MOBILE SOFTWARE DEFINED RADIO SOLUTION USING HIGH-PERFORMANCE, LOW-POWER RECONFIGURABLE DSP ARCHITECTURE

Nader Bagherzadeh (Morpho Technologies, Irvine, CA, USA; naderb@morphotech.com); Tom Eichenberg (Morpho Technologies, Irvine, CA, USA; tome@morphotech.com)

ABSTRACT

In this paper we present Morpho Technologies' reconfigurable DSP platform for handling key baseband processing for mobile handheld devices that intend to feature multimode communications capabilities. PHY layer signal processing functions that were previously implemented in dedicated hardware blocks are now capable of running completely in software, in keeping with the philosophy of Software Defined Radio. First, we present an overview of Mopho's MS2 reconfigurable DSP architecture. The MS2 is a highly optimized and efficient parallel processing engine for meeting the real-time requirements of mobile baseband processing at very competitive power consumption levels. Next, we contrast the MS2 to FPGAs and traditional DSPs used in current SDR implementations. Lastly, we describe the detailed implementations of some of the most computationally intensive kernels that constitute the critical components of OFDM and GPS. The flexibility of our technology empowers SoC designers to not only meet the current multimode PHY layer requirements, but to create a platform that can accommodate future changes within these standards.

1. INTRODUCTION

Current implementations of the U.S. Department of Defenses' Joint Tactical Radio System (JTRS) [1] fall short of the goal to provide a software programmable radio in a mobile handheld unit, with high power consumption and high cost being the chief issues. Figure 1 shows a typical Software Defined Radio (SDR) architecture being designed today for the Physical Layer (PHY) and Medium Access Control (MAC) portions of the JTRS applications.

While architectures such as those in Figure 1 may be suitable for the initial research and proof-of-concept phases, these architectures which rely on Field Programmable Gate Arrays (FPGA) and traditional programmable Digital Signal Processors (DSP), suffer from high power consumption and significantly higher cost. For the JTRS Cluster 5 program, which is geared towards a small battery-powered handheld device, these architectures also face challenges integrating into small form factor designs being more suitable for a laptop or small PC. Hence, for the production units which are scheduled for FY2007, a more practical approach needs to be considered.

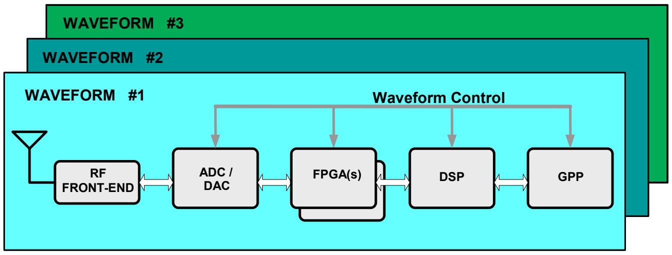


Figure 1. Current JTRS SDR implementation

A better approach that can meet the low-power and low-cost objectives of production units would be to deploy a new class of programmable parallel DSP, i.e., a reconfigurable DSP. Figure 2 illustrates a JTRS SDR implementation based on this approach.

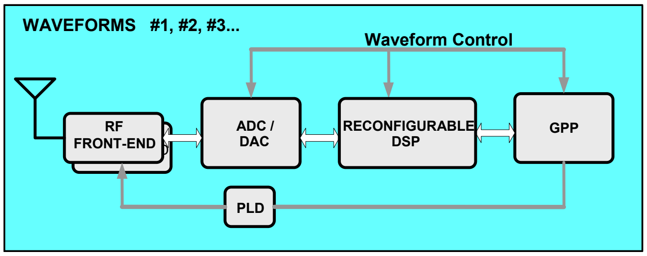


Figure 2. A better JTRS SDR implementation

In the implementations such as in Figure 1, the FPGAs perform the signal processing associated with channelization, including filtering, interpolation, decimation, correlation, Fast Fourier Transforms (FFT), etc.. The DSP typically handles the modulation and demodulation. Forward Error Correction (FEC) decode is usually done by a hardware acceleration block in the DSP.

The General Purpose Processor (GPP) is responsible for implementing the Medium Access Control (MAC) layer, the JTRS Software Communications Architecture (SCA) and associated CORBA framework, on top of a real-time operating system. Due to the JTRS requirement to support a variety of different waveforms, the Figure 1 type of implementation has to be brute-force replicated for each waveform [2].

In contrast, the implementation shown in Figure 2 utilizes a reconfigurable DSP to replace the power hungry FPGAs, as well as the traditional DSP. Multiple waveforms are supported by simply switching to a new context for the reconfigurable DSP. To support multiple RF front-ends, a small Programmable Logic Device (PLD) may be utilized.

2. MORPHO MS2 RECONFIGURABLE DSP

Morpho Technologies offers the low power, high performance MS2 reconfigurable DSP core as a means to implement the type of architecture shown in Figure 2. Previous generations of the MS2, DSP have been proven in commercial applications; for example, the MRC6011, a wireless basestation chipset from Freescale (Motorola) Semiconductor [3].

The MS2 DSP architecture shown in Figure 3, combines the power of a software programmable 32-bit RISC processor (mRISC) and a highly-parallel array of Reconfigurable Cells (RC) Array (16 Cells). Each RC cell has an Arithmetic Logic (ALU), multiply-and-accumulate (MAC) and Logic units that can be used for different wireless applications, switching from one application-specific set of instructions to another, on a single clock cycle. A special complex correlation unit is also included in each RC, accelerating waveforms such as WNW, Soldier Radio, Software GPS and MUOS.

In addition to the fundamental blocks, the MS2 solution offers a complete baseband processing subsystem. The MS2 has embedded specialized, programmable blocks for baseband processing such as: a flexible I/Q Input Buffer, a programmable Sequence Generator block and a programmable Interleaver engine. A set of generic peripheral blocks complement the signal processing blocks: Multi-master DMA controller, Interrupt Controller and Timers. For software development, the MS2 comes with extensive built-in hardware support for debug and trace.

The MS2 software development tools are based on the Eclipse IDE and GNUPro® tool set using the C/C++ programming language, creating a very familiar programming environment to DSP programmers. The RC Array is programmed using the Morpho-C syntax, which is an extension to the C language that makes parallel processing easier to implement. Development tools include a cycle-accurate software simulator and testchip-based hardware development platforms.

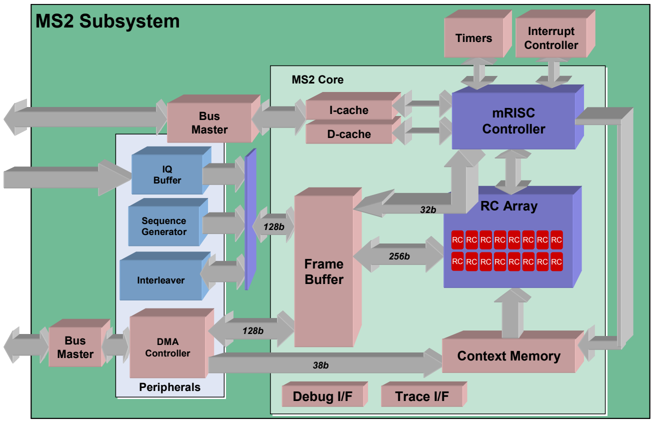


Figure 3. MS2 Subsystem Block Diagram

An extensive optimized kernel library for commonly used math functions, signal processing functions, and wireless communications specific algorithms rounds out the offering.

2.1. MS2 Operation

The processing element within the RC Array is a Reconfigurable Cell (RC). Each RC has several functional units (e.g. MAC, ALU, etc.), a small register file, and is configured through a 32-bit context word. 16 RCs are grouped into a 2 x 8 matrix, called the RC Array.

Reconfiguration of the RC Array is accomplished in a single cycle by caching within the core several "contexts" from off-chip program memory. Each context provides the programming information necessary to configure and control the RC Array.

The Frame Buffer is similar to an internal data cache for the RC Array and is implemented as a two-port memory. This architecture provides for transparent memory accesses by overlapping of computation with the data load and store. The Frame Buffer is organized as 5 banks and can be sized as required by the customer. The Frame Buffer can provide all 16 RCs with data - either as two 8-bit operands or one 16-bit operand – on every clock cycle.

The Context Memory is the local memory to store the configuration contexts of the RC Array. It is similar to an instruction cache for the RC Array. The context word from a context set is broadcast to all RCs in the corresponding row or column. Thus, all RCs in a row or column share a context word and perform the same operation as shown in Figure 4. This provides the ability for the array to operate in Single Instruction, Multiple Data (SIMD) mode. The context memory has a 2-port interface that enables the loading of new contexts from off-chip memory (e.g. Flash memory) while the array executes other instructions.

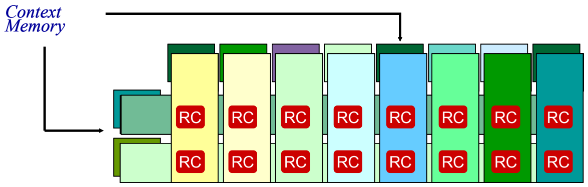


Figure 4. Context memory and RC Array

RC cells in the array are connected in 2 levels of hierarchy. The RC cells are grouped in a 2x4 block. The cells within each group are fully connected to each neighboring cell in a row or column. The two groups are connected to each other via fast lanes - there are two fast lanes in each direction that enable a cell in a group to broadcast its results to the cells in the other group.

The controlling component of the MS2 is the 32-bit mRISC processor. The mRISC handles general-purpose operations and also controls the operation of the RC Array. It initiates all data transfers to and from the Frame Buffer and the loading of contexts to the Context Memory through the DMA Controller. When not executing normal RISC instructions, the mRISC processor controls the execution of operations inside the RC Array on a cycle-by-cycle basis by issuing special instructions that broadcast SIMD contexts to the RCs or load data between the Frame Buffer and the RC Array. This provides for a simple programming model since only one thread of control flow is running through the system at any given time.

2.2. MS2 Code Development

The first step in mapping an application to the MS2 is to identify the parts of the application that lend themselves to efficient SIMD processing. Obvious examples include filters, correlators, and transformations. The applications are divided into sequences of SIMD instructions that are stored in the context memory. The sequencing of these operations is then implemented as special instructions running on the mRISC processor. The remaining parts of the application are implemented as normal RISC instructions.

The MS2 kernel library simplifies the application mapping as shown in Figure 5.

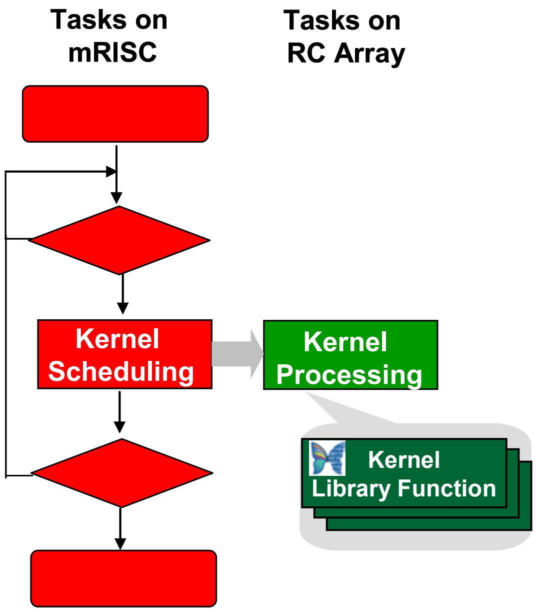


Figure 5. Unified simplified system programming view

3. COMPARING MS2 TO FPGAS AND DSPS

The architecture and implementation of the MS2 allows it to be 15X – 20X more power efficient than even the leading FPGAs. For example, consider the complex correlation function, which is useful for several communications standards, including GPS. Each complex correlation requires 4 multiplies and 2 additions, for an equivalent of 6 operations.

The MS2 can perform 64 8-bit complex correlations in a single clock cycle. At its maximum rated clock speed of 250MHz, this equates to 96,000 MOPS. With a power rating of 0.5mW/MHz, the MS2 achieves a figure-of-merit of 768 MOPS/mW.

Xilinx's Virtex-4, which is considered an industry leading "low-power" FPGA, provides a Multiply-Accumulate (MAC) function as an embedded hard IP block. This MAC block has a power rating of 5.7mW @ 100MHz [4]. Since each MAC operation is considered to be 2 operations, the equivalent figure-of-merit for the Virtex-4 equates to 35 MOPS/mW.

Next, the MS2 subsystem can be compared to an industry leading programmable DSP, such as the TMS320C641x from Texas Instruments. The C641x has 8 MAC units in its CPU for a computational performance of 16 operations per second. The C641x's CPU and L1 cache section has a power rating of 0.44mW/MHz [5], giving it a figure-of-merit of 36.4 MOPS/mW.

Table 1 lists the respective complex-correlation figures-of-merit for all three solutions. All solutions are in 90nm CMOS process.

Table 1. Complex-Correlation Comparison

Benchmark	Morpho MS2 DSP	Xilinx Virtex-4	TI C641x
8-bit Complex Correlation MOPS per mW	768	35	36.4

A similar analysis is conducted for performing the FFT, which is central to all OFDM communications standards. For example, consider the case of a complex 1024-pt 16-bit FFT (1kCFFT).

The MS2 takes only 3080 cycles to complete the FFT, giving a throughput of 81,169 1kCFFTs/sec at 250MHz clock rate. At its rated power consumption, the MS2 achieves 650 1kCFFTs/sec/mW.

In comparison, the Virtex-4 at 125MHz achieves 125,000 1kCFFTs/sec [6]. Based on the FPGA resources needed, the Virtex-4 would consume 619mW [7], giving a 202 1kCFFTs/sec/mW figure-of-merit.

The TI C641X can compute a 1kCFFT in 6002 clock cycles [8], yielding a figure-of-merit of 378 1kCFFTs/sec/mW. Table 2 lists the respective FFT figures-of-merit for all three solutions. All solutions are in 90nm CMOS process.

Table 2. FFT Comparison

Benchmark	Morpho MS2 DSP	Xilinx Virtex-4	TI C641x
1024 Complex FFTs per second per mW	650	202	378

4. MS2 KERNEL LIBRARY

Morpho has developed an extensive optimized kernel library with the following features:

- Optimized C-callable routines
- PHY layer signal processing functions
    - Commonly used functions
    - Application specific functions
- Licensable source code
    - Can be modified by user
- Documented cycles/code size metrics
- Exhaustively tested against reference C model

4.1 OFDM Receiver

Morpho has an ongoing development activity in implementing a robust kernel library for the OFDM broadband communications protocols. In particular, the WiMax and WiBro standards (IEEE 802.16-2004 and IEEE 802.16e, respectively) have been extensively modeled and studied. The kernel library functions implement the receiver chain as shown in Figure 5. This entire receiver chain can be completely handled in software on the MS2 consuming just 125mW.

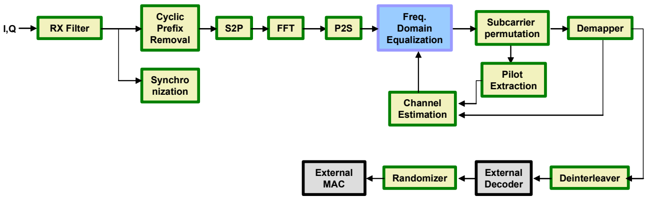


Figure 5. OFDM Receiver Chain

4.2 GPS Receiver

Morpho has extensively studied and implemented critical sections of the civilian (L1) GPS receiver. The baseband processing for GPS can be completely handled in software on the MS2, at a loading of only 15% for a typical autonomous (unassisted) outdoor tracking. Time-to-fix (i.e. acquisition) is approximately 50msec.

An optimized kernel library implementing the GPS receiver chain is shown in Figure 6.

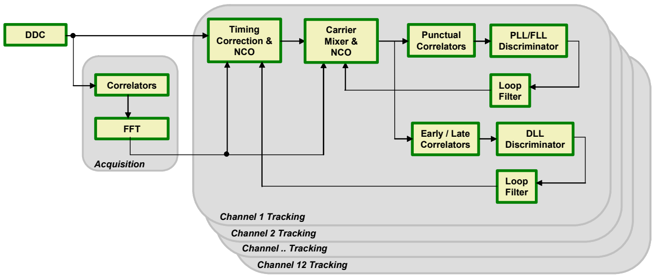


Figure 6. GPS Receiver Chain

4. CONCLUSION

We have demonstrated that the MS2 reconfigurable DSP meets the power and performance challenges of next generation multimode communications devices using a software defined radio approach.

5. REFERENCES

[1] www.jtrs.org

[2] M. Uhm, J. Belzile, “Meeting Software Defined Radio cost and power targets: Making SDR feasible”, *Military Embedded Systems*, June 2005

[3] www.freescale.com/webapp/sps/site/prod_summary.jsp ?code=MRC6011&nodeId=012795LCWs

[4]http://www.xilinx-china.com/esp/wireless/collateral/ wireless_networking_app.pdf, pages 31-32

[5] TMS320C6414T/15T/16T Power Consumption Summary, Application Report SPRAA45, August 2004, http://focus.ti.com/lit/an/spraa45/spraa45.pdf

[6] Dillon Engineering, ”Ultra High-Performance FFT/IFFT IP Core”, http://www.dilloneng.com/documents/ fft_spec.pdf

[7] Results of Xilinx Power Estimator Worksheet for Virtex-4, http://www.origin.xilinx.com/cgi-bin/power_tool/ power_Virtex4

[8] TMS320C64x DSP Library Programmer’s Reference, Document SPRU565B, October 2003, http://focus.ti.com/lit/ug/spru565b/spru565b.pdf

Mobile Software Defined Radio Solution Using High-Performance, Low-Power Reconfigurable DSP Architecture

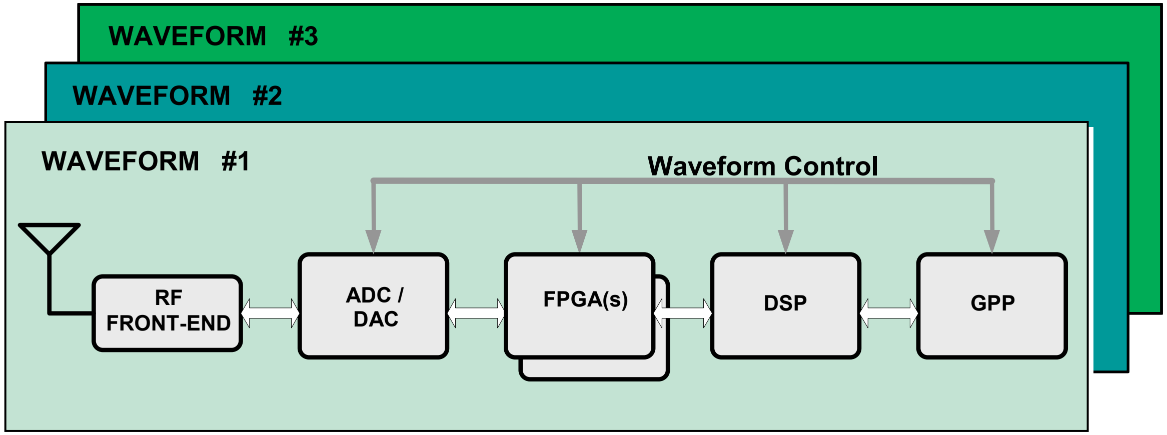
Nader Bagherzadeh, Tom Eichenberg
Morpho Technologies

2005 Software Defined Radio Technical Conference

JTRS SDR Architectures

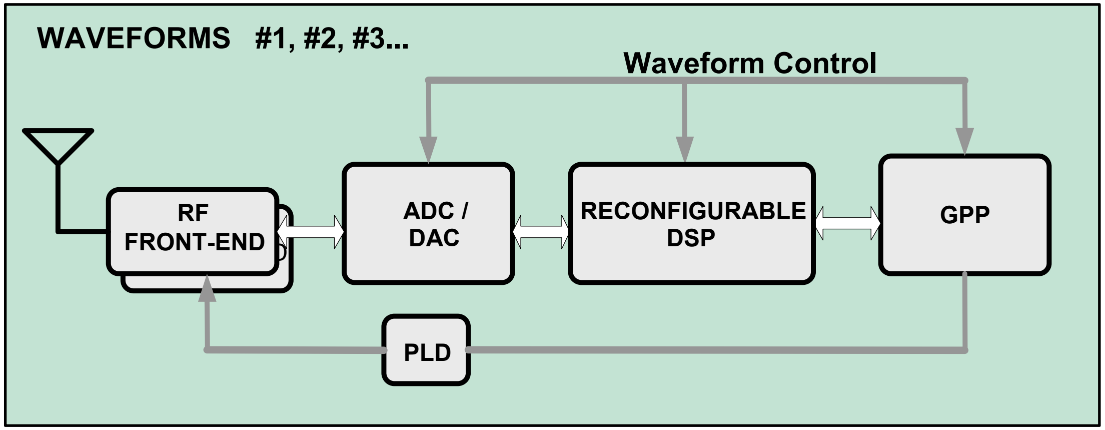
<u>Current JTRS SDR Solution</u>

- High power consumption
- High cost
- Not suitable for handheld units

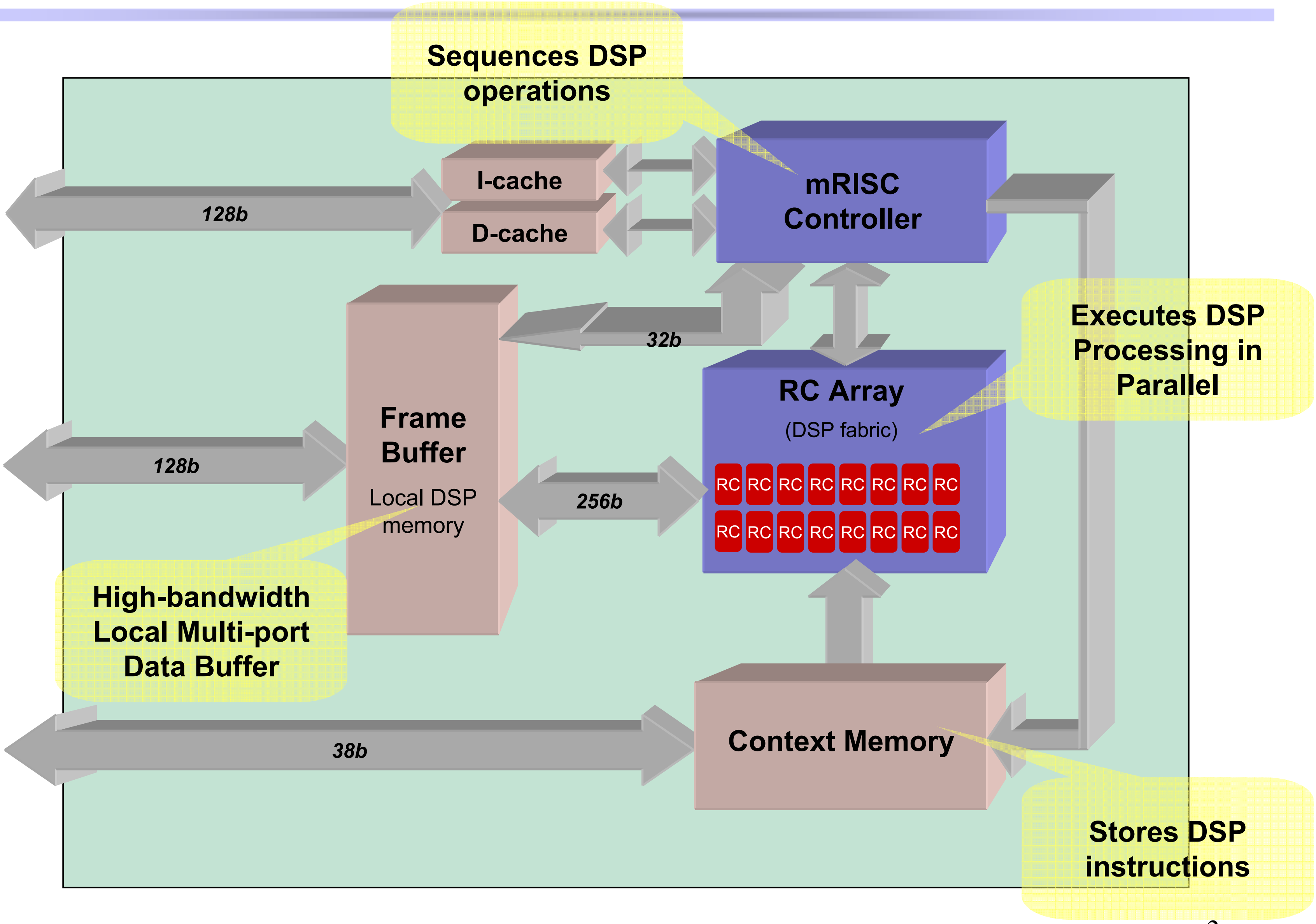


<u>Better JTRS SDR Solution</u>

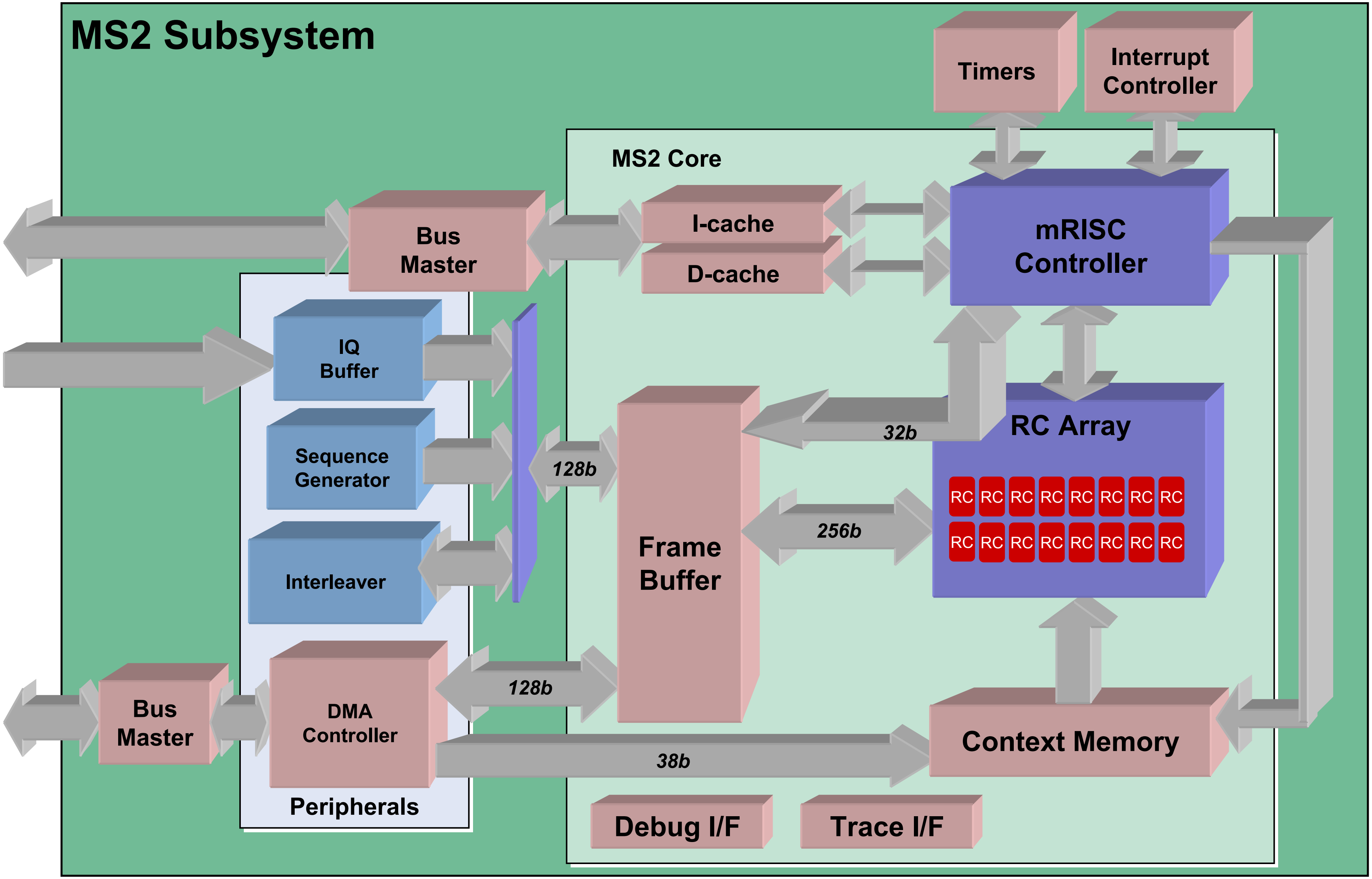
- Low power consumption
- Lower cost
- Meets handheld formfactor



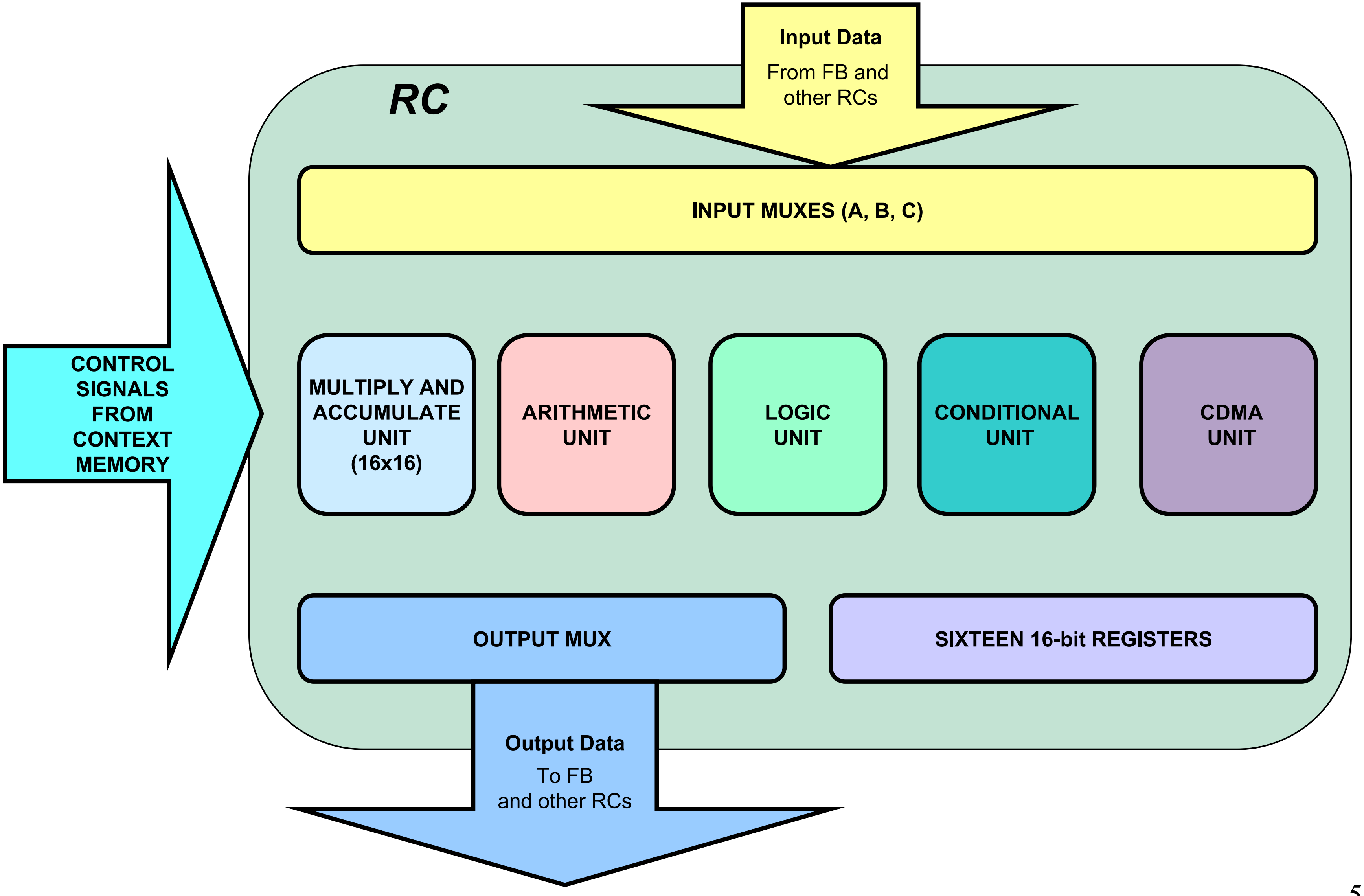
Morpho MS2 Core



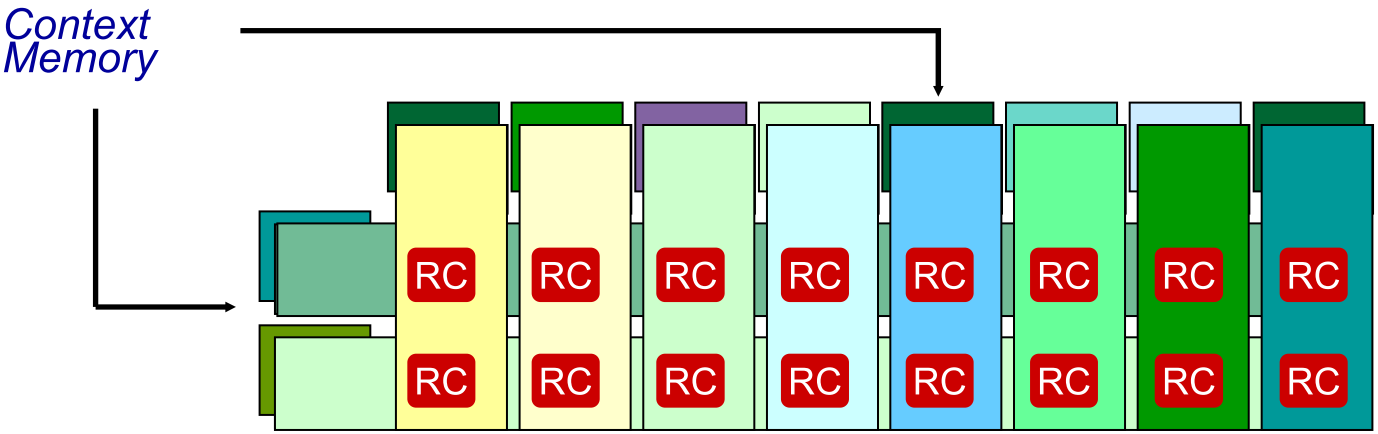
MS2 Subsystem



MS2 Reconfigurable Cell

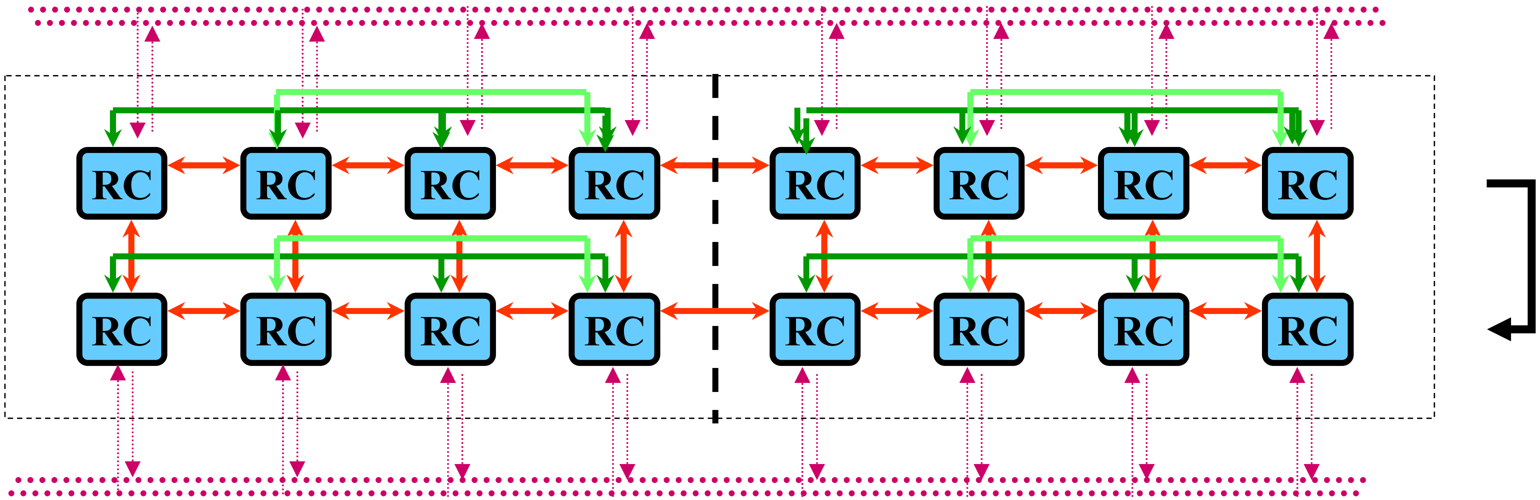


RC Array



- **Flexible modes that can be reconfigured every cycle:**
- **SIMD (Single Instruction, Multiple Data) <u>Row</u> Context Mode**
- **SIMD <u>Column</u> Context Mode**

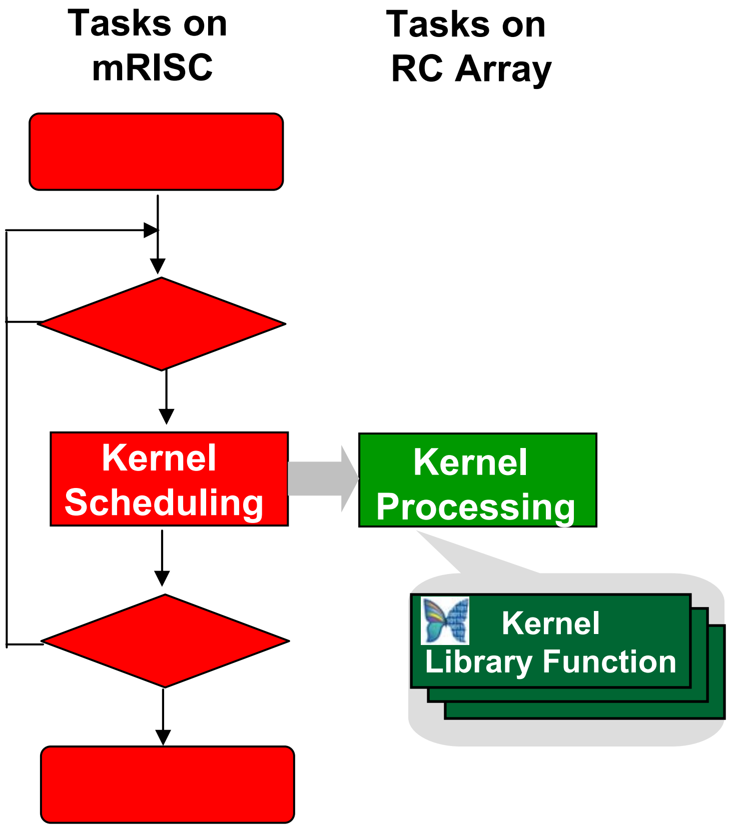
Connectivity in the RC Array



- Full nearest-neighbor connectivity (red)
- Full row and column connectivity between 4x4 partition (green)
- Wrap-around connectivity (black)
- Fast lanes between 4x4 partitions (purple)

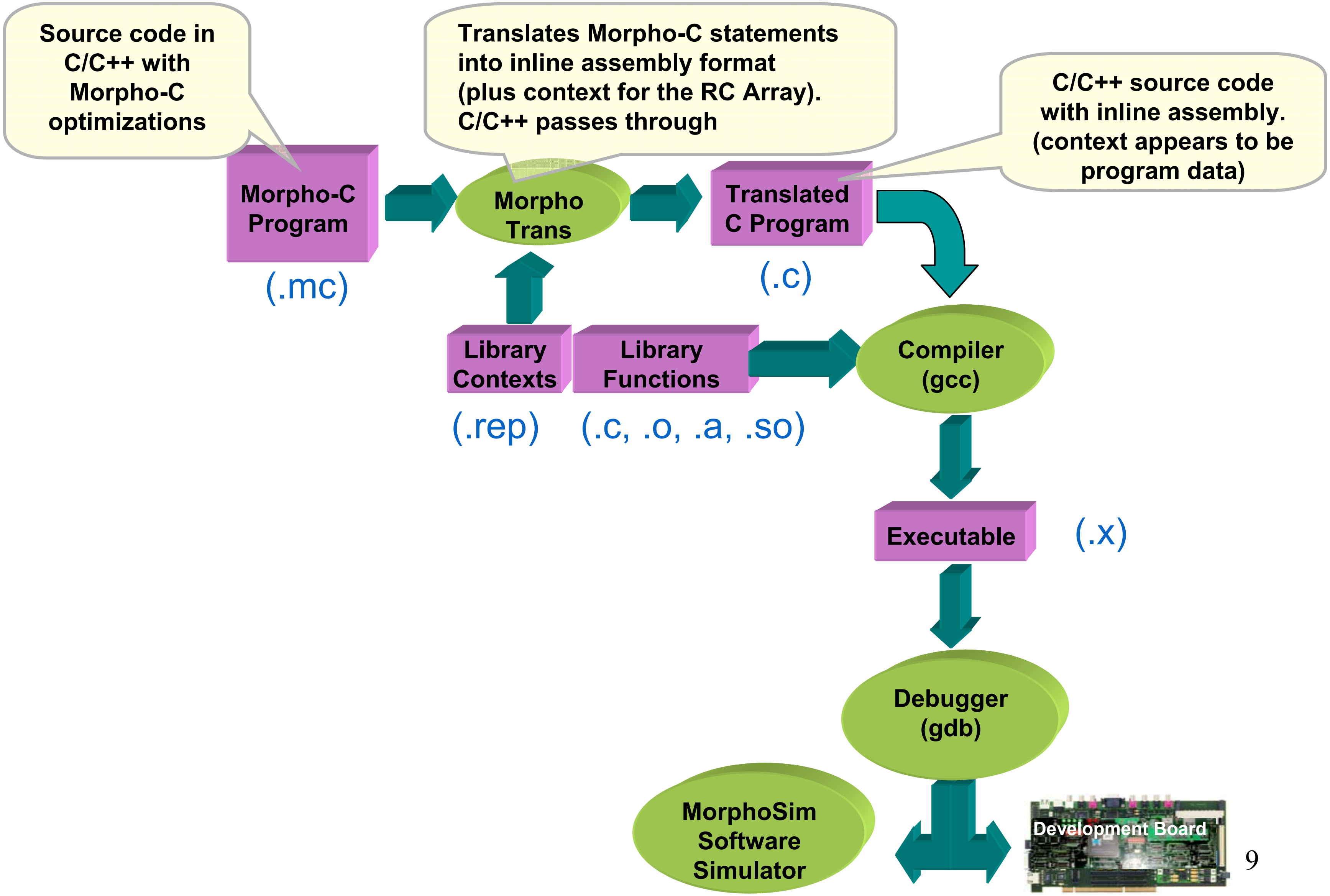
MS2 Programming

- **Code development methodlogy**
  - **mRISC:** Integration and scheduling tasks like control of peripherals, loop handling, initialization ...
    Programmed in C (or ASM)
  - **RC array:** Data path signal processing tasks with high degree of data parallelism.
    Programmed in Morpho-C
- **Kernel libraries from Morpho**
  - Optimized C-callable routines
  - Signal processing functions
    - Commonly used functions
    - Application specific functions
  - Source code licensed
    - Can be modified by user
  - Cycles/code size metrics avail
  - Tested against reference C model
- **Example apps from Morpho**
  - Shows data arrangement, data movement, integration of kernel library functions



Unified simplified system programming view

MS2 Development Tools



Source code in C/C++ with Morpho-C optimizations

Translates Morpho-C statements into inline assembly format (plus context for the RC Array). C/C++ passes through

C/C++ source code with inline assembly. (context appears to be program data)

Morpho-C Program

(.mc)

Morpho Trans

Translated C Program

(.c)

Library Contexts

(.rep)

Library Functions

(.c, .o, .a, .so)

Compiler (gcc)

Executable

(.x)

Debugger (gdb)

MorphoSim Software Simulator

Development Board

MS2 Performance

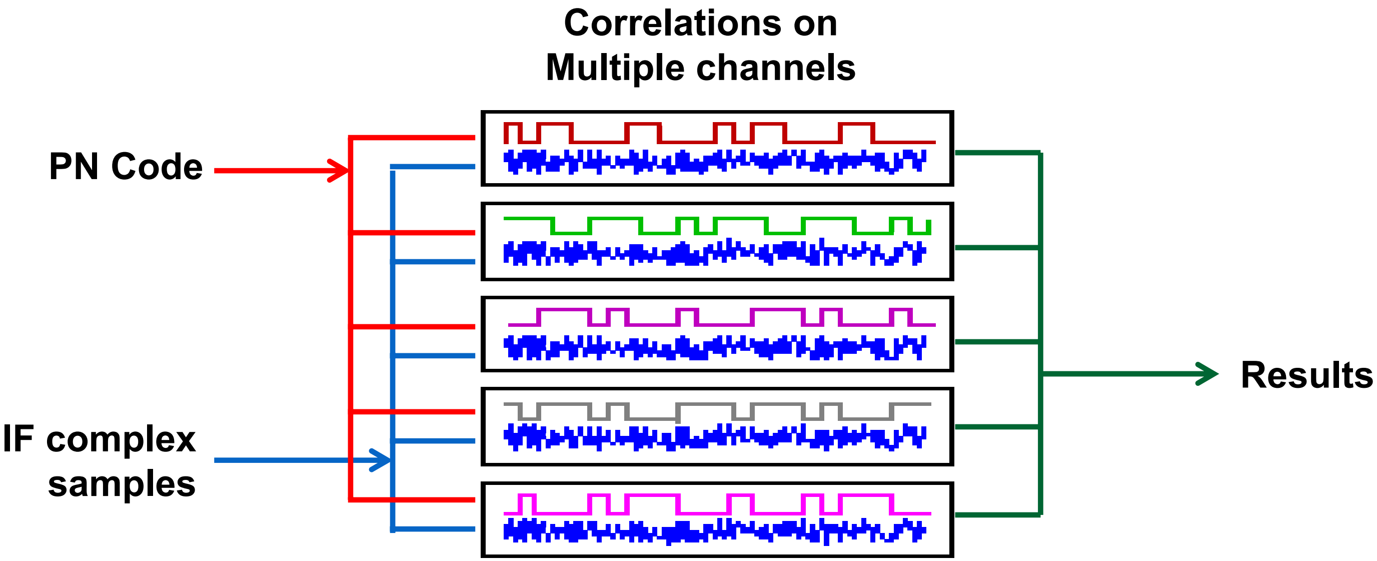
Statistics

- 96 GOPS @ 250MHz
- 384 OPS/CYCLE
- 0.5mW/MHz
- 0.8 GOPS/mW

Benchmarks

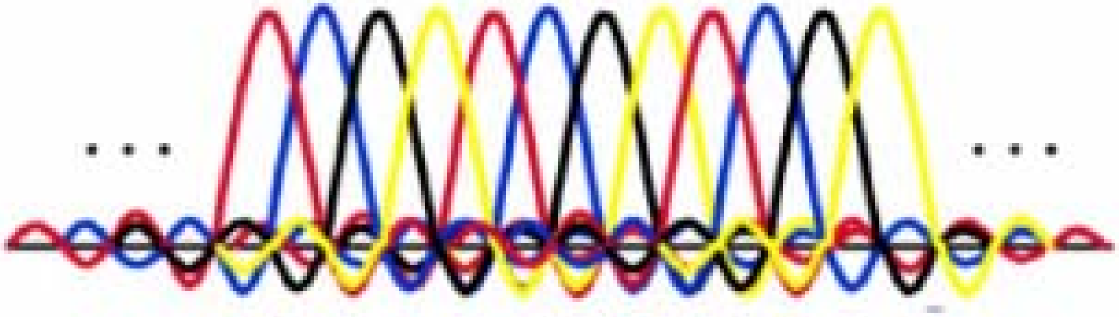
- 1024-pt Complex FFT :
  - 3080 cc
- 48-tap Complex RRC :
  - 3420 cc/512 samples
- 8-finger WCDMA Rake :
  - 1653 cc/256 chips
- Viterbi Decoder (K = 7, ½ Rate) :
  - 25 cc/bit

Complex Correlation



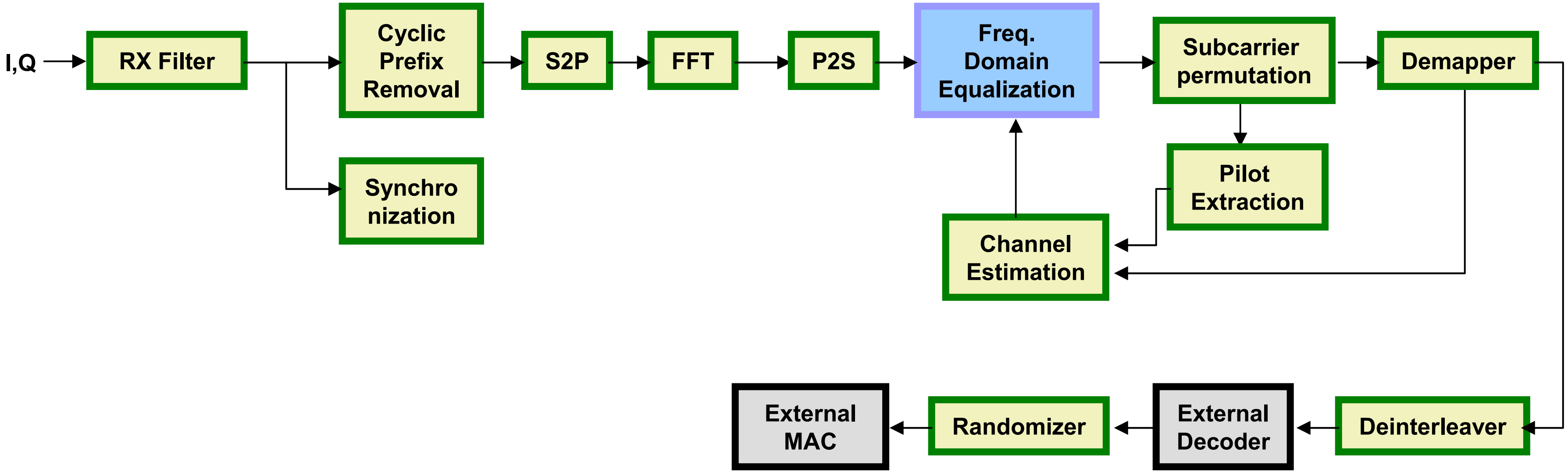
Benchmark	Morpho MS2 DSP	Xilinx Virtex-4	TI C641x
8-bit Complex Correlation MOPS per mW	768	35	36.4

1024-pt Complex FFT



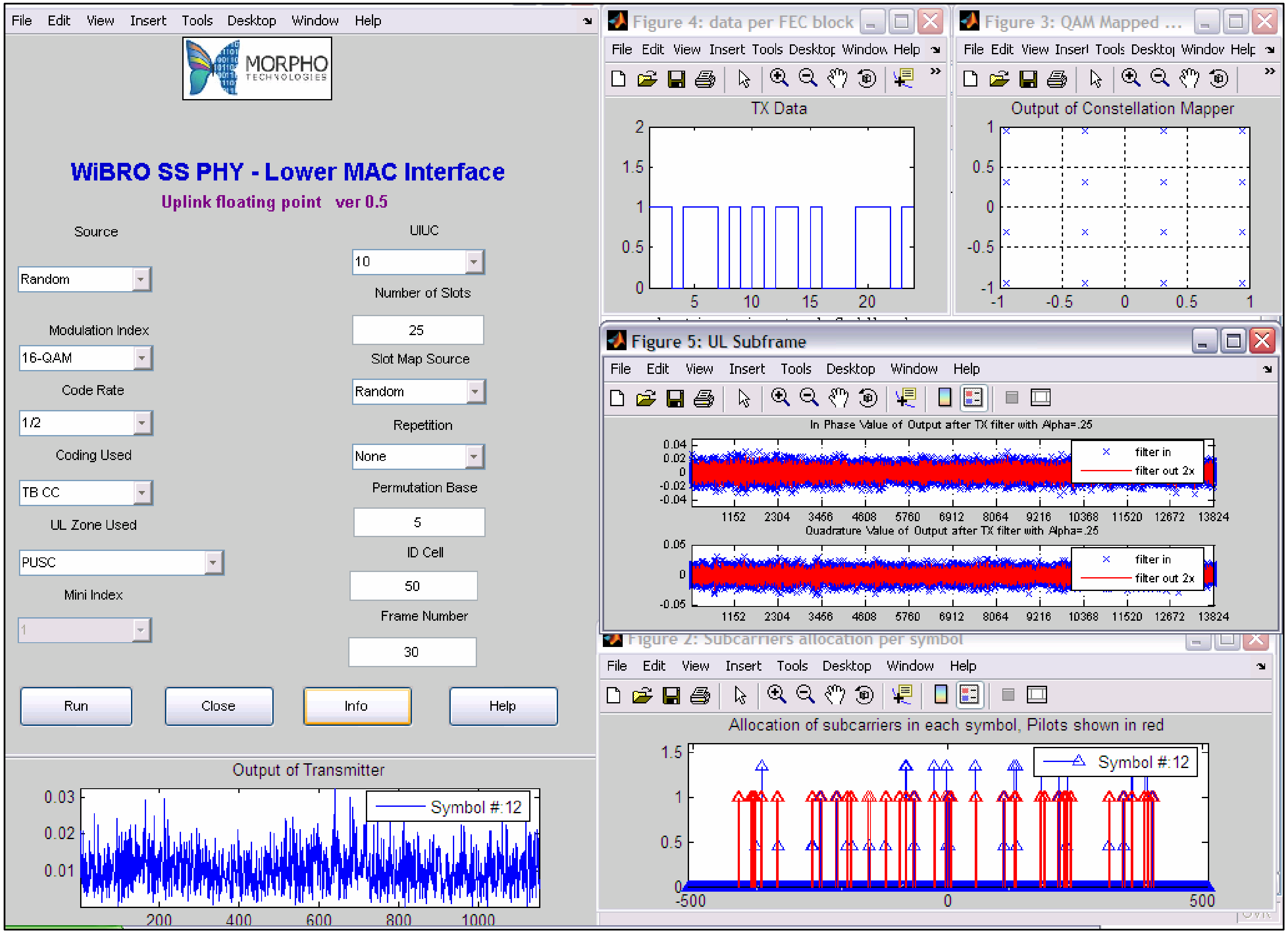
Benchmark	Morpho MS2 DSP	Xilinx Virtex-4	TI C641x
1024 Complex FFTs per second per mW	650	202	378

OFDM Receiver Chain



- 802.16e WiBro PHY layer in software at 125mW

802.16e WiBro PHY Simulation



WiBRO SS PHY - Lower MAC Interface

Uplink floating point ver 0.5

Source: Random

Modulation Index: 16-QAM

Code Rate: 1/2

Coding Used: TB CC

UL Zone Used: PUSC

Mini Index: 1

UIUC: 10

Number of Slots: 25

Slot Map Source: Random

Repetition: None

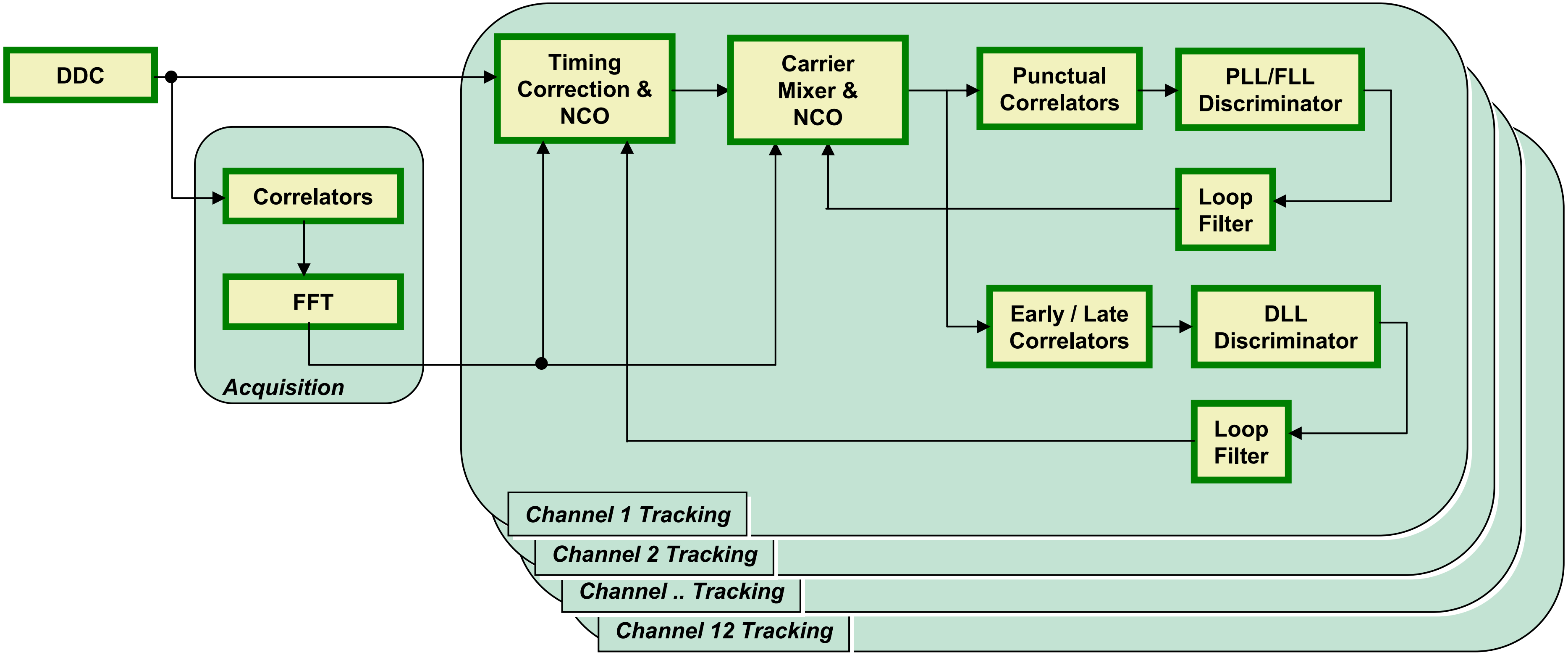
Permutation Base: 5

ID Cell: 50

Frame Number: 30

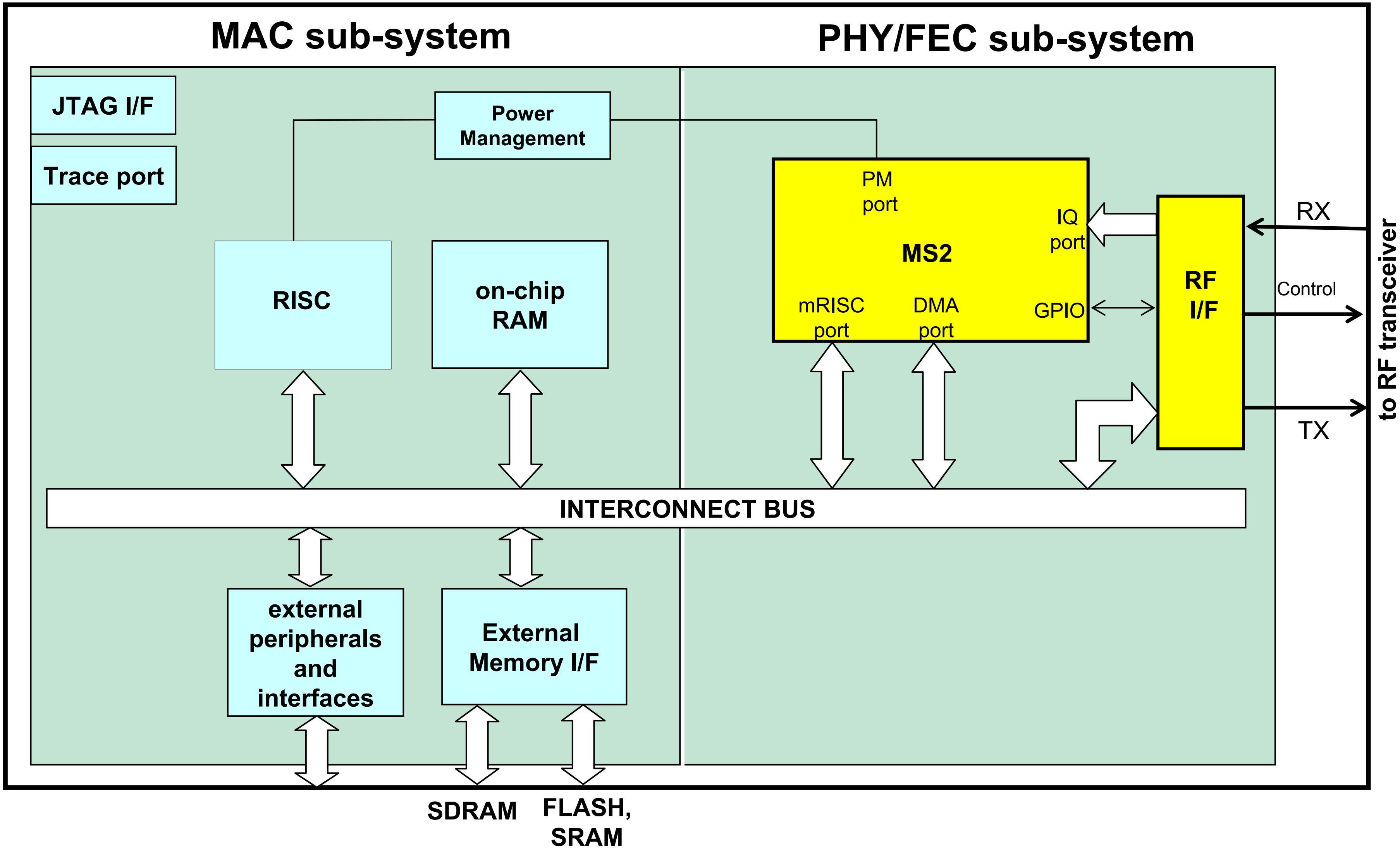
File Edit View Insert Tools Desktop Window Help
MORPHO TECHNOLOGIES
Run
Close
Info
Help
Output of Transmitter
Symbol #:12
0.03
0.02
0.01
200
400
600
800
1000
Figure 4: data per FEC block
TX Data
2
1.5
1
0.5
0
5
10
15
20
Figure 3: QAM Mapped ...
Output of Constellation Mapper
1
0.5
0
-0.5
-1
Figure 5: UL Subframe
In Phase Value of Output after TX filter with Alpha=.25
filter in
filter out 2x
0.04
0.02
0
-0.02
-0.04
1152
2304
3456
4608
5760
6912
8064
9216
10368
11520
12672
13824
Quadrature Value of Output after TX filter with Alpha=.25
0.05
0
-0.05
Figure 2: Subcarriers allocation per symbol
Allocation of subcarriers in each symbol, Pilots shown in red
Symbol #:12
1.5
1
0.5
0
-500
0
500

GPS Receiver Chain



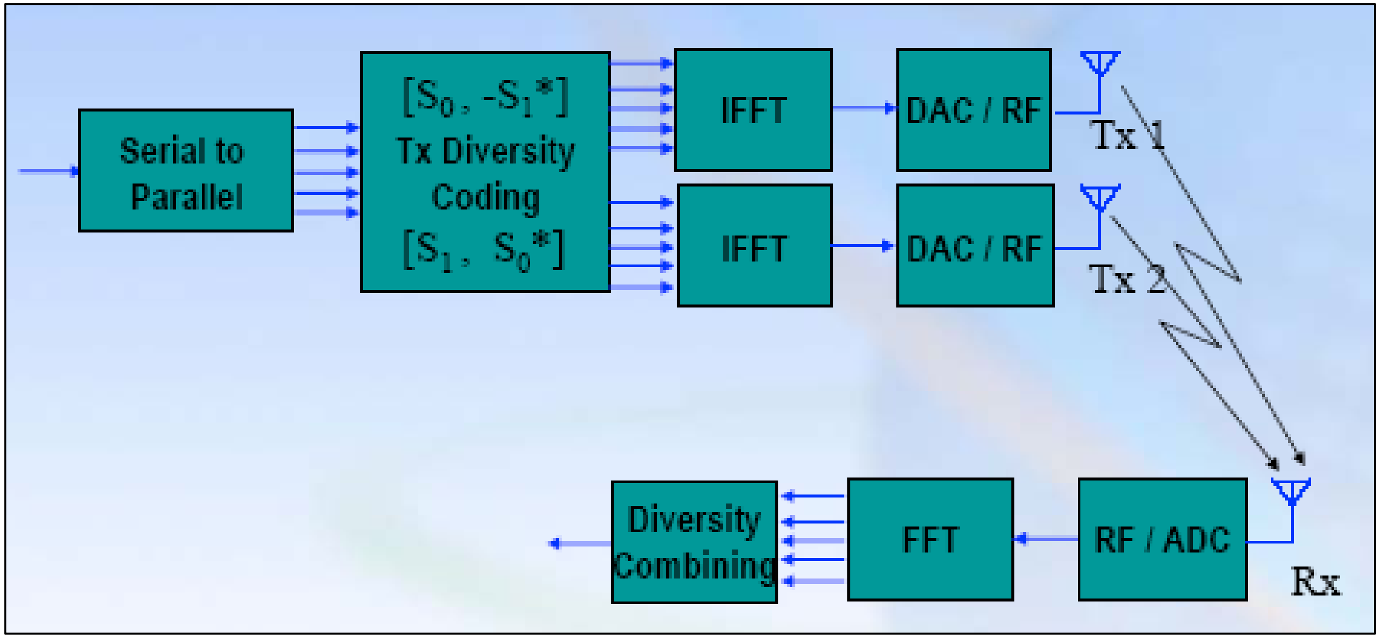
- Only 15% loading for autonomous unassisted outdoor tracking
- Approximately 50msec time-to-fix (acquisition)

MS2-based SOC



MS2 and Advanced Features

- Multiple Input Multiple Output (MIMO)
- Adaptive Antenna
- Space-Time Coding



Thank You