Power optimization of the CV32A6 RISC-V soft-core

Yasser Ajellabi[1],Willy Perrin[1],Adrien Stoecklin[1],Hamady Ka[1],Freddy Anstotz[2]

***Abstract*— The power optimization of processors is nowadays some of the challenging topics that need to be understood and studied in order to find effective solutions. It's a real concern for the cutting edge technologies of today. That's why we're being part of the Thales Group competition as RiscEtMorty team for the power optimization of the CVA32A6 RISV-V soft core which is implemented in an FPGA Zybo-Z7 architecture of the Xilinx Zinq-7000 family. The hardware description language is System Verilog. Some power reduction techniques have been studied but just some of them is implemented by our team.**

***Index Terms*— RISC-V, ISA, FPGAs, SRAMs, Power Optimization, Reduction Techniques, Dynamic Power, Memory Read Access, Memory Splitting.**

I. INTRODUCTION

RISC-V processors are new, open and free ISA (Instruction Set Architecture) developed by the University of California and its foundation is supported by many companies. It's a RISC-based architecture that has the instructions of a standard RISC processor but with some optional extensions of this one. It's a base integer ISA with no branch delays and the support for optional variable length encodings. RISC-V are known by their fast growth and adoption and also the fact that they are supported by GCC, Clang, Linux Kernel but also several RISC-V simulators and emulators. The target Hardware is an FPGA Zybo-Z7 from Xilinx.

FPGAs (Field Programmable Gate Arrays) are semiconductor devices that are based around a matrix of configurable logic blocks connected via programmable interconnects. The key feature of FPGAs is that they are reprogrammable respecting application requirements which highlight their difference with the ASIC boards. They are made of thousand of Configurable Logic Blocks (CLBs) connected with a network of Interconnects and they are made up also of Input Output Blocks(IOBs). CLBs can be LUTs(Look Up Tables) which are function generators based on SRAM memories which realize logic tables,they contain also Flip-Flops ,Multiplexers ad so on. FPGAs can also contain others components like PLLs, Memory Controllers, DSPs and High Speed Transceivers. Nowadays FPGAs can also embed Hard Processors or even Soft Processors. The fact that FPGAs can be reconfigured allow them to be used in acceleration computing or in many other applications where optimization and reconfiguration are needed frequently.There are used also for design prototype.

Despite all of these features FPGAs present some disadvantages like low frequency rates, power consumption and low performance respecting some other circuits like ASICs but this fact cannot avoid them to be used in many applications. Nowadays FPGAs are optimized in a way that they can be programmed at high speed allowing them to be used in high speed applications.

The power consumption is a headache inside FPGAs (and in large ICs in general) and a part of this power rate is caused by the SRAM memories used to control the switches for the flexibility of the FPGAs. It's in this concern that our team adopts the Memory Splitting power reduction technique in order to come over this power rate.

In fact many power reduction techniques have been studied during the researches for the competition such as: Clock Gating, Power Gating also known as Power Shut Off (PSO), Multi-VDD, Multi-Vth and RAM power reduction. Finally the RAM power reduction has been implemented through the SRAM blocks Memory Spleeting that will be discussed in the next section.

[1]RiscEtMorty Team, Université de Strasbourg, Faculté de Physique et Ingenierie

[2]Freddy Anstotz , Enseignant, Université de Strasbourg, Faculté de Physique et Ingenierie

II. Memory Splitting

SRAM-based blocks reprensent a large area inside FPGAs and they are also responsible of power increasing. That's why a soft implementation of the Memory Splitting reduction technique to reduce the dynamic power consumption is the solution adopted in this report. In fact the write operations consume slighlty less power than the read access, so the idea is trying to reduce the number of the read accesses or the power cost of a read access. The idea behind this technique is to reduce the frequency inside the equation of the dynamic power which is:

$$P = \alpha C V^2 f \tag{1}$$

Where α is the activity factor that relates to how many $0 \rightarrow 1$ or $1 \rightarrow 0$ transitions occur in a chip, V the Voltage Supply, f the Frequency and C the physical Capacitance.

The SRAM memory blocks are divided into smaller blocks activated only in needed ones in each access. In fact the blocks can independently transition into different power modes. A minimal data clustering can be used by Compilers and Operating Systems in just few blocks then powering down the unused ones which reduce the power.

Another advantage that make us choose this technique is that it does not require a huge architecture changing. In our case the 64k RAM has been divided in two blocks of 32k RAM. To do so we have used the description model of RAM in the architecture and then created two identical 32k SRAM . After doing that we have added a condition in order to choose in what sub-block the data is going to be read allowing the other block to be shut off during the read operation. Here below is the architecture of the Memory Splitting using two blocks of 32k RAM.

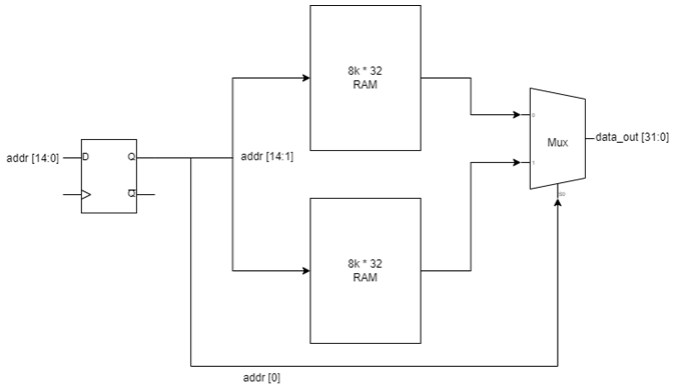


Fig1: Memory Splitting Architecture (2X32k Block RAM).

III. Results before implementation: Reference

Here below we have the results with the reference project before our architecture implementation.

Before implementation	Values
Total Power as reported by Vivado (mw)	307
Number of Cycles	2098749
Actual Clock Period(ns)	22.2
Reference Processing Time (ms)	49.59
FPGA Resources	14675 LUTS,9292 FFs, 36 RAM36
Credence	82

Fig1: Reference

IV. Results after implementation

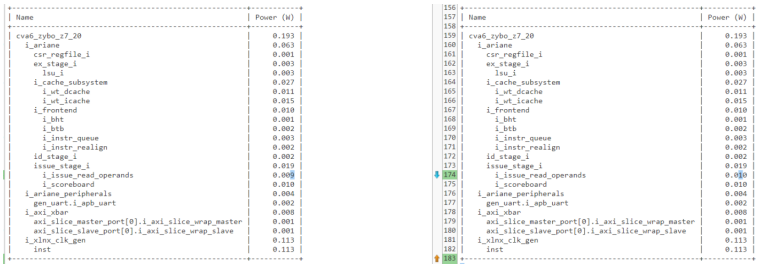
After the implementation of the Memory Splitting, a little power gain has been noticed by the diminution of 0.001 in the operand reads. At energy level we have the following results after calculation. A diminution of 0.07% has been noticed on the number of cycles.

- Reference Energy: **14.30 mJ**
- Energy after implementation: **14.24 mJ**
- Energy Gain: **- 0.41 %**
- Time of Simulation: **46.5585 ms**

Fig2: Energy Gain Report.

After implementation	Values
Total Power as reported by Vivado (mw)	306
Number of Cycles	2097231
Actual Clock Period(ns)	22.2
Reference Processing Time (ms)	46.5585
FPGA Resources	14675 LUTS,9292 FFs, 36 RAM36
Credence	82

Fig3: Table after implementation



Name	Power (W)
cva6_zybo_z7_20	0.193
i_ariane	0.063
csr_regfile_i	0.001
ex_stage_i	0.003
lsu_i	0.003
i_cache_subsystem	0.027
i_wt_dcache	0.011
i_wt_icache	0.015
i_frontend	0.010
i_bht	0.001
i_btb	0.002
i_instr_queue	0.003
i_instr_realign	0.002
id_stage_i	0.002
issue_stage_i	0.019
i_issue_read_operands	0.009
i_scoreboard	0.010
i_ariane_peripherals	0.004
gen_uart.i_apb_uart	0.002
i_axi_xbar	0.008
axi_slice_master_port[0].i_axi_slice_wrap_master	0.001
axi_slice_slave_port[0].i_axi_slice_wrap_slave	0.001
i_xlnx_clk_gen	0.113
inst	0.113

Name	Power (W)
cva6_zybo_z7_20	0.193
i_ariane	0.063
csr_regfile_i	0.001
ex_stage_i	0.003
lsu_i	0.003
i_cache_subsystem	0.027
i_wt_dcache	0.011
i_wt_icache	0.015
i_frontend	0.010
i_bht	0.001
i_btb	0.002
i_instr_queue	0.003
i_instr_realign	0.002
id_stage_i	0.002
issue_stage_i	0.019
i_issue_read_operands	0.010
i_scoreboard	0.010
i_ariane_peripherals	0.004
gen_uart.i_apb_uart	0.002
i_axi_xbar	0.008
axi_slice_master_port[0].i_axi_slice_wrap_master	0.001
axi_slice_slave_port[0].i_axi_slice_wrap_slave	0.001
i_xlnx_clk_gen	0.113
inst	0.113

Fig5: Power diminution in the read operands.

V. Conclusion

During the competition, many techniques of power optimization have been studied and two of them have been implemented. In fact we have tried the Memory Splitting method and the Clock Gating method. The second one didn't give amelioration in the power. The first one has given at least a little diminution of 0.33 % in the energy consumption.

We thank the Thales Group for giving us the opportunity to be part of this challenge. We have learnt a lot regarding power optimization techniques and how to implement them efficiently. We thank also our Professor Freddy Anstotz for being our mentor and make us able to be part of this journey too.

References

[1] 1. Naresh Grover; Dr. M.K.Soni; (2012). "Reduction of Power Consumption in FPGAs - An Overview".

[2] Shashi Kumar V; Gurusiddayya Hiremath; (2016). "Low Power Implementation of RISC-V Processor".

[3] VASANTH VENKATACHALAM; MICHAEL FRANZ; (2005). "Power Reduction Techniques For Microprocessor Systems".

[4] M.Sathish Kumar, S.K.Senthil Kumar, R.Venkatesh, (2016). "Design and Implementation of Clock Gated ALU performing 16 functions using DeMux and AND Gate"

[5] L. Sterpone; L. Carro; D. Matos; S. Wong; F. Fakhar; (2011). "A New Reconfigurable Clock-Gating Technique for Low Power SRAM-based FPGAs".

[6] Ajay Kumar Singh; (2017). "Power Efficient Data-Aware SRAM Cell for SRAM-Based FPGA Architecture".

[7] X.S. Zhang, Washington University "Power Optimization (Part 1 and Part 2)".

[8] Smitha Sundaresan, Frederic Rivoallon,(2012) "Analysis of Power Savings from Intelligent Clock Gating".

[9] M. Gautschi et al.,(2017), "Near-Threshold RISC-V Core With DSP Extensions for Scalable IoT Endpoint Devices".