LOW POWER BASEBAND PROCESSOR FOR IOT TERMINALS WITH LONG RANGE WIRELESS COMMUNICATIONS

Shunyao Wu[*], *Sungmoon Kang*[†], *Chaitali Chakrabarti*[*], *Hyunseok Lee*[†*]

[*] WISCA and School of ECEE, Arizona State University, Tempe, AZ 85287
[†] Dept. of ECE. Kwangwoon Univ., Seoul, South Korea 01897
{vincentw, chaitali}@asu.edu, {rkd1206, hyunseok}@kw.ac.kr

ABSTRACT

In this paper, we propose a new baseband architecture for Internet of Things (IoT) terminals that support long range communications such as those based on Orthogonal Frequency-Division Multiple Access (OFDMA) and spread spectrum technologies. We analyze the workload profiles of both systems and find that the frame detection unit has by far the highest computational load. Based on this analysis, we propose a simple architecture that uses only a scalar datapath. To optimize for low power consumption, we introduce application specific instructions that minimize register accesses and include address generation units for streamlined memory access. Preliminary synthesis results in 65nm node show that this architecture has an area of $0.204\mu m^2$ and average energy consumption of 2.41 nJ/cycle when operated at 3MHz with supply voltage of 1.08V.

Index Terms— Baseband processor, IoT, Long range communication, Low power

1. INTRODUCTION

In the very near future, we anticipate that machine to machine (M2M) communication will dominate internet traffic [1]. Smart sensing devices in autonomous cars, surveillance cameras, smart meters, health monitors, etc. will talk to other sensing devices without human intervention. This is the new era of IoT [2],[3]. To support IoT communications, there are short range systems such as WiFi Bluetooth, Zigbee, and z-wave, and long range systems such as LoRa, Sigfox, and LTE-M [4]. While short range systems are good for indoor applications, long range systems are designed for monitoring water, gas or infrastructure health without power and backbone network connectivity [5]. Legacy cellular networks, such as those based on LTE, are too expensive in terms of power and operation cost, and not applicable for long range systems.

Several architectures have been proposed for IoT terminals for short range range communications. There are commercial designs such as those from CEVA which consists of DSP with SIMD units [6], DSP with two VLIW slots from Tensilica [7], ARM core-based ARTIK series from Samsung [8]. There are also designs from academia such as the custom SIMD architecture with flexible bit-width [9] and CISC processor with reconfigurable microcode[10]. For long range IoT applications, there are a number of commercial solutions from Silicon Lab, Semtech, and TI. These are all low power versions of conventional RISC processors.

In this paper, we propose a baseband processor architecture for long range IoT systems based on OFDMA and spread spectrum technologies. Since an IoT terminal spends most of its time in the idle mode, we design an architecture which is optimized for idle mode. We perform detailed workload analysis of the two technologies and find that frame detection is by far the most dominant workload. Since frame detection implemented using sliding window is essentially a scalar operation, we propose a scalar processor based architecture. Reducing power consumption is extremely important, and so we introduce (i) application-specific instructions that help reduce power in register files through instruction chaining (where instructions are executed one after one without storing intermediate results), and (ii) streamline data access in memories through address generating units which hide the overhead of address calculations. The proposed architecture was synthesized using Cadence in 65 nm technology node. Preliminary synthesis results show that the area of this architecture is $0.204\mu m^2$ and that it consumes only 2.41 nJ/cycle whenclocked at 3MHz with supply voltage of 1.08V.

2. BACKGROUND

Long range wireless communication protocols for IoT systems support communications between small sensor nodes to a network server with minimum power. LTE-M, LoRa, and Sigfox are representative long range protocols. LTE-M is a narrow band version of LTE protocol that is used for M2M communication. It is based on OFDMA technology. LoRa is a new technology based on spread spectrum that targets low power long range (tens of kilometers) IoT terminals [11]. This technology is different from CDMA due to the use of relatively narrow band and chirp modulation scheme. While

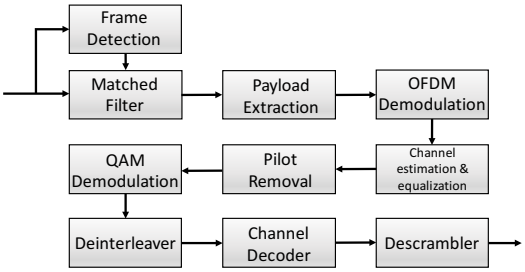

Fig. 1. OFDMA-based IoT receiver

the exact details of Sigfox are not known, we expect it to have characteristics similar to other narrow band protocol such as long idle mode and preamble based frame structure.

2.1. OFDMA-based IoT Terminal

Fig. 1 shows the receiver structure of an OFDMA-based IoT terminal. Frame detection unit detects the starting point of frames. This operation can be divided into coarse grain synchronization and fine grain synchronization. The computations in coarse grain synchronization can be represented by

$$s[n] = s[n-1] + p_n - p_{n-L}, \text{ where } p_n = r_n^* \cdot r_{n-L} \quad (1)$$

In this equation, r_n represents n-th input data and s[n] represents the n-th synchronization result computed with respect to received data r_n. An OFDM receiver generates a coarse grain detection signal if the value of $s[n]$ is greater than a predefined threshold value. Note that this operation consists of three simple operations when implemented using sliding window and is processed sequentially. This step is followed by fine grain synchronization which is essentially matched filter computation. Here cross-correlation between reference preamble and received signal is computed and the maximum value point is selected as the start of the signal. Since, cross correlation is implemented on a small range, the number of computations in fine grain synchronization is quite low.

The next step is payload extractio followed by OFDM demodulation using FFT. The user data placed on sub-carriers is extracted and fed to the channel estimator/equalizer. For the estimator, least mean square (LMS) algorithm is selected because of its mid range channel equalization performance without high computation load [12]. After pilot removal, N-QAM is used for demodulation. The data is then de-interleaved in order to achieve time diversity.

For forward error correction, convolutional code is used and decoding is done using the Viterbi algorithm. This algorithm can be represented by a set of vector operations, such as branch metric computation (BMC) and add compare select (ACS) operations [13]. Descrambler operation consists of bit-wise exclusive OR operations between the channel decoded data with a pseudo random sequence. Because bit wise operations can be performed independently, the descrambler operation can be computed using a bit vector operation.

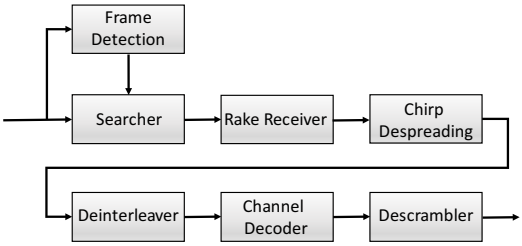

Fig. 2. Spread Spectrum-based IoT receiver

2.2. Spread Spectrum-Based IoT Terminal

Fig. 2 show the structure of spread spectrum-based IoT receiver. It consists of frame detection, searcher, Rake receiver, despreader, deinterleaver, channel decoder, and descrambler. The spread spectrum system uses a high frequency waveform to transmit information bits over the air. Its operation is similar to a CDMA system except that it uses chirp sequences. Using chirp sequence in modulation is advantageous for IoT terminals because it is robust to frequency offset error and does not require precise synchronization at base station. The searcher and Rake receiver are selected in order to get lower Bit Error Rate although they require additional computations.

After frame detection, the searcher estimates delay spread of received signal caused by multipath fading. The delay spread is estimated by continuous computation of correlations between received signal and chirp sequence. The high correlation peaks correspond to the location of the received signals. Rake receiver implements multiple demodulation paths with different delays. Each demodulation path (so called finger) performs matched filtering and its computation pattern is similar to vector inner product. Chirp sequence with varied frequency is used for chirp despreading. Channel code used in the spread spectrum-based system is typically Reed-Solomon (RS) code which shows good correction performance for burst errors. The decoding consists of four blocks, namely, syndrome computation, Berlekamp-Messy algorithm, Chien Search algorithm, and Forney algorithm. Of these, syndrome computation and Chien search algorithm can be parallelized [14]. Other modules such as frame detection, deinterleaving, and descrambling are similar to those in the OFDMA-based IoT terminals, and are not described here.

Table 1 shows the summary of workload characteristics of major computation kernels of OFDMA-based and spread spectrum-based IoT terminals. We see that while the frame detection and deinterleaver are scalar, most of the other kernels are suitable for vector processing. However frame detection is, by far, the most dominant workload and hence the baseband processor has to be optimized for scalar processing.

3. WORKLOAD CHARACTERISTICS

Tables 2 and 3 present the workload profiles of OFDMA and spread spectrum-based IoT terminals. To obtain the profiles, we built Matlab models of the two IoT terminals and executed them on an X86 machine. We measured the cycle count of

Table 1. Computational characteristics of IoT terminals

Algorithm	Vector	Scalar	Short Vect.
Frame detection		√	
Matched filter	√		
FFT	√		
Channel estimation	√		
Equalization	√		
Demodulation			√
Deinterleaver		√	
Viterbi decoder	√	√	
Descrambler	√		
Searcher	√		
Rake receiver	√		
RS decoder	√	√	√

Table 2. Workload Profile of OFDMA Terminal

Block	Idle:Active=10:1	Idle:Active=100:1
Frame detection	97.4763%	99.1896%
Matched filter	0.5408%	0.0603%
Payload extraction	0.4507%	0.1206%
OFDM demodulation	0.0451%	0.010%
Equalization	0.1352%	0.0502%
Pilot removal	0.0901%	0.0134%
QAM demodulation	0.2253%	0.0167%
Deinterleaving	0.0451%	0.0033%
Channel decoding	0.9013%	0.5291%
Descrambling	0.0901%	0.0067%

each algorithm, and computed the percentile contributions. In addition, to see the impact of long idle periods, we changed the ratio of operation time between idle state and active state from 10:1 to 100:1, though the ratio of idle to active state, in reality, is much longer.

In the OFDMA-based terminals, for ratio 10:1, frame detection is the dominant workload (>97%) followed by channel decoding. Consider a scenario where 1 out of 10 frames contains information. If each frame has 4,000 symbols, then the frame detection unit operates on 10×4,000 symbols. In contrast, the matched filter only operates on 100 symbols. After changing the idle to active ratio to 100:1, the frame detection block accounts for 99% of the workload.

In the spread spectrum-based terminals, frame detection is more than 99% of the workload even when the ratio is 10:1. This is because in spectrum spread systems, the frame size is much larger and so there are more calculations per frame. Note that in the active mode, our workload profile for OFDMA protocol is almost identical compared to that in [9].

4. PROPOSED BASEBAND PROCESSOR

An IoT terminal has to be optimized for idle mode operation. In the deep idle mode, all hardware is turned off except for timer. We focus on the light idle mode where only frame detection is done. Since the number of messages that are exchanged in an IoT application is very low, it is very important

Table 3. Workload Profile of Spread Spectrum Terminal

Block	Idle:Active=10:1	Idle:Active=100:1
Frame detection	99.4264%	99.8456%
Searcher	0.1941%	0.1261%
Rake receiver	0.0131%	0.0004%
Chirp despreading	0.0174%	0.0008%
Deinterleaving	0.0044%	0.0002%
RS decoding	0.3402%	0.0264%
Descrambling	0.0044%	0.0004%

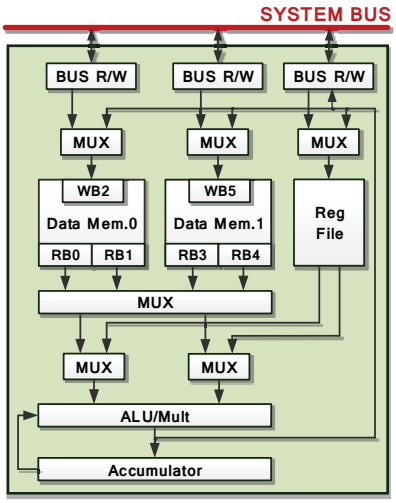

Fig. 3. Architecture of processor for long range IoT terminal

to reduce the cost of frame detection. This is done through use of a scalar processor (Section IV A) and low power instructions (Section IV B).

4.1. Processor Architecture

Fig. 3 shows the architecture of the proposed baseband processor designed for OFDMA and spread spectrum-based IoT terminals. It is essentially a 32bit scalar processor which consists of arithmetic and logic unit, an accumulator, register file of size 32bit×16, and two data memories each of size 8Kbytes. Each data memory has two read ports and one write port and can be programmed to read two entries and write one entry in one cycle. Each read or write port has a dedicated address generation unit (AGU) to access memory with minimal address calculation overhead. The ALU can operate on data from the register files or data memories. Alternately, the data memories can load data from the ALU and the RF unit. The control path consists of instruction memory and instruction decoder which are not shown in this figure for simplicity.

The choice of scalar processor was derived from the workload analysis results which showed that even when idle to active period ratio is 10:1, frame detection (which is a sequential algorithm) accounts for 97%-99% of the workload. Apart from frame detection, the other baseband signal processing algorithms can be represented by vector inner products and

Table 4. Key application-specific instructions of the proposed baseband processor

Instruction	Operation, $n \in \{0, \ldots, N-1\}$	Target algorithms
adsub R1, R2	ACC = ACC + R1 - R2	Frame detection
agurd R_n	if(R_n/bit(i)) == 1), memory read/write using AGU_i	All algorithms
cblt R1, R2, #label	branch to #label address if (R1<R2)	Frame detection
mac R1, R2	ACC = ACC + R1 * R2	FFT, channel estimation, searcher, rake receiver
diff R1, R2, R3	R1 = abs(R1 - R2)	Viterbi-BMC, demodulation
csel R1, R2, R3	R1 = (R1<R2) ? R1 : R2	Viterbi-ACS, Viterbi-TB, demodulation
gfmul R1, R2, R3	$R1 = R2 \otimes R3$, where $\otimes$ is Galois field multiplier	RS Berlekamp-Massey

Table 5. Synthesis results of the proposed baseband processor with 3MHz operation frequency

Blocks	**Area** (μm^2)	**Energy** (nJ/cycle)
ALU/MUL	0.0068	0.11
Other datapath	0.0085	0.12
Reg. file	0.0063	0.82
AGU, RW buffer	0.002	0.16
Data mem.	0.09	0.55
Control	0.0903	0.64
Total	0.204	2.41

implemented by a SIMD style architecture. However, in reality, idle period is much longer than active period and thus the power saving and throughput enhancement that we can expect from using SIMD datapath, is almost negligible.

4.2. Application-Specific Instructions

Table 4 shows the key application-specific instructions of the proposed processor that account for 30% of all instructions; the rest of the instructions are standard RISC instructions. Most of the application-specific instructions employ instruction chaining [15], where instructions are executed one by one without storing the intermediate values in the register file and helps to reduce the power consumption in the register file.

For efficient computation of frame detection, we introduce *adsub* a chained instruction which subtracts a value from the accumulator and adds another value to the accumulator. Another instruction *agurd* allows selective execution of memory read/write operation; the selection is done through the i-th bit of the operand register. Using this instruction, up to four memory reads and two memory writes can be performed in parallel. The *agurd* instruction is used to optimize operations in frame detection and other vector operations. The *mac* instruction, which multiplies two input data and adds the product to the accumulator, is widely used in implementing vector inner product used in FFT, channel estimation, searcher, and Rake receiver algorithms. The *diff* instruction computes the absolute difference between two values. This operation frequently appears in the N-QAM demodulators and BMC operation in Viterbi decoder. Other instructions include *csel* which implements compare and select operation required for Viterbi and *gfmac* which is almost identical to *mac* except being designed for Galois field arithmetic.

To demonstrate the effectiveness of this instruction set, we implemented the frame detection algorithm both on a legacy scalar processor and on the proposed processor. By exploiting application-specific instructions, the number of required instructions reduced from 10 to 4. Since frame detection accounts for 97 to 99% of the workload, this reduction results in significant energy savings.

5. EXPERIMENTAL RESULTS

In this section, we discuss the hardware implementation results of the proposed baseband processor. We implemented the processor using Verilog and synthesized it using Cadence tool with 65nm library cells. The area and energy results of the synthesized hardware (without layout optimization) are shown in Table 5. The area of the processor is 0.204 μm^2 with half of the area being occupied by the two data memories. The operation frequency is set at 3MHz. This was determined by the maximum allowed processing time for baseband and medium access control (MAC) layer processing.

The average energy consumption is 2.41 nJ/cycle for 3MHz operation frequency and 1.08V operation voltage. Table 5 shows that energy consumption of register file is the highest followed by energy consumption of control system and data memory. Together they account for about 80% of the total energy. Finally, if we assume the operation condition of an IoT terminal for telemetry application as 10 sec active duration and 4 activations per day, the proposed processor only consumes 0.0003 mAH per day.

6. CONCLUSION

In this paper we proposed a baseband processor for long range wireless communication IoT terminals based on OFDMA and spread spectrum technologies. We first characterized the computation pattern of baseband processing operations in the two types of terminals and found that frame detection is, by far, the dominant workload. So we developed a scalar processor that is optimized for frame detection. It uses specialized chained instructions to reduce power overhead caused by register file accessing and address generation units to minimize address calculation overhead. Preliminary synthesis results in 65nm show that our processor architecture consumes 2.41nJ/cycle when running at 3MHz with 1.08V supply voltage and has an area of 0.204μm^2.

7. REFERENCES

[1] G. Lawton, "Machine-to-Machine Technology Gears Up For Growth," *Computer*, vol. 37, no. 9, pp. 12–15, Sept 2004.

[2] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of Things for Smart Cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, Feb 2014.

[3] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context Aware Computing for The Internet of Things: A Survey," *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 414–454, First 2014.

[4] R. Ratasuk, N. Mangalvedhe, A. Ghosh, and B. Vejlgaard, "Narrowband LTE-M System for M2M Communication," in *2014 IEEE 80th Vehicular Technology Conference (VTC2014-Fall)*, Sept 2014, pp. 1–5.

[5] Texas Instruments Inc. (2014) Long-range RF communication: Why narrowband is the de facto standard. [Online]. Available: http://www.ti.com/

[6] CEVA Inc. (2015) Ceva-xc5. [Online]. Available: http://www.ceva-dsp.com/CEVA-XC5

[7] Cadence Design Systems, Inc, "Tensilica Fusion F1 DSP for IoT/Wearables," 2016. [Online]. Available: http://ip.cadence.com/ipportfolio/tensilica-ip/fusion#fusion-features

[8] SAMSUNG, Inc. (2016) Artik modules overview. [Online]. Available: https://www.artik.io/modules/overview/artik-1/

[9] Y. Chen and et al, "A Low Power Software-Defined-Radio Baseband Processor for the Internet of Things," in *Proc. of IEEE International Symposium on High Performance Computer Architecture (HPCA)*, March 2016.

[10] N. Ma, Z. Zou, Z. Lu, L. Zheng, and S. Blixt, "A Hierarchical Reconfigurable Micro-Coded Multi-Core Processor for IoT Applications," in *Reconfigurable and Communication-Centric Systems-on-Chip (ReCoSoC), 2014 9th International Symposium on*, May 2014, pp. 1–4.

[11] LoRa Alliance Technical Marketing Workgroup 1.0, "A Technical Overview of LoRa and LoRaWAN," Nov. 2015.

[12] X. Cai and G. B. Giannakis, "Error Probability Minimizing Pilots for OFDM with M-PSK Modulation Over Rayleigh-Fading Channels," *IEEE Transactions on Vehicular Technology*, vol. 53, no. 1, pp. 146–155, Jan 2004.

[13] Y. Lin, H. Lee, M. Woh, Y. Harel, S. Mahlke, T. Mudge, C. Chakrabarti, and K. Flautner, "SODA: A Low-power Architecture For Software Radio," in *33rd International Symposium on Computer Architecture (ISCA'06)*, 2006, pp. 89–101.

[14] H. Lee and et al, "Implementation of the Chien Search Algorithm on Application Specific Instruction Set Processor," in *Proc. of International Conference on Comsumer Electronics (ICCE)*, Jan. 2012.

[15] C. E. Kozyrakis and D. A. Patterson, "A new Direction For Computer Architecture Research," *Computer*, vol. 31, no. 11, pp. 24–32, Nov 1998.