Efficient Split Radix FFTs in FPGAs

Tom Dillon
Dillon Engineering, Inc.

This presentation outlines methods for split radix FFTs implemented in FPGAs. Analysis of various algorithms with regards to performance, cost and power consumption are presented.

FPGAs are rapidly finding their way into high performance DSP applications, specifically real time signal processing applications. Large FPGAs offer a significant cost, size, and power advantage over other alternatives for many front end real time processing operations. FPGAs offer the advantage of short and flexible design cycles, high performance and relatively low NRE.

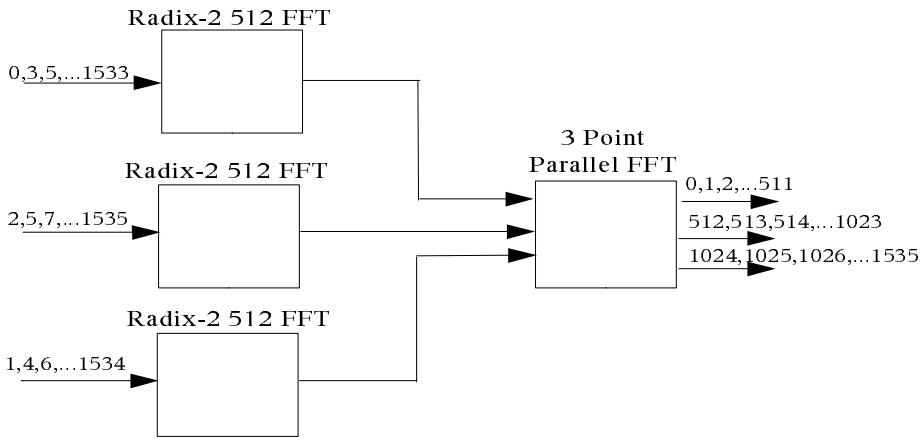
The FFT is at the heart of many real time signal processing applications, including radar, communication, and image processing. Logic for high speed FFTs can account for up to 90% of the cost and power of a given application, making efficient resource usage critical.

Architectures for radix-2 FFTs are well known and have been in use in FPGAs for some time with excellent results. Many applications require bin spacing that can't be achieved with radix-2 FFTs since one is limited to a power of 2 length. In order to attain more useful bin spacing, many times a split radix FFT architecture is used. For example, using a radix-2 with radix-3 architecture can produce 384 (128 x 3), 768 (256 x 3), 1536 (512 x 3) and so on. This architecture is not limited to radix-2 and radix-3, as with proper data routing structures any split radix combination can be implemented.

Two common methods for implementing split radix FFT architectures are Kolba-Parks and Cooley-Tukey. The advantages of both architectures pertaining to FPGA implementations will be detailed.

The basic structure of a 1536 point FFT architecture is given in the following diagram.

Figure 1: High Performance Parallel Radix-2 Times 3 Pt. Architectu
1536 Implementation



The size, cost, and power consumption of the FPGA are directly proportional to the throughput

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
20 AUG 2004	**N/A**	**-**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Efficient Split Radix FFTs in FPGAs	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Dillon Engineering, Inc.**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release, distribution unlimited

13. SUPPLEMENTARY NOTES
See also ADM001694, HPEC-6-Vol 1 ESC-TR-2003-081; High Performance Embedded Computing (HPEC) Workshop (7th)., The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**UU**	**12**	

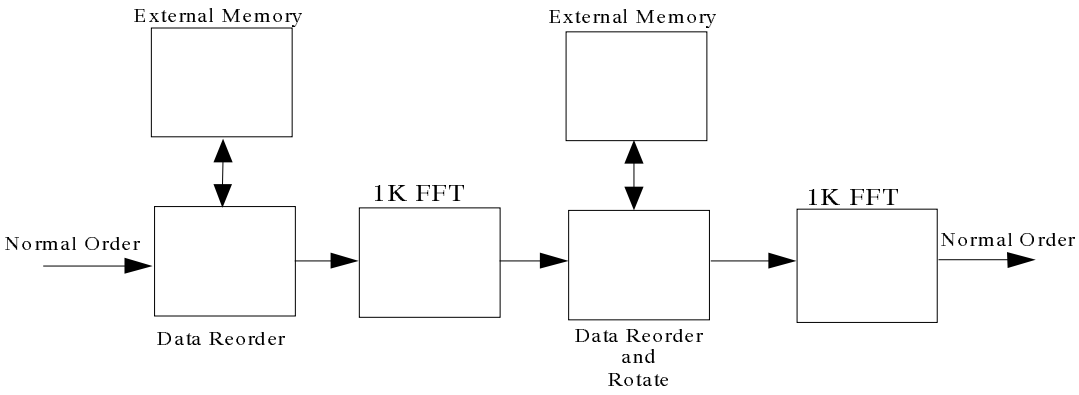
Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

of this FFT engine since logic is added as required to keep up with processing requirements. Some real examples of FPGA device usage, power consumption and cost will be presented. The architecture as shown accepts 3 inputs and produces 3 outputs per clock cycle, but can easily be extended for higher or lower performance applications. Clock rates in excess of 200MHz can be used with today's FPGA technology.

Another simple extension to this architecture is for ultra long FFTs. A similar architecture has been used to produce 512K and even 1M point FFTs in FPGAs with the simple addition of external memory to store intermediate results. Longer lengths only require more external memory.

The basic structure is shown below and will be presented with device usage, power and cost examples.

Figure 2: Ultra Long FFT Architecture
1M Continuous Data



A strong case is made for using FPGA technology for FFT processing in real time DSP applications.

Dillon ENGINEERING

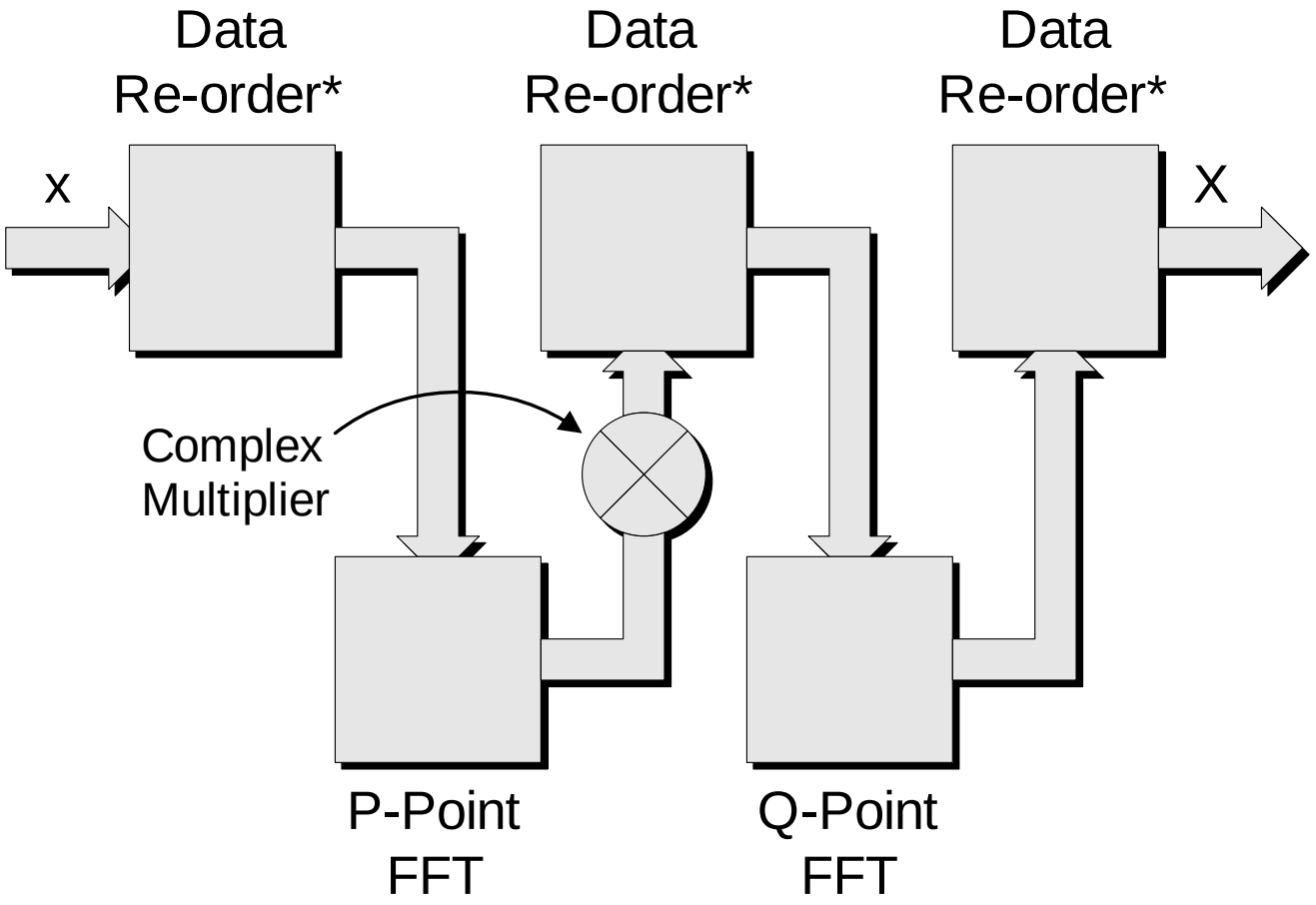
Implementing Efficient Split-Radix FFTs in FPGAs

- Radix-2 and Radix-4 FFTs are common
- Many applications benefit from other lengths
- OFDM Transceiver, Digital Video Broadcasts, and software defined radios often require FFTs that aren't a radix-2 or radix-4 length
- Split-radix simplifies the logic for these applications
- Common Split-radix algorithms are Cooley-Tukey, Kolbe-Parks, and Good-Thomas

Split-Radix FFTs

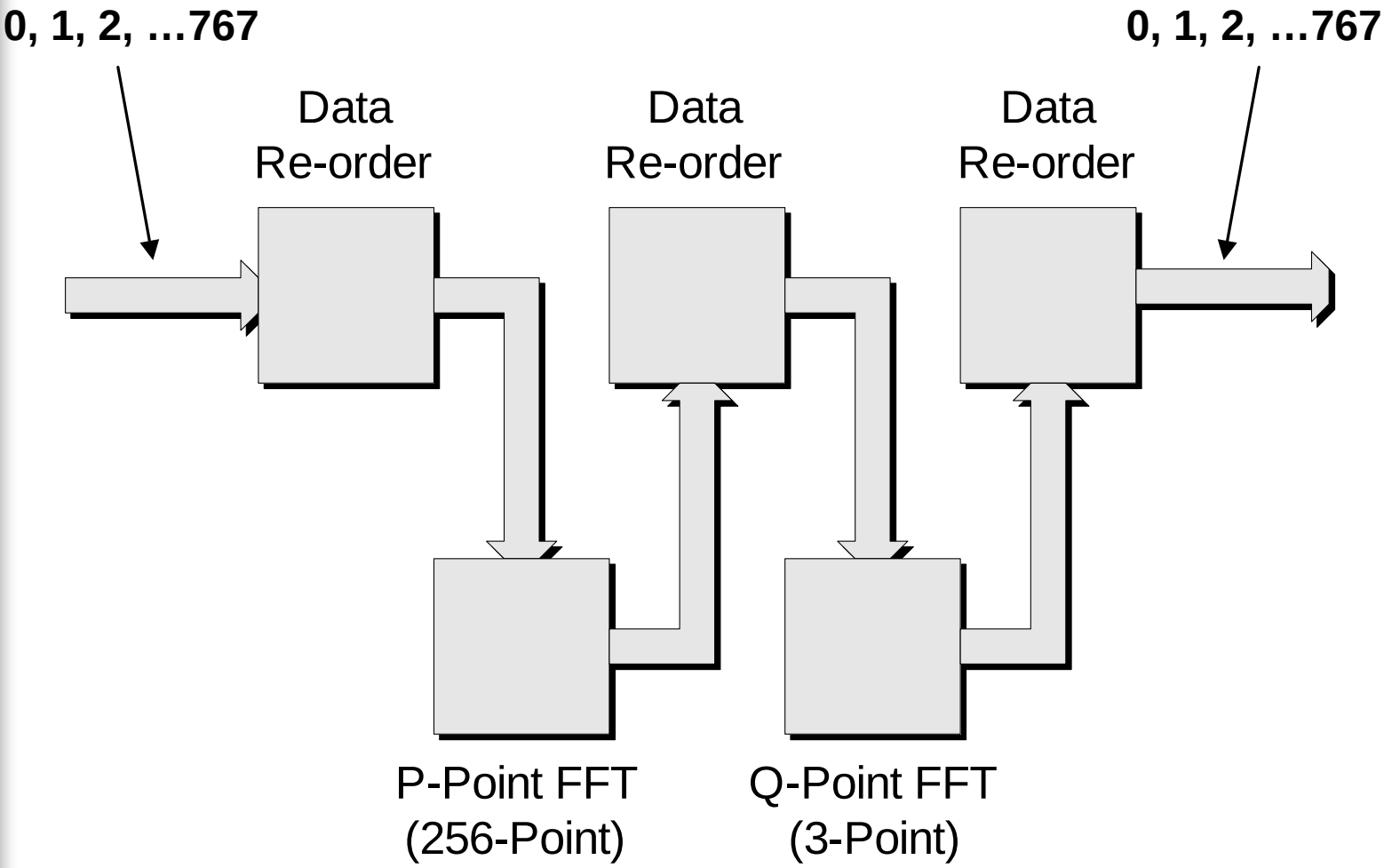
- Split-radix refers to combinations of two (or more) FFT engines
- Split-radix FFTs have a similar structure to 2D FFTs
- Split-radix FFTs provide bin spacing that produce better results for many applications
- Two split-radix approaches employed by DE:
  - Serial (traditional) – lower performance, higher memory requirements by using serial versions of both FFTs
  - Parallel – higher performance by placing larger radix FFTs in parallel and using a parallel version of the smaller radix FFT
- Parallel version mainly used when combining a larger FFT with a 3 or 5 point FFT, since it is feasible to use 3 or 5 large FFTs in a single device

Traditional Serial Split-Radix Approach

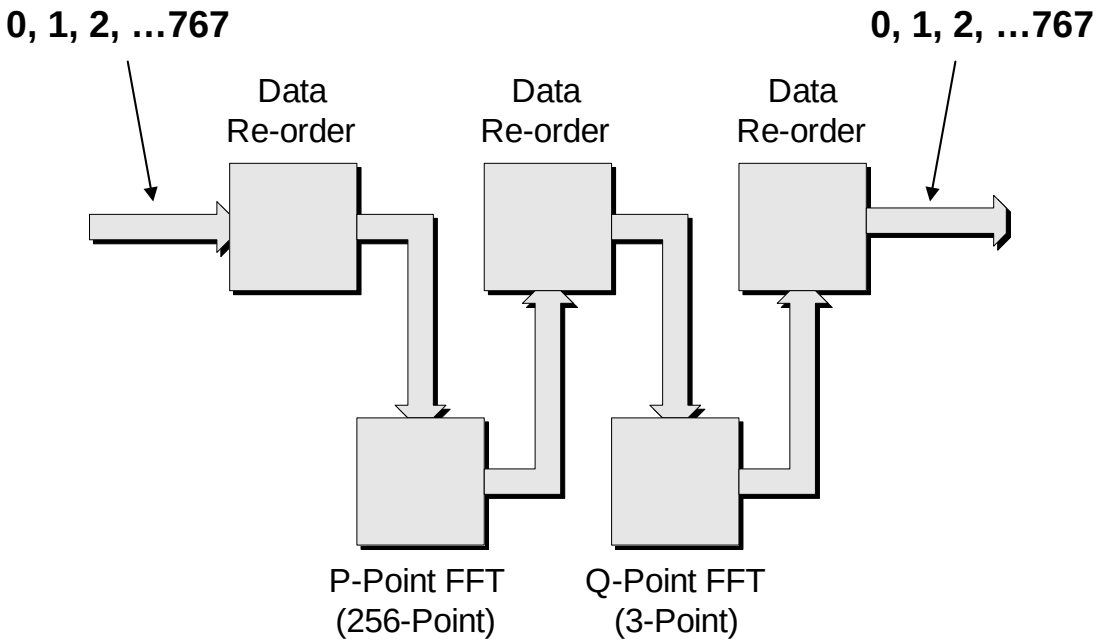


***Continuous data FFTs require enough memory to store two full copies of the data for each re-order stage**

Serial 768-Point Split-Radix FFT

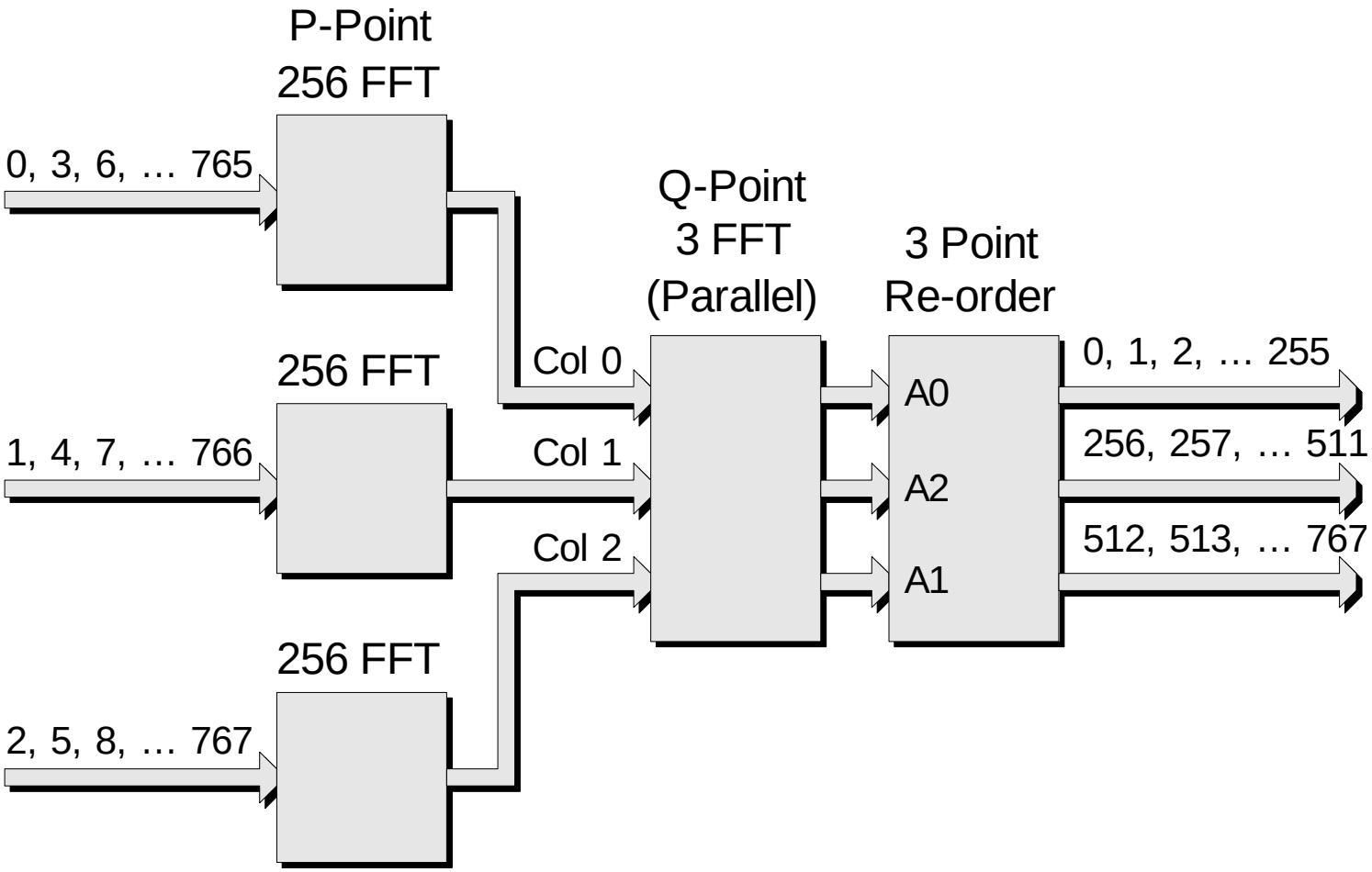


Serial 768-Point Split-Radix FFT (cont.)

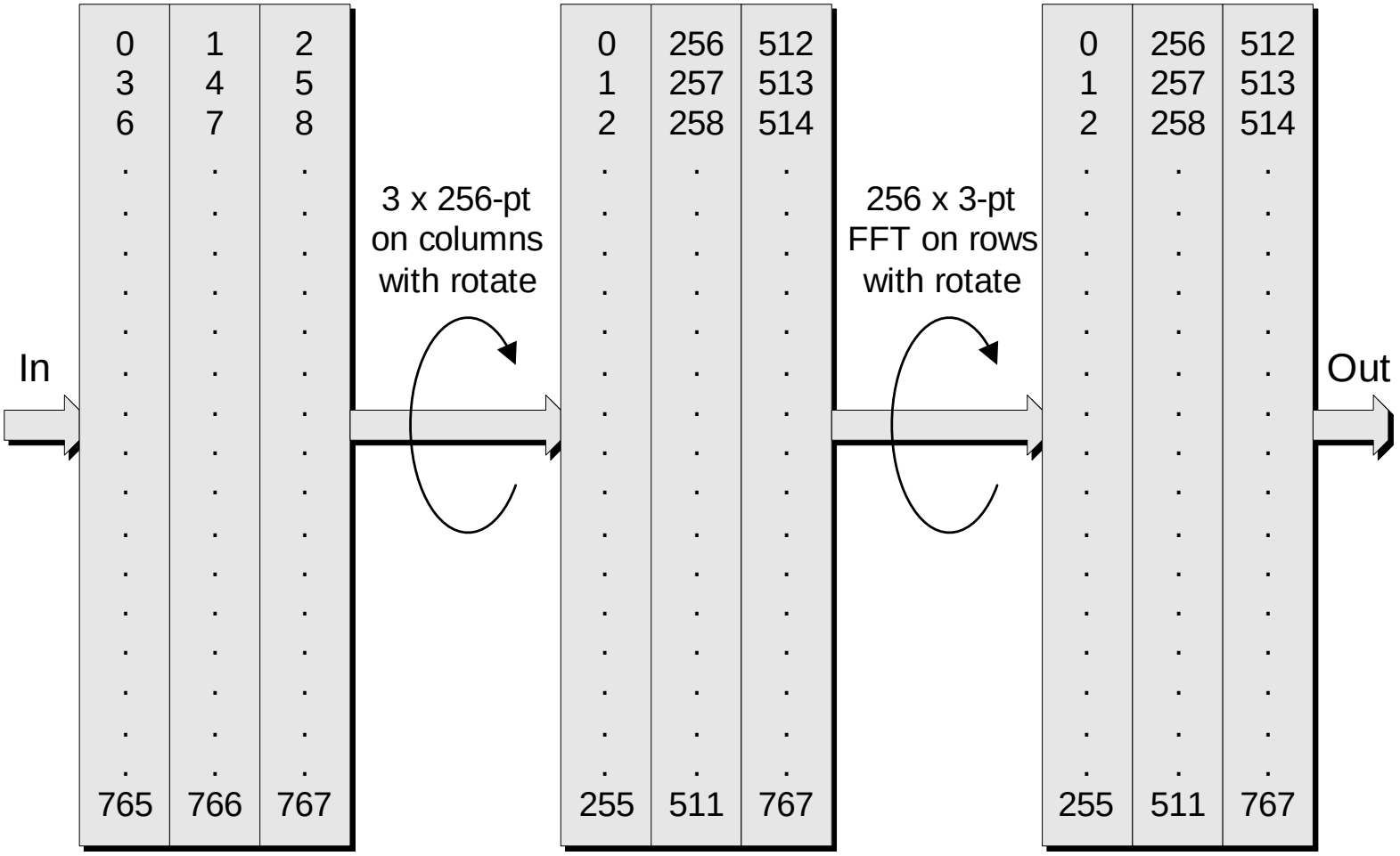


- Single engine of each radix (256-point FFT followed by 3-point FFT)
- Lower device utilization, with performance suitable for most applications
- High memory requirements for data re-ordering
- Speeds up to continuous data, slower data rates require less logic
- Same structure (with external memory) used for ultra-long FFTs

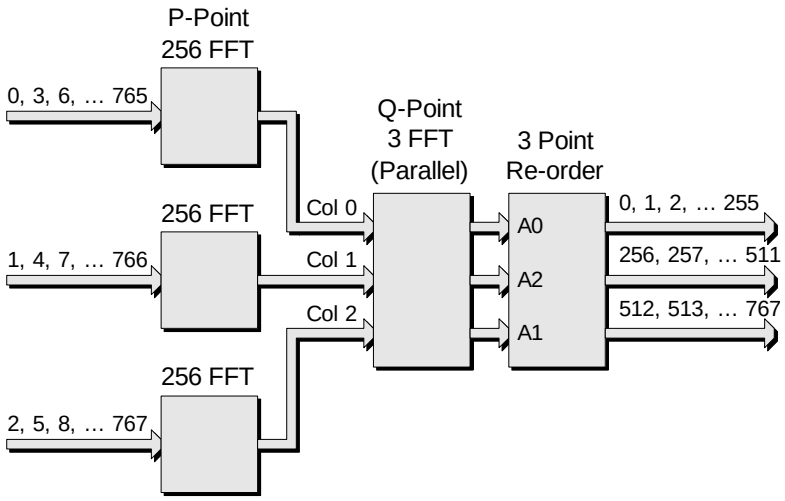
Parallel 768-Point Split-Radix FFT



Parallel 768-Point Split-Radix FFT Data Flow



Parallel 768-Point Split-Radix FFT (cont.)



- Combines 256-point FFT with 3-point FFT
  - 3 x 256-point FFT executions
  - 256 x 3-point FFT executions
- Eliminates the need for intermediate memory
- Higher resource (logic) usage as more computations are performed in parallel
- Very high performance – perform a new 768-point FFT every 256 clock cycles (1.7uS @ 150 MHz)

Virtex II Performance

Type	Virtex II Performance @ 150 MHz (18-bit Complex I/O)							
	Number of Butterflies	Latency (uS)	FFT Rate (uS)	Sizes	Block RAM	Multipliers	Power (mW)	Cost ($)
Serial	1	20.67	20.42	3,562	12	4	756	290
Serial	4	15.67	5.12	5,224	18	16	969	500
Parallel	3	6.96	6.83	4,180	24	12	974	350
Parallel	6	3.54	3.41	6,260	45	24	1,331	700
Parallel	12	1.84	1.70	11,123	75	48	1,840	1,400

- **Latency: Time from last point in to first out**
- **FFT rate: Rate to input FFT data sets**
- **Power: Estimate via Xilinx XPower**
- **Cost: Based on single piece XC2V3000-6 from Partminer.com**

Other Dillon Engineering Resources

- ParaCore Architect (parameterized core builder)
- DSP Algorithms
  - Ultra-long FFTs (2k x 2k = 4M points)
  - 2D FFTs for image processing
  - Fixed or floating-point FFTs
  - Floating point math library
- System level DSP
  - OFDM Transceivers
  - Radar Processing on single FPGA
  - Image Compression/Processing
- FPGA-based DSP development platforms
- Hareware/Software SOC
  - High speed Ethernet Appliances
  - Linux Based SOC in FPGA
  - MicroBlaze application