Implementation of Programmable Baseband Processors

Dake Liu[1]*, Eric Tell, and Anders Nilsson*
{dake, erite, andni}@isy.liu.se

Abstract – Implementation of programmable baseband DSP processors for digital radio communications is discussed in this paper. An implementation example based on IEEE802.11a/b/g WLAN is given. *Key words: Baseband signal processing, ASIP.*

1. Introduction

Three kinds of processors can be found in a communication terminal, DSP baseband processors (BBP), DSP application processors (APP), and micro controllers (MCU). Baseband signal processing covers all operations between ADC/DAC and the MAC (Medium Access Control) layer. The function coverage of a baseband processor is given in Figure 1.

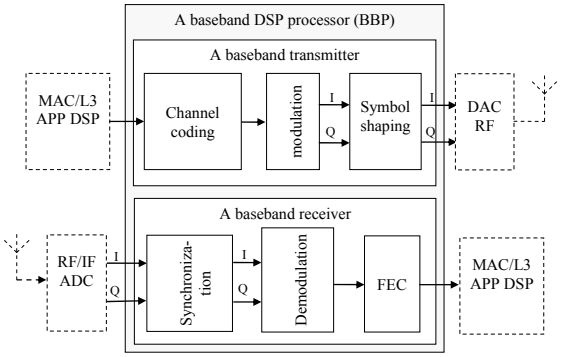

Figure 1 A BBP in a radio communication transceiver

Baseband DSP is a heavy task in communication systems requiring high performance because:

1. We must manage heavy computing for radio channel estimation and adaptation in order to manage the ISI (Inter Symbol Interference) problem.
2. The heavy task of channel estimation and adaptation must be repeated frequently because the estimated radio channel changes fast under high mobility conditions.
3. We must manage the baseband DSP processing on both word level and bit level in a short time because we must execute interleaving to eliminate burst error.
4. Heavy baseband DSP computing is required because we must decode the convolution code to recover data from noisy channels, for example Viterbi or Turbo codes.
5. To design a practical baseband processor, we must use limited processing precision and adapt ultra high dynamic range. Complicated and fast gain control is required.
6. We must schedule executions of baseband DSP tasks in a short time interval because the computing latency is limited by the MAC/L3 requirements.

It is difficult to design a high quality Baseband IC to fulfill requirements listed above, especially the high performance and the low power consumption. Therefore, classical Baseband IC was not programmable. By mapping tasks in every step in the receiver and the transmitter, baseband DSP use to be implemented as an ASIC based on many complex multiplier-accumulators (CMAC), many look up tables (LUT), many small memories, and many FSMs (Finite State Machines). A direct ASIC solution is implemented by mapping algorithm dataflows into circuits. Hardware reuse and multiplexing is almost impossible because the complexity of an ASIC must be limited and verification time of an ASIC must be acceptable. An example of implementing IEEE802.11a/b into an

---

[1] Dr. Dake Liu is the chair professor and director of Computer Engineering, Dept of electrical engineering at Linköping University, Linköping Sweden. Eric Tell and Anders Nilsson are currently Ph.D. students of Computer Engineering. Cooperation with ICC Shanghai and Chip Channel Inc. is acknowledged. The project is under Socware and STRINGENT of SSF in Sweden.

ASIC requires up to 8 real-valued MACs, 17 CMACs, 15 LUTs, and many small memory blocks. Including control and memory buffers, the equivalent gate count of this solution could be up to 500k gates, yet it is neither flexible nor scalable. However, as SDR (Software Defined Radio) requirements emerge, even more unknown standards should be included in an advanced radio product. This makes programmability and the use of DSP processors instead of baseband ASIC circuits necessary.

In the following section, the assembly instruction set will be investigated following the requirements discussed above. The processor core architecture and hardware accelerators will also be discussed and investigated. Furthermore task and loop level scheduling including hardware acceleration will be discussed. Finally, conclusions are drawn based upon silicon test chip.

2. Instruction set architecture of our baseband DSP processor

A BBP for IEEE802.11a requires about 4000 MIPS (based on a real-valued datapath, 1000 MIPS based on complex-valued datapath). 100% programmability is not feasible. Fortunately, a BBP does not need 100% programmability. The same as designing other ASIP (Application Specific Instruction set Processor) the essential issue in design of programmable BBP is to find the right partition of programmability and configurability. Taking IEEE802.11a/b as examples, we list tasks in table 1.

In the following table, the MIPS cost is defined as the number of instructions required to process a certain task divided by the time available for that task. The MIPS costs in table below are based on a complex datapath able to run one complex arithmetic operation per cycle. The cost does not include hidden MIPS costs of memory access for operands and results.

Table 1 Partition of main transceiver algorithms and map them to processor or accelerators

Tasks	Function	Algorithms	MIPS a/b	HW usage
Receive Filter	Anti alias & BPF	Complex FIR convolution	1200/440	accelerator
Packet detection	Start synchronization	Auto correlation $\Sigma x_i * x_{i-\tau}$	80/44	processor
Energy detection	Antenna diversity	Auto Correlation	80/44	processor
Freq. offset est.	Rotor coefficient	FFT / phase parameter decision	100/44	processor
Ch-estimation	Update channel model	FFT, FSM	160/80	processor
Payload Rotor	Rotor	Vector product	40/22	processor
IFFT/FFT	IFFT/FFT	64 points FFT and IFFT	60/-	processor
Normalization	Normalization	1/x and vector product	40/-	processor
Ch Compensation	Sub carrier compensate	LMS adaptive algorithms	80/20	processor
De-mapping	From symbols to bits	Decision FSM	40/22	accelerator
De-interleaving	Permutation	Permutation algorithm	260/-	accelerator
Descrambling	Bit permutation	$Y = x^7+x^4+1$, algorithm	80/16	accelerator
Viterbi decoding	Viterbi decoding	ACS and acceleration	1700/-	accelerator
Channel coding	Convolution coding	Convolutional coding	160/-	accelerator
Mapping	Modulation	Look up table	80/11	accelerator
Symbol shaping	Low pass filter	Two Real FIR convolution	1200/220	accelerator
RAKE receiver	4 fingers receiver	FIFO buffer / sum up taps	-/242	accelerator
De-spread	Despread	11 taps convolution	-/88	processor
CCK decoding	CCK decoding	DWT decoder	-/280	accelerator
CCK coding	Modulation	CCK FSM	-/70	processor
Spreading	Spread	Vector multiplication	11	processor

Notice that we actually do not need to execute all tasks in the table above at the same time. Algorithms are actually distributed into eight different operating modes: (see the scheduling in Figure 2). An example of a timing critical path is the processing of the long preamble in IEEE802.11a; This requires at least 20k clock cycles to be performed within 8 microseconds. This equals at least 2500 MIPS (not including FEC/Viterbi). Running a processor at more than 2500 MHz (1 operation/Clock cycle) is obviously impossible and we must either lower the MIPS cost for the algorithm or perform several operations in parallel. There are three ways to lower the MIPS requirements: Reducing the memory access cost; instruction level acceleration; and moving tasks out of the processor core by hardware acceleration.

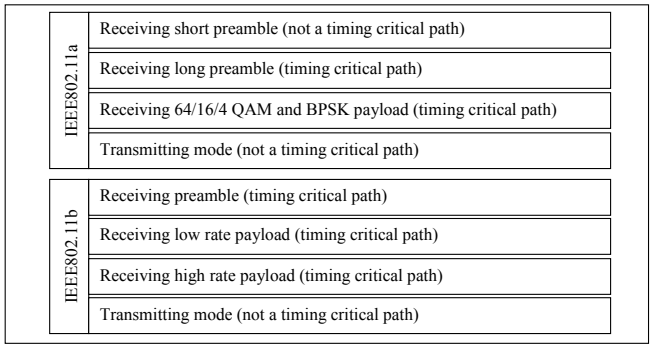

Figure 2 Scheduling plan for IEEE802.11a/b baseband DSP

To accelerate tasks on instruction level, we need to conduct careful benchmarking and profiling. The basic rule for instruction optimization is to find 10%-90% locality, i.e. to find 10% of instructions running during 90% of the time. The result is given in the following table:

Table 2: six most used instructions for a baseband DSP processor

Instructions	Functional specification
Complex convolution with implied modulo addressing	For I = 1 to N do {Complex REG <= Complex REG + V1[i] * (Conjugate of) V1(or 2)[i]}
Complex vector product	For I = 1 to N do {V3 [i] <= V1[i] * (Conjugate of) V2[i]}
Sum energy of a vector	For I = 1 to N do {accumulator <= Re $(V[i]^2)$+Im $(V[i]^2)$}
ABS approximation of complex data	REG <= Approximation of a complex $(Real^2 + Imaginery^2)^{1/2}$
Fast modulo FIFO memory access	Memory access with implied modulo addressing
Look up table	REG2 <= Memory [Segment + REG1]

By using a CMAC (Complex multiplication and accumulation unit) in the datapath and smart address generators, the MIPS requirement can be lowered by about 6 times. This lowers the MIPS cost from about 2500 MIPS to about 420 (FEC arithmetic not included). Memory accesses still consume about 30% to 50% of the total MIPS. By using memory sharing and DMA techniques we further reduce the MIPS cost by 30%. By these measures we have reached a final MIPS requirement of around 300 MIPS. All opportunities of hardware acceleration were carefully investigated under the flexibility requirement. All recurring fixed functions were identified and hardware accelerators were designed with enough configurability. After hardware acceleration, the MIPS cost of preamble processing was only about 150 MIPS including MIPS for controlling the program flow.

Table 3 Accelerators make the programmability feasible

Accelerators	Functions	Usage
FIR filter	Configurable complex / dual integer data types	Anti aliasing and symbol shaping
1/x circuit	Accelerates 1/x and gives a result per cycle	Normalization
De-mapper	Accelerates it and gives one result per cycle	BPSK/QPSK/16QAM/64QAM
Inerleaver	Accelerate permutation and its addressing	Block interleaving de-interleaving
CRC/Scrambler	A circuit for configurable CRC/SCR polynomial	CRC, scrambling, descrambling
Conv encoder	Circuit to accept configurable conv polynomial	Encoding convolution codes
Viterbi decoder	Trellis decoding, ACS (Add compare select)	Decoding convolution codes
DMA manager	Manage DMA and bus connector of memories	SOC platform

A programmable baseband DSP processor and its CMAC are given in Figure 3. The Silicon implementation is shown at the left part of figure 4 and the system level firmware scheduling is shown at the right part of figure 4. The scheduling shown in figure 4 is for payload reception of 64QAM and is the most timing critical. The chip was fabricated using 0.18um digital CMOS silicon with 6 metal layers. To be able to share the digital system clock with the AD/DA converters, the processor runs on 154MHz for 11b and 160MHz for 11a. The die size of the implemented BBP is 2.8 mm^2, and the chip size of the BBP is 4.9 mm^2 including 120 pins. Actually, the die size can be significantly reduce since only 50% of the memory is used for running firmware for 11a/b/g The total

silicon area consumed by memories is 1 mm^2. Therefore, the minimum silicon area can be 2.3 mm^2. However, this silicon cost does not include a Viterbi decoder. With Viterbi decoder included, the total die area of our next test chip will not be more than 3.3 mm^2 for 11a/b/g.

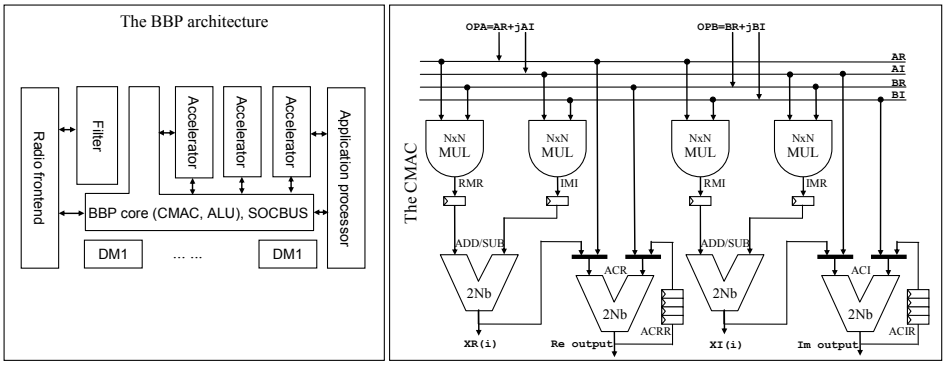

Figure 3 Architecture of a programmable BBP and its CMAC

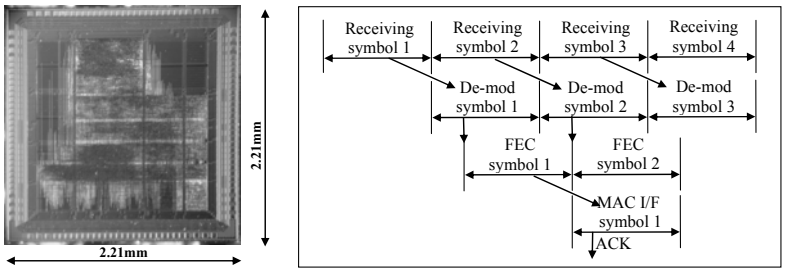

Figure 4 silicon implementation and task level (inter symbol) scheduling

4. Conclusion

Programmable baseband processors are required in SDR systems. By using advanced hardware parallelization techniques, hardware acceleration techniques, memory sharing technique, and custom static scheduling, we have demonstrated a programmable baseband DSP processor for IEEE802.11a/b(g). The silicon cost is considerably less compared to a custom ASIC solution. The power consumption is not higher than the power consumption of an ASIC when an operand masking technique is applied in the datapath. However the flexibility is not enough in the current solution to support even more demanding wireless standards. We are currently working on a second demonstrator based on a VLIW/SIMD mixed architecture.

References

[1]. Dake Liu and Eric Tell, Low-Power Baseband processors for Communications, To appear as chapter 23 in Low power Electronics Design, edited by C. Piguet, CRC 2004, ISBN 0849319412.

[2]. Anders Nilsson, Eric Tell, and Dake Liu, An accelerator architecture for programmable multi-standard baseband processors, in *Proc. of WNET2004, July, 2004, Banff Canada.*

[3]. Eric Tell and Dake Liu, A Hardware Architecture for a Multi-Standard Block Interleaver, Moscow, Russia July, 2004

[4]. Eric Tell and Dake Liu, A Converged Hardware Solution for FFT, DCT and Walsh Transform, in *Proc. of ISSPA2003, July, 2003, Paris France*

[5]. Heiskala, H. and Terry, J. T., *OFDM Wireless LANs: A Theoretical and practical guide*, Sams Publishing, 2002, ISBN 0672321572.

[6]. IEEE Std. 802.11a-1999 and IEEE Std. 802.11b-1999