Mapping DSP Algorithms Into FPGAs

Sean Gallagher, Senior DSP Specialist,
Xilinx Inc.
seang@xilinx.com
215-990-4616

Agenda

- History of Algorithm implementations in FPGAs
- Why FPGAs for Signal Processing
- Overview of Xilinx FPGA
- Interesting Algorithms for FPGA implelentation
  - Critically sampled channlizer
  - Divide and Conquer DFT
  - Winograd FFT
- The Xilinx DSP tool flow

History FPGAs for Algorithm Implementation

- **Systolic Array processing techniques were established in the '70s**
  - S.Y. Kung, others
- **FPGA technology invented by Xilinx in 1984**
  - Glue logic integration
  - Super Computing Research Center (SRC) built Splash I and II coprocessing boards in early '90s
  - Board of 32 Xilinx FPGAs slaved to a Sun workstation
  - Computation speeds of 6-7 times greater than a Cray II computer

My History With FPGAs

- **Visited SRC in early ‘90s to sell synthesis tools**
  - Had no clue what they were talking about
- **Pursued MSCE at Villanova focused on algorithms in FPGAs**
  - They had no idea what I was talking about
  - Master’s thesis in ‘95, Implementing Algorithms in FPGAs
- **Came to Xilinx in 2001 as DSP Specialist**
  - Still learning

Emerging Applications Drive Demand for Next Generation FPGAs

Next Gen Wireless Communications

Next Gen Wired Communications

Automotive Infotainment

•Consumer

Lowest Power and Cost	Industry's Best Price-Performance	Industry's Highest System Performance and Capacity
• Handheld portable ultrasound • Digital SLR lens control module • Software defined radio	• Wireless LTE infrastructure • 10G PON OLT line card • LED backlit and 3D video displays • Medical imaging • Avionics imaging	• 100GE line card • 300G bridge • Terabit switch fabric • 100G OTN • MUXPONDER • RADAR • ASIC emulation • Test & Measurement

•Aerospace
•& Defense

•Test & Measurement

Medical Imaging

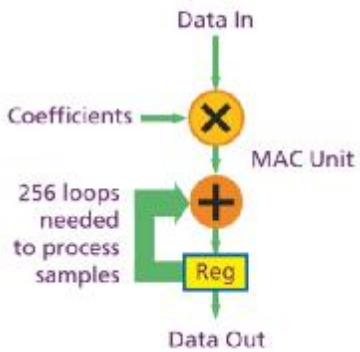
Audio Video Broadcast

XILINX

Why FPGA for Signal Processing?

256-tap Filter Example

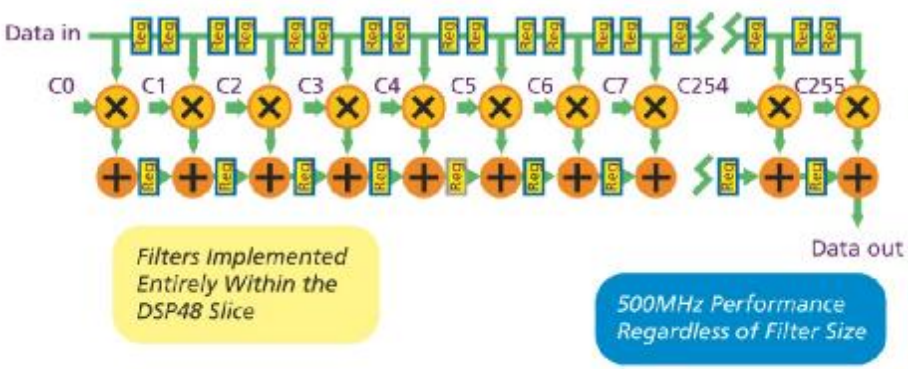
Conventional DSP Processor – Serial implementation



$$\frac{1\ \text{GHz}}{256\ \text{clock cycles}} = 4\ \text{MSPS}$$

- How much computational power do you need?

Virtex-4 Parallel Implementation Consumes Zero Logic Resources



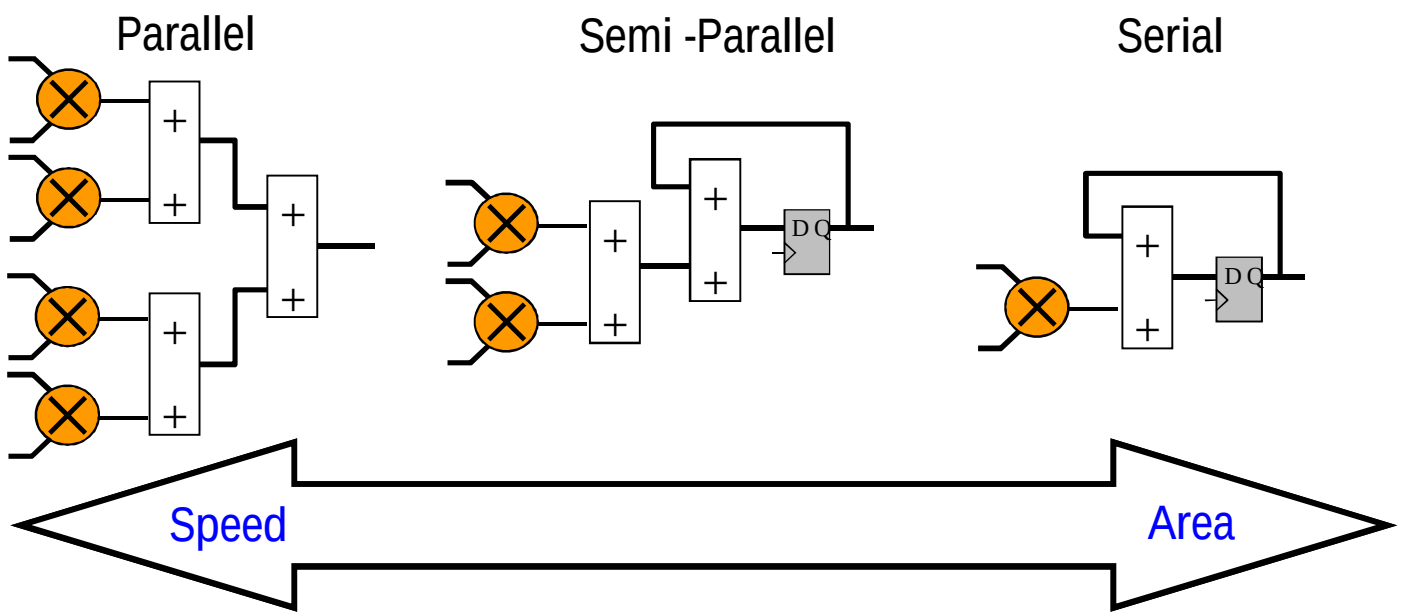
$$\frac{500\ \text{MHz}}{1\ \text{clock cycle}} = 500\ \text{MSPS}$$

XILINX

Flexibility

- How many MACs (multiply accumulator) do you need?
- For Example, in FIR Filter,

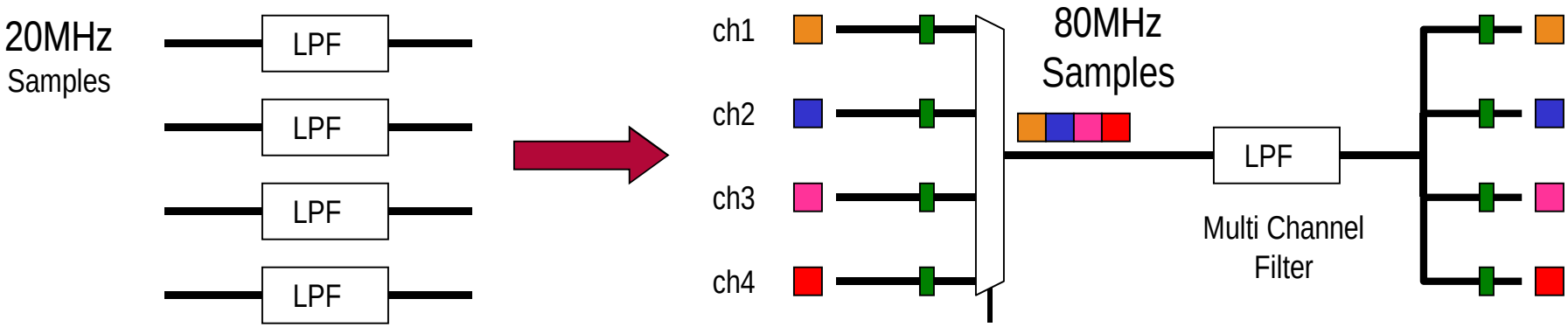
$$\text{Number of MACs required} = \frac{OutputDataRate * NumberOfTaps * NumberOfChannels}{InputDataRate * ClockRate}$$



FPGAs can meet various throughput requirement

XILINX

“Multi-Channel Friendly”

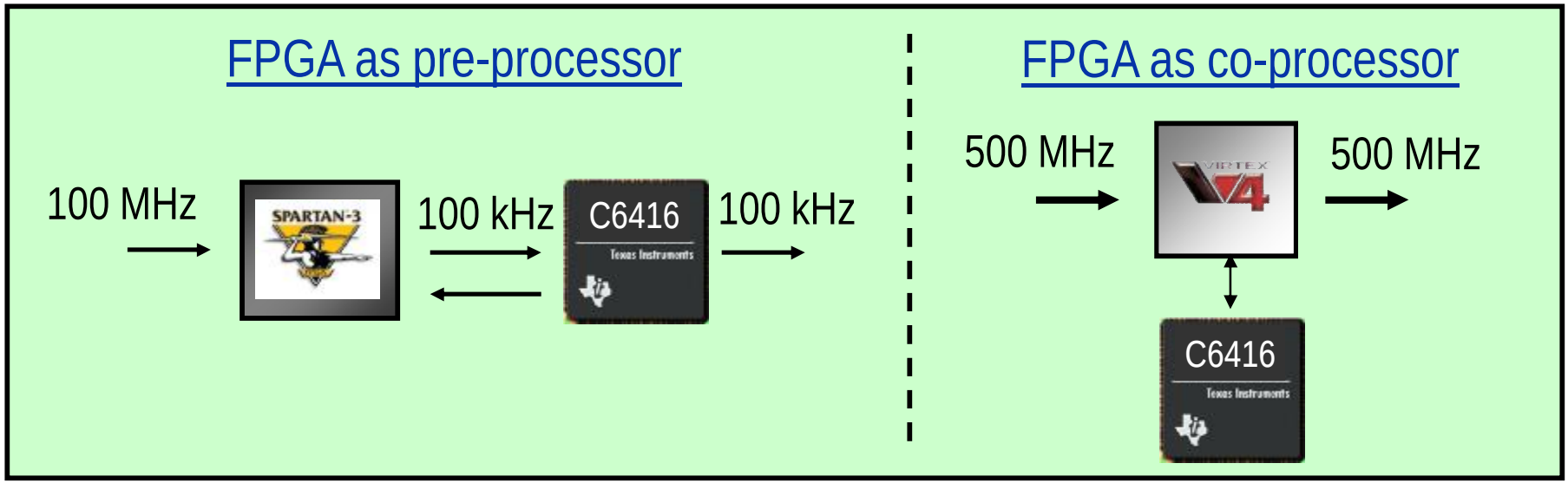


- Parallelism enables efficient implementation of multi-channel into a single FPGA
- Many low sample rate channels can be multiplexed (e.g. TDM) and processed in the FPGA, at a higher rate
- Many of Xilinx IPs takes advantage of multi-channel implementation - FIRCompiler, FFT

XILINX

FPGA + DSP Processor

- § FPGA enables DSP processor acceleration – mapping speed critical loop of DSP code to FPGA
- § FPGAs enables consolidation of glue logic, memory, interfaces, ASSP
- § For detail on interface (EMIF,VLYNQ,LinkPort), see http://www.xilinx.com/esp/wireless.htm



XILINX

6 Series Xilinx FPGAs

•*Now Shipping*

§ Virtex-6 - Industry leading DSP performance

§ Spartan-6 Industry leading DSP cost / performance

	XILINX SPARTAN 6	VIRTEX 6
	Industries Best Price/Performance	Industries Highest System Performance
Logic Cells	**3.8K – 147K**	**74K – 567K**
DSP Slices	**8-180**	**288-2016**
Max Transceivers	**8**	**72**
Transceiver Performance	**3.125 Gbps**	**6.6 Gbps 11.18 Gpbs**
Memory	**4,824 Kbits**	**38,309 Kbits**
Max. SelectIO	**576**	**1200**
SelectIO Voltages	**1.2v to 3.3v**	**1v to 2.5v**

XILINX

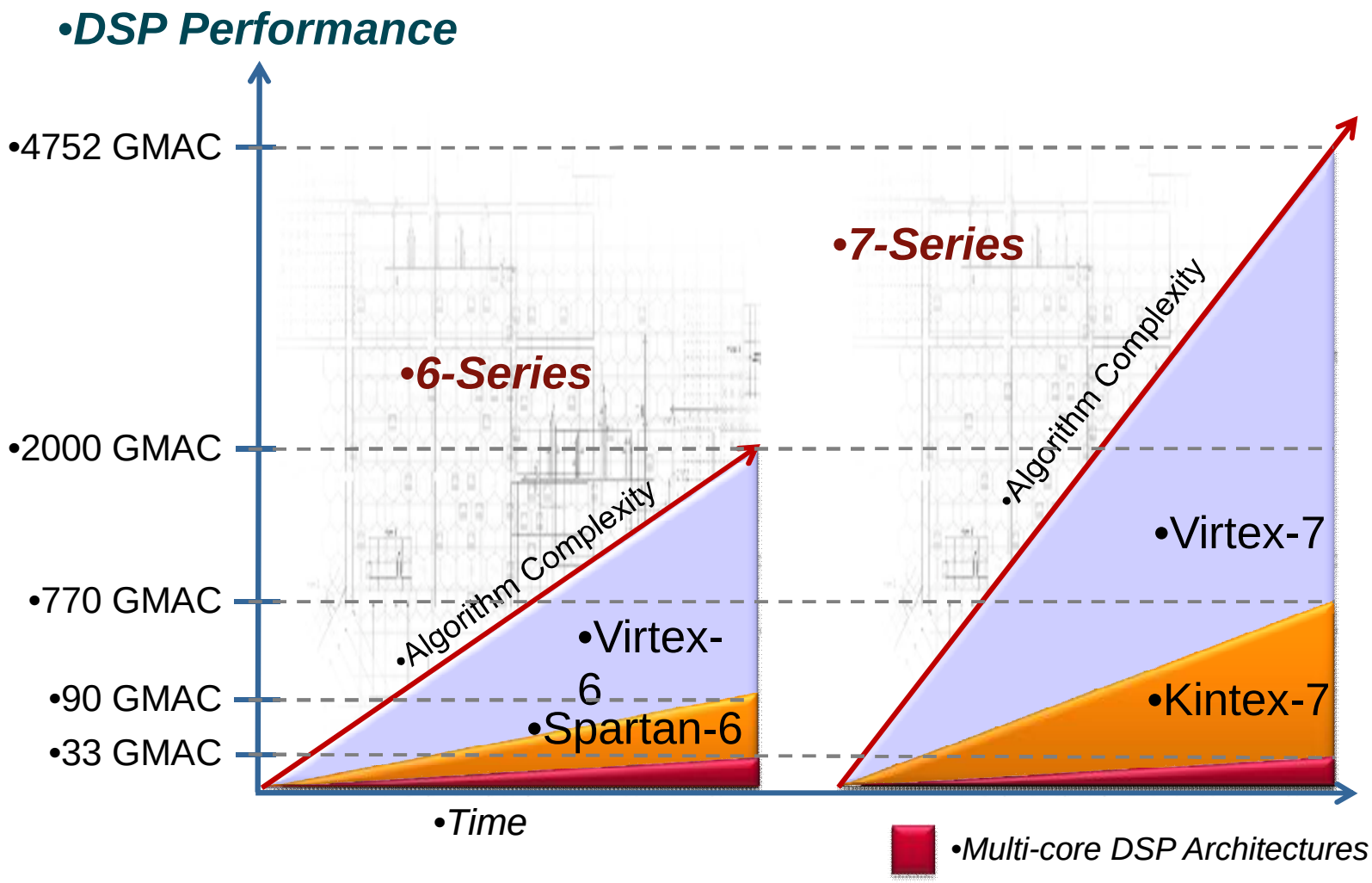
Introducing the 7 Series FPGAs

- **Industry's Lowest Power and First Unified Architecture**
  - Spanning Low-Cost to Ultra High-End applications
- **Three new device families with breakthrough innovations in power efficiency, performance-capacity and price-performance**

	ARTIX 7	KINTEX 7	VIRTEX 7
	Lowest Power & Cost	Industry's Best Price/Performance	Industry's Highest System Performance
Logic Cells	20K – 355K	30K – 410K	285K – 2,000K
DSP Slices	40 – 700	120 – 1540	700 – 3,960
Max. Transceivers	4	16	80
Transceiver Performance	3.75Gbps	6.6Gbps 10.3Gbps	10.3Gbps 13.1Gbps 28Gbps
Memory Performance	800Mbps	2133Mbps	2133Mbps
Max. SelectIO™	450	500	1200
SelectIO™ Voltages	3.3V and below	3.3V and below 1.8V and below	3.3V and below 1.8V and below

XILINX

Bridging the DSP Performance Gap with 7-Series



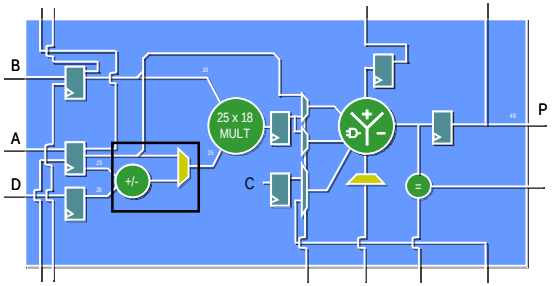
•* Peak performance for symmetric filters

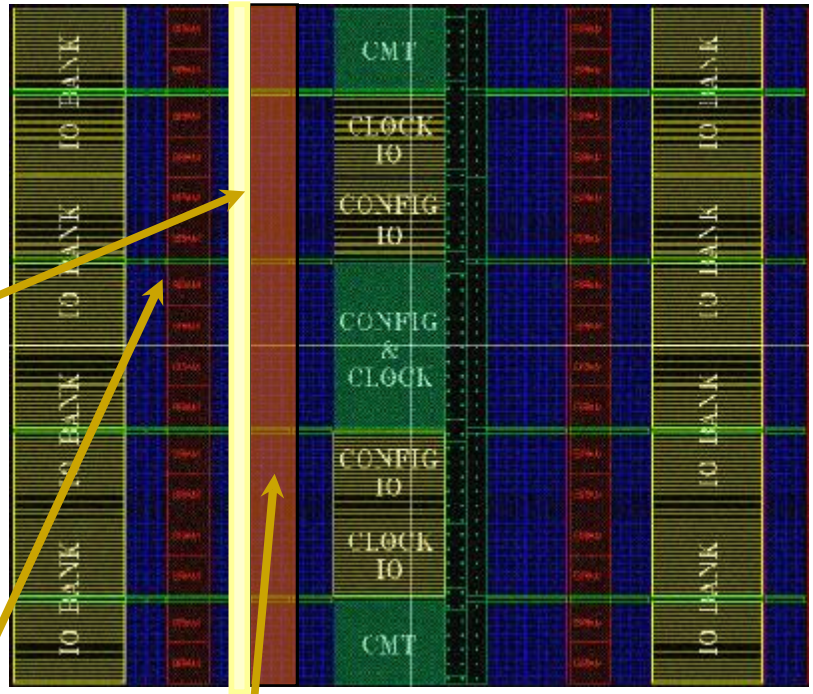
XILINX

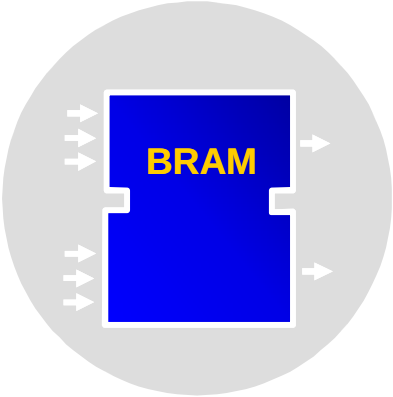
FPGA Resource

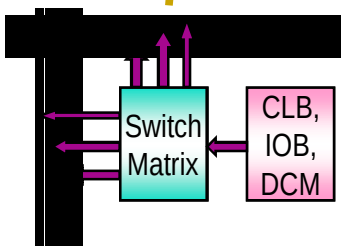
§ **Challenge: How do we make the best use of these resources in most efficient manner?**

DSP48





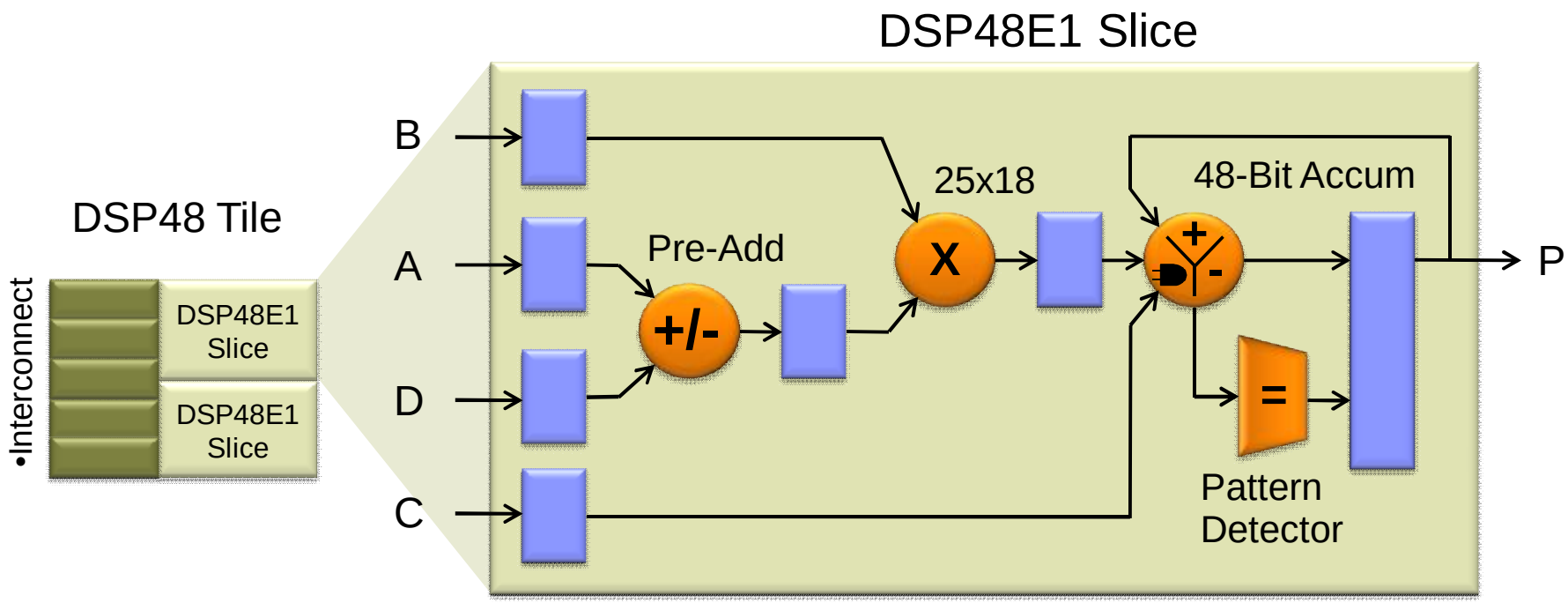




Logic Fabric

DSP Performance through the DSP48E1 Slice

Virtex-6, Artex-7, Kintex-7, Virtex-7

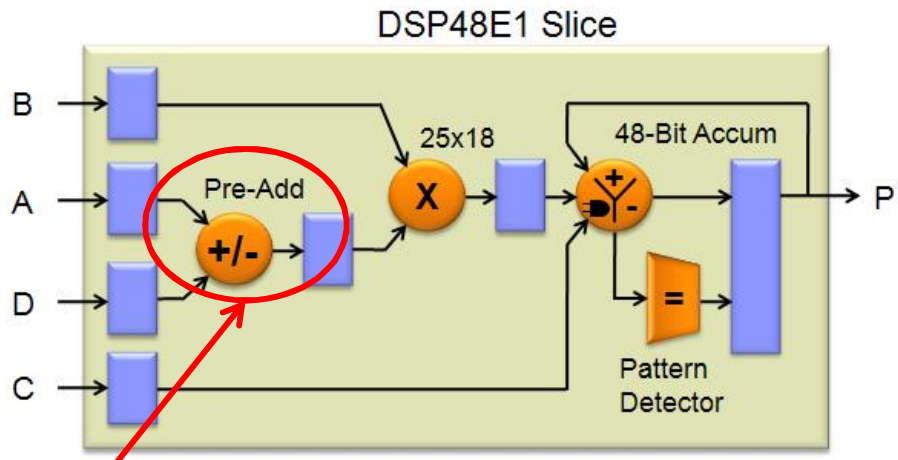


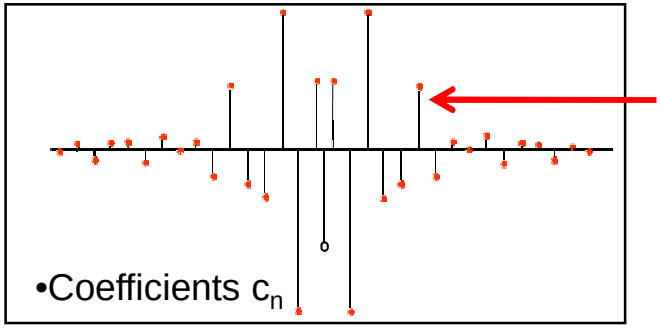
- § 2 DSP48E1 Slices / Tile
- § Column Structure to avoid routing delay
- § Pre-adder, 25x18 bit multiplier, accumulator
- § Pattern detect, logic operation, convergent/symmetric rounding
- § 638 MHz Fmax

XILINX

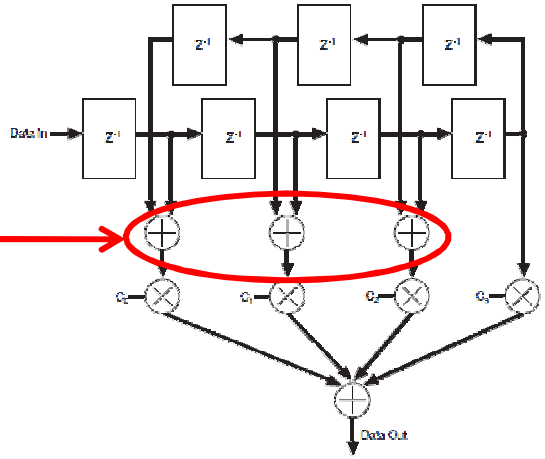
Pre-Adder

- § Hardened Pre-Adder leverages filter symmetry to reduce Logic, Power and Routing
- § No restriction to coefficient table size

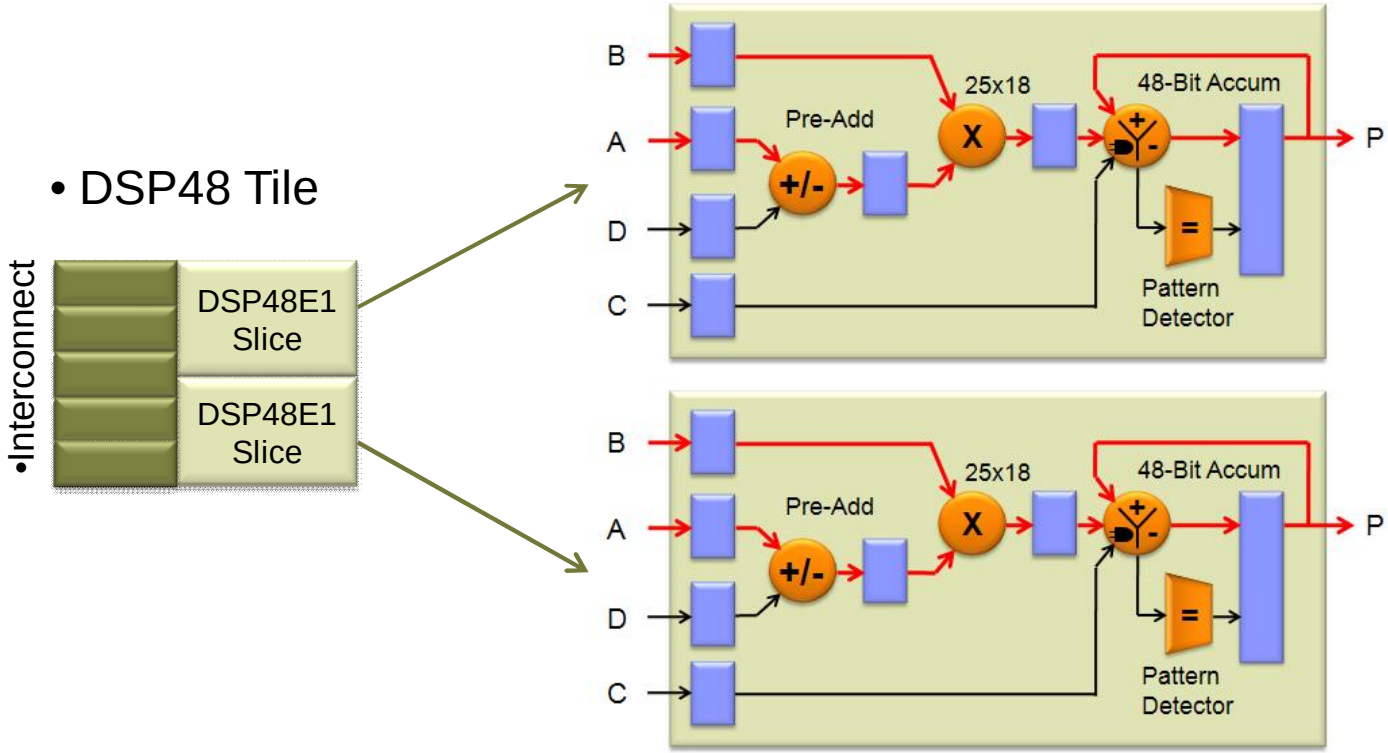




•Filter symmetry exploited to pre-add tap delay values and reduce multiplies by 50%



Greater Flexibility with Fully Independent Multipliers



§ Full, independent access to every multiplier

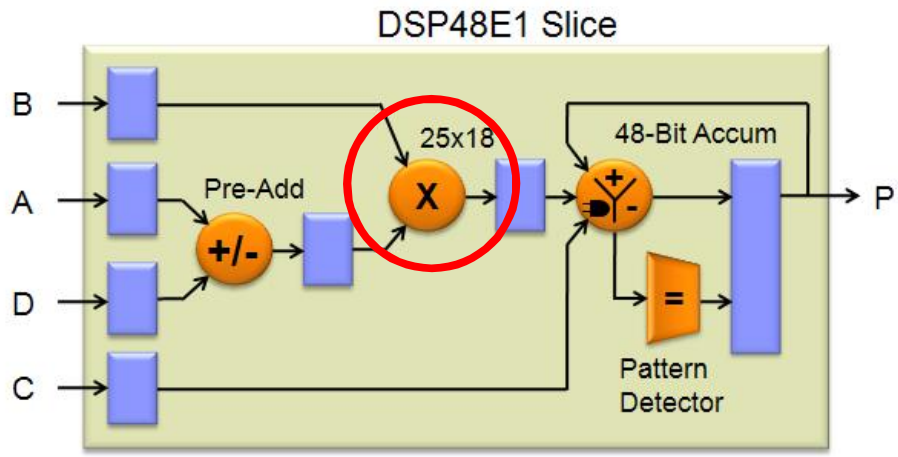
§ One accumulator for each multiplier

§ 5 Interconnects support up to 50 bit multiplies per tile

XILINX

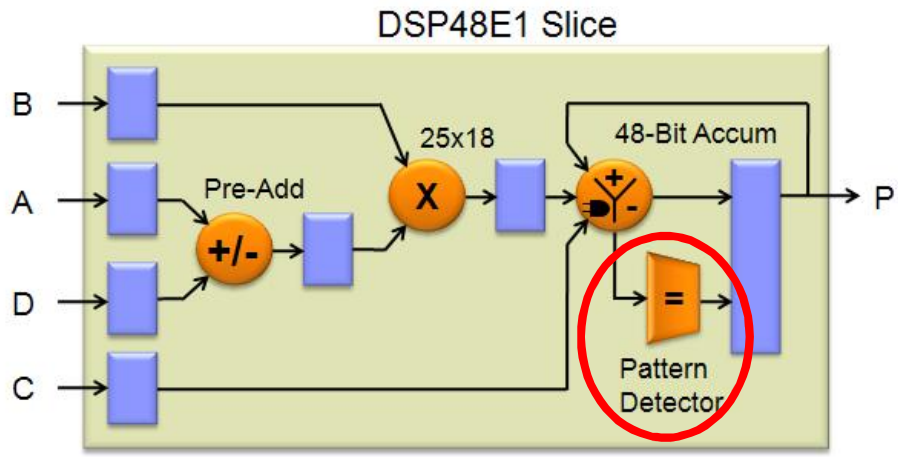
25x18 Multiplier

- **Single DSP slice supports up to 25x18 multiplies**
  - 50% fewer DSP resources required for high-precision multiplies
  - Efficient FFT Implementations
  - Efficient single-precision floating-point implementations
- **Single DSP Tile supports up to 50x36 multiplies**
- **Delivers higher performance and lower power**



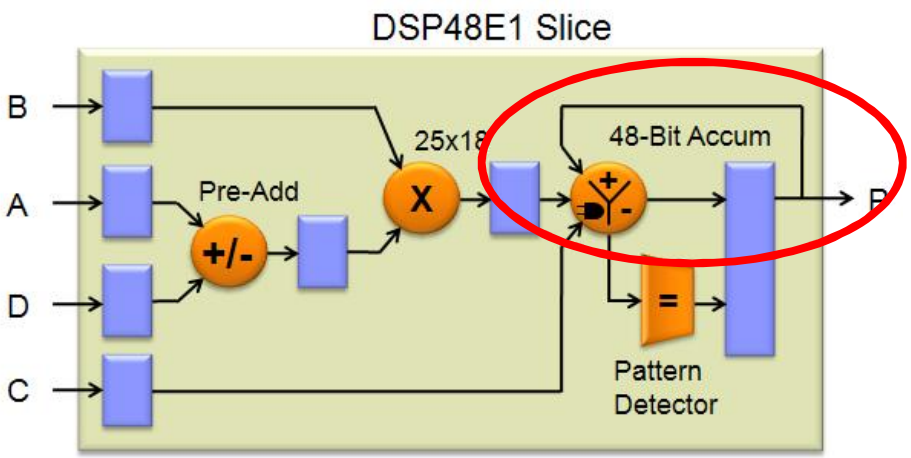
Efficient Rounding Modes using Pattern Matching

- **Only FPGA architecture that supports pattern detection**
  - Pattern can be constant (set by attribute) or C input
- **Efficient implementation of rounding modes**
  - Symmetric
  - Convergent
  - Saturation



One Accumulator for each Multiplier

- § **DSP48E1 slice provides an accumulator for each multiplier**
  - 2X more than competitive architectures
- § **Up to 48-bits accumulation per DSP slice**
  - 25x18 multiply
- § **Up to 96-bits accumulation per DSP tile**
  - 50x36 multiply

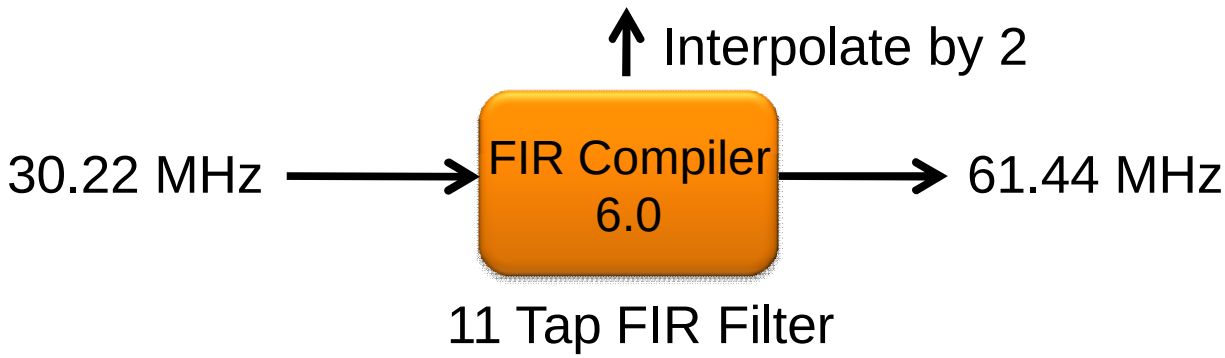


DSP IP Portfolio

- § Comprehensive IP portfolio
- § Constraint Driven
- § IP can be imported into RTL, System Generator and Platform Studio

Category	IP Blocks
Math	mult, adder, accumulator, divider, trig, CORDIC
Filters	FIR, CIC
Memory	RAM, register, FIFO, shift register
Transforms	FFT, IFFT, LTE FFT
Processors	MicroBlaze
Video	Color correction, CFA, pixel correction, image characterization, edge enhancement, noise reduction, statistics, CSC, VFBC, Scaler, timing controller,
Wireless	DDS, DUC/DDC, MIMO Decoder/encoder, RACH preamble det, DPD, CFR,
Floating-Point	Add/sub, mult, div, sqrt, compare, convert, FFT

Constraint Driven IP



Parameter	Result 1	Result 2	Result 3	Result 4
Channels	2	2	4	4
Clock Frequency	122.7	245.4	245.4	368.1
DSP Slice Count	3	1	3	1

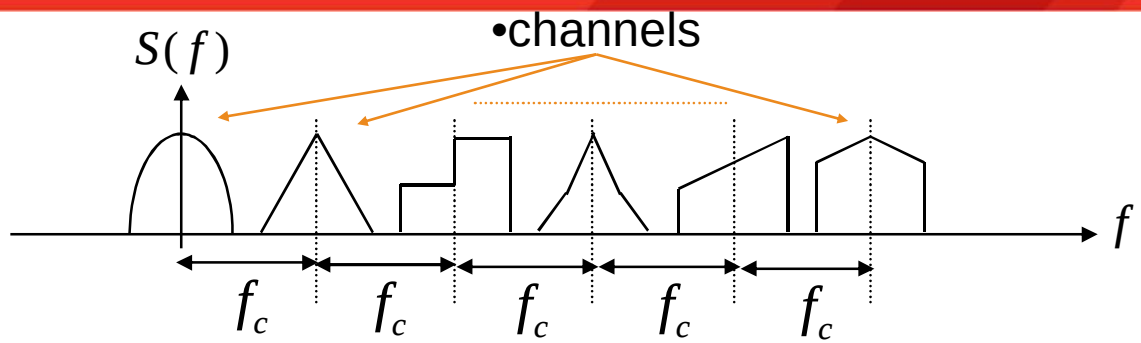
- **Overclocking automatically used to reduce DSP slice count**
- **Quick estimates provided by IP compiler GUI**
- **Insures best results for your design requirements**

XILINX

Interesting Algorithms For FPGA Implementation

- **Critically sampled channelizers**
  - Polyphase with a DFT bank
- **Divide and conquer DFT**
  - Calculating a 1D FFT as a 2DFFT
- **Winograd FFT Transform**
  - Least amount of multiplies

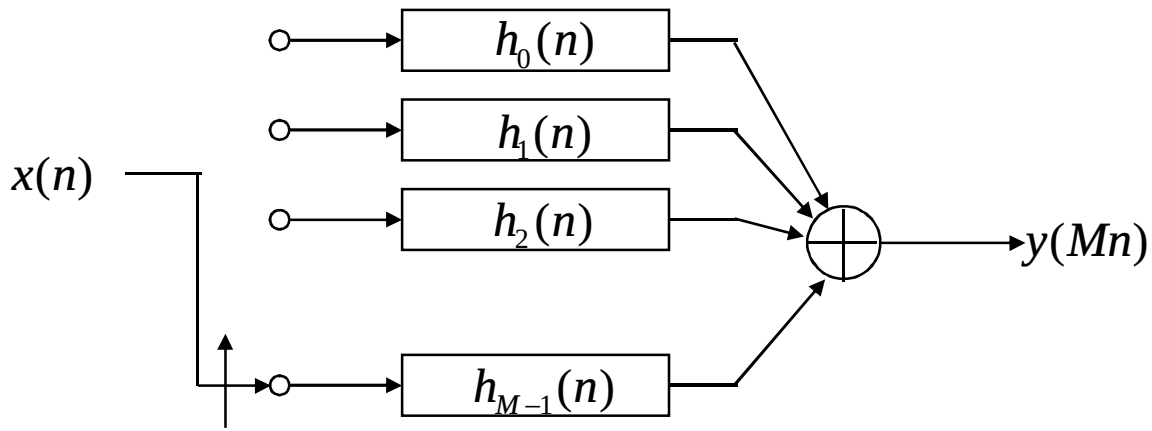
Passband Polyphase Filters



- § **In a FDM digital communication system a common requirement is, for each channel:**
  - translate the channel to baseband
  - shape the channel spectrum
  - reduce the sample rate to match the channel bandwidth
- § **This is the function of a *channelizer***
- § **When the channel spacing's are equal a computationally efficient structure for performing the above functions is the carrier centered polyphase transform**

XILINX

Baseband Polyphase Filter



$h_0(n) =$	h_0	h_M	L	h_{N-M}
$h_1(n) =$	h_1	h_{M+1}	L	h_{N-M+1}
M	M	M	L	M
$h_{M-1}(n) =$	h_{M-1}	h_{2M-1}	L	h_{N-1}

XILINX

Passband Polyphase Filters

Express the filter coefficient set in terms of a course and vernier index

r_1 and r_2 respectively

$$h(n) = h(r_1 + Mr_2) \qquad r_1 = 0, \mathbf{K}, M-1, \quad r_2 = 0, \mathbf{K}, \frac{N}{M} - 1$$

•Invoke the modulation theorem to convert a prototype baseband filter to its equivalent carrier centered, or spectrally shifted version

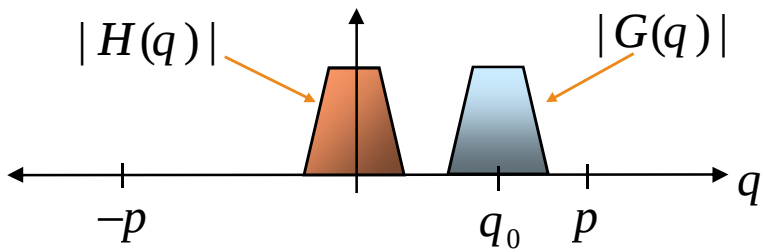
if $\quad h(n) \Leftrightarrow H(q)$

then $\quad h(n)e^{jq_0 n} \Leftrightarrow H(q - q_0)$

Passband Polyphase Filters

The coefficients of the carrier centered filter are

$$g(n) = h(n)e^{jq_0 n}$$



Now perform a polyphase partition on the modulated coefficients

$$g_{r_1}(r_2) = h(r_1 + Mr_2)e^{jq_0(r_1 + Mr_2)}$$

$$= h(r_1 + Mr_2)e^{jq_0 r_1}e^{jq_0 M r_2}$$

Select q_0 so that a single period of the series $e^{jq_0 n}$ is harmonically related to M

XILINX

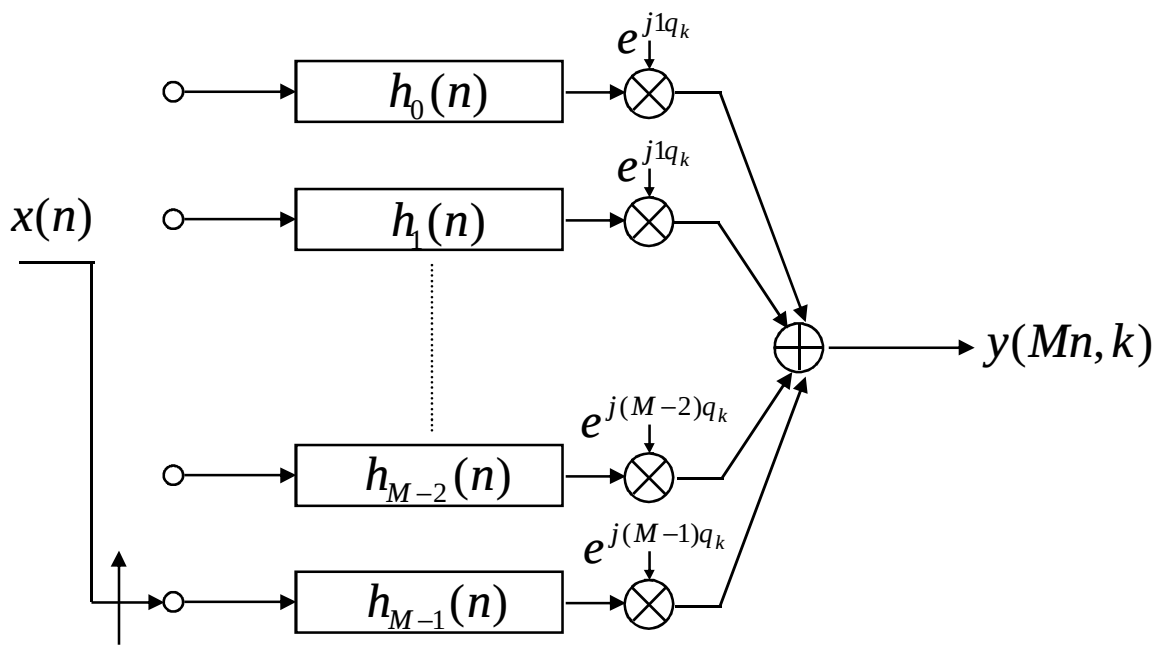
Passband Polyphase Filters

$$q_0 = k\frac{2p}{M}$$

$$g_{r_1}(r_2) = h(r_1 + Mr_2)e^{jq_0 r_1}e^{jk\frac{2p}{M}Mr_2}$$

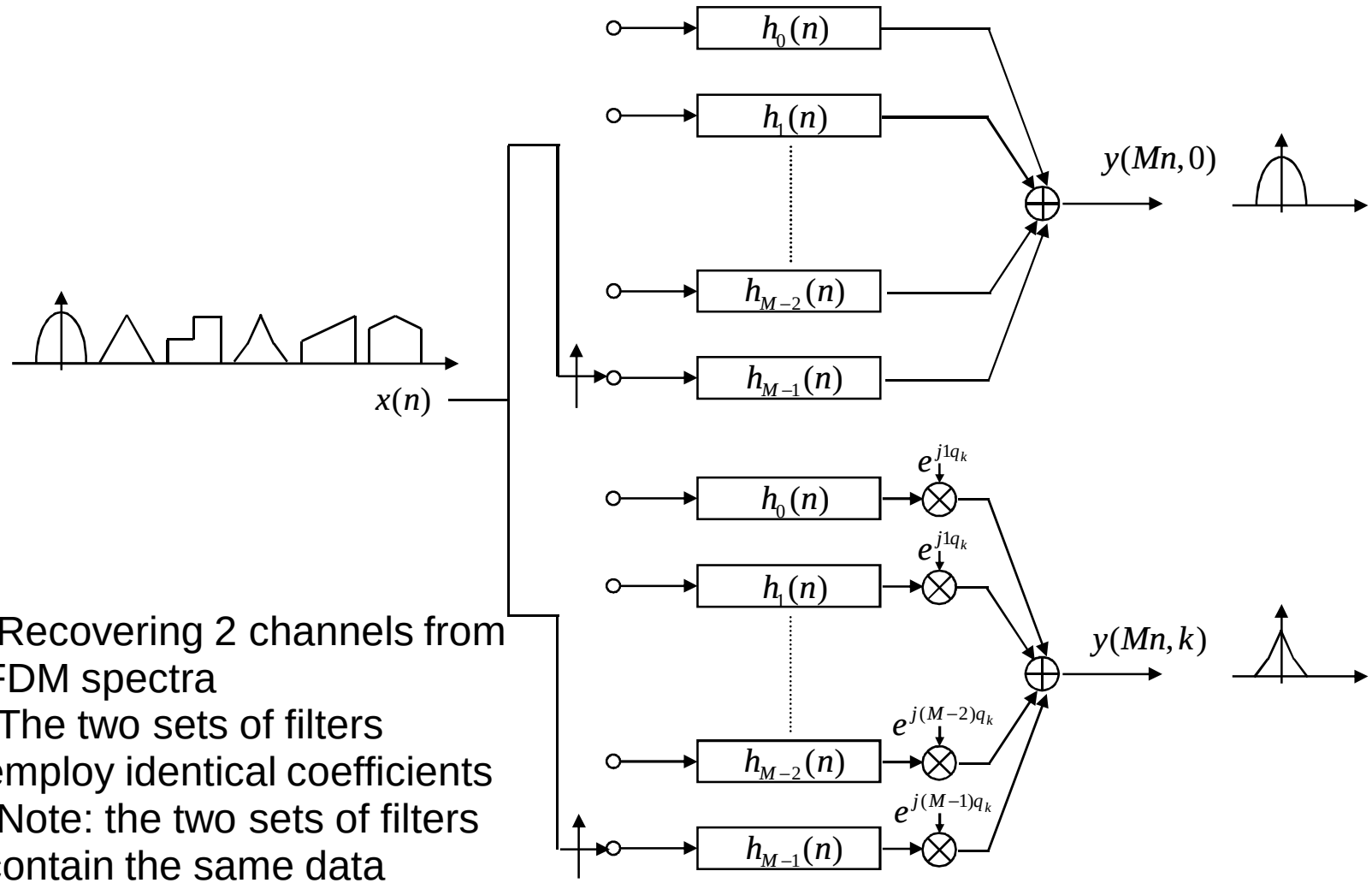
$$= h(r_1 + Mr_2)e^{jk\frac{2p}{M}r_1}$$

- Carrier centered polyphase filter
- the one structure
  - baseband's the channel
  - shapes the signal
  - reduces the sample rate



XILINX

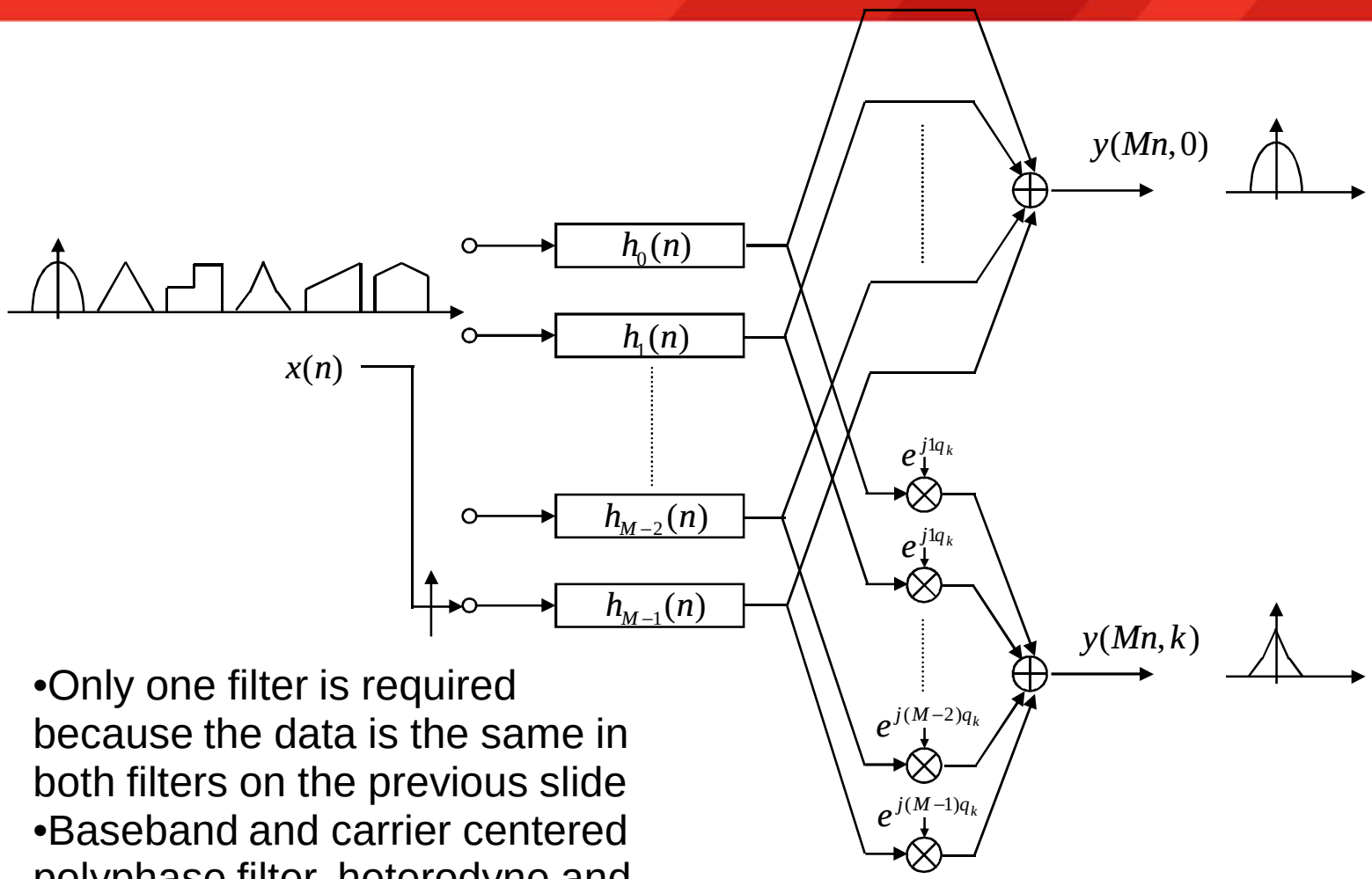
Passband Polyphase Filters



- Recovering 2 channels from FDM spectra
- The two sets of filters employ identical coefficients
- Note: the two sets of filters contain the same data

XILINX

Passband Polyphase Filters



- Only one filter is required because the data is the same in both filters on the previous slide
- Baseband and carrier centered polyphase filter, heterodyne and downsample

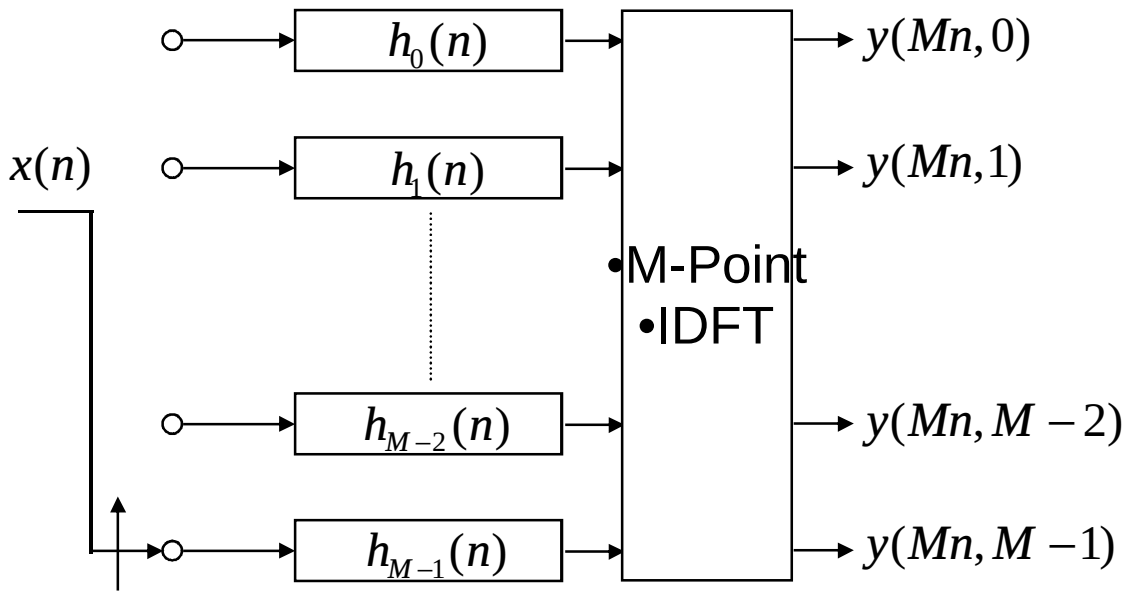
XILINX

Polyphase Transform

Recall that the IDFT of an M-point sequence $Y(k)$ is

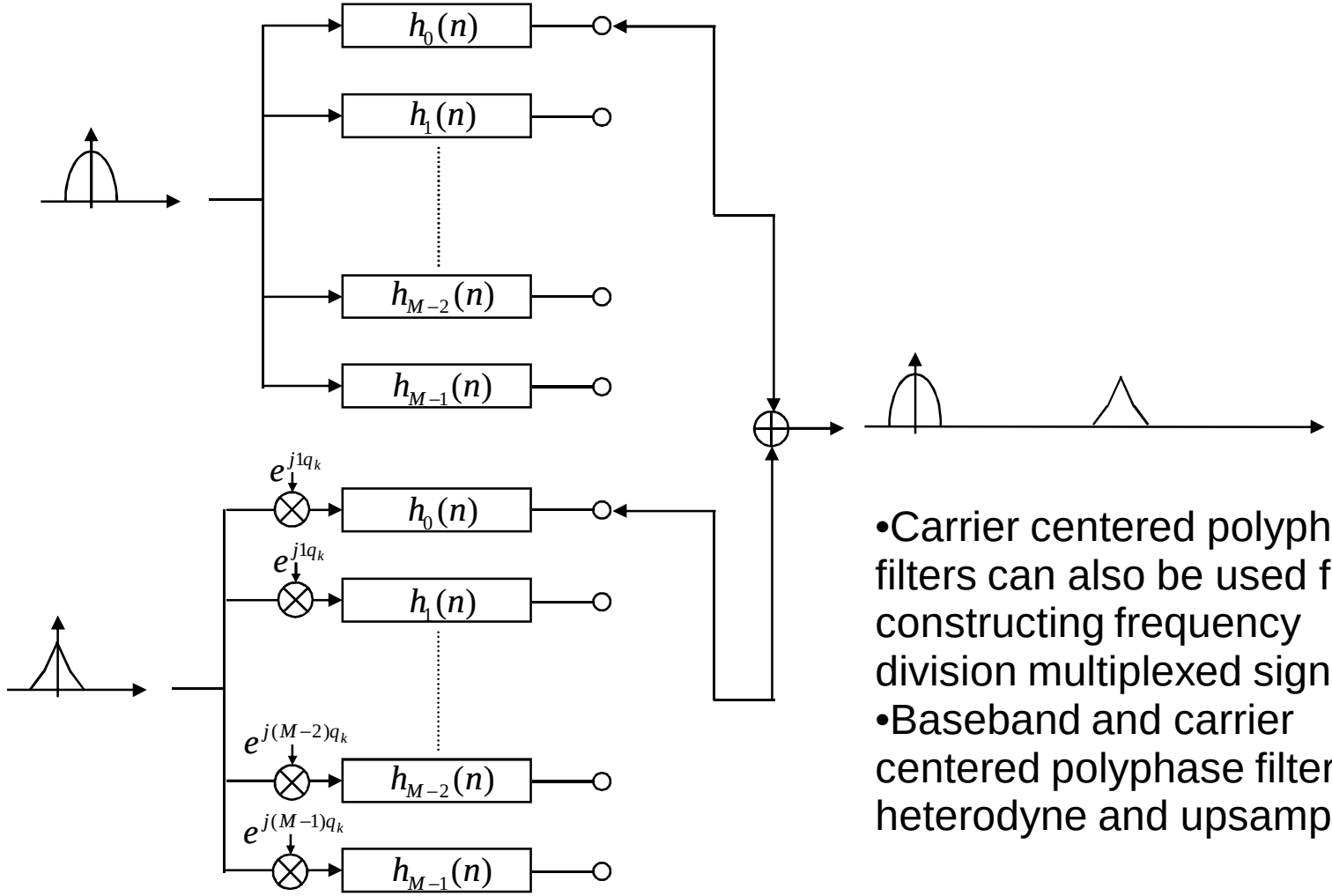
$$y(n) = \sum_{k=0}^{M-1} Y(k) e^{j2p\,nk/M} \qquad n = 0,1,\mathbf{K},M-1$$

If the M phase rotators are sequenced over all of the M values of k we recognize that this is the same as computing an IDFT



XILINX

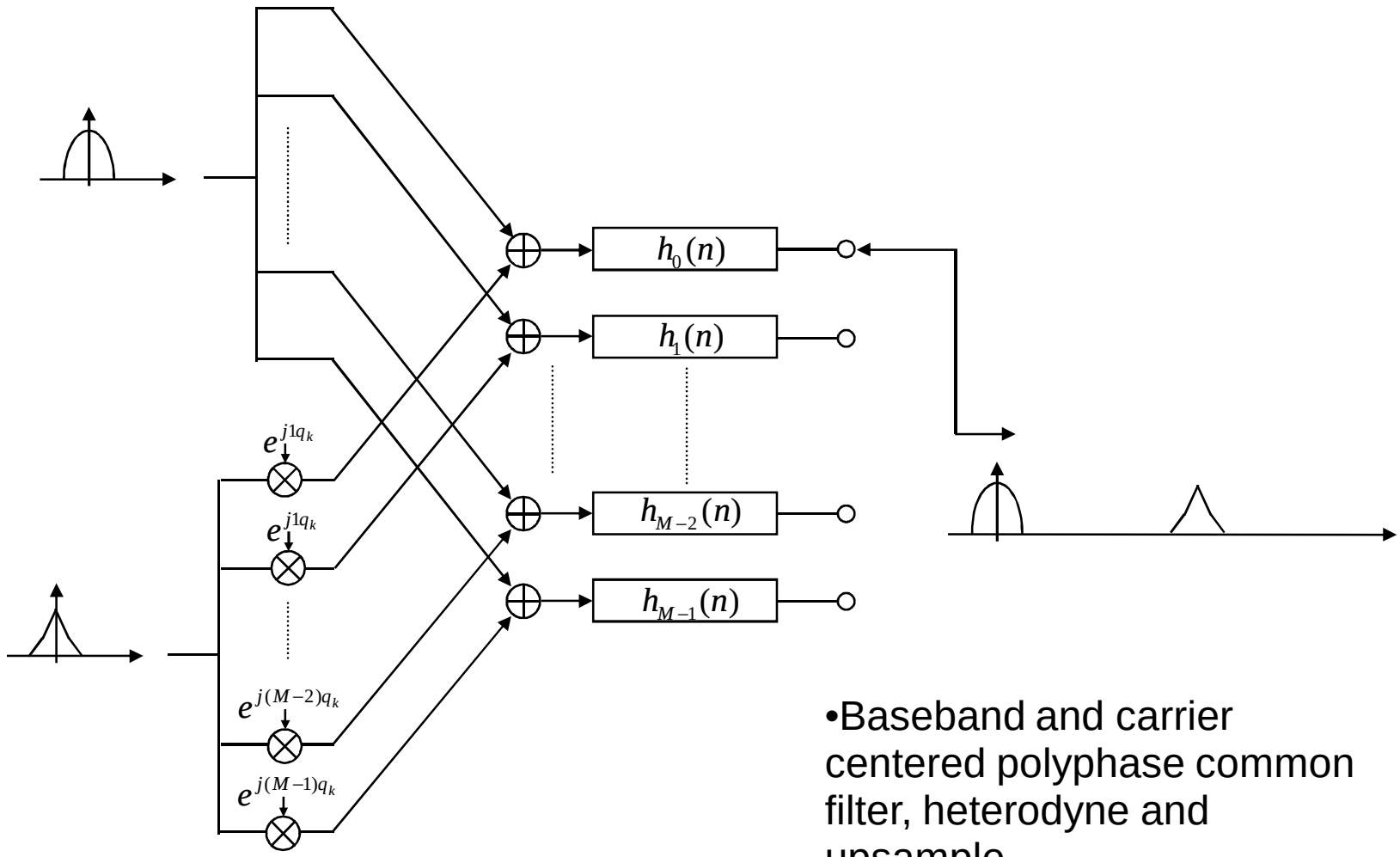
•Passband Polyphase Filters



- Carrier centered polyphase filters can also be used for constructing frequency division multiplexed signals
- Baseband and carrier centered polyphase filter, heterodyne and upsample

XILINX

•Passband Polyphase Filters



•Baseband and carrier centered polyphase common filter, heterodyne and upsample

XILINX

Divide and Conquer FFT

§ **It is possible to compute a one dimensional DFT as a two dimensional DFT**

– Ideal for processing hi rate data that has been demuxed to multiple paths at a lower rate

Decompose DFT into two dimensions:

$$X(p,q) = \sum_{m=0}^{M-1} \sum_{l=0}^{L-1} x(l,m) W_N^{(Mp+q)(mL+l)}$$

But:

$$W_N^{(Mp+q)(mL+l)} = W_N^{MLmp} W_N^{MLq} W_N^{Mpl} W_N^{lq}$$

However:

$$W_N^{Nmp} = 1, W_N^{mqL} = W_{N/L}^{mq} = W_M^{mq} \text{ and } W_N^{Mpl} = W_{N/M}^{pl} = W_L^{pl}$$

$$X(p,q) = \sum_{l=0}^{L-1} \left\{ W_N^{lq} \left[\sum_{m=0}^{M-1} x(l,m) W_M^{mq} \right] \right\} W_L^{lp}$$

XILINX

Divide and Conquer FFT

These simplifications lead to:

$$X(p,q) = \sum_{l=0}^{L-1} \left\{ W_N^{lq} \left[\sum_{m=0}^{M-1} x(l,m) W_M^{mq} \right] \right\} W_L^{lp}$$

Process Steps:

1. Store signal column-wise
2. Compute the M point DFT for each row
3. Multiply the resulting array by the phase factors W_N^{lq}
4. Compute the L-point DFT of each column
5. Read the resulting array row wise

Winograd FFT

Developed by mathematician Schmuel Winograd in 1976

- Goal was to reduce the number of multiplies required
- Multiplies minimized but at expense of increased complexity
- Memory mappings became very complex too
- Due to complexity, cost of doing an fft did not significantly go down
- Problem with algorithm is that multiplies and accumulates were separated so execution on DSP processor was not efficient