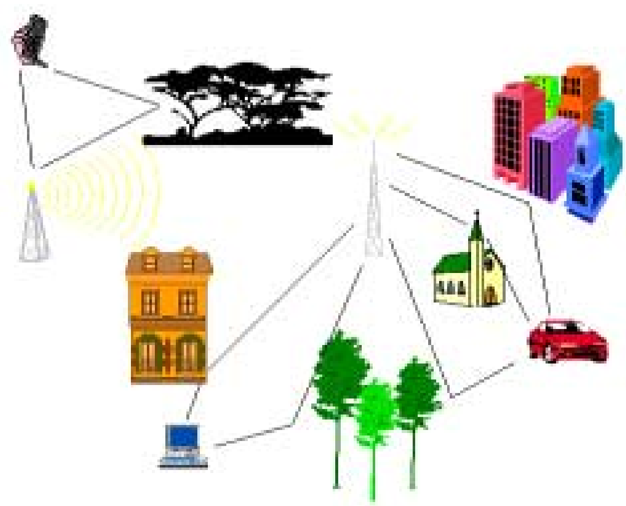
Adaptive Equalizers

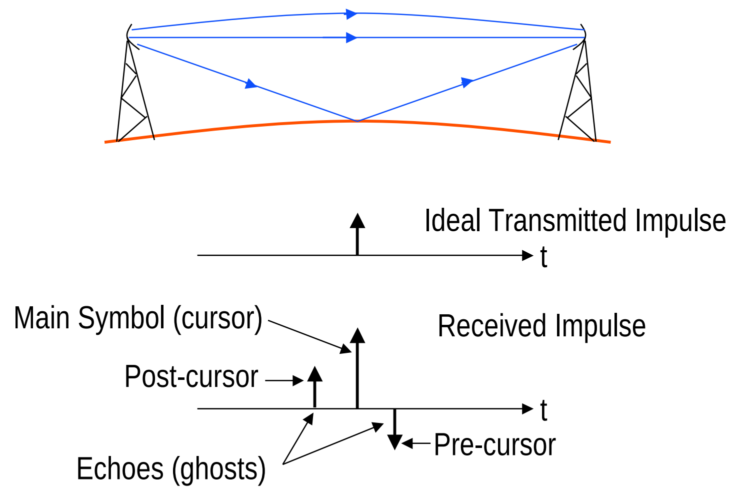
Dr Chris Dick
DSP Chief Architect
Director, Signal Processing Engineering
Xilinx Inc.

Multipath Environment



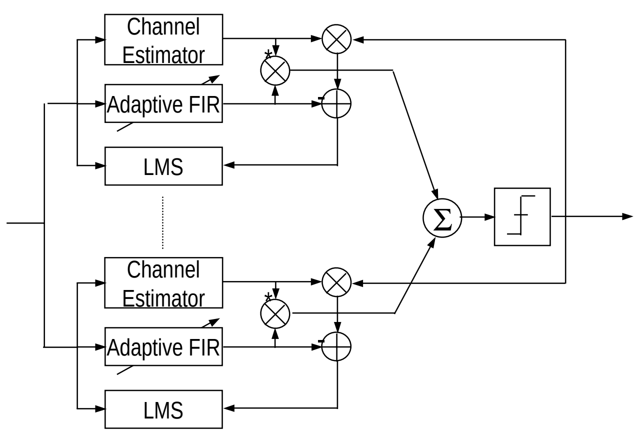
Adaptive Filters & Channel Equalizers 2

Why an Equalizer?



Adaptive Filters & Channel Equalizers 3

Adaptive LMMSE-RAKE Receiver

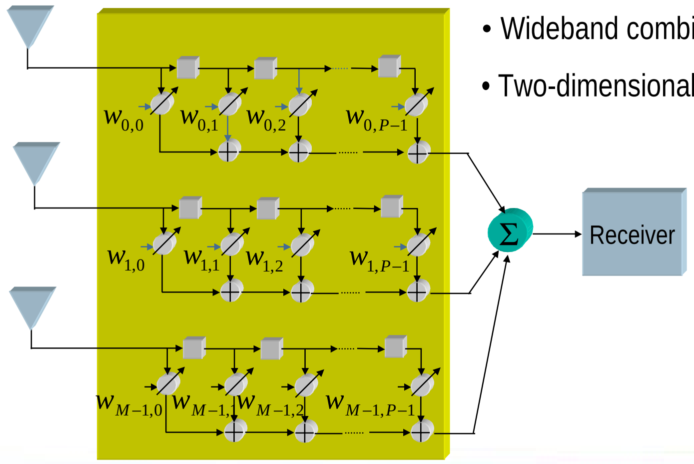


LMMSE: *linear minimum mean squared error*

F. Swarts, Ed., CDMA Techniques for Third Generation Mobile Systems, Kluwer AcademicNorwell, Mass., 1999.

Adaptive Filters & Channel Equalizers 4

3G Wireless Space-Time Processor

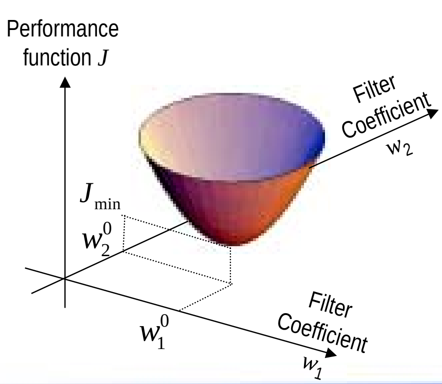


- Wideband combining network
- Two-dimensional processor

Adaptive Filters & Channel Equalizers 5

Two-Tap Filter

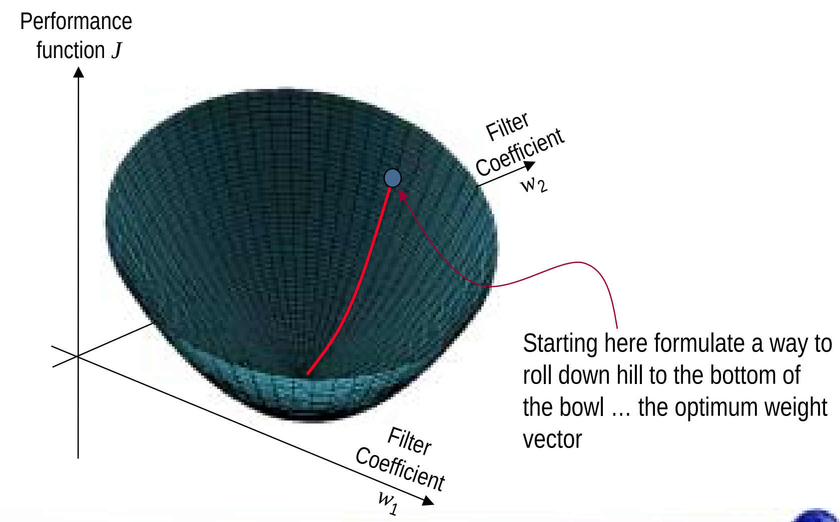
- Adaptive filter has two coefficients
- w_1 and w_2 must be chosen so that the squared-error surface J is minimized



- Error surface is now a paraboloid
- Surface obtains minimum value $J_{\min}$ where w_1 and w_2 equal their optimum values w_1^0 and w_2^0

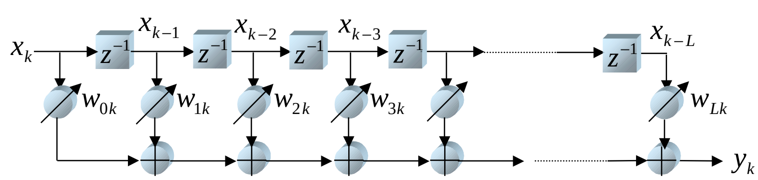
Adaptive Filters & Channel Equalizers 6

Search Technique



Adaptive Filters & Channel Equalizers 7

Adaptive Filters



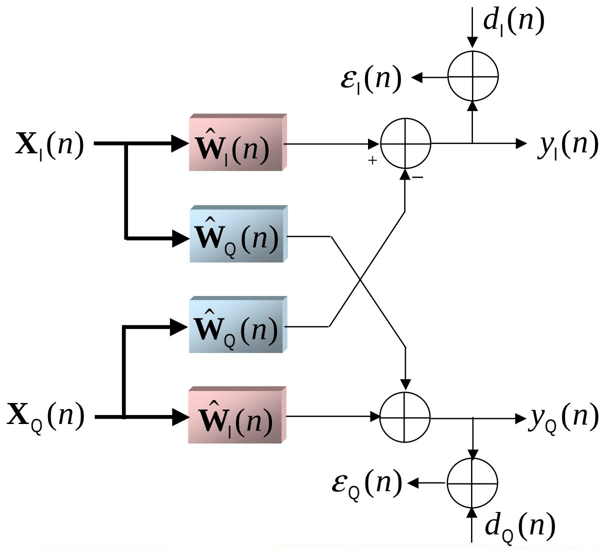
$$\mathbf{X_k} = [x_{k,} x_{k-1}, \ldots, x_{k-L}]^{\mathrm{T}}$$

$$\mathbf{W_k} = [w_{k,} w_{1k}, \ldots, w_{Lk}]^{\mathrm{T}}$$

$$y_k = \mathbf{X}_k^{\mathrm{T}} \mathbf{W}_k = \mathbf{W}_k^{\mathrm{T}} \mathbf{X}_k$$

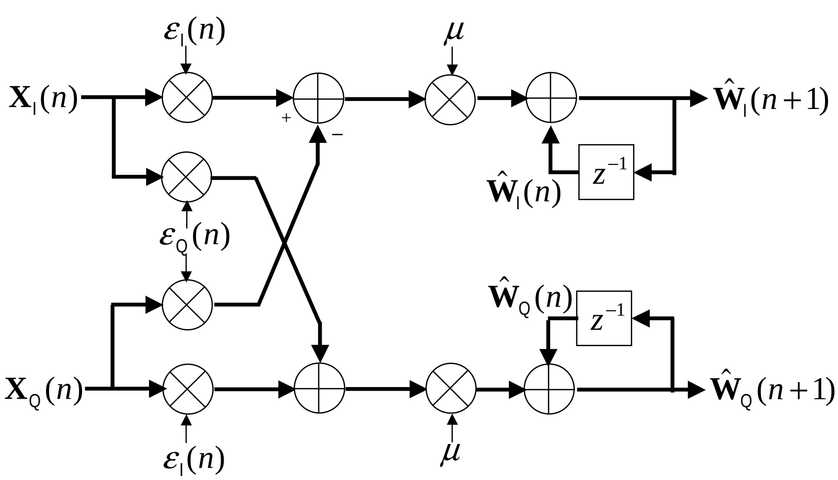
Adaptive Filters & Channel Equalizers 8

Cmplx LMS Algorithm: Outp & Err



Adaptive Filters & Channel Equalizers 9

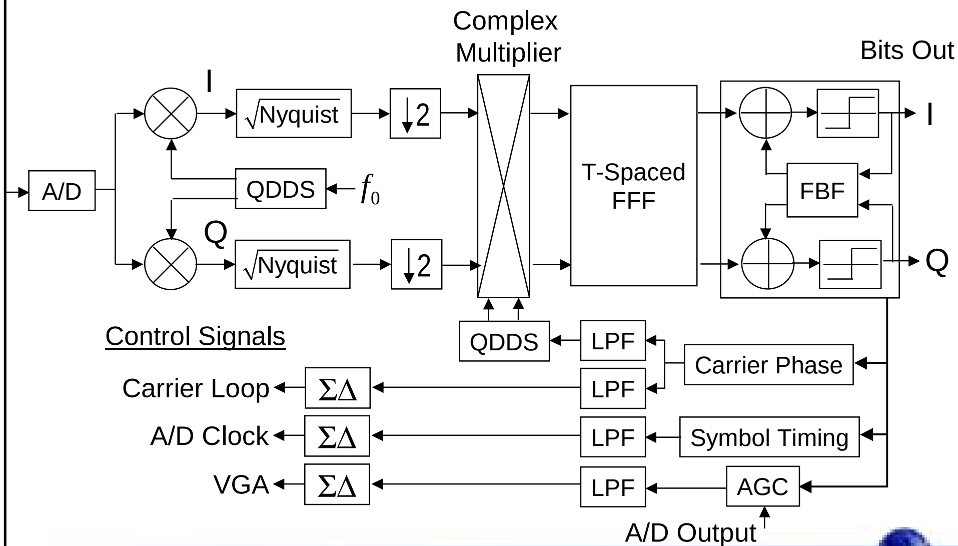
Complex LMS Algorithm: Coeff Update



→ Denotes vector quantity

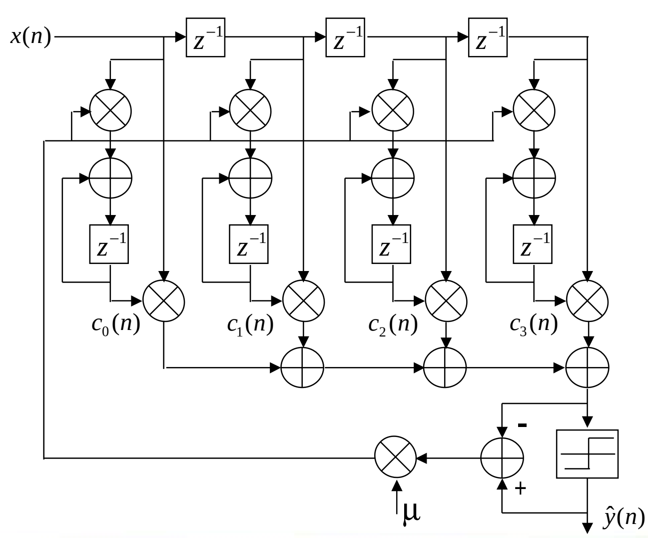
Adaptive Filters & Channel Equalizers 10

QAM Demod Block Diagram (1)



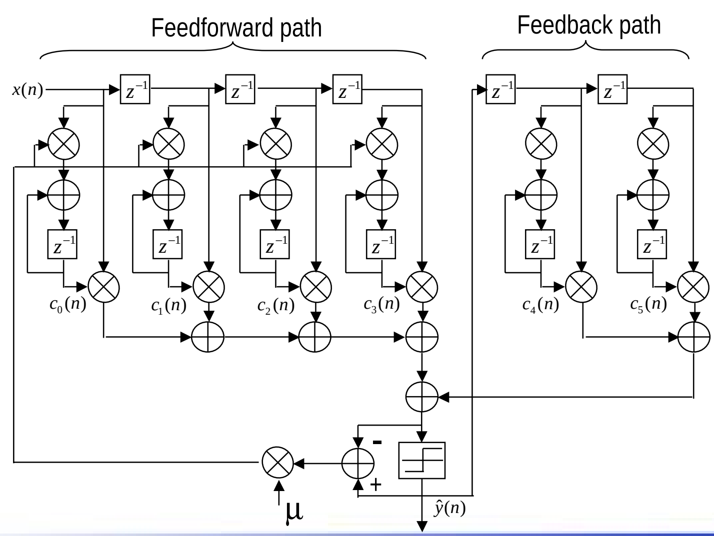
Adaptive Filters & Channel Equalizers 11

Decision Directed Equalizer



Adaptive Filters & Channel Equalizers 12

Decision Feedback Equalizer



Adaptive Filters & Channel Equalizers 13

Processing Requirements

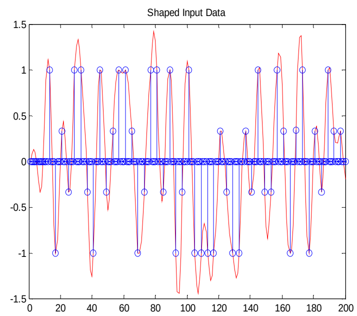
	Baud Rate	Delay Spread	Equalizer Length	Filter Computation (MMACs)
VGC (128-QAM)	2400 Hz	3 ms	32	0.307
Cable (64-QAM)	5.1 MHz	2 us	32	700
Benign static microwave (64-QAM)	33 MHz	6 ns	11	1.452
Static Microwave (64-QAM)	33 MHz	1 us	128	17,000
"On the move" microwave (QPSK = 4-QAM)	33 MHz	0.3 us	40	5,200

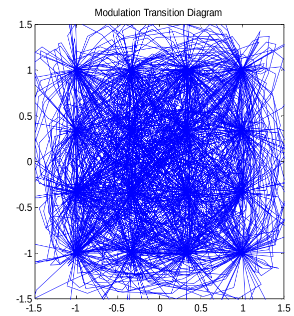
J. R Treichler, M. G. Larimore and G. C. Harp, "Practical Blind Demodulators for High-Order QAM Signals, *Proc. IEEE,* Vol. 86 No. 10, pp. 1907-1926, 1998.

Adaptive Filters & Channel Equalizers 14

16-QAM Channel Equalizer Transmitter (Tx)

- Data and Constellation

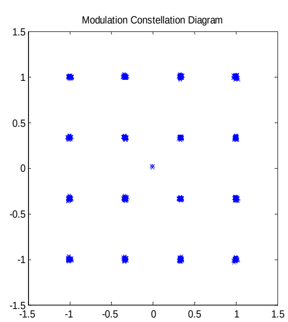


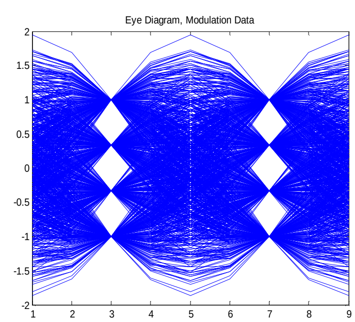


Adaptive Filters & Channel Equalizers 15

16-QAM Channel Equalizer Transmitter

- Constellation and eye diagram

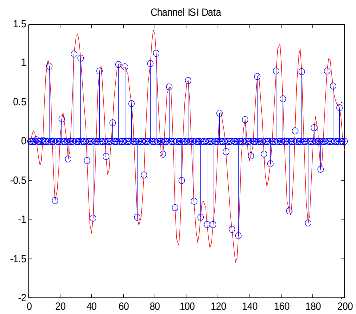


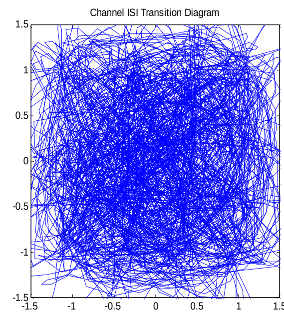


Adaptive Filters & Channel Equalizers 16

16-QAM Channel Equalizer Receiver (Rx)

- Inter-Signal Interference (ISI) data and constellation

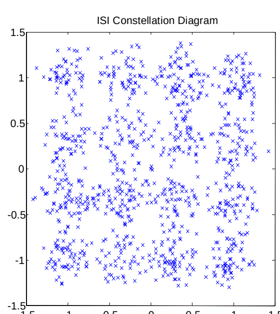


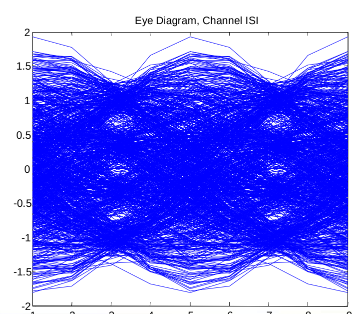


Adaptive Filters & Channel Equalizers 17

Before Equalizer

- Unequalized Rx data sequence and eye diagram
- Observe eye is completely closed
- Rx cannot make symbol decisions

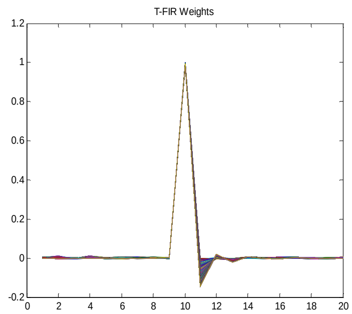


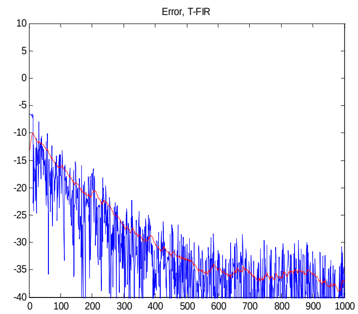


Adaptive Filters & Channel Equalizers 18

Adaptive Equalizer Runs

- Plots show temporal evolution of filter and error function

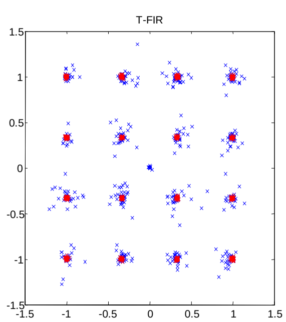


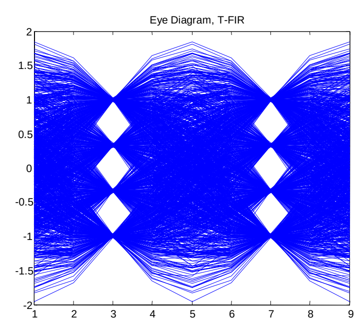


Adaptive Filters & Channel Equalizers 19

After Equalizer

- Equalized data
- Constellation and eye
- Red symbols are with start-up transient removed





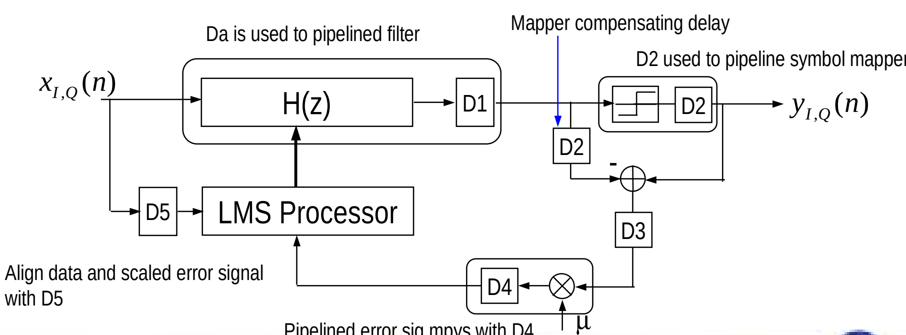
Adaptive Filters & Channel Equalizers 20

The Design Space is Rich

- Decision directed T/2 Adaptive Equalizer - LMS based update
- Using FPGAs There are multiple architectural choices available to meet a desired area/performance objective
- Fully parallel
  - N MAC processing elements (PEs)
  - N LMS PEs
- Folded architecture
  - 1 MAC PE & 1 LMS PE for each polyphase segment
- … Many others

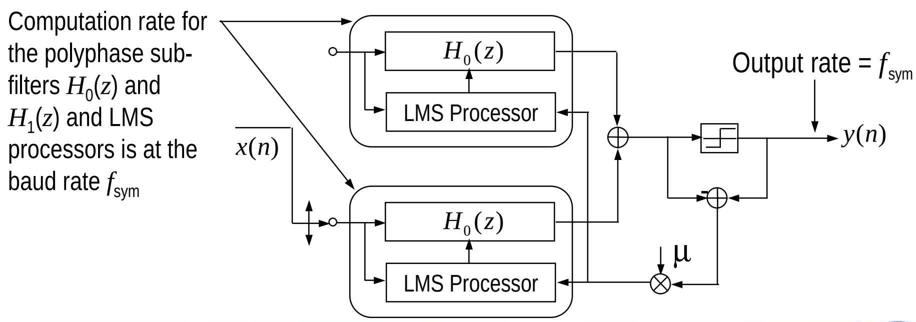
Pipelined Equalizer

- Pipeline datapath to support high data rates
- FFE is relatively straightforward to pipeline
- DFE much more difficult to achieve high-speed operation



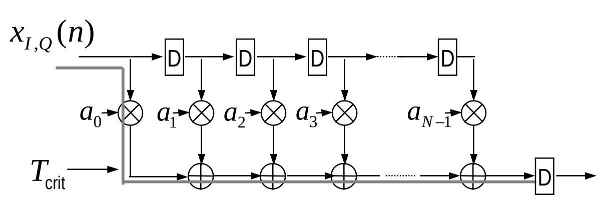
T/2 FSE Implementation

- Data delivered at 2 samples/symbol
- 2-stage polyphase implementation
- All functional units run at the symbol rate



Parallel Implementation

- Sub-filter architecture
- Direct form implementation
- FFE can be pipelined



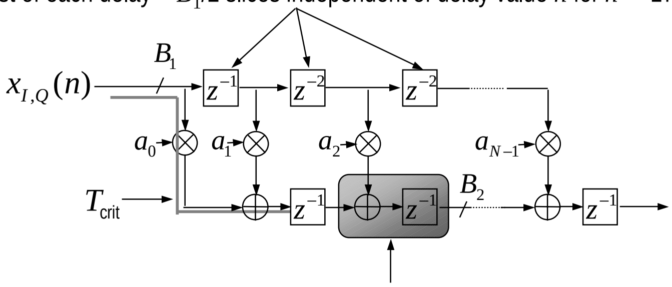
Non-pipelined implementation with critical path $T_{crit} = T_{mpy} + (N-1)T_{add}$

$\boxed{D}$ = symbol period delay

Pipelining (1)

- Pipelined direct form filter

Cost of each delay = $B_1/2$ slices independent of delay value n for n <=17



$B_2/2$ slices since a 1-bit sum/delay can be realized using 0.5 of a slice

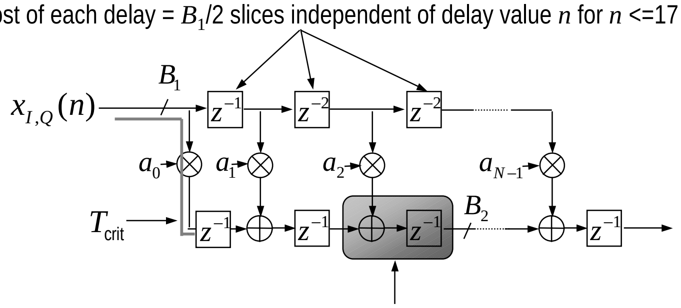
Pipelined implementation with critical path $T_{crit} = T_{mpy} + T_{add}$

Adaptive Filters & Channel Equalizers 25

Pipelining (2)

- Pipelined direct form filter
  - Pipelined multiplier

Cost of each delay = $B_1/2$ slices independent of delay value n for n <=17



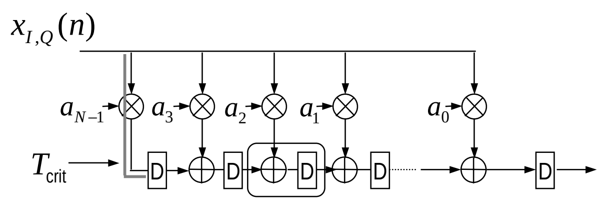
$B_2/2$ slices since a 1-bit sum/delay can be realized using 0.5 of a slice

Pipelined implementation with critical path $T_{crit} = T_{mpy}$

Adaptive Filters & Channel Equalizers 26

Transposed FIR

- Sub-filter architecture
- Transposed FIR implementation
- Pipelining inherent component of transposed filter



Transposed FIR with critical path $T_{\text{crit}} = T_{\text{mpy}}$

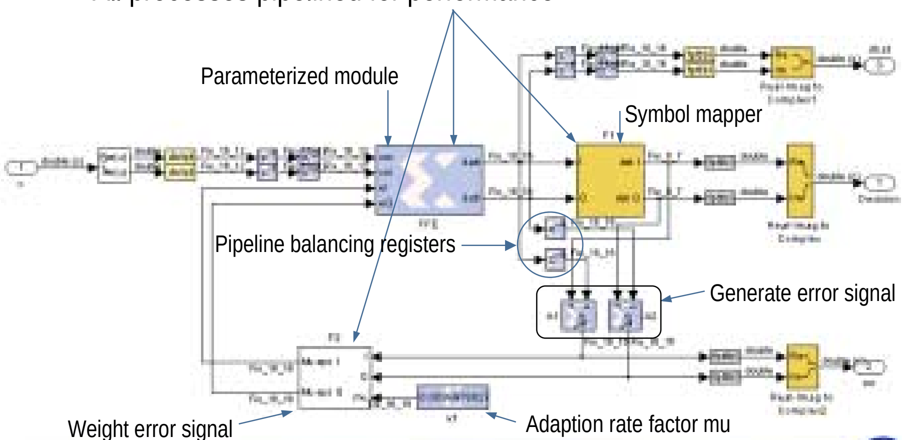
$\boxed{\text{D}}$ = symbol period delay

Implementation

- Parallel T/2 FSE
- Polyphase decomposition
- 8-taps total
  - – 4 taps in each polyphase segment
- 8-LMS PEs
- Coefficients updated at the symbol rate

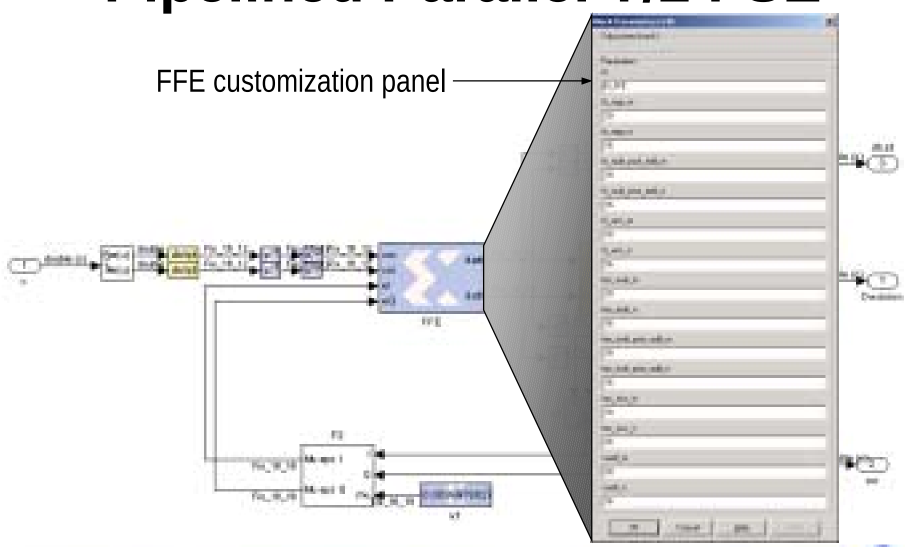
Pipelined Parallel T/2 FSE

- All processes pipelined for performance



Adaptive Filters & Channel Equalizers 29

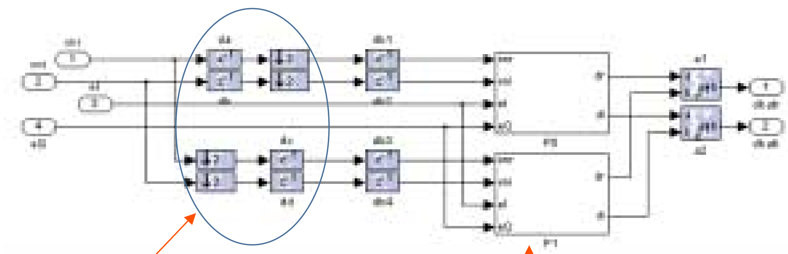
Pipelined Parallel T/2 FSE



Adaptive Filters & Channel Equalizers 30

Pipelined Parallel T/2 DD FSE

- Design components are based on a library of highly optimized module generators



Input sample commutator

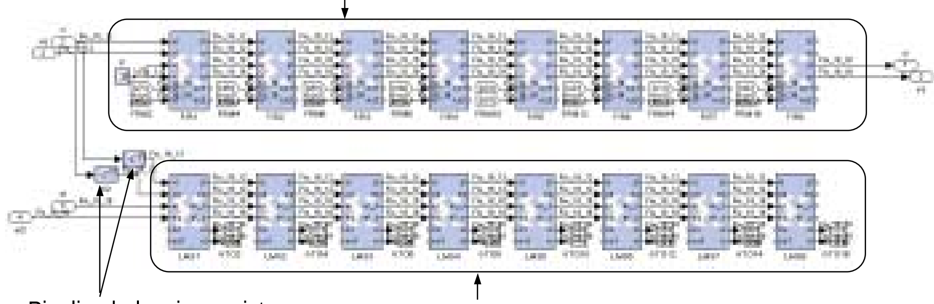
Polyphase Filter produces samples at the symbol rate

Adaptive Filters & Channel Equalizers 31

Pipelined Parallel T/2 FSE

- One polyphase segment

8-tap transposed complex FIR



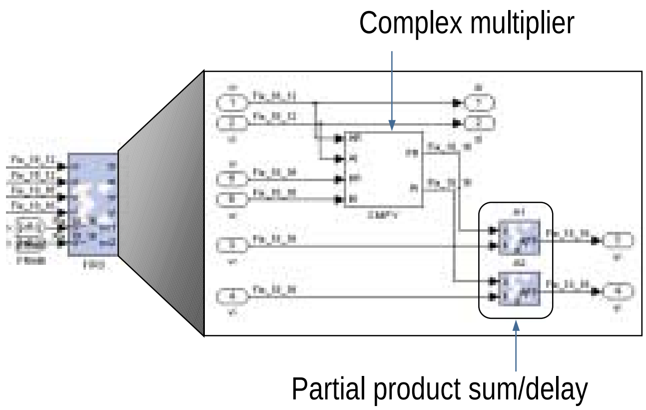
Pipeline balancing registers
To implement delayed LMS update

8 LMS PEs

Adaptive Filters & Channel Equalizers 32

Pipelined Parallel T/2 FSE

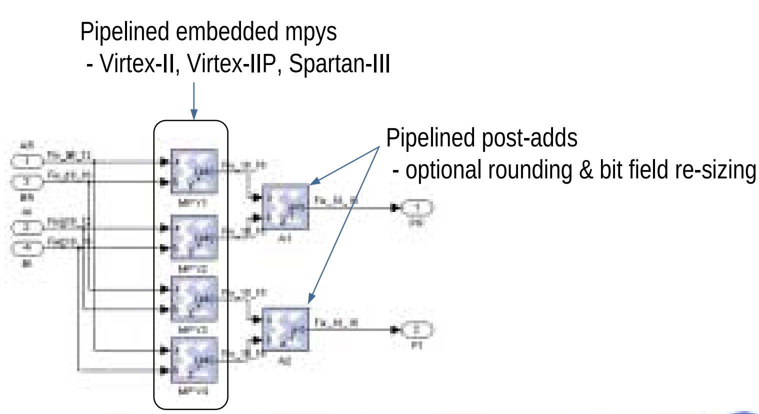
- One tap (complex) of transposed FIR



Adaptive Filters & Channel Equalizers 33

Pipelined Parallel T/2 FSE

- Pipelined complex multiplier

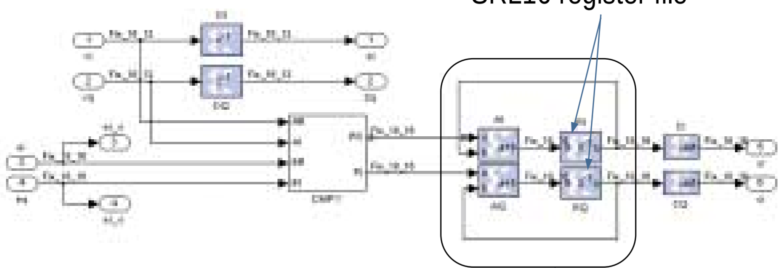


Adaptive Filters & Channel Equalizers 34

Pipelined Parallel T/2 FSE

- LMS PE architecture

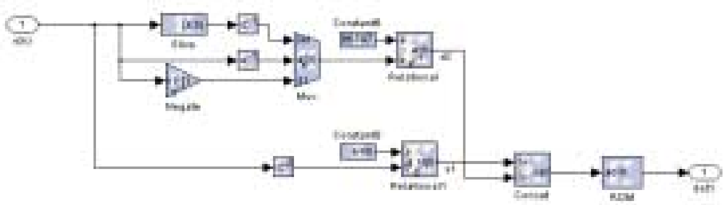
SRL16 register file



Complex accumulator

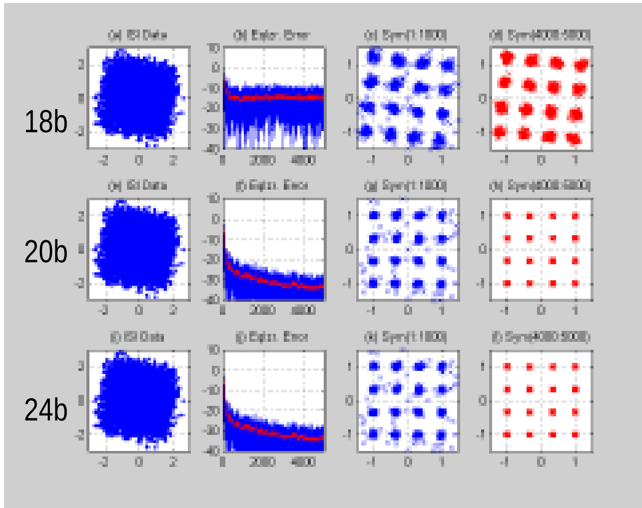
Pipelined Parallel T/2 FSE

- Pipelined symbol mapper (slicer)



Finite Arithmetic Implementation

- Rapid design interations enabled by parameterizing datapath using m-file
- Quickly find optimum (meets performance with minimum area) implementation
- Plot shows three implementations of the equalizer for 18b, 20b and 24b datapath
- Select 20b in this case



Adaptive Filters & Channel Equalizers 37

Pipelined Parallel T/2 FSE

- Benchmark data
  - 8-tap FSE
    - Polyphase implementation 4 taps/filter segment
    - 16-QAM
  - 1037 logic slices
  - 66 embedded multipliers
  - Max fclk † = 240 MHz (XC2VP50ff1148-7)
- For fclk = 240 MHz a symbol rate of 120 Msym/s can be supported
  - For 16-QAM this is a data rate of 480 Mbps

† software version 5.1.03i, speedfile version 1.93, par -ol 5
System Generator v 3.1

Adaptive Filters & Channel Equalizers 38

Pipelined Parallel T/2 FSE

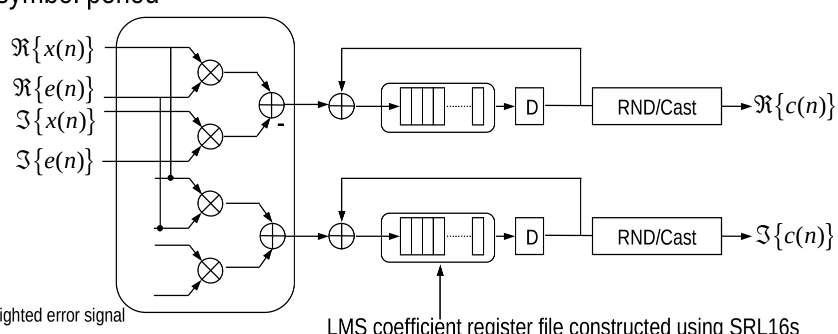
- Benchmark data
  - 16-tap FSE
    - Polyphase implementation 8 taps/filter segment
    - 16-QAM
  - 2332 logic slices
  - 66 embedded multipliers
  - Max fclk † = 225 MHz (XC2VP50ff1148-7)
- For fclk = 225 MHz a symbol rate of 112.5 Msym/s can be supported
  - For 16-QAM this is a data rate of 450 Mbps
  - For 64-QAM this is a data rate of 675 Mbps
  - For 256-QAM this is a data rate of 900 Mbps
- – Computation rate = ~15 Giga-MACs/second

† software version 5.1.03i, speedfile version 1.93, par -ol 5
System Generator v 3.1

Adaptive Filters & Channel Equalizers 39

Folded FSE

- Diagram of the implementation showing the filter and LMS engines in each of the two polyphase arms
- LMS PE clocked at M x symbol rate to update all coefficients in a symbol period



$e(n)$ = weighted error signal
$x(n)$ = regressor vector samples
$c(n)$ = filter coefficients

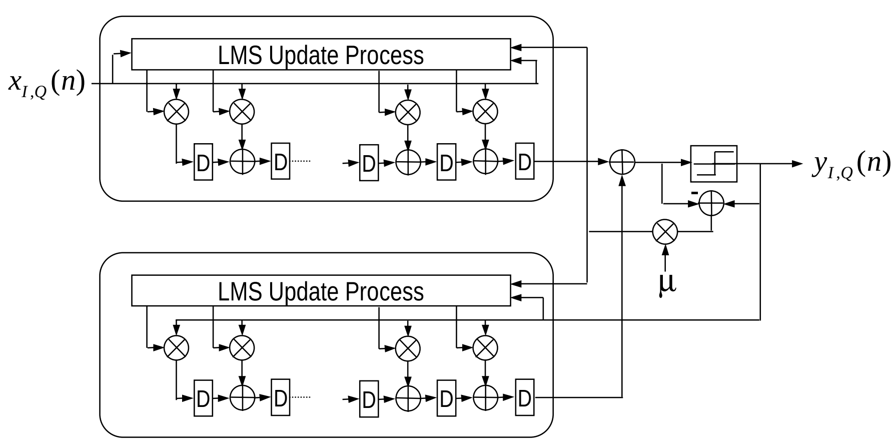
Adaptive Filters & Channel Equalizers 40

Folded FSE

- Benchmark data
  - 8-tap FSE
    - Polyphase implementation 4 taps/filter segment
    - 16-QAM
  - 833 logic slices
  - 18 embedded multipliers
  - Max fclk † = 215 MHz (XC2VP50ff1148-7)
- For fclk = 200 MHz a symbol rate of 20 Msym/s can be supported
  - For 16-QAM this is a data rate of 100 Mbps

† software version 5.1.03i, speedfile version 1.93, par -ol 5
System Generator v 3.1

Decision Feedback Equalizer



DFE

- While FFE is relatively straightforward to pipeline DFE is not due to its recursive structure
- Standard techniques (e.g. [1] [2]) for pipelining recursive structures do not work since the data decision is a non-linear process

[1] K. K. Parhi and M. Hatamian, "A High Sample Rate Recursive Filter Digital Filter Chip", *VLSI Signal Processing III*, IEEE Press New York, 1988.

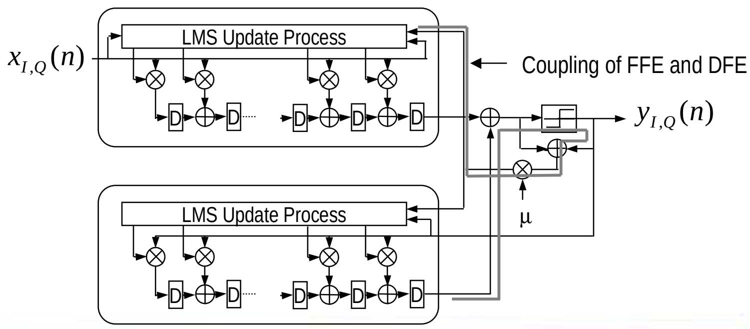
[2] K. K. Parhi , *VLSI Digital Signal Processing Systems,* Wiley, 1999.

DFE: Finite Arithmetic Considerations (1)

- Special attention must be given to finite arithmetic implementations of a DFE
  - More care than required for implementation of a fractionally-spaced FFE
- Rounding/Truncation
  - Generally rounding arithmetic is preferred for the implementing the DFE
  - The bias introduced by a datapath employing truncation arithmetic will eventually cause an incorrect decision at the slicer potentially causing a catastrophic failure of the entire equalizer

DFE: Finite Arithmetic Considerations (2)

- Feedforward and recursive sections of the equalizer are coupled by the error signal
- The datapath precision employed for a stable FFE-only structure may not be adequate when this same FFE is extended using decision feedback



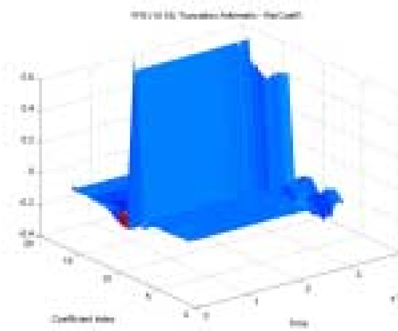
Adaptive Filters & Channel Equalizers 45

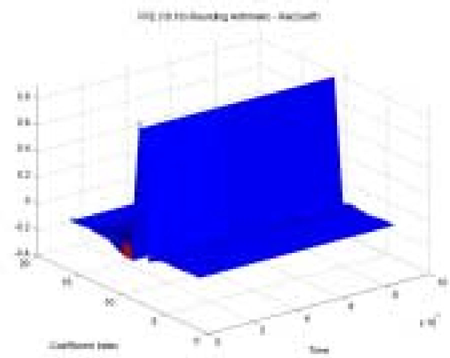
DFE: Finite Arithmetic Considerations (3)

- The next sequence of 4 mesh plots show the evolution with time of the coefficients in the FFE and DFE for stable (at least within the simulation epoch) and unstable implementations
- Unstable implementation employs truncation arithmetic
- Stable implementation employs truncation arithmetic

Adaptive Filters & Channel Equalizers 46

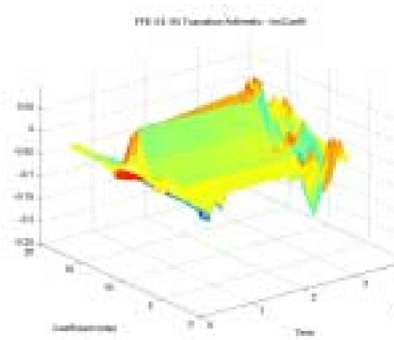
FFE/DFE: Trnc Vs Rnd

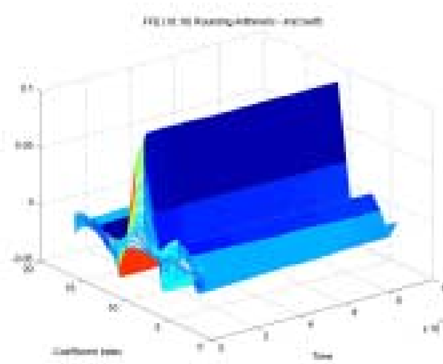




Adaptive Filters & Channel Equalizers 47

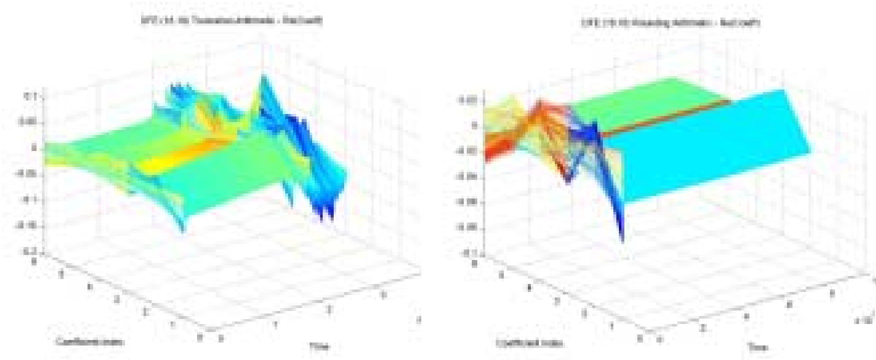
FFE/DFE: Trnc Vs Rnd





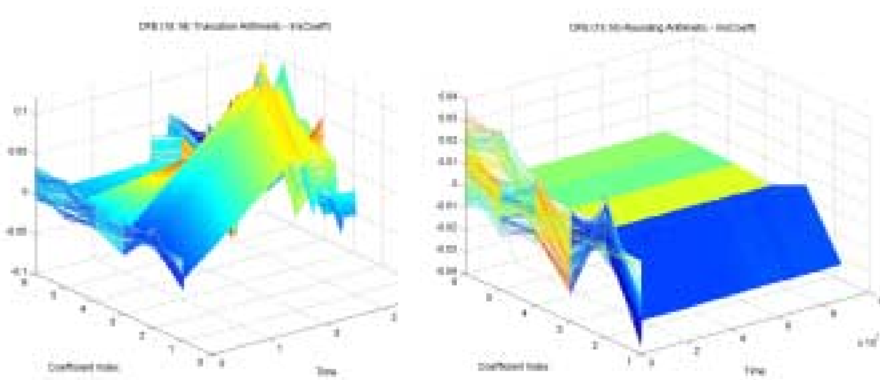
Adaptive Filters & Channel Equalizers 48

FFE/DFE: Trnc Vs Rnd



Adaptive Filters & Channel Equalizers 49

FFE/DFE: Trnc Vs Rnd



Adaptive Filters & Channel Equalizers 50

DFE: Finite Arithmetic Considerations (3)

- Many millions of input samples may need to be processed for the FFE/DFE equalizer to become unstable due to finite arithmetic effects
- This can equate to days even weeks of simulation time
- Hardware platform is extremely useful for doing rapid design turns

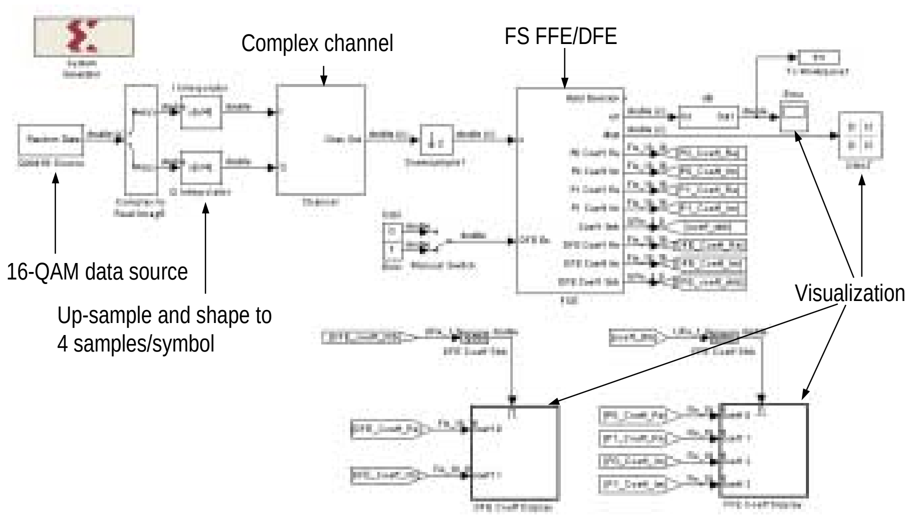
DFE: Finite Arithmetic Considerations (4)

- The hypothesized stable equalizer in the previous simulation coefficient evolution surfaces was not in fact stable
- A Simulink simulation of the equalizer was run for 24 hours and maintained stability
  - ~1 million samples processed
- When the design was ported to a hardware platform the equalizer failed immediately
  - Hardware platform in this case was the Xilinx/Nallatech DSP kit using System Generator for DSP (hardware-in-the-loop) to generate the implementation

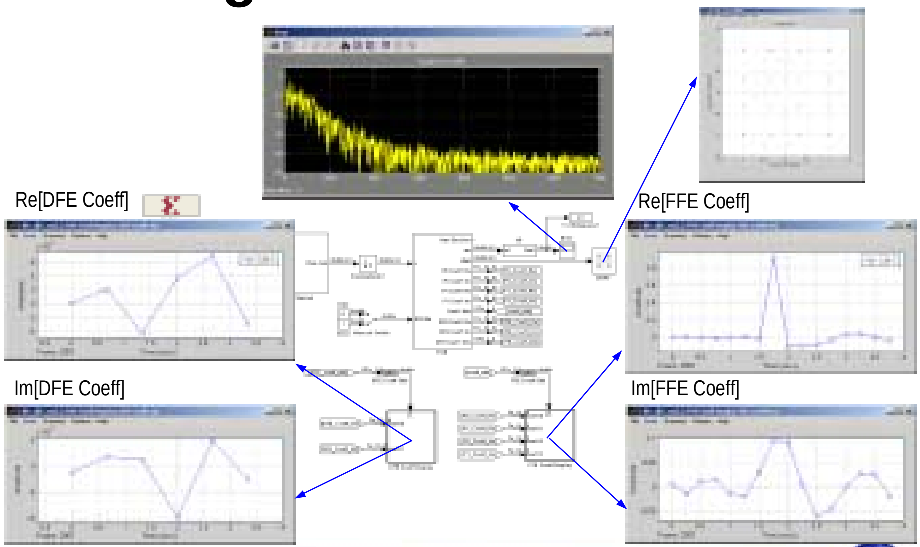
DFE: Finite Arithmetic Considerations (5)

- The datapath in the FFE and DFE was extended to a 28.24 implementation and was run for 1 week and was found to be stable

Equalizer Testbench

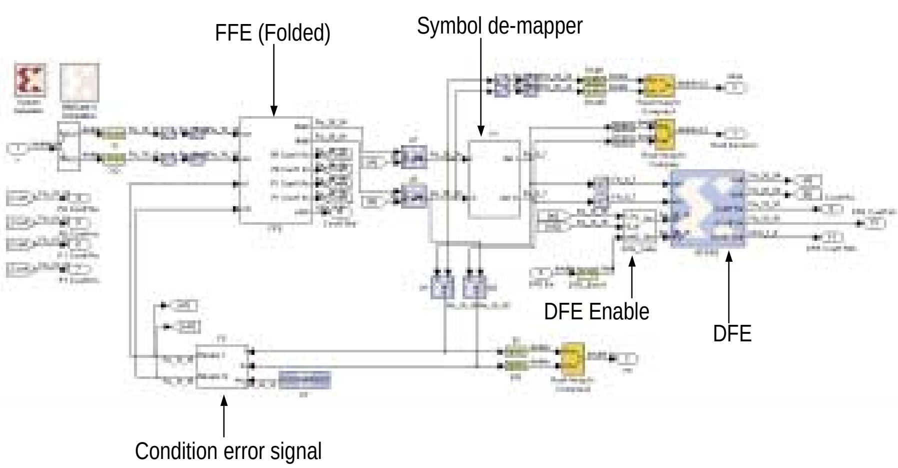


Design Data Visualization

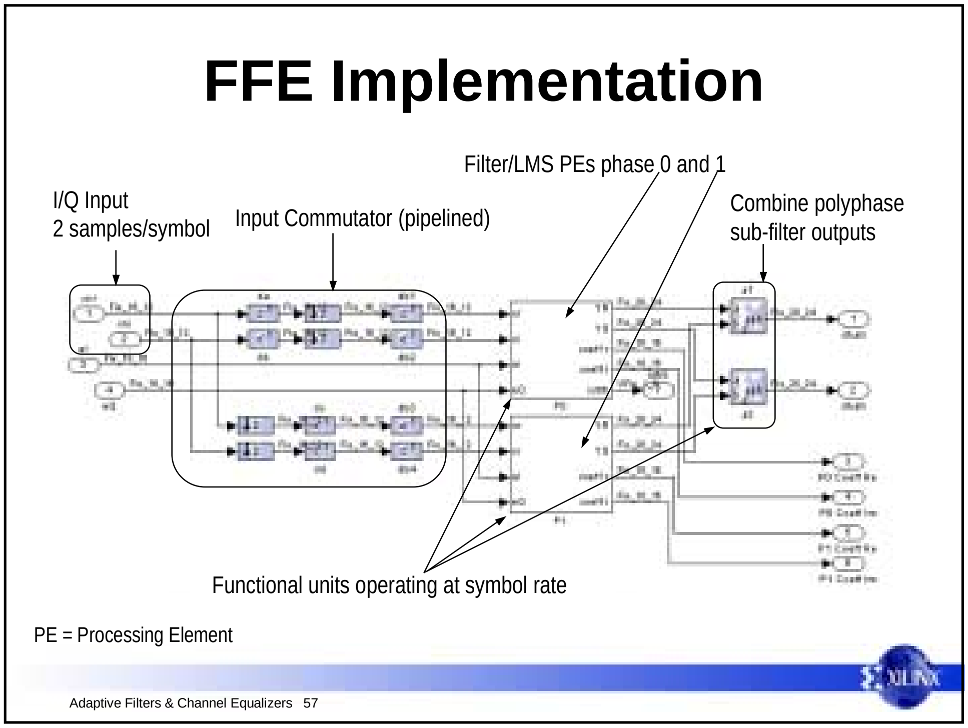


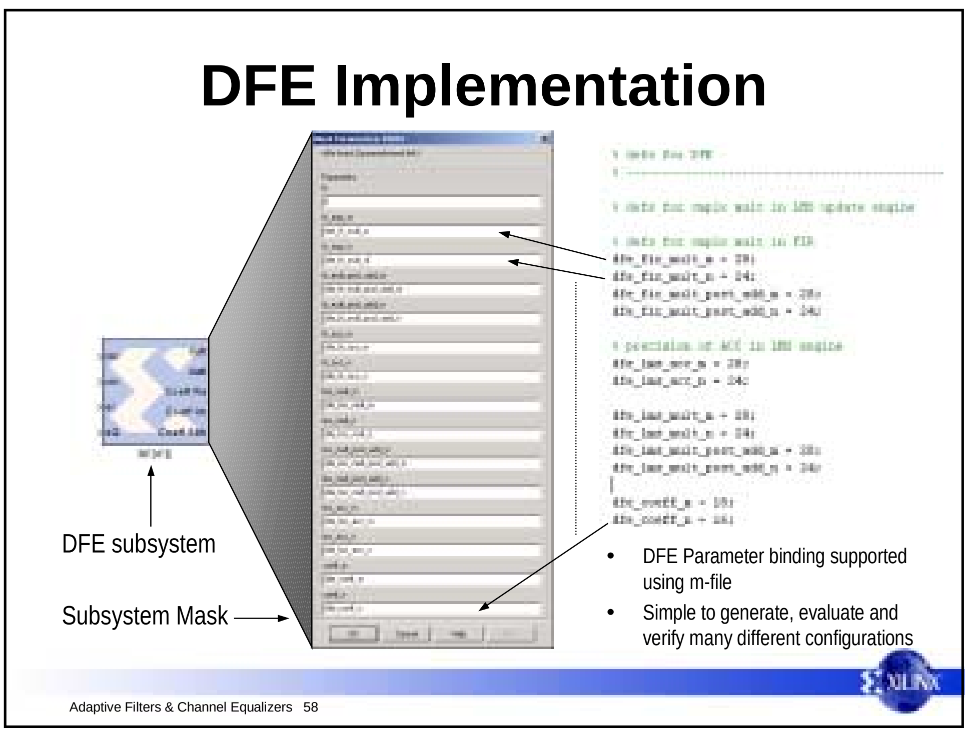
Adaptive Filters & Channel Equalizers 55

FFE/DFE



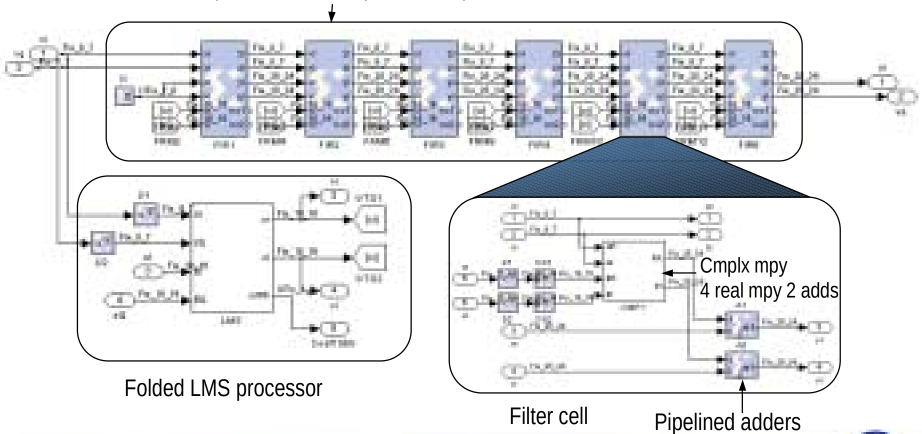
Adaptive Filters & Channel Equalizers 56


FFE Implementation
Filter/LMS PEs phase 0 and 1
I/Q Input
2 samples/symbol
Input Commutator (pipelined)
Combine polyphase
sub-filter outputs
Functional units operating at symbol rate
PE = Processing Element
XILINX
Adaptive Filters & Channel Equalizers 57


DFE Implementation
DFE subsystem
Subsystem Mask
DFE Parameter binding supported using m-file
Simple to generate, evaluate and verify many different configurations
XILINX
Adaptive Filters & Channel Equalizers 58

DFE Implementation

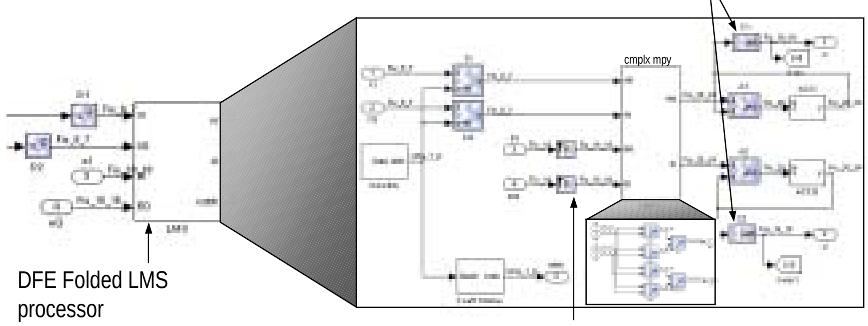
- Transposed filter
- Parallel filter architecture
- Number of taps defined via subsystem mask parameter



Adaptive Filters & Channel Equalizers 59

DFE Implementation

Round LMS acc. to field 18b width to minimize number of mpys used in filter implementation



Upsample by M (DFE length) so that LMS PE runs at 6x symbol rate
All taps updated at symbol rate

Adaptive Filters & Channel Equalizers 60

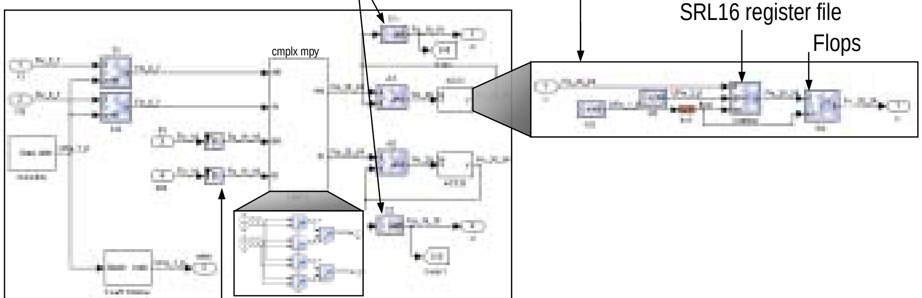
DFE Folded LMS PE

Round LMS acc. To field 18b width so minimize number of mpys used in filter implementation

SRL16-based accumulator bank

SRL16 register file

Flops

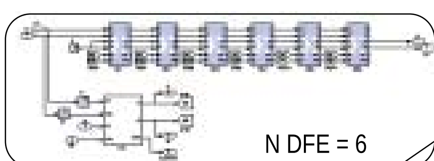


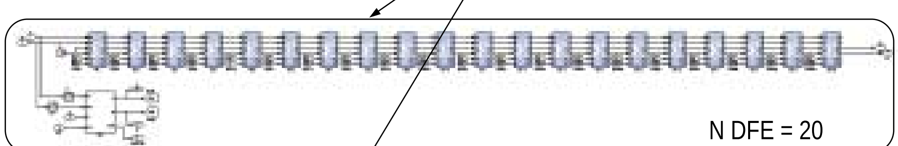
Upsample by M (DFE length) so that LMS PE runs at 6x symbol rate
All taps updated at symbol rate

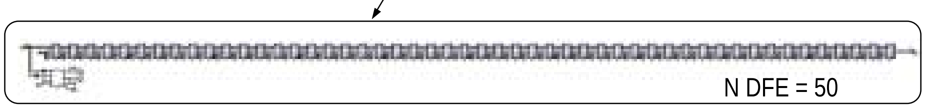
Adaptive Filters & Channel Equalizers 61

Parameterized Graph

- Modify one subsystem mask parameter to generate different implementations
- No manual re-drawing of graph







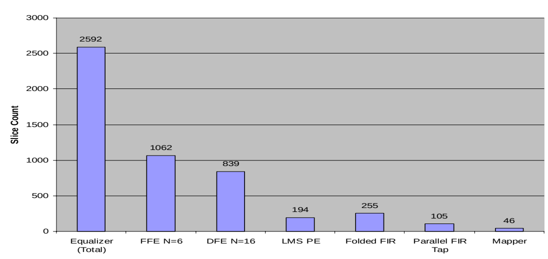
Adaptive Filters & Channel Equalizers 62

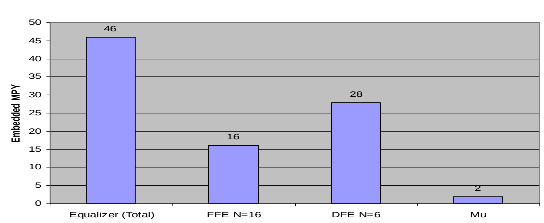
FFE/DFE Resource Utilization

- Benchmark data
  - 16-QAM
  - 16-tap FSE
    - Polyphase implementation 8 taps/filter segment
  - 6-tap DFE
  - 2592 logic slices
  - 46 embedded multipliers
  - Max fclk † = MHz (XC2VP50ff1148-7)
- For fclk = MHz a symbol rate of 20 Msym/s can be supported
  - For 16-QAM this is a data rate of Mbps

Adaptive Filters & Channel Equalizers 63

FFE/DFE Resource Utilization





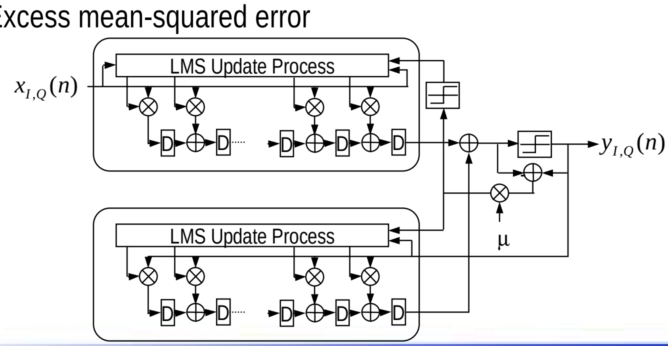
Adaptive Filters & Channel Equalizers 64

Extensions to LMS Algorithm

- Basic LMS
- normalized LMS (NLMS)
- signed data LMS (SD_LMS)
- signed error LMS (SE_LMS)
- signed data signed error LMS (SDSE_LMS)
- Gear-shifting algorithms

Signed Data LMS

- Minimize resource utilization
  - Primarily embedded multipliers
- Sacrifice
  - Rate of convergence
  - Excess mean-squared error



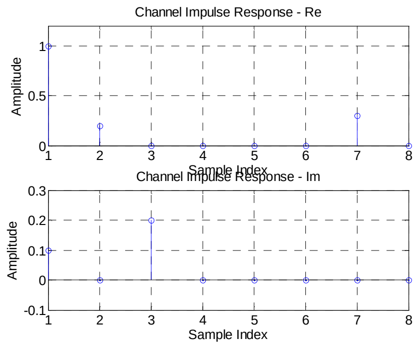
Signed Data FFE/DFE

- Benchmark data
  - Signed-data update used only in FFE
  - 16-QAM
  - 16-tap FSE
    - Polyphase implementation 8 taps/filter segment
  - 6-tap DFE
  - 2412 logic slices
  - 38 embedded multipliers
  - Max fclk † = MHz (XC2VP50ff1148-7)
- For fclk = MHz a symbol rate of 20 Msym/s can be supported
  - For 16-QAM this is a data rate of Mbps

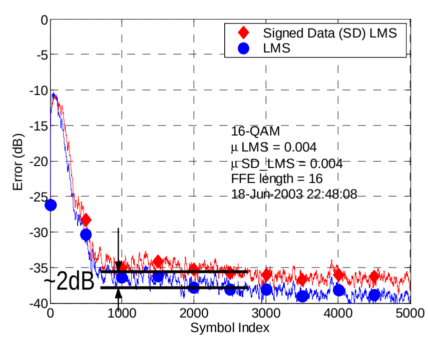
Adaptive Filters & Channel Equalizers 67

Convergence Comparison

- Complex channel time response



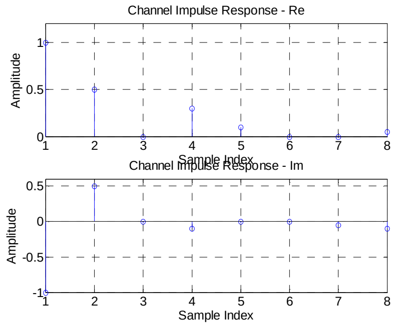
- FFE Only
- Compare smoothed error signal (dB)
  - Signed data LMS
  - LMS



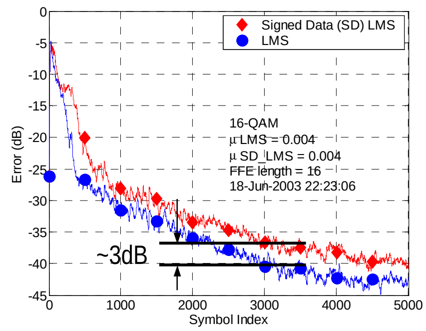
Adaptive Filters & Channel Equalizers 68

Convergence Comparison

- Complex channel time response



- FFE Only
- Compare smoothed error signal (dB)
  - Signed data LMS
  - LMS



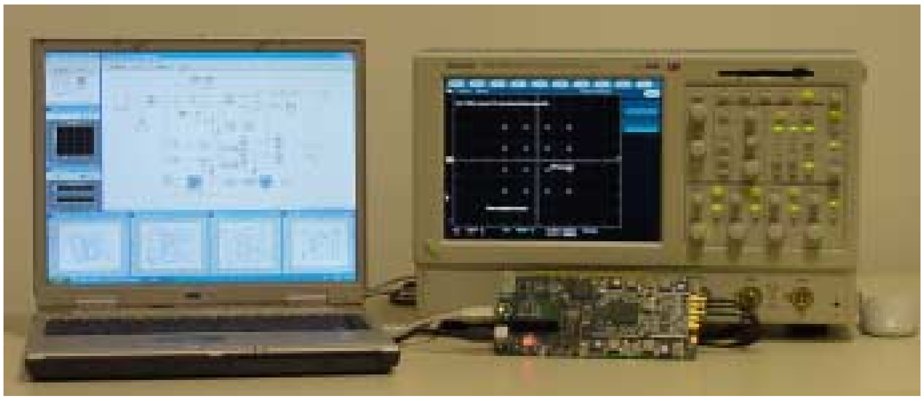
Adaptive Filters & Channel Equalizers 69

Xilinx DSP Eval Board

- Virtex-II (XC2V1000/3000)
- Dual A/D D/A
- PCI & USB interface to host system

Adaptive Filters & Channel Equalizers 70

QAM Demod Simulation



Extensions to LMS Algorithm

- Basic LMS
- normalized LMS (NLMS)
- signed data LMS (SD_LMS)
- signed error LMS (SE_LMS)
- signed data signed error LMS (SDSE_LMS)
- Gear-shifting algorithms

Normalized LMS (1)

- In the standard LMS algorithm the adaptation constant μ determines the convergence of the algorithm
- One practical problem confronted in the choice of μ is to ensure that it does not become large enough to impact the algorithm stability
- The largest value of μ is determined by the largest eigen value of the autocorrelation matrix $\mathbf{R}$

Adaptive Filters & Channel Equalizers 73

Normalized LMS (2)

- However $\mathbf{R}$ is not typically available and even if it were computing its eigen values would by computationally expensive
- Reasonable approach is to find a bounds for the largest eigenvalue
- It can be shown that

$$\text{Avrg}\left\{\mathbf{X}^t(k)\mathbf{X}(k)\right\} = \sum_{i=1}^{N} \lambda_i \quad \text{where } \lambda_i \text{ is the } i\text{'th eigenvalue}$$

Adaptive Filters & Channel Equalizers 74

Normalized LMS (3)

- Because all of the eigenvalues are non-negative

$$\text{Avrg}\left\{\mathbf{X}^t(k)\mathbf{X}(k)\right\} = \sum_{i=1}^{N} \lambda_i > \lambda_{\max}$$

meaning the average value of the inner product is an upper bound to $\lambda_{\max}$ - the largest eigenvalue

Normalized LMS (4)

- This suggests defining μ as

$$\mu(k) = \frac{\alpha}{\mathbf{X}^{\mathrm{T}}(k)\mathbf{X}(k)}$$

where α is a positive constant chosen to be between 0 and 2

- Using this form for μ results in the *Normalized LMS* algorithm

Normalized LMS (5)

- Using this form for μ results in the *Normalized LMS* algorithm

$$y(k) = \mathbf{W}^{\mathsf{T}}(k)\mathbf{X}(k)$$
$$e(k) = d(k) - y(k)$$
$$\mathbf{W}(k+1) = \mathbf{W}(k) + \frac{\alpha e(k)\mathbf{X}(k)}{\gamma + \mathbf{X}^{\mathsf{T}}(k)\mathbf{X}(k)}$$

- α is the new “normalized” adaptation constant, while γ is a small positive term included to ensure that the update term does nor become excessively large should $\mathbf{X}^{\mathsf{T}}(k)\mathbf{X}(k)$ temporarily become small

Normalized LMS (6)

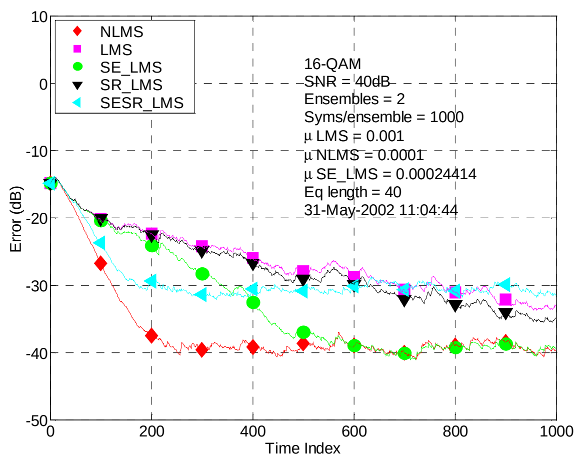
- At first it may appear that inclusion of the term $\mathbf{X}^{\mathsf{T}}(k)\mathbf{X}(k)$ in the denominator increases the computation requirement by another N multiplications and additions but this can be avoided if N extra storage locations are available
- At time k, $\mathbf{X}^{\mathsf{T}}(k)\mathbf{X}(k)$ is given by

$$\mathbf{X}^{\mathsf{T}}(k)\mathbf{X}(k) = \sum_{i=0}^{N-1} x^2(k-i)$$

Normalized LMS (7)

- The term $\mathbf{X}^T(k+1)\mathbf{X}(k+1)$ can be computed by adding in $x^2(k+1)$ and subtracting $x^2(k-N+1)$
- By storing the intermediate values of $x^2(.)$ the computation required to update the inner product is reduced to a squaring, an addition, and a subtraction
- Note that a division operation is required

Normalized LMS (8)

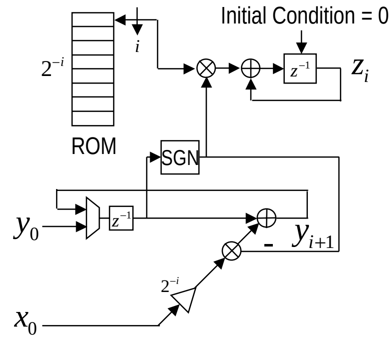


NLMS Implementation (9)

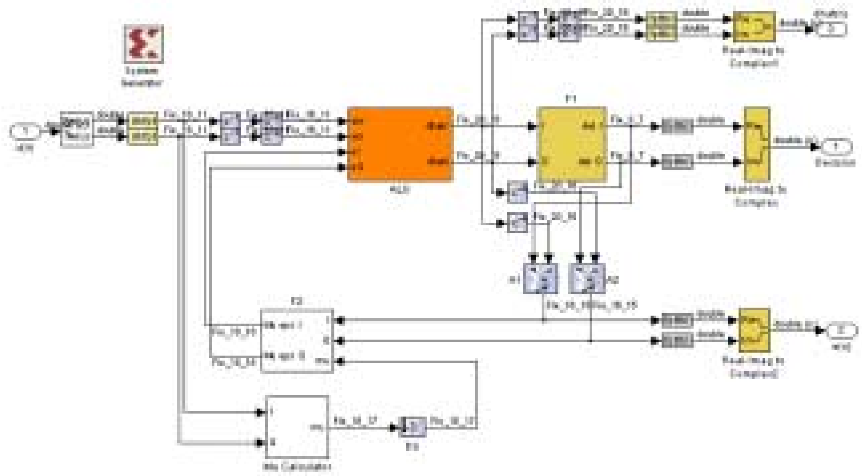
- The main computation to address in the NLMS is an effective method for computing the division
- The linear rotation mode of the CORDIC algorithm is useful for this operation

NLMS Implementation (10)

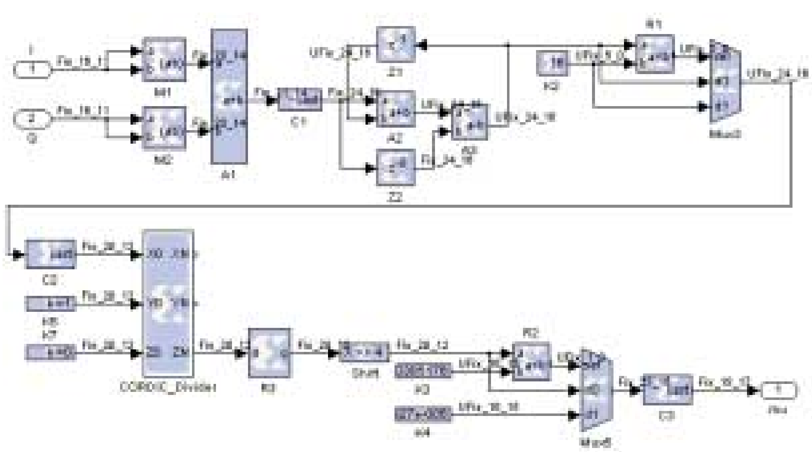
- Linear CORDIC datapath



NLMS Implementation



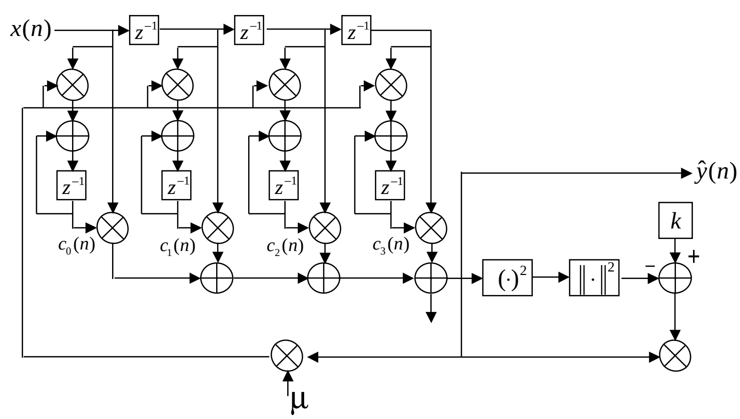
Computing μ



Blind Equalizer

- Initially receiver will not have locked
  - frequency or Phase locked loops
- FSE & DFE are decision directed procedures that will not function correctly in the presence of Doppler
- Need a non decision directed equalizer to acquire the channel when a link is opened
  - Blind equalizer

Blind Equalizer



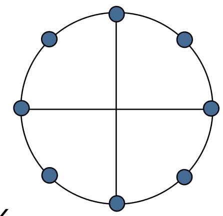
Blind Equalizer Adaptation

- Need a new process for adapting the coefficients of the blind equalizer
- Particularly successful method is the *constant modulus algorithm* (CMA)
- CMA equalizer utilizes knowledge of structure in the constellation
  - e.g. for QPSK the constellation points are on a circle

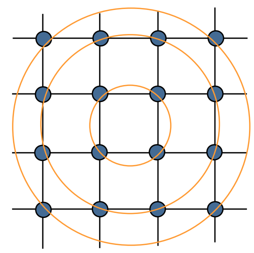
CMA Equalizer

- *Constant modulus algorithm* (CMA) seeks to minimize a cost defined by the Constant Modulus(CM) criterion
- CM criterion penalizes deviations in the modulus of the equalized signal away from a fixed value

CMA Equalizer



8-PSK
constant modulus alphabet



16-QAM
non-constant modulus alphabet

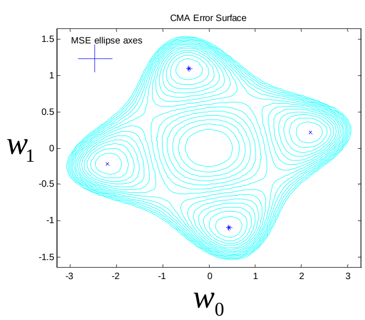
- Remarkably the CM criterion can successfully equalize signals characterized by source alphabets *not* possessing a constant modulus (e.g. 16-QAM)

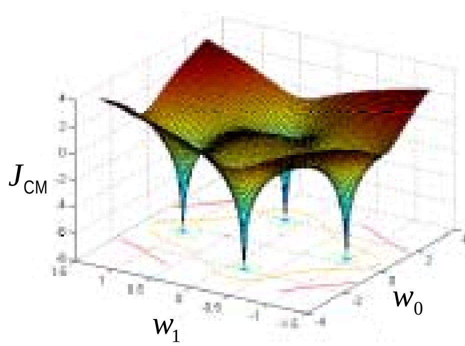
Adaptive Filters & Channel Equalizers 89

CMA Equalizer

CMA performance surface

chan impulse response = [0.2 0.5 1.0 -0.1], BPSK alphabet, 100 dB SNR, T/2 spaced

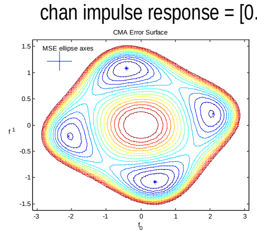


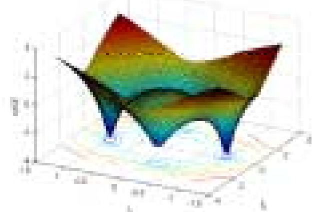


Adaptive Filters & Channel Equalizers 90

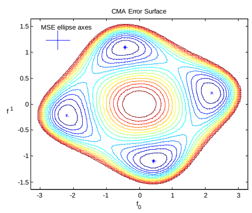
CMA Equalizer

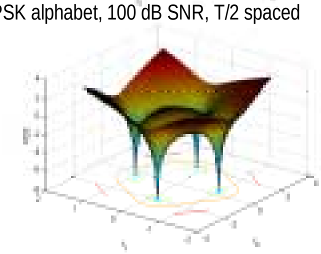
chan impulse response = [0.2 0.5 1.0 -0.1], QPSK alphabet, 20 dB SNR, T/2 spaced



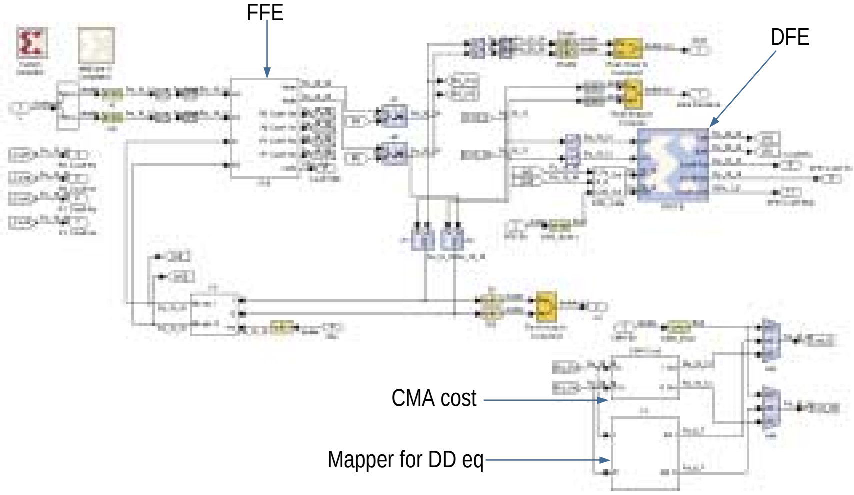


chan impulse response = [0.2 0.5 1.0 -0.1], QPSK alphabet, 100 dB SNR, T/2 spaced





EQ: CMA/FFE/DFE



Burst Equalizer

- When the data packet length is short there may not be enough samples for the equalizer to acquire the channel
- Equalize the data using an iterative process
- Run the equalizer over the input data using the coefficient set from iteration i as the initial condition for iteration $i+1$