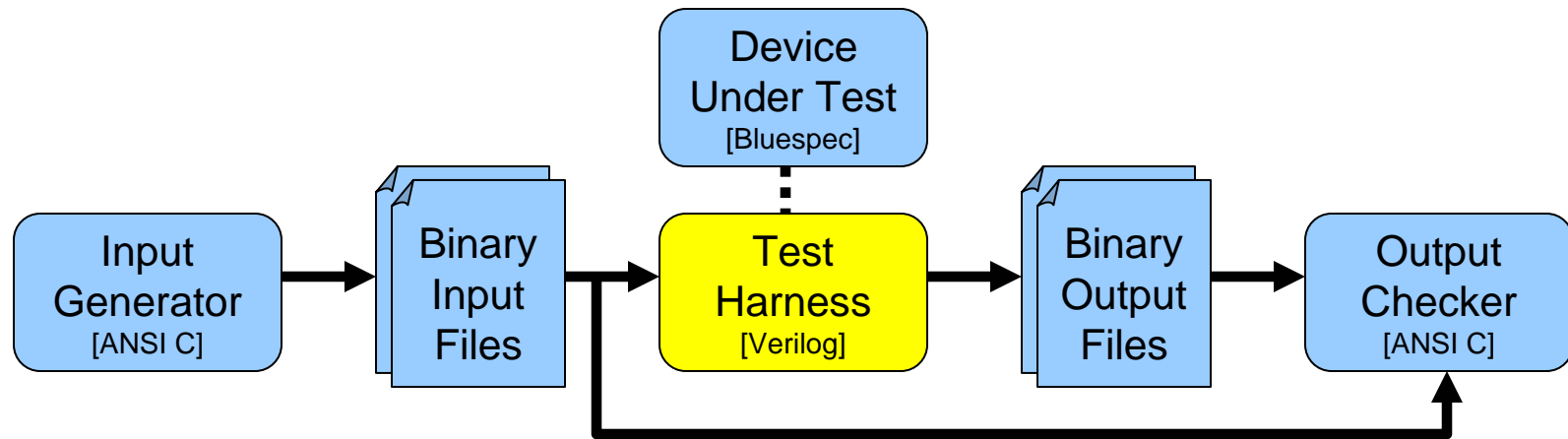All XBS benchmarks have the following structure:
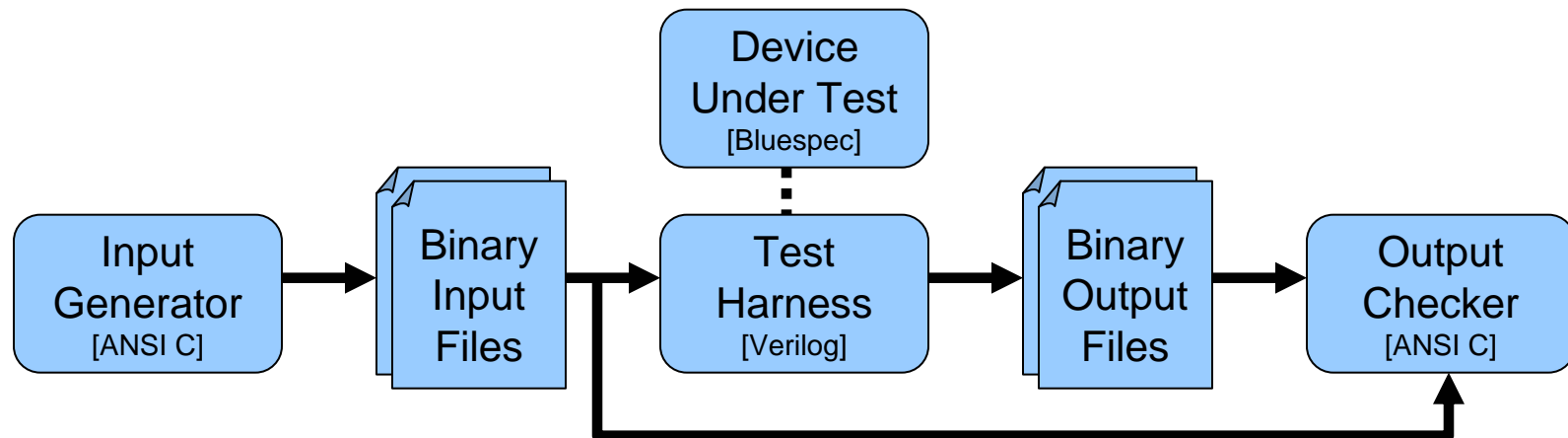
XBS Overview

Input Generator

- Creates sets of test inpu~~t~~... ad by test harness and

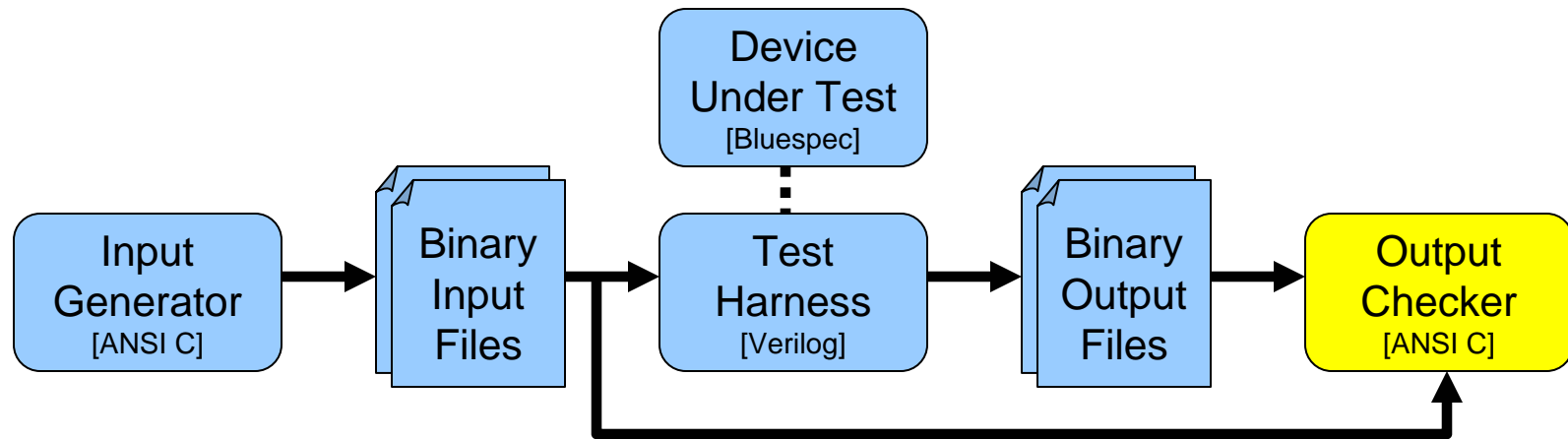 ~~output check~~ ...es b~~...dom...~~ ed t...

XBS Overview

Test Harness

- Encapsulates device under test

- Reads in input files and generates output files

- Measures performance [throughput in bits per cycle] of device under test

XBS Overview

Output Checker

- Reads in input and output files and determines if output files are correct

- Usually contains an ANSI C reference version of DUT

- If output is incorrect, displays location of discrepancy and correct value for debugging purposes

Results

Place and route results:

Total Area	5.27 mm^2
Critical Path Delay	32.89 ns

XBS testing results:

Module	Throughput					
	Input bits per cycle			OFDM symbols per cycle		
Scrambler	24	24	24	1	0.5	0.25
Convolutional Encoder	24	24	24	1	0.5	0.25
Interleaver	12	12	12	0.5	0.25	0.125
Mapper	12	16	19.2	0.5	0.333	0.2
IFFT	6	12	24	0.25	0.25	0.25
Cyclic Extend	24	48	96	1	1	1
Transmitter System	**6**	**12**	**12**	**0.25**	**0.25**	**0.125**

= 6 Mbps

= 12 Mbps
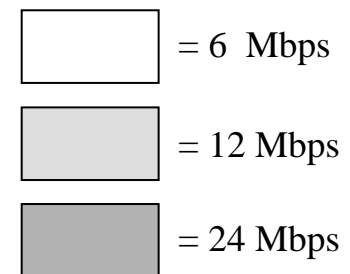
= 24 Mbps

Evaluation

- Our design fully conforms to the IEEE 802.11a standard
- Our design meets timing for the 6, 12, and 24 Mbps transmission rates
  - Total system throughput (24 Mbps) = 12 bits / cycle
  - Clock Frequency = 30.4 MHz
  - Maximum data rate of system = 364.8 Mbps
- We can turn our timing slack into power savings by reducing V_{DD} and clock frequency
  - Clock frequency can be reduced to 2.0 MHz