Power Reduction Method and Design of ASIPs for Embedded Systems

January 2013

Hirofumi IWATO

Power Reduction Method and Design of ASIPs for Embedded Systems

Submitted to
Graduate School of Information Science and Technology
Osaka University

January 2013

Hirofumi IWATO

Publications

Journal Articles (Refereed)

[J1] Hirofumi Iwato, Keishi Sakanushi, Yoshinori Takeuchi, and Masaharu Imai, ”A Small-area and Low-power SoC for Less-invasive Pressure Sensing Capsules in Ambulatory Urodynamic Monitoring,” IEICE TRANSACTIONS on Electronics, Vol.E95-C, No.4, pp.487-494, 2012.

[J2] Hirofumi Iwato, Keishi Sakanushi, Yoshinori Takeuchi, and Masaharu Imai, ”Low-power ASIP Generation Method by Extracting Minimum Execution Conditions,” IPSJ Transactions on System LSI Design Methodology, Vol.3, pp. 222-233, 2010.

International Conference Papers (Refereed)

[I1] Hirofumi Iwato, Keishi Sakanushi, Yoshinori Takeuchi, and Masaharu Imai, ”Low Power SoC for Pressure Measurement Capsules in Ambulatory Urodynamic Monitoring,” COOL Chips XIII, April, 2010.

[I2] Hirofumi Iwato, Keishi Sakanushi, Yoshinori Takeuchi, and Masaharu Imai, ”Low Power ASIP Generation Method by Means of Extracting Non-redundant Activation Conditions,” Student Forum of the 13th Asia and South Pacific Design Automation Conference (ASP-DAC), January, 2008.

[I3] Hirofumi Iwato, Keishi Sakanushi, Yoshinori Takeuchi, and Masaharu Imai, ”Low Power VLIW Processor Generation Method by Means of Extracting Non-redundant Activa-

tion Conditions," in Proceedings the 5th IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES/ISSS), pp. 227-232, 2007.

[I4] Hirofumi Iwato, Keishi Sakanushi, Yoshinori Takeuchi, and Masaharu Imai, "Low Power Configurable Processor Generation Method with Fine Clock Gating to Pipeline Registers and Register Files," in Proceedings the 13th Workshop on Synthesis and System Integration of Mixed Technologies (SASIMI), pp.241-245, 2006.

International Conference Papers (Invited)

[I5] Hirofumi Iwato, Keishi Sakanushi, Yoshinori Takeuchi, and Masaharu Imai, "Architecture Level Power Reduction Method for Configurable Processor Generation," in Proceedings 8th International Forum on Application-Specific Multi-Processor SoC (MPSoC), June, 2008.

Domestic Conference Papers

[D1] Hirofumi Iwato, Keishi Sakanushi, Yoshinori Takeuchi, Masaharu Imai, Akira Matsuzawa, and Yoshihiko Hirao, "The Architecture of an SoC for Miniature Size Capsule to Measure Pressure Inside Body," Medical Information and Communications Technology, no. 9, pp. 99-112, July, 2009 (in Japanese).

[D2] Hirofumi Iwato, Takuji Hieda, Hiroaki Tanaka, Jun Sato, Keishi Sakanushi, Yoshinori Takeuchi, and Masaharu Imai, "Highly Extensible Base Processors for Short-term ASIP Design," IEICE Technical Report, VLD2007-92, vol. 107, no. 336, pp. 19-24, November, 2007 (in Japanese).

[D3] Hirofumi Iwato, Keishi Sakanushi, Yoshinori Takeuchi, and Masaharu Imai, "Generation Method for Low Power Configurable Processor," IPSJ Symposium Series, vol. 2005, no. 9, pp. 219-224, August, 2005 (in Japanese).

Summary

In recent years, embedded systems are widely used in our daily life. To implement a lot of requirements in a small embedded system, tight constraints must be met at the same time. Power consumption is the most important constraint for modern embedded systems. For high performance systems, power consumption increases heat of embedded systems, and for small systems, power consumption requires bigger battery capacity and increases system size.

Application Specific Instruction-set Processors (ASIPs) are appropriate to implement embedded systems because ASIPs can offer high performance per energy and high programmability. To design high performance embedded systems, Very Long Instruction Word (VLIW) type ASIPs (VLIW ASIPs) are suitable, and Design Space Exploration (DSE) must be performed to determine the optimal architecture parameters. On the other hand, to implement small embedded systems, small scalar ASIPs with specific instructions must be designed by taking into account the characteristics of the target systems.

This thesis discusses two topics to challenge low power embedded system designs for high performance systems and small systems. First, this thesis proposes a low power VLIW ASIP generation method. The proposed method generates VLIW ASIPs with clock gating technique, which is a well known low power technique to reduce power consumption of flip flips. Power reduction by clock gating depends on gating signals of registers, and the gating signals are derived from execution conditions of the registers. In order to minimize power consumption of pipeline registers in a large-scale data path of a VLIW ASIP, the proposed method automatically extracts the minimum execution conditions from ASIP generation procedures. The experimental results show that the proposed method drastically reduce power consumption of VLIW ASIPs, and overheads to implement clock gating are small.

The second topic is to design the pressure sensing system for Ambulatory Urodynamic Mon-

itoring (AUM). In AUM, the sensors measure pressure data of the bladder and the rectum, and the size of the sensor must be small for less-invasive tests. Therefore, designing the small capsular sensor is beneficial to less-invasive urodynamic tests. To design small pressure capsule, the System-on-a-Chip (SoC) which integrates almost functions of the capsule is strongly required. This thesis describes the requirements and constraints of the SoC, the designs of the capsule, the SoC, and the ASIP which is designed for this application. The experimental results show that the designed SoC and ASIP are low power consumption and small area implementation, the implemented capsule satisfies constraints for AUM, and correct operations of the sensing system are confirmed.

Acknowledgments

I would like to deeply thank Prof. Masaharu Imai for guiding me throughout my PhD days. I appreciate his advices, technical suggestions, and insights for my researches. Through discussions with him, I have acquired expertise in my research area.

I would like to thank Prof. Takao Onoe and Prof. Tetsuya Yagi for reviewing this thesis and comment on my study.

I would like to thank Prof. Yoshinori Takeuchi for his guidance. His sound advices motivate me to improve my research and papers, and he takes a great care of me at all times.

I am thankful to Dr. Keishi Sakanushi. His beneficial comments on my work help me to progress my research. He teaches me a lot of things about researching, writing paper, and academic mentality.

I would like to thank Dr. Kyoko Ueda, Dr. Mohamed AbdElSalam Hassan, Dr. Yuki Kobayashi, and Dr. Hiroaki Tanaka. They friendly discuss with me on my research and give many useful comments and advices.

I thank to Dr. Takuji Hieda. He is co-worker and helps me at various situations. Wit talking with him makes my PhD days enjoyable.

I would like to thank Dr. Makiko Itoh, Dr. Shinsuke Kobayashi. They made the basis of my research.

I would like to thank Prof. Masaya Miyahara, Dr. Takashi Kurashina, and Mr. Kennichi Matsunaga. They help development of the SoC that is described in this thesis, and their modules are implemented in the SoC.

I am thankful to Ittetsu Taniguchi, Takashi Hamabe, Takahiro Notsu, Hyun Kim Long, Takeshi Shiro, Takahiro Itoh, Akira Kobashi, Yu Okuno, Hitoshi Nakamura, Ayataka Kobayashi, Aiko Takabe, Kazuhiro Kobashi, Masahiro Kondo, Hiroki Ohsawa, and the rest of all members

of Integrated System Design Laboratory in Osaka University. They helped me and made my student life joyful and vivid.

I wish to thank my parents Yuji and Michiko, and my sister Mina and her husband Hideyuki Nishizawa, and also nephew Ryo and niece Yuki.

And Finally, for my dear Yuki, thank you for supporting me with your precious smile.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Nowadays embedded systems are used to make our daily life useful and safe. Ubiquitous information systems such as cellular phones, tablet computers, sensor networks, and medical systems are widely adopted in modern society. Since each application requires a lot of functions in small size implementation, the design constraints of embedded systems are very tight. Especially in modern embedded systems, power consumption is the critical problem. For the systems which require high performance computing, power consumption increases heat of the systems and makes the design difficult. On the other hand, for the systems which require portable and small implementation, not requiring high performance, power consumption increases battery capacity, and makes the system size bigger. There are a various types of requirements for embedded systems and power reduction is the common design challenge for designers.

Embedded systems are implemented by Application Specific Integrated Circuits (ASICs), General Purpose Processors (GPPs), or Application Specific Instruction-set Processors (ASIPs). ASICs can achieve high computation performance with low power consumption, but they have disadvantage of flexibility: they are cannot reprogrammed after design completion. In contrast, GPPs can offer flexibility to designers, however, they are not efficient implementation in terms of performance per energy. On the other hand, as shown in Figure 1.1, ASIPs can meet the conflicting requirements by means of using application specific instructions. ASIPs offer high flexibility and high performance per energy at the same time. Therefore ASIP is appropriate

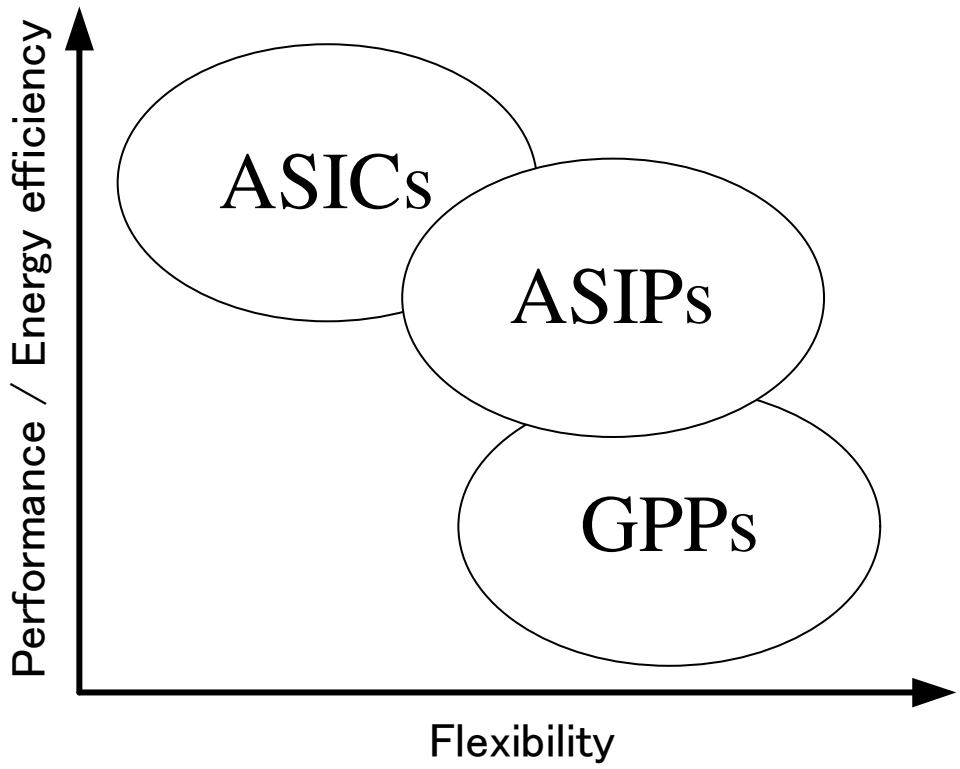

Figure 1.1: A trade-off between flexibility and performance per energy.

architecture to implement embedded systems.

1.1 Embedded System Design with ASIPs

For high performance processor architecture, superscalar processor [1] and Very Long Instruction Word (VLIW) processor [2] are proposed. Both types exploit Instruction-Level Parallelism (ILP) to increase Instructions Per Cycle (IPC) performance; however, they have difference in instruction scheduling mechanism. Superscalar processors fetch multiple instructions at the same cycle, detect ILP by hardware, and schedule the instructions at execution time. Therefore, superscalar processors need extra hardware to implement parallel execution. Such extra hardware increase implementation size and power consumption. On the other hand, VLIW processors do not need extra hardware to detect ILP and schedule instructions because instructions are analyzed by software, and the scheduled instructions are placed on instruction memories in

advance. Since VLIW processors can perform parallel execution by simple hardware, VLIW is the suitable architecture for high performance embedded systems.

For small and low power embedded systems, VLIW and superscalar architecture are not suitable; small scalar ASIPs with a limited instruction set are appropriate. In such systems, designing small and low power ASIPs is not sufficient to meet the design constraints. By taking into account the rest of the functional blocks in the system, appropriate implementation of special instructions is required for low power design. Therefore, the low power ASIP design for small systems depends on each particular application.

1.2 Design Challenges

To design low power embedded systems with ASIPs is still challenging task for designers.

1.2.1 ASIP Generations with Low Power Technique

When designing VLIW type ASIPs (VLIW ASIPs), or scalar ASIPs, simultaneously satisfying the tight constraints is required: computation performance, area, and power consumption. For this purpose, Design Space Exploration (DSE) should be performed to determine the optimal architecture parameters of ASIPs [3]. One of the main concerns is that DSE is time consuming task; manually performed DSE decreases design productivity.

For rapid DSE, the automatic ASIP generation methods that achieves significant short design time have been proposed [4–8]. The generation methods automatically generate ASIPs based on the specifications described with Architecture Description Language (ADL) [9]. Designers can flexibly describe architecture with ADL and reduce design time by using architecture generation. However, the conventional ASIP generation methods place a priority on minimizing area and delay. For reducing power consumption, it is required to generate ASIPs with low power techniques.

There are many low power techniques through all design levels. In Register Transfer Level (RTL), clock gating [10] and operand isolation [11] are familiar. Based on the gating conditions, clock gating cuts off the clock supplies causing unnecessary switching inside the registers. Operand isolation stops unnecessary switching to temporarily unused blocks in a data path.

Since an enormous amount of energy is consumed by registers in synchronous circuits [12], the clock gating appears to be a promising technique for power reduction. Generally, there are many pipeline registers in the large-scale data path of a VLIW ASIP. Furthermore, the number of pipeline registers rapidly increases when widening the parallel issue or deepening the pipeline; thus a large amount of energy is dissipated in the pipeline registers [13]. For this reason, clock gating on pipeline registers is expected to be effective for VLIW ASIP generation.

1.2.2 An SoC for Pressure Measurement Capsule in AUM

Biomedical information sensing has become an essential part of health care systems. Measuring accurate biomedical information throughout the day is indispensable for diagnosing symptoms. This thesis deals with pressure measurement capsules for urodynamics as a beneficial specific application of a small embedded system.

Detrusor pressure measurement is indispensable for diagnosing Lower Urinary Tract Symptoms (LUTS) attributed to underlying causes such as neurogenic bladder, prostatic hyperplasia, and malignancy. Since the detrusor pressure can be calculated by subtracting the abdominal pressure from the intravesical pressure [14], the pressures in the urinary bladder and rectum are simultaneously evaluated by using cystometry [15], which is a major method to measure the detrusor pressure [16]. However, the results of cystometry tend to fail to reproduce and explain symptoms because cystometry involves artifacts by means of non-physiological procedures, e.g., confining patients to a stressful examination room for hours, inserting catheters through the urethra, and filling the bladder with saline. For better urodynamic study, any artifacts should be removed.

Ambulatory Urodynamic Monitoring (AUM) has become an established diagnosing method for LUTS [17]. AUM is expected to reduce artifacts for pressure measurement becuase AUM releases patients from the stressful environment and investigates the natural activity of the urinary system in daily life. To conduct pressure measurement in AUM, several pressure sensing systems have been proposed by using catheters or implanting sensing devices. Although these systems can be used in home, the results are still supposed to contain artifacts. To improve plausibility of the recorded data, inserting catheters and invasive incision should be removed

for less-invasive pressure measurement in AUM.

In order to perform AUM, sensing devices must be small capsule form because big sensors make patients feel uncomfortable, or even pain, resulting undesirable artifacts. A small capsule can be inserted in the bladder and directly measure pressure data. Since insertion is less-invasive and the inserted capsule does not disturb daily activities, the capsule sensor contributes less-invasive pressure sensing in AUM. Several pressure sensing devices have been proposed to tackle realizing AUM. However, the proposed sensing devices are too big to implement in a miniature size capsule. The main reason of the big size is due to the multiple chips on a Printed Circuit Board (PCB). To implement small size pressure measurement sensor for AUM, a System on a Chip (SoC), which integrates multiple components in a single chip, is an appropriate solution to the capsule form pressure sensing devices.

1.3 Contributions

This thesis discusses low power embedded system design with ASIP generation for both high-performance and small applications.

As a low power design for high-performance systems, a low power VLIW ASIP generation method is described. To apply clock gating to VLIW ASIP generation method is necessary for low power ASIP design because VLIW ASIPs contain a lot of pipeline registers. Power reduction by clock gating depends on gating conditions. Extracting the minimum execution conditions is important for reducing power consumption. However, the extraction in a complex data path is time consuming and error-prone task. The low power VLIW ASIP generation method automatically extracts the minimum execution conditions in ASIP generation procedures. By using the minimum execution conditions, the proposed method generates clock gating control signals and minimizes pipeline register activities.

As a low power design for small systems, the design of an SoC with an ASIP for pressure sensing capsules for AUM is tackled. By using the ASIP, the SoC is designed as small enough to implement the pressure sensing capsules which are inserted in the bladder and rectum. The main tasks of the SoC are: sensing pressure data in the bladder and rectum, receiving commands and transmitting the data to an outside recorder via wireless link. Designs of sensing system

framework, capsule implementation, SoC architecture, wireless protocols, digital circuits for wireless modem, and an ASIP are studied in this thesis. The ASIP design is the main concern of the SoC. The ASIP in the SoC is very small and pipeline registers are not used in it. Therefore, although the low power VLIW ASIP generation method described in this thesis is applicable, it is not effective. Effective implementation of specific instructions is required to contribute power reduction and small area implementation.

1.4 Organization

The remainder of this thesis is organized as follows. Chapter 2 discusses related work. Chapter 3 describes low power VLIW ASIP generation method by means of extracting minimum execution conditions. The designs of the pressure sensing capsule and the SoC for AUM are studied in Chapter 4. Conclusion and future work are described in Chapter 5.

Chapter 2

Related Work

This chapter describes related work of ASIP generation methods, low power techniques and methods, and pressure sensing systems for AUM.

2.1 Low Power Techniques for Digital Circuits

There are mainly two low power techniques for digital circuit design: clock gating and operand isolation. Figure 2.1 and 2.2 show clock gating circuit and operand isolation circuits.

Clock gating shuts off redundant clock supplies to flip flops when the output of the flip flips are not necessary in the following calculation stages. Additionally, clock gating can reduce power consumption in clock trees if the clock gate can be placed on higher level of the tree. Although clock gating is the effective low power technique, clock gating has a disadvantage in layout design; the placed gates make clock tree synthesis difficult because clock skew increases.

On the other hand, operand isolation stops unnecessary signal switchings to combinational logics. Operand isolation can reduce power consumption if the target combinational logic consumes large energy. However, Operand isolation involves many circuit overheads. The inserted gates increases size, power consumption, and critical path delay. To use operand isolation, designers have to care the overheads and determine isolation targets to gain net power reduction.

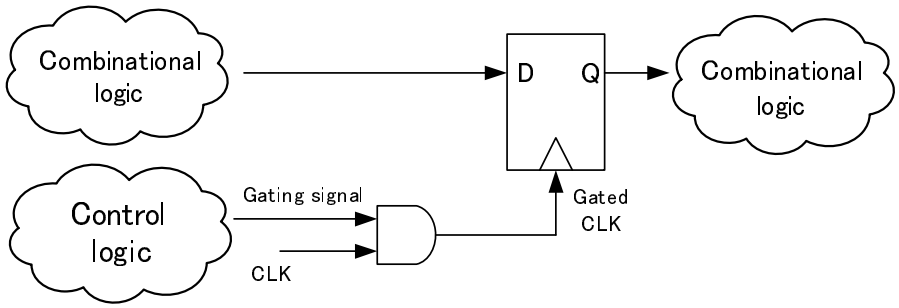

Figure 2.1: Clock gating.

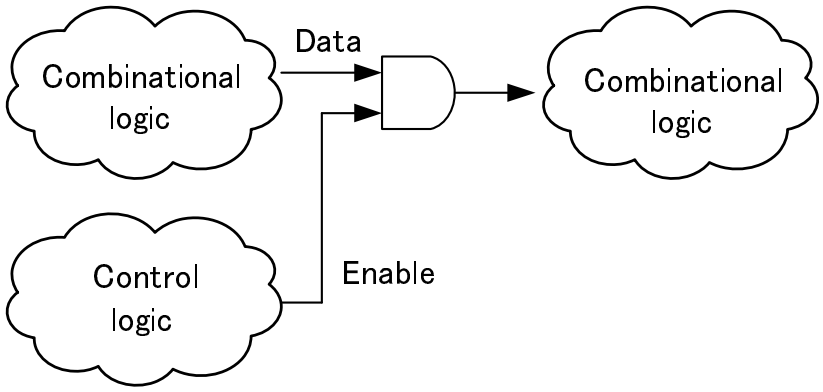

Figure 2.2: Operand Isolation.

2.2 ASIP Generation Methods

In this section, the overview of ASIP generation method is described, and the issue for applying clock gating to the ASIPs generated by those method is discussed.

2.2.1 ASIP Generation Methods

In order to automatically generate ASIPs, scalar ASIP and VLIW ASIP generation methods are proposed [7] [18]. Those methods generate ASIPs with an ADL called Micro Operation Description (MOD). The MOD describes architecture parameters, port information, using resources, the data flows of instructions. By using MODs, ASIP generation methods automatically construct a data path and a controller. In the generation methods, pipeline registers are

automatically inserted, and control logic for the pipeline registers are created as follow:

$$en_p = \overline{stall_{stage_p}}, \tag{2.1}$$

where p is a pipeline register and $stage_p$ represents the pipeline stage number to which p belongs. $stall_n$ stands for a condition of a pipeline interlock caused by several situations, e.g., structural hazard or multi-cycle operation. When en_p is true, data are stored and calculation advances to the next pipeline stage, and when en_p is false, pipeline stage $stage_p$ is stalled. Since the aim of conventional VLIW ASIP generation is to design high-speed and small-area ASIPs; the form of signal (2.1) is simplified as much as possible.

2.2.2 Minimum Execution Conditions

To apply clock gating to pipeline registers in an ASIP, using control signal (2.1) is a trivial method. By using control signal (2.1) for clock gating, clocks are supplied to all pipeline registers when the pipeline is not stalled. However, applying control signal (2.1) has less effect on power reduction because the pipeline rarely stalls in well-optimized VLIW programs. To reduce power consumption in such cases, it is necessary to consider the fact that a large portion of the data path is idle during instruction execution even if the pipeline is not stalled. Clocks must be supplied to only necessary pipeline registers for power reduction.

In order to supply clocks to the only necessary pipeline registers, *minimum execution conditions* are introduced in this thesis. The minimum execution conditions of a resource are conditions for clock supply and represented by a set of the instructions which use the resource for their execution, i.e., the resource is used only when the instructions in the condition are executed. By using the minimum execution conditions for the control signals of clock gating, clock supplies are limited to only necessary pipeline registers; therefore, power consumption is reduced.

Manual extraction of the minimum execution conditions is not practical because it requires a long-term design period and is error-prone. An automated extraction method of the minimum execution conditions is strongly required. To automatically obtain gating signals, two major approaches are known: forward [19] [20] and backward [21]. The forward approach extracts the gating signals through high-level synthesis processes using high-level architecture infor-

mation. On the other hand, the backward approach extracts the signals by analyzing low-level designs. A merit of the backward approach is that it can be widely applied to arbitrary circuits. However, they need long calculation time and involve large area overhead due to the analysis of their complex design. Therefore, the forward approach is suitable for VLIW ASIP generation. Obviously, a forward approach must be designed for each high-level synthesis method. With respect to the VLIW ASIP generation method, an automated extraction method of minimum execution conditions is not known yet.

2.3 Automatic Clock Gating Methods

Several automatic clock gating insertion methods and tools are currently available.

Power Compiler [22], which is the most widely known commercial tool, automatically inserts gates into the clock lines of registers. However, it does not extract the gating signals of the registers; that is, the efficiency of clock gating by Power Compiler rests on the shoulders of designers. Power Compiler forces designers to manually derive the gating signals from complex RTL designs for additional power reduction. Manual extraction of the signals is very time consuming, because VLIW ASIP contains several hundred pipeline registers; it is not suitable for design space exploration. Automated extraction of the gating signals is strongly required.

A clock gating method based on finite state machines [19] [23] extracts the gating signals of registers by analyzing the finite state machines. To use this method, the circuit must obviously contain the finite state machines and be feedback-free pipelines. Since the finite state machines are not suitable for pipeline processors, it cannot be applied to VLIW ASIP generation.

A clock gating method based on the Observability Don't Care (ODC) [21] can derive the gating signals of registers by ODC calculation. At present, this approach appears to be the most powerful clock gating method because of its high scalability and applicability. Unfortunately, ODC calculation takes too long time to completely apply such large-scale circuitry as VLIW ASIPs. To complete the calculation in practical time, a calculation time limit is introduced in the method. In addition, ODC-based clock gating extraction causes an enormous area overhead due to duplicating the circuits to create gating signals. A more suitable extraction method is still required to calculate the minimum execution conditions for VLIW ASIPs.

2.4 Urodynamic Monitoring Systems

The pressure monitoring systems for AUM must be small for less-invasive measurement, and the system must not make patients feel pain, or even discomfort.

To conduct pressure measurement in AUM, several pressure sensing systems using transurethral catheters have been proposed [24–27], Although these systems can be used in home, the results are still supposed to contain artifacts because the catheters adversely affect the test results [28] and limit patient's mobility. To improve plausibility of the recorded data, catheters should be removed for less-invasive pressure measurement in AUM.

Several pressure sensing systems which do not use catheters have been proposed [29–31].

The system in [29, 30] implants a capsule in the bladder and transmits data. Although the system requires incision, it could be used in the less invasive tests because the capsule is sufficiently small to be inserted in the bladder through the urethra. However, the system cannot simultaneously measure multiple pressure in the body because the system does not support multiple wireless communication. To correctly perform AUM, pressure data at multiple locations must be measured at the same time. Therefore, these systems are not enough to use pressure sensing for AUM.

The balloon-type remote bladder pressure sensor reported in [31] can measure and transmit the bladder pressure every five minutes. However, the balloon is difficult to employ in AUM because of its size, power consumption, time resolution, and wireless functionality:

- Since diameter of the balloon is 25 mm, invasive incision is supposed to be necessary to implant the balloon.
- Collecting data every five minutes is too coarse to capture the rapid transition of pressure.
- Unidirectional wireless communication is inconvenience for the quality control, communication control for the multiple capsules, and research and development of the capsules.

Especially, the size of the balloon becomes an big obstacle for the less-invasive system. The large size of the balloon is due to multiple chips on a PCB. The multiple chips needs large implementation area as well as many batteries for high power consumption. Accordingly, the most important technical issue to downsize the capsule is to integrate almost system in one

chip. A small-area and low-power SoC is strongly required to design such miniature capsules for AUM.

Chapter 3

Low Power VLIW ASIP Generation Method

This chapter describes a VLIW ASIP generation method to generate low-power VLIW ASIPs by automatically extracting minimum execution conditions for clock gating. The proposed method extracts the conditions of pipeline registers based on the forward approach using MOD [7], which specifies the high-level architecture of a VLIW ASIP. By employing the MODs, the proposed method can extract the minimum execution conditions of the pipeline registers through the VLIW ASIP generation processes, analyzing neither RTL descriptions nor netlists.

3.1 Basic of VLIW ASIP Generation

A VLIW ASIP is generated based on MODs and a VLIW ASIP model [18] [8]. In this section, MODs, the VLIW ASIP model, and the VLIW ASIP generation flow are described.

3.1.1 Micro-Operation Description

The architecture of a VLIW ASIP is specified with MODs. An MOD is identified as a pair of an operation ope and a resource group rg: $m = (ope, rg)$. A set of all MODs specified by designers are represented by M. The MOD specifies the architecture of operation ope that is executed on resource group rg. An operation is a minimum executable unit such as an

```
wire [31:0] reg1;
wire [31:0] reg2;
wire [31:0] sum;

stage 1: {
  wire [31:0] cur_pc;
  wire [31:0] inst_word;

  cur_pc     = PC.read();
  inst_word = IMEM.read(cur_pc);
  IR.write(inst_word);
};

stage 2: {
  reg1 = GPR.read0(rs);
  reg2 = GPR.read1(rt);
};

stage 3: {
  sum = ADD0.add(reg1, reg2);
};

stage 4: {
};

stage 5: {
  GPR.write(rd, sum);
};
```

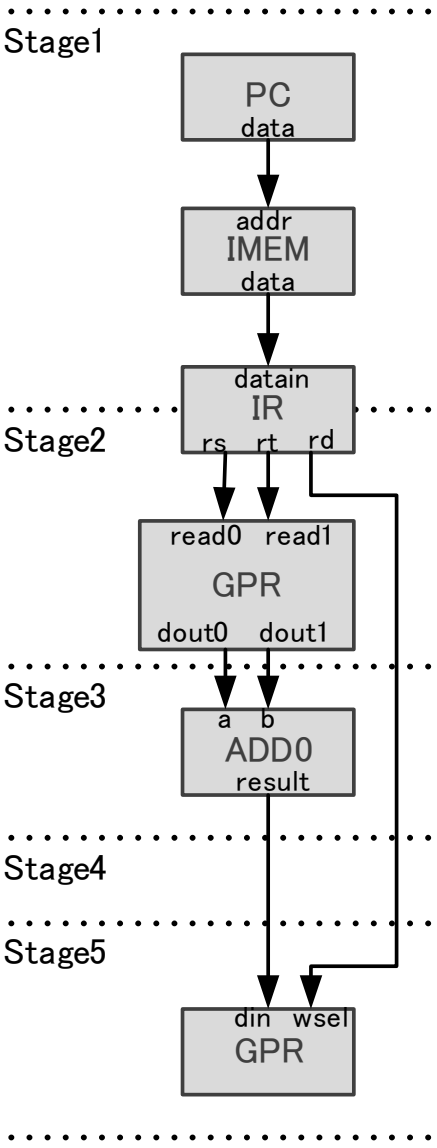

Figure 3.1: MOD of ADD on RG1 (ADD, RG1).

arithmetic operation, and a resource group is a set of hardware resources necessary for the operation. The architecture specified by an MOD can be converted to a Resource Connection Graph (RCG), which represents the data path of the corresponding operation in the form of the connections of resource ports.

Figure 3.1 is the example of an arithmetic addition described as the MOD of operation ADD on resource group RG1 = {PC, IMEM, IR, GPR, ADD0}, where PC, IMEM, IR, GPR, and ADD0 are hardware resources. In Fig.3.1, the left side description is the MOD of ADD on RG1, and the right side diagram is the corresponding RCG. In Fig.3.1, the two pieces of data, *reg1* and *reg2*, from register file GPR at stage 2 are sent to adder ADD0 at stage 3, and then the output is stored in the GPR at stage 5. Signals *rs*, *rt*, and *rd* stand for register indexes from instruction register IR. IMEM is a memory accessing unit that fetches VLIW instructions indicated by the address data from the program counter PC. In this way, the MOD contains not only connection information, but also such high-level architecture information as resource function and purpose.

3.1.2 VLIW ASIP Dispatching Model

Describing the dispatching behavior is the main concern of the VLIW ASIP specification. To handle this issue, the dispatching pattern of the VLIW ASIP is modeled as following three concepts: a slot, an operation group, and a resource group. Designers can specify the dispatching rule of the VLIW ASIP by slots, operation groups, resource groups, and their relations.

Slots are parallel dispatching units of the VLIW ASIP. One operation can be dispatched from a slot in one clock cycle.

Let $slot\_num$ be the number of slots, O be a set of all operations, and $ope_n \in O$ be an operation dispatched from n-th slot, a VLIW instruction $inst$ is composed of multiple operations:

$$inst = (ope_1, ope_2, ope_3, ..., ope_{slot\_num}). \tag{3.1}$$

Note that single-scalar ASIP corresponds to a one slot VLIW ASIP; our method can also deal with both single-scalar and VLIW ASIPs.

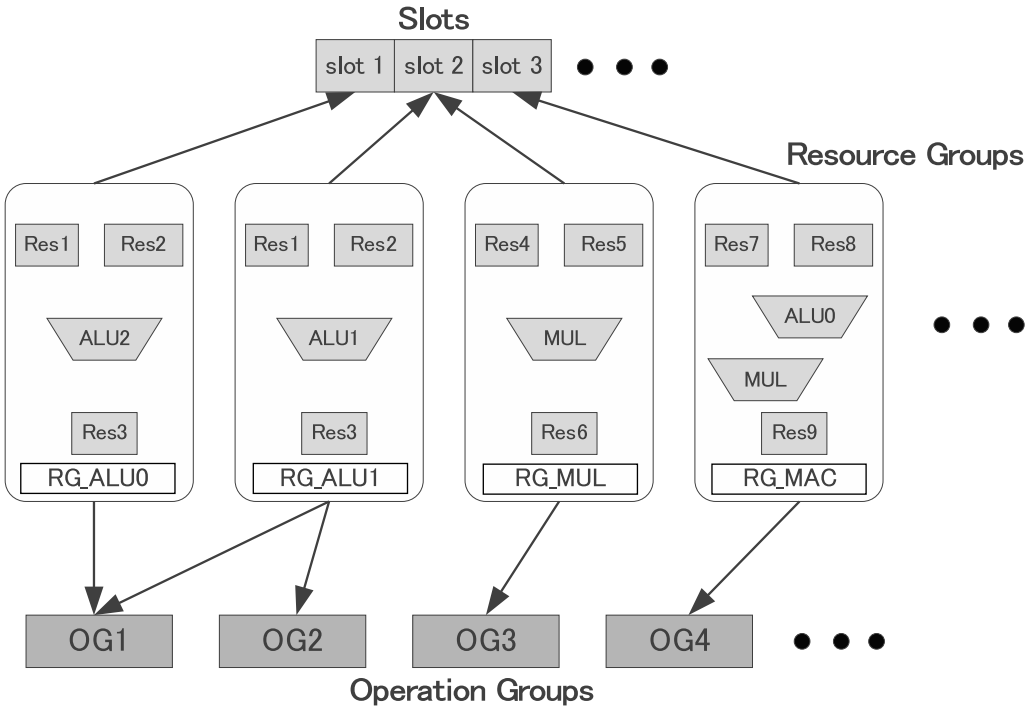

Figure 3.2: VLIW ASIP model.

Let RG be a set of all resource groups and R is a set of all resources, resource group $rg \in RG$ is a set of hardware resources: $rg \subseteq R$.

The relation between slots and resource groups: RSR is

$$RSR = \{(s, rg) \,|\, 1 \leq s \leq slot\_num, rg \in RG\}. \tag{3.2}$$

Pair (s, rg) represents that the operations dispatched from s-th slot are executed on rg.

Let OG be a set of all operation groups, an operation group $og \in OG$ is a set of operations: $og \subseteq O$.

The relation between a resource group and an operation group: RRO is

$$RRO = \{(rg, og) \,|\, rg \in RG, og \in OG\}. \tag{3.3}$$

Pair $(rg, og) \in RRO$ indicates that the operations categorized in og can be performed on rg.

As specification, the number of slot $slot\_num$, operations O, resources R, resource groups

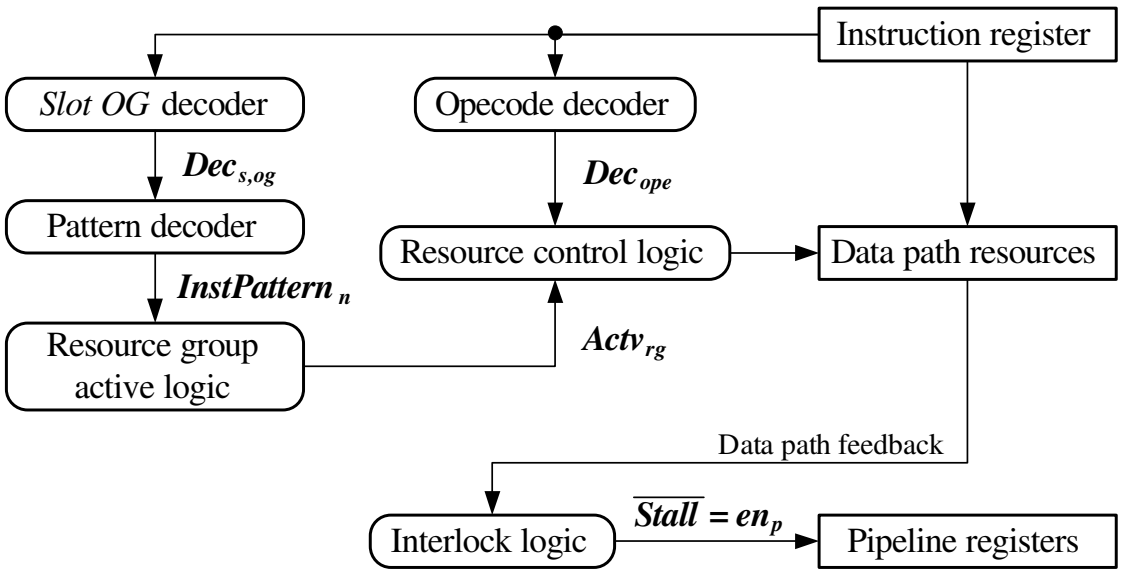

Figure 3.3: Decoder model of VLIW ASIPs.

RG, operation groups OG, the relation between slots and resource groups RSR, and the relation between a resource group and an operation group RRO are given by designers.

Figure 3.2 illustrates a model of a VLIW ASIP. In Fig.3.2, the operations categorized in operation group OG1, which can be performed on resource groups RG_ALU0 and RG_ALU1, can be simultaneously issued from slots 1 and 2.

3.1.3 VLIW ASIP Controller Model

The decoder model of VLIW ASIPs is shown in Fig.3.3. To calculate the control logic for data path resources, Dec_{ope} and $Actv_{rg}$ are needed.

Dec_{ope} of operation ope is decode logic on opecode. Dec_{ope} is calculated as

$$Dec_{ope}(inst) = \begin{cases} true & \text{if } \exists ope_n \in inst, \text{code}_{ope} = \text{code}_{ope_n} \\ false & \text{otherwise}, \end{cases} \tag{3.4}$$

where code_{ope} is the function that returns the opecode of ope and VLIW instruction $inst$ is stored in the instruction register.

To dispatch operations to appropriate resource groups, the controller calculates $Dec_{s,og}$, $InstPattern_n$, and $Actv_{rg}$. $Dec_{s,og}$ is the decode logic indicating that the operation dispatched from s-th slot belongs to $og \in OG$:

$$Dec_{s,og}(inst) = \bigvee_{\substack{(s,rg) \in RSR \\ ope \in og}} Dec_{ope}(inst) \wedge Exist(rg, ope), \tag{3.5}$$

$$Exist(rg, ope) = \begin{cases} true & \text{if } \exists (rg, og) \in RRO,\ ope \in og \\ false & \text{otherwise.} \end{cases} \tag{3.6}$$

In order to activate the appropriate resource groups for the dispatched instruction, pattern detecting logic $InstPattern_n(inst)$ is calculated:

$$InstPattern_n(inst) = \bigwedge_{\substack{T_{IDP_n} \in T_{IDP} \\ (s,rg,og) \in T_{IDP_n}}} Dec_{s,og}(inst), \tag{3.7}$$

where T_{IDP} is the table of instruction dispatching patterns calculated by the algorithm proposed by Kobayashi, et al. [8] [18], and T_{IDP_n} is the n-th entry in T_{IDP}. Dispatching pattern T_{IDP_n} can be described as a set of (s, rg, og) which indicates that the operations in og can be dispatched from s-th slot and executed on rg.

An example of T_{IDP} is shown in Table 3.1. $InstPattern_n(inst)$ indicates that the pattern of dispatched instruction $inst$. For instance, following Table 3.1, when $InstPattern_2(inst)$ is true, the pattern of $inst$ is $(OG2, OG2, OG2, OG2)$ and the corresponding resource groups which execute $inst$ are $(RG5, RG2, RG3, RG6)$.

$Actv_{rg}$ is an activation logic of resource group rg:

$$Actv_{rg}(inst) = \bigvee_{\substack{T_{IDP_n} \in T_{IDP} \\ (s,rg,og) \in T_{IDP_n}}} InstPattern_n(inst). \tag{3.8}$$

Table 3.1: Example of T_{IDP}

n	1st slot	2nd slot	3rd slot	4th slot
1	OG1	OG2	OG3	OG4
	RG1	RG2	RG3	RG4
2	OG2	OG2	OG2	OG2
	RG5	RG2	RG3	RG6
3	OG1	OG1	OG3	OG4
	RG1	RG7	RG8	RG4

According to the pattern of VLIW instructions, the assignments of operations to appropriate resource group are controlled by $Actv_{rg}$.

Resource control logic is calculated by using Dec_{ope} and $Actv_{rg}$. On the other hand, the control logic for pipeline registers is calculated only with feed back signals from data path. However, such control incurs unnecessary activation of pipeline registers. As a result, the pipeline registers dissipate unnecessary power.

3.1.4 VLIW ASIP Generation Flow

VLIW ASIP generation consists of two parts. The first part is data path construction that consists of four procedures: RCG conversion, RCG merging, signal conflict resolution, and pipelining. First, RCGs are converted from MODs. Second, all RCGs are combined by RCG merging to construct a prototype of the data path. Third, multiplexers are inserted, and finally, pipeline registers are inserted into appropriate locations. At each procedure, the generation method retrieves execution conditions that are used to generate steering signals and resource control signals in the second part. The second part is controller construction. The control signals for the resources in the VLIW ASIP are generated using the execution conditions obtained in the previous data path construction part.

3.2 Low-power VLIW ASIP Generation

In this section, the low-power VLIW ASIP generation method is proposed. First, the insertion of gating circuits is discussed, then, the extraction of minimum execution conditions is proposed.

3.2.1 Insertion of Gating Circuits

Due to the power overhead of the gating circuit, inserting one gating circuit before a few flip flops is ineffective; to increase the clock gating impact, one gating circuit must be shared by as many flip flops as possible. Such shared gating circuits reduce not only the power overhead of the gating circuits but also the power consumption of the clock trees at the same time. In clock tree synthesis, the shared gating circuit can be placed on the upper level of the clock tree. The proposed method also follows this basic insertion strategy.

Additionally, as mentioned in [10] and [32], selecting gating circuit schemes is crucial to increase circuit stability and to decrease implementation overhead. As shown in Fig. 3.4, the gate based (a) and flip flop based (b) schemes are mainly used. Gate based scheme is low overhead, but cannot stop glitches from gating signals, and the flip flop based scheme has overhead of gating signal latch, but can stop glitches. Despite its area and power overhead, the proposed method adopts the flip flop based gating scheme because it can block glitch noises that cause incorrect register activation.

3.2.2 Extraction of Minimum Execution Conditions

In this section, a formal extraction method of minimum execution conditions of the pipeline registers is proposed in the following four data path construction procedures.

3.2.2.1 RCG conversion

Resource Connection Graphs (RCGs) are generated from MODs. An RCG is represented by directed graph $G_m = (R_m, E_m)$ where $m = (ope, rg)$ is an identifier of an MOD, R_m is a

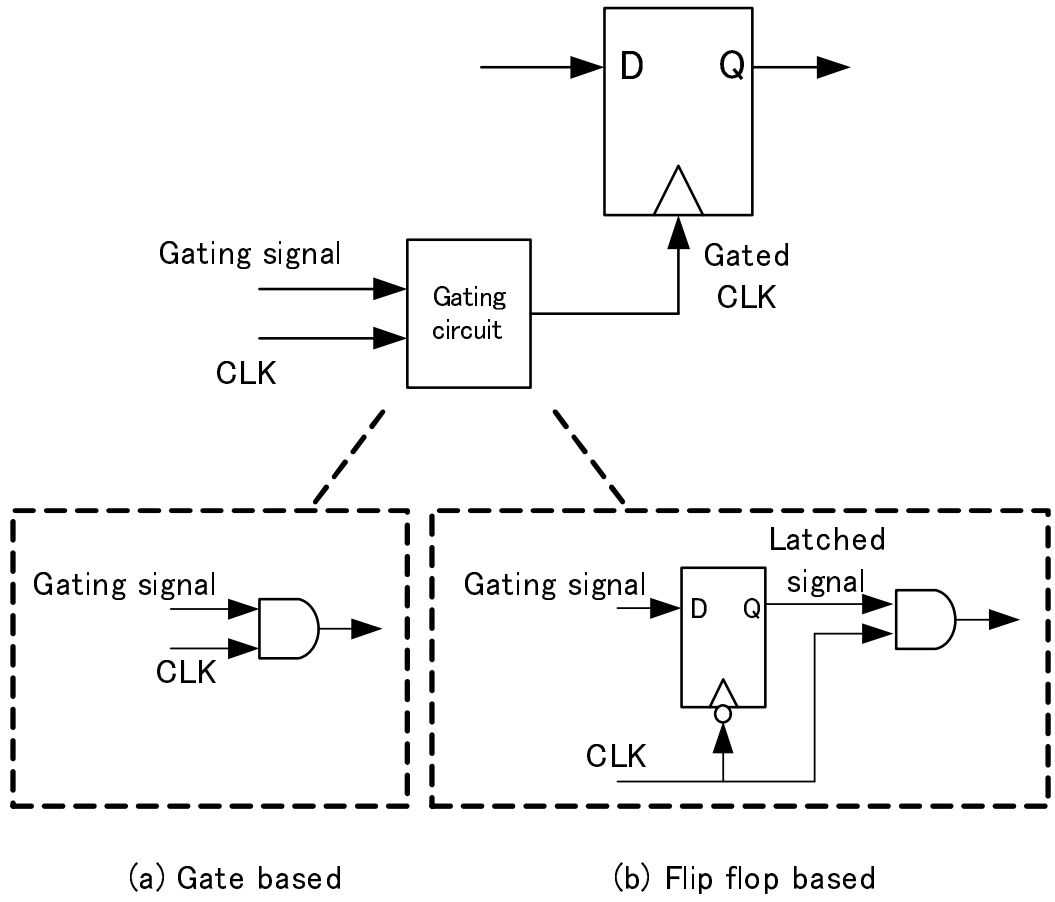

Figure 3.4: Clock gating schemes.

resources used by m, $E_m = \{(o,i)|o,i \in P\}$ is a set of data transfers used by m, pair (o,i) represents a data transfer from output port o to input port i, and P is a set of all resource ports.

A set of execution conditions $Cond_e$ for each data transfer $e \in E_m$ is retrieved in the RCG extraction. $Cond_e$ is described as

$$Cond_{e \in E_m} = \{(ope, rg) \,|\, m = (ope, rg)\}. \tag{3.9}$$

$Cond_e$ denotes that data transfer e is executed when operation ope on resource group rg is dispatched.

Figure 3.5 shows an example of extracted RCGs. For clarity, the ports of the RCGs and Stage1 are omitted. In the example, the three RCGs correspond to ADD on RG_1, SUB on RG_2, and $ANDI$ on RG_3, respectively. GPR, IR, EXT, $ALU1$, and $ALU2$ are the resources,

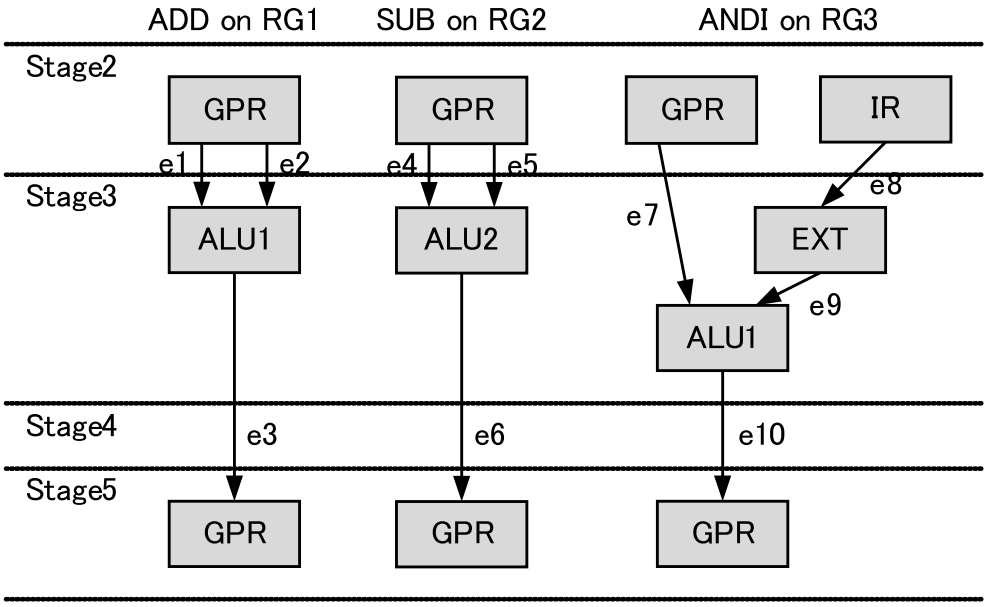

Figure 3.5: Converted RCGs.

and the edges labeled $e1$ to $e10$ are the data transfers. Here, sets of the conditions for the data transfers in Fig.3.5 are retrieved in the form of Eq.(3.9) as follows:

$$Cond_{e1} = \{(ADD, RG_1)\}, \qquad Cond_{e2} = \{(ADD, RG_1)\},$$
$$Cond_{e3} = \{(ADD, RG_1)\}, \qquad Cond_{e4} = \{(SUB, RG_2)\},$$
$$Cond_{e5} = \{(SUB, RG_2)\}, \qquad Cond_{e6} = \{(SUB, RG_2)\},$$
$$Cond_{e7} = \{(ANDI, RG_3)\}, \qquad Cond_{e8} = \{(ANDI, RG_3)\},$$
$$Cond_{e9} = \{(ANDI, RG_3)\}, \qquad Cond_{e10} = \{(ANDI, RG_3)\}.$$

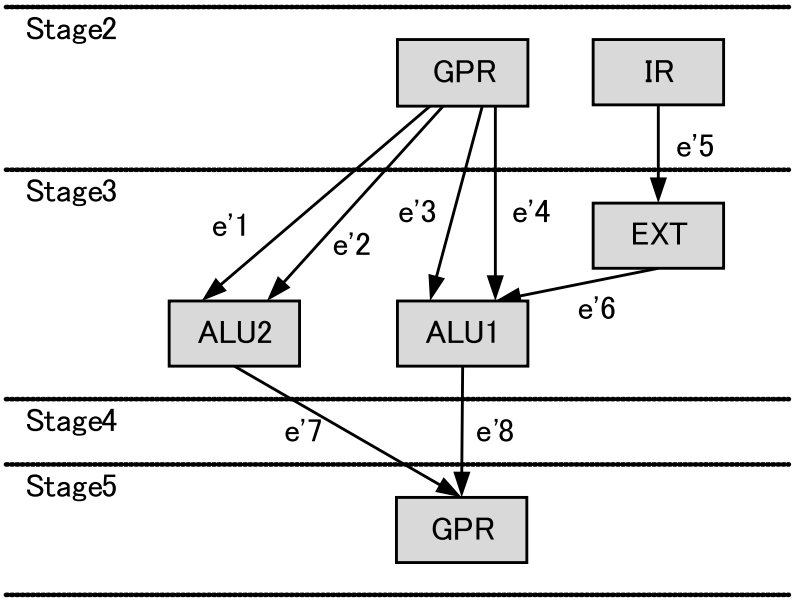

Figure 3.6: Unified RCG.

3.2.2.2 RCG Merging

After all MODs are converted, they are merged into a unified RCG $G' = (R', E')$. R' and E' are calculated as follows:

$$R' = \bigcup_{\forall m \in M} R_m$$
$$E' = \bigcup_{\forall m \in M} E_m,$$

where M is a set of all identifiers of MODs consisting of the VLIW ASIP. Since data transfers E_m are merged into E', conditions $Cond_e$ of all extracted RCGs are also merged. The new conditions of data transfers $Cond'_e$ are calculated as

$$Cond'_{e' \in E'} = \bigcup_{\forall m \in M, \forall e \in E_m, e' = e} Cond_e. \tag{3.10}$$

The three operations in the example in Fig.3.5 are merged into the unified RCG illustrated as Fig.3.6. The conditions of data transfers $e'1$ to $e'8$ are newly calculated as follows:

$$Cond'_{e'1} = \{(SUB, RG_2)\}, \qquad Cond'_{e'2} = \{(SUB, RG_2)\},$$
$$Cond'_{e'3} = \{(ADD, RG_1), (ANDI, RG_3)\}, \qquad Cond'_{e'4} = \{(ADD, RG_1)\},$$
$$Cond'_{e'5} = \{(ANDI, RG_3)\}, \qquad Cond'_{e'6} = \{(ANDI, RG_3)\},$$
$$Cond'_{e'7} = \{(SUB, RG_2)\},$$
$$Cond'_{e'8} = \{(ADD, RG_1), (ANDI, RG_3)\}.$$

The unified RCG is the prototype of the data path. Then multiplexers and pipeline registers are inserted in the following procedures.

3.2.2.3 Signal Conflict Resolution

Multiplexers are inserted in order to resolve signal conflicts occurring in unified RCG G' such as $e'7$ and $e'8$ in Fig.3.6. The RCG in which the multiplexers are inserted is described as $G'' = (R'', E'')$.

A multiplexer is inserted before an input port, which is the multiple destination of some data transfers. Here, a set of edges conflicting at identical input port i: ECI_i is described as

$$ECI_i = \{(o', i') \mid o' \in P,\ i' \in P,\ (o', i') \in E',\ i' = i\}. \tag{3.11}$$

Since the multiplexers are inserted, the signal connections change. The conditions of data transfers E'' should be calculated, too. A set of the conditions of data transfer $e'' \in E''$ is described as follows:

$$Cond''_{e''} = \begin{cases} \bigcup_{\forall e' \in ECI_{dest_{e''}}} Cond'_{e' \in E'} & \text{if } e'' \in E_{MR}, \\ Cond'_{e' \in E'} & \\ \qquad \text{such that } \ dest_{e'} = dest_{e''} & \text{if } e'' \in E_{RM}, \\ Cond'_{e' \in E'} \ \text{ such that } \ e' = e'' & \text{otherwise}, \end{cases} \tag{3.12}$$

where $dest_e$ is a destination port of data transfer e, E_{MR} is a set of data transfers that connect a multiplexer to a resource, and E_{RM} is a set of data transfers selected by a multiplexer.

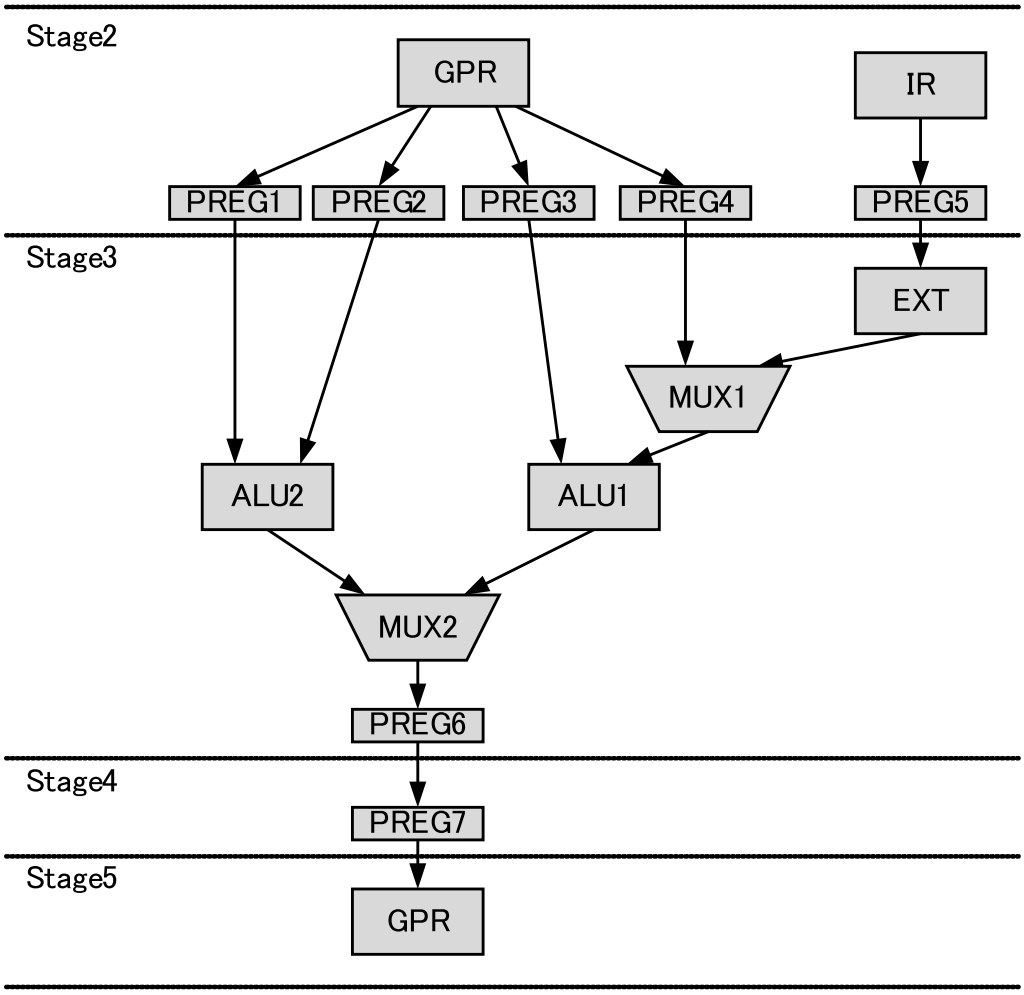

Figure 3.7: Constructed data path.

3.2.2.4 Pipelining

For pipelining G'', pipeline registers are required for data transfers that cross pipeline stage boundaries. The RCG in which the pipeline registers are inserted is described as G'''. The location of a pipeline register $p = (o, n)$ can be described as a pair of output port o and stage number n where the pipeline register is placed because one pipeline register is shared by several data transfers from o. A set of edges connected to identical output port o: ECO_o is

$$ECO_o = \{(o'', i'') \mid o'', i'' \in P,\ (o'', i'') \in E'',\ o'' = o\}. \tag{3.13}$$

In ECO_o, a set of data transfers crossing stage boundaries $ECOX_o$ is

$$ECOX_o = \{(o', i') \mid stage_{o'} < stage_{i'}, \text{for all } (o', i') \in ECO_o\}, \tag{3.14}$$

where $stage_x$ represents the pipeline stage number to which port x belongs.

In G'', a set of data transfers crossing stage boundaries is

$$E''_{CROSS} = \bigcup_{o \in P_{out}} ECOX_o. \tag{3.15}$$

Finally, a set of pipeline registers $PREG$ is obtained as

$$PREG = \bigcup_{(o,i) \in E''_{CROSS}} \{p \,|\, p = (o, n), stage_o \leq n < stage_i\}. \tag{3.16}$$

Figure 3.7 depicts the data path after inserting multiplexers MUX and pipeline registers PREG in the unified RCG in Fig.3.6. Pipeline registers $PREG1$ to $PREG7$ are inserted in the appropriate points.

Here, the execution conditions of the inserted pipeline registers are calculated. The execution conditions for the pipeline registers are derived from the conditions of the data transfers calculated by Eq.(3.12). Since the pipeline registers are shared by some data transfers, a set of execution conditions EC_p of pipeline register $p = (o, n)$ is calculated as

$$EC_p = \bigcup_{p=(o,n),\ (o,i) \in ECOX_o} Cond''_{(o,i)}. \tag{3.17}$$

EC_p is a set of $m = (ope, rg)$ such that ope dispatched to rg requires p for execution. Let $G'''_m \subseteq G'''$ be a necessary data path for executing m and $PREG_m \in G'''_m$ be a set of the pipeline registers that is required for execution by ope dispatched to rg, extracted EC_p can be also described as:

$$EC_p = \{m \,|\, m \in M, p \in PREG_m\}. \tag{3.18}$$

For the pipeline registers in Fig.3.7, execution conditions EC_p are calculated as follows:

$$EC_{PREG1} = \{(SUB, RG_2)\},$$

$$EC_{PREG2} = \{(SUB, RG_2)\},$$

$$EC_{PREG3} = \{(ADD, RG_1), (ANDI, RG_3)\},$$

$$EC_{PREG4} = \{(ADD, RG_1)\},$$

$$EC_{PREG5} = \{(ANDI, RG_3)\},$$

$$EC_{PREG6} = \{(ADD, RG_1), (SUB, RG_2), (ANDI, RG_3)\},$$

$$EC_{PREG7} = \{(ADD, RG_1), (SUB, RG_2), (ANDI, RG_3)\}.$$

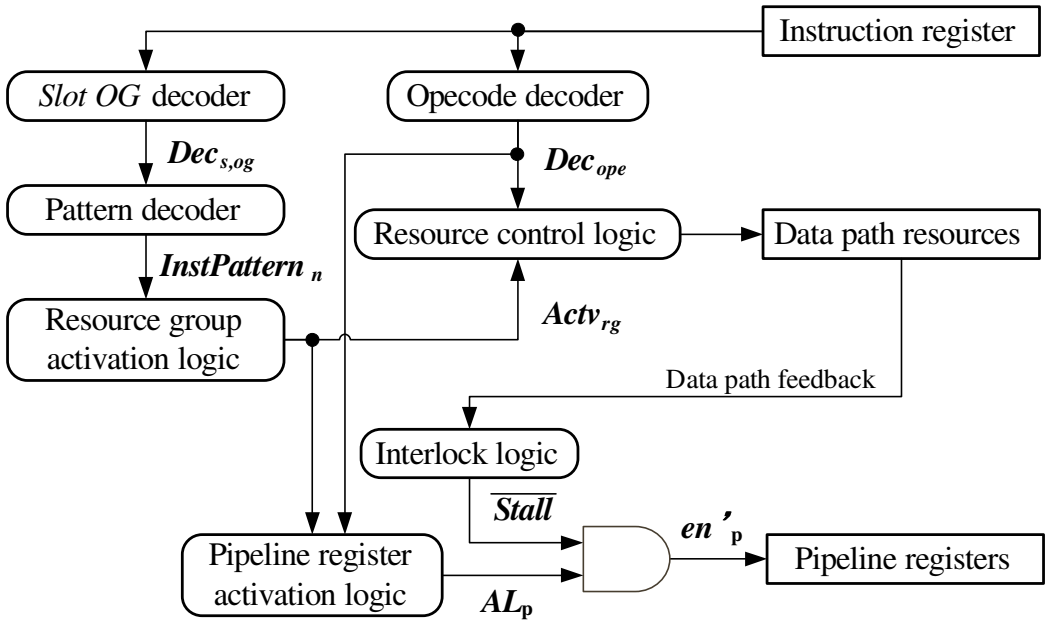

Figure 3.8: Modified decoder model.

Thus the execution conditions for pipeline registers are calculated. In the next section, creating gating signals for pipeline registers is discussed.

3.2.3 Generating Gating Signals with Minimum Execution Conditions

To suppress the unnecessary activations of pipeline register p, a new control signal en'_p with additional logic AL_p is introduced:

$$en'_p(inst) = \overline{stall_{stage_p}} \wedge AL_p(inst), \tag{3.19}$$

$$AL_p(inst) = \bigvee_{\substack{m \in EC_p \\ (ope, rg) = m}} Dec_{ope}(inst) \wedge Actv_{rg}(inst), \tag{3.20}$$

where EC_p is the derived execution condition in Eq.(3.18). The modified decoder model of VLIW ASIP with AL_p is shown in Fig.3.8.

Here, correct execution is defined as that the execution result of an operation on the VLIW ASIP generated by the proposed method is same as that by the traditional method.

Theorem 1. *VLIW ASIPs clock-gated with en'_p guarantee correct execution.*

Proof. The new logic en'_p does not prevent the data flow in G'''_m because every pipeline register $p \in PREG_m$ is activated when m is dispatched. Therefore, the execution result of the VLIW ASIP by the proposed method is same as that by the traditional method, i.e., correct execution is guaranteed. □

Theorem 2. *Let a minimum execution condition be an execution condition such that removing one element from the execution condition causes incorrect execution of the generated VLIW ASIP, execution condition Eq.(3.18) is minimum.*

Proof. Consider that any $m \in EC_p$ is removed, p is not activated when m is dispatched. However, this removing causes incorrect execution because $p \in PREG_m$. Hence, EC_p in Eq.(3.18) is the minimum execution condition. □

Using execution condition $EC'_p \supseteq EC_p$ for Eq.(3.20) can correctly execute operations. Therefore, the traditional VLIW ASIP generation method constantly deals with EC'_p as $EC'_p = M$ for all p. This generation strategy results in reducing area and delay because AL_p can be constantly treated as $true$.

Using Eq.(3.19), the gating signals of the pipeline registers in the VLIW data path illustrated in Fig.3.7 can be calculated as follows:

$$
\begin{aligned}
en'_{PREG1}(inst) &= \overline{stall_2} \wedge \{Dec_{SUB}(inst) \wedge Actv_{RG_2}(inst)\}, \\
en'_{PREG2}(inst) &= \overline{stall_2} \wedge \{Dec_{SUB}(inst) \wedge Actv_{RG_2}(inst)\}, \\
en'_{PREG3}(inst) &= \overline{stall_2} \wedge \{Dec_{ADD}(inst) \wedge Actv_{RG_1}(inst) \\
&\qquad \vee Dec_{ANDI}(inst) \wedge Actv_{RG_3}(inst)\}, \\
en'_{PREG4}(inst) &= \overline{stall_2} \wedge \{Dec_{ADD}(inst) \wedge Actv_{RG_1}(inst)\}, \\
en'_{PREG5}(inst) &= \overline{stall_2} \wedge \{Dec_{ANDI}(inst) \wedge Actv_{RG_3}(inst)\}, \\
en'_{PREG6}(inst) &= \overline{stall_3} \wedge \{Dec_{ADD}(inst) \wedge Actv_{RG_1}(inst) \\
&\qquad \vee Dec_{SUB}(inst) \wedge Actv_{RG_2}(inst) \\
&\qquad \vee Dec_{ANDI}(inst) \wedge Actv_{RG_3}(inst)\}. \\
en'_{PREG7}(inst) &= \overline{stall_4} \wedge \{Dec_{ADD}(inst) \wedge Actv_{RG_1}(inst) \\
&\qquad \vee Dec_{SUB}(inst) \wedge Actv_{RG_2}(inst) \\
&\qquad \vee Dec_{ANDI}(inst) \wedge Actv_{RG_3}(inst)\}.
\end{aligned}
$$

3.2.4 Generation of Scalar ASIPs

Scalar ASIPs correspond to the special case of VLIW ASIP: single slot, single resource group, and single operation group. Therefore, the proposed method can be easily applied to generation of scalar ASIPs.

3.3 Experiments

Two experiments were carried out using the integer subset of DLX [33] [34] to confirm the effectiveness of the proposed method.

For each experiment, the following three types of ASIPs were generated:

NCG: Not clock gated VLIW ASIPs by the traditional generation method

PC: VLIW ASIPs clock gated by Power Compiler

PM: VLIW ASIPs generated by our proposed method.

Note that clock gating was only applied to the pipeline registers in the data path of the generated VLIW ASIPs. The generated ASIPs were synthesized using Design Compiler under a minimizing area constraint and physically synthesized by IC Compiler using a $0.18\mu m$ CMOS technology library operating on 1.8 V.

Programs were randomly generated based on the appearance rate of each instruction. The appearance rates of the instructions in a compiled program are a much more dominant factor than the instruction sequence in the case of using clock gating. Therefore, the characteristics of the programs were modeled as appearance rate in this experiment. For instance, a low IPC program can be modeled as a high NOP appearance rate.

3.3.1 Evaluation of Hardware Variation

In the first experiment, both single-scalar and VLIW type ASIPs were generated. The single-scalar type ASIPs were extended with Multiply ACcumulate (MAC) instruction by varying the pipeline depth from two to seven stages. The VLIW type ASIPs were designed on a single-scalar ASIP of five stages and homogeneously expanded to two, four, and six slots. Note that the 2-slot processor contains 65 pipeline registers, the 4-slot processor 124, and the 6-slot processor 183; they are large-scale designs. In this experiments, the appearance ratios of the program are as follows: load/store are 30%, multiplication is 3.5%, division is 1%, branch/jump are 5%, and integer instruction is 60.5%. These rates were determined by reference to the analysis report of the compiled SPEC benchmarks for MIPS processors [35]. All cache access were assumed to hit in this experiment.

Note that applying Power Compiler to PM did not affect the circuits because Power Compiler does not insert clock gating into the already gated registers. All pipeline registers of PM are already clock gated by the proposed method.

The experimental results are shown in Table 3.2 and Table 3.3. The results show the area, delay, and power comparison of the generated ASIPs. Slot # and the pipeline depth are the number of parallel issues and the pipeline depth, respectively. Total power is the power con-

Table 3.2: Power comparison on single-scalar DLX, varying number of pipeline stages.

Pipeline depth	Type	Area [μm2]	Delay [ns]	Total power [μW/MHz]	Pipeline registers [μW/MHz]	Delta [%]	Clock tree [μW/MHz]	Skew [ps]
3	NCG	445255	20.09	169.1	17.4		32.8	46
	PC	440625	20.09	147.3	8.5	-51.3	30.7	88
	PM	441330	20.43	146.5	3.1	-78.2	27.8	137
4	NCG	446785	17.34	179.0	25.3		33.3	62
	PC	440632	17.50	163.7	16.1	-36.6	30.6	105
	PM	442388	15.57	153.3	6.6	-73.9	30.4	106
5	NCG	451319	17.33	185.8	31.5		34.9	67
	PC	443782	15.57	169.1	22.8	-27.6	32.2	84
	PM	447188	15.57	155.3	8.2	-73.9	29.8	109
6	NCG	512422	12.10	239.3	63.2		44.0	74
	PC	496831	12.06	209.7	48.4	-23.3	38.6	107
	PM	503537	12.06	174.4	12.1	-80.9	32.4	113
7	NCG	526083	12.10	250.1	68.0		44.9	68
	PC	510137	12.06	218.7	49.2	-27.6	41.4	96
	PM	517711	12.06	183.5	12.4	-81.7	34.6	104

Table 3.3: Power comparison on VLIW DLX, varying number of parallel issues.

Slot #	Type	Area [μm2]	Delay [ns]	Total power [μW/MHz]	Pipeline registers [μW/MHz]	Delta [%]	Clock tree [μW/MHz]	Skew [ps]
1	NCG	451319	17.33	185.8	31.5		34.9	67
	PC	443782	15.57	169.1	22.8	-27.6	32.2	84
	PM	447188	15.57	155.3	8.2	-74.0	29.8	109
2	NCG	651106	12.06	300.7	97.9		54.1	100
	PC	625762	12.06	262.1	80.3	-18.0	66.9	112
	PM	631341	12.06	193.5	11.2	-88.6	38.3	123
4	NCG	1074753	12.02	462.1	179.4		87.0	117
	PC	1029118	12.07	395.5	152.4	-15.1	81.4	140
	PM	1038249	12.07	265.8	20.8	-88.4	52.3	116
6	NCG	1596043	12.08	659.3	263.6		118.9	78
	PC	1530117	12.12	553.0	233.4	-11.5	110.5	114
	PM	1544843	12.14	350.9	35.1	-86.7	63.9	133

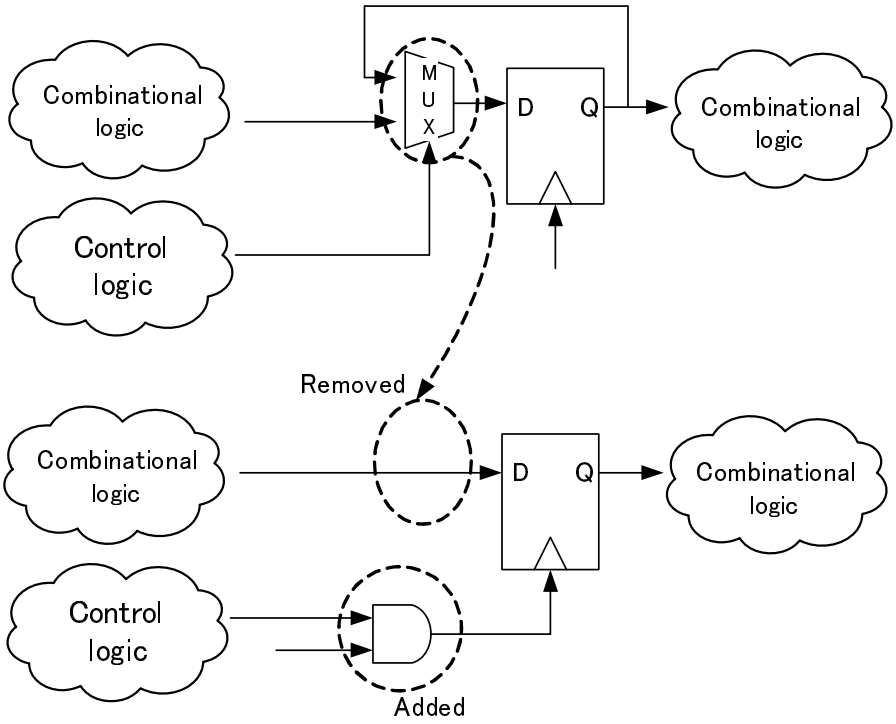

Figure 3.9: Area reduction by applying clock gating.

sumption of all circuits, and Pipeline registers stands for the power breakdown by the pipeline registers. Delta is the ratio of the power reduction compared to NCG. Clock tree is the power breakdown of the clock trees, and Skew is the global clock skew of the ASIPs.

As observed in Table 3.2, the power consumption of the pipeline registers of PM is reduced approximately 80% compared to NCG in every case, and PC is reduced approximately 35 %. In addition, the power consumption of every clock tree also decreased. The same trend is observed in Table 3.3. These results show that the proposed method shuts off more redundant clock supplies than the clock gated VLIW ASIPs by Power Compiler.

Area reduction (from NCG to the others) is confirmed because the multiplexers inside the registers are removed. As shown in Fig 3.9, multiplexers are removed and a clock gate is inserted. In general, an N-bit register needs N multiplexers, and once clock gating is applied, multiple multiplexers are replaced by one clock gate because the flip flops in a register shares one clock source. Therefore, implementation area is reduced when clock gating is applied.

Table 3.4: Power consumption of pipeline registers in 4-slot VLIW ASIP according to appearance rate.

Case	Int.	Multi Cycle	Load/ store	NOP	Ctrl.	Power [μW/MHz]		
	[%]	[%]	[%]	[%]	[%]	NCG	PC	PM
1	60.5	4.5	30	0	5	201.2	197.0	49.8
2	90	0	5	0	5	264.8	278.1	126.7
3	70	0	5	20	5	246.9	265.9	103.4
4	50	0	5	40	5	241.3	262.1	85.9
5	84	6	5	0	5	190.4	165.9	38.3

Area reduction is an innate advantage of clock gating. On the other hand, negligible area overheads occur owing to the implementation of the minimum execution conditions (from PC to PM). The area overhead reflects the increase of operators in Eq.(3.19). Compared to Eq.(2.1), Eq.(3.19) has more operators and terms to implement minimum execution conditions. Nevertheless, the overheads are small. This is because the extra terms in Eq.(3.19) also exist in the VLIW ASIPs generated by the traditional VLIW generation method; therefore they do not affect the overheads. The extra operators only result in small area overhead.

Critical delay overheads are confirmed to be negligible. Besides being an innate disadvantage of clock gating, clock skew increases in almost cases. Though the skew increases, the difference of skew between PC and PM is small. The skew results suggest that the proposed method has less extra effect on the clock skew.

For all the ASIPs, the overheads of the calculation time by the proposed method are within several seconds on a workstation operating on 3 GHz using 4 GB memory. This shows that the proposed method has little extra computational overhead compared to the traditional VLIW ASIP generation.

3.3.2 Evaluation of Software Variation

In the second experiment, to confirm the impact of various programs on the power consumption of the pipeline registers, the appearance rates of the integer, multi-cycle, load/store, No OPeration (NOP), and control operations were varied on the 4-slot VLIW ASIP. The multi-cycle operations include multiplication and division, which take 32 cycles to finish operation.

Table 3.4 shows the results of the five cases. The differences of power reduction are due to the differences of data path utilization in each program.

With respect to PC, the power consumption is increased from case 1 to 2 because no multi-cycle operations were issued. This is because PC cannot reduce power consumption when no multi-cycle operations are issued. In case 2, 3, and 4, the power overhead by the gating circuit is just accumulated from NCG to PC. The combination of the traditional VLIW generation method and Power Compiler worsens power consumption in such cases. By increasing multi-cycle operation in case 5, power consumption of both PC and PM are reduced because pipelines are stalled.

PM achieves substantial power reduction, as shown in Table 3.4. In cases 2, 3, and 4, power consumption decreased, although no multi-cycle operations were issued, showing that gating condition generated by the proposed method stopped unnecessary clock supplies while the pipeline was not stalled. Power consumption decreased while the NOP rate increased because NOP did not activate any data path modules.

The proposed method can reduce the power consumption of any programs because even a well optimized program contains the unused registers.

3.4 Conclusion

In this chapter, a low-power VLIW ASIP generation method is proposed. The proposed method automatically extracts the minimum execution conditions of the pipeline registers in the generated VLIW ASIPs and shuts off the excess clock supplies to the pipeline registers by clock gating. The experimental results showed that the power consumption of the pipeline registers in the VLIW ASIPs generated with the proposed method was reduced about 80% compared to the VLIW ASIPs that were not clock gated, and about 60% compared to the VLIW ASIPs that

were clock gated by Power Compiler with negligible delay and area overhead.

Chapter 4

Design of an SoC for Pressure Sensing Capsules in AUM

In this chapter, the design of pressure sensing capsules for AUM is described. The almost functions of the capsules are integrated in an SoC: Medical SOC (MeSOC). The MeSOC is designed with an ASIP: MeDical Instruction-set eXtension type I (MeDIX-I).

First, technical requirements for pressure sensing capsule are clarified, and the pressure sensing system and the capsule are described. Secondly, the design of the MeSOC is described. Then, the designs of the digital systems, including MeDIX-I, are described.

4.1 Requirements for Pressure Sensing Capsules

Many reports in urodynamics suggest the requirements for pressure sensing devices. The technical requirements for the instruments in AUM are summarized as follows:

- To record the detrusor pressure, the pressure data of the bladder and rectum must be measured simultaneously.
- For less invasive tests, the instruments must be sufficiently small to pass through the urethra, the outer diameter of which is about 6.5 mm.
- The capsule should continuously operate for at least 72 hours to enable the investigation of irregularly occurring symptoms [36] .

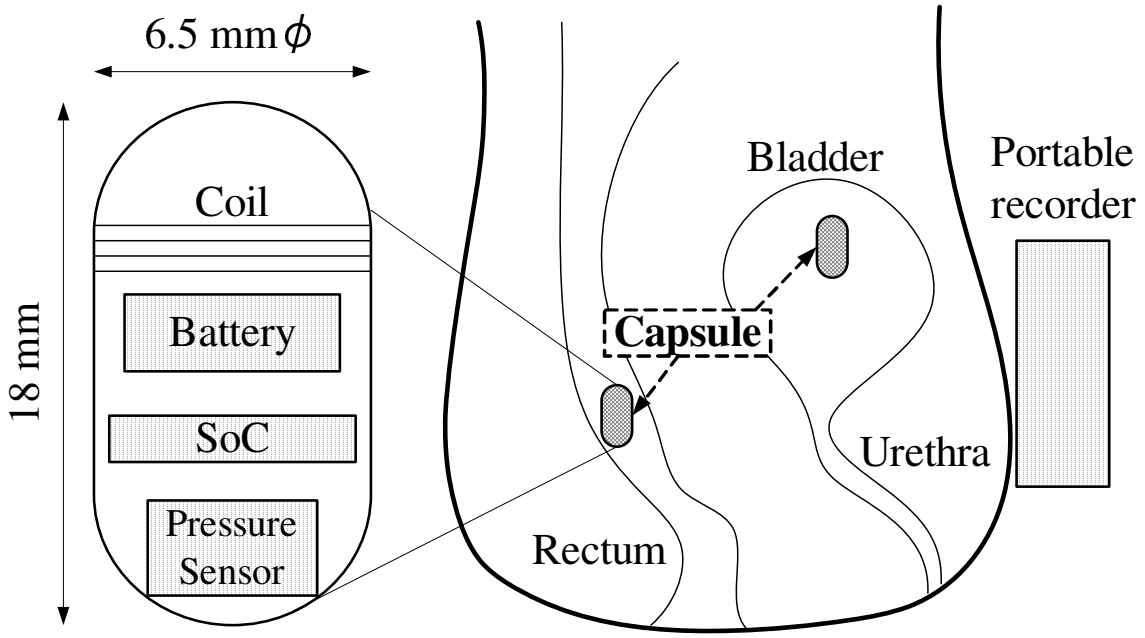

Figure 4.1: Pressure measurement system and capsules.

- To create urinary diary, the pressure data can be self-monitored or automatically recorded [37].
- According to the International Continence Society (ICS), a pressure range of 0-225 mmHg is necessary for correct measurement [38].
- For quality control, the instruments require calibration in the body before measurement starts [16].
- For the same reason, the minimum time resolution required for recording is 15 Hz to carry out cough tests [38]. Coughs should be recorded to ensure that abdominal and intravesical pressures respond equally [16].

4.2 System Overview

In order to satisfy the strict requirements shown in Section 4.1, the less-invasive pressure sensing system with tiny capsules is presented in this section. Figure 4.1 shows the biotelemetry

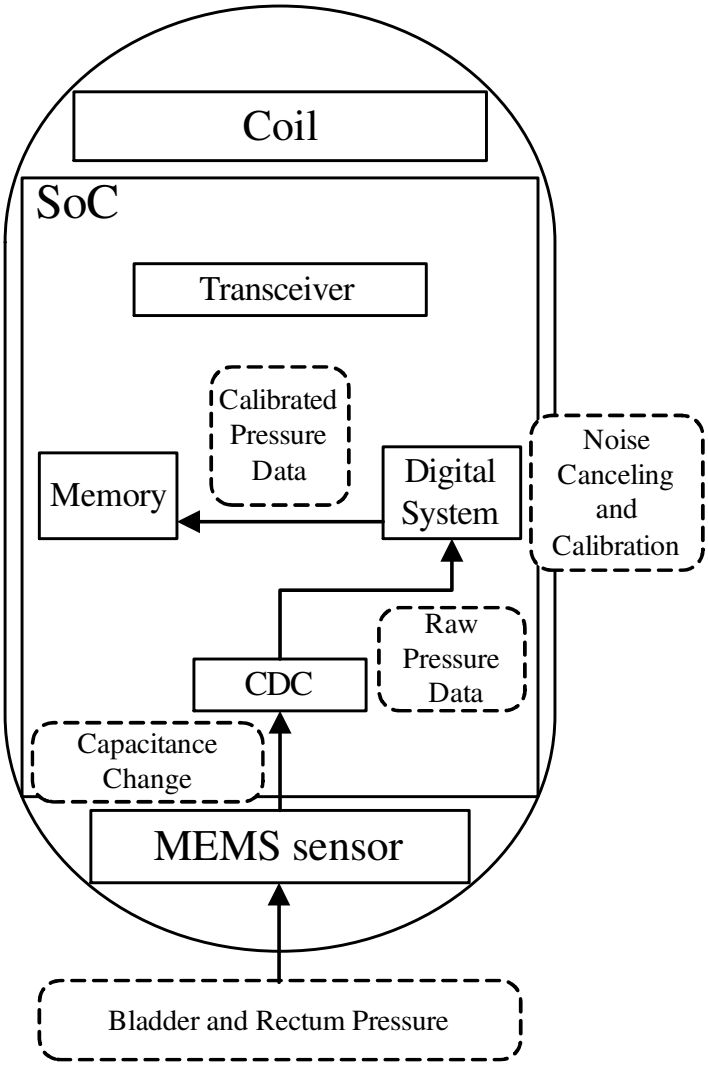

Figure 4.2: The work flow of pressure sensing.

system with two capsules. To record pressure data, the tiny airtight capsules are inserted into the bladder and rectum. The capsules communicate with a portable recorder attached on the body and transmit data as the patient goes about his/her daily life. The capsule consists of four components as shown in Fig. 4.1: a communication coil, a battery, a MEMS capacitive pressure sensor [39], and the MeSOC. The MeSOC integrates the digital converter for the MEMS sensor [40], which is called Capacitance-to-Digital Converter (CDC), a wireless transceiver [41], and a digital system based on the MeDIX-I.

Work flows of the capsule are shown in Fig. 4.2 and 4.3. As shown in Fig. 4.2, to perform

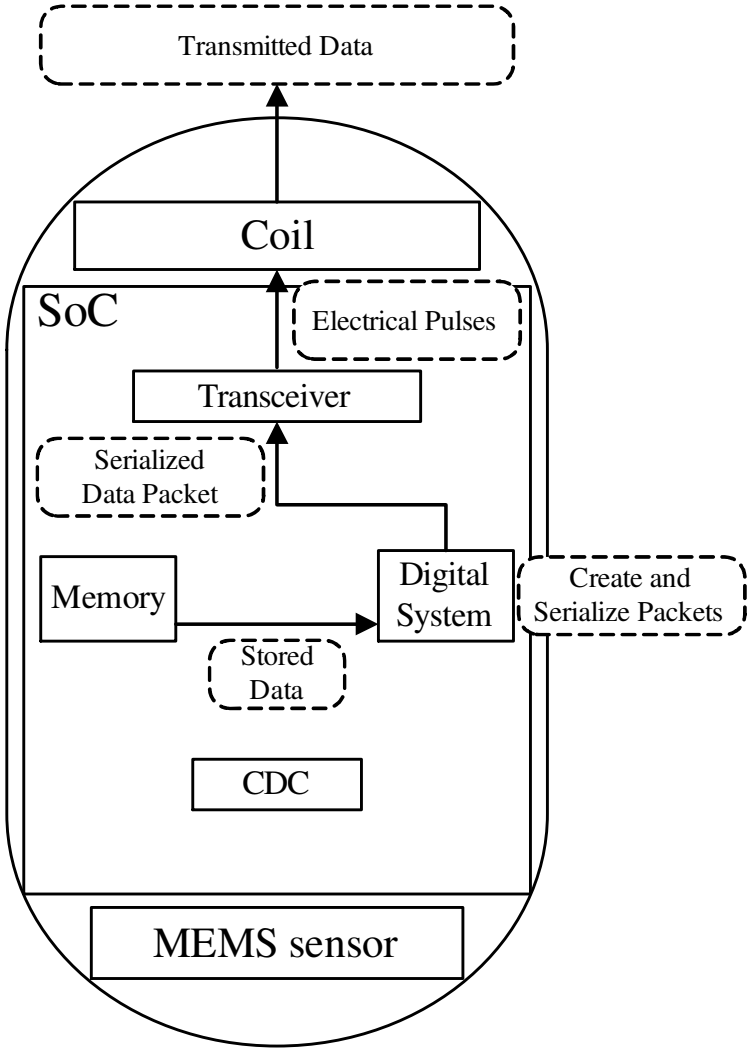

Figure 4.3: The work flow of wireless transmission.

pressure sensing, the MEMS sensor in the capsule measures the bladder and rectum pressure. The CDC in the SoC reads the MEMS capacitance and outputs the digital value of raw pressure data. Then the digital system calibrates the raw data and stores them in the memory.

As shown in Fig. 4.3, to perform wireless transmission, the digital system reads the stored pressure data and creates packets for wireless transmission. Then the digital system serializes the packets. The transceiver converts the serialized packets to electrical pulses, and the data are transmitted by using the coil in the capsule.

Since the CDC and transceiver have already been proposed, the main design challenges in this thesis are the integration of all components in a single chip and the design of the digital

system. First, all the components in the system must be integrated under the tight constraints of a chip size of no more than 2.5 x 2.5 mm^2 to fit the capsule, and of operation time at least 72 hours powered by a tiny single battery. Secondly, a low-power digital system with the MeDIX-I must be designed. Since the wireless protocol is complex and must be programmable for further development *in vivo*, the instruction extension for wireless communication of MeDIX-I is important. Additionally, the MeDIX-I contributes to the low-power design by means of special instructions for wireless communication and sleep, because the special instructions reduce the operation frequency and computation time of the MeSOC.

4.3 MeSOC Overview

In this section, design constraints, a task flow, and the power management system of the MeSOC are described.

4.3.1 Design Constraints

Strict design constraints should be cleared for the design of the digital system. The MeSOC size is limited to 2.5 x 2.5 mm^2. The energy capacity of the adopted battery (SR421SW) is 12 mAh at 1.55 V in ideal conditions, i.e., the battery provides 166.6 μA for 72 hours. Actually, our prior experiments on this battery indicate that this battery actually can provide up to 150 μA for 72 hours at 1.55 V. Assuming the analog system consumes 57 μA, power budget of the digital system is under 93 μA.

4.3.2 MeSOC Task and Architecture

The typical task of the MeSOC is shown in Fig. 4.4. First, the analog system in the MeSOC is initiated with wireless power supply. In wireless power mode, the frequency of the ring oscillator is adjusted at 968.571 kHz (1/14 of 13.56 MHz) using carrier wave. Owing to the lack of crystals, the source frequency of 968.571 kHz involves $\pm 1\%$ error and 0.1% jitter. After the digital system starts, target programs are downloaded. Then, the digital system switches the power source to the battery. This power-on mechanism allows users to turn on the capsule

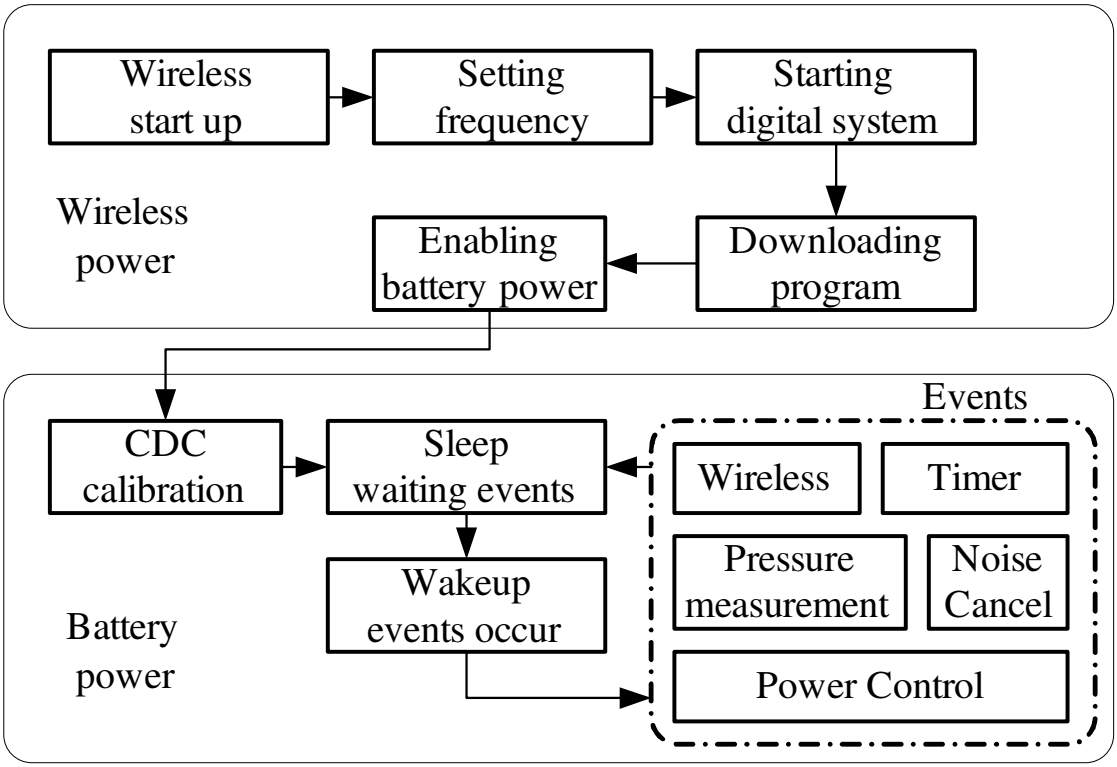

Figure 4.4: Flow chart of a typical MeSOC task.

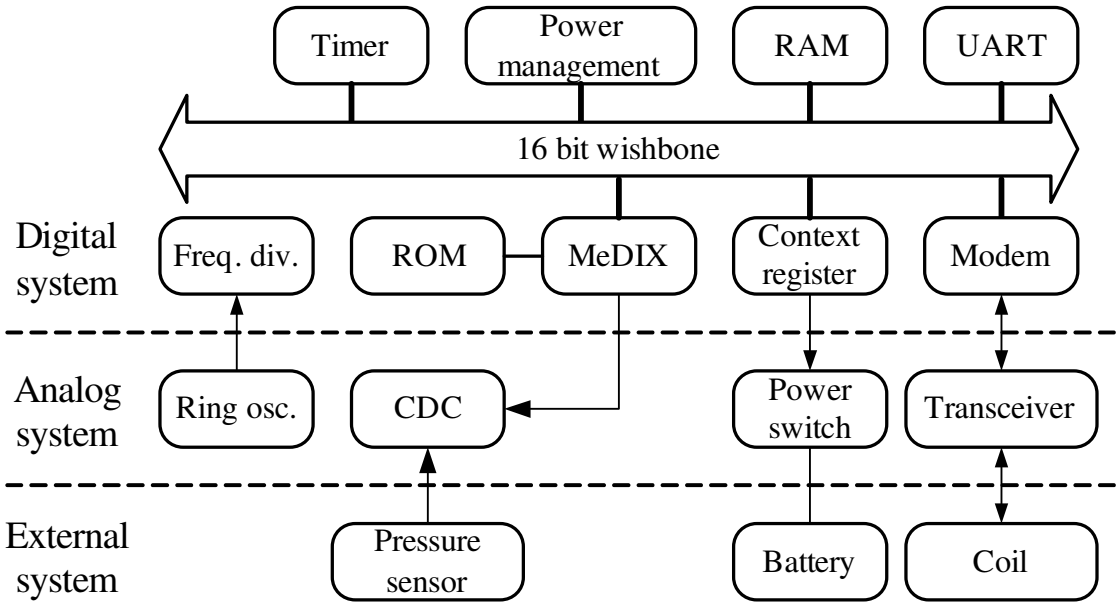

Figure 4.5: Block diagram of the MeSOC architecture.

without breaking the seal. In battery power mode, the CDC is interactively calibrated before measurement starts. Then, the digital system goes to sleep state for waiting events such as pressure measurement, receiving wireless data, and timers. The digital system is awaken by the events, deals with them, and sleeps again.

The block diagram of the MeSOC architecture is shown in Fig. 4.5. The MeSOC is designed as a mixed signal chip to integrate almost functions for the capsule. The digital system of the SoC consists mainly of MeDIX-I processor, a 6 KB instruction ROM, a 8 KB RAM shared for instructions and data, a 16550 compatible UART for development and debugging, a clock frequency divider, timers, a state machine for sleep and reset control, a context register for special controls including power switching, and a digital modem for wireless communication. They are all connected with 16-bit wishbone bus [42]. To satisfy the tight design constraints while keeping high programmability, the MeSOC is MeDIX-I centric design. MeDIX-I effectively processes CDC control, wireless communication, and power management with the help of special instructions.

4.3.3 Power Management

The operating frequency of MeDIX-I has a serious impact on power consumption. Assuming the tasks shown in Fig. 4.4, the minimum operating frequency is estimated about 150 kHz to meet the time constraint of the measurement. Taking account of frequency derivation from the ring oscillator, 161.429 kHz is appropriate for clock frequency for the digital system.

By using clock gating, the almost digital blocks shown in Fig. 4.5 can sleep according to the timing generated by MeDIX-I. The main tasks of the MeSOC are pressure measurement and wireless communication, and both tasks are periodical short-time task; therefore, such sleep mechanism is effective for low power. Programmers can shut down the MeSOC with the sleep instruction of MeDIX-I and the MeSOC wakes up according to the events shown in Fig. 4.4. In other words, reducing calculation time with the help of the special instructions results in power reduction.

Since the minimum size of the programs is difficult to determine, implemented RAM size is large. In order to reduce power consumption of the unused RAM, the RAM is split in four banks and each bank can be shut down independently.

4.4 Analog System

Following all analog modules have been proposed for the MeSOC: a CDC [40], a wireless transceiver [41], a ring oscillator which generates system clock at 968.571 kHz $\pm 1\%$, and a power controller. The CDC converts the capacitance of the MEMS pressure sensor, which is electrically equivalent to a variable capacitance. The wireless transceiver controls the communication coil. The ring oscillator generates system clock at 968.571 kHz $\pm 1\%$. Since crystals are not available due to assembly constraints of the capsule, there are large frequency error and jitter. In order to remotely control a power switch, the power controller connects and disconnects the battery without mechanical switches. In following sections, the featured analog modules, the CDC and the wireless transceiver, are introduced.

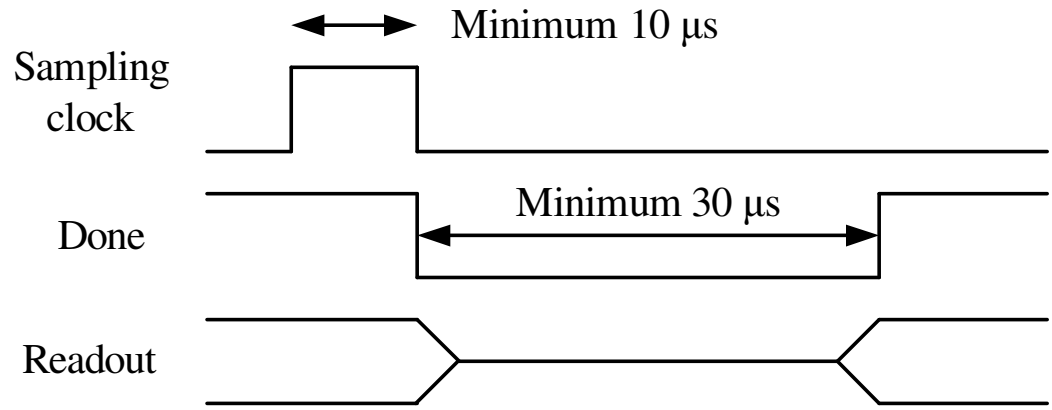

Figure 4.6: Asynchronous protocol of CDC.

4.4.1 CDC

A CDC directly converts capacitance value of the MEMS pressure sensor to digital value. Since the CDC is small, low-power, and robust for voltage fluctuation to fit miniature size capsule, it can be simply applied to the MeSOC. According to ICS, the pressure range of 0-225 mmHg is adequate for measurement [38]. The pressure range of the MEMS pressure sensor and the CDC are 0-280 mmHg and 0-300 mmHg, respectively. Therefore, the combination of them can satisfy the ICS recommendation. On the other hand, the previous report indicates that converted data include random error [43]; noise canceling mechanism is required for the digital system.

Besides, flexible asynchronous control system for the CDC is required for the digital system because all control signal timings shown in Fig. 4.6 change depending on operating conditions. The CDC converts data in asynchronous manner as shown in Fig. 4.6 [44]. To start conversion, the CDC needs one sampling clock pulse at least 10 μs. After the sampling pulse is asserted, the readout is available 30 μs later at earliest.

From the aspect of the CDC, it is desirable that both offset capacitance and the capacitance variation according to pressure change are several pF. If both capacitances are too large, the implementation area of the CDC increases. On the other hand, if both capacitances are too small, noise tolerance decreases. The choice of the MEMS sensor also depends on mechanical issue of the capsule. Taking into account capacitance, pressure range, low power capability, ease in handling, and size, the MEMS sensor by [39] is good for this application.

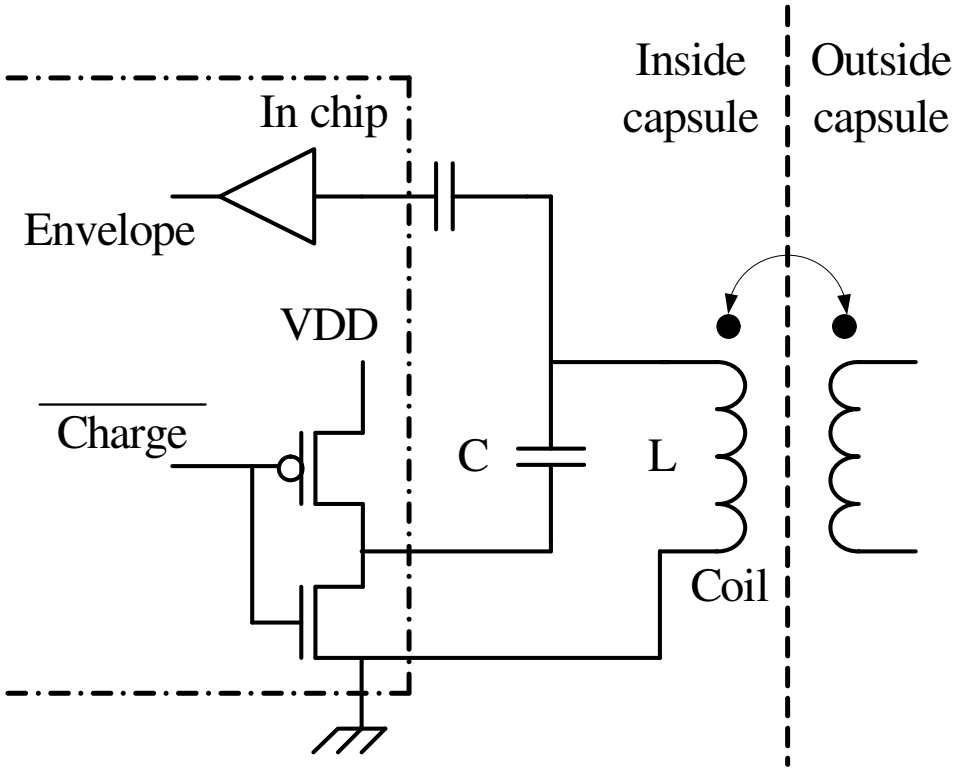

Figure 4.7: Wireless circuit for the capsule sensor.

4.4.2 Wireless Transceiver

The simplified wireless transceiver circuit is shown in Fig. 4.7. The capsule communicates with the outside recorder via inductive link, which is popular technique in RFID [45]. Since lower frequency has better characteristic of penetrating water, i.e., human body [46] [47], the transceiver selects 13.56 MHz as carrier wave from Industrial-Scientific-Medical (ISM) bands [48]. In Fig. 4.7, capacitance C and coil L generates a pulse with 13.56 MHz by LC oscillation. This modulation circuit consumes less power, and good communication property at a distance of 15 cm has been confirmed [41]. On the other hand, communication speed is slow; bit rate is under 10 K bps. Therefore, digital modulation and communication protocol is important for smooth communication. Additionally, in order to remotely power up the MeSOC in the airtight capsule, the transceiver also provides wireless power transmission by inductive coupling. After

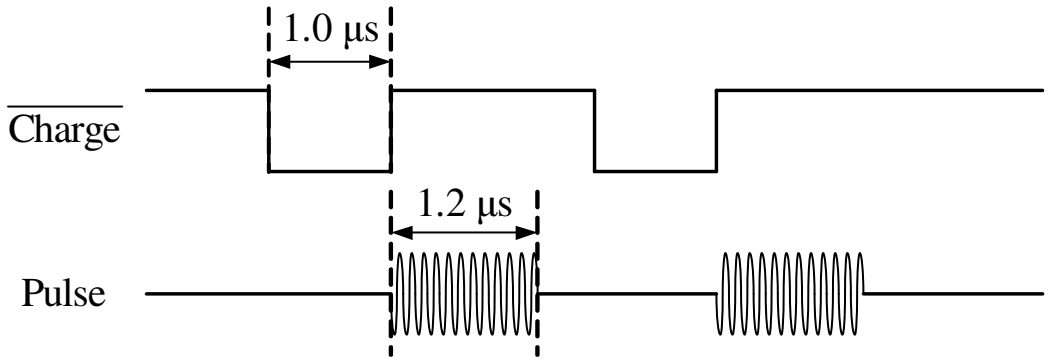

Figure 4.8: RF signal transmission.

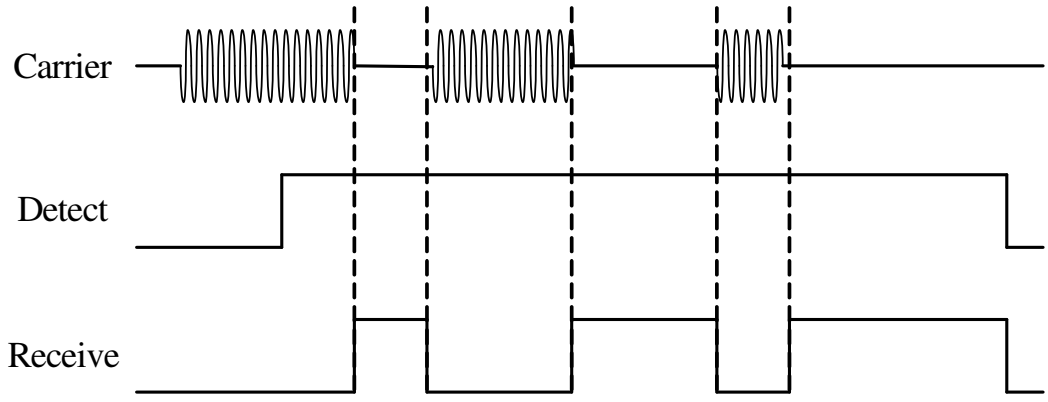

Figure 4.9: RF signal receiving.

the MeSOC is powered up by wireless power, the digital system needs to switch the power source to the battery at proper timing.

Transmission procedure is as follows: first, negating $\overline{Charge}$ for 1 μs charges up capacitance C. Next, asserting $\overline{Charge}$ connects charged C and L, oscillating at 13.56 MHz for 1.2 μs. Carrier cannot be continuously generated as shown in Fig. 4.8. With respect to receiving, digital signals appear at $Receive$ by On-Off-Keying (OOK) when 13.56 MHz carrier is detected as shown in Fig. 4.9.

4.5 Digital Architecture

In this section, the architecture of MeDIX-I, the wireless modem, and wireless protocols are described.

4.5.1 MeDIX-I

The block diagram of MeDIX-I is shown in Fig. 4.10. MeDIX-I is a 16-bit, 3-stage pipelined ASIP with 44 instructions. Double fetch pipeline stage is dedicated to hide the latency of the instruction memory. IF1 sends an address of Program Counter (PC) and IF2 stores an instruction to Instruction Register (IR). To reduce implementation area, the instruction set is small: 12 arithmetic and logical, 5 shift, 3 bit operation, 4 load and store, 6 branch, 3 CDC, 1 sleep, 7 Error Correcting Code (ECC), and 3 interrupt instruction. CDC control, wireless communication modem, and power management are programmed on MeDIX-I.

MeDIX-I consists of an ALU, a barrel shifter, a Multi-Dimensional Parity Check (MDPC) code module, and 16-bit 30 General Purpose Registers (GPRs). Normally, 16 GPRs are visible; the remaining 14 registers are shadow registers to accelerate interrupt response. MDPC, which is described later, is used for encoding and decoding of ECC. MeDIX-I does not comprise divider and multiplier to reduce gate counts. MeDIX-I can handle 8 external interrupts and 1 exception. The exception provides a break mechanism for GDB, which is the GNU debugger for development.

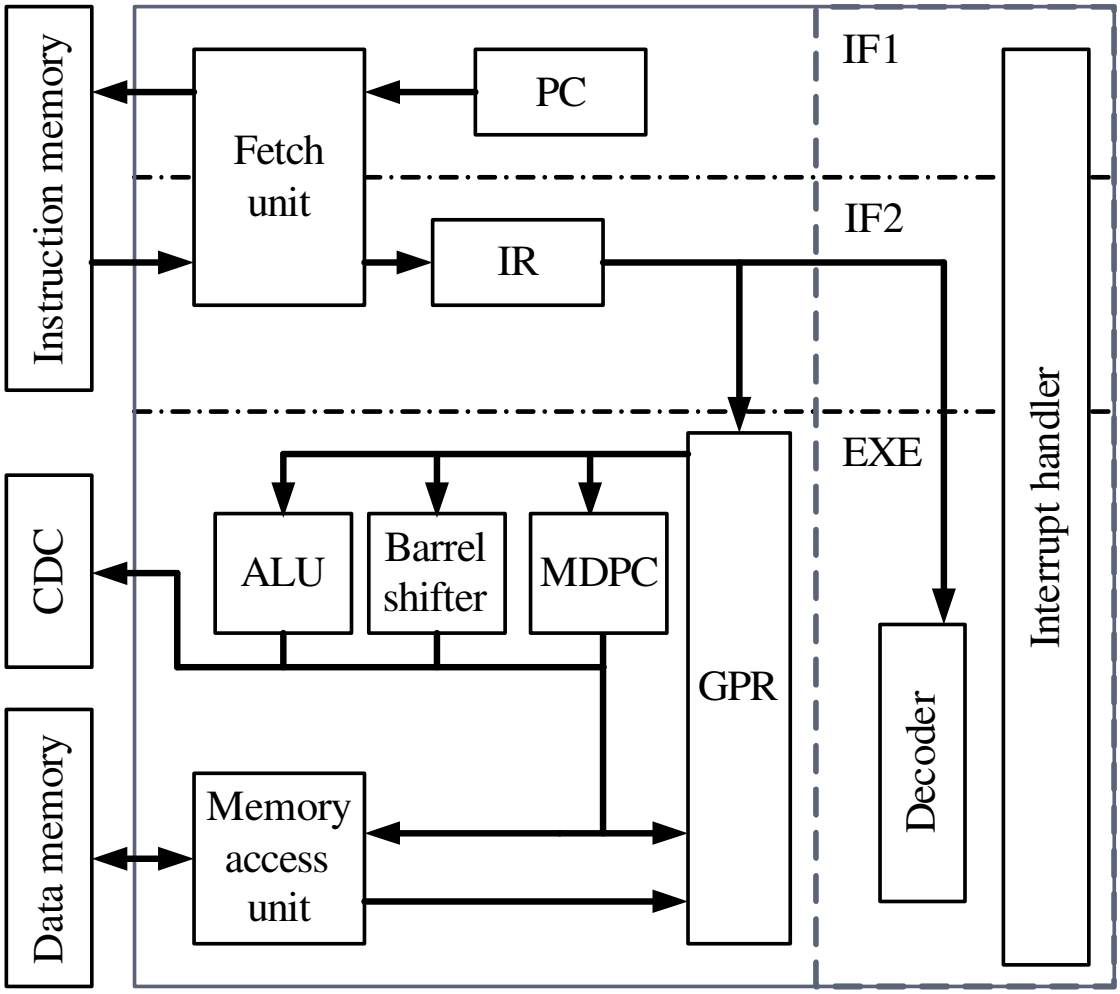

Figure 4.10: Block diagram of MeDIX-I.

4.5.2 CDC Control

The CDC requires noise canceling and asynchronous protocol. Before transmitting measured pressure data, noise canceling should be performed because the outputs of the CDC contain random noises in lower bits. For one pressure measurement, MeDIX averages 32 samples of pressure data to cancel the noise. This is because, to reduce gate counts, the shifter can be exploited for division.

For asynchronous control shown in Fig. 4.6, two major approaches are known: interrupt and polling. Since the delay of CDC conversion is several cycles in MeDIX clock, the polling approach has advantage because the overhead cycles of the interrupt mechanism is considerable for the wait in several cycles. To efficiently control the CDC, the dedicated port access instructions are implemented in MeDIX to directly access the CDC. Compared to general purpose I/O module hooked on the bus, the direct access can reduce overhead cycles for bus communication. Such efficient CDC control mechanism can contribute to low power by increasing idle time of MeDIX and decreasing implementation area.

4.5.3 Wireless Modem

The analog transceiver provides OOK modulation and demodulation; however, it is not sufficient to complete a wireless communication system. The digital system needs the additional upper layer protocols as shown in Fig 4.11. In this section, digital baseband modulation, packet format, and ECC coding are described.

4.5.3.1 Digital Baseband Modulation

The SoC adopts Bi-phase Mark Code (BMC) [49] for fundamental baseband coding. BMC is used floppy disks, optical digital audio interface (S/PDIF), and recently IEEE standard for long wavelength wireless network protocol (IEEE 1902.1) [50]. Figure 4.12 shows the relationship between BMC and Non Retruns to Zero (NRZ). In BMC, there are transitions, which is called timing edges, at the beginning and at the end of the NRZ bit period. When NRZ bit is zero, there is additional one transition, called a data edge, at the middle of the NRZ bit period. On

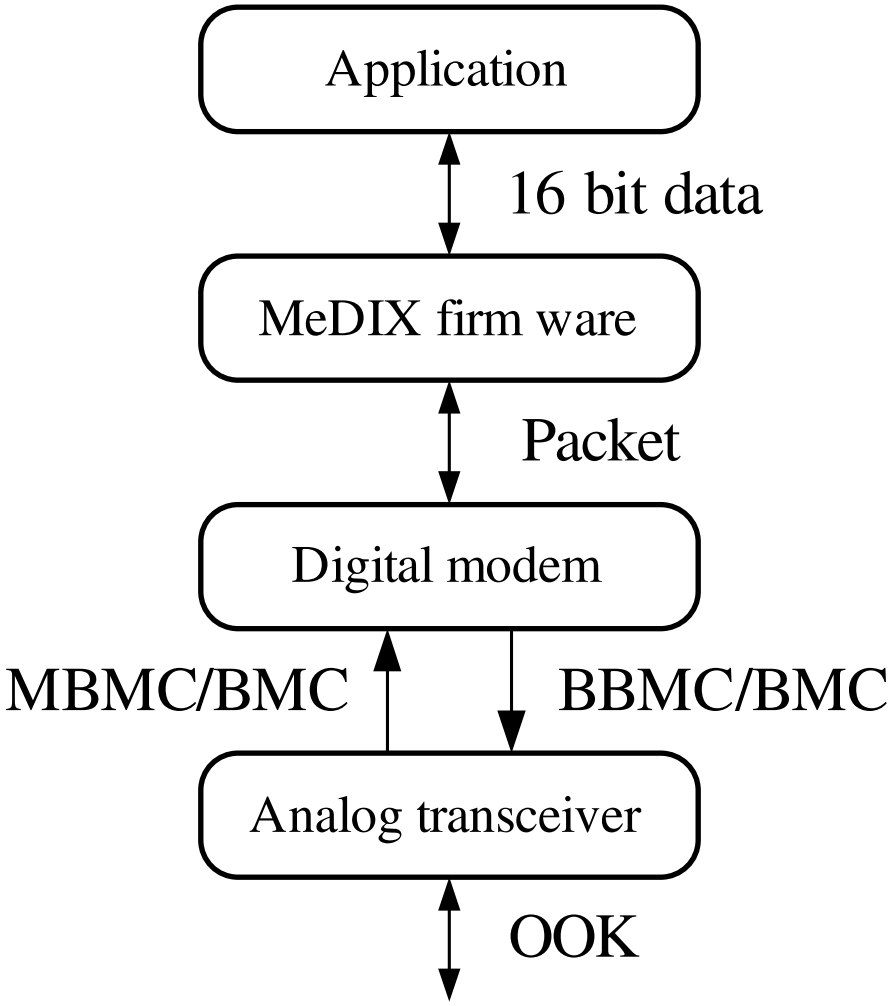

Figure 4.11: Protocol stack on this SoC.

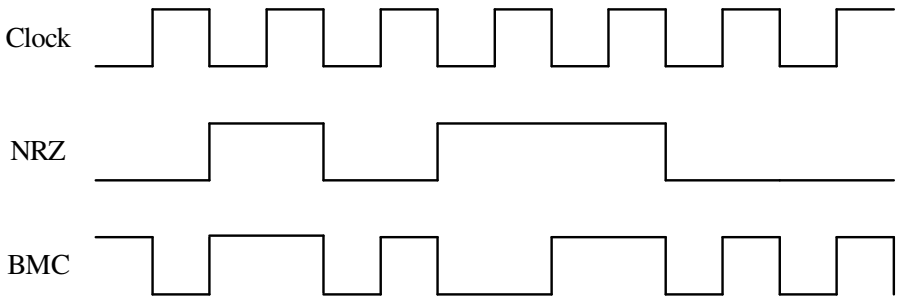

Figure 4.12: Line codes for the SoC.

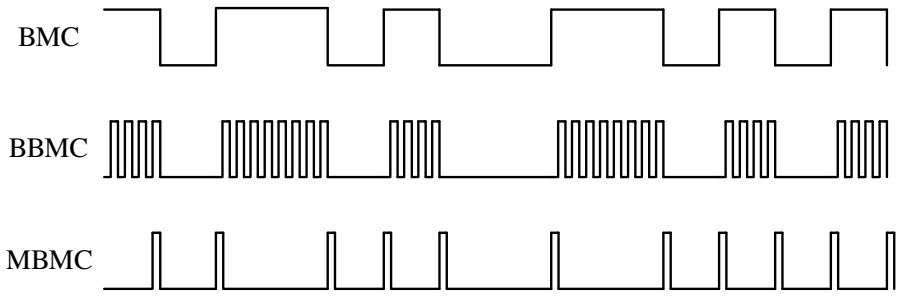

Figure 4.13: Relationship with BMC, BBMC, and MBMC.

Table 4.1: Use of baseband coding

	Uplink	Downlink
Wireless power	(none)	MBMC
Battery power	BBMC	BMC

the other hand, there is no additional transitions when NRZ is zero. BMC provides low DC bias, easy clock deriving, and good noise tolerance; it is suitable for this transceiver.

Though BMC can be used in down-link communication on battery power mode, it is not suitable for up-link communication because the transceiver can only generate very short-term and weak carrier. Similarly, BMC is not suitable for down-link communication on wireless power mode. To handle those cases, two new line codes, Burst BMC (BBMC) and Modified BMC (MBMC), are proposed.

As shown in Fig. 4.13, BBMC can be used for data transmission by generating series of the pulses while BMC is high. In contrast, MBMC can be used for down-link in wireless powered mode by generating pulses at the position of edges of BMC. The assignments of those coding are shown in Table 4.1.

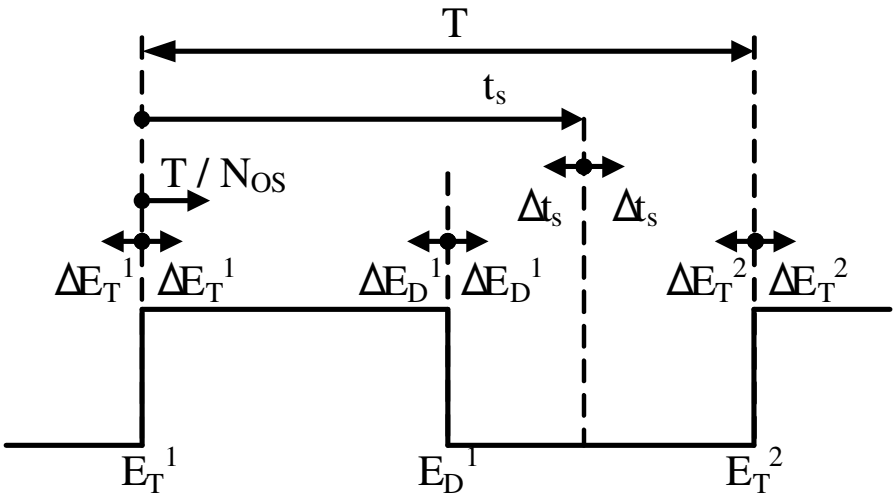

Figure 4.14: Timing degradation of received signal.

4.5.3.2 Oversampling Decoder of BMC

To accurately decode BMC signal, Clock and Data Recovery (CDR) is required. The policy of oversampling based CDR [51] is appropriate because Phase-Locked Loop (PLL) is not suitable for this MeSOC. The strategy to decode BMC with timing adjust is simple; the decoder checks the signal inversion at appropriate clock cycles after detecting a timing edge, then, the decoder waits the next timing edge. Oversampling rate for digital nature CDR affects jitter tolerance and power consumption; higher oversampling rate has higher signal jitter tolerance and higher oversampling rate consumes higher power. There is a trade-off between jitter tolerance and power consumption. For low power design, the optimum oversampling rate must be calculated.

Owing to the characteristics of the transceiver and the ring oscillator, timing degradation of received signal occurs as shown in Fig. 4.14. In Fig. 4.14, E_T^1, E_D^1, and E_T^2 are the timing edge of BMC bit, the data edge of BMC bit, and the timing edge of the next BMC, respectively. The time of these edges are $E_T^1 = 0$, $E_D^1 = T/2$, and $E_T^2 = T$ where T is a baseband period. t_s and Δt_s represent the latency to check signal inversion of the signal and a timing error due to

sampling clock error δ_{CLK} and jitter σ_{CLK}. t_s and Δt_s are described as follows:

$$t_s = C_{OS} \cdot \frac{T}{N_{OS}} \tag{4.1}$$

$$\Delta t_s = (\sigma_{CLK} + \delta_{CLK}) \cdot t_s \tag{4.2}$$

where C_{OS} is the cycle count of sampling clock and N_{OS} is an oversampling rate. The received signal is accompanied by 10 % of Duty-Cycle Distortion jitter (DCD), which is denoted as ΔE_T^1, ΔE_D^1, ΔE_T^2 and

$$\Delta E_T^1 = \Delta E_D^1 = \Delta E_T^2 = \sigma_{DCD} \cdot T. \tag{4.3}$$

Initial timing error denoted as T/N_{OS} due to oversampling also must be taken into account, where N_{OS} is an oversampling rate. The condition to accurately decode is to guarantee two timing slacks:

$$0 < TS^- = (t_s - \Delta t_s) - \left(E_D^1 + 2\Delta E_D^1\right) \tag{4.4}$$

$$0 < TS^+ = \left(E_T^2 - 2\Delta E_T^2\right) - \left(t_s + \Delta t_s + \frac{T}{N_{OS}}\right) \tag{4.5}$$

where TS^- is the worst case timing slack between E_D^1 and t_s, and TS^+ is the worst case timing slack between t_s and E_T^2. The minimum N_{OS} satisfying $TS^- > 0$ and $TS^+ > 0$ is the optimum oversampling rate.

The ideal sampling latency t_s^{ideal} is the center of the timing margin T_M:

$$\begin{aligned} T_M &= \left(E_T^2 - 2\sigma_{DCD} \cdot T - \frac{T}{N_{OS}}\right) - (E_D^1 + 2\sigma_{DCD} \cdot T) \\ &= T\left(\frac{1}{2} - 4\sigma_{DCD} \cdot T - \frac{1}{N_{OS}}\right) \\ t_s^{ideal} &= (E_D^1 + \Delta E_T^1 + \Delta E_D^1) + \frac{T_M}{2} \\ &= \frac{T}{2}\left(\frac{3}{2} - \frac{1}{N_{OS}}\right). \end{aligned} \tag{4.6}$$

C_{OS} can be calculated by using (4.6).

$$C_{OS} = \left\lfloor t_s^{ideal} \div \frac{T}{N_{OS}} \right\rfloor = \left\lfloor \frac{3N_{OS}}{4} - \frac{1}{2} \right\rfloor \tag{4.7}$$

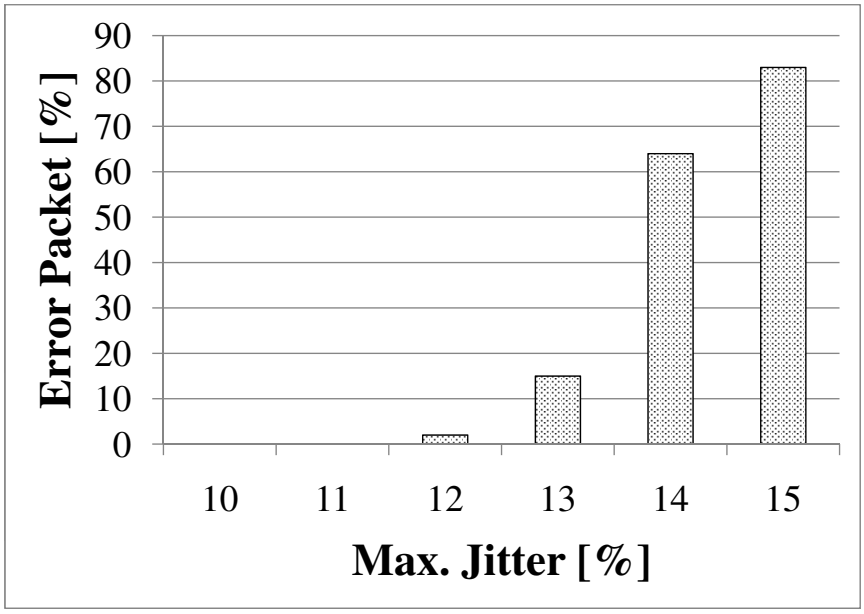

Figure 4.15: Packet error rate on jitter rate.

As a result,

$$TS^{-} = T\left\{ \left\lfloor \frac{3N_{OS}}{4} - \frac{1}{2} \right\rfloor \cdot \frac{1 - \sigma_{CLK} - \delta_{CLK}}{N_{OS}} - (0.5 + 2\sigma_{DCD}) \right\}$$
$$TS^{+} = T\left\{ \left(1 - 2\sigma_{DCD} - \frac{1}{N_{OS}}\right) - \left\lfloor \frac{3N_{OS}}{4} - \frac{1}{2} \right\rfloor \cdot \frac{1 - \sigma_{CLK} - \delta_{CLK}}{N_{OS}} \right\}$$

Here, by empoying given value $\sigma_{CLK} = 0.001$, $\delta_{CLK} = 0.01$, σ_{DCD}, following formulas are obtained.

$$TS^{-} = T\left(\frac{0.989}{N_{OS}} \cdot \left\lfloor \frac{3N_{OS}}{4} - \frac{1}{2} \right\rfloor - 0.714\right) \tag{4.8}$$
$$TS^{+} = T\left\{\left(0.786 - \frac{1}{N_{OS}}\right) - \frac{1.011}{N_{OS}} \cdot \left\lfloor \frac{3N_{OS}}{4} - \frac{1}{2} \right\rfloor\right\} \tag{4.9}$$

Consequently, $N_{OS} = 36$ and $C_{OS} = 26$ are the optimum value.

Figure 4.15 shows the simulation result of packet error rate according to jitter. As shown in Fig. 4.15, the decoder tolerates 11 % jitter injection, and it is enough to applicate for this SoC.

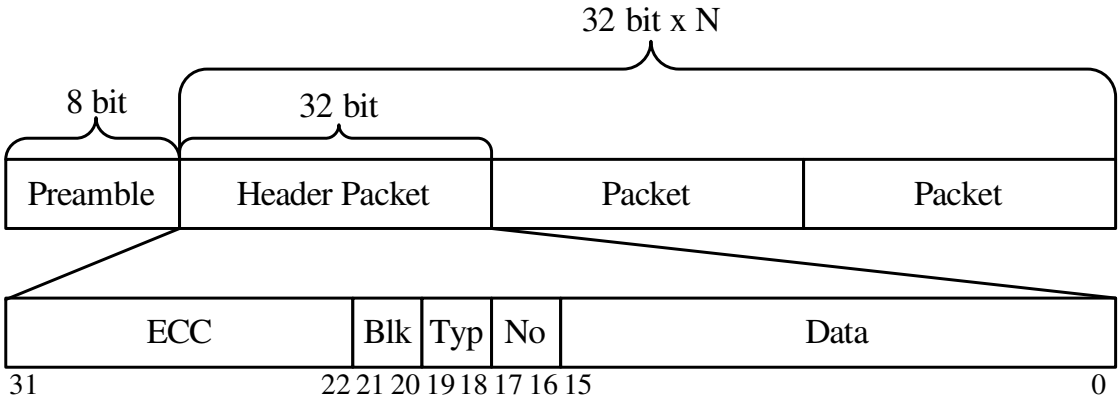

Figure 4.16: Packet format.

4.5.4 Packet Format

As shown in Fig. 4.16, packet format is designed as flexible to provide customize capability. Preamble format is configurable in 8 bit. Packet length is variable in 32 bit x N where $0 \leq \mathrm{N} \leq 8$. Basic packet format composed of 10-bit ECC, 2-bit Blk, 2-bit Typ, 2-bit No, and 16-bit data is shown in Fig. 4.16. Blk is the number of packet sequence. If packets are lost in communication, MeDIX-I can figure out the lost packets and request data again. Typ is a type flag, which indicates control (1) or data (0) packet. No is a capsule number and the capsules identify packet destination by it. ECC is Single-Error Correction and Double-Error Detection (SEC-DED) code. The detail of the ECC is described in the next section.

4.5.5 Error Correcting Code

Our prior experiment on a prototype program of MeDIX-I shows that over 50% of energy is consumed by wireless communication, especially, ECC calculation. Therefore, implementation of the special instructions for ECC calculation is effective to reduce of total power consumption.

MDPC codes [52] is adopted in this SoC. MDPC is a SED-DED code dealing with A^M data bits: $\mathrm{MDPC(A^M)}$. MDPC is defined as geometrical generalization of single-parity check

Table 4.2: MDPC calculation of 2^2.

	D[3]	D[2]	D[1]	D[0]
	$u_{2,2}$	$u_{2,1}$	$u_{1,2}$	$u_{1,1}$
$p_{1,1}$			$\oplus$	$\oplus$
$p_{1,2}$	$\oplus$	$\oplus$		
$p_{2,1}$		$\oplus$		$\oplus$
$p_{2,2}$	$\oplus$		$\oplus$	

codes [53]:

$$p_{m,j} = \sum_{i_1} \cdots \sum_{i_{m-1}} \sum_{i_{m+1}} \cdots \sum_{i_M} u_{i_1,\ldots,i_{m-1},j,i_{m+1},\ldots,i_M} \tag{4.10}$$

where $p_{m,j}$ is a parity bit, $u_{i_1,\ldots,i_M}$ is a data bit, $M > 1$, $i_1, \ldots, i_M$ range from 1 to A, $m = 1, 2, \ldots, M$, and $j = 1, 2, \ldots, A$.

Owing to the systematic construction, the calculation process of MDPC has recursiveness. The calculations of $\mathrm{MDPC}(2^2)$ and $\mathrm{MDPC}(2^3)$ are shown in Table 4.2 and 4.3. Table 4.2 shows that parity bit $p_{m,j}$ is calculated by XOR of data bits u_{i_1,i_2} indicated by $\oplus$, e.g., $p_{1,1}$ in Table 4.2 is calculated as $p_{1,1} = u_{1,2} \oplus u_{1,1}$. D[x] named u_{i_1,i_2} stands for data of x-th bit. With respect to Table 4.3, the calculation of parity bits from $p_{2,1}$ to $p_{3,2}$ corresponds to $\mathrm{MDPC}(2^2)$. Accordingly, $\mathrm{MDPC}(2^3)$ is composed of $\mathrm{MDPC}(2^2)$ and additional XORs. By exploiting this recursiveness, the pseudo-code of a loop calculation algorighm for MDPC is shown in Fig. 4.17.

In Fig. 4.17, $\mathrm{MDPC}(2^{\mathrm{M}})$ is calculated by using $\mathrm{MDPC}(2^{\mathrm{m}})$ calculation module where $m < M$. $D[x]$ is source data, p is the parity of $D[x]$, function $zero(p)$ sets p to zero, and function $binary(x)$ returns a binary form of x. Function $\mathrm{MDPC_m(x)}$ returns parity p' of $\mathrm{MDCP}(2^{\mathrm{m}})$ and additional parity pa'.

Figure 4.18 shows the $\mathrm{MDPC}(2^8)$ unit employing $\mathrm{MDPC}(2^4)$ in accordance with Fig. 4.17. After resetting counter b, by consecutively inputting 16-bit data fragments, the unit can calculates MDPC code of 16-bit to 256-bit wide data.

By exploiting the $\mathrm{MDPC}(2^8)$ unit shwon in Fig. 4.18, MeDIX-I can calculate variable length

Table 4.3: MDPC calculation of 2^3

	D[7]	D[6]	D[5]	D[4]	D[3]	D[2]	D[1]	D[0]
	$u_{2,2,2}$	$u_{2,2,1}$	$u_{2,1,2}$	$u_{2,1,1}$	$u_{1,2,2}$	$u_{1,2,1}$	$u_{1,1,2}$	$u_{1,1,1}$
$p_{1,1}$					$\oplus$	$\oplus$	$\oplus$	$\oplus$
$p_{1,2}$	$\oplus$	$\oplus$	$\oplus$	$\oplus$				
$p_{2,1}$			$\oplus$	$\oplus$			$\oplus$	$\oplus$
$p_{2,2}$	$\oplus$	$\oplus$			$\oplus$	$\oplus$		
$p_{3,1}$		$\oplus$		$\oplus$		$\oplus$		$\oplus$
$p_{3,2}$	$\oplus$		$\oplus$		$\oplus$		$\oplus$	

zero(p)
zero(pa)
for $(b \leftarrow 0; b < M - m; b \leftarrow b + 1)$ **do**
    $(p', pa') \leftarrow \mathrm{MDPC_m(D[2^m \cdot (b + 1) - 1 : 2^m \cdot b])}$
    for $(i \leftarrow 1; i \leq m; i \leftarrow i + 1)$ **do**
        $j = (M - m + i)$
        $p_{j,1} = p_{j,1} \oplus p'_{i,1}$
        $p_{j,2} = p_{j,2} \oplus p'_{i,2}$
    end for
    for $(i \leftarrow 1; i \leq M - m; i \leftarrow i + 1)$ **do**
        if $\mathrm{binary(b)[i - 1]} = 0$ **then**
            $p_{i,1} \leftarrow p_{i,1} \oplus pa'$
        else
            $p_{i,2} \leftarrow p_{i,2} \oplus pa'$
        end if
    end for
    $pa = pa \oplus pa'$
end for
return(p, pa)

Figure 4.17: $\mathrm{MDPC}(2^{\mathrm{M}})$ calculation algorithm.

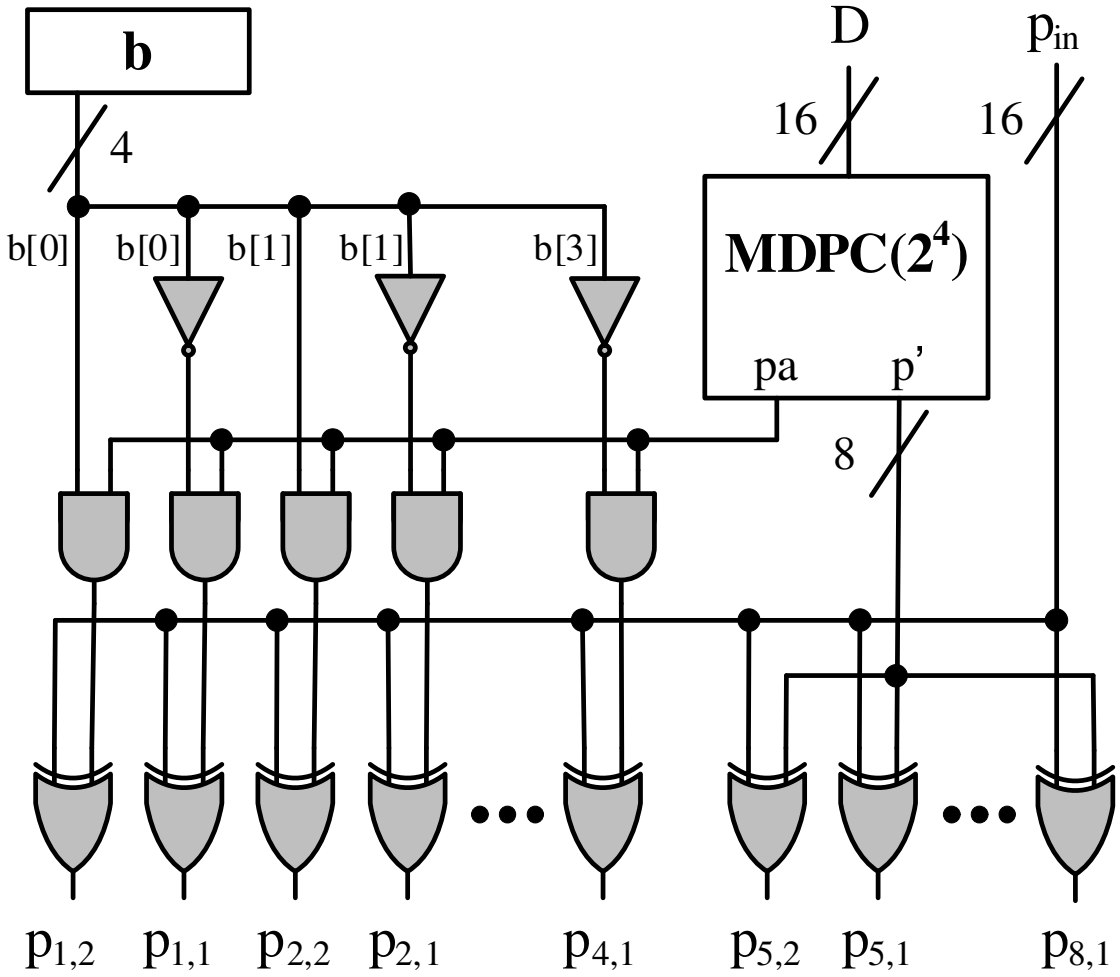

Figure 4.18: $\mathrm{MDPC}(2^8)$ calculation circuit by using $\mathrm{MDPC}(2^4)$.

MDPC in a few cycles. The special instructions for MDPC are implemented in MeDIX-I: MDPCGEN, MDPCCHK, MDPCFIX, and MDPCINIT. MDPCINIT sets internal counter b and must be call before MDPC calculation. MDPCGEN calculates MDPC code by using $\mathrm{MDPC}(2^8)$ unit. MDPCCHK checks errors and specifies the location of the errors. MDPCFIX fixes the data according to the result of MDPCCHK. By using the instructions, the ECC of $16n$ bit where $n = 1, 2, 3, \ldots, 16$ can be calculated in a few cycle. For instance, MDPC calculation of 32 bit data are shown below.

1: MDPCINIT
2: XOR r3, r3
3: MDPCGEN r3, r1
4: MDPCGEN r3, r2

Assuming the lower 16 bit and the upper 16 bit are stored register r1 and r2 respectively, after initialization (line 1, 2), the MDPC is calculated with two MDPCGEN instructions (line 3, 4), thus the result is in register r3. This flexible ECC generation scheme is useful for actual development *in vivo* because this mechanism offers programmability that can control the trade off between error tolerance and communication speed.

4.5.6 Communication Protocol

A communication timeline is shown in Fig 4.19. The communication control is based on master-slave model. A master recorder attached on the body communicates with the multiple capsules. The capsules send the pressure data at every measurement. The master recorder requests pressure data to each capsule and the capsule returns measured data to recorder. If the request or response is lost due to bad channel conditions, the capsule buffers data in memory until the channel recovers. After the channel recovers, the capsule returns the stored multiple data to the recorder. Such data storing is important for increasing patient's comfort in AUM because the mechanism allows patients to detach recorder. For example, in the case of taking a bath, data sending fails because the patients detach the recorder. The MeSOC can store the pressure data during data sending fails and transmit the data after the patient attach the recorder again.

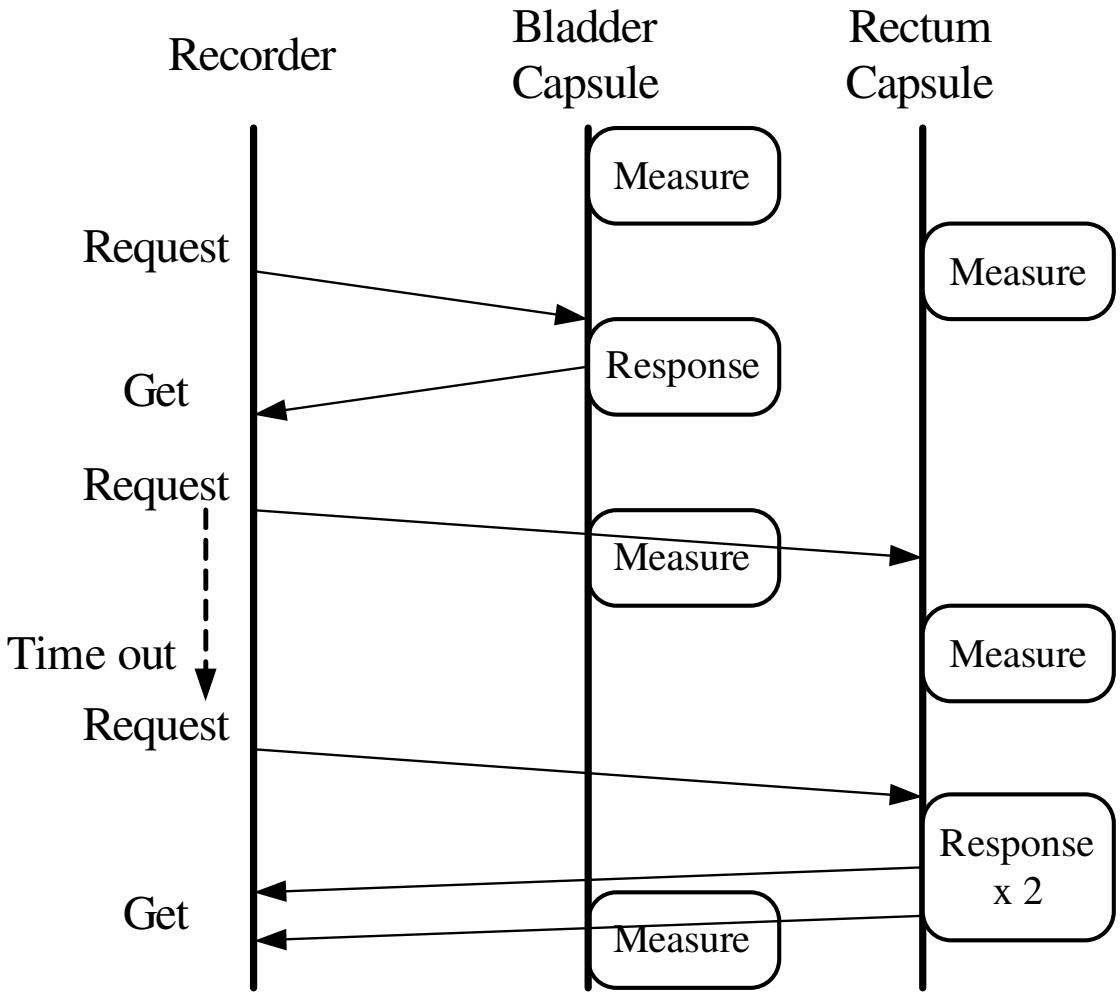

Figure 4.19: Communication timeline.

Table 4.4: Power consumption of the digital block

No.	MDPC enc.	Sleep	Sensing Freq.	MeDIX-I [μW]	Memory [μW]	Digital block [μW]
C1	RISC	No	30 Hz	17.2	66.1	102.0
C2	RISC	Yes	30 Hz	4.8	30.9	46.7
C3	MDPC ISA	Yes	30 Hz	4.0	29.3	43.7
C4	MDPC ISA	Yes	15 Hz	2.8	27.2	39.7

4.6 Experimental Results

In this section, simulation results and implementation results of the MeSOC are described.

4.6.1 Simulation Results

The MeSOC consists of the digital system, which is proposed in this thesis, and the analog systems proposed in [40, 41]. In order to evaluate detail power breakdown of the proposed digital system, power consumption was estimated based on simulations. Since the power data of 0.18 μm standard cell library was based on the rated voltage 1.8 V, the simulations were performed on 1.8 V. Simulations were carried out using the programs which measured pressure, received requests, created packet with ECC, then sent data to the transceiver.

The measurement conditions are in the following cases:

C1 measures pressure in 30 Hz without sleep control, encode MDPC with RISC instructions,

C2 measures pressure in 30 Hz with sleep control, encode MDPC with RISC instructions,

C3 measures pressure in 30 Hz with sleep control, encode MDPC with the special instructions,

C4 measures pressure in 15 Hz with sleep control. encode MDPC with the special instructions,

Note that, according to ICS, 15 Hz is adequate for pressure measurement. Typical sampling rate is set to 30 Hz as double as the recommendation.

The estimated power consumptions of the digital system are shown in Table 4.4. Power consumption of the digital block in C2 is reduced 54 % compared to C1. This reduction shows

Table 4.5: Chip summary

Technology	0.18 μm 1P6M Mixed Signal
Die size	2.5 mm x 2.5 mm
Supply voltage	1.55 V
Logic gates	28.2 K gates
ROM	6 K bytes
RAM	8 K bytes
Clock frequency	161.429 kHz
Working power	93.5 μW

the effect of the sleep control. Compared to C2, power consumption of MeDIX-I in C3 is reduced 17 %. This is because, faster calculation by the MDPC instructions increases sleeping time. Shorter working time results in less power consumption. As the same reason, power consumption of MeDIX-I in C4 is less than 30 % because the sampling rate is less than C3. This result suggests that the MeSOC offers programmability of controlling the trade off between power consumption and sensing resolution.

On the other hand, the power consumption of the analog system was also estimated in simulation. The simulation results reported that the power of the analog system is approximately 60 μW.

4.6.2 Implementation Results

The MeSOC was fabricated as 2.5 x 2.5 mm^2 under 0.18 μm mixed-signal CMOS technology. The micrograph of the MeSOC is shown in Fig. 4.20. In Fig 4.20, the analog block is upper left and the other area is the digital system. I/Os were implemented in bump form for chip size packaging. The detail of the fabricated MeSOC is summarized in Table 4.5. As shown in Table 4.5, the actual total power consumption of the MeSOC is 93.5 μW in the same condition of C4 in Table 4.4 except for supply voltage. By using the MeSOC, the capsule is able to run over seven days. The systems are designed so that it can operate on 1.55V, which is the supply

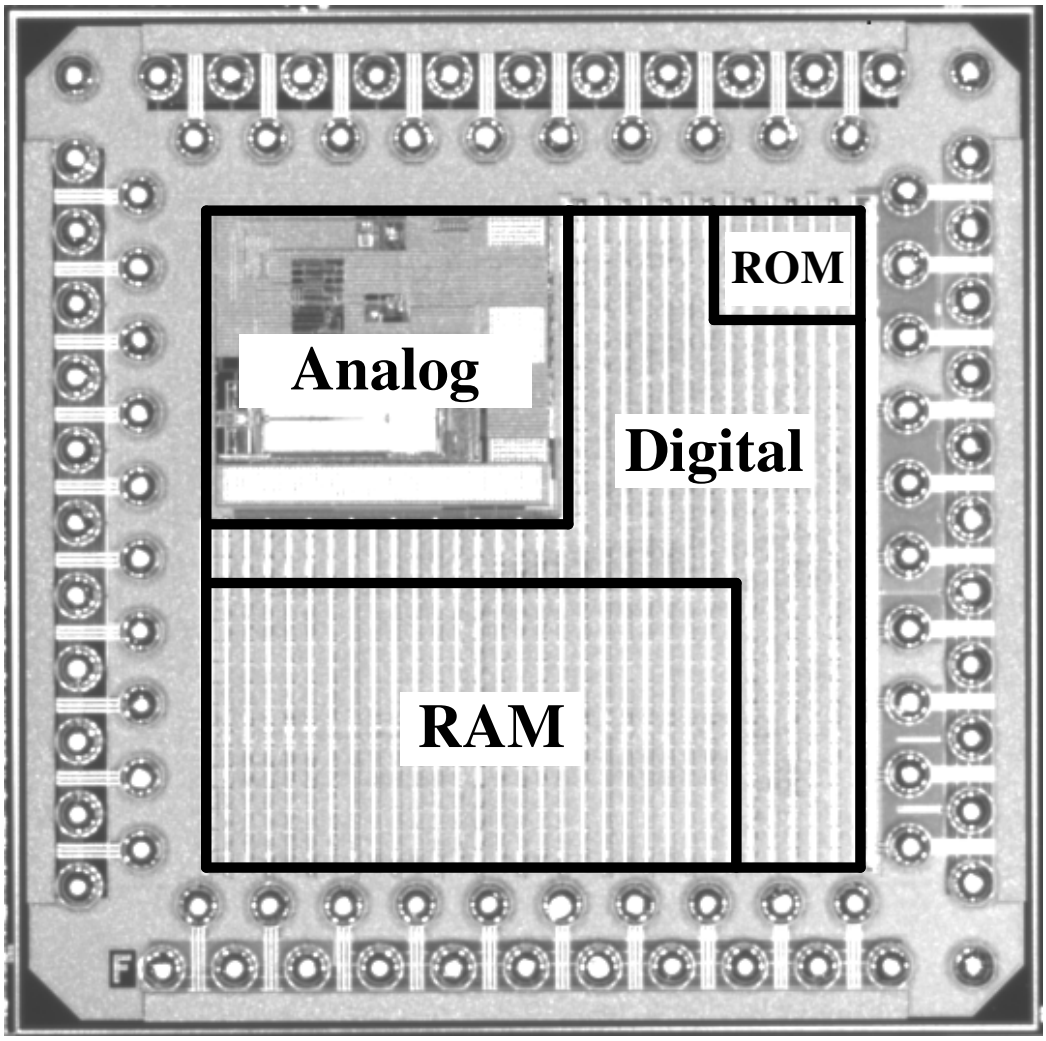

Figure 4.20: Chip micrograph.

voltage of the battery. And the systems have been confirmed that they could operate on 1.55V. The MeSOC was tested on an evaluation board and wireless communication was confirmed. By using GDB, the program was loaded via wireless link and interactively debugged. The pressure measurement with the MEMS sensor was also confirmed.

The simulation result shows that the digital system consumes 39.7 μW in case C4 of Table 4.4 and analog systems does 60 μW, and implementation result shows that total power consumption of the MeSOC is 93.5 μW, nearly equal to the sum of simulation results. In simulations, power consumption of the analog block was measured by multiplication of voltage and current in the power-supply line. The simulation result shows that power consumption

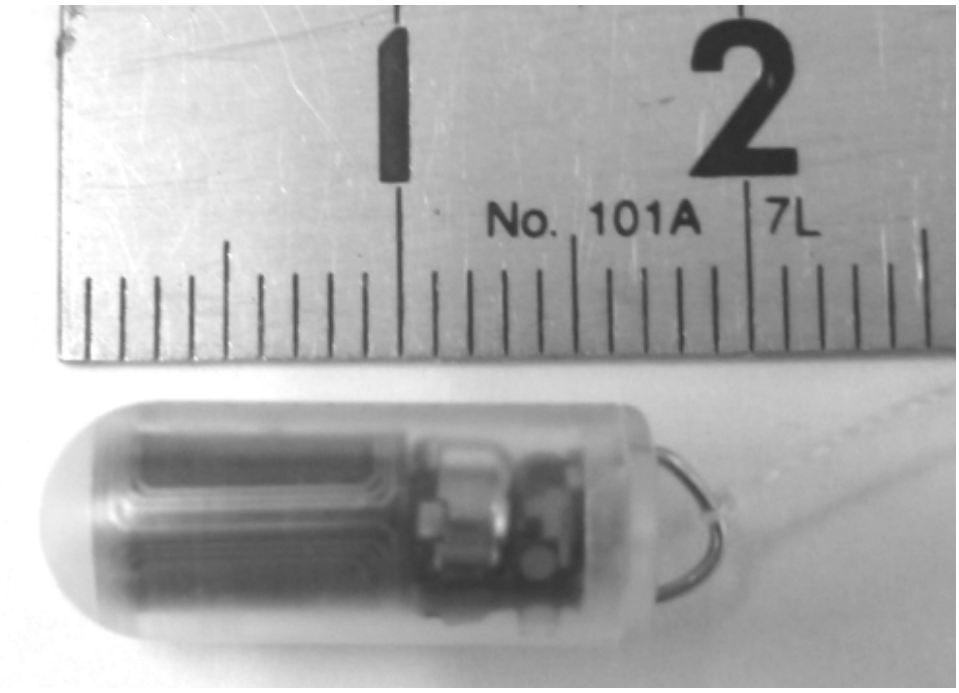

Figure 4.21: Photo of an assembled capsule prototype.

is approximately 60 μW . Implementation results were measured by inserting testers to the power-supply line. The implementation result shows that power consumption is 57 μW . Both values are near, and the implemented analog block was confirmed that it properly operates.

We assembled a capsule prototype with the communication coil, the battery, the MEMS sensor, and the fabricated MeSOC. The photograph of the prototype is shown in Fig. 4.21. The diameter and length of the prototype are 6 mm and 18 mm, which satisfy the requirement for urethra insertion. The weight of the capsule is 0.53 g, which is enough light to float in urine. A thin string of nylon, which is connected to the right side of the capsule, is used for pulling out the capsule from the body. As shown in Fig. 4.21, the form and size satisfy the medical constraint for insertion.

Pressure measurement was tested on the test environment as shown in Fig. 4.22. The results are shown in Fig. 4.23. The test environment consists of a syringe, a pressure meter, a MEMS pressure sensor, and a MeSOC evaluation board. By pushing a plunger, air pressure changed,

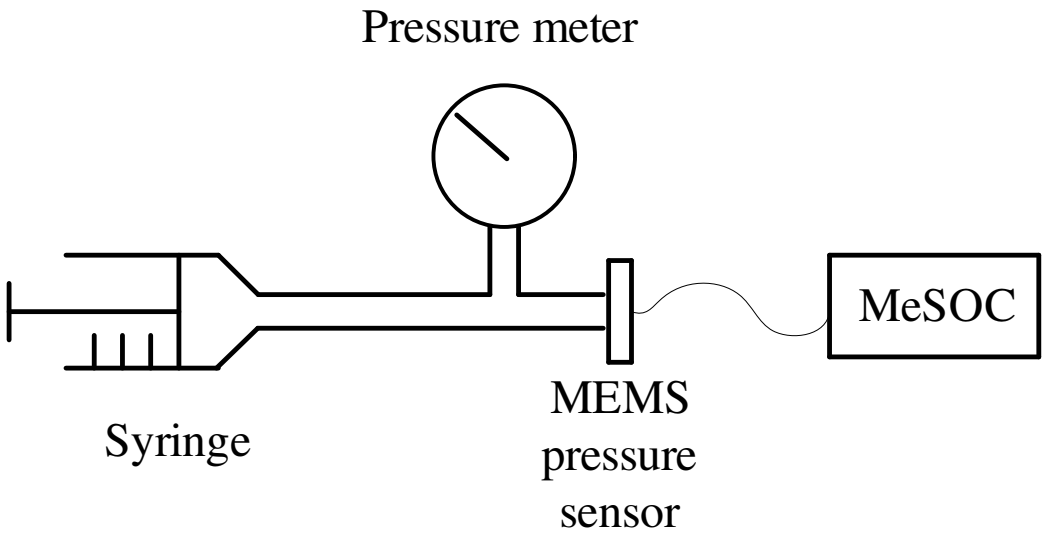

Figure 4.22: Test environment of CDC pressure sensing.

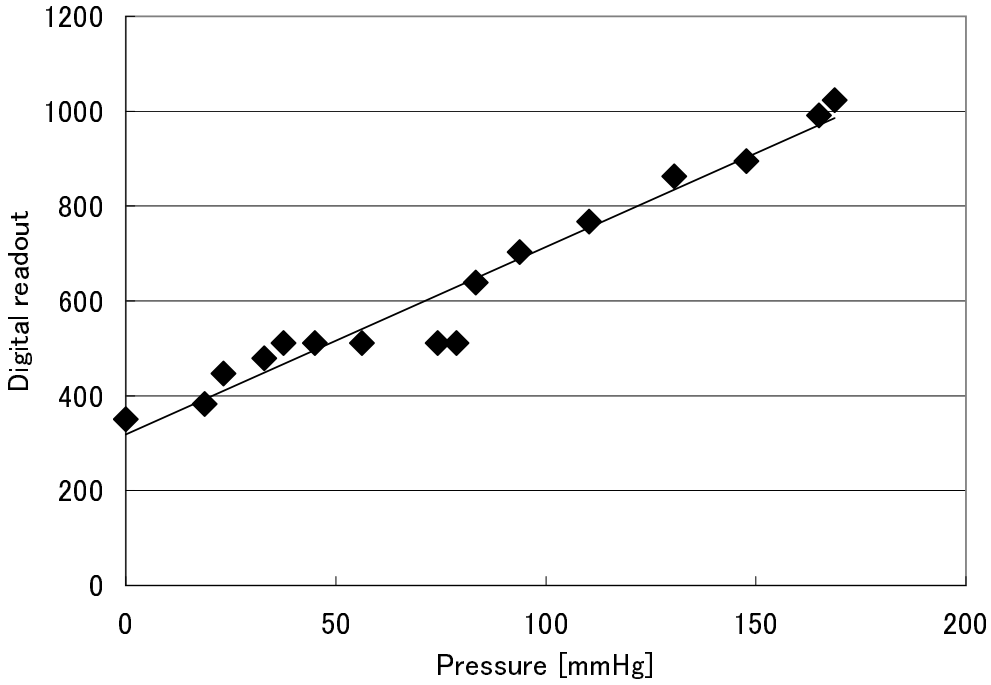

Figure 4.23: Pressure readout of CDC.

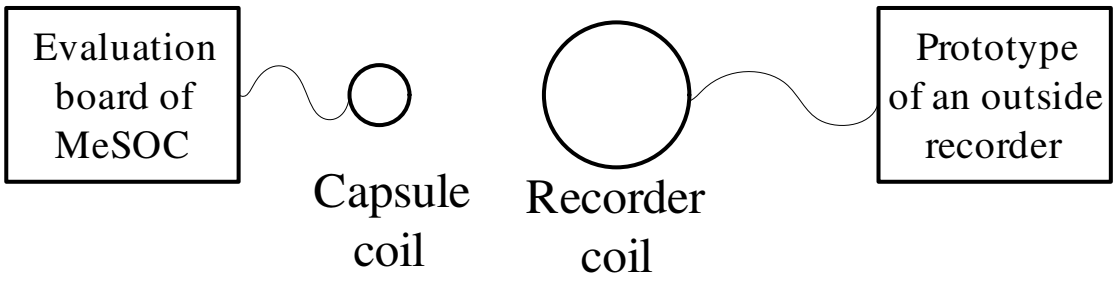

Figure 4.24: Test environment of wireless communication.

and the pressure meter measured correct value, then the MEMS sensor and MeSOC measured the pressure data. As shown in Fig. 4.23, in most range, CDC readout is confirmed to be linear.

Wireless communication was also confirmed on the test environment as shown in Fig. 4.24. The values of L and C of the antenna were 0.74 μH and 186.8 μW *in vitro* experiments. By using those values, 0.15 meter communication was confirmed. However, actual values of L and C of the antenna should be determined *in vivo* experiments and that is our future work. By using GDB, The program was loaded via wireless link and interactively debugged. The measured pressure data was uploaded to master controller and confirmed to be correct.

The comparison of the pressure sensing devices is shown in Table 4.6. The power consumption shown in Table 4.6 contains the digital block, the analog block, and the antenna. Power consumption of the antenna is calculated as follow:

$$P = fCV^2$$

where P is power consumption, f is switching frequency, C is capacitance, and V is operating voltage. As a result, power consumption of the antenna is 0.55 μW because f is approximately 1.0 Kbps in this system, C is 186.6 pF, and V is 1.55 V. Power consumption of the MEMS pressure sensor is negligible because the capacitance of the sensor is few pF and sampling frequency is 30 Hz. Therefore, total power consumption of the capsule is 94.1 $\mu\mathrm{W}$. Compared to related work [31], the proposed capsule consumes 1/10 less power. Compared to related work [30], the proposed capsule consumes 10 times more power. However, the system [30] cannot be used in AUM, because the system [30] needs incision to implant, cannot calibrate

Table 4.6: Comparison of the pressure sensing systems

	This work	[31]	[30]
Shape	cylinder	sphere	box
Size (mm)	ϕ = 6.5, h = 12	ϕ = 25	7 x 4 x 15
Weight (g)	0.53	4.1	unknown
Pressure range (mmHg)	0-280	0-1019	0-183
Pressure resolution (mmHg)	0.27	15.9	0.74
Sampling frequency (Hz)	0-30 (variable)	0.22 (average)	10
Communication	bi-direction	uni-direction	uni-direction
Com. distance (m)	0.15	1.00	0.30
Operating Voltage (V)	1.55	3.0	unknown
Battery (mAh)	12 x 1	70 x 4	3 x 1
Power (μW)	94.1	1250	8.6
Life time	7 days at 30 Hz	14 days	rechargeable
Setup	insertion	incision	incision

sensors *in vivo*, and cannot simultaneously measure bladder and abdominal pressure.

The capsule implemented with the MeSOC shows sufficient characteristics with respect to size, weight, pressure range, pressure resolution, life time, and less-invasive capsule setup. However, communication distance is shorter than other systems and communication may not stabilize in fat patients. Therefore, the communication protocol should be optimized in actual development *in vivo*.

4.7 Conclusion

In this chapter, the design of the MeSOC for less-invasive pressure sensing capsules for AUM is studied. To install the MeSOC to the capsules, the strict constraints are imposed on the SoC: low power, small area, bi-directional wireless communication, and programmability. The digital system is designed with MeDIX-I, which is the ASIP dedicated to this SoC. All systems including the CDC and the transceiver are integrated in the SoC.

The MeSOC was fabricated as 2.5 x 2.5 mm^2 under 0.18 $\mu\mathrm{m}$ mixed-signal technology. The experimental results showed that the power consumption of the MeSOC is 93.5 $\mu\mathrm{W}$ at 1.55 V and the capsule can continuously run over seven days.

Chapter 5

Conclusion and Future Work

This chapter describes the conclusion and future work of this thesis.

5.1 Conclusion

This thesis describes a low power VLIW ASIP generation method and the design of the SoC called MeSOC for pressure sensing capsules in AUM. The MeSOC is designed with the MeDIX-I, which is the ASIP customized for the MeSOC.

The low power VLIW ASIP generation method uses clock gating to reduce power consumption of pipeline registers. The power reduction by clock gating depends on execution conditions, and the proposed method automatically extracts the minimum execution conditions in ASIP generation procedures. By using the minimum execution conditions, the low power VLIW ASIP generation method creates the gating signals, and achieves maximum power reduction by clock gating. Experimental results show that the power consumption of the pipeline registers is drastically reduced. Compared to the ASIPs which are clock gated by Power Compiler, the ASIPs generated by the proposed method achieves 60% power reduction. This is because the gating control signals using the minimum execution conditions stops unnecessary clock supply to the pipeline registers. The delay and area overhead due to the low power VLIW ASIP generation are confirmed to be small. Therefore, the proposed method can provide low power VLIW ASIPs to designers.

To design the SoC which integrates almost functions in the pressure sensing capsule is im-

portant for realizing AUM. The MeSOC mainly consists of a CDC, a wireless transceiver, digital modem, and MeDIX-I. In order to design small and low power MeSOC, MeDIX-I has the special instructions for CDC control, ECC decode and encode, and control sleep mechanism. Experimental results show that the fabricated MeSOC satisfies the constraints and the correct operations of pressure sensing and wireless communication. The power consumption to calculate ECC is reduced by MeDIX-I. A prototype capsule is assembled with the MeSOC. The MeSOC can be implemented in a tiny airtight capsule for pressure sensing, and the size is small enough to pass through the urethra. The power consumption of the capsule including the MeSOC is confirmed to be low and the the capsule can run over seven days. Therefore, the pressure sensing capsule with MeSOC can be used in AUM.

5.2 Future Work

The future work includes the following items.

5.2.1 Future Work on the Low Power VLIW ASIP Generation

Applying clock gating causes Design For Test (DFT) problems. Since the number of inner states of circuits increases, the calculation time of test pattern generation increases. In order to apply the low power VLIW ASIP generation method to industrial productions, DFT problems due to clock gating must be handled.

The use of clock gating is studied in this thesis. However, clock gating reduces only dynamic power consumption. The progress of semiconductor manufacturing technology decreases gate size. As a result, leakage power consumption is also the big concern in recent embedded system design. Power gating is a technique to reduce leakage power consumption [54] by shutting power supply according to idle conditions. The extraction method of the minimum execution conditions is applicable to power gating. However, fine-grain gating control such as the low power VLIW ASIP generation is not suitable for power gating because fine-grain power gating is difficult to layout and additional wake up controls are necessary. To handle leakage power reduction using power gating is indispensable for future VLIW ASIP generation.

An ASIP generation method with operand isolation is useful in design of small ASIPs. In

small ASIPs, they may have big functional blocks to implement special instructions. To stop excess switchings to those blocks can contributes to low power design. Since the operand isolation is a high-overhead technique in terms of area and power, to determine candidate functional blocks for power reduction is a difficult task. Extraction of isolation conditions with profiling data of programs is the key to implement operand isolation.

5.2.2 Future Work on the Design of an SoC for Pressure Sensing Capsules

The basic functions of the pressure sensing capsule are confirmed *in vitro* in this thesis. In order to advance the pressure sensing system to the next step, *in vivo* development and test must be performed. *In vivo* development, calibrations are performed in wireless communication and pressure sensing. For such calibrations, programmable MeDIX-I can be a great help. If pressure sensing timing changes, re-program MeDIX-I to handle new timings. If wireless communication is not stabilized, increase error tolerance by using the ECC instructions. Therefore, the pressure sensing capsule with MeSOC is applicable to *in vivo* development, and actual use.

However, the size of RAM should be reduced in the next fabrication. The size of RAM in MeSOC contains extra memory to prepare re-programming *in vivo* development. After *in vivo* development, reducing the extra size of RAM can contribute power and size reduction.

Development of an outside recorder is the another future work. The outside recorder collects data from the capsules in a body and automatically creates a urinary diary to diagnose LUTS. As the same reason of the pressure sensing capsule, the size of the recorder must be small in AUM. An SoC design with ASIPs is also the key to design such small recorder.

Bibliography

[1] M. Johnson, Superscalar Microprocessor Design, Prentice-Hall, Inc., 1991.

[2] J.A. Fisher, "Very long instruction word architectures and the eli-512," in Proc. of the 10th annual International Symposium on Computer Architecture (ISCA '83).

[3] M.F. Jacome, G. de Veciana, and V. Lapinskii, "Exploring performance tradeoffs for clustered VLIW ASIPs," in Proc. International Conference on Computer Aided Design (ICCAD '00), pp.504–510, 2000.

[4] B. Middha, A. Gangwar, A. Kumar, M. Balakrishnan, and P. Ienne, "A Trimaran based framework for exploring the design space of VLIW ASIPs with coarse grain functional units," in Proc. 15th International Symposium on System Synthesis (ISSS '02), pp.2–7, 2002.

[5] A. Hoffmann, H. Meyr, and R. Leupers, Architecture Exploration for Embedded Processors with LISA, Kluwer Academic Publishers, Boston, 2002.

[6] P. Mishra, A. Kejariwal, and N. Dutt, "Rapid Exploration of Pipelined Processors through Automatic Generation of Synthesizable RTL Models," Proceedings of 14th IEEE International Workshop on Rapid Systems Prototyping, pp.226–232, June 2003.

[7] M. Itoh, Y. Takeuchi, M. Imai, and A. Shiomi, "Synthesizable HDL Generation for Pipelined Processors from a Micro-Operation Description," IEICE Trans. Fundamentals of Electronics Communications and Computer Sciences, vol.E83-A, no.3, pp.394–400, March 2000.

[8] Y. Kobayashi, S. Kobayashi, K. Okuda, K. Sakanushi, Y. Takeuchi, and M. Imai, "HDL generation method for configurable VLIW processor," Trans. Information Processing Society of Japan, vol.5, no.45, pp.1311 – 1321, 2004. (in Japanese).

[9] M. Prabhat and D. Nikil, eds., Processor Description Languages : Applications and Methodologies, Morgan Kaufmann, 2008.

[10] T. Lang, E. Musoll, and J. Cortadella, "Individual flip-flops with gated clocks for low power datapaths," IEEE Trans. Circuits Syst. II, vol.44, no.6, pp.507–516, June 1997.

[11] M. Munch, B. Wurth, R. Mehra, J. Sproch, and N. Wehn, "Automating RT-level operand isolation to minimize power consumption in datapaths," in Proc. Design Automation and Test in Europe (DATE '00), pp.624–633, 2000.

[12] M. Pedram, Power Aware Design Methodologies, Kluwer Academic Publishers, Norwell, MA, USA, 2002.

[13] D.E. Duarte, N. Vijaykrishnan, and M.J. Irwin, "A clock power model to evaluate impact of architectural and technology optimizations," IEEE Trans. VLSI. Syst., vol.10, no.6, pp.844–855, Dec. 2002.

[14] P. Abrams, J.G. Blaivas, S.L. Stanton, and J.T. Andersen, "The standardization of terminology of lower urinary tract function recommended by the international continence society," International Urogynecology Journal, vol.1, no.2, pp.45–58, 1990.

[15] T. Nevus, A. von Gontard, P. Hoebeke, K. Hjlms, S. Bauer, W. Bower, T. Jrgensen, S. Rittig, J. Walle, C. Yeung, and J. Djurhuus, "The standardization of terminology of lower urinary tract function in children and adolescents: Report from the standardisation committee of the international children's continence society.," The Journal of Urology, vol.176, no.1, pp.314–324, 2006.

[16] W. Schäfer, P. Abrams, L. Liao, A. Mattiasson, F. Pesce, A. Spangberg, A.M. Sterling, N.R. Zinner, and P. van Kerrebroeck, "Good urodynamic practices: uroflowmetry, filling cystometry, and pressure-flow studies," Neurourology and Urodynamics, vol.21, no.3, pp.261 – 274, 2002.

[17] E. van Waalwijk van Doorn, K. Anders, V. Khullar, S. Kulseng-Hanssen, F. Pesce, A. Robertson, D. Rosario, and W. Schäfer, "Standardisation of ambulatory urodynamic monitoring: Report of the standardisation sub-committee of the international continence society for ambulatory urodynamic studies," Neurourology and Urodynamics, vol.19, no.2, pp.113–125, 2000.

[18] Y. Kobayashi, S. Kobayashi, K. Okuda, K. Sakanushi, Y. Takeuchi, and M. Imai, "Synthesizable HDL generation method for configurable VLIW processors," Proceedings of the Asia and South Pacific Design Automation Conference (ASPDAC '04), pp.842–845, June 2004.

[19] L. Benini and G.D. Micheli, "Transformation and synthesis of FSMs for low-power gated-clock implementation," in Proc. International Symposium on Low Power Electronics and Design (ISLPED '95), pp.21–26, 1995.

[20] A. Chattopadhyay, B. Geukes, D. Kammler, E.M. Witte, O. Schliebusch, H. Ishebabi, R. Leupers, G. Ascheid, and H. Meyr, "Automatic ADL-based operand isolation for embedded processors," in Proc. Design Automation and Test in Europe (DATE '06), pp.600–605, 2006.

[21] P. Babighian, L. Benini, and E. Macii, "A scalable algorithm for RTL insertion of gated clocks based on ODCs computation," IEEE Trans. Computer-Aided Design, vol.24, no.1, pp.29–42, Jan. 2005.

[22] "Power compiler." http://www.synopsys.com.

[23] L. Benini, G.D. Micheli, E. Macii, M. Poncino, and R. Scarsi, "Symbolic synthesis of clock-gating logic for power optimization of synchronous controllers," ACM Trans. Des. Autom. Electron. Syst., vol.4, no.4, pp.351–375, 1999.

[24] K.S. Kim, Y.S. An, J.H. Seo, and C.G. Song, "Ambulatory urodynamics monitoring system using personal digital assistance," in Proc. International Conference on Multimedia and Ubiquitous Engineering, pp.122–125, 2008.

[25] R. Tan, T. McClure, C. Lin, D. Jea, F. Dabiri, T. Massey, M. Sarrafzadeh, M. Srivastava, C. Montemagno, P. Schulam, and J. Schmidt, "Development of a fully implantable wireless pressure monitoring system," Journal of Biomedical Microdevices, vol.11, no.1, pp.259–264, 2009.

[26] M. Damaser, G. Andros, J. Walter, L. Schlehahn, K. Brzezinski, J. Wheeler, and D. Hatch, "Home monitoring of bladder pressure in pediatric patients with spina bifida," in Proc. IEEE International Conference on Engineering in Medicine and Biology Society, pp.1915–1918 vol.5, 1997.

[27] J.J. Pel and R. van Mastrigt, "Non-invasive measurement of bladder pressure using an external catheter," Journal of Neurourology and Urodynamics, vol.18, no.5, pp.455–469, 1999.

[28] E. Costantini, L. Mearini, S. Biscotto, A. Giannantoni, V. Bini, and M. Porena, "Impact of different sized catheters on pressure-flow studies in women with lower urinary tract symptoms," Journal of Neurourology and Urodynamics, vol.24, no.2, pp.106–110, 2004.

[29] P. Fletter, S. Majerus, P. Cong, M. Damaser, W. Ko, D. Young, and S. Garverick, "Wireless micromanometer system for chronic bladder pressure monitoring," in Proc. International Conference on Networked Sensing Systems (INSS '09), pp.1 –4, 2009.

[30] S.J.A. Majerus, P.C. Fletter, M.S. Damaser, and S.L. Garverick, "Low-power wireless micromanometer system for acute and chronic bladder-pressure monitoring," IEEE Trans. Biomedical Engineering, vol.58, no.3, pp.763 –767, 2011.

[31] C.C. Wang, C.C. Huang, J.S. Liou, Y.J. Ciou, I.Y. Huang, C.P. Li, Y.C. Lee, and W.J. Wu, "A mini-invasive long-term bladder urine pressure measurement asic and system," IEEE Trans. Biomedical Circ. and Syst., vol.2, no.1, pp.44–49, March 2008.

[32] M. Mueller, A. Wortmann, S. Simon, M. Kugel, and T. Schoenauer, "The impact of clock gating schemes on the power dissipation of synthesizable register files," in Proc. the IEEE International Symposium on Circuits and Systems (ISCAS '04), pp.II – 609–12 Vol.2, may 2004.

[33] J.L. Hennessy and D.A. Patterson, Computer Architecture: A Quantitative Approach, Morgan Kaufmann Publishers, Inc., California, 1990.

[34] P.M. Sailer and D.R. Kaeli, The DLX instruction set architecture handbook, Morgan Kaufmann Publishers, Inc., California, 1996.

[35] R.F. Cmelik, S.I. Kong, D.R. Ditzel, and E.J. Kelly, "An analysis of MIPS and SPARC instruction set utilization on the SPEC benchmarks," in Proc. international conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '91), pp.290–302, 1991.

[36] S.D. Wachter and J.J. Wyndaele, "Frequency-volume charts: A tool to evaluate bladder sensation," Journal of Neurourology and Urodynamics, vol.22, no.7, pp.638–642, 2003.

[37] J. Nordling, P. Abrams, K. Ameda, J. Andersen, J. Donovan, D. Griffiths, S. Kobayashi, T. Koyanagi, W. Schfer, S. Yalla, and A. Mattiasson, "Outcome measures for research in treatment of adult males with symptoms of lower urinary tr act dysfunction," Neurourology and Urodynamics, vol.17, no.3, pp.263–271, 1998.

[38] D. Rowant, E.D. James, A.E.J.L. Kramer, A.M. Sterling, and P.F. Suhel, "Urodynamic equipment: technical aspects," Journal of medical engineering & technology, vol.11, no.1, pp.57–64, 1987.

[39] I. Kimura and T. Itakura, "Development of pressure sensor for blood pressure monitors," Omron Tech., vol.41, no.2, pp.138–143, 2001. (in Japanese).

[40] T.M. Vo, Y. Kuramochi, M. Miyahara, T. Kurashina, and A. Matsuzawa, "A 10-bit, 290 fJ/conv. steps, $0.13mm^2$, zero-static power, self-timed capacitance to digital converter," in Proc. International Conference on Solid State Devices and Materials (SSDM '09), October 2009.

[41] T. Yamagishi, K. Matsunaga, D.T.N. Huy, M. Miyahara, and A. Matsuzawa, "A 6.5 uw inverter based self biasing tranceiver," in Proc. IEICE Society Conference, pp.C–12–21, 2009. (in Japanese).

[42] "Opencores." http://www.opencores.org/.

[43] K. Tanaka, Y. Kuramochi, T. Kurashina, K. Okada, and A. Matsuzawa, "A 0.026 mm^2 capacitance-to-digital converter for biotelemetry applications using a charge redistribution technique," in Proc. IEEE Asian Solid-State Circuits Conference (ASSCC '07), pp.244–247, 2007.

[44] T.M. Vo, Y. Kuramochi, M. Miyahara, T. Kurashina, and A. Matsuzawa, "Asynchronous differential capacitance-to-digital converter for capacitive sensors," in Proc. Workshop on Synthesis And System Integration of Mixed Information technologies (SASIMI '09), March 2009.

[45] M. Ghovanloo and S. Atluri, "An integrated full-wave cmos rectifier with built-in back telemetry for RFID and implantable biomedical applications," IEEE Trans. Circ. and Syst. I, vol.55, no.10, pp.3328–3334, Nov. 2008.

[46] U.M. Jow and M. Ghovanloo, "Modeling and optimization of printed spiral coils in air, saline, and muscle tissue environments," IEEE Trans. Biomedical Circuits and Systems, vol.3, no.5, pp.339–347, Oct. 2009.

[47] Y. Chan, M.H. Meng, K.L. Wu, and X. Wang, "Experimental study of radiation efficiency from an ingested source inside a human body model*," in Proc. International Conference of the Engineering in Medicine and Biology Society, pp.7754–7757, 2005.

[48] K. Domdouzisa, B. Kumarb, and C. Anumbaa, "Radio-frequency identification (rfid) applications: A brief introduction," Advanced Engineering Informatics, vol.21, no.4, pp.350–355, 2007.

[49] J. Moore, "A formal model of asynchronous communication and its use in mechanically verifying a biphase mark protocol," Formal Aspects of Computing, vol.6, pp.60–91, 1994.

[50] IEEE Standard for Long Wavelength Wireless Network Protocol, IEEE Standard 1902.1, 2009.

[51] S. Ahmed and T. Kwasniewski, "Overview of oversampling clock and data recovery circuits," in Proc. Canadian Conference on Electrical and Computer Engineering (CCECE '05), pp.1876–1881, May 2005.

[52] T. Wong and J. Shea, "Multi-dimensional parity-check codes for bursty channels," in Proc. IEEE International Symposium on Information Theory (ISIT '01), p.123, 2001.

[53] T. Wong, J. Shea, and X. Li, "Using multi-dimensional parity-check codes to obtain diversity in rayleigh fading channels," in Proc. IEEE Global Telecommunications Conference (GLOBECOM '01), pp.1210–1214 vol.2, 2001.

[54] Z. Hu, A. Buyuktosunoglu, V. Srinivasan, V. Zyuban, H. Jacobson, and P. Bose, "Microarchitectural techniques for power gating of execution units," in Proc. International Symposium on Low Power Electronics and Design (ISLPED '04), pp.32–37, 2004.