Reconfigurable Parallel Image Processing System Using Three-Dimensional LSI

Mitsumasa Koyanagi, Takeaki Sugimura, and Tetsu Tanaka

Department of Bioengineering and Robotics, Graduate School of Engineering, Tohoku University
6-6-01 Aza Aoba, Aramaki, Aoba-ku, Sendai 980-8579, Japan.

Abstract

We have proposed a new reconfigurable parallel image processing system in which three-dimensional (3D) LSI is used. A new dynamical multi-context reconfiguration scheme is employed in this system. This scheme can decrease the number of instructions between a reconfigurable processing element array and control RISC processor, and also enable efficient execution of various image processing functions. The proposed image processing system consists of image sensor, frame memory, the coarse-grain dynamically reconfigurable image processor and RISC processor for control and reconfiguration. The reconfigurable processor is tightly coupled with frame memory which is achieved by using 3D LSI. It gives rise to a fully scalable parallel image processing architecture which can provide a new robot vision system with high speed and high flexibility. To achieve such reconfigurable parallel image processing system, we have developed a new three-dimensional integration technology called a super chip integration.

1. Introduction

With the developments of robot technology, it is expected that robots can work in the same environment as the working environment of human. To achieve this objective, advanced robot vision systems with high performance are necessary for recognizing moving objects and obstacles with real-time image processing. To realize such kind of novel robot vision system, dynamically reconfigurable system [1][2][3] is an attractive alternative because it enables highly parallel image processing with great flexibility.

Conventionally, FPGAs (Field Programmable Gate Arrays) were used for reconfigurable computing, which enable to implement gate level logic circuits. Though the bit-oriented computation using FPGAs has great flexibility of the implementation, there are several disadvantages too. For example, a number of interconnection deteriorates the performance. The configuration time of FPGAs is too long to reconfigure the configuration data dynamically. Instead of bit level implementation, coarse-grained reconfigurabe archtectures which have 8 bit or 16 bit datapath have been proposed. The size of the datapath is suitable for image processing, and also reduced configuration data enables the dynamical reconfiguration with multi-context configuration data. MorphoSys[1], REMARC[2], and DRP[3] are examples of the coarse-gained reconfigurable architectures.

On the other hand, there are two problems with the image processing using the conventional dynamically reconfigurable system. First problem is the lack of I/O bandwidth between an image sensor and the image processing system. Since the image processing with larger image size and higher frame rate requires high speed and scalable data I/O, conventional sequential connection is not appropriate for such vision system. Second problem is the dynamic reconfiguration and function execution trade-off with configuration data grain size. Usually, image processing functions for robot vision such as filtering and optical flow extraction are divided into several stage configuration data, and dynamically reconfigured during execution. If a function is divided into small stages, it requires huge number of reconfiguration times with small amount of configuration data. On the other hand, dividing into large stages requires long reconfiguration time. Therefore, efficiency of the execution is greatly affected by the grain size of the reconfiguration.

To solve these problems, we propose a hierarchical multi-context configuration scheme with ring configuration register file and three-dimensional structure which can optimize the number of contexts for reconfiguration. In addition, the three-dimensional integration technology [4][5][6][7] enables parallel I/O among image sensor, data memory and reconfigurable image processor. It can realize fully scalable parallel architecture for image processing.

2. Architecture of the Proposed Image Processing System

Configuration of the overall image processing system is shown in Fig.1. It consists of CMOS image sensor, A/D converter, frame memory, and reconfigurable image processing unit. Each component is connected using vertical interconnection with three-dimensional integrated circuit.

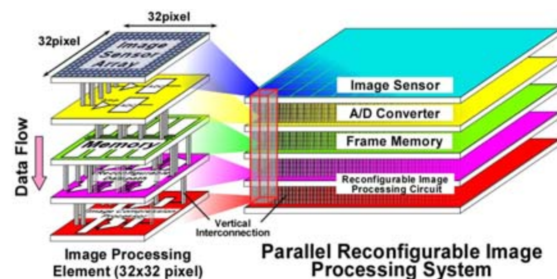


Fig. 1. Configuration of the overall proposed image processing system

2.1 Configuration of Dynamically Reconfigurable Image Processor

As the reconfigurable image processing unit of the proposed system, dynamically reconfigurable image processor has been designed. Configuration of the designed processor is shown in Fig. 2. The reconfigurable image processor consists of multiple PEs (Processing Elements), data memory, interconnection network, configuration memory, branch control unit, and RISC processor for control. As an evaluation system, the system with four PE for 64×64 pixel (32×32pixel/PE) image has been designed.

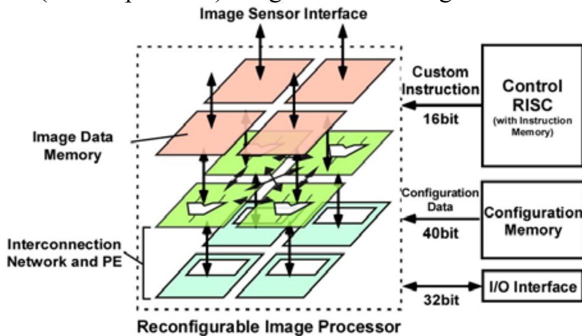


Fig. 2. Configuration of the designed dynamically reconfigurable image processor

Input image is divided into a number of image blocks and stored into memories. Each memory is tightly connected with the corresponding processing element for parallel processing by using vertical interconnection. Between the processing element and the memory, there is an interconnection network to communicate with neighborhood PE and memories. There are only adjacent image block connections through the interconnection network, and the architecture of the proposed image processing system is fully scalable. Therefore, image size enlargement such as VGA and XGA can be realized with the same architecture.

2.2 Design of Components

2.2.1 Processing Element

Configuration of the processing element is shown in Fig. 3. The PE consists of MAC, ALU, data register file, data memory interface, and ring configuration register file controlled by branch control unit. The PE can store 32 context configuration data for evaluation system and can dynamically change these contexts in every clock cycle. The configuration sequence is controlled by the branch control unit using context pointer. All PE are connected to data memory through the memory interconnection network, and all frame buffers can be connected to image sensor interface. Therefore, a scalable and flexible architecture with high performance is realized in the implemented system by employing three-dimensional configuration.

2.2.2 Branch Control Unit

The branch control unit stores the sequence and jump condition of configuration data context, and broadcast the context number that is executed on next cycle to all PEs, memories, and interconnection network. The branch control unit consists of 16 jump condition register and context number controller.

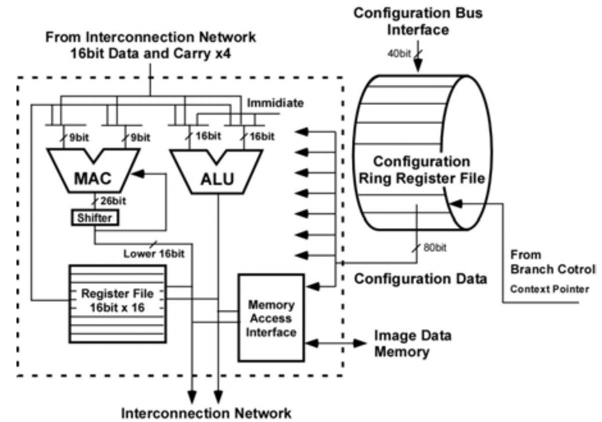


Fig. 3. Configuration of the processing element with ring configuration register file

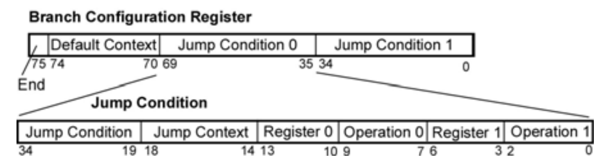


Fig. 4. Branch control unit configuration data

The configuration data format of the branch control unit is shown in Fig. 4. The configuration data can hold four jump condition for each context. If the jump condition is equal to a value of the designated jump condition register, next context is changed to the value which is stored in jump context.

2.2.3 Control RISC Processor

Control RISC processor controls reconfiguration of the proposed image processing system. The RISC processor dispatch the reconfigurable custom functions for reconfigurable PE array, and send instruction for it. Configuration of the RISC processor is shown in Fig. 5. In this system, image processing operations for reconfigurable processor and ordinary RISC pipeline operations are executed simultaneously. Therefore, an instruction scheduling mechanism to protect the data dependency between reconfigurable PE array part and RISC pipeline is necessary for the RISC processor.

The RISC processor has a programmable instruction decode unit, which enables implementation of various image processing functions for reconfigurable image processor. The instruction set architecture of the RISC processor can be divided into three categories, configuration instructions, image processing instructions for reconfigurable processor and general arithmetic instructions for RISC pipeline. These instructions are decoded in the instruction decoder and then executed. Instructions for reconfigurable processor are sent

to function table which stores configuration memory index address and number of context set of custom instruction, and then the custom instruction is sent to reconfigurable processor array. Dependencies of the image processing instruction and the arithmetic instruction are described in the instruction script and referred when executing. Thus, data dependency can be protected during the instruction execution.

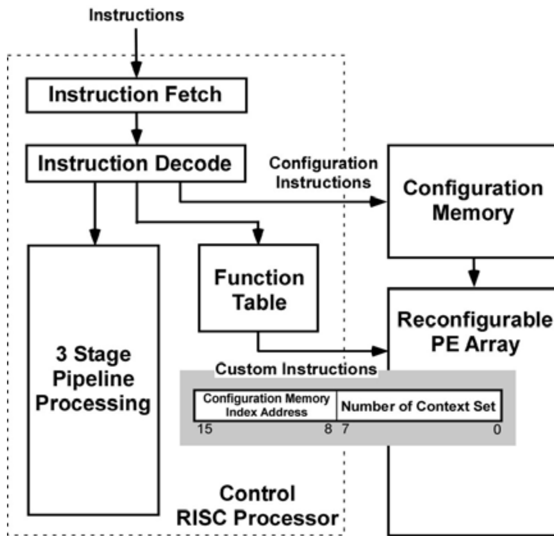


Fig. 5. Configuration of the RISC processor for reconfiguration control

2.2.4 Image Data Memory with Interconnection Network

The image data memory consists of a number of dual port memory and interconnection network among those memories with interface for PE. The configuration of the memory and interconnection network is shown in Fig. 6. All processors can access the corresponding image data memory simultaneously in parallel, and also can access adjacent image blocks through the MUX based interconnection network for boundary processing between the divided blocks.

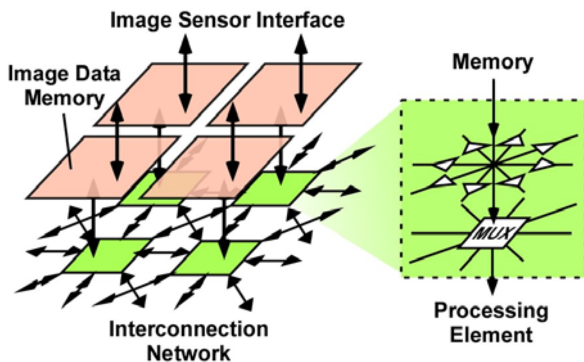


Fig. 6. Image data memory and interconnection network

3. Hierarchical and Dynamical Multi-Context Reconfiguration Scheme

On the first section, it was described that the dynamically reconfigurable system has trade-off between the grain of the divided image processing function and the reconfiguration time. To optimize the efficiency of reconfiguration and execution of image processing functions, hierarchical and dynamical multi-context reconfiguration scheme has been proposed.

Image processing functions such as filtering and optical flow extraction can be divided into multiple parts of operation loop that have to be applied for all pixels. To execute such an operation loop with the reconfigurable PE array, multiple contexts configuration data have to be used repeatedly. Therefore, these contexts should be put together into one "contexts set" and stored into the configuration memory together. In this way, image processing functions are divided into multiple contexts set by extracting the operation loop.

Configuration of the proposed configuration architecture is shown in Fig. 7. As described in the previous section, 16-bit custom instruction from control processor to reconfigurable PE array consists of the configuration memory index address and number of context sets. In the configuration memory, divided image processing functions are converted to context sets data and stored in it. Each context set has context set header, which contains information of the context set such as the number of context, the configuration broadcast for all PEs, and required clock cycles for configuration. Address of the context set header is stored into configuration memory index to invoke context sets for configuration. As a result, a custom instruction from control RISC processor is dispatched in two stages hierarchically, and converted into a number of configuration data sequence like micro program decoding. Therefore, high bandwidth for instruction between control RISC processor and reconfigurable PE array is not needed.

The configuration data of the context set indicated on configuration memory index is transferred to ring configuration register file in each components through the 40bit configuration bus. If configuration data for all PE are completely same, configuration data is transferred in broadcast mode, which can broadcast the same configuration data for all PEs, and then configuration time can be reduced. Configuration data transfer from configuration memory to configuration register is continued until the configuration register is full, and several context sets can be stored into the configuration register file. After each context set is transferred to the configuration register, configuration control put the start address of the context set on the configuration register file to the executable context set address FIFO. When the context set address is stored into the FIFO, execution control detects the contents of the FIFO and sends the address to the branch control unit as start offset of

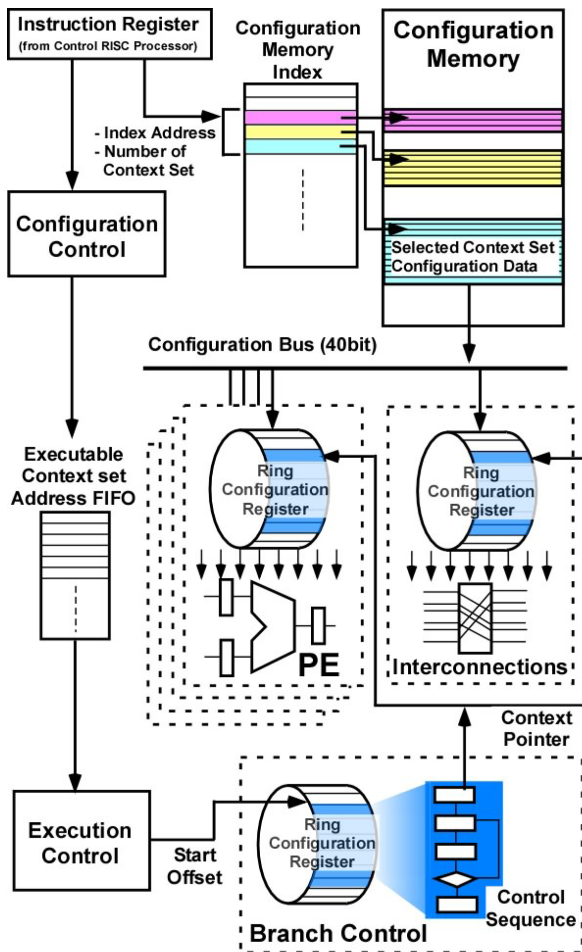


Fig. 7. Hierarchical Multi-Context Configuration Architecture

control sequence and start execution of the context set. During the execution, context pointer from the branch control points the context address which is executed next.

After the execution finished, context set start address is removed from FIFO and configuration control can store new context set from configuration memory into configuration register file. This reconfiguration is executed dynamically during the other context execution. As described above, context set consists of loop operation for all pixels in the image block. For most context sets, execution time is longer than configuration time and hence configuration time is overlapped and terminated.

To realize the efficient reconfiguration and execution of the context set, the more number of the context set which can be stored into the ring configuration register file invokes the less configuration time which is not overlapped with context set execution. However, there is a trade-off among the number of configuration register and area, delay, and power consumption issue. Parts of these problems are investigated on the next section.

4. Parameter of Architecture and Performance Analysis

The reconfigurable image processor has been designed in Verilog-HDL and evaluated with RTL simulation. On this section, we will discuss the trade-off between area and number of contexts in the configuration register file, and also explain the area reduction by three-dimensional structure. The breakdown of the PE cell area, which consists of the data path and 32 contexts configuration register, is shown in Fig. 8. As is clear from this figure, nearly 70% of the cell area is occupied by configuration register file. Therefore, the number of contexts of the configuration register file makes great effects with the area and performance of the whole system. The performance for several image processing functions is evaluated from the result of RTL simulation. Fig. 9 shows the execution clock cycles for thresholding, 3×3 convolution filter, and optical flow extraction with change of the number of contexts. Input image is 64×64 pixel and divided into four pieces of image blocks for four PEs.

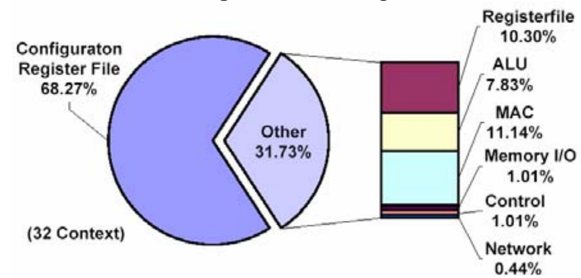
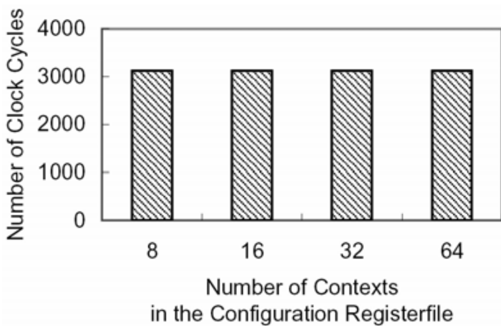


Fig. 8. Breakdown of the processing element cell area

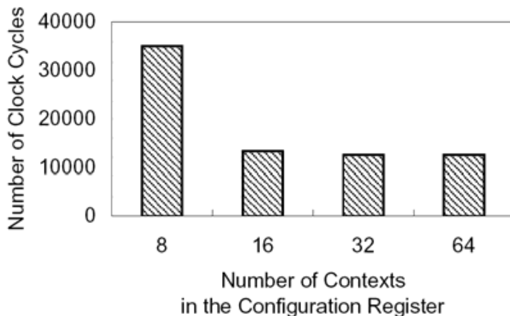
For the optical flow extraction, template size is 5×5 pixel and search range is 12×12 pixel. All functions have been successfully implemented except the optical flow extraction for 8 and 16 context architecture which was not performed due to lack of contexts. From this figure, it has been shown that the context size becomes larger and the performance also becomes higher especially for complex image processing functions that need a lot of contexts and context sets. It is due to the redundant clock cycles during the context sets switch, loading time, and storing time for the data between PE and memory for data saving. Therefore, the reconfigurable system for image processing system which has to process complex image processing functions such as pattern matching, object recognition, and object tracing has to have more than 32 context configuration register for it.

On the other hand, the large size configuration register file invokes the area increase. Three-dimensional integration mitigates this problem because three-dimensional integration for the processing element can suppress the area increase. An area evaluation for the conventional two-dimensional circuit and the three-dimensional circuit is shown in Fig. 10. This figure shows the placement and routing result with 0.35 μm CMOS triple-metal technology. The area is scaled to eight context architecture with the conventional two-dimensional

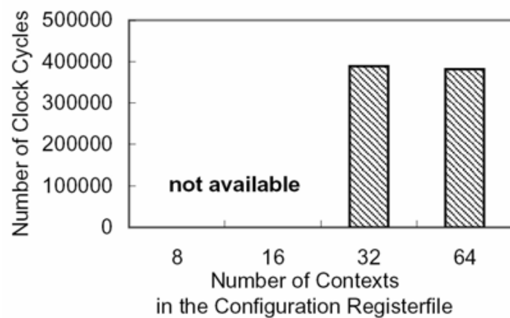
implementation. For three-dimentional integration, datapath and configuration register are divided in two layers. In the case of the architecture with 64 and 32 contexts, the configuration register is divided into two layers to optimize the area between the layers. From this figure, the area with three-dimensional integration is less than half size of conventional 2D one for the proposed architecture, which has more than 32 context configuration register. As the result, not only to realize the scalability of the proposed architecture and I/O bandwidth, three-dimentional integration also can relax the trade-off between the application complexity and area.



(a) Thresholding function (6 contexts)



(b) 3×3 Convolution filter function (88 contexts)



(c) Optical flow extraction (1092 contexts)

Fig. 9. The performance analysis with implementation of image processing functions (Input image : 64× 64pixels, for four PE)

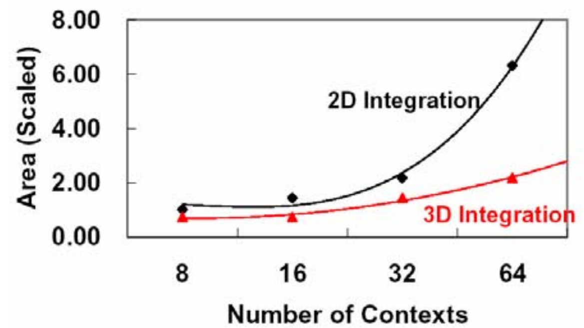


Fig. 10. Area evaluation with two-dimensional and three-dimensional integration for change of number of contexts

5. Super Chip Integration Technology for New 3D LSI

Figure 11 shows a cross-sectional view of 3D LSI fabricated by super chip integration technology[8]-[19]. The fabrication process sequence for super chip integration technology is as follows. First, the deep trenches for buried interconnections are formed through the thick interlayer dielectrics into Si substrate by using RIE (Reactive Ion Etching) and filled with conductive materials such as polycrystalline Si (poly-Si), W or Cu after the formation of dielectric layer onto the trench surface. These KGDs with buried interconnections for the first chip layer are bonded onto the supporting LSI wafer after alignment using a self-assembly technique and the adhesive is injected underneath the KGDs. Then, KGDs are thinned from the backside by the mechanical grinding and chemical mechanical polishing (CMP) to expose the bottom of buried interconnections and the metal microbumps are formed onto the bottom of buried interconnections. Next, the KGDs with buried interconnections for the second chip layer are flip-chip-bonded onto the KGDs for the first chip layer again using a self-assembly technique. By repeating this sequence, we can obtain a new 3D LSI called a super chip as shown in Fig.12. The most striking feature of this super chip is that various kinds of thin chips with different sizes such as MEMS chip, sensor chip, CMOS RF-IC, MMIC, power IC, control IC, analog LSI, and logic LSI are vertically stacked.

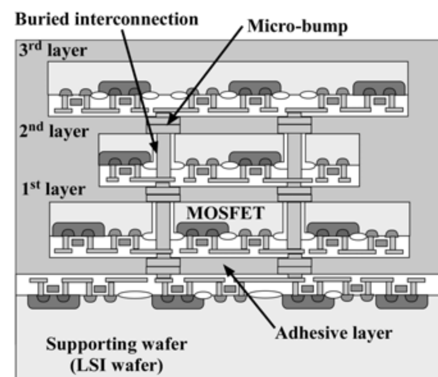


Fig.11. Cross-sectional structure of 3D LSI.

Thus, super chip integration technology can provide new advanced LSIs with small form factor, high packing density, high performance, low power consumption, and new functionality. We have developed several key technologies of buried interconnection formation, chip alignment and bonding by self-assembly technique, adhesive injection, chip thinning and planarization for super chip integration technology. By using these key technologies, three-layer stacked test chips with three kinds of chip sizes were successfully fabricated as shown in Fig.13.

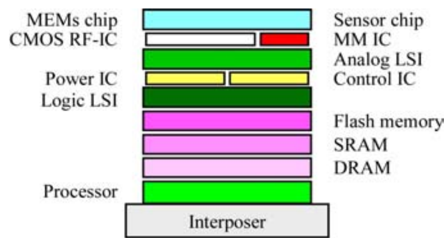


Fig.12. Conceptual structure of 3D super chip.

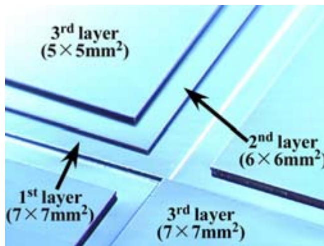


Fig13. Photomicrographs of three-layer stacked super chips with different chip sizes.

6. Conclusions

A new dynamical multi-context reconfiguration scheme for reconfigurable image processing architecture with three-dimensional structure has been proposed and evaluated. It can decrease the number of instructions and bandwidth between a reconfigurable PE array and control RISC processor, and realize efficient execution of various image processing functions.

As examples of the image processing functions, thresholding, 3×3 convolution filter, and optical flow extraction are implemented and evaluated with RTL simulation of the designed architecture. As a result, proposed architecture with three-dimensional structure is more efficient than that with the convensional structure if the number of context in the configuration register file is more than 32 contexts.

We have developed a new three-dimensional (3D) integration technology called super chip integration technology using a novel self-assembly technique that allows us to stack various kinds of KGDs with different chip size and chip thickness which are fabricated using different technologies.

7. References

[1] H. Singh, M. Lee, G. Lu, F. J. Kurdahi, N. Bagherzadeh, and E. M. Chaves Filho, IEEE Trans. Computers, Vol.49, No.5, pp.465-481, May 2000

[2] T. Miyamori, and K. Olukotun, IEICE Trans. Inf. & Syst., Vol.E82-D, No.2, pp.389-397, Feb. 1999

[3] H. Amano, and A. Jouraku, Proc. COOL Chips VI, pp. 321-334, Apr. 2003

[4] H.Kurino, K.W.Lee, T.Nakamura, K.Sakuma, K.T.Park, N.Miyakawa, H.Shimazutsu, K.Y.Kim, K.Inamura and M.Koyanagi , Technical Digest of the International Electron Devices Meeting, pp879 – 882, 1999

[5] D. Kawae, T. Nakamura, T. Morooka, Y. Igarashi, J. C. Shim, H. Kurino, and M. Koyanagi, Proc. SSDM, pp.314-315, 2002

[6] H. Kurino, Y. Nakagawa, T. Nakamura, Y. Yamada, K. W. Lee, and M. Koyanagi, IEICE Trans. Electronics, Vol.E84-C, No.12, pp.1717-1722, 2001

[7] M. Koyanagi, Y. Nakagawa, K. W. Lee, T. Nakamura, Y. Yamada, K. Inamura, K. Park, H. Kurino, Proc. of the 2001 IEEE International Solid State Circuits Conference, pp.270-271, 2001

[8] H. Takata, M. Koyanagi, International Semiconductor Device Research Symposium, 327-330, 1991.

[9] T. Matsumoto, M. Koyanagi *et al.,* SSDM (International Conf. on Solid State Devices and Materials), 1073-1074, 1995

[10] T. Matsumoto, M. Koyanagi *et al.,* SSDM (International Conf. on Solid State Devices and Materials), 460-461, 1997.

[11] M. Koyanagi *et al.,* IEEE MICRO, 18 (4), 17-22, 1998

[12] H. Kurino, M. Koyanagi *et al.,* IEDM, 879-882, 1999

[13] K. W. Lee, M. Koyanagi *et al.,* IEDM, 165-168, 2000

[14] M. Koyanagi *et al.,* ISSCC, 270-271, 20).

[15] H. Kurino, M. Koyanagi *et al.*, IEICE Trans. on Electronics, E84-C (12), 1717-1722, 2001

[16] T. Ono, M. Koyanagi *et al.,* IEEE COOL Chips, 186-193, 2002

[17] T. Sugimura, M. Koyanagi *et al.*, IEEE International Conf. on Field- Programmable Technology (ICFPT), 372-374, 2003

[18] J. Deguchi, M. Koyanagi *et al.,* Jpn. J.A.P., 1685-1689, 2004

[19] T. Fukushima, M. Koyanagi *et al.*, SSDM (International Conf. on Solid State Devices and Materials) 64-65, 2005