Decoder with the Dynamic CMOS Matrix

Lukáš Paločko, Jan Broulím, Jan Moldaschl

***Abstract* — This paper presents the Full Custom Design technique in the IC digital design, which is used to achieve the maximum performance or minimum power. The technique was applied on the decoder of the ARM1 register file. Presented research combines a static CMOS logic and a dynamic logic, which has higher speed than the equivalent static family. The validation of the design is made by the SPICE3 simulator (BSIM3 model) considering RC parasitic values of metal wiring. Our results demonstrate the size of the register file decoder and behavior of the dynamic matrix.**

***Keywords* — decoder, ARM, register file, dynamic CMOS matrix.**

I. Introduction

AN Integrated circuit (IC) can be made by many kinds of design processes. We are focused on the CMOS process, which can be built by a different logic type. One of these types is a conventional CMOS, where for each n-type devices there are a complementary counterparts p-type devices [3]. It means that the logic function is implemented using two networks (PUN – Pull-Up Network and PDN – Pull-Down Network). The dynamic logic family was developed for increasing speed and reduction of used area. Due to the reduced number of transistors in the circuitry the load capacitance is reduced [6]. Then the element, which represents the logic function, can be faster than an equivalent circuit in the conventional (static) CMOS logic [4], but this implementation of logic needs a different timing scheme. The operation of the circuitry has to be divided into two phases. Due to the advantages mentioned above (speed and area), we have developed dynamic matrix as a part of decoder for the register file.

In section II a computer-aided design and MOSIS technology for our research are discussed.

Section III is divided into two parts and gives some information about architecture ARMv1 and ARMv2.

Section IV is focused on our research and describes an organization and implementation of the decoder. The detailed implementation of dynamic matrix is discussed.

Section V describes how the full-custom layout was validated.

Lukas Palocko is with the Faculty of Electrical Engineering, University of West Bohemia, Univerzitni 22, 30614 Pilsen, Czech Republic (e-mail: lpalocko@kae.zcu.cz).

Jan Broulim is with the Faculty of Electrical Engineering, University of West Bohemia, Univerzitni 22, 30614 Pilsen, Czech Republic (e-mail: broulim@kae.zcu.cz).

Jan Moldaschl is with the Faculty of Electrical Engineering, University of West Bohemia, Univerzitni 22, 30614 Pilsen, Czech Republic (e-mail: moldy@kae.zcu.cz).

Section VI summarizes presented research and gives information of future work.

II. Design Flow

A “full-custom” design flow is used for our research, which is shown in figure 1. The goal of is to completely customize all aspects of the design.

We started with a specification. First step is known as a schematic capture, where an element of logic or transistors are manually placed and connected. Second step is verification. In third step the layout is drawn. Final step is verification of the layout [2].

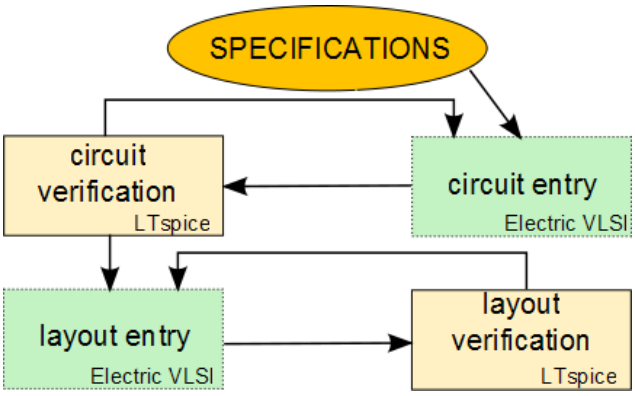

Fig. 1. Design flow

A. The Electric VLSI Design System

The most favor computer-aided design (CAD) tools for the IC Design are created by Cadence and Mentor Graphics. These tools are powerful and very complex and of course expensive.

The Electric VLSI Design System is a CAD tool developed by Steve M. Rubin [9]. It is written in the Java language and runs under Windows, Linux and Macintosh OS. It is also GNU-licensed and the free distribution of the Electric can be downloaded from www.staticfreesoft.com. This environment integrates many tools like DRC, ERC, Simulation Interface, Placement and Routing, VHDL and Silicon Compilation and etc [10].

B. C5 Semiconductor process

Semiconductor process can be offered by many different foundries. Our design of the layout and simulations use vendor rules and SPICE models provided by MOSIS. The MOSIS service access SPICE parameters and electrical test data from specified foundry (IBM, TSMC and ON Semiconductor). Table 1 gives basic information about semiconductor process accessed by MOSIS.

TABLE 1: FABRICATION PROCESSES AVAILABLE THROUGH MOSIS

Vendor	*Feature size[um]*	*Process*
ON Semiconductor	1.5	ABN
	0.5	**C5**
TSMC	0.35	CL035, CM035
	0.25	CM025,CL025,CR025
	0.18	CM018,CL018,CR018

The C5 process family from ON Semiconductor is optimized for 5V applications. This non-silicided CMOS process has 3 metal layers and 2 poly layers.

The standard CMOS technology accessed by MOSIS is a single polysilicon, double metal. It currently offers only n-well processes (SCN). SCMOS options are used to designate projects that use additional layers (3M means, second via and third metal layer is added), see table 2 and table 3 [11].

TABLE 2: MOSIS SCMOS-COMPATIBLE MAPPINGS [11]

Foundry	*Process*	*λ[um]*	*Options*
ON Semi	C5F/N (0.5 um n-well)	0.35	SCN3M, SCN3ME
TSMC	0.35um 2P4M (3.3/5 V)	0.25	SCN4ME
TSMC	0.35um 1P4M (3.3/5 V)	0.25	SCN4M

TABLE 3: MOSIS SCMOS_SUB (SUB-MICRON)-COMPATIBLE MAPPINGS [11]

Foundry	*Process*	*λ[um]*	*Options*
ON Semi	C5F/N (0.5 um n-well)	0.30	SCN3M_SUBMS SCN3ME_SUBM
TSMC	0.35um 2P4M (3.3/5 V)	0.20	SCN4ME_SUBM
TSMC	0.35um 1P4M (3.3/5 V)	0.20	SCN4M_SUBM
TSMC	0.25um 5M1P	0.15	SCN5M_SUBM
TSMC	0.25um 6M1P	0.10	SCN6M_SUBM

C. MOSIS Layout rules

Our layout design was checked by the Design Rule Check (DRC) Tool includes Scalable CMOS design rules. The Scalable CMOS (SCMOS) is a set of logical layers with design rules and was developed by the MOSIS Service. The design is done by abstract SCMOS layers and lambda as the metric unit is used for this purpose. This is useful, because the same design can be handled to the new mapping and then new fabrication process can be chosen [11].

III. IMPLEMENTATION OF THE ARMV1 AND ARMV2 DATAPATH ARCHITECTURE

Architectures ARMv1 and ARMv2 were developed by Arcon Computers Ltd. with VLSI Technology Inc. as "silicon partner" [7]. Table 4 describes the dependence between the architecture and the implementation of the Core [5].

TABLE 4: DEPENDENT BETWEEN ARCHITECTURE AND THE IMPLEMENTATION OF THE CORE

Architecture	*Core*
ARMv1	ARM1
ARMv2	ARM2, ARM3
ARMv3	ARM6, ARM7

A. ARM Register File

The register file includes the register bank with three buses; two buses (A and B) are dynamic and used to read value from the register cell. Third bus is used to write data into a register. The write bus is static and overdrives the storage node in the cross coupled inverter pair by using NMOS pass transistor [5].

B. ARM pipelining

Operations of the ARM1 pipeline are spread to three stages. The first stage is FETCH, second stage is DECODE and third stage is EXECUTE [1]. The ARM execution unit is not pipelined, but some operations can be done by a single cycle execution [5].

IV. FULL-CUSTOM DESIGN

The design has been started with the specification of the process and the layout rules. The C5 process has been chosen to the research and this paper and the technology scale was set to 300 nm.

A. Structure of the Decoder for the Register File

A decoded operation is spread to two stages. The first stage uses a full static logic to select one bit (one register) for every port of the register file. There are four sub-blocks to enable the requested register (the mode and the register are selected from the instruction). The second stage includes the dynamic CMOS matrix as the decoder, which sets one of the outputs HIGH according to the output from the first stage. However, the information of selected mode is gated to ensure that right register is selected. This information is determined by the instruction decoder.

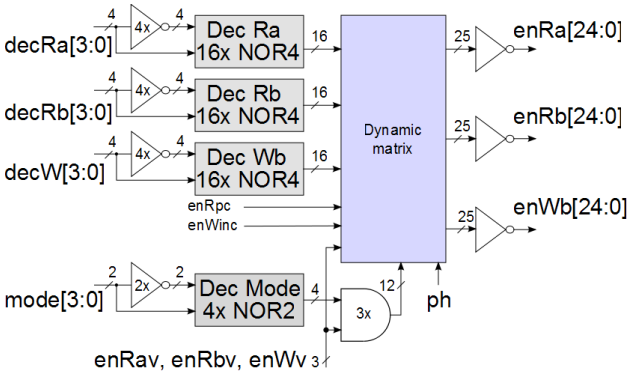

Fig. 2. Structure of the decoder

Structure of the decoder is presented in figure 2. Four sub-blocks are small decoders, which are built by the four

–input NOR gate. Then, there are three groups of AND gates. Every group includes four two-input AND gates to enable requested register. It is necessary because the instruction set and the instruction class is very complex. That means that same bits, which are used to select register, carry information about other instructions.

A. *The dynamic CMOS matrix*

This sub-block is built using the dynamic logic type. Then, output nodes must be precharged to high level for evaluating. It allows that all registers are disconnected from buses without additional gates. The transistor level design for one register is shown in figure 3.

Output nodes (sel_Ra, sel_Rb, sel_W) drive buffers, which are used to select the register from the register file. Inputs (en_Ra_a, en_Rb_a, en_W_a) are set by static NOR decoders.

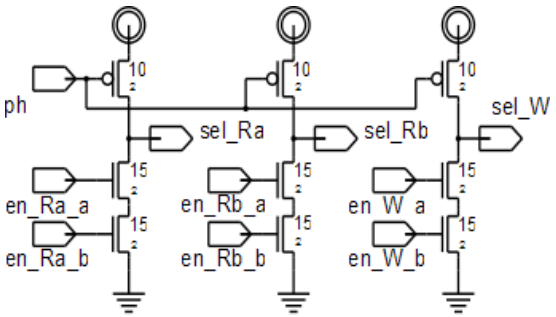

Fig. 3. The transistor level implementation of the dynamic matrix for one register

For this purpose, the special dynamic CMOS cell was designed by our team. Detailed layout of the dynamic matrix cell is shown in figure 4. The design occupies two levels of metal and one polysilicon layer. Ground runs from left to right across the cell. The power is connected to the power supply individually.

There are three PMOS transistors on the top and six NMOS transistors on the bottom of horizontal rows. P-channel devices allow charging output nodes during a pre-charge phase. Three groups of two N-channel devices are used to evaluate an operation.

B. *Layout of the decoder*

The general layout of the decoder as a macrocell is shown in figure 6. There are sub-blocks, which correspond to structure in figure 2, and the channel routers to make connection between sub-blocks. Metal interconnections are routed in three metal layers.

The first layer is used to make internal connections to transistors within the cell and also it is used to generate horizontal routing tracks in the routed channel. The second and third layer is used to connect sub-blocks.

The macrocell is folded by sub-blocks and the sub-block is folded by cells. The design is not using any standard cells. The cells are a “full custom” design.

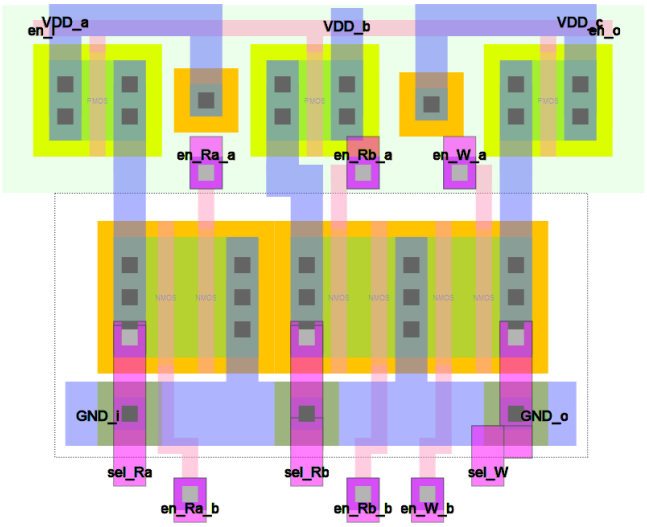

Fig. 4. Detailed layout of the dynamic matrix cell

V. Design Validation

The validation of the design is made by LTspice IV, which includes the SPICE simulator of electronic circuits. The Berkeley Short-channel IGFET Model (BSIM3) and parasitic values of metal wiring were used for the layout simulation.

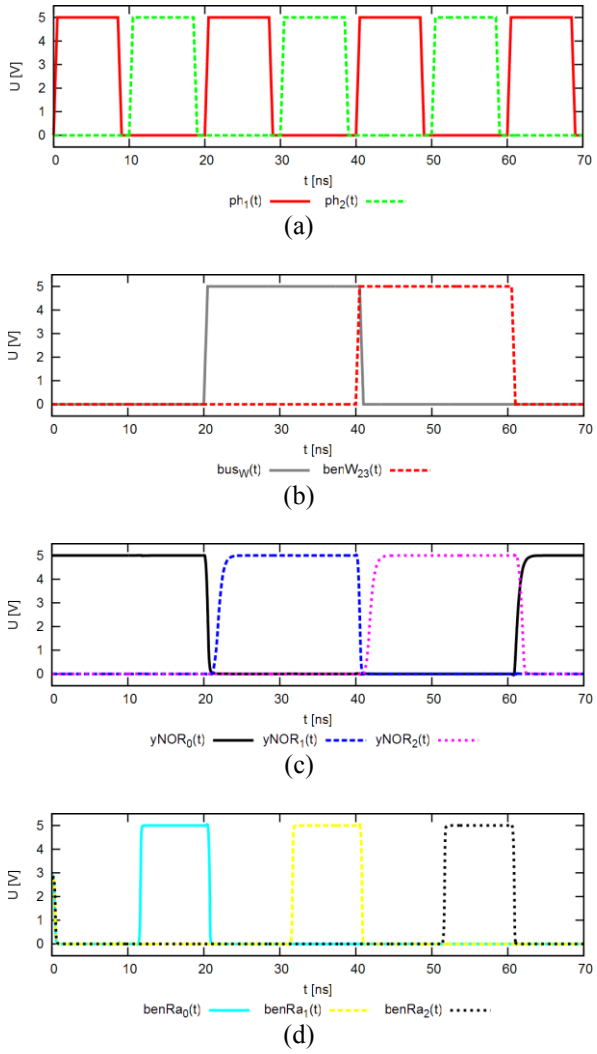

Fig. 5. Dependence of the internal and external signals.

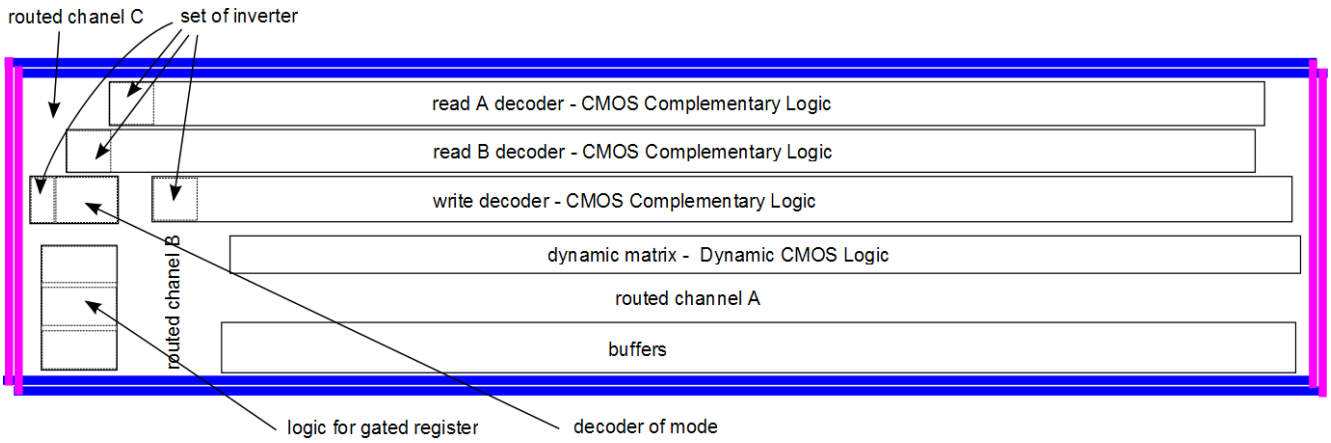

Fig. 6. Decoder floorplan

The timing of internal and external signals is shown in figure 5. There are four graphs to imagine how the full-custom (layout) design was validated. The graph (a) describes the two phase timing and the graph (b) gives information about selected register in time. Output signals of the static NOR decoder are presented in the graph (c). The final case illustrates signals to enable the register from the register file (see figure 3, (d)). The presented implementation was validated at 50MHz.

Our basic results demonstrate the size of the register file decoder and the number of the required transistors. This information is summarized in table 5.

TABLE 5: DESIGN SUMMARY

Design	*Number of transistors[-]*	*Area[λ]*
Full-custom	875	2810x704

VI. CONCLUSION

In this paper we presented the "full custom" design of the register file decoder. Today, many processors cores are based on standard cell design. The control logic and the datapath of the processors cores are built by Place and Route (P&R) Tools. Synthesized ASICs typically operate at lower frequency. The "full custom" design of digital CMOS IC is used for high-end microprocessors datapaths, memories, standard cells, RF designs.

We designed the decoder of the register file by using one of CMOS design methods. This method is used for high performance and low-power design. The future work will be focused on other CMOS design method and the "full-custom" design and the "cell-based" design will be compared (power consumptions, area, performance, time to market).

The behavior of the dynamic matrix is presented in the section V. The validation of the design was presented at 50MHz, but the design was successfully validated even at 100MHz.

ACKNOWLEDGMENT

This work was supported by the program Student Grant Project no. SGS-2012-019.

REFERENCES

[1] Stephan B. Furber. VLSI RISC Architecture and Organization, Marcel Dekker, Inc. New York, USA, 1989

[2] Clein, Dan. CMOS IC Layout: Concepts, Methodologies, and Tools, Boston: Newnes. 2000, ISBN: 9780750671941

[3] Etienne Sicard, Donia D. Bendhia. Basic of CMOS Cell Design, The MeGraw-Hill Companies , Inc., USA 2007

[4] Kiat-Seng Yeo, Kaushik Roy. Low-Power VLSI Sybsystems, The MeGraw-Hill Companies , Inc., USA 2005

[5] Steve Furber. ARM System-on-Chip Architecture. Addison-Wesley, 2nd edition, 2000.

[6] Rafati, R.; Fakhraie, S.M.; Smith, K.C., "Low-power data-driven dynamic logic (D3L) [CMOS devices]," Circuits and Systems, 2000. Proceedings. ISCAS 2000 Geneva. The 2000 IEEE International Symposium on , vol.1, no., pp.752,755 vol.1, 2000. doi: 10.1109/ISCAS.2000.857205

[7] Furber, S.B.; Wilson, A.R., "The Acorn RISC Machine Â¿ an architectural view," *Electronics and Power* , vol.33, no.6, pp.402,405, June1987 doi: 10.1049/ep.1987.0249

[8] Levy, Markus. "The History of The ARM Architecture: From Inception to IPO". Retrieved 14 March 2013..

[9] Rubin, Steven M. (1987), *Computer Aids for VLSI Design*, Addison-Wesley, Reading Massachusetts

[10] Steven M. Rubin, "Electric User's Manual, version 9.04", June 14, 2013, [Online]. Availible: http://www.staticfreesoft.com/jmanual/ [Accessed: June 14, 2013]

[11] The MOSIS Service, "MOSIS Scalable CMOS (SCMOS)" Wired, May 11, 2009, [Online]. Availible: http://www.mosis.com/files/scmos/scmos.pdf [Accessed: May 11, 2009].