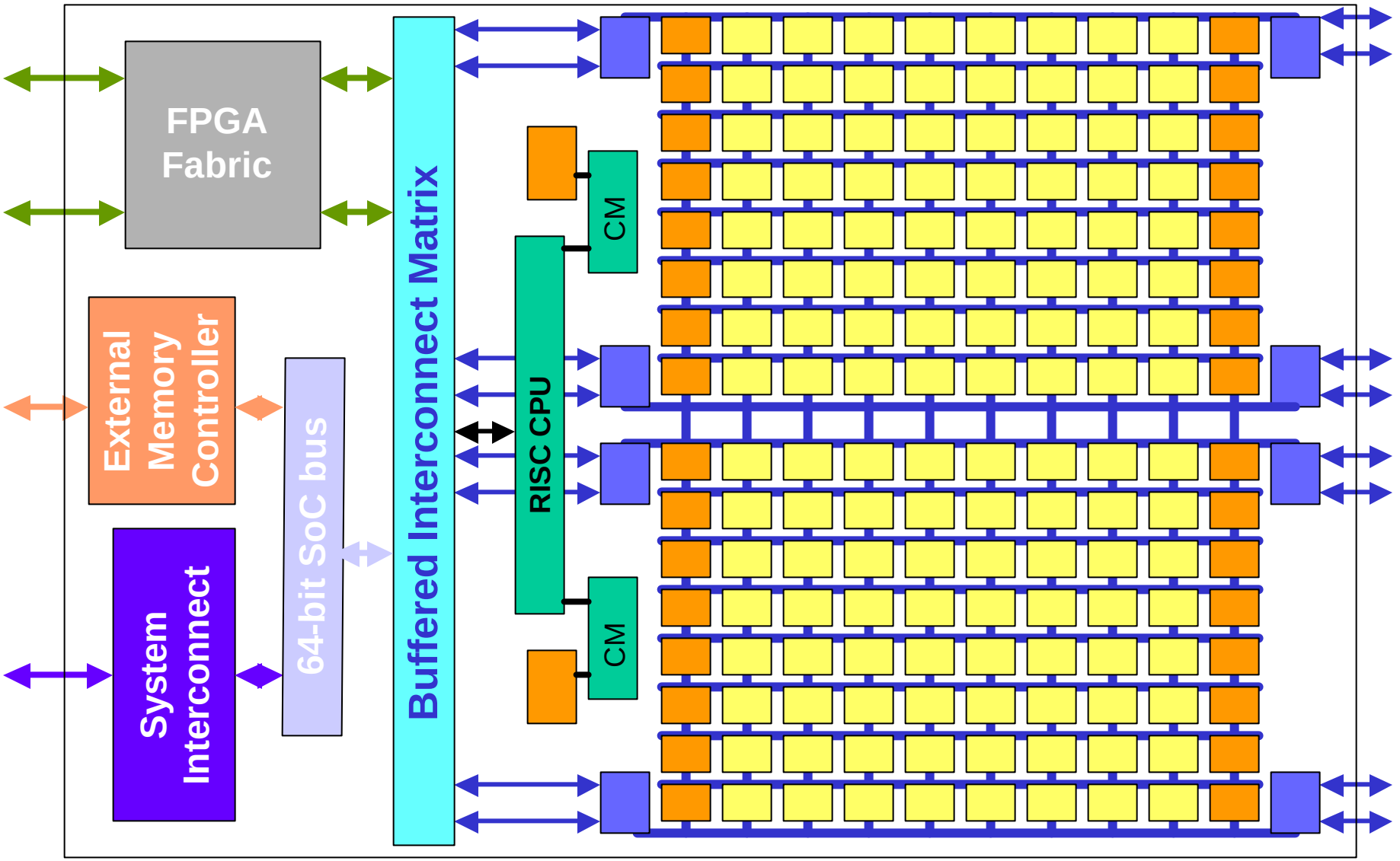
The eXtreme Adaptive DSP Solution to Sensor Data Processing

***Martin Vorbach**, PACT XPP*

***Leo Mirkin,** SKY Computers, Inc.*

Example of eXtreme DSP Architecture - QuickMIPS-XPP



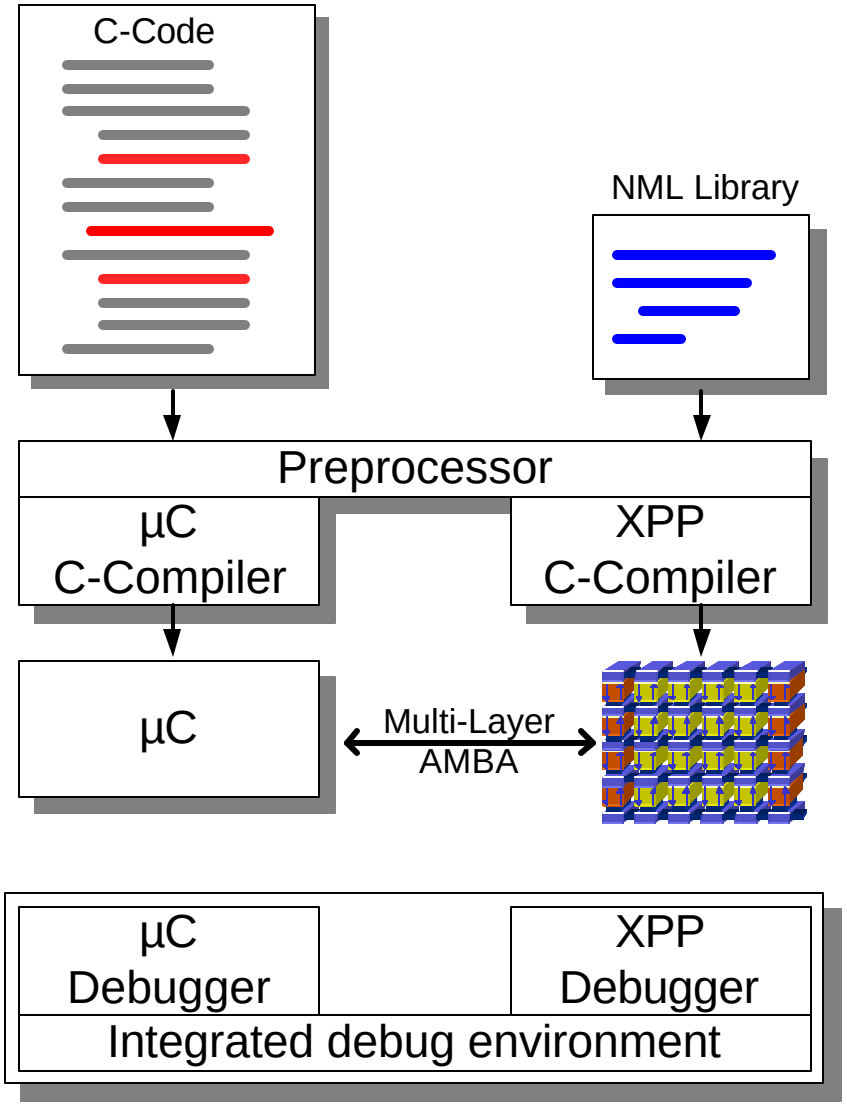
Direct Compilation from C Language onto Computing Grid

- eXtreme DSP processor combines RISC CPU with a scalable DSP coprocessor made from a regular grid of Processing Array Elements (PAE) and supported by FPGA-controlled I/O interface
- PAE grid is made from 16-32 bit fixed or floating point MAC-ALU with streaming I/O cells delivering external data and RAM cells holding constants and intermediate results
- Grid regularity and PAE control simplicity allows DSP programming by unrolling legacy C-code into the space of computing grid
- All DSP processing carried by scalable XPP co-processor with non-critical scalar and glue code running in RISC CPU

Key Elements of the eXtreme DSP Architecture

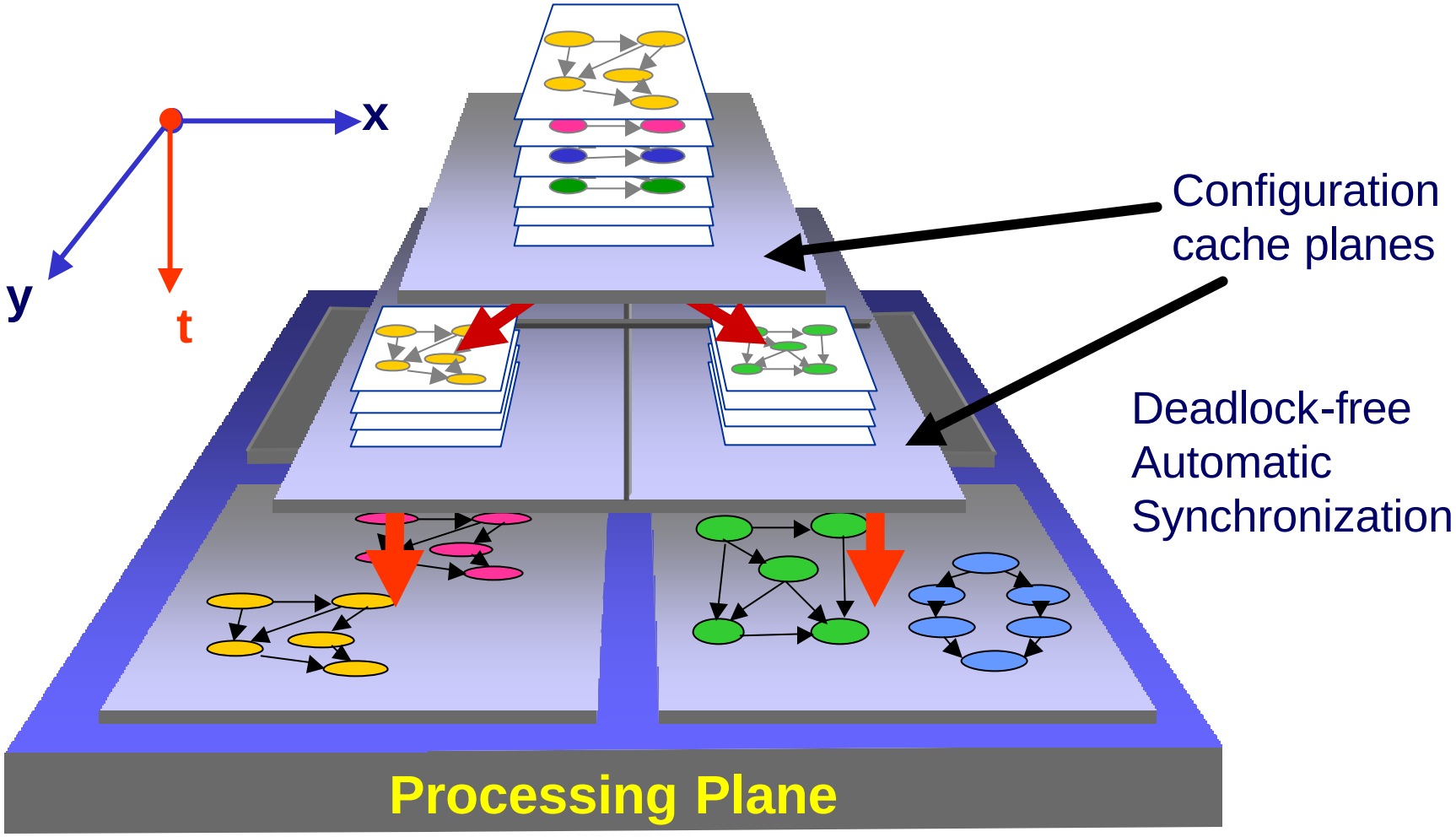
- PAE grid is tied together by a data packet carrying high-bandwidth interconnect
- Configuration network links all PAEs and operation of each PAE can be independently re-programmed in one cycle
- Automatic data flow synchronization makes PAE operations deadlock-free
- PAE operations are data driven and *don't consume power in the absence of input data*

Integrated C-based DSP Programming for QuickMIPS-XPP

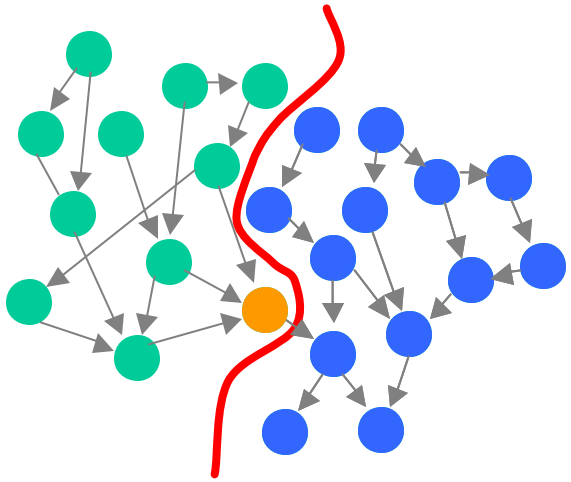


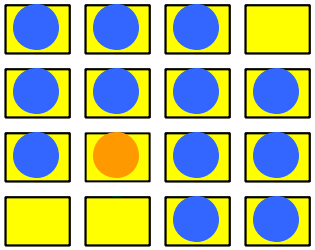
- eXtreme DSP tools integrated in host processor C-toolchain
- One source code for XPP and µC
  - Code exchanged by
    - -Source code annotation
    - -Library subroutine calls
- Automatic insertion of interface routines for µC and XPP intercommunication
- Fully integrated debug environment

DSP Fabric Configuration and Deadlock-Free Synchronization



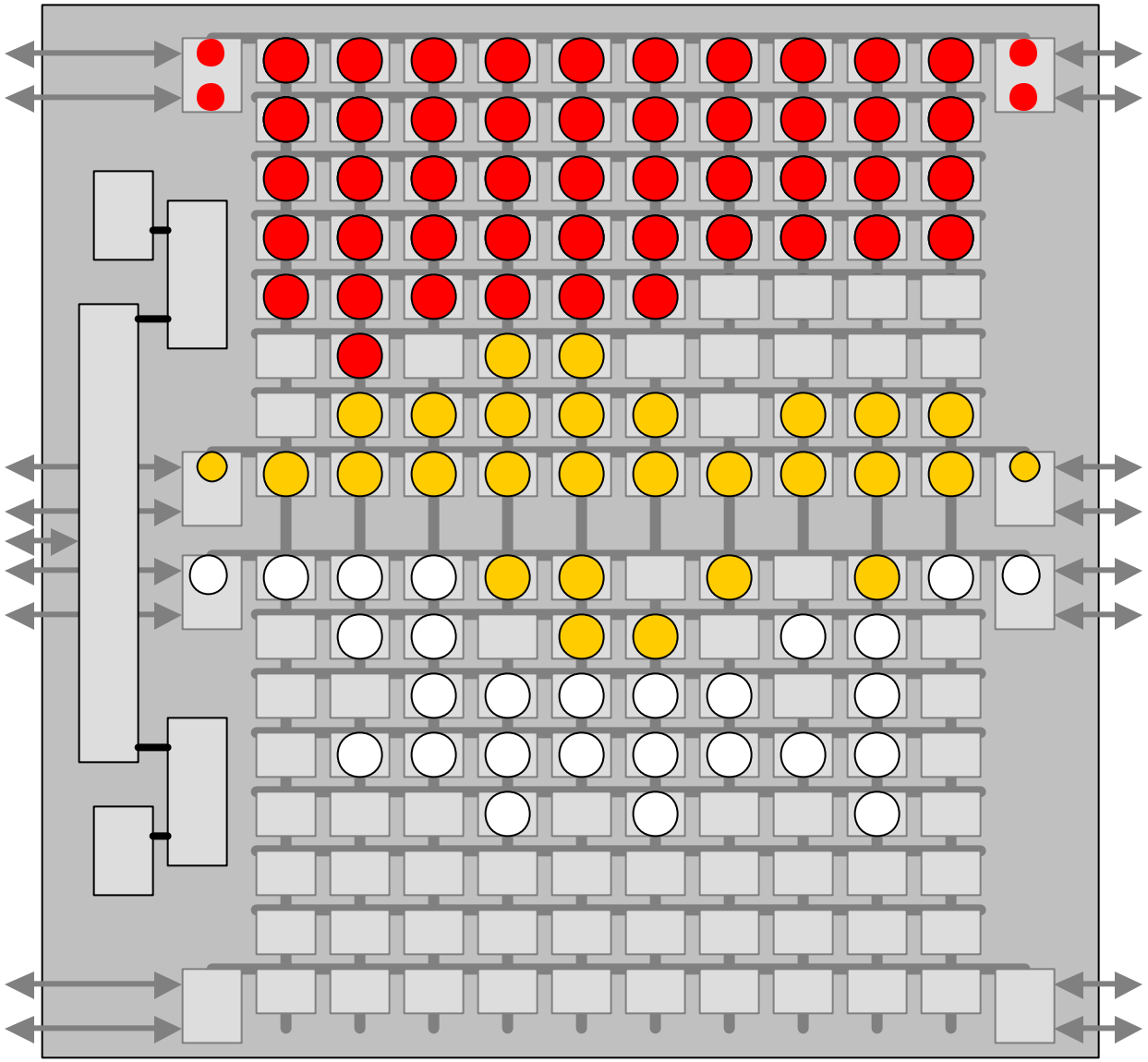
Fitting Large Algorithm to XPP Grid Using Instant PAE Reprogramming





- If large flow graph does not fit into PAE grid
- First, locate a good separation point, partition graph into parts **1** and **2** using shared PAEs or Memory as a destination
- Program partition 1 into XPP grid
- After calculation, remove partition 1
- Data is still available in shared ALU-PAE or in RAM-PAE
- Re-program XPP grid and compute part 2

Partial One-Cycle Reconfiguration Supports Adaptive Processing



Task 2

Task 3

Task 4

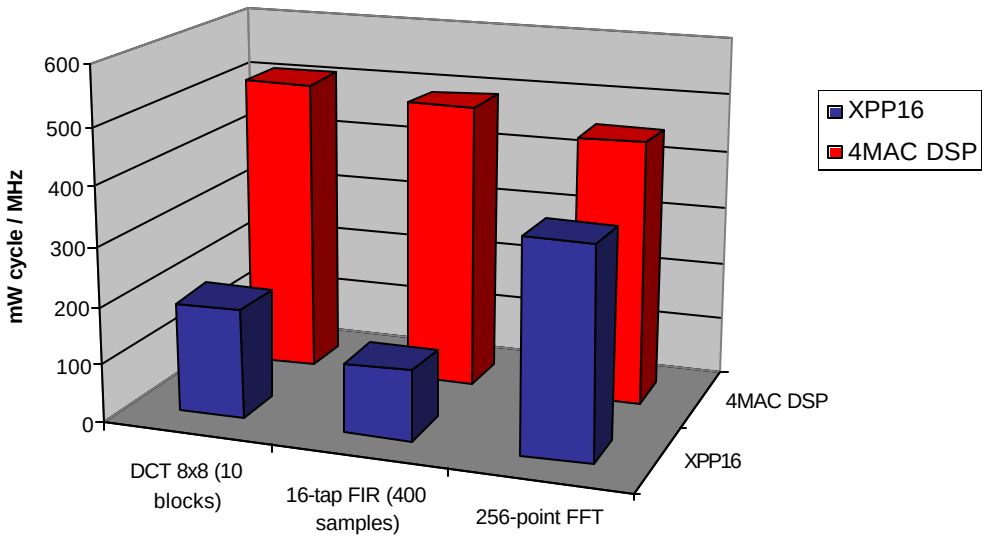
Task 5 is partially configured

waiting for Task 2 resources to become free

Configuration is completed

Power-Efficient DSP Computing

Algorithm		XPP16	4xMAC DSP
FFT 256-points	*mW/ MHz*	360	453
	cycles	1,200	1,619
MPEG Video 2D DCT (8x8)	*mW/ MHz*	19	51
	cycles	64	181
Real 16 Tap FIR Filter (40 Samples)	*mW/ MHz*	12	49
	cycles	40	176



- XPP trades clock frequency for high spacial parallelism
- Saves power by dramatically reducing need for
  - opcode fetch and decode
  - temporary data transfer to register/cache/memory

Key Engineering Advantages of eXtreme DSP

- Gaining performance by trading silicon space for higher clock
- Familiar C-language programming model for all grid sizes dramatically speeds up software development and verification
- Getting ASIC/FPGA-level optimal DSP performance combined with full or partial on the fly re-programming
- Elimination of the unnecessary gate switching delivers power efficient DSP computing
- Processor versions with different PAE grid sizes offer wide range of DSP performances with identical programming model

Addressing Critical Needs of COTS DSP Programs

eXtreme DSP architecture:

- Provides significant increases in DSP performance while lowering power consumption
- Drastically speeds-up design and upgrade cycles and simplifies technology upgrades for legacy products
- Makes DSP software portable between product generations
- Assures long-term economical viability of design by riding on future semiconductor density increases (the Moore's Law)