EURASIP Journal on Applied Signal Processing 2005:11, 1753–1768

Multichannel Baseband Processor for Wideband CDMA

Louay M. A. Jalloul
Infineon Technologies, 1730 North First Street, San Jose, CA 95112, USA
Email: lj07@aub.edu.ib

Jim Lin
Infineon Technologies, 1730 North First Street, San Jose, CA 95112, USA
Email: jlin@streamprocessors.com

Received 12 March 2004; Revised 1 December 2004

The system architecture of the cellular base station modem engine (CBME) is described. The CBME is a single-chip multichannel transceiver capable of processing and demodulating signals from multiple users simultaneously. It is optimized to process different classes of code-division multiple-access (CDMA) signals. The paper will show that through key functional system partitioning, tightly coupled small digital signal processing cores, and time-sliced reuse architecture, CBME is able to achieve a high degree of algorithmic flexibility while maintaining efficiency. The paper will also highlight the implementation and verification aspects of the CBME chip design. In this paper, wideband CDMA is used as an example to demonstrate the architecture concept.

Keywords and phrases: CDMA, chip-rate processing, symbol-rate processing, micro-DSP, multipath, RAKE.

1. INTRODUCTION

Code-division multiple access (CDMA) is the physical layer access method used in third-generation (3G) mobile radio systems [1]. 3G systems have improved receiver performance over any of the current cellular systems due to the introduction of several new physical layer techniques such as coherent demodulation [2] and turbo codes [3], just to name a few. Relative to the second-generation (2G) standards [4], 3G systems are also designed for both voice and high-speed data. The wideband CDMA 3G technology uses larger bandwidth and a higher chip-rate than the 2G CDMA counterpart [4]. Furthermore, 3G standards are evolving to support additional features such high-speed downlink packet access, enhanced uplink dedicated channel and multiple-input multiple-output antenna technology. 3G transceiver solutions must also have a rapidly decreasing cost per channel as a function of time. These features present a serious challenge in the efficient design and implementation of 3G transceiver solutions.

There are two typical approaches for development of baseband solutions to handle the above-mentioned design features. The first is a dedicated implementation, such as an application-specific integrated circuit (ASIC). The second implementation is a programmable architecture, such as a general purpose central processing unit (CPU) or field-programmable gate arrays (FPGA). ASIC solutions continue to be the technology of choice for high-volume production due to their efficiency, measured in terms of MOPS/mW, but they do not provide the required algorithmic flexibility. Fine-tuning the algorithms' performance delays the chip tapeout which typically results in a large cost. On the other hand, the FPGA approach offers a high degree of flexibility that makes them highly suitable for prototyping, but lacks the energy/power efficiency. FPGA implementations are also costly for designs that have a large number of channels. Therefore, there emerges the need for a paradigm shift in the approach for designing solutions that address these two fundamental constraints. The solution discussed in this paper occupies the energy efficiency and flexibility gap between dedicated hardware and CPU/FPGA.

There is a significant difference in the design and implementation of the base station and user equipment[1] (UE) receivers. The UE receiver, that is, for downlink reception, has to process a few channels from a single base station or a small number of base stations in the case of soft handoff. On the other hand, base station receivers, that is, for uplink reception (many-to-one), have to processes signals from a large number of users simultaneously. In this paper, we focus on uplink reception and describe the system architecture of the cellular base station modem engine (CBME), a multichannel receiver that is capable of processing and demodulating signals from multiple users simultaneously. A class of

[1] Also referred to as the handset or mobile station.

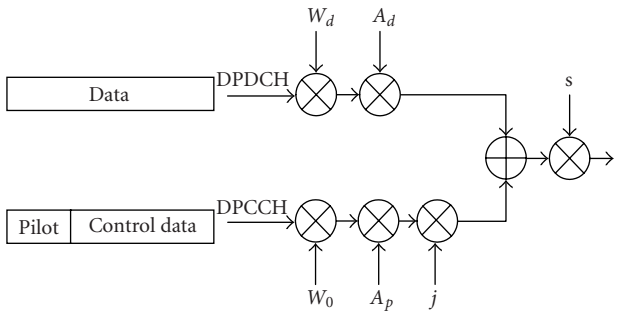


Figure 1: Spreading for uplink DPCCH and DPDCH.

flexible baseband digital signal processing architectures characterized by heterogeneous multiprocessors is presented. The architecture presented is this paper addresses the challenges described by arriving at a scalable and flexible solution.

This paper is organized as follows. Section 2 contains an overview of the CDMA system under consideration. The overall receiving system partitioning is described in Section 3. System resource dimensioning is addressed in Section 4. Section 5 contains a description of the software architecture. The chip verification and back-end design flow are described in Sections 6 and 7. Simulations and measurement results are shown in Section 8. Finally, some concluding remarks are given in Section 9.

2. CDMA SYSTEM OVERVIEW

The CDMA systems in [1] are based on direct-sequence spread-spectrum technology. In this paper, we focus on the wideband version, that is, on the 3.84 Mcps basic chip-rate which can be expanded to higher chip-rates for increased information bit rates. The uplink transmitted signal from the UE takes the form shown in Figure 1. There are two channels: (1) dedicated physical control channel (DPCCH) that carries layer 1 control data, and (2) dedicated physical data channel (DPDCH) that carries the encoded information. The DPCCH is spread to the chip-rate by a fixed spreading factor channelization code W_0, while the DPDCH is spread to the chip-rate by a variable spreading factor channelization code W_d. Only one code is used for DPCCH. For small data rates a single code used for DPDCH and for high data rates up to six parallel DPDCHs can be transmitted simultaneously. Each DPDCH can carry up to 384 Kbps of information data, thus when using all six code channels, a peak transmit data rate of 2 Mbps can be achieved.

The frame structure of DPCCH and DPDCH is shown in Figure 2 (also refer to [5]). The radio frame has a 10 milliseconds duration and is divided into 15 time slots. Each slot on the DPCCH contains 10 bits that are time-multiplexed among 4 subfields: pilot, transport format combination indicator (TFCI), feedback information (FBI) bits, and transmit power control (TPC). The DPCCH is spread using a code with spreading factor (SF) equal to 256. The pilot symbols are used by the inner receiver to do many of the demodulation functions, such as channel estimation for coherent detection, timing and frequency control, multipath searching, and so forth. The TFCI bits inform the receiver about the instantaneous parameters of each transport channel multiplexed on the physical channel. The TPC bits are used for controlling the downlink transmit power. Finally, the FBI bits are used for controlling the transmit diversity weights on the downlink, site-selection-diversity transmit, and enhanced physical downlink shared channel power control.

The DPDCH carries layer 2 dedicated data, where each slot contains $10 \times 256/\mathrm{SF}_{\mathrm{DPDCH}}$ symbols, $\mathrm{SF}_{\mathrm{DPDCH}}$ is the SF on the DPDCH. Knowledge of $\mathrm{SF}_{\mathrm{DPDCH}}$ is obtained once the receiver decodes the TFCI on the DPCCH.

3. BASEBAND TRANSCEIVER PARTITIONING

The transceiver processing partitioning is broken into two orthogonal layers: a physical layer processing and a control/software layer as shown in Figure 3. The software functions for uplink and downlink processing are available to the layer 1 programmer as a pool of resources that can be allocated and deallocated across users and antennas depending on the operating scenarios. The physical layer processing of the digital receiver can be derived from a canonical structure consisting of an inner receiver and an outer receiver. The inner receiver does the demodulation function to produce a soft symbol stream. The outer receiver uses as input the soft symbol stream produced by the inner receiver to finally produce estimates of the transmitted information bit stream.

Our focus in this paper will be on the inner receiver functions (or the modem). The inner receiver functions are also partitioned into two categories: (1) chip-rate processing and (2) symbol- (or slot-) rate processing. The chip-rate processing is described in terms of a data path and a control path, whereas the symbol- (or slot-) rate processing is described in terms of an estimation algorithm.

The data path is a hardwired circuit that has parameterizable inputs based on the control path. The estimation algorithms all reside within the microdigital signal processing (micro-DSP) units. Three types of micro-DSPs (see [6]) are used in the inner receiver: (1) finger micro-DSP, (2) combiner micro-DSP, and (3) timing micro-DSP.

3.1. Heterogeneous multiprocessing

The CBME design is based on a data-flow principle employing both traditional data path and programmable digital signal processor as shown in Figure 4. The entire data flow goes through several stages of these data paths and digital signal processor (DSP) combinations. The design is pipelined at these macrostages, denoted by P_rP_1, P_rP_2, DSP, and P_0P_1, so that throughput does not become an issue. Multiple streams of data can be processed by the device through the physical replication of these heterogeneous data-flow engines as well as time slicing the same data-flow engine.

At the heart of the CBME is a hardware-based scheduler that minimizes the amount of overhead in context-switching the different channel information in and out of the physical resources. It ensures that there is sufficient bandwidth to

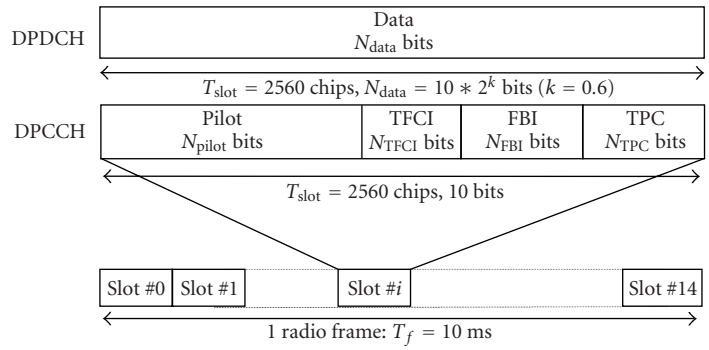


Figure 2: Frame structure for the uplink DPDCH/DPCCH.

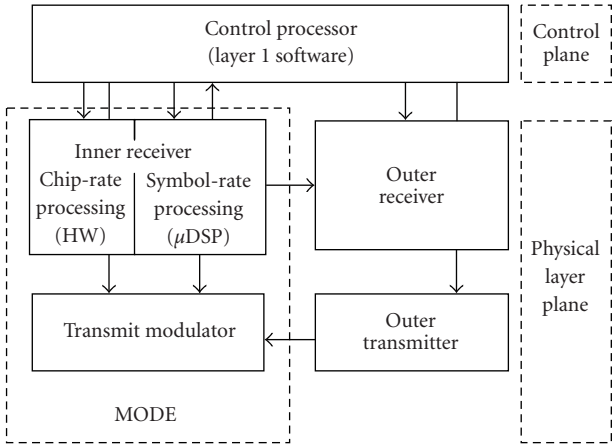


Figure 3: System partitioning.

carry out the context switching to meet the hard real-time requirements. It also minimizes the amount of intervention needed to simplify the software design.

Time slicing is done based on a predefined time interval. By running the chip at a multiple of the chip-rate, that is, 3.84 Mcps, a reuse factor can be derived based on the architecture to determine how many times a physical resource can be time sliced to serve different logical resources. The design was built with additional context and compute resources just in case that the time slicing can be increased by higher clock speed and better silicon technology. The CBME can even be speed-binned to support a different number of channels at different frequencies because of the time-slicing attribute.

Based on an initial chip area budget, chip frequency target, and estimates on sizes of key resources, a physical replicate factor is determined that would meet the dimensioning requirements done from the system analysis. Section 4 contains an indepth treatment of the dimensioning issues. In the cases of the digital signal processors, the number of cycles, instruction memory, and data memory had to be estimated for all the algorithms and control activities that needed to be carried out on them. Based on load assumptions for both cycles and memory usage, an optimal size was determined for the micro-DSP array.

The tight coupling of data path and digital signal processor is a key to minimizing the interface overhead that has plagued the traditional bus-based or DMA-based systems. By keeping the data flowing through the chain to completion on a predefined system time tick, CBME minimizes the amount of data thrashing among different resources prevalent in a DSP-centric design with hardware acceleration. CBME as a result is more power, area, and bus utilization efficient.

3.2. Noncausal processing

Before getting into the detailed description of the various inner receiver functions, buffering requirements for noncausal processing are discussed. Two approaches are discussed to evaluate the overall buffering requirement needed for noncausal processing such as TFCI decoding and channel estimation for coherent demodulation.

In the first approach, termed chip-rate buffering, an entire radio frame of 10 milliseconds is buffered. A dual finger, with dual-pass processing is used, where in the first pass the TFCI symbols are despread and decoded and then in the second pass, information about the TFCI is used to despread and demodulate the DPDCH data stream.

Using an input sample bit width of 8 bits per sample (I and Q) and two times oversampling of the input data, the buffer size is given by

$$\begin{aligned} B_{\text{chip buffer}} &= 8 \times 2 \times 2 \times 10 \times 10^{-3} \times 3.84 \times 10^{6} \\ &= 1.2288\ \text{Mbits}. \end{aligned} \quad (1)$$

Note that the buffer size shown in (1) is independent of the number of mobiles, the information bit rate or the number of fingers supported by the modem. Additional buffering is needed to store channel estimation filter states across the entire radio frame in order to aid in the noncausal channel estimation for the second pass. This additional buffering is given by

$$B_{\text{chip buffer control}} = N \times L \times (15 + D) \times 2 \times 8\ \text{bits}, \quad (2)$$

where N is the number of mobiles supported, L is the number of fingers per mobile, and D is the channel estimation filter delay. The total buffering requirement for the chip-rate buffering approach is therefore the sum of (1) and (2).

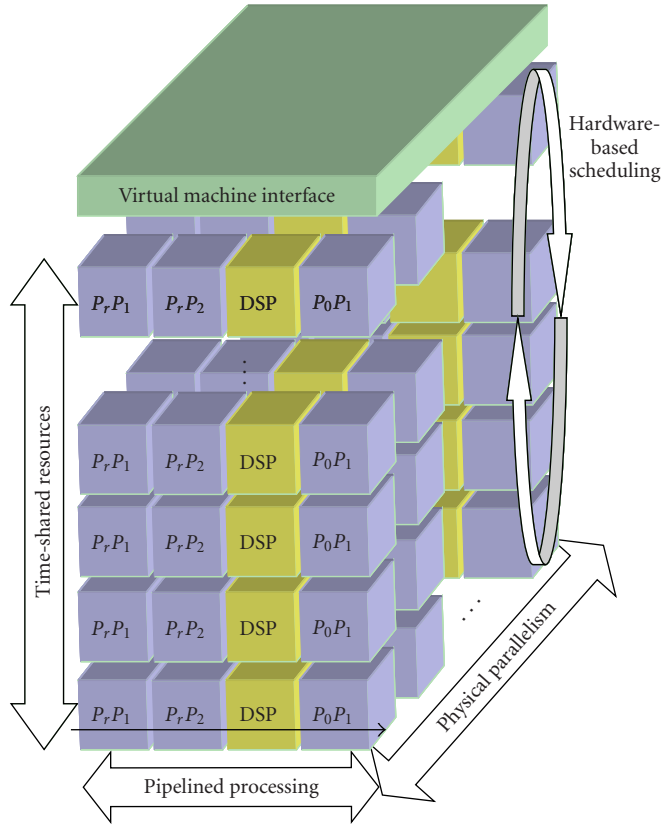


Figure 4: Heterogeneous multiprocessing.

The second approach, termed symbol-rate buffering, uses partial despreading of the received signal. The lowest spreading factor has to be assumed for the DPDCH stream which is 4 chips per symbol. The channel estimates are obtained from the DPCCH and applied noncausally to the DPDCH data stream. The buffering requirement in this case is given by

$$B_{\text{symbol buffer}} = N \times \frac{3.84 \times 10^6}{4}(L \times D \times 2 \times 8 + 12)\ \text{bits}. \tag{3}$$

Note that in (3) the bit width resolution of the partial despread symbols is assumed to be 12 bits. This larger bit width is due to the channel estimate conjugate being multiplied by the DPDCH stream and the combining across L fingers. Additional buffering is required for channel estimation which is given by

$$B_{\text{symbol buffer control}} = N \times L \times B \times 2 \times 8\ \text{bits}, \tag{4}$$

where B is the span of the channel estimation filter in terms of the number of slots. Note that 8 bits of resolution are assumed for the channel estimation filter states. The total buffering requirement for the symbol buffering approach is the sum of (3) and (4).

Remarks

(1) The memory requirement of the chip-rate buffering approach is largely independent of the number of mobiles/users supported by the modem.

(2) Chip-rate buffering allows the architecture to easily scale with the data rate without reducing the number of simultaneous channels supported.

(3) The symbol-rate buffering approach is highly dependent on the number of mobiles and fingers supported.

(4) For fixed allocated memory, using the symbol-rate buffering approach requires the number of mobiles supported to decrease as the per-user information bit rate increases.

(5) The chip-rate buffering and symbol-rate buffering approaches result in the same memory requirements, when the number of simultaneously supported mobiles is 4.

(6) The symbol-rate buffering approach requires 10 times more memory than the chip-rate buffering approach when the number of simultaneously supported mobiles is 36.

(7) The CBME uses the chip-rate buffering approach.

3.3. Multipath searcher

In a dense multipath environment, the channel is modeled, in baseband, as a linear filter with an impulse response for

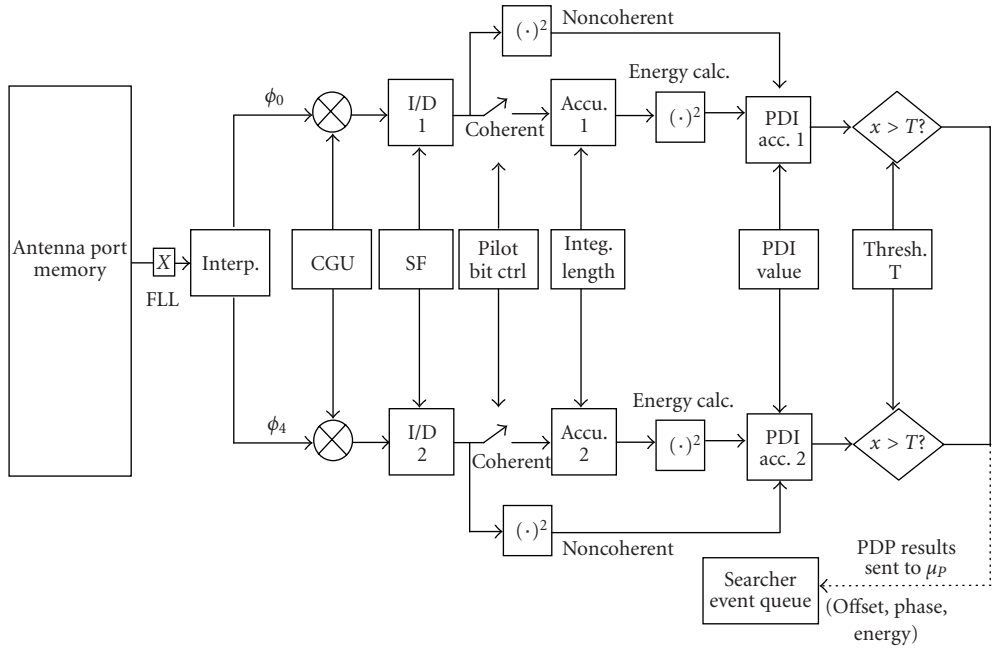


Figure 5: Searcher processing.

the kth user given by

$$h_k(i) = \sum_{l=1}^{L_k} \alpha_{k,l}\delta(i - \tau_{k,l}), \tag{5}$$

where i is the sample index, $\alpha_{k,l}$ and $\tau_{k,l}$ are the path gains and delays, respectively, and L_k is the number of delay paths.[2] The path gains are random variables that typically follow a Rayleigh or Rician distribution. The received baseband signal is then given by

$$r(i) = \sum_{l=1}^{L} \alpha_l s(i - \tau_l) + n(i), \tag{6}$$

where $s(i) = \sum_{k=1}^{K} s_k(i)$ is the total transmitted signal from K users (that includes their modulation symbols multiplied by the spreading, channelization codes, and chip pulse shaping filter) and $n(i)$ is the thermal noise.

The maximum likelihood estimate of the delay parameters for the kth user is computed as

$$\hat{\tau}_{k,l} = \arg\max_{\tau_{k,l}} \frac{1}{M} \sum_{m=1}^{M} \left| \frac{1}{N} \sum_{n=(m-1)N+1}^{mN} r(n) s_k^*(n - \tau_{k,l}) \right|^2. \tag{7}$$

[2]The number of delay paths seen by the receiver is a function of the environment and is also impacted by the ratio of the spreading signal bandwidth to the channel delay spread.

As can be seen from (7), the searcher processing is a one-dimensional search over a finite grid of delay values. Also note that searcher processing in (7) is broken into two operations; first coherent accumulation (over N symbols) is carried out and second is the noncoherent accumulation (over M coherent averages) as shown in Figure 5. The normalizations in (7) are needed for the finger management and assignment.

Note that the outputs of two searchers from the diversity antennas may be combined before threshold comparison is performed. Combining is done as long as the delay due to antenna distance separation is much smaller than the chip duration. This diversity combining improves searcher receiver operating characteristics.

3.4. Tracker RAKE finger

The RAKE receiver architecture enables combining of the received multipath signal components, through a bank of matched filters, which is implemented as a set of correlators each locked to a multipath component. The output of each correlator is given by

$$Z_{k,l}(\hat{\tau}_{k,l}) = \sum_{i=1}^{\mathrm{SF}} r(i) e^{-j2\pi \hat{f}_k i T_c} s_k^*(i - \hat{\tau}_{k,l}) \hat{c}_{k,l}^*(i), \tag{8}$$

where $\hat{f}_k$ is the frequency offset estimate, $\hat{c}_{k,l}$ is the channel estimate, $\hat{\tau}_{k,l}$ is the estimate of the path delay, and SF = $\mathrm{SF}_{\mathrm{DPCCH}}$ or $\mathrm{SF}_{\mathrm{DPCCH}}$. Note that the frequency offset and channel estimates are kept constant over 256 chips. A block diagram of a RAKE finger is shown in Figure 6. It can be seen from the figure that two despread processing paths are carried out on the interpolated received signal. The first on-time

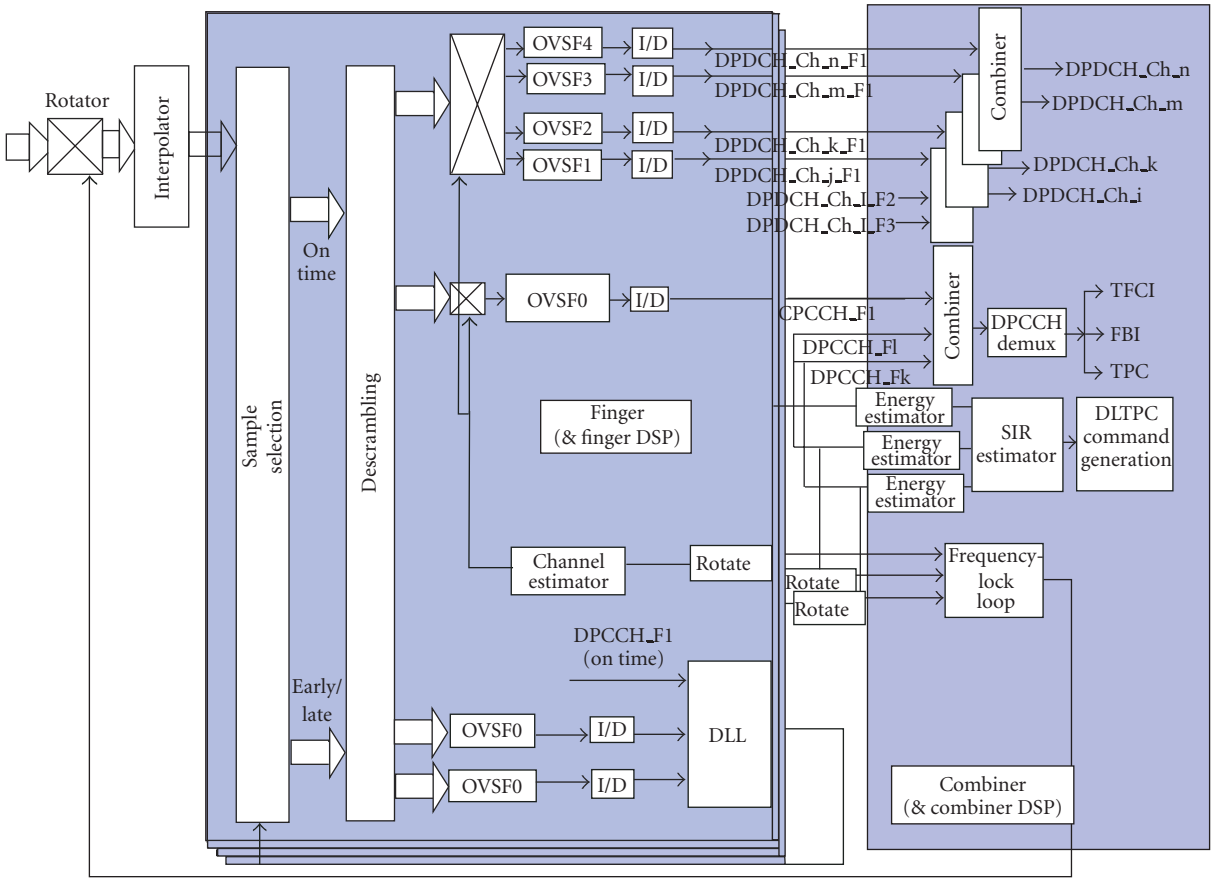


Figure 6: RAKE receiver structure.

despreading path operates on the DPCCH stream to extract pilot, TFCI, FBI, and TPC symbols. The second on-time despreading path, delayed by 1 radio frame (10 milliseconds), operates on the DPDCH data stream. When processing the DPDCH in the second despreading path, frequency correction and channel estimation have already been applied using hardware multipliers to reduce the burden on the micro-DSP.

3.5. Microdigital signal processor architecture

The micro-DSP is a tightly coupled fully programmable DSP that is instantiated multiple times as part of the signal processing data flow in the CBME design. There are two implementations of the micro-DSP within the CBME: one operates at the finger level and the other at the combined pilot level for various parameter estimations. The two micro-DSP implementations differ in terms of internal memory sizes, I/O register usage, and context-switching mechanisms. Both micro-DSPs share the same architecture and instruction set. Therefore, the description in this section is common to both micro-DSPs. The specifics about the finger micro-DSP and the combiner micro-DSP software are explained in more details in later sections.

The micro-DSP is a very large instruction word (VLIW) digital signal processor. It contains

(i) four 16-bit general-purpose registers (x0, x1, y0, y1);

(ii) two 40-bit accumulators (d0, d1);

(iii) a single STATUS register;

(iv) a single 40-bit MAC unit;

(v) a single 24-bit arithmetic unit;

(vi) a single 16-bit logical unit;

(vii) a single 40-bit barrel shifter unit;

(viii) two load/store units, one for each embedded data RAM (M0, M1). Each load/store unit contains four autoincrementing memory pointers used to address memory. For the M0 memory, these pointers are named mp0a, mp0b, mp0c, and mp0d. For the M1 memory, these pointers are named mp1a, mp1b, mp1c, and mp1d;

(ix) an array of I/O registers used to move data into and out of the processor (rw*, ro*) at every context switch boundary. These registers embed the micro-DSP into CBME's data-flow architecture.

The 33-bit VLIW instruction word can encode either a single 33-bit wide instruction, or three smaller "instruction slots" that execute in parallel. These three instruction slots consist of

(i) a 12-bit EXECUTE slot, which can contain arithmetic operations or flow-control instructions,

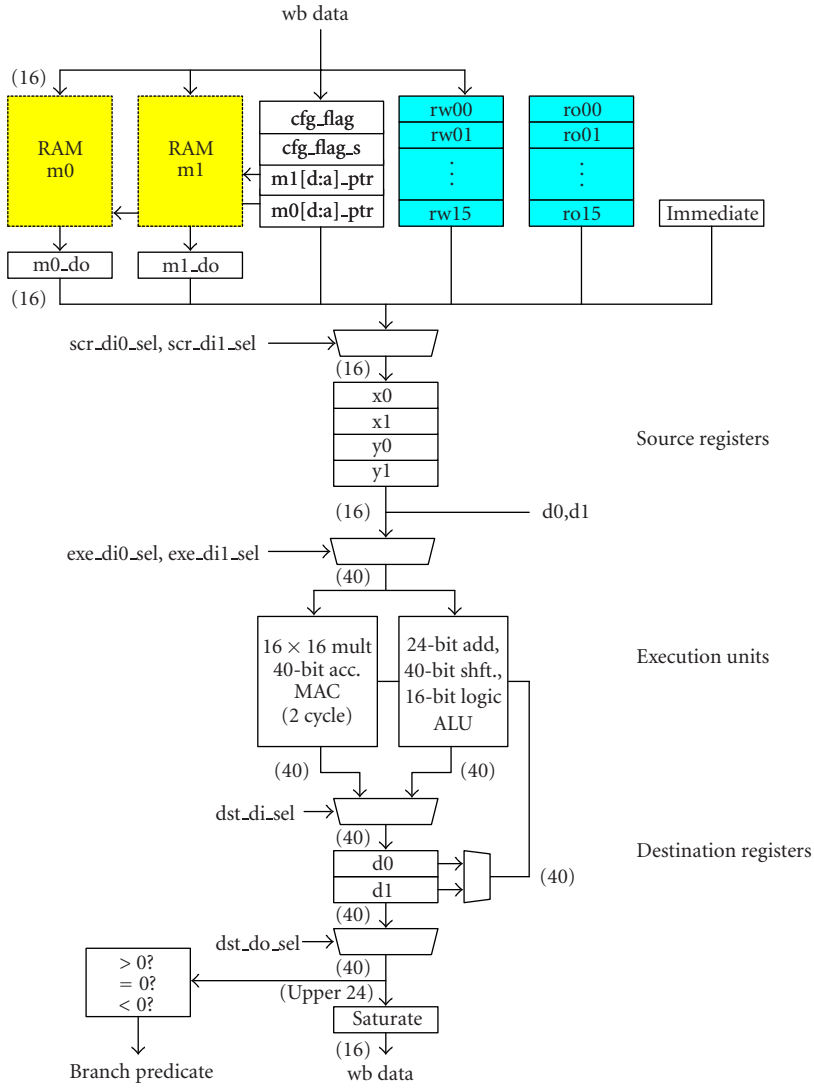


Figure 7: μDSP core architecture.

(ii) a 10-bit LOAD slot, which can only contain a LOAD instruction,

(iii) an 11-bit LOAD/STORE slot, which can contain either a STORE instruction or a second LOAD instruction.

Alternately, some instructions consume all 33 bits of the instruction word in their encoding. These instructions are called MACRO instructions, and cannot be executed in parallel with any other instruction.

The micro-DSP is a pipelined design. There are three pipeline stages and no operations for pipeline flushing or stalling. All operations execute in a single cycle except loads from memory, loads from memory with pointer auto-increment or decrement, multiply, and MAC. MAC and multiply take two cycles to complete, but they are pipelined and thus are nonblocking. Loads from, and stores to, memory require three cycles to complete. Therefore, two instructions are always fetched and executed following any branch instruction (including conditional branch, unconditional branch, and end-of-loop instruction). The micro-DSP core is shown in Figure 7 and an overview of the micro-DSP architecture is shown in Figure 8.

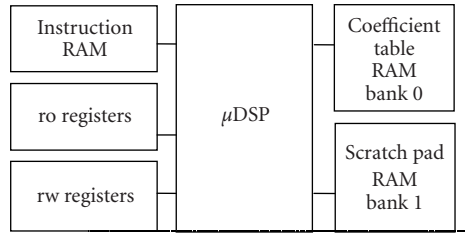


Figure 8: μDSP architecture overview.

Table 1: Micro-DSP capacity.

Micro-DSP	Cycles/256 chips	Instruction memory (kB)	Data memory (kB)
Finger micro-DSP	512	2	0.192
Combiner micro-DSP	4000	8	2.4
Timing micro-DSP	512	2	2

Table 2: Finger micro-DSP cycle count.

Function	Cycles/256 chips
DLL	58
Channel estimation	59
Messaging	20

3.6. Finger micro-DSP

Two main estimation functions are programmed on the finger micro-DSP:

(i) channel estimation,
(ii) timing correction using a delay-locked loop (DLL).

The despread DPCCH symbols are inputs to the finger micro-DSP. The symbol type (pilot, TFCI, TPC, FBI) is also fed to the finger micro-DSP. For channel estimation, only pilot symbols are used. Depiloting is achieved my multiplying the received despread symbols by the pilot bit, producing an unmodulated symbol stream used for channel estimation. The finger micro-DSP uses the unmodulated symbols to form slot averages which are then input to an FIR channel estimation filter. This is called weighted multislot averaging (WMSA) filter. Consecutive outputs of the WMSA filter are interpolated to produce a channel estimate at the DPCCH symbol-rate, that is, once every 256 chips. The channel estimate is fed back and applied at the chip-rate, as shown in Figure 6, to derotate the received DPDCH stream. Applying the channel estimate at the chip-rate rather than the symbol-rate allows the use of a single multiplier for all the multicodes and the receiver to carry only real-valued signals which is beneficial in the case of multicode channels.

The timing correction is done using a delay-locked loop based on the unmodulated received pilot symbol stream. The magnitude difference between early and late energies reflects the time/phase between reference timing and locally generated timing signals. To reduce the noise sensitivity of the estimator the magnitudes are passed through a lowpass filter. Due to the interpolation filter, the timing error measurement has a resolution of $T_c/8$.

The finger micro-DSP characteristics, that is, available cycles, instruction, and data memory are shown in Table 1. Note that there are 8 finger micro-DSPs, each dedicated to a finger. Table 2 shows the cycle count of the processes running on each finger micro-DSP.

3.7. Combiner micro-DSP

There are several estimation functions programmed in the combiner micro-DSP:

(i) frequency correction using frequency-locked loop,
(ii) power balancing,

Table 3: Combiner micro-DSP cycle count.

Function	Cycles/256 chips
FLL	350
Finger energy processing	250
SIR	350
TFCI assist	360
Power control	200
FBI decoding	60
Parameter messaging	400
Miscellaneous messaging	210

(iii) closed-loop power control,
(iv) TFCI decoding,
(v) closed-loop transmit diversity,
(vi) signal-to-interference ratio (SIR) estimation:
  (1) received signal code power measurement (RSCP),
  (2) interference signal code power measurement (ISCP),
(vii) power delay profile measurement and energy lock detection (used in combiner mask setting and finger management and assignment),
(viii) pilot bit error rate measurement (PBER),
(ix) transmit code power measurement,
(x) parameter and miscellaneous messaging.

Only a few of the above-mentioned algorithms are described in this section.

Frequency correction is done using a frequency-locked loop. The frequency offset is estimated based on the cross-product over two consecutive pilot symbols. To improve the accuracy of the estimate, the real and imaginary parts of the cross-product are averaged over all the pilots symbols and fingers in a slot. The resulting average phase shift is averaged over one frame before the corresponding frequency shift is determined through a lookup table. The frequency shift is applied to a number controlled oscillator after passing it through the control filter.

The SIR estimate is the ratio of two estimates: RSCP and ISCP. The RSCP estimate is done over the received unmodulated pilot symbols using noncoherent processing as shown in [5]. The ISCP estimate is made by averaging differences of consecutive received unmodulated pilot symbols, also shown in [5].

The combiner micro-DSP characteristics, that is, available cycles, instruction, and data memory are shown in Table 1. The cycle count of the algorithms and other processes running on the combiner micro-DSP is shown in Table 3. This cycle count represents the worst-case scenario

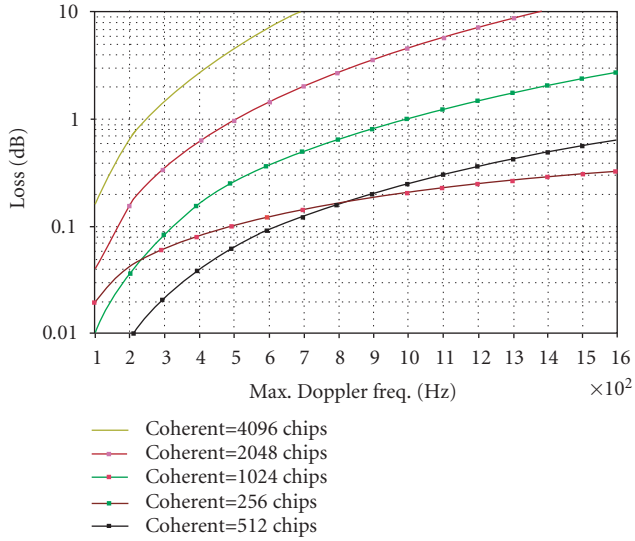


Figure 9: SNR loss due to coherent averaging.

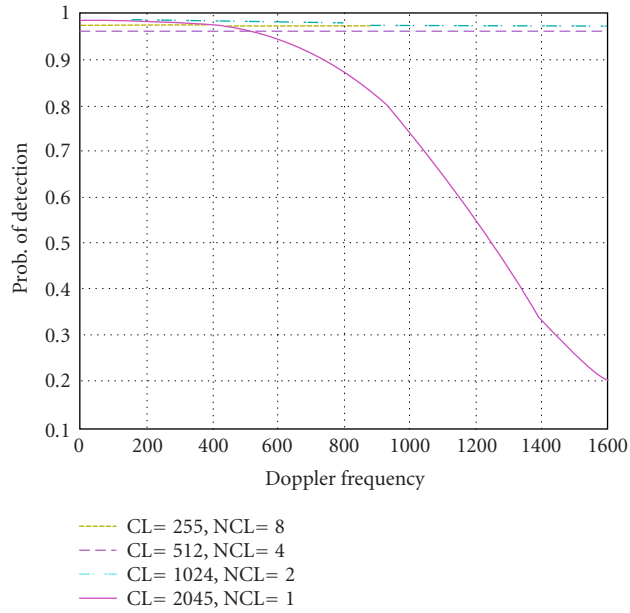


Figure 10: Coherent and noncoherent integration (PDE) (detection probability versus Doppler and MF spans (up to 2048 chips), false alarm probability = 0.2).

and is not always in the critical path. By profiling all the processing running on the combiner micro-DSP, the critical path was found to be 3280 cycles.

3.8. Timing micro-DSP

Unlike the finger and combiner micro-DSPs that are part of the data-flow pipeline, the timing micro-DSP falls in the control scheduler's pipeline. The timing micro-DSP is responsible for tasks like keeping track of slot format, symbol timing, and code generation state control, on a per-channel basis. It is in constant communication with the data-flow pipe. It affords the channels running on the CBME an additional level of programmability. Messaging mechanisms are available from the host processor and to the other micro-DSPs on the CBME.

The timing micro-DSP characteristics, that is, available cycles, instruction, and data memory are shown in Table 1.

3.9. Combiner data processor

The combiner data processor (CDP) sums the DPDCH outputs from the multiple fingers and forms the soft symbols as inputs to the outer receiver. The CDP output is given by

$$Z_k = \sum_{l=1}^{L_k} Z_{k,l}. \tag{9}$$

The CDP also contains a multiplier needed to ensure the proper bit width of the soft symbols. Note that the processing in the CDP is data rate and spreading factor independent and is therefore more efficient to do in hardware in order to reduce the burden on the combiner micro-DSP.

3.10. Reed-Muller decoder

The TFCI bits are encoded using a Reed-Muller (RM) code and are spread across an entire radio frame. The RM decoder is a hardware coprocessor used to aid in the decoding of the TFCI. A hardware coprocessor is used for RM decoding since it is the most computationally intensive portion of the TFCI decoding process. A slight increase in hardware (gate count) is paid to reduce the complexity of the combiner micro-DSP. A fast Walsh-Hadamard transform (FHT) of size 32 is the core used for decoding the RM code.

3.11. Preamble detection engine

The random-access channel (RACH) is an uplink channel used to carry control information, and also short packets, from the UE to the Node B. The RACH transmission is based on slotted-ALOHA approach with fast acquisition indication. The random-access transmission consists of one or several preambles of length 4096 chips followed by a message of length 10 milliseconds.

The preamble detection engine (PDE) is a 4096-tap matched filter used to detect the transmitted preamble. The maximum frequency offset during RACH can be as large as ±1500 Hz due to oscillator mismatch between the transmitting UE and the receiving Node B as well as the effect of twice the maximum Doppler frequency. Coherent integrating over the entire preamble length results in a loss in the signal-to-noise ratio and thus a degradation in the PDE performance as shown in Figure 9. In order to mitigate the effect of the initial large frequency offset, a discrete frequency maximum likelihood approach is used in the PDE. Four discrete frequencies are used in the PDE that can be programmed through the layer 1 control processor based on the initial frequency offset. This coarse frequency offset correction scheme reduces the overall frequency deviation and the best out of the four frequencies is used to initialize the FLL for message demodulation. Dividing the length of integration into coherent and noncoherent parts also reduces the effect of initial frequency offset as shown in Figure 10.

Table 4: 3GPP multipath fading channel models.

Case 1 (km/h)		Case 2 (km/h)		Case 3 (120 km/h)	
Relative delay (ns)	Average power (dB)	Relative delay (ns)	Average power (dB)	Relative delay (ns)	Average power (dB)
0	0	0	0	0	0
976	−10	976	0	260	−3
—	—	20 000	0	521	−6
—	—	—	—	781	−9

Table 5: Multipath Case 1 channel finger placement and FRP.

Finger (units of T_c)	Finger (dB)	FRP (%)
0	−0.414	91.0
3.8	−10.44	99.94

Table 6: Multipath Case 2 channel finger placement and FRP.

Finger (units of T_c)	Finger (dB)	FRP (%)
0	−4.77	33.34
3.7	−4.78	66.57
77	−4.76	99.99

4. SYSTEM RESOURCE DIMENSIONING

A key to an efficient modem design requires proper dimensioning of the resources. Two types of analysis are shown here, one is based on the worse case assumptions and the other is a mean-value analysis that uses a statistical model of the system behavior.

4.1. Deterministic analysis

The analysis below examines the issue of the number of fingers needed to ensure adequate radio link performance. This analysis is based on computing the power delay profile of each of the 3GPP channel models [7]. Based on the power delay profile, RAKE fingers may be placed at subchip offsets (this is essentially what the multipath searcher and finger management would perform).

The following two rules are used in this analysis.

(1) A multipath that is within 10 dB from the peak ray is assigned a finger.

(2) Based on the number of fingers assigned, the fraction of recovered power (FRP) is computed.

The C/I at the receiver is given by

$$\frac{C}{I} = \frac{\alpha P}{(1-\alpha)P + N}, \tag{10}$$

where α is the FRP, P is the transmitted power, and N is the noise power (including thermal, multiple-access interference or other types of interference). The C/I may be rewritten as

$$\frac{C}{I} = \frac{\alpha(P/N)}{(1-\alpha)(P/N) + 1}. \tag{11}$$

In the limit as $P/N \to \infty$, then the C/I becomes

$$\lim_{P/N\to\infty} \left(\frac{C}{I}\right) = \frac{\alpha}{(1-\alpha)}. \tag{12}$$

Table 7: Multipath Case 3 channel finger placement and FRP.

Finger (units of T_c)	Finger (dB)	FRP (%)
0	−2.74	53.25
1	−5.74	79.94
2	−8.73	93.35
3	−11.72	99.99

Table 8: Multipath COST 259 channel finger placement and FRP.

Finger (units of T_c)	Finger (dB)	FRP (%)
0.1	−5.52	28.09
1	−7.88	44.37
2.1	−6.23	68.19
3	−8.37	82.75
5	−10.89	90.89
5.9	−14.12	94.77

Setting a maximum of 13 dB for C/I as a requirement for demodulation, then solving for the FRP, we get that

$$\alpha = \frac{10^{1.3}}{10^{1.3} + 1} = 0.95. \tag{13}$$

This design metric will be used to determine the number of fingers that need to be assigned for a given power delay profile.

A summary of the channel models defined in 3GPP [7] is given in Table 4. For multipath Case 1, 2, and 3 channel models, the finger placement and the associated fraction of recovered power are shown in Tables 5, 6, and 7, respectively. For multipath Case 1 channel, Table 5 shows that 2 fingers can be assigned for the channel and be able to recover almost 100% of the power/energy. Table 6 shows that for multipath Case 2 channel, RAKE receiver can recover almost 100% of the power/energy using 3 fingers. For multipath Case 3 channel, Table 7 shows that 3 fingers can be assigned and the RAKE receiver is able to recover almost 94% of the power/energy. Note that with 4 fingers the RAKE receiver can recover almost 100% of the power/energy. However, the last finger will be weak, which implies that the channel estimation, FLL, and DLL errors will be large and therefore will negatively impact the demodulation process.

For the COST 259 TUx channel model [8], the finger placement and the associated fraction of recovered power are shown in Table 8 which shows that 6 fingers can be assigned for the channel and be able to recover almost 95% of the

Table 9: CDMA system parameters.

Parameter	Notation	Value
Chip rate	R_c	3.84 Mcps
Information bit rate	R_b	12.2 Kbps
Information bit energy-to-interference-plus-noise ratio	$E_b/(I_0 + N_0)$	4 dB for low-speed mobiles 7 dB for high-speed mobiles
Reuse factor	$1 + f$	1.5
Voice activity factor	v_a	0.5
Number of sectors per Node B	s	3
Power control error	g	0.5
Loading	Z_L	0.7 (6 dB rise)
Soft handoff factor	h	1.5
Blocking probability	B_p	2%

Table 10: Number of resources.

Case	Number of mobiles	Number of fingers
$E_b/(I_o + N_o) = 4$ dB (low-speed mobile)	263	1052
$E_b/(I_o + N_o) = 7$ dB (high-speed mobile)	132	1056
Total number of fingers (50% low-speed mobiles and 50% high-speed mobiles)	198	1054

power/energy. However, 5 fingers will be better since the last finger will be weak. In that case, the FRP will only be 91%.

Based on the PDP results shown in the previous sections, it can be seen that in most channels 2 fingers are needed. For the COST 259 channel model, up to 5 fingers may be required.

Since the uplink typically uses 2-fold antenna diversity, then more fingers need to be provided but the number of fingers need not be doubled. Since with maximal ratio combining, the receiver enjoys a large diversity benefit and therefore weak fingers may be discarded and only the strongest rays are captured, then the total number of fingers needed per mobile is around 8.

4.2. Statistical analysis

A second type of dimensioning analysis is based on a statistical analysis of the CDMA system and the propagation channel. In this architecture, the receiver supports the entire cell capacity. The receiver maintains a pool of signal processing resources, that is, fingers and searchers, that are flexibly allocated to different mobile stations across different sectors. The allocation is based on various criteria such as cell loading, propagation multipath channel, required quality of service, and so forth. Note that the deterministic or static allocation of resources is independent of the system and channel conditions and resources are allocated equally across users. The flexible allocation of fingers and searcher results in a trunking efficiency and is therefore more efficient than the static or deterministic allocation of resources.

The cell capacity is given by [9]

$$C = \frac{R_c/R_b}{E_b/(I_o + N_o)} \times \frac{1}{1+f} \times \frac{1}{v_a} \times Z_L \times g \times s, \tag{14}$$

where

(i) R_c: chip-rate,
(ii) R_b: information bit rate,
(iii) $E_b/(I_o + N_o)$: information bit energy-to-interference-plus noise ratio,
(iv) $1 + f$: reuse factor (other-cell-to-incell-interference ratio),
(v) v_a: voice/data activity factor,
(vi) s: number of sectors per Node B,
(vii) g: excess power due to closed-loop power control error,
(viii) Z_L: effect of loading which depends on the $(I_o + N_o)/N_o$ ratio often called rise over thermal noise [10].

Due to soft handoff, the overall number of mobiles that need to be supported is given by

$$C_s = h \times C, \tag{15}$$

where h is the soft handoff factor.

Using Erlang-B [9], the blocking probability is given by

$$B_p = \frac{C_s^K/K!}{\sum_{j=0}^{K} (C_s^j/j!)}, \tag{16}$$

where K is the number of combiners available at the receiver.

Using the parameters in Table 9, and assuming no blocking due to lack of resources, then the number of mobiles (or combiners) that need to be supported is shown in Table 10.

Next assume that there is a mixture of low and high-speed mobiles in each cell. Further, assume that there are an equal number of low-speed mobiles as there are high-speed

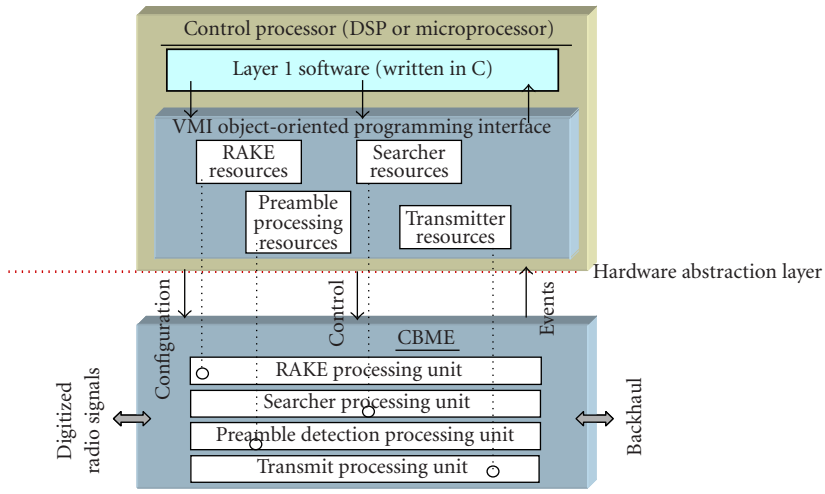


Figure 11: Software model.

mobiles. The number of fingers allocated for the low- and high-speed mobiles are 4 and 8 (including diversity antenna support), respectively. Then the total number of fingers that need to be supported is shown in Table 10.

5. SOFTWARE ARCHITECTURE

The layer 1 control software runs on the host microprocessor on the same channel card as the CBME as shown in Figure 3. CBME offers an abstracted software interface, called the virtual machine interface (VMI). The VMI is used to program the processing on the CBME from the host microprocessor. Figure 11 shows the software programming model of the CBME.

The VMI has chip-specific knowledge regarding scheduling and resource allocation. It keeps track of the resource usage so that users can get accurate feedback on the runtime status of the CBME. It drives the CBME with chip-specific commands and definition and makes the user interface implementation independent. Each resource type is made into an object whose properties and functions can be programmed onto the CBME to achieve the desired effects.

The host microprocessor has the ability to send "timed" control messages to the CBME so that specific parameters or control information can take effect on well-timed boundaries, like slot or frame boundaries. To ease the programming of these messages, the CBME allows slack in the timing of these messages by providing buffering and message processing capabilities. Internal buffering of the results allows for time-based, for example, 10 milliseconds, process to be kicked off in a deterministic fashion on the host microprocessor.

The VMI also includes an interrupt-driven mechanism to signal to the host when results, like preamble detection and searcher results, are ready to be shipped back. Both commands and results are written and read via a memory-mapped FIFO-based messaging scheme. Because CBME runs the real-time scheduling of the tasks in hardware, there is very little timing control overhead from the host microprocessor.

The micro-DSP code running on the CBME can be changed by the user to optimize the algorithms, add measurements, and collect statistics after the CBME has been taped out. This flexibility decouples the firmware development from the chip development.

Debug software was also developed to ease software development by placing the CBME device in step mode. Step mode in CBME is defined as executing the chip in real time a certain number of cycles and then stopping. Because much of the software is symbol-, slot-, or frame-based, it becomes advantageous to be able to stop the software on these boundaries. Once the chip is stopped, internal states can be scanned out for debugging purposes. The chip can then be resumed. The same mechanism used for step mode can also be triggered through micro-DSP breakpoints that are user programmable. This allows the micro-DSP code running on the CBME to be easily debugged.

The VMI is also in charge of configuring the CBME upon reset using the scan chain. CBME has a large number of reset-time programmable states that can be specified by when the CBME is shipped to customers. A subset of these states can then be further changed by customers in the laboratory or field. CBME uses a similar mechanism to introduce firmware download onto the CBME.

6. VERIFICATION PROCESS

The system verification relies on a range of test benches and models with emphasis on different aspects of the verification problem.

The UE network (UEN) is a software model of the WCDMA UE transmitter that also includes the channel and the receive chip matched filter as shown in Figure 12. This model is used to generate input antenna data for the CBME. It can be scripted to generate multiple mobiles with different radio and channel parameters to simulate a base station

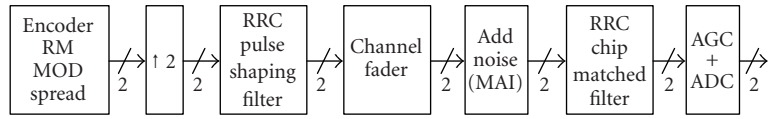


Figure 12: UE network.

environment. The ease of control and tight integration with the verification environment makes the UEN much more suited over an emulator box.

The data-flow simulation kernel (DFSK) is a complete C-based bit-exact model of the CBME that can be used to compute expected results. This model was originally developed to evaluate system performance and later adapted so that it can be used in the chip verification environment. The existence of this golden model is the foundation to any data comparison done during verification of the CBME.

Random verification techniques were used to create scenarios that were not conceived by the test writer or could not be covered exhaustively by directed testing. Directed random vectors were applied to the device under test (DUT) using a set of constraints that are controlled by the verification engineer. Expected results are computed by the DFSK model so that checking can be done dynamically. The random test bench also has the ability to monitor reactions from the DUT and change the random input vectors accordingly. Such random simulations are started and can run for an arbitrary amount of time until first miscompare is found.

The verification is carried out in two phases, one covering block-level functionality and the other full-chip functionality. At the block level, data-path testing is carried out by subjecting the DUT with input data that stresses computation correctness, including various corner cases like saturation and rounding. Data-path testing, while having variety in data inputs, is limited in the amount of control logic that gets exercised. Control tests also tend to be more labor intensive to create and maintain. Random control testing using techniques mentioned earlier along with models of the datapath, shown in Figure 13b, is done with gain coverage and confidence on control logic.

Both directed and random were created to exercise the CBME at the full chip level. A list of real scenarios, complete with pseudofirmware driver, was run on the CBME. These scenarios started out being completely manually conceived and written. As verification progressed, these scenarios were generated using a similar constrained random approach as those described for the subsystem level.

All tests are used to gauge verification coverage. There are several sources of coverage criteria. From a directed test point of view, a plan describing all directed functional tests, both at the block and chip level, was used to track progress of the direct verification effort. Functional coverage code in the random test benches was also used to gauge coverage from random tests. Lastly, a coverage tool was used to perform code coverage for statement and a subset of toggle and branch.

7. CHIP DESIGN

The chip was designed with a COT (customer own tooling) flow. All of the methodology and tool issues were done in-house using a combination of point tool from industry and custom written software needed for the chip's unique challenges. The chip design and design methodology were carried out in close relationship with the system and architecture design. Logical and physical partitioning of the design was done early on to make sure that the chip is implementable with predictable schedule and performance. A hierarchical design flow was also used to take advantage of block partitioning. Consequently the chip had a short back-end schedule for chip implementation and physical design. Accurate backend data were used early on to enable cost performance tradeoffs and avoid surprises late in the design.

The size and complexity of the chip presented challenges in term of tool capacity. In order to more efficiently solve the problem, the design was partitioned physically from the very beginning to take advantage of the hierarchy. Blocks were divided up according to their logic hierarchy and were partitioned until no block was more than 1 million gates. Such a block size allows reasonable runs for both place-and-route and timing analysis. A 24-hour turn-around time for any block was a goal that was met by using such restrictions. CBME has many replicated instances of blocks. The hierarchical approach also allows a block to be routed just once and then replicated using the identical routed object. CBME has 65 instances at the top-level using 26 unique routed block types.

Floorplanning was also done early on to better understand the top-level effects to the chip timing and architecture. Using a derated signal velocity metric based on idealized buffer insertion and Manhattan distance of top-level port connections, realistic top-level wire delays can be calculated to be used in a top-level timing analysis. Because of the sheer size of the chip, many signals that traveled a long distance at the top level needed to be broken up using registers. In certain instances, the logic design needed to be reworked to accommodate such changes. Block placement and port assignments were iteratively improved throughout the design.

Each block is required to register all of its inputs and outputs to ensure a uniform look and feel. This same uniformity is also translated in place-and-route as input and output buffers are placed at the boundaries of each block so that input loads and output loads can be characterized and abstracted from the block level. At the top level, the block level timing abstractions are then stitched up to complete the fully hierarchical analysis.

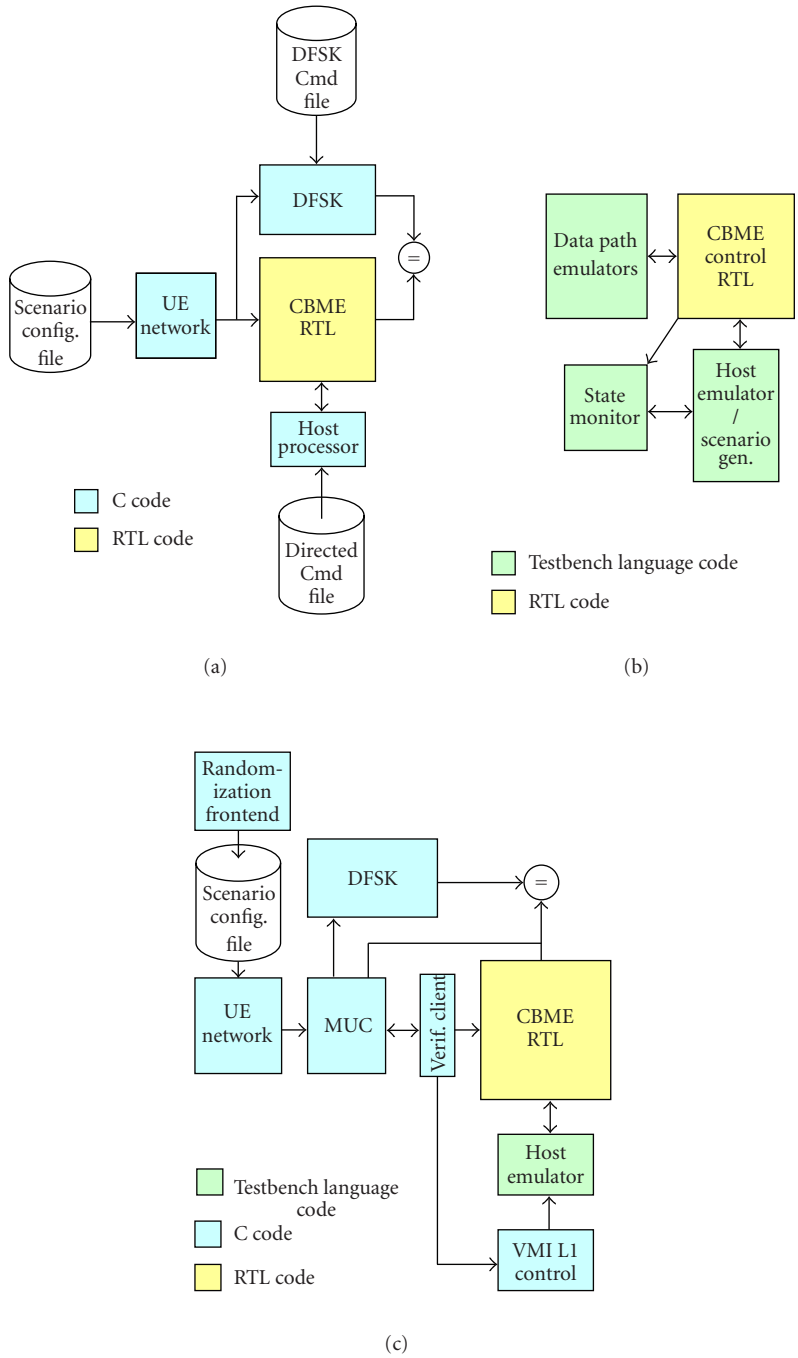


Figure 13: Verification environments: (a) directed testing, (b) random control, and (c) random system.

A significant amount of software was developed for the implementation of the CBME to both stitch together other point tools from the industry and to apply better techniques and algorithms at problems that industry tools do not solve well. The methodology balanced the need for performance and schedule. By taking on the design hierarchically and bringing up many of the physical implementation issues early and building in margins, the back-end process for the CBME was able to produce the design on time and at the predicted performance. A plot of the die is shown in Figure 14. The chip was fabricated in 0.18 μm process. Including pad ring, the die of the chip measures 16.7 mm $\times$ 16.7 mm. The total gate count after logic synthesis was 11 million nand2 equivalent gates. However, after buffer insertion and upsizing, the total increased to 14 million. There is a total of 4Mb of on-chip memory. The chip is designed to have 32 channels of capacity at 266 MHz and consumes 10 W at 1.8 V. The chip's frequency can be scaled down to do fewer channels at lower

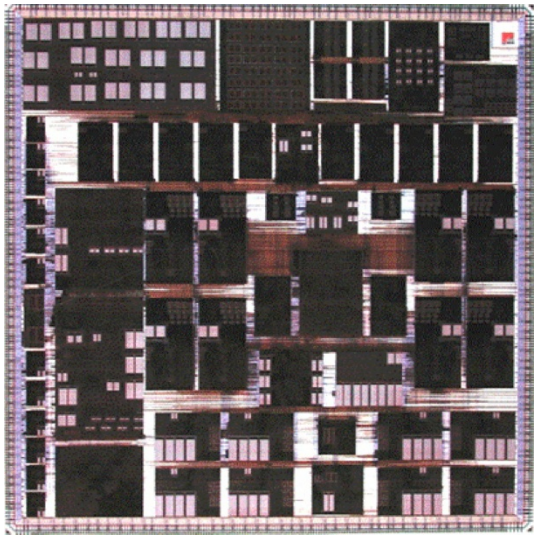


Figure 14: Die plot.

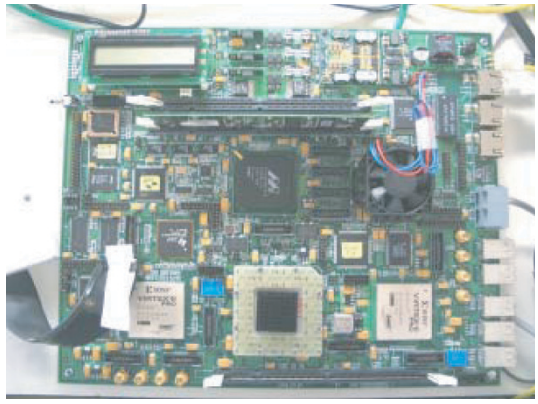


Figure 15: Channel card based on CBME 1.1.

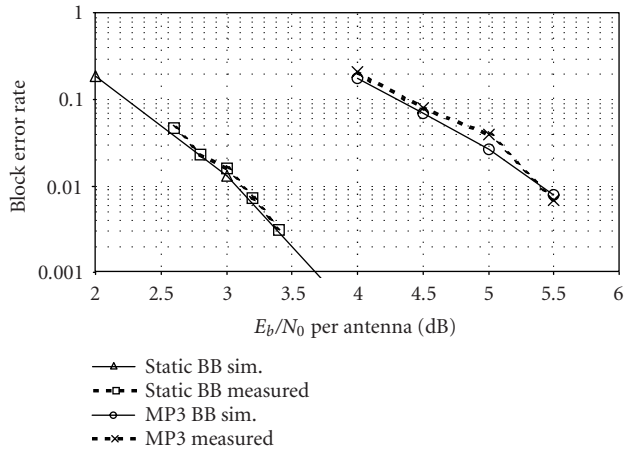


Figure 16: Link performance results for CS 12.2 Kbps.

frequencies and reduced power consumption. A channel card assembly using the CBME chip is shown in Figure 15.

8. SIMULATIONS, MEASUREMENTS, AND FIELD TRIAL RESULTS

The performance results of a bit-exact simulation model of the CBME are compared to measured results at baseband. The block error rate (BLER) versus E_b/N_0 from both simulations and measurements are shown in Figure 16 for circuit switched (CS) 12.2 Kbps service. Two diversity antennas are used at the receiver, all the loops were enabled (i.e., the frequency-locked loop and delay-locked loop and channel estimation). These results are an example of one configuration; other configurations may lead to improved link performance results. Figure 16 shows that there is a significant margin over the minimum performance specification [7]. The notation BB and MP3 denote baseband and multipath Case 3, respectively. The CBME chip is currently in field trials that include a complete 3GPP network with radio base stations, radio network controller and radio access network.

9. CONCLUSION

This paper presented the CBME processor design that can efficiently deal with 3G CDMA signal processing of various flavors. The current CBME design is able to support 32 mobiles when run in noncausal and on-chip variable spreading factor processing mode. The simulation of the baseband performance of the chip correlates well with laboratory bench testing. The CBME design continues to evolve in system performance, architecture and implementation techniques. Specifically, support of advanced receiver algorithms such as data-aided channel estimation [11] and beamforming/multiantenna support are in progress. Advanced process technology migration and new architecture improvements are also underway. All the new work is based on the foundation built up from all aspects of the CBME experience presented in this paper.

ACKNOWLEDGMENTS

The authors would like to thank R. Cavatur for his help in providing some of the micro-DSP data, and Michael Kohlmann for the baseband simulation results. Also, thanks to M. Nuotio, R. Krishnamurthy, and Dr. C. Drewes for their review and comments.

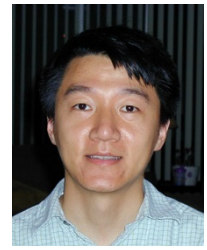
REFERENCES

[1] M. H. Callendar, "International Mobile Telecommunications-2000 Standards Efforts Of The ITU [Guest Editorial]," *IEEE Pers. Commun.*, vol. 4, no. 4, pp. 6–7, 1997, Special Issue.

[2] H. Andoh, M. Sawahashi, and F. Adachi, "Channel estimation filter using time-multiplexed pilot channel for coherent RAKE combining in DS-CDMA mobile radio," *IEICE Transactions on Communications*, vol. E81-B, no. 7, pp. 1517–1526, 1998.

[3] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: turbo-codes," in *Proc. IEEE International Conference on Communications (ICC '93)*, vol. 2, pp. 1064–1070, Geneva, Switzerland, May 1993.
[4] TIA/EIA/IS-95, *Mobile Station Base Station Compatibility Standard for Dual-Mode Wideband Spread Spectrum Cellular System*, July 1993.
[5] L. M. A. Jalloul, M. Kohimann, and J. Medlock, "SIR estimation and closed-loop power control for 3G," in *IEEE 58th Vehicular Technology Conference (VTC '03)*, vol. 2, pp. 831–835, Orlando, Fla, USA, October 2003.
[6] L. M. A. Jalloul and D. Martin, "System solutions based on multiple, small, and optimized DSP cores offer the best of both worlds," in *Proc. DesignCon*, Santa Clara, Calif, USA, February 2004.
[7] 3GPP TSG RAN TS 25.141, Base Station Conformance Testing (FDD).
[8] 3GPP TSG RAN TR 25.943, Deployment Aspects, June 2001.
[9] A. J. Viterbi, *CDMA: Principles of Spread Spectrum Communication*, Addison-Wesley, Reading, Mass, USA, 1995.
[10] R. Padovani, "The application of spread spectrum to PCS has become a reality: Reverse Link Performance of 1S-95 Based cellular systems," *IEEE Pers. Commun.*, vol. 1, no. 3, pp. 28–34, 1994.
[11] L. M. A. Jalloul and R. Misra, "Data-aided channel estimation for wideband CDMA," to appear in *IEEE Transactions on Wireless Communications* .

Louay M. A. Jalloul received his B.S. degree from the University of Oklahoma, Norman, Okla, in 1985, the M.S. degree from the Ohio State University, Columbus, Ohio, in 1988, and the Ph.D. degree from Rutgers University, Piscataway, NJ, in 1993, all in electrical engineering. Since September 2004, he has been an Associate Professor at the Department of Electrical and Computer Engineering, the American University of Beirut. He was with Motorola Inc. from 1993 until 2001 where he focused on CDMA research and development, including IS-95 base station modem ASIC, CDMA network design, receiver for CDMA location-finding, physical layer design, and development of a CDMA subscriber prototype used in a field trial. He made many significant contributions to CDMA standards bodies. He codeveloped the technology used in the third-generation enhanced modulation and encoding concept for integrated voice and high-data-rate CDMA system using spectrally efficient adaptive modulation and coding. He joined Morphics Technology Inc., Campbell, Calif, in February 2001, as the Director of Systems Architecture Group. After the acquisition of MorphICs, Inc. by Infineon Technologies AG in April 2003, he remained to be the Systems Design Group Manager working on the design of the CDMA cellular digital signal processor. He was a Teaching Assistant at the ECE Department, the University of Oklahoma, from 1985 to 1986, a Research Associate at the Electroscience Laboratory, the Ohio State University, from 1987 to 1988, and at the Wireless Information Networks Laboratory (WINLAB), Rutgers University from, 1989 to 1993. He received numerous engineering awards for his innovations for Motorola products and has 15 issued US patents. He is a Member of Eta Kappa Nu and is listed in the *American Men and Women of Science*.



Jim Lin was the VLSI Designer and Manager for the CBME chip at Morphics Inc. He has designed various kinds of embedded processors for over 11 years, in printing, multimedia, and communications. Recently he has joined Stream Processors, a new startup that promises ASIC efficiency and DSP programmability for a large class of applications. He holds an M.S. degree in electrical engineering from Stanford University and a B.S. degree in electrical and computer engineering from Carnegie Mellon University.