# A Scalable Implementation of a Reconfigurable WCDMA Rake Receiver

**Marc Quax, Jos Huisken & Jef van Meerbergen**

## Abstract

*The demands in terms of processing performance, communication bandwidth and real-time throughput of new generation mobile communication applications (mobile and base-stations) are much higher than today's programmable processing architectures can deliver. On the other hand standards and market uncertainties, non-recurring engineering costs, and lack of access to (or knowledge of) application IP will require the next generation of embedded computing platforms to be fully programmable. In terms of silicon cost and power, practical yet fully programmable embedded computing platforms are enabled by reconfigurable processors that replace fixed ASICs in current standard platforms [8]. This paper explains the concepts behind a novel reconfigurable WCDMA Rake receiver and gives benchmark results. The proposed Rake receiver enables a high performance, yet flexible computing platform for WCDMA.*

## 1 Introduction

Code division multiple access (CDMA) is used in spread spectrum systems to enable multiple access. It is a transmission technique in which the frequency spectrum of a data signal is spread using a code uncorrelated with that of the user data signal. Different spread spectrum techniques exists: Direct Sequence (DS), Frequency Hopping (FH), Time Hopping (TH) and multi carrier CDMA (MC-CDMA). In this paper the direct sequence (DS) frequency-division duplex (FDD) mode of the wideband code-division multiple access (WCDMA) standard [1, 2] is considered. In the transmitter the user information bits (symbols) are spread over a wide frequency bandwidth by directly multiplying the user data bits with a spreading code sequence at high rate (see Figure 1). The Rake receivers are used for increasing multipath diversity which is important to increase the capacity of a CDMA system. Traditionally a Rake receiver consists of a number of Rake fingers (see Figure 2), each assigned to a different multipath signal, and a maximum ratio combiner for coherently combining the outputs of each finger. Each multipath component is despread by correlating the received signal with the spreading code over a period corresponding to the spreading factor (SF).

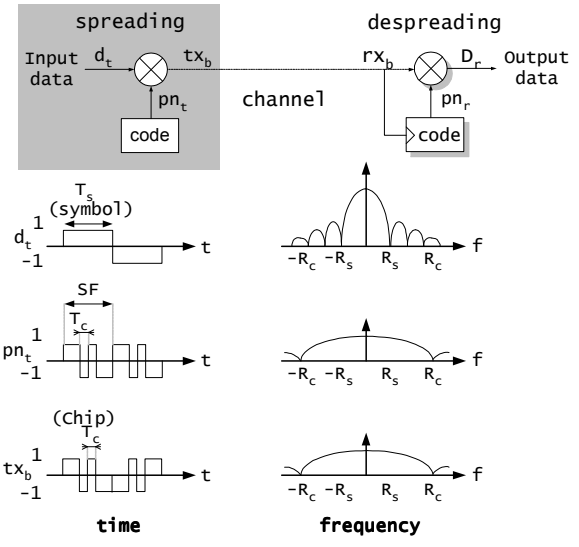



**Figure 1** Spreading and despreading

After the despreading, maximum ratio combining (MRC) is applied to the symbols from the fingers. The phases of the symbols are aligned, and amplitudes are weighted according to a channel estimate from the channel estimator. From the implementation point of view many challenges have to be faced to ensure overall receiver flexibility. Wideband CDMA applications require the execution of a large number of tasks at high data rate. The amount of processing power demanded by such applications is so large that executing multiple Rake receivers on a traditional DSP is not feasible. The reason for this is mainly the sequential behaviour of the application, due to the conditional operations, and not having domain specific operations. Much higher performance is obtained by executing multiple tasks in parallel. This can be exploited in a multiprocessor approach. However, this approach increases internal communication, often leading to performance and scalability bottlenecks. This paper presents a novel VLIW DSP core that can exploit the data parallelism by placing the conditional operations not in the control path of the processor but in the data-path. The processor and its associated programming tools were created using a proprietary automatic processor and tool generation flow. An innovative C-compiler, generated from the same methodology, aggressively exploits massive instruction parallelism in the Rake application kernel mapped onto the

1530-1591/04 $20.00 (c) 2004 IEEE

core. The architecture is scalable and flexible enough to target different standards.

## 2 Related work

### 2.1 Conventional Rake receiver

A conventional Rake receiver is depicted in Figure 2. The input to the Rake receiver is a baseband sample stream of oversampled chips (see Figure 1).

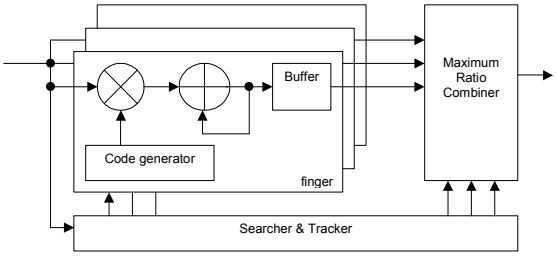



**Figure 2** Block diagram of a conventional Rake receiver based on multiple parallel Rake fingers

In order to receive several multipath components of the transmitted signal, a dedicated Rake finger is allocated to each of the tracked multipath components. Thus the Rake finger count corresponds to the maximum number of multipaths which is typically between two and six [3, 4]. In a Rake finger, the received samples are correlated with a time aligned wideband spreading code and integrated over a period corresponding to the spreading factor. Because the delay spread can be several times longer than the symbol integration period, the symbol dumps for a specific data symbol are completed at different times.

Therefore, each finger stores symbol dumps in a deskew buffer (random access FIFO buffer) from which they can be accessed for channel correction and maximum ratio combining after all multipath symbol dumps are available [3, 4]. The maximum delay spread and the lowest spreading factor supported by a Rake finger specifies the size of the deskew buffer.

### 2.2 FlexRake receiver

Instead of using a number of dedicated fingers, FlexRake receiver performs correlation operations sequentially by accessing a buffer that serves as a time-sliding window to the received I/Q samples [5]. This makes the FlexRake more efficient than the standard conventional Rake receiver approach. The FlexRake consists of a sample buffer and a correlator engine. The main optimisation is that it uses one correlator engine that works time multiplexed between the Rake fingers that are active.

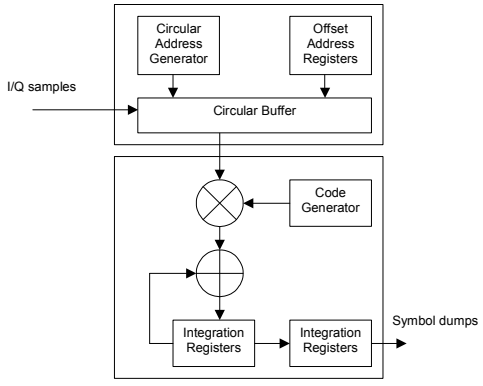



**Figure 3** FlexRake architecture

This means that, to be able to process up to eight active fingers, the correlator engine needs to be eight times faster than the conventional Rake receiver. The correlator engine consists of code generators (for both scrambling and channelization code), a complex multiplier, a number of integration registers (one for each finger) and a deskew buffer for the correlated symbol dumps.

The correlator performs a complex multiplication of the chips with the combined scrambling and channelization code. The result is accumulated with the previous correlated sample and stored in the integration registers for that particular finger. Then the next finger is processed. When one finger has integrated over the spreading factor, the resulting symbol is stored in the deskew buffer for later combining.

### 2.3 Post-buffer Rake receiver, ASIC implementation

The Post-buffer Rake receiver takes a different approach to the buffering problem [6]. If the oversampling factor ($O_s$) is eight and the maximum delay (Delayspread, $D_c$) between two fingers is 296 $T_c$, then the sample buffer used in the FlexRake receiver needs to be:

$$O_s \cdot D_c = 2368 \text{ samples}$$

With a sample width of 2*6 bits, the total memory size is 28416 bits. The memory access rate is:

$$\frac{\left(O_s + N_{fingers}\right)}{T_c} = 61.44\text{MHz}$$

where $N_{fingers}$ is the number of active fingers.

By processing each finger individually (like a conventional Rake receiver), and keeping track of their respective delays, it is possible to do the time alignment at symbol dump resolution. Since the time alignment is done at symbol rate, instead of sample rate, the memory access is much lower. The memory size used by the Post buffer Rake receiver is [6]:

$$\frac{\tau_{max}}{SF_{min}} \cdot 2 \cdot \text{symbol}_{width} = 4736 \text{ [bits]}$$

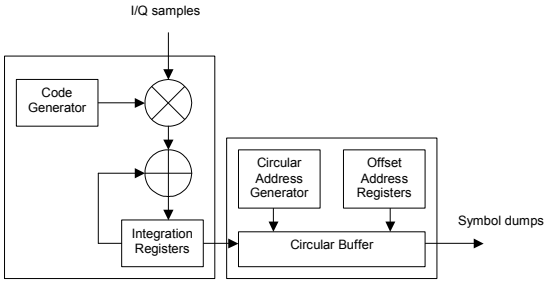



**Figure 4** Post buffer Rake receiver

## 3 Reconfigurable WCDMA Rake receiver

The demands in terms of processing performance, communication bandwidth and real-time throughput of the WCDMA Rake receiver is much higher than today's programmable processing architectures can deliver. The architectures presented in the previous chapter are all optimised RAKE solutions with little or no flexibility to map other applications.

At the same time standards and market uncertainties, non-recurring engineering costs, and lack of access to (or knowledge of) application IP will require the next generation of embedded computing platforms to be fully programmable.

The presented reconfigurable WCDMA Rake processor architecture is different to all previous WCDMA Rake receiver designs because it is a programmable VLIW digital signal processor (DSP) optimised for WCDMA applications. The processor performance is increased by executing multiple operations in parallel.

### 3.1 Scalability of design

#### 3.1.1 Problem

The limitations on scalability of a design depend in general on the existence of resource dependencies. Shared resources in a processor create resource dependencies, e.g. when multiple functional units try to access a shared resource. These operations have to be scheduled sequentially. These shared resources form a bottleneck in the scalability of a design. In the FlexRake architecture the shared memory forms the bottleneck in scalability.

#### 3.1.2 Solution

In the case of mapping a Rake application to a scalable processor template, locality of reference is important. Otherwise internal communication forms the bottleneck in scalability. There are multiple solutions to map Rake algorithms. The solutions can be divided into different groups, each with its own pros and cons:

- Alignment at chip level.
  - One memory before correlation.
- Alignment at symbol level.
  - Distributed memory after correlation.
  - Distributed memory in MRC.

Alignment at chip level is a solution with one shared memory, like the FlexRake solution. The memory stores the incoming over-sampled chips. The samples from memory are delivered to each Rake finger.

This implementation does not scale with the number of fingers mapped because of the presence of one shared memory and its limitations on memory access bandwidth.

Implementation of the alignment at symbol level can be done in two ways.

The first solution is placing the alignment memory in the finger after the despreading operation but before the maximum ratio combination. Basically a deskew buffer for each finger as in a conventional Rake receiver.

The second solution is placing the memory in the maximum ratio combining. The despreading operations belonging to one Rake receiver are executed sequentially. The symbol dumps are, after accumulation with partially combined symbols from memory, stored in the same memory location. In this way the memory is more effectively used. The maximum ratio combiner stores the partial accumulated symbols in a circular buffer, which is implemented in a similar way to the circular buffer in the Post buffer solution. Each Rake receiver has a dedicated memory range (see Figure 5), i.e. each Rake finger has a base address pointing to the allocated Rake receiver. Each Rake finger has a second modulo offset pointer which is used to address the delayspread memory range allocated for one Rake receiver. The allocated memory size for one Rake receiver has the same length as the delay spread in symbols. This second solution is implemented in the reconfigurable WCDMA Rake receiver.

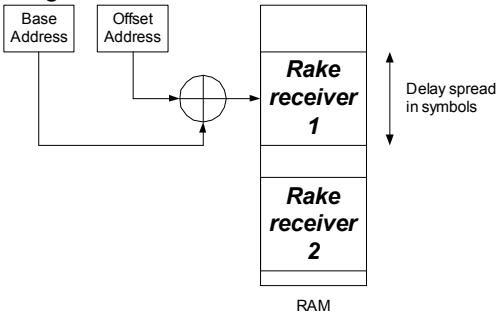



**Figure 5** Memory map

The fingers can be relocated dynamically to different Rake receivers by changing the base offset address pointer.

## 3.2 Exploiting parallelism

### 3.2.1 Problem

The limitation on exploiting the parallelism on a VLIW machine is given by dependencies. A dependency between two operations is a conflict that prevents the operations from executing concurrently. In the Rake application the dependency that dominates is the control dependency (see Figure 6). A control dependency occurs when an operation is guarded by the output of another operation. In the Rake application these conditional operations are frequently used. The problem of the conditional statements is the need to interact with the branch controller. Since in a single-threaded machine there is only one branch controller this causes a bottleneck.

```
Local_acc_finger1 += despread_chip;
if (local_SF_finger1 == 0 )
{
  symbol_finger1 = Local_acc_finger1;
  local_acc_finger1 = 0;
  local_sf_finger1 = SF;
  address = delay_finger1 + counter;
  mrc_symbol_finger1 = MRC[address];
  MRC[address] = mrc_symbol_finger1 +
                 symbol_finger1;
}
local_sf_finger1 -= 1;
```

Finger

MRC

**Figure 6** Example of conditional operation

Because of this dominant control dependencies, the conditional statements are executed in a sequential manner. This sequential behaviour forms the bottleneck in exploiting the parallelism in single-threaded DSPs. Figure 7 shows the Rake algorithm mapped on a single-threaded processor without exploiting the parallelism.

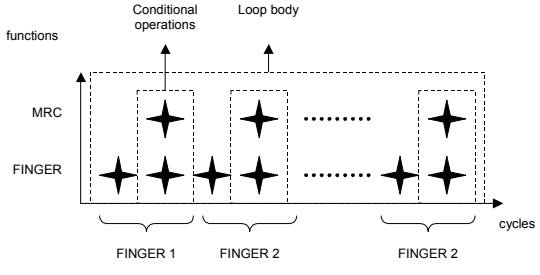



**Figure 7** Schedule Rake receiver on single-threaded DSP processor

In a multi processor implementation of the Rake algorithm, all Rake fingers and maximum ratio combiners can be modelled as independent processes. Each process has its own (branch) controller. The processors are mutually connected via a common interconnect. The limitation of this approach is the scalability of the design, since there is a resource common for all processors (the common interconnect). This resource dependency exists between two operations when they both need to use the same physical resource at the same time.

### 3.2.2 Solution

From previous chapters it can be concluded that:

- Parallelism exists in the baseband Rake processing algorithm
- Shared resources form a limitation on scalability
- Control dependency is highly dominant in the Rake application, causing the algorithm to be executed in a sequential manner on a single-threaded DSP.

```
if (finger1.loopcount == 0 )
{
  finger1.local_acc = 0;
  finger1.loopcount = SF;
}
if (finger2.loopcount == 0 )
{
  finger2.local_acc = 0;
  finger2.loopcount = SF;
}
finger1.local += input;
finger2.local += input;
--finger1.loopcount;
--finger2.loopcount;
```

**Figure 8** Multiple conditional operations

At the end of the correlation period, a jump operation takes place and a number of counters have to be reset, to their initial values (see Figure 8).This moment is not synchronized (when aligning at symbol level) for all correlation operations. The key idea is to place the condition in the data-path as a selection input to a multiplexer. These conditional operations can be mapped on a multiplexer functional unit (see Figure 9).

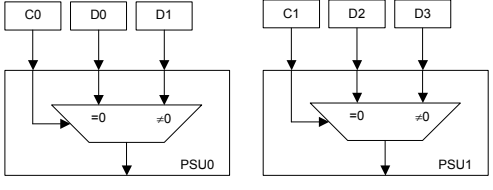



**Figure 9** Conditional operation in datapath

The variable finger1.loopcount is the selection input of the pass unit 0, placed in register C0. The finger1.local_acc value is placed in register D1 and the initiation value (0) is placed in register D0. The multiplexer selects register D0 when the condition is zero, i.e. at the end of the correlation interval (loopcount==0). When the loopcount is non zero, the value in register D1 is passed. The output of the pass unit is transferred to the ALU where the next operation is executed.

It is now possible to perform multiple conditional operations in parallel, because there is no interaction with a branch controller. In the architecture of the processor

there are multiple conditional pass units to support conditional statements in the data-path.

### 3.3 Programming

Making use of the massively parallel resources of the novel WCDMA Rake receiver processor requires powerful programming tools. The core is supported by programming tools that abstract the complex details of the architecture away from the programmer. Instead of having to perform low-level scheduling and resource allocation manually, programmers can therefore focus on optimising applications in C.

Novel scheduling techniques are able to analyse the constraints of the architecture (partial interconnect, extremely partitioned register files, multi input/output functional units) and the application (severe timing constraints) and use them to their advantage rather than be hampered by them [9, 10].

#### 3.3.1 Schedule

The WCDMA Rake algorithm mapped on the core executes one WCDMA finger and maximum ratio combining in two cycles, making use of software pipelining. This implies that 3 fingers, allocated to one Rake receiver, use 6 cycles (see Figure 10) as body of the loop.

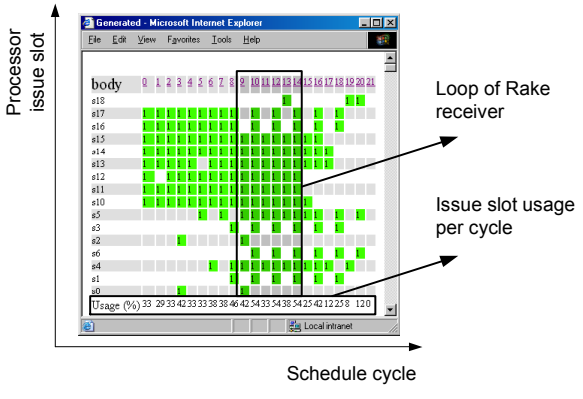



**Figure 10** Schedule of Rake receiver

The architecture can be conceptually extended to support multiple instances of the Rake receiver. Multiple Rake receivers can be scheduled in parallel and/or sequential (see Figure 11). The number of fingers per Rake receiver can be allocated dynamically.

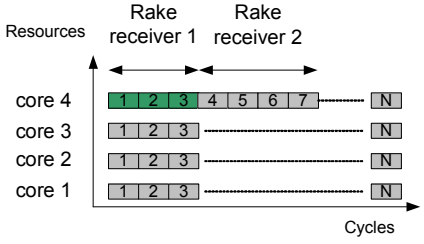



**Figure 11** Schedule of multiple Rake receivers sequentially and parallel

These parallel cores can all be placed under the control of one controller (see Figure 12). In this way multiple Rake receivers can be scheduled in parallel.

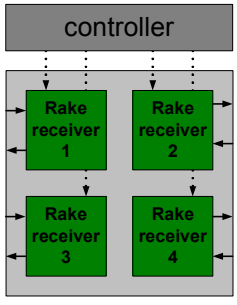


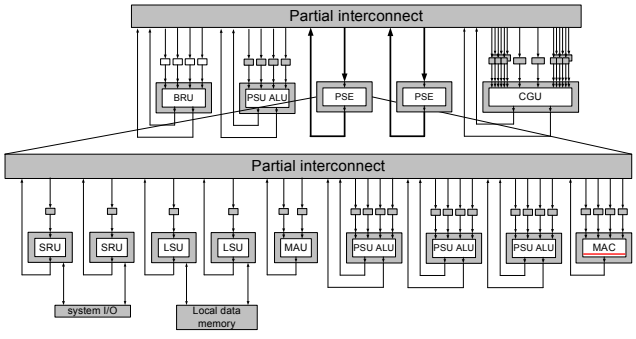

**Figure 12** Multiple parallel Rake receivers

### 3.4 Reconfigurable accelerator architecture template

The reconfigurable WCDMA Rake accelerator (see Figure 13) is a specific instantiation from a general architecture template underlying our processor generation methodology. The core contains 16bit ALUs, 16x16bit MAC, four send/receive units connected to system I/O, four 16 bit load/store units connected to local data memories, and a programmable code generation unit (CGU).

Our proprietary internal design methodology and tools enable automatic instantiation of the architecture template. Our highly abstract specification language TIM allows for the very quick design of different architectures. A typical TIM description describing a complete processor is only a few hundred lines long. Both the processor hardware and the associated programming tools are automatically generated from this same TIM description, enabling a huge productivity increase. In this way, a virtually unlimited variety of application-domain-specific cores can be created very fast, as dictated by market evolution. A big benefit of this approach is that instead of going for a catch-all solution that may be over-dimensioned and therefore too costly for the application at hand, one only pays for the flexibility required when powerful application domain tailoring is available. In a nutshell, our architecture

template is configurable at design time, i.e. before fabrication, whereas each generated core is reconfigurable after fabrication, and flexible enough to tackle all




application needs within its target domain.

**Figure 13** WCDMA Rake processor

## 4 Results

Synthesizable VHDL was generated for one reconfigurable Rake receiver. The core was synthesized using standard cells in 0.12 µm technology resulting in 0.2 $mm^2$ area and 6.64 mW power consumption @ 100 MHz using a wire load model.

## 5 Conclusions

A key enabler to truly programmable SoCs are reconfigurable accelerators that can replace currently-used ASIC designs with a comparable computational efficiency (MOPS/W) and low silicon overhead. We have presented a reconfigurable WCDMA RAKE processor that addresses these market needs. The presented reconfigurable WCDMA Rake accelerator is a massively parallel reconfigurable core supported by an innovative compiler. It is tailored to perform inner receiver WCDMA Rake kernels. We have presented results on the reconfigurable WCDMA Rake core and its supporting compiler that prove that C-programmable coarse-grained reconfigurable processors with a computational efficiency approaching that of ASICs at a modest area cost have finally become reality.

## References


[1] H. Holma and A. Toskala, WCDMA for UMTS, John Wiley & Sons, Ltd., New York, U.S.A., 2000.

[2] T. Ojanperä and R. Prasad, Wideband CDMA for third Generation Mobile Communications, Artech House, Boston, MA, U.S.A., 1998.

[3] M. Kuulusa and J. Nurmi, "Baseband implementation aspects for W-CDMA mobile terminals", in Proc.Baiona Workshop on Emerging Technologies in Telecommunications, Baiona, Spain, Sep. 1999, pp. 292-296.

[4] S.D. Lingwood, H. Kaufmann, and B. Haller, "ASIC implementation of direct-sequence spread spectrum RAKE-receiver", in Proc. IEEE Vehicular Technology Conference, Stockholm, Sweden, Jun. 1994, pp. 1326-1330.

[5] L. Harju, M. Kuulusa, and J. Nurmi, "A flexible Rake receiver architecture for WCDMA mobile terminals", in Proc. IEEE Workshop on Signal Processing Adavances in Wireless Communications, Taoyuan, Taiwan, Mar. 2001.

[6] M. Nilsson, "Efficient ASIC implementation of a WCDMA Rake receiver", Master thesis Luleå University of technology, Stockholm, Sweden, Apr. 2002.

[8] Silicon Hive, "Technology Primer", http://www.siliconhive.com

[9] A.H. Timmer, M.T.J. Strik, J.L. van Meerbergen and J.A.G. Jess, "Conflict modelling and instruction scheduling in code generation for in-house DSP cores", Proc. ACM/IEEE Design Automation Conference, pp. 593-598, 1995

[10] B. Mesman, "Constraint analysis for DSP code generation", Ph.D.thesis, Eindhoven University of Technology, The Netherlands, 2001