Parallel Implementation of the DSP Functions of the PAM-5 10Gb/s Transceiver

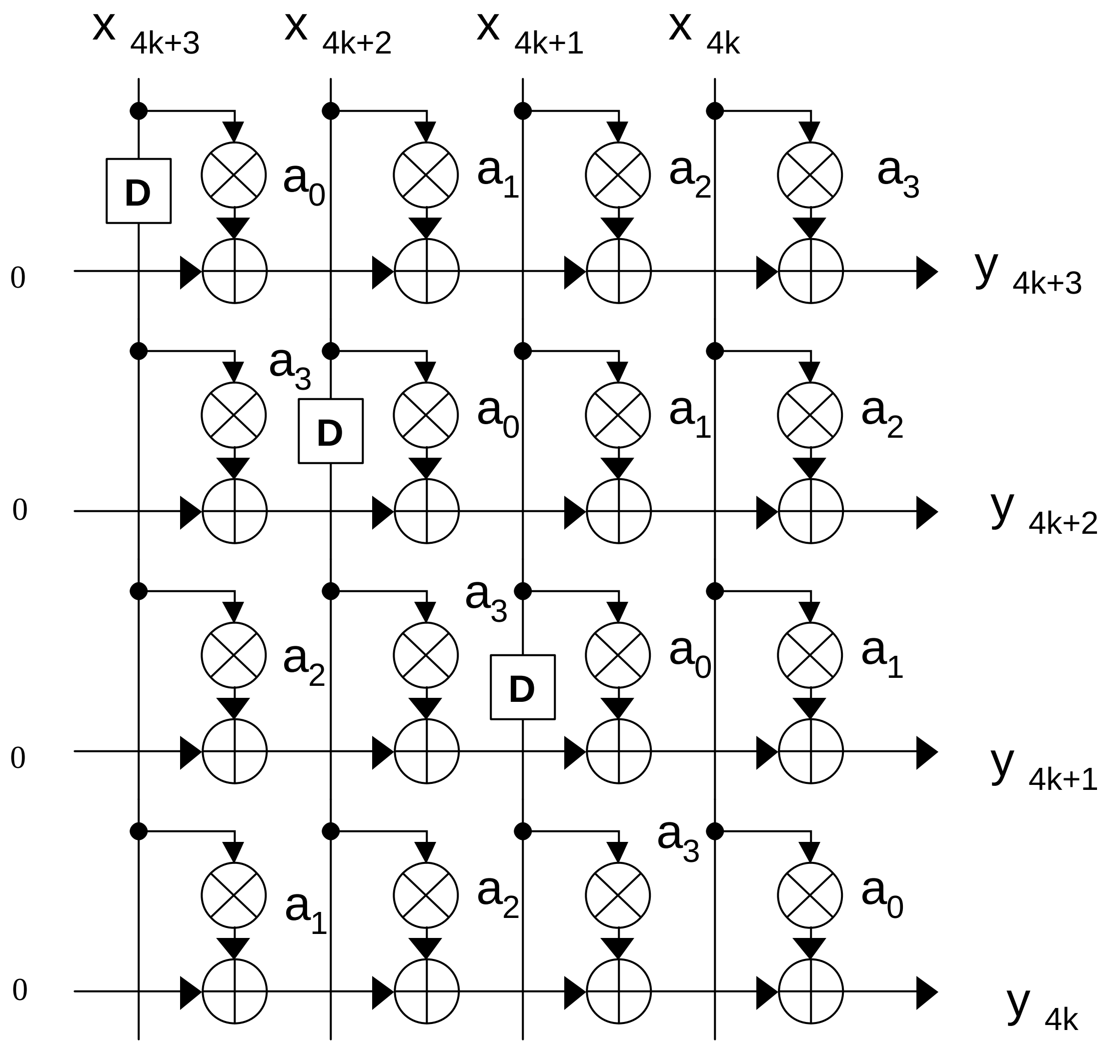
Keshab K. Parhi, Christian Lütkemeyer,
Arthur Abnous, and Mehdi Hatamian

Broadcom Corporation, Irvine, CA

Receive Linear Equalizer

- Implemented as FIR Filter
- Parallelization Required to meet timing

4-Parallel 4-Tap FIR Filter

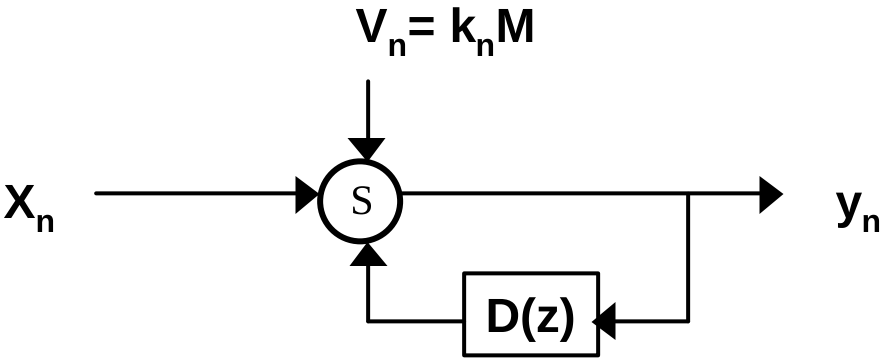


$$y(n) = a_0\,x(n) + a_1\,x(n-1) + a_2\,x(n-2) + a_3\,x(n-3)$$

Parallel FFE

- Feed-Forward Equalizer (FFE) can be implemented in parallel easily
- For 16-tap, 16-parallel FFE, #mult = 256

Tomlinson-Harashima Precoding



- Parallel implementation of precoder is difficult due to feedback loop at high speeds
- However, look-ahead techniques can be used to implement the precoder in pipelined or parallel manner (Parhi, VLSI Digital Signal Processing Systems, Wiley, 1999, Chapter 11)

2-Parallel 3-Tap IIR Filter

$y(n) = a_1\, y(n\text{-}1) + a_2\, y(n\text{-}2) + a_3\, y(n\text{-}3) + x(n)$ (1)

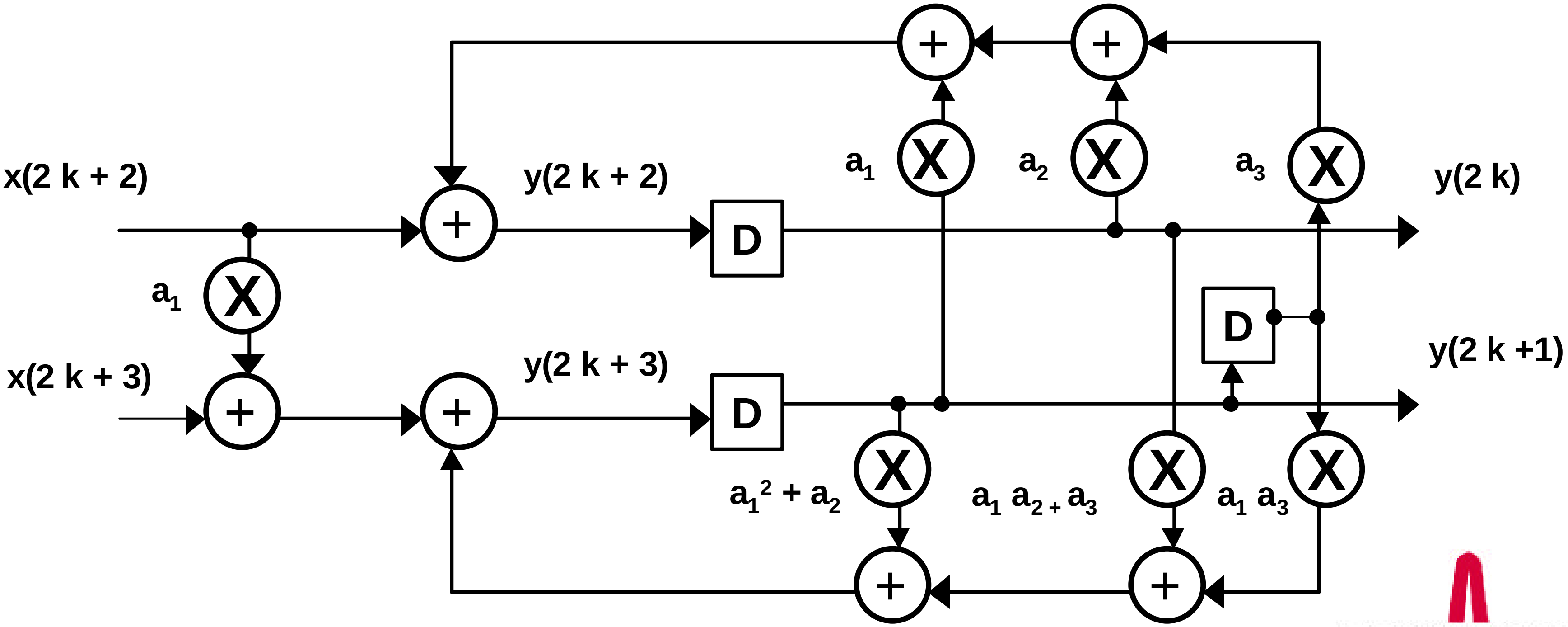
$y(n) = a_1 [\, a_1\, y(n\text{-}2) + a_2\, y(n\text{-}3) + a_3\, y(n\text{-}4) + x(n\text{-}1)\,] + a_2\, y(n\text{-}2) + a_3\, y(n\text{-}3) + x(n)$

$= (a_1^{\,2} + a_2\,)\; y(n\text{-}2) + (a_1\, a_{2\,+}\, a_3)\; y(n\text{-}3) + a_1\, a_3\, y(n\text{-}4) + a_1\, x(n\text{-}1) + x(n)$ (2)

$n = 2\,k + 2$ in equation (1) and $n = 2\,k + 3$ in equation (2)

$y(2\,k + 2) = a_1\, y(2\,k + 1) + a_2\, y(2\,k) + a_3\, y(2\,k - 1) + x(2\,k + 2)$ (3)

$y(2\,k + 3) = (a_1^{\,2} + a_2\,)\; y(2\,k + 1) + (a_1\, a_{2\,+}\, a_3)\; y(2\,k) + a_1\, a_3\, y(2\,k - 1) + a_1\, x(2\,k + 2) + x(2\,k + 3)$ (4)



L-Parallel Implementation

- y(kL+L) … y(kL+2L-1) computed in terms of y(kL) … y(kL+L-1)
- In L-Parallel implementation, the clock speed is L-times slower than symbol speed, i.e., each delay element is L-slow

Precoder

$$y_n = \sum_{i=1}^{N} a_i \cdot y_{n-i} + x_n + v_n$$

- X_n is 5-Level PAM

- Choose v_n from

x_n	v_n
-2	5 or 0
-1	5 or 0
0	0
1	-5 or 0
2	-5 or 0

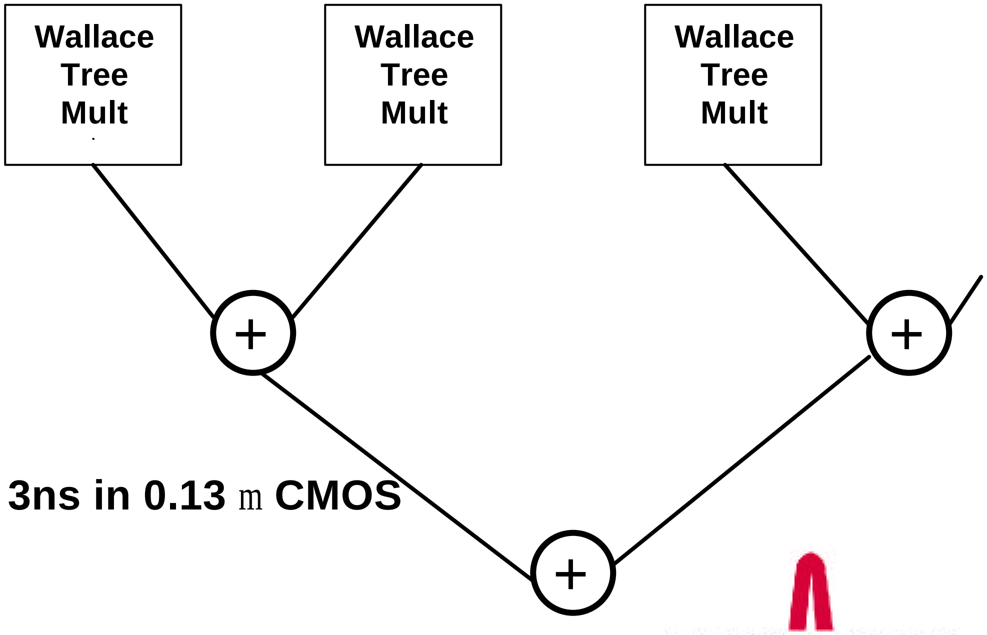
to obtain a 9-level signal and to minimize power of *y*

Parallel Precoder

- **Advantages**
  - Higher Speed, lower latency
  - More stable than original, poles are raised to L-th power
  - Better finite precision effect

Parallel Precoder Performance

- **N = Filter Length = 30**
- **L = Level of Parallelism = 16**
- **Hardware Complexity**
  - ➢# mult = 16 x17/2 + 16 x 30 = 616 (8 x 8 - b)
- **Critical Path**
  - ➢1 8 x 8 - b mult
  - ➢+ [$\log_2 30$] = 5 adds



- **Critical Path of**

- 8 x 8 - b Wallace Tree mult = 3 t_{fa}

- 5 stages of adds in carry-save = 5t_{fa}

- Vector Merging = 5tmux = 2 t_{fa}

- **Total critical path = 10 t_{fa}= 10 (0.3 ns) = 3ns in 0.13 m CMOS**
- **Desired speed achievable!**

Viterbi Decoder

- Use of a 4-dimensional 4-way interleaved trellis code requires a clock speed of 312.5 MHz for the Viterbi Decoder
- Viterbi Decoder is much simpler to implement than 1000 Base-T because it does not need to be combined with DFE (even though the decoder speed is 312.5/125 = 2.5 times higher)
- Branch Metrics Computations can be pipelined
- This eliminates all critical path issues of the Viterbi decoder

Conclusions

- **Desired Speed is Achievable**
- **Area is Insignificant**
- **Power Consumption of Digital Part about 2W**