NTNU | Norwegian University of Science and Technology

ANALYSIS OF POWER BENEFIT FROM DMA

Venkata Anusha Jampala

Spring 2023

Master's Thesis in Electronic System Design
Faculty of Information Technology and Electrical Engineering
Department of Electronic Systems
Norwegian University of Science and Technology

Supervisor : Per Gunnar Kjeldsberg, NTNU

External-Supervisor: Lars Sundell, Nordic Semiconductor

Assignment text

Candidate name : Venkata Anusha Jampala
Assignment title : Analysis of power benefit from DMA
Assignment text :
Nordic chips include some rudimentary DMA controllers. The usual motivation for DMA is performance, but DMA also presents an interesting potential for power optimization.
Assuming that :

- The alternatives for memory transactions are either uC based memory access or DMA based memory access
- The uC (being bigger/more gates) requires more power to perform a bus transaction than a DMA controller performing the same bus transaction
- The uC can be in low power mode (sleep) while the DMA occurs
- There could be significant advantages to having more/better/smarter DMA. This project is to investigate whether we do already achieve a power benefit and to get some numbers for this benefit. This will give us a starting point to measure any improvements against.
- The student would start by identifying some measures for calculating power/performance benefit. There is existing literature on the topic.
- The student would investigate our existing DMA controllers - how power and performance vary (if at all) with the various parameters that configure our DMA controller.
- The student would research alternate DMA architectures and present a proposal of which ones might be suitable candidates for experiments.
- The student would implement the architectures after agreeing with the supervisors.
- The student will then need to carry out power estimation with the candidate DMA architectures.

Assignment proposer: Lars Sundell, Nordic Semiconductor ASA
Supervisor: Per Gunnar Kjeldsberg, Department of Electronic Systems, NTNU

Abstract

Data transfer and memory usage can have a significant impact on power consumption in computing systems. Inefficient data transfer mechanisms or excessive data transfer operations can result in higher power consumption. Data transfer and memory are also interconnected, with efficient data transfer often relying on sufficient memory to temporarily store data during transfer. Optimizing both data transfer and memory performance is essential for maximizing the overall performance and usability of a computer system. Direct Memory Access(DMA) controllers play a crucial role in chip performance. The DMA controller is used to transfer the data blocks between memory locations and external devices without interrupting the execution flow of the CPU. DMA is used to increase the overall system performance by reducing the load on the CPU. The DMA controller performs bus transactions with low power consumption compared to bus transactions performed by the microcontroller.

This thesis presents power optimization techniques for DMA controllers and information regarding the power consumption and energy consumption of DMA controllers with different buffer widths. The simulation waveforms for the DMA controller with different buffer widths are provided. The existing DMA controller is modified to incorporate the identified power optimization techniques. The power consumption and energy consumption values for the existing and modified DMA controllers are measured. The power consumption of a DMA controller increases when the buffer width increases due to the amount of data transferred in each cycle. The power consumption values table is presented with power consumption values of high and low activities at different time intervals. Compared to the existing DMA controller, results show a reduction of energy consumption by 37% and 60% for 16-bit and 32-bit DMA controllers. When the buffer width of a DMA controller is doubled the energy consumption reduces due to less number of bus transfers for data transmission.

Preface

This master's thesis is the final assignment for me as a student at Electronics Systems Design and Innovation at NTNU. Doing a thesis of this scope is not something I have done before, and it has been challenging. The prior experiences from project assignments I did in the previous semester have been useful for structuring the work and understanding the task.

I want to thank all persons involved in helping me and guiding me throughout this thesis work; supervisor Per Gunnar Kjeldsberg at the Department of Electronic Systems, NTNU, and supervisors Lars Sundell from Nordic Semiconductor. Also, I thank Nordic Semiconductor engineers Asbjørn Steensland and Artur Antunes for taking the time to help me during the project. The company named Nordic Semiconductor ASA in Trondheim proposed the project title "Analysis of power benefit from DMA". Tools from Synopsys are utilized, and Nordic Semiconductor has provided workplace and computer equipment with access to all the other design tools needed, which I am grateful for.

Trondheim April 3,2023
Venkata Anusha Jampala

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The growth of digital electronics and integrated circuits has led to the development of smarter, more complex, and power-efficient devices [1]. One of the crucial parameters in hardware development is reducing power consumption. Reducing power consumption has advantages in terms of improving the time to charge the battery, which is especially important for the Internet of Things (IoT), where battery life is critical[3].

Initially, DMA controllers were used to transferring data between memory and peripherals, and it was observed that DMA presents potential benefits in power consumption in low-energy Bluetooth devices. The inclusion of a Direct Memory Access (DMA) controller in IoT devices has brought about substantial benefits, including enhanced power usage, performance, and CPU bandwidth [4]. By enabling more efficient data transfer between devices, DMA reduces the CPU's workload, allowing it to concentrate on other tasks. As a result, data processing is faster and more effective, and energy consumption is reduced, which is crucial for IoT devices powered by batteries. All in all, the integration of DMA in IoT devices has enhanced their functionality and increased their energy efficiency.

DMA requires less power to transmit data compared to the microcontroller because it reduces the CPU workload, and enables faster data transfer. Low energy consumption in Bluetooth Low Energy (BLE) applications is an essential requirement for applications that use a significant number of BLE devices, such as smart home sensor systems [5]. The analysis of power benefits from using DMA shows that it can significantly improve power efficiency and extend the battery life of devices while improving their performance.

1.2 Objectives, limitations, and research methodologies

In this thesis, the focus will be on power optimization techniques for the existing DMA Controller. Power optimization in DMA controllers can provide significant benefits in terms of overall system performance. The optimizations for the DMA controller can be performed in memory architecture, bus architecture, and software, based on end-user requirements. This thesis will investigate the power consumption values of an existing DMA controller, as well as the power consumption values of an optimized DMA controller.

The objectives of this thesis are:

- To understand and measure power consumption values for the existing DMA Controller.
- To investigate how can the existing DMA be optimized for power consumption.
- To adopt the best power optimization technique for the existing DMA controller from the identified optimization techniques in the specialization project.
- To analyze and implement the optimization technique identified in the existing DMA controller.
- To measure power consumption values for the optimized DMA Controller.

This work will explore power estimations in Spyglass, which is an early analysis design tool. The aim is to provide a comparison table for the power consumption values with respect to the existing DMA controller, as well as the optimized DMA controller. The simulations are performed using Questa sim.

The research methodology of this thesis is:

- To investigate Nordic DMA based on the results from the specialization project, and perform experiments on the existing DMA controller.
- To perform optimizations based on the results from the specialization project.
- To perform power estimation at various design phases, such as Register Transfer Level(RTL), and estimate power consumption after Netlist Simulation. Power estimation at the RTL level provides good design tradeoffs in the initial design phase. The power optimizations in this thesis are performed at the RTL level.
- The analysis of power consumption results is performed to check if the existing DMA controller has the lowest power consumption values possible, or if the optimized DMA controller can provide a significant benefit in terms of power consumption and energy consumption compared to the existing DMA controller.

The identified power optimization techniques for the DMA Controller are based on application requirements, which is a limitation when choosing the optimization technique for the existing DMA controller in low-energy Bluetooth devices.

This thesis specifically focuses on Nordic DMA. The focus of optimization techniques is limited to the techniques identified in the specialization project. Another limitation is that the simulation is carried out at a high level. This thesis report provides information regarding identified power optimization techniques, as well as the analysis behind the implemented optimization technique.

1.3 Main contributions

The main contributions of this thesis are:

- Analysis of the identified modification techniques from the specialization project is performed to choose the best modification technique for the existing DMA controller.
- The power consumption values of the existing DMA controller are measured, with different scenarios.
- The identified modification technique is implemented in the existing DMA controller. The power consumption values of the optimized DMA controller are measured.
- The power consumption values are measured at different simulation phases by defining activity files for both the existing and optimized DMA controllers.
- Analysis of the power estimation values is performed for both the existing and optimized DMA controllers.
- The energy consumption values for existing and optimized DMA controllers are measured.
- A comparison table of power consumption values is provided for different scenarios at different time intervals.
- The results show a reduction of energy consumption by 37% and 60% for 16-bit and 32-bit DMA controllers compared to the existing DMA controller

1.4 Report structure

The rest of the report is structured as follows:

Chapter 2 - Theory
This chapter presents relevant background theory on DMA Controller as well as RTL power estimation..

Chapter 3 - Previous work and tools
This chapter contains information regarding previous work and the tools used for simulation and estimating the power.

Chapter 4 - Methodology
This chapter describes the existing methods and applied modifications.

Chapter 5 - Results and Discussions
This chapter provides the obtained results and a discussion of the results.

Chapter 6 - Conclusion and Future work
This chapter summarizes the results of the project and describes the future work that can be performed.

Abbreviations

DMAC - Direct Memory Access Controller

IoT - Internet of Things

CPU - Central Processing Unit

PAR - Peripheral Access Register

PPI - Programmable Peripheral Interconnect

SVI - System Verilog Interface

DCP - DMA Channel Peripheral

PCGC - Power & Clock Gating Control

AMBA - Advanced Micro-controller Bus Architecture

SoC - System on Chip

RTL - Register-Transfer Level

FSDB - Fast Signal Data Base

VCD - Value Changed Dump

SAIF - Switching Activity Interchange Format

SDC - Synopsys Design Constraint

UPF - Unified Power Format

M2P - Memory to Peripheral

RFFs - Retention Flip-flops

GUI - Graphical User Interface

Chapter 2

Theory

This chapter explores the required theory regarding the DMA controller and power estimations.

2.1 Direct memory access controller

A DMA controller is a hardware device that allows data to be transferred between devices in a computer system without involving the CPU (Central Processing Unit)[1]. In large computing systems, the DMA controller is responsible for managing data transfers between devices such as the hard disk drive, network interface card, or graphics card and the computer's memory, without the intervention of the CPU. When devices need to transfer or receive data from or to memory, the device sends a DMA Request signal to access the DMA Controller[1]. After receiving the DMA Ack signal from the DMA controller, devices transfer or receive the data from or to memory. This is the basic working principle of a DMA Controller, as shown in Figure 2.1.

In the present world, we can experience more advanced and complex DMA controllers based on the type of application. The DMA controller is introduced in System on Chip(SoC) for transferring data between memory and input/output peripherals without disturbing the execution flow of the central processing unit[6]. Data transfer in the DMA controller can be performed from memory to memory, memory to peripheral, peripheral to memory, and peripheral to peripheral. The introduction of the DMA Controller in SoC achieved significant benefits in terms of high-speed data transfer and the volume of data[7].

A typical example of the operation of a DMA Controller during Data transfer is as follows: the DMA controller sends the Bus Request signal to the CPU, and then the CPU completes the operation it is performing and sends the Bus Grant signal to the DMA Controller [1]. The DMA controller then takes over the system buses and performs the data transfer operation. When the data transfer operation is completed, the DMA Controller Interrupts the signal to the CPU, no-

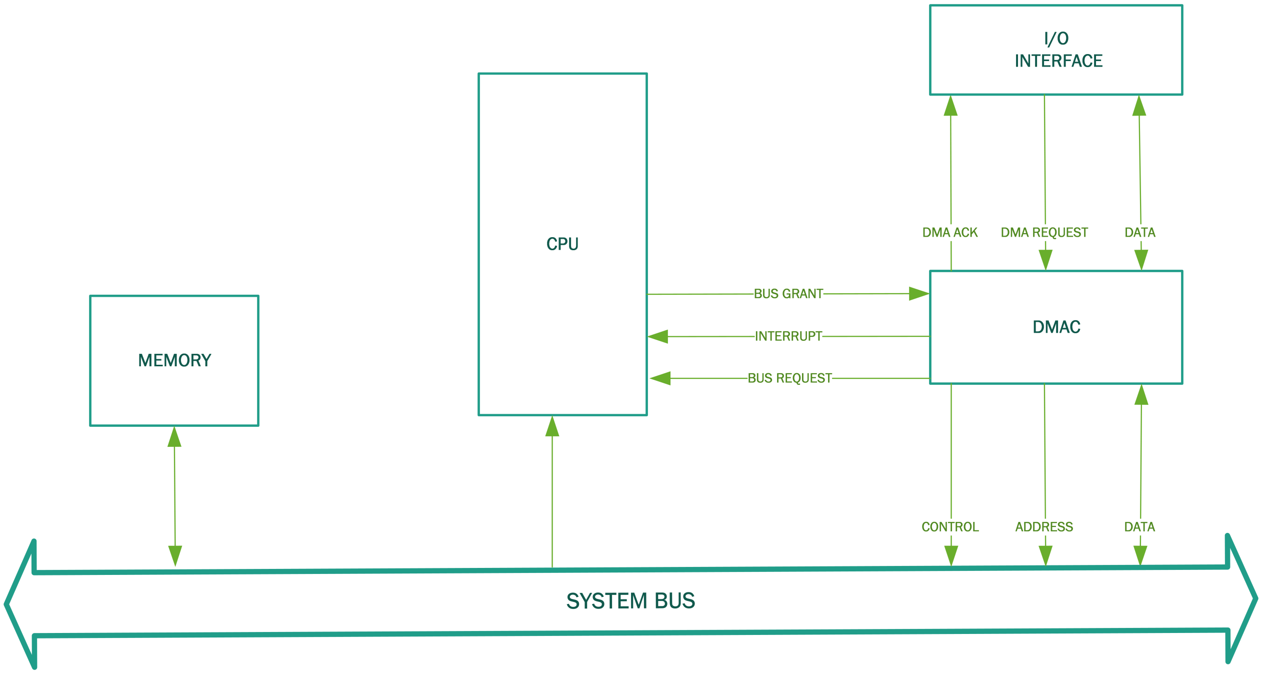


Figure 2.1: The Architecture of DMA Controller in Embedded Application (adopted from [1])

tifying it that the data transfer operation is completed, and then the CPU takes over the system bus. This is one way of doing data transfer operations with the DMA controller.

The architecture of the DMA Controller varies based on the application requirements. The DMA Controller for Embedded Applications often operates on the Advanced Microcontroller Bus Architecture specifications(AMBA). An AMBA-based DMA controller has significant benefits in terms of high speed and transferring a high volume of data[1]. There is a significant improvement in the performance of the DMA controller when it works with the bus architecture. Apart from the various buses for DMA operation, Advanced Microcontroller Bus Architecture (AMBA) is a specific bus architecture for embedded products in SoC. AMBA is used for connecting various functional blocks in SoC and also provides support to various controllers, processors, and peripherals [8]. Two buses are defined with AMBA specifications: Advanced High-Performance Bus (AHB) and Advanced Peripheral Bus (APB). AMBA is widely used in Application-Specific Integrated Circuits (ASIC) and SoC-based portable devices [8].

The DMA controller can be decentralized, which is the DMA controller is not located on a centralized bus or system but instead is distributed across the peripherals[9]. The decentralized DMA controller reduces network congestion by allowing data to be transferred directly between the peripheral and the bus without passing through a centralized system. Decentralized DMA can reduce the need for data to be processed by the CPU, which reduces power consumption. More details of decentralized DMA controller are presented in Chapter 3.4.

The DMA controller can be centralized, meaning a single DMA controller manages data transfers between multiple devices and the system memory. Systems that require memory access from multiple devices, like multi-channel audio and video systems, frequently use centralized DMA[10]. The DMA controller in this architecture arbitrates the memory access requests coming from various devices and schedules the data transfers to make sure data transfers are executed efficiently. The benefit of centralized DMA is that it provides a single point of control for data transfers, which is useful to simplify the overall system design and reduce the conflicts associated with managing multiple DMA channels. Centralized DMA can improve the efficiency of data transfers by reducing the use of system resources and minimizing conflicts between different devices.

2.2 Power Consumption

The power consumption in digital circuits can be categorized into static power and dynamic power. Static power is also known as leakage power, and dynamic power can also be called switching power. The low-power design aims to reduce overall dynamic and static power consumption in digital circuits[11]. Low-power designs are used to reduce the power of individual components as much as possible to minimize overall power consumption. The power dissipation in any digital complementary metal oxide semiconductor (CMOS) is shown in Eq (2.1)

$$P\_avg = P\_static + P\_dynamic \tag{2.1}$$

where P_avg = Average Power Dissipation, P_static = Static Power Dissipation, and P_dynamic = Dynamic Power Dissipation

2.2.1 Static Power

Static power is defined as power dissipated by the logic gates even when they are in idle state[11]. It is typically caused by the flow of current through a circuit or device when it is in a non-operational state. Static power consumption can be a significant factor in a device's overall power consumption, particularly in devices designed to operate in low-power modes for extended periods. Minimizing static power consumption is important in the design of many electronic devices, particularly those intended for battery-powered or portable applications. From a design perspective, static power is nothing but a leakage power dissipated by the logic gates[12].

Static power can also refer to the power dissipated within a circuit due to its internal resistances and leakage currents, even when the circuit is not actively switching. It includes the power consumed by static logic elements, such as latches and flip-flops, and the leakage current through transistors that are supposed to be turned off. Several techniques can be used to reduce static power consumption in electronic devices, such as using low-leakage transistors, optimizing circuit layout and design, and implementing power gating or power shutdown features to turn off unused circuitry when it is unneeded[12].

2.2.2 Dynamic Power

Dynamic power refers to the amount of power used by a circuit as a result of the charging and discharging of capacitances when digital signals transition between high and low states[10]. It is also known as switching power or transient power. The amount of dynamic power consumed is influenced by several factors, including the supply voltage, clock frequency, activity factor (which is the proportion of gates that switch within a given period), and the capacitance of the load. Dynamic power is a combination of both switching power and internal power[11]. The equation for dynamic power is shown in Eq (2.2)

$$P\_dynamic = 1/2 * C\_load * v^2 * f\_clock * activityfactor \tag{2.2}$$

where:

- P_dynamic is dynamic power consumption,
- C_load is the total capacitance of the circuit,
- v is the supply voltage,
- f_clock is the clock frequency, and
- activity factor is the charging of capacitors from 0 to 1.

The above equation shows that the dynamic power consumption is proportional to the capacitance being switched, the square of the supply voltage, and the clock frequency. Increasing the clock frequency, the number of gates, or the voltage will increase the dynamic power consumption. Dynamic power consumption is a crucial factor in the overall power consumption of a device or system, specifically in high-performance computing systems. Clock gating, power gating, and voltage scaling techniques can be used to reduce dynamic power consumption and improve the energy efficiency of digital circuits[13].

2.3 Energy Consumption

The energy consumption of a circuit is directly related to its power consumption, which is the rate at which the circuit uses energy. Lowering power consumption through techniques such as low-power design can therefore help to reduce energy consumption, which can have important implications for energy efficiency and sustainability in areas such as electronics, computing, and telecommunications. Reduction frequency does not alone reduce energy consumption, since the execution time will be longer, but combined with a possible reduction in voltage the energy consumption can be reduced. The equation for energy consumption is shown in Eq (2.3)

$$Energyconsumption = Power * Time \tag{2.3}$$

where Power is the rate at which energy is consumed (measured in watts) and Time is the duration for which the power is consumed (measured in seconds).

The difference between power and energy is very important in battery-operated devices[14] because their performance is directly impacted by the power and energy consumption of the circuits. To extend the battery life of battery-operated devices, reducing energy consumption is crucial. This can be accomplished by optimizing the circuit and component design to reduce power consumption[15]. On the other hand, energy consumption is the total amount of energy used by a device during a certain time period or to perform a certain task. To get the maximum battery runtime, energy efficiency is crucial. This can be accomplished by designing the circuits to operate at the lowest voltage levels possible and selecting components that are optimized for low energy consumption.

2.4 Power and clock gating control

Power gating and clock gating are well-known techniques in digital circuits in order to reduce dynamic and leakage power.

2.4.1 Clock Gating

Clock gating is used to reduce dynamic power dissipation by temporarily turning off the clock when there isn't any valid data to be computed, transmitted, or stored. In digital circuits clocking system is responsible for significant chip power by including switching activities of flip flops, latches as well as clock distribution networks[2]. The clock gating can be implemented at the gate level and RTL level. At the gate level, clock gating involves inserting additional logic gates in the clock path to control the clock signal. At the RTL level, clock gating is typically implemented using a conditional assignment statement in the Verilog or VHDL code[16]. Implementation of clock gating at a gate level is more challenging compared to implementing it at the RTL level.

The clock gating at the gate level involves modifying the gate level netlist and the design is repre-

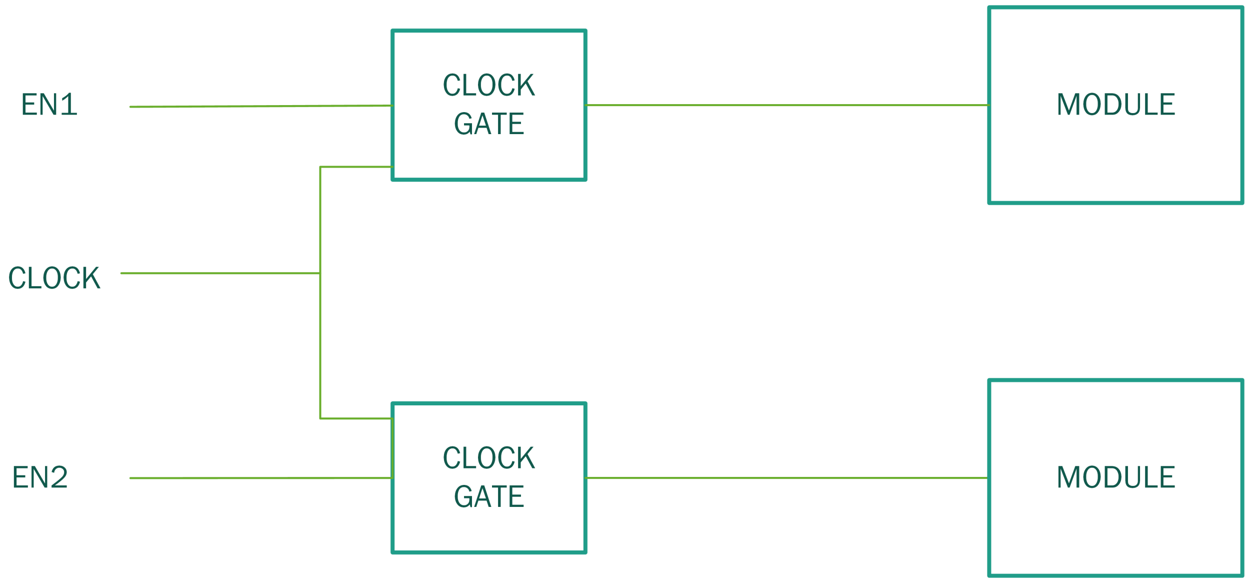


Figure 2.2: Module Level Clock Gating Block Diagram [2]

sented as a network of logic gates and flip-flops, making it more complex to identify and isolate clock signals. Clock gating at the RTL level can be easier as it involves adding or modifying RTL code, which is a higher-level abstraction of the design.

The clock gating technique can be implemented in different ways, for example, module level clock gating, enhanced clock gating, multi-stage clock gating, and hierarchical clock gating. The block diagram for module-level clock gating is shown in Figure 2.2. The module-level clock gating is carried out by gating the clock with synchronous control signals[2].

Modern design tools are supporting automatic clock gating, the clock gating can be adopted without modifying the function of the logic[14]. In [17] a chip design is implemented with and without clock gating and the power consumption values are measured. The results showed an area reduction of 20% and power reduction of 34% with clock gating implementation[17]. The area reduction is possible due to the insertion of single clock gating in place of multiple multiplexers[14].

2.4.2 Power Gating

The power gating technique has a significant role in order to reduce leakage power in the chip during a particular mode of operation by turning off power in the nonoperational power domain[18]. Power gating can also be implemented using the stacking transistors approach. Power gating is useful to maintain functionality while providing the opportunity to save as much power as possible by turning off as many domains as possible as often as possible. It is also possible to drive a power gating circuit with the output signal from the clock gating circuit[18]

The power gating can be implemented by turning off power for part of the design, turning off power in parts of a design can reduce power consumption and increase the overall energy efficiency of a system. when the part of the design is powered down, flip-flops in that part will lose their state. This can result in unpredictable behavior when power is turned on in that part, which can cause errors. To avoid this risk, flip-flops with retention can be used to save their state during a power down[19]. These retention flip-flops (RFFs) can hold their state even when power is turned off. Typical RFFs have an additional power supply pin to operate in retention when the main power is turned off[20]. When power is turned on to the design, the RFFs can quickly return to their previous state without requiring a complete reset of the system[21]. Power gating is often used in combination with clock gating to further reduce power consumption in digital circuits. By selectively gating the clock and power to specific circuit blocks, power gating, and clock gating can work together to optimize power consumption in a circuit.

2.4.3 Power consumption in memories and buses

In digital circuits, power consumption in registers, buses, and memories must be taken into account because these parts can use a lot of power in a system[22]. Memory types, such as Static Random Access Memories (SRAM) and Dynamic Random Access Memories (DRAM), can consume a significant amount of power when they are accessed. This is due to the fact that energy is needed to charge or discharge memory cells. The address and data buses also consume power because the data being transferred requires energy to drive the bus. Registers, such as flip-flops and latches, consume power when they are being clocked because the clock signal needs to charge and discharge the internal capacitance of the register, which requires energy. Minimizing power consumption in memory, buses, and registers is critical for improving the power efficiency of digital circuits.

Chapter 3

Previous work and tools

This chapter presents results from the literature study carried out during the specialization project[23]. This Chapter also presents the tools used for simulation, and power measurement, and gives information regarding existing DMA controller.

3.1 TFE4580 - Specialization Project

A specialization project has previously been carried out to perform a Literature review to find Power optimization techniques relevant for DMA Controller[23]. In the literature review, a total of four papers were selected for the depth review based on the level of implementation details, architecture, and interesting results in terms of power and energy consumption. An analysis was performed to answer questions regarding memory architecture, bus architecture, and software implementation. Selected identified modification techniques in the specialization have been implemented as part of the current work described in this master thesis. The power consumption values can be measured between an existing Nordic DMA controller used by the company Nordic Semiconductor, see Section 3.4, and the modified DMA controller. A comparison table was prepared in the specialization project by addressing the identified modification techniques. The information provided in the comparison table refers to the following papers: [24],[8],[4] and[25]. The comparison table is shown in Table 3.1

C

Title	Architectural Implementation	Power Consumption & Area	Energy Consumption & Speed	Performance	Advantages & Disadvantages
DMA-circular: An Enhanced High-Level Programmable DMA Controller for Optimized Management of On-chip Local Memories	The provided implementation is to embed the cache functionality into the DMA controller, and additional hardware to trigger the control actions related to local memories.	The power consumption is the same as the traditional DMA controller.	The results showed that energy consumption is reduced from 5% to 40%.	The proposed technique achieved good performance by using small buffers. The overall performance of the kernel is increased between 1.2x to 2x when the DMA circular is used.	The proposed technique is evaluated only in a single core. The results showed a reduction in control code overheads to 15% of the execution time.
AMBA Based Advanced DMA Controller for SoC	This paper proposed a DMA controller which works on AMBA specifications, the DMA works as a bridge between AHB and APB.	It is mentioned that the proposed design achieved low power consumption.	The proposed design achieved a high speed of data transfer at high frequency with the use of fewer lookup tables (Transistors).	It is mentioned the performance is improved in terms of data transfer.	The proposed design solved the timing and volume of data problems. The proposed DMA is a good alternative for multimedia processing.
A Low-Area Direct Memory Access Controller Architecture for a RISC-V-Based Low-Power Microcontroller	The proposed DMA Controller architecture is implemented with a configurable number of channels and FIFO depth.	The configuration with 8 channels got the best area reduction, up to 75% in comparison to the reference image. The proposed solution is claimed to use only 4.2% of the total chip area.	Based on a comparison made in the paper, the proposed solution seems to achieve scalable bandwidth from 1.5Gbit/s @48 MHz to 12.2Gbit/s @96 MHz	The proposed DMA Controller with three-stage pipeline execution can showcase correct behavior for USB 1.0 / 2.0 and QSPI with DMA access.	The proposed DMA controller's execution is simple and effective with a three-stage pipeline. This can reduce area and improve throughput (bandwidth).
Design and implementation of Efficient Direct Memory Access (DMA) Controller in Multiprocessor SoC	The modifications have been applied to the way of coding for individual blocks in DMA to reduce the power consumption as well as area.	The provided results showed that the power consumption was reduced by 22.61% and 23.99% as well as area reduction to 19.43% and 20.10% by source and destination address generators, respectively.	There is no information provided regarding the speed of execution in this paper.	There is no significant improvement in performance with the designed DMA controller in multiprocessor SoC.	The modifications in the source & destination address generator provided good results in terms of power and area. The presented model is very useful in high computational work like Digital Signal Processing.

Table 3.1: Comparision of optimization techniques identified in specialization project

The review performed in the specialization project showed that the introduction of a DMA controller resulted in significant benefits in various applications. The usage of CPU-driven methods in sensor node control resulted in significant power consumption [26]. The introduction of a DMA controller in the sensor resulted in 37% less power consumption compared to the CPU-driven method [27]. The method is carried out by adjusting the CPU to sleep while the DMA controller is executing, which provides significant results in terms of power consumption. The adoption of DMA controllers in Multiprocessor Systems on Chip presented significant benefits in terms of power and area. In a multiprocessor SoC, the processors communicate with each other and with peripherals via a shared memory, and the DMA controller is responsible for managing data transfers between the memory and the peripherals[10]. A well-designed DMA controller can significantly improve the overall system performance by offloading data transfer tasks from the processors, which allows them to focus on computation and reduces the time required to complete data transfer operations. This can result in faster system response times and improved throughput. The presented results show that the adopted DMA controller achieved 22% to 23% power and 19% to 20% area benefits [25]. The benefit is achieved by reducing the width of the source address generator and destination address generator

One of the possible modifications identified for the DMA controller from the literature review is based on the memory architecture of the system [24]. The technique is implemented by introducing the cache functionality in the DMA controller for on-chip local memories. The presented technique resulted in significant benefits in terms of speed of execution and reduced energy consumption. The proposed architecture required external hardware to add cache functionality, which affects the area. The proposed technique is implemented only in single-core processors. This modification will be implemented in the existing DMA controller by modifying the buffer size.

The second optimization technique identified in the literature review is based on the bus architecture of the system[6]. The proposed technique is based on the AMBA specifications and the proposed DMA controller works as a bridge between AHB and APB busses, which allows them to work concurrently. This technique uses a buffer mechanism for different speeds of peripherals, and it is implemented for one master and multiple slaves. A multiplexer is used to ensure that only one slave is accessing the data bus at a time, and a decoder is used for the selection of the slave to perform the transfer operation[6]. The presented technique is used to transfer a large volume of data with low time characteristics. The presented technique is only tested in one master and multiple slave method.

3.2 Questa sim

Questa sim software developed by Mentor graphics supports many hardware description languages such as Verilog, system Verilog, VHDL, and system C. Questa sim is used to simulate RTL designs and netlists, execution of test benches, and supports design verification[28].

3.3 Estimating power in Spyglass

The Spyglass tool is introduced for early design analysis with in-depth analysis at the RTL design phase. RTL power estimation is faster and does not demand a netlist, but decreases the accuracy of the estimations[29]. Spyglass supports various power exploration techniques like clock gating, micro-architectural modifications, and memory access optimizations[30]. The overview of power estimation in spyglass is shown in Figure 3.1

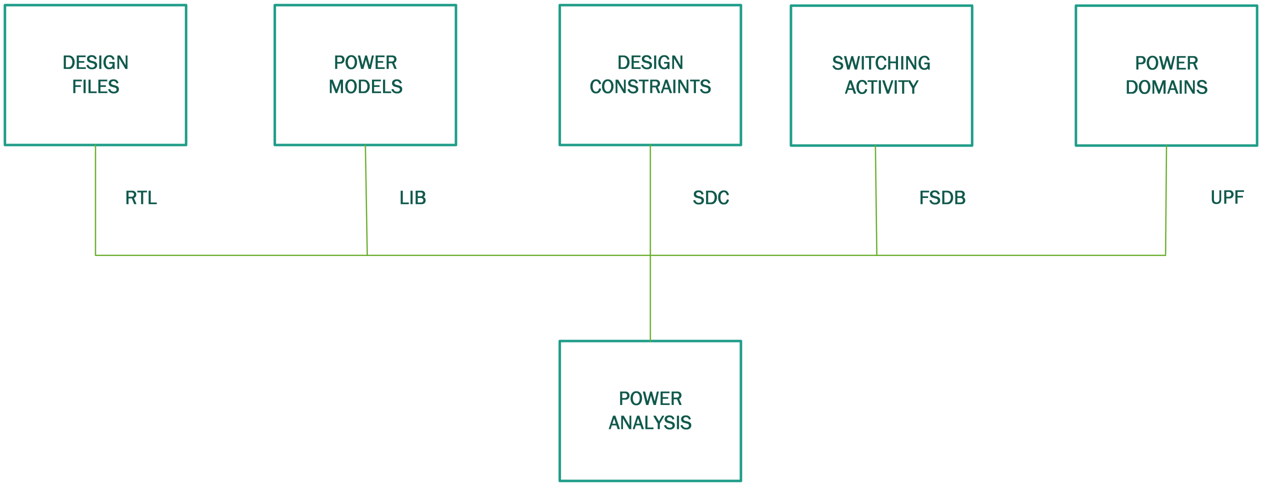


Figure 3.1: Power Estimation flow in Spyglass

The design files are RTL code that describes the design. The tool analyzes the RTL code and translates it to the gate level for power analysis[31]. The power model is used to estimate leakage and internal power dissipated by each type of cell and this is provided by the power models in lib files. For the switching activity, several file formats are available, FSDB(Fast Signal Database), VCD(Value Changed Dump), and SAIF(Switching Activity Interchange Format). The switching activity of a design indicates how often different nets change the signal value, this information is used to measure the dynamic power consumption of the circuit.

The SAIF file logs the average activity of each signal in a simulation, the FSDB file is an event-based format and the VCD file is similar to FSDB, which logs each toggle in every signal.

The activity files are dumped from the RTL simulation in Questasim. Based on the RTL code and given scenario FSDB dumper outputs the activity file from the RTL code. The SDC(Synopsys

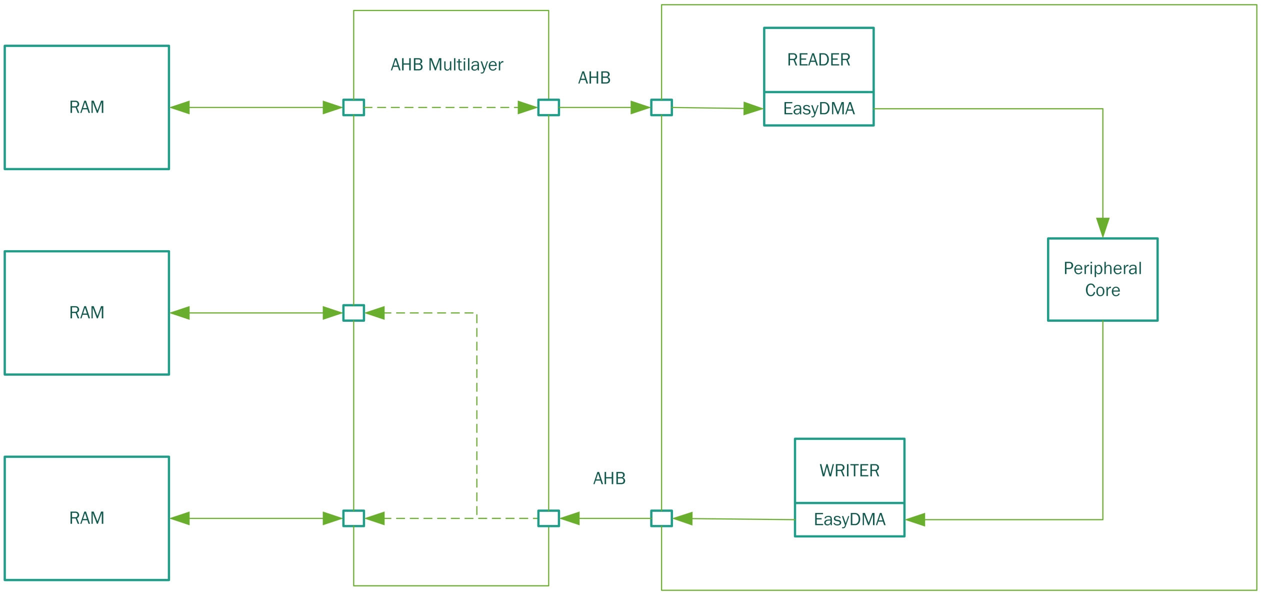


Figure 3.2: Block diagram for multiple instances of EasyDMA in Peripheral

Design Constraint) sets the parameters that affect the power. The UPF(Unified Power Format) file describes the power intent in the design. The file format consists of syntax for describing power supplies, power switches, level shifters, and power states.

3.4 Existing DMA controller

EasyDMA is used in Nordic Semiconductor, which is a user-friendly direct memory access module implemented in peripherals to get direct access to Data RAM[32]. The EasyDMA is the AHB master and it is connected to the AHB multi-layer interconnect for direct access to Data RAM.EasyDMA can only access the RAM. The peripheral can use multiple EasyDMA instances, for instance, to provide a dedicated channel for reading data from RAM into the peripheral at the same time as a second channel for writing data to the RAM from the Peripheral. The block diagram for multiple EasyDMA instances in the peripheral is shown in Figure 3.2

In Nordic Semiconductor DCP(DMA channel peripheral)is used, which is a direct memory access channel for peripherals that uses DMA towards memory. DCP communicates with the peripheral on the DMA bus and memory on an AHB bus. DCP channel peripheral contains multiple sub-modules assigned to perform various tasks like data transfer between peripheral and memory. Several signals are defined to indicate the beginning of data transfer and end of data transfer as well as to set up the buffering and, address and data on the bus. The data can be transferred from the memory to the peripheral and peripheral to memory. The existing DMA supports PCGC(Power and Clock Gating Control). The functional block diagram of the DMA

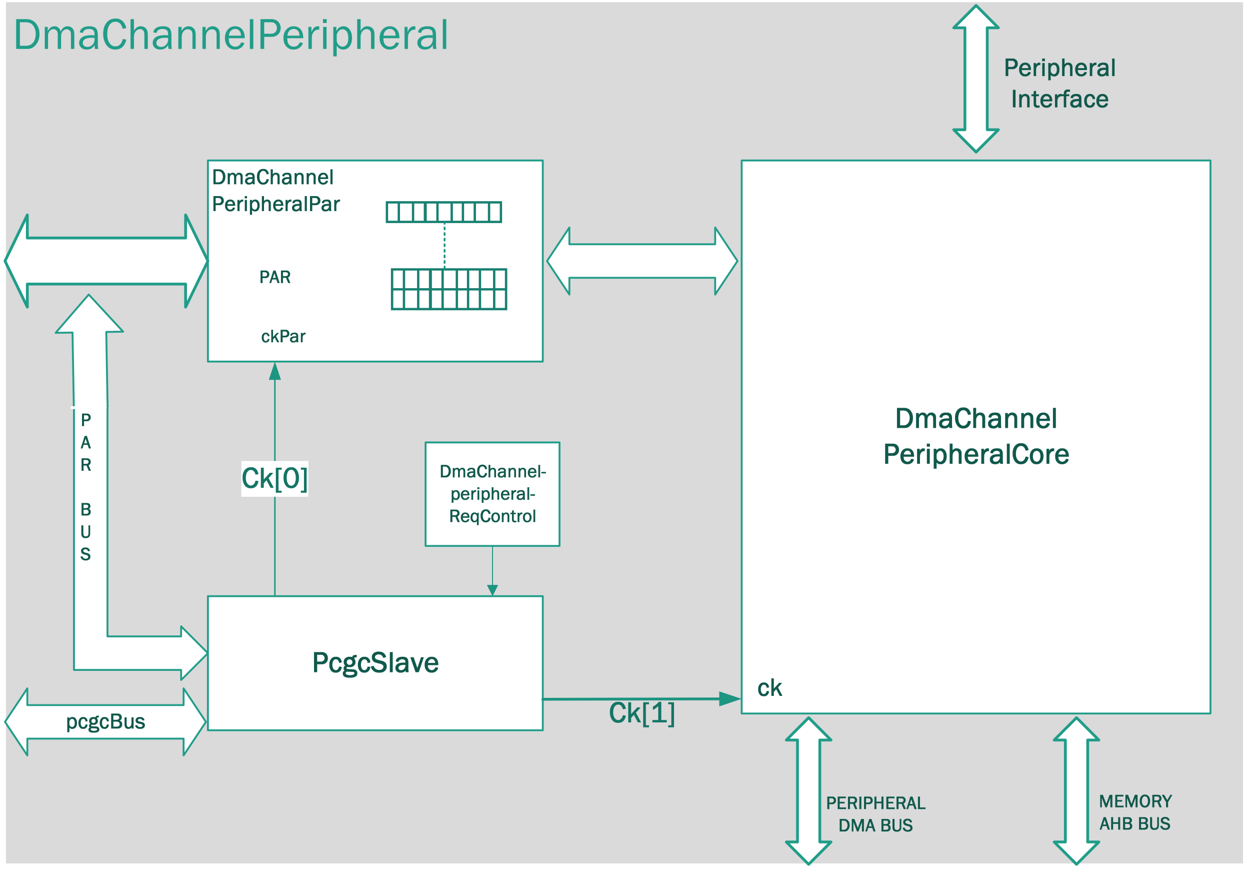


Figure 3.3: Functional Block Diagram Of DCP

channel peripheral is shown in Figure 3.3. The block diagram for integration of peripheral and memory through the DMA channel which is indicating input and output signals from the DMA channel is shown in Figure 3.4. The task start, task enable, and transfer direction signals are input signals, and eventDmaend and eventReady signals are output signals for DCP. The DCP communicates with the memory through the memory bus arbiter.

The DCP will be powered on before any data can be transmitted. If the memory is the source then the initial bytes will be prefetched as soon as the DMA started. Then the DMA channel waits for the peripheral, while DMA waiting for the peripheral prefetching can be stalled. As soon as the peripheral is ready, the transfer starts and there are stalls between DMA and memory and DMA and peripheral depending on the speed. The DMA transfer ends with an active end signal from the peripheral and the DMA can also be interrupted directly by the CPU through a task trigger. The data is transferred byte by byte due to the size of the DMA bus. The EasyDMA is implemented in several peripherals to get direct access to the RAM without the involvement of the CPU and to make data transfer as efficient as possible. The arbitration technique can be used when multiple Easy DMA's are present[33]. One way of arbitration is for the DMA controller

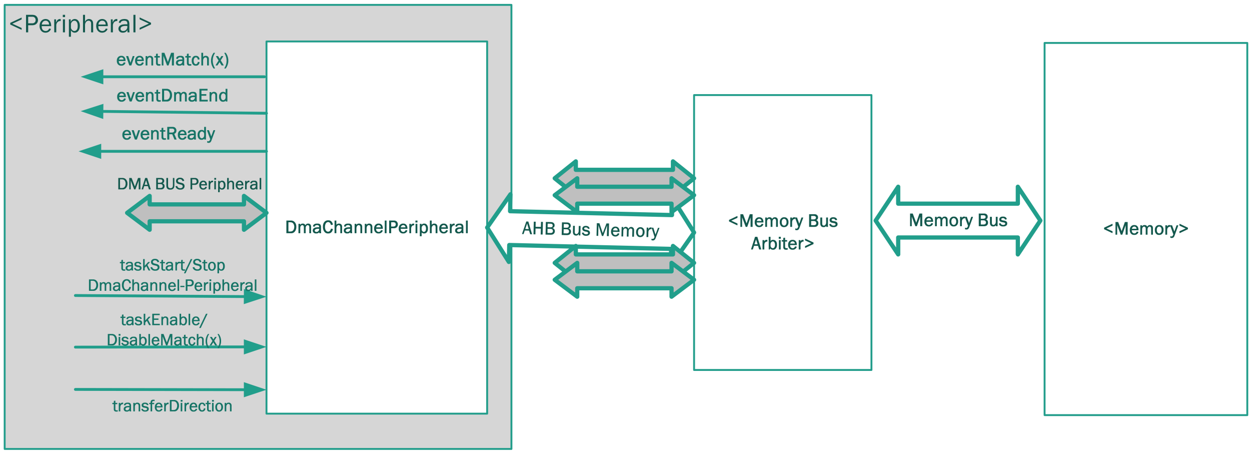


Figure 3.4: Integration of peripheral and memory through DMA channel

with the highest priority level to be given access to the memory first in the fixed priority arbitration method, which assigns a fixed priority level to each DMA controller. If there are several controllers with the same priority level, they could be arbitrated based on a set of rules.

The EasyDMA can also be implemented in Radio. The combination of Easy DMA with automated packet assembler, packet disassembler, automated Cyclic Redundancy Checker(CRC) generator, and CRC checker[34] makes it simple to configure and use the RADIO. The Radio block diagram is shown in Figure 3.5. The Radio uses Easy DMA for reading data packets from and writing to memory. Where IFS(Interframe Spacing Control unit) is used to simplify the address while listing and interframe spacing respectively in low energy and similar applications. The Radio contains Received Signal Strength Indicator(RSSI) and a bit counter. A bit counter usually generates an event, when a predefined number of bits is sent or received by radio.

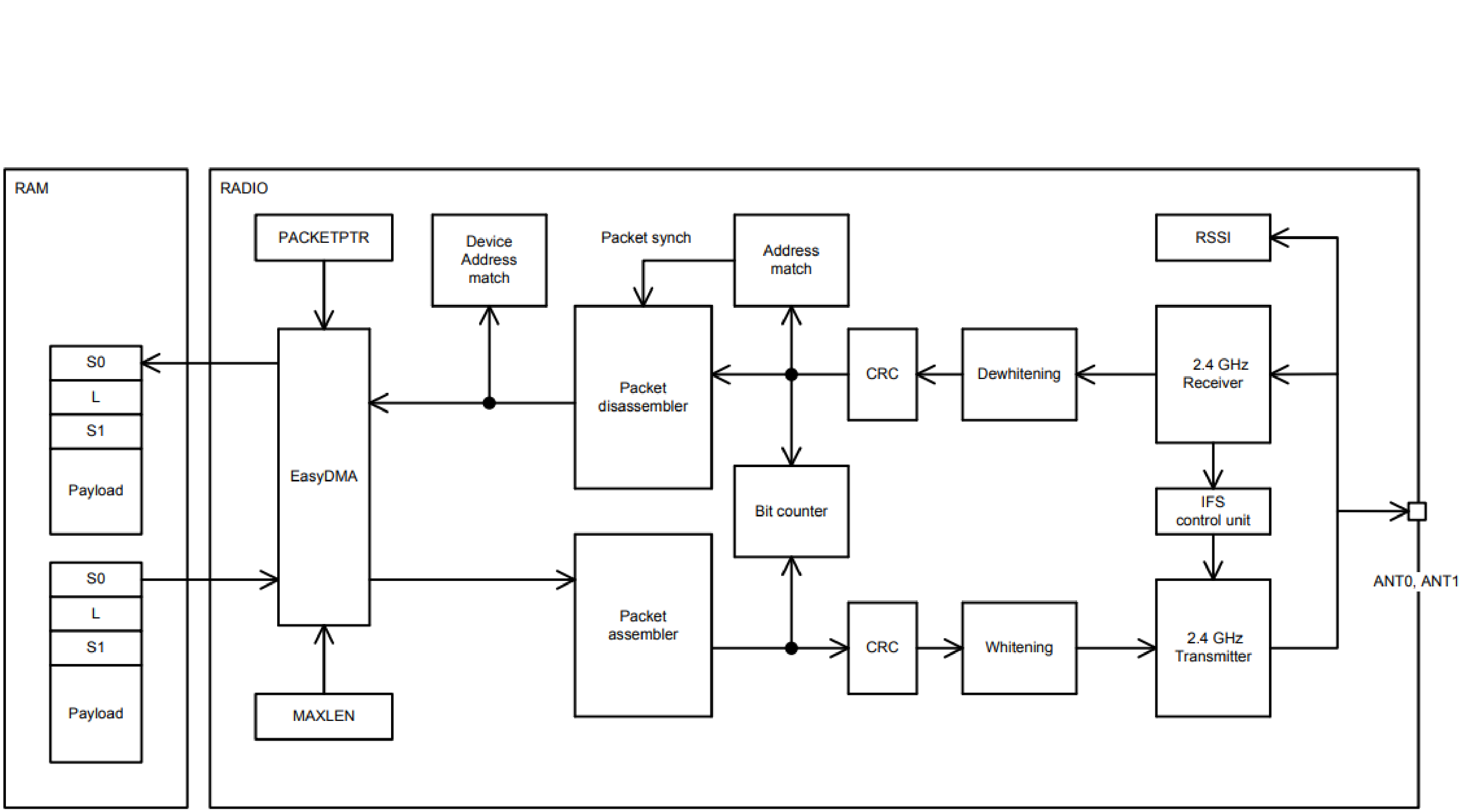


Figure 3.5: Block diagram for Radio with EasyDMA

Chapter 4

Analysis and Implementation

This chapter gives an overview of modification techniques, power measurement details, and implementation of modification techniques.

4.1 Analysis of possible modifications

Three modification techniques were identified from the literature review for the specialization project[23]. The modification techniques will now be analyzed to find the best modification technique for the existing DMA controller described in Section 3.4. Three modification techniques were determined based on memory architecture, bus architecture, and software. As a result of the analysis, one technique is considered for implementation in the existing DMA controller. Among three modification techniques, one technique is implemented in the existing DMA controller. The details about identified modification techniques and applied modification techniques are explained below.

One modification is the DMA controller functioning as a bridge between AHB and APB buses[6], allowing both buses to work in parallel. This modification technique is implemented for one master and multiple slaves setup. The existing DMA controller does not have multiple slaves accessing the data bus. In the existing DMA controller, the peripherals have individual DMA, which can access memory through AHB multi-layer. In the existing DMA controller, DmaChannelPeripheralCore handles all the logistics between a peripheral on the DMA bus and memory on the AHB bus. The existing DMA controller does not need to work as a bridge between AHB and APB buses. Instead, the existing DMA controller has a core module to handle DMA and AHB buses.

Another modification technique is to modify the algorithm design to optimize power at a system level[25]. The proposed DMA had a source address generator, which is used to generate the address of a source port, and it also had a destination address generator to generate the address for the destination port. The width of an input signal of the source address generator and destination address generator was reduced to 2 bits instead of 4 bits by changing the way of coding. It is said that the reduction in the width of a signal resulted in both power and area reduction[25]. In the existing DMA controller, a default width of the signals is defined, hence it is not possible to apply this modification technique.

The existing DMA core module has an internal FIFO(First in First out) for the alignment of data. The FIFO contains a buffer, which is connected to the AHB bus or DMA bus depending on the direction of data transfer. A counter is used to keep track of the number of bytes in FIFO. Another modification technique, which is chosen to implement in existing DMA controllers, is to modify this memory architecture. The existing DMA controller and this modification technique have similar features. The modification technique is adopted because of the similar architecture and functionality. The experiments are carried out at different widths of a buffer in the existing DMA controller. This technique is proposed to introduce cache functionality to increase the speed of execution and reduce energy consumption[24]. The proposed optimization technique used excess hardware to implement cache functionality. The DMA carries out data transfer in different phases for improved buffer management. This technique is only suitable for single-core processors, and evaluating parallel execution is out of reach. This technique is modified to be more appropriate for the existing DMA controller.

The existing DMA controller buffer transfers 8 bits at a time, where the width of the data bus is 32 bits. The modification technique applied is to increase the width of the buffer to 16 and 32 bits at a time and measure the power consumption values for three cases. The analysis of power consumption values is performed to define the effect of buffer width on power consumption.

Another modification technique that is considered for implementation in the existing DMA controller is to increase the depth of FIFO to perform burst transfer of data instead of transferring one byte at a time. During a burst transfer, a large amount of data is transferred in a short period, which results in faster data transfer speeds. Burst data transfer in a DMA controller can also reduce power consumption. This is because burst transfer allows for the transfer of a large amount of data in a shorter amount of time, which reduces the time that the DMA controller needs to be active, resulting in reduced power consumption. The current buffer in the existing Dma controller can only transfer one byte at a time and optimization is to increase the buffer size, so that it can transfer multiple bytes at a time instead of one byte at a time. The burst data transfer in DMA can also reduce the number of clock signals needed to transfer data because, in burst data transfer, multiple bytes of data are transferred once instead of transferring byte by byte.

4.2 Power Measurements for Existing DMA Controller

Spyglass is used to measure power consumption values and the Questa sim is used to run simulations. The power consumption values are measured for the entire simulation as well as different scenarios. Two scenarios are defined using FSDB dumper, which is high activity and low activity. The high activity and low activity timings are defined from the simulation waveform. The high activity file indicates the time required to transfer 256 bytes and the low activity file indicates the idle time between the data transfer. The power consumption values are measured for high activity and low activity with different buffer sizes. The Questa sim waveform generator is used in this thesis and the waveform is used to measure the time between signals. An example of a Questa sim-generated simulation waveform which shows both high activity and low activity when transferring 8 bits at a time is shown in Figure4.4. The existing DMA is configured to transfer 8 bits of data at a time and the power consumption values are measured for the existing DMA controller. An example of a Questa sim-generated simulation waveform in Questa sim when transferring 8 bits at a time is shown in Figure 4.3.

The simulation waveform indicating the transfer of 256 bytes when transferring 8 bits at a time is shown in Figure 4.1. Once taskStart is active, DCP starts prefetching from the start address in memory. Prefetching continues until the internal buffer cannot accept the next incoming transaction and must transfer data to the peripheral. The dmaReqP is a DMA request signal from the peripheral and for M2P transfer dmaReqP is set to high. An ahbHtrans signal indicates an AHB transfer type to memory. when ahbHtrans is 00, which indicates it is in idle transfer, and when ahbHtrans is 01, which indicates it's in data transfer type. The ahbHaddr is the AHB address to memory and the Data is DMA data to peripheral. The eventDMAend is high after data transfer which indicates the DMA end condition has been reached. DMAweP is DMA write enable to peripheral and ahbHRdata is AHB read data from memory. From simulation waveform Figure 4.1 it is observed that 32 bus transfers are needed to transfer data when the buffer width is 8. The simulation waveform which indicates high and low activity is shown in Figure 4.2. From the simulation waveform, it can be observed that when transferring 8 bits at a time high activity duration is more compared to transferring 16 and 32 bits at a time. The duration of high and low activities changes when transferring 16 and 32 bits.

A spyglass setup is performed for the existing DMA controller to measure power consumption values. To set up the spyglass appropriately for the design following steps are performed.

- Required files are modified according to the design
- A sanity check is performed on spyglass and the outcome is analyzed using the tool's GUI(Graphical User Interface)

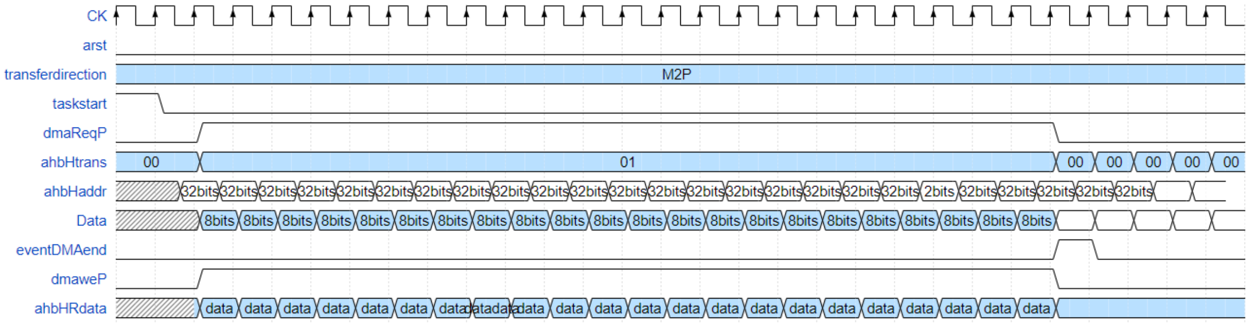


Figure 4.1: Simulation waveform when buffer width is 8

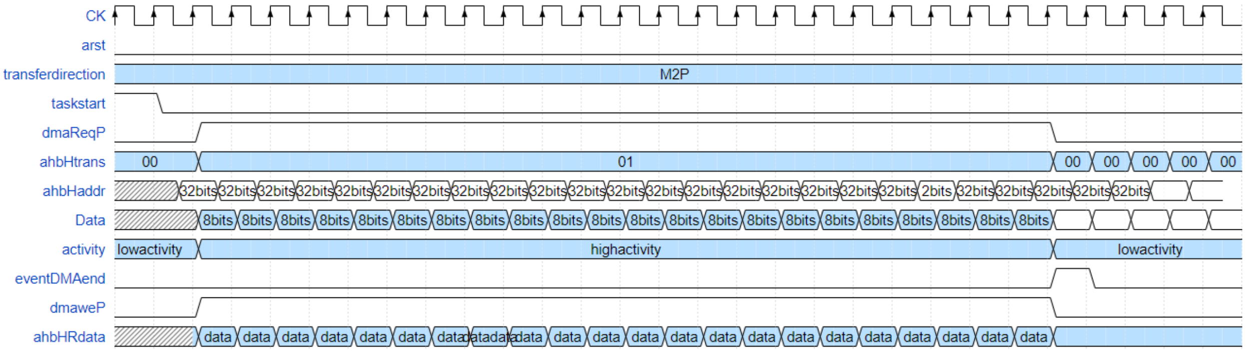
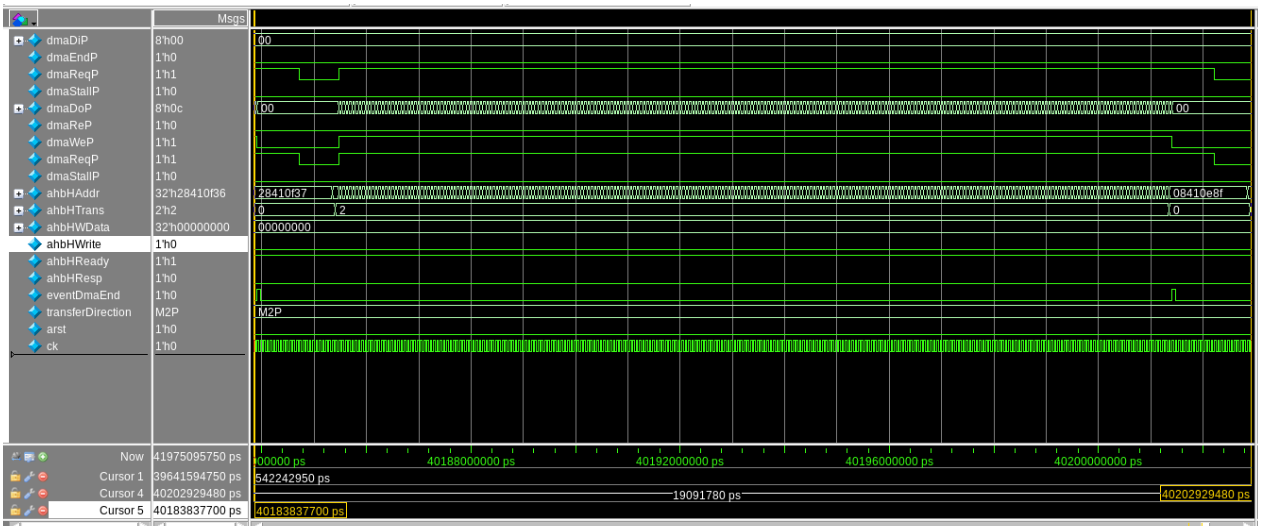


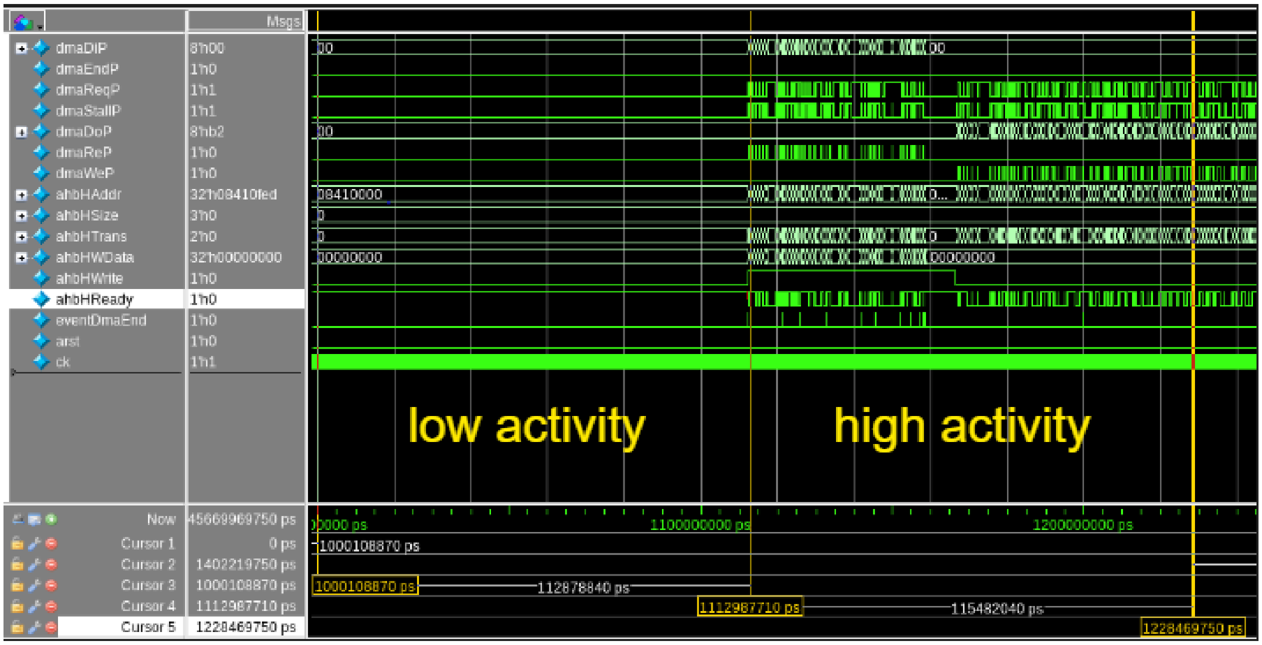
Figure 4.2: simulation waveform for high activity and low activity when buffer width is 8



	Msgs
dmaDiP	8'h00
dmaEndP	1'h0
dmaReqP	1'h1
dmaStallP	1'h0
dmaDoP	8'h0c
dmaReP	1'h0
dmaWeP	1'h1
dmaReqP	1'h1
dmaStallP	1'h0
ahbHAddr	32'h28410f36
ahbHTrans	2'h2
ahbHWData	32'h00000000
ahbHWrite	1'h0
ahbHReady	1'h1
ahbHResp	1'h0
eventDmaEnd	1'h0
transferDirection	M2P
arst	1'h0
ck	1'h0

Now	41975095750 ps
Cursor 1	39641594750 ps
Cursor 4	40202929480 ps
Cursor 5	40183837700 ps

Figure 4.3: Questa sim full Simulation waveform when buffer width is 8



	Msgs
dmaDIP	8'h00
dmaEndP	1'h0
dmaReqP	1'h1
dmaStallP	1'h1
dmaDoP	8'hb2
dmaReP	1'h0
dmaWeP	1'h0
ahbHAddr	32'h08410fed
ahbHSize	3'h0
ahbHTrans	2'h0
ahbHWData	32'h00000000
ahbHWrite	1'h0
ahbHReady	1'h0
eventDmaEnd	1'h0
arst	1'h0
ck	1'h1
Now	45669969750 ps
Cursor 1	0 ps
Cursor 2	1402219750 ps
Cursor 3	1000108870 ps
Cursor 4	1112987710 ps
Cursor 5	1228469750 ps

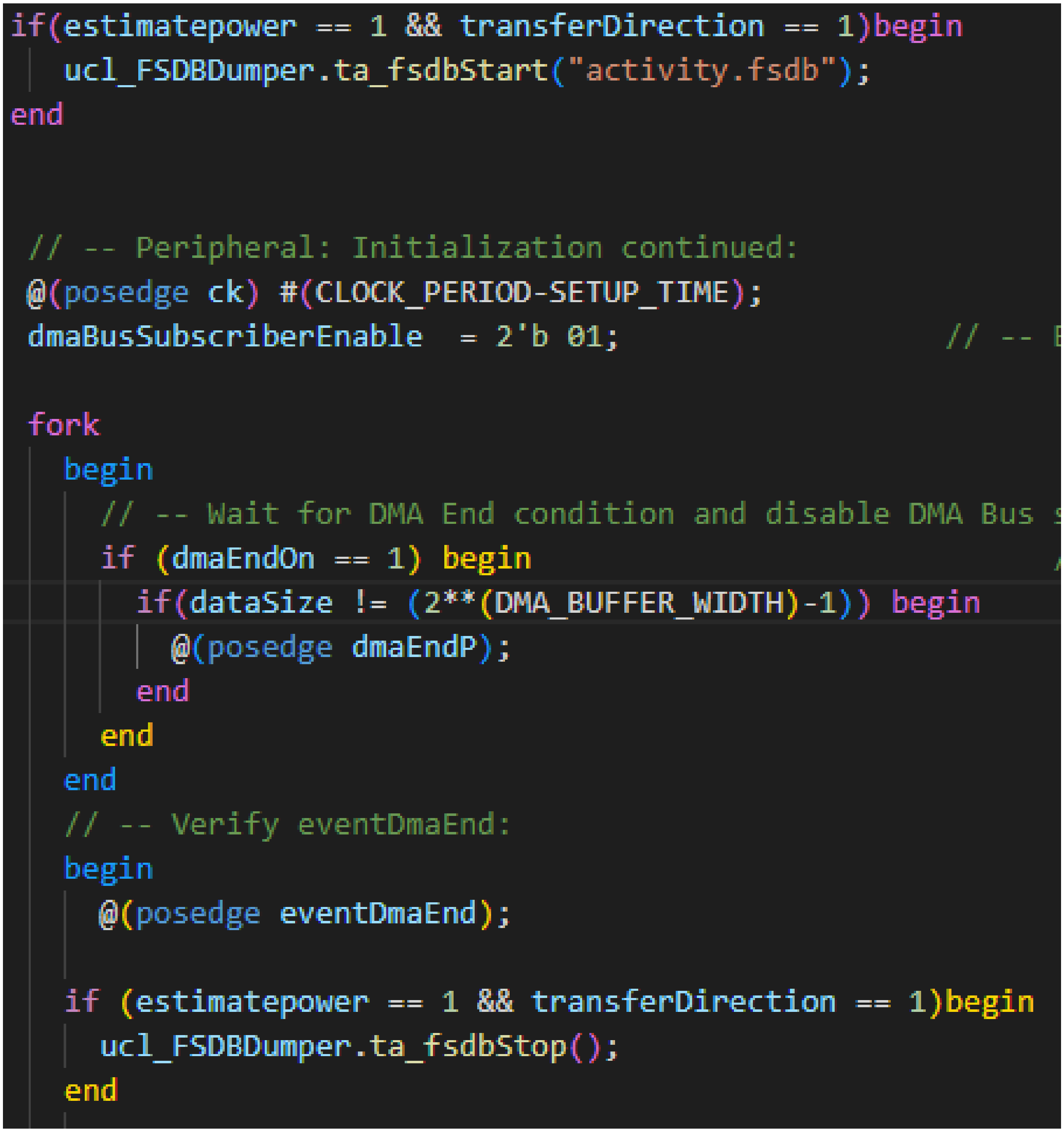
Figure 4.4: Questa sim simulation waveform for High Activity and Low Activity when buffer width is 8

- The above steps are repeated until all errors and warnings are fixed
- The FSDB file, which contains activity data of a simulation is generated
- Activity check is performed on scenarios, it will verify all signals and reports clock frequency
- Finally, power estimation is performed on scenarios and power consumption values can be seen in GUI

The power estimation results for the existing DMA controller are reported in Chapter 5. The power estimation values of high activity and low activity at different time intervals for the existing DMA controller are also reported in the same chapter.

4.3 Power optimizations

The modification chosen to implement in the existing DMA controller is to increase buffer width, which is to be able to transfer 16 and 32 bits at a time. The buffer width is changed by making some modifications to the code provided by Nordic Semiconductor which is shown in Figure 4.5. After the modifications, the DMA was been able to transfer 16 and 32 bits of data at a time. Another modification is turning off the clock during low activity between data transfers.



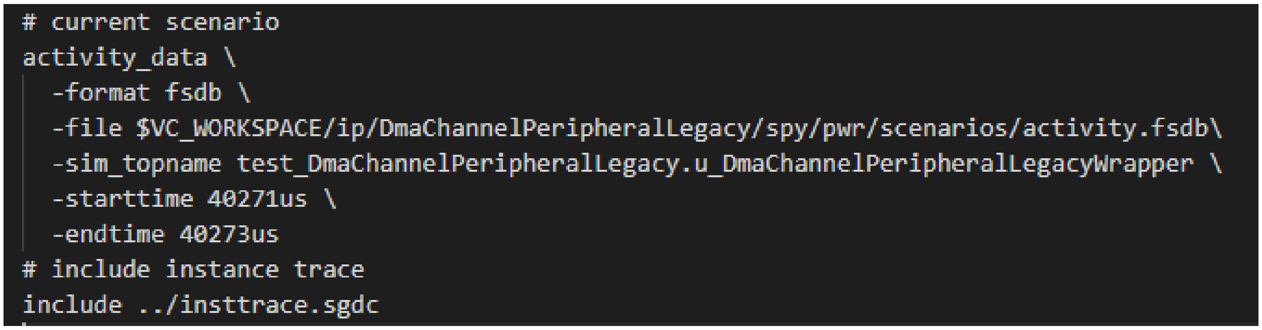
```
if(estimatepower == 1 && transferDirection == 1)begin
   ucl_FSDBDumper.ta_fsdbStart("activity.fsdb");
end


 // -- Peripheral: Initialization continued:
 @(posedge ck) #(CLOCK_PERIOD-SETUP_TIME);
 dmaBusSubscriberEnable  = 2'b 01;                 // -- E

 fork
   begin
     // -- Wait for DMA End condition and disable DMA Bus s
     if (dmaEndOn == 1) begin                             /
       if(dataSize != (2**(DMA_BUFFER_WIDTH)-1)) begin
         @(posedge dmaEndP);
       end
     end
   end
   // -- Verify eventDmaEnd:
   begin
     @(posedge eventDmaEnd);

   if (estimatepower == 1 && transferDirection == 1)begin
     ucl_FSDBDumper.ta_fsdbStop();
   end
```

Figure 4.5: Measurement of power consumption using FSDB

4.3.1 Optimization of Buffer width

The width of DMA data is increased to 16 bits at a time and 32 bits at a time, and the buffer size is increased to 16 and 32, respectively. When the buffer size is 8, it takes 32 bus transfers which is a long data transfer. When the buffer width is 16, it reduces the number of bus transfers to 16, which is a short data transfer. When the buffer width is 32, it reduces the number of bus transfers to 8 and data transfer is faster.

The power consumption values are measured at different widths of a buffer. The simulation waveform when transferring 16 bits at a time is shown in Figure 4.7, from the simulation waveform it is observed that the number of bus transfers is reduced to half compared to transferring 8 bits at a time. The simulation waveform when transferring 32 bits at a time is shown in Figure 4.9. It is observed from the simulation waveform that the number of bus transfers is reduced to half compared to transferring 16 bits at a time and bus transfers are reduced two times compared to transferring at a time. It is also noted that when buffer width increases simulation time also increases. The simulation waveform for high activity and low activity when transferring 16 bits at a time and buffer width is 16 is shown in Figure 4.8. The definition of high activity and low activity files in testbench is shown in Figure 4.6 The simulation waveform for high activity and low activity when transferring 32 bits at a time and buffer width is 32 is shown in Figure 4.10. From the simulation waveform, it is observed that the duration of high activity is reduced to 16 bits and 32 bits compared to transferring 8 bits at a time and the duration of low activity increases to 16 bits and 32 bits compared to transferring 8 bits at a time.



```
# current scenario
activity_data \
  -format fsdb \
  -file $VC_WORKSPACE/ip/DmaChannelPeripheralLegacy/spy/pwr/scenarios/activity.fsdb\
  -sim_topname test_DmaChannelPeripheralLegacy.u_DmaChannelPeripheralLegacyWrapper \
  -starttime 40271us \
  -endtime 40273us
# include instance trace
include ../insttrace.sgdc
```

Figure 4.6: Definition of high activity and low activity

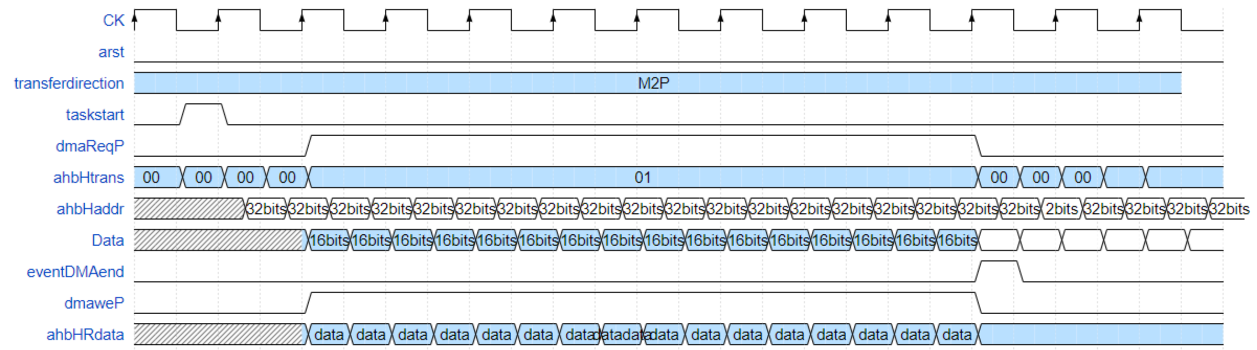


Figure 4.7: Simulation waveform when buffer width is 16

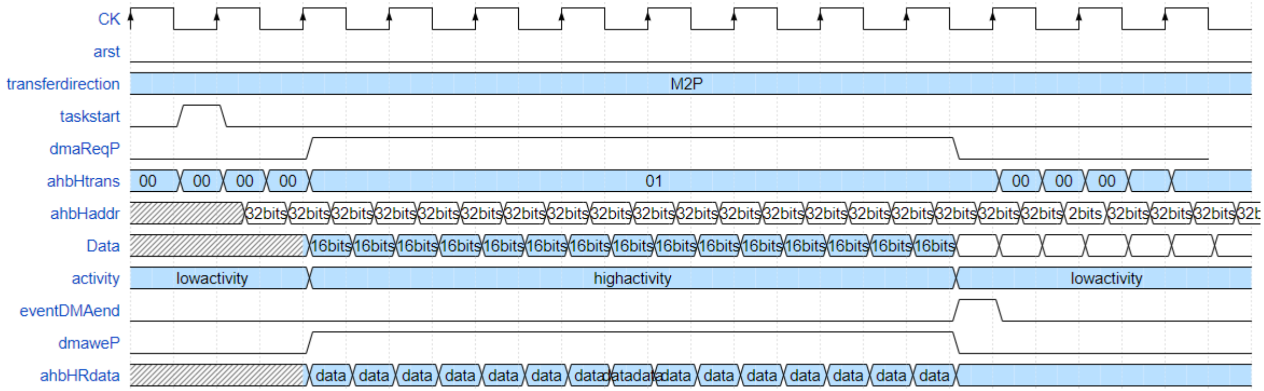


Figure 4.8: Simulation waveform for high activity and low activity when buffer width is 16

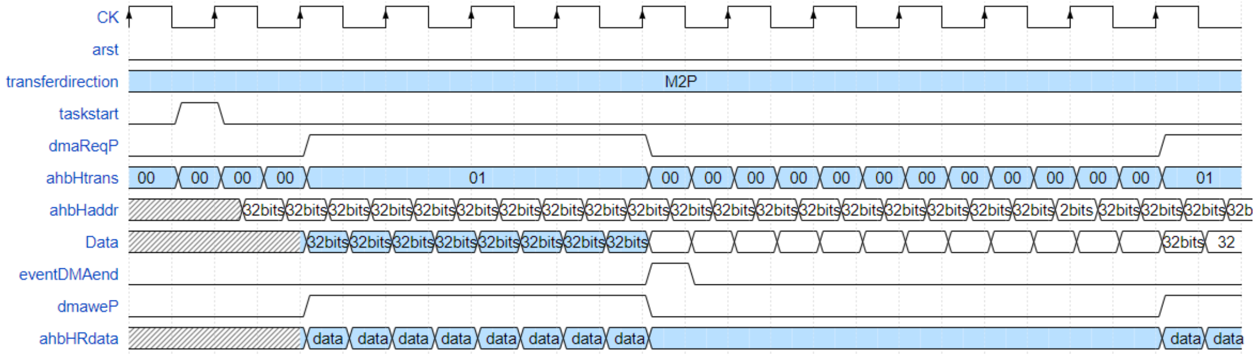


Figure 4.9: Simulation waveform when buffer width is 32

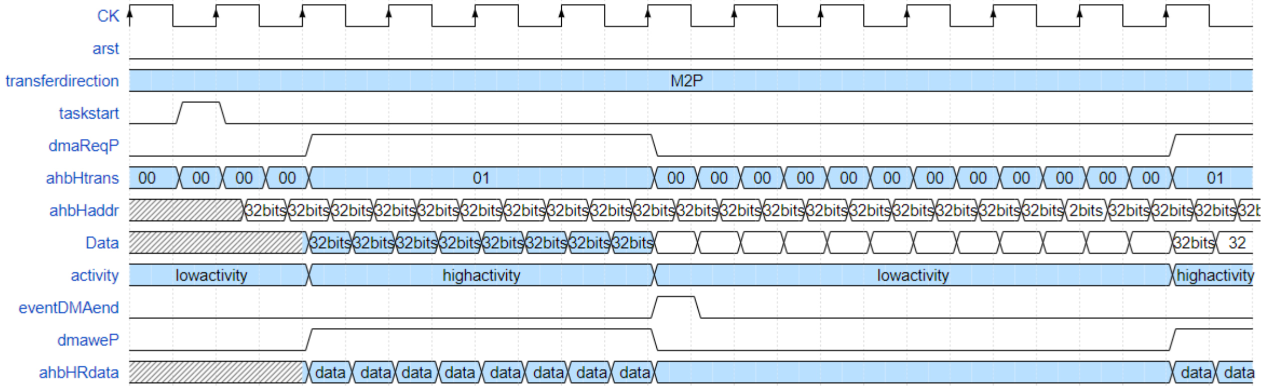


Figure 4.10: Simulation waveform for high activity and low activity when buffer width is 32

4.3.2 Optimization of Clock gating

The optimization in clock gating is considered for implementation in the existing DMA controller to turn off the clock during low activity to reduce power consumption. In general, when the DMA controller is not actively transferring data, it may still be consuming power to maintain its state and to keep the clock signal running. Clock gating can be implemented in the existing DMA controller by turning off the clock signal to the DMA controller after a data transfer is complete, and turning it back on again when a new data transfer is ready to be initiated. The modification technique which is experimented with is to turn off the clock during low activity. The clock gating experiments are performed in the DMA controller, but they were not completed due to bugs in the tools and there wasn't enough time to fix the bugs. The duration of the clock gating period between data transfers can vary depending on the number of bits transferring at a time and buffer width. Longer clock gating periods can save more power but may result in longer latency when initiating new data transfers.

Chapter 5

Results and Discussions

The power consumption values for full simulation, high activity, and low activity were measured for the existing DMA controller. The power consumption values are measured using a test bench, which is provided by Nordic Semiconductor.

5.0.1 Power consumption for existing DMA

The measured power consumption values for the existing DMA controller are shown in Table 5.1. The high-activity and low-activity files are defined for the existing DMA controller and the timing intervals for both activities are measured from the simulation waveform. From the table, it is observed that power consumption is more high activity when transferring 8 bits at a time. During the data transfer in high activity DMA controller continuously fetches data from the source and writes it to the destination. In low activity, the power consumption is reported because the clock is running and some signal transitions are going on.

5.0.2 Power consumption for 16-bit DMA

The measured power consumption values for 16 bit DMA controller are shown in Table 5.2. For the 16-bit DMA controller, the power consumption values are more than doubled compared to the existing DMA controller. The power consumption is increased due to an increase in data

Scenarios	Transfer direction	Power consumption
Full simulation	M2P	37.38nW
High activity	M2P	32.12nW
Low activity	M2P	1.12nW

Table 5.1: Power consumption values for existing DMA controller

Scenarios	Transfer direction	Power consumption
Full simulation	M2P	120.5nW
High activity	M2P	107.4nW
Low activity	M2P	1.2nW

Table 5.2: Power consumption values for 16-bit DMA

Scenarios	Transfer direction	Power consumption
Full simulation	M2P	209.06nW
High activity	M2P	167.1nW
Low activity	M2P	1.62nW

Table 5.3: Power consumption values for 32-bit DMA

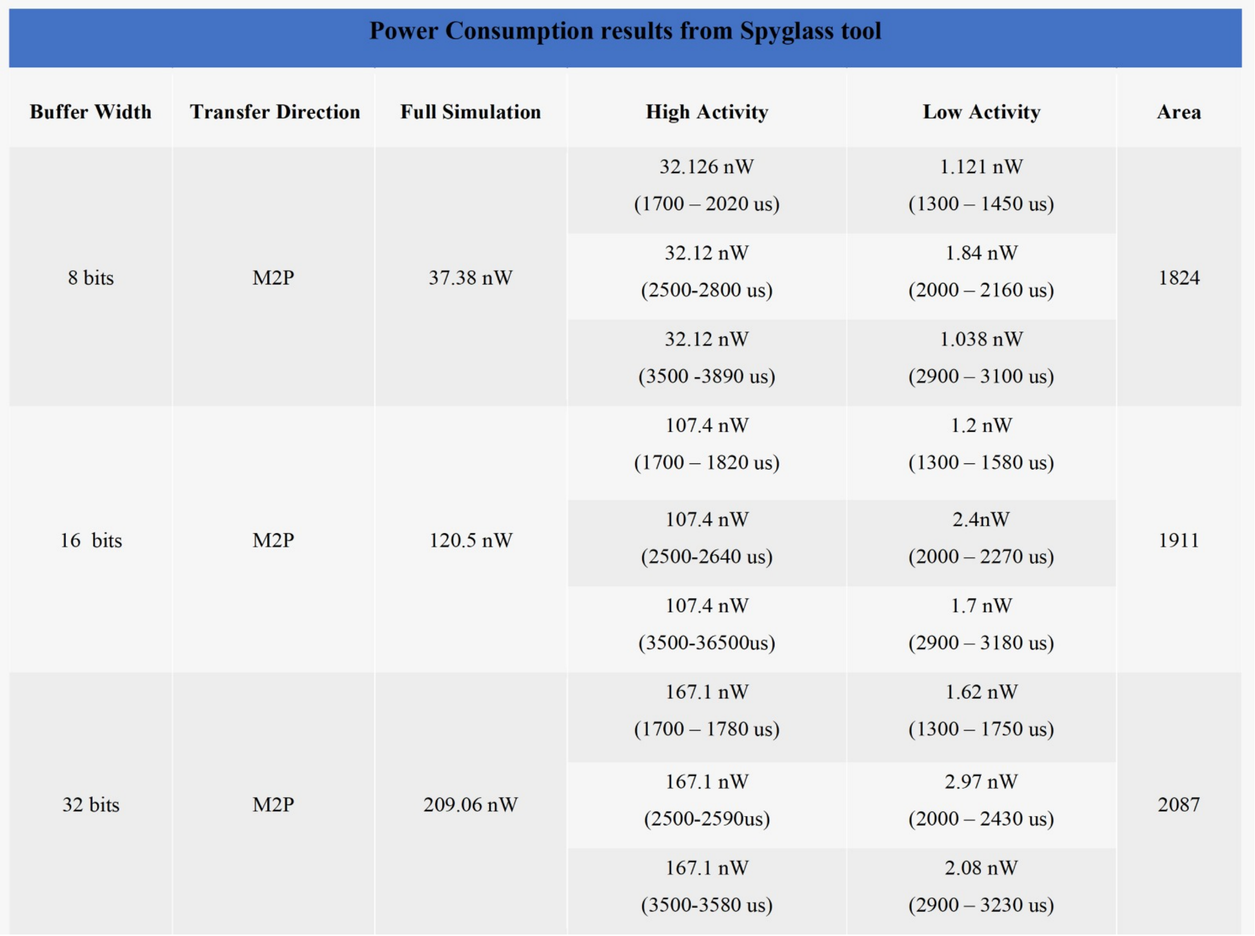
lines and additional hardware components to increase the buffer width to 16. The duration of high activity and low activity are measured from the simulation waveform.

5.0.3 Power consumption for 32-bit DMA

The measured power consumption values for 32 bit DMA controller are shown in Table 5.3. The power consumption values are increased for the 32-bit DMA controller compared to the 16-bit and existing DMA controllers. The low activity time duration increases for 32 bit DMA controller.

5.0.4 Discussion of overall results

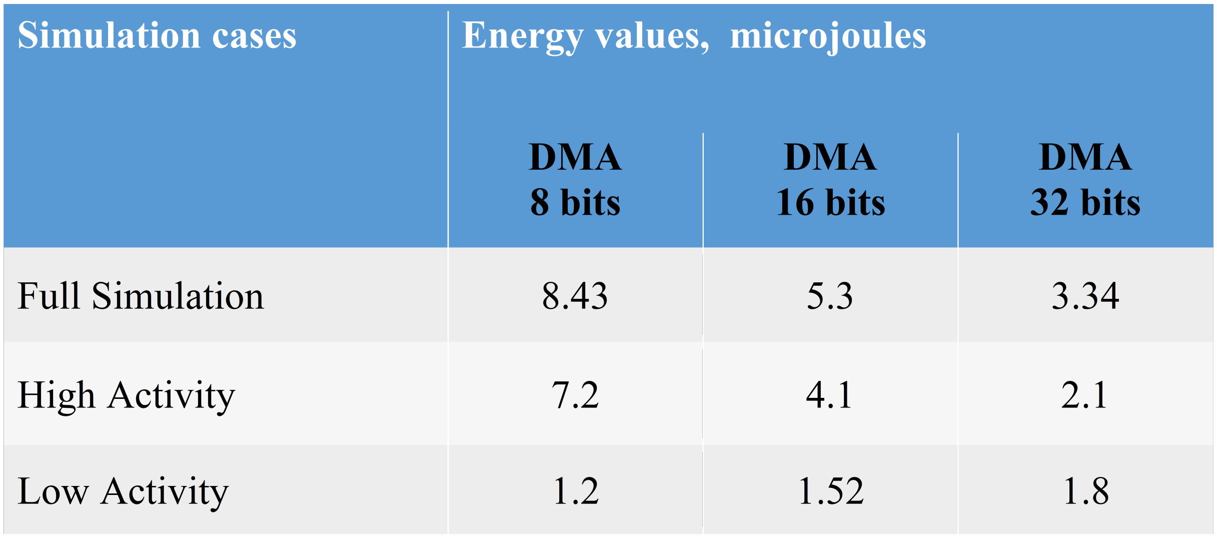
The power consumption values for the DMA controller with high activity and low activity defined at different time intervals are shown in Table 5.4. Different time intervals of high activity and low activity are defined from the simulation waveform. The high activity power consumption values are the same for different time intervals regardless of time intervals because high activity time duration is defined during data transfer. In the high activity, the time duration is less for 16-bit and 32-bit and the current consumption is more for data transfer. Whereas the low activity power consumption values are changing at different time intervals, which is due to the measurement of time intervals for low activity at different points in a simulation and leakage current in low activity. In one case the time interval for low activity is defined before the initial data transfer started, where the power consumption values are different compared to the time interval between data transfers. In another case, the time interval for low activity is defined at the very end of the simulation, which is the idle state after the final data transfer. In another case of low activity, the time interval is defined between data transfers. Due to the simulation point where time intervals are measured for low activity power consumption values are different in different time intervals. The area of the DMA controller is increasing when the buffer width is 16 bits and 32 bits because the DMA controller has other components in addition to the buffer.



Power Consumption results from Spyglass tool

Buffer Width	Transfer Direction	Full Simulation	High Activity	Low Activity	Area
8 bits	M2P	37.38 nW	32.126 nW (1700 – 2020 us)	1.121 nW (1300 – 1450 us)	1824
			32.12 nW (2500-2800 us)	1.84 nW (2000 – 2160 us)	
			32.12 nW (3500 -3890 us)	1.038 nW (2900 – 3100 us)	
16 bits	M2P	120.5 nW	107.4 nW (1700 – 1820 us)	1.2 nW (1300 – 1580 us)	1911
			107.4 nW (2500-2640 us)	2.4nW (2000 – 2270 us)	
			107.4 nW (3500-36500us)	1.7 nW (2900 – 3180 us)	
32 bits	M2P	209.06 nW	167.1 nW (1700 – 1780 us)	1.62 nW (1300 – 1750 us)	2087
			167.1 nW (2500-2590us)	2.97 nW (2000 – 2430 us)	
			167.1 nW (3500-3580 us)	2.08 nW (2900 – 3230 us)	

Table 5.4: Power consumption values for optimized DMA controller



Simulation cases	Energy values, microjoules		
	DMA 8 bits	DMA 16 bits	DMA 32 bits
Full Simulation	8.43	5.3	3.34
High Activity	7.2	4.1	2.1
Low Activity	1.2	1.52	1.8

Table 5.5: Energy consumption values for DMA controller

The energy consumption values for existing DMA, 16-bit DMA, and 32-bit DMA controllers are presented in Table 5.5. In 16-bit and 32-bit DMA the bus is wider the energy consumption is less compared to existing DMA. The high-activity and low-activity energy consumption values are different because of the time interval in which the values are calculated. In 32-bit DMA the memory is occupied less compared to 16-bit and existing DMA controllers. When DMA is 8 bits 32 bus transfers are required to transfer 256 bytes of data, hence energy required for transferring 256 bytes of data is energy consumption values multiplied by 32. When DMA is 16 bits and 32 bits the energy required to transfer 256 bytes is multiplied by 16 and 8 respectively. The results for energy consumption for the full simulation, high activity, and low activity for 8 and 32 bits are shown in Figure 5.1. The full simulation time period for existing DMA, 16-bit DMA, and 32-bit DMA controllers is the same. The energy consumption values for high activity and low activity are calculated for one high-activity period defined from the simulation waveform. The time period at which energy values are calculated for low activity is the time period before the initial data transfer started.

The clock gating experiments are performed which is to turn off the clock between data transfers but the experimental values are not provided due to challenges with the simulations and tools. Assuming that the power consumption values are less for clock gating compared to running with the clock. For the existing DMA controller, the idle period between data transfers is less. In 16-bit and 32-bit DMA controllers the idle period between data transfers is more compared to existing DMA controller. For example, the current consumption is 100uA when all components are active. In the current consumption of 100uA, the data generation part which is to generate one bit at a time causes 40% of current consumption and the DMA channel causes 60% of cur-

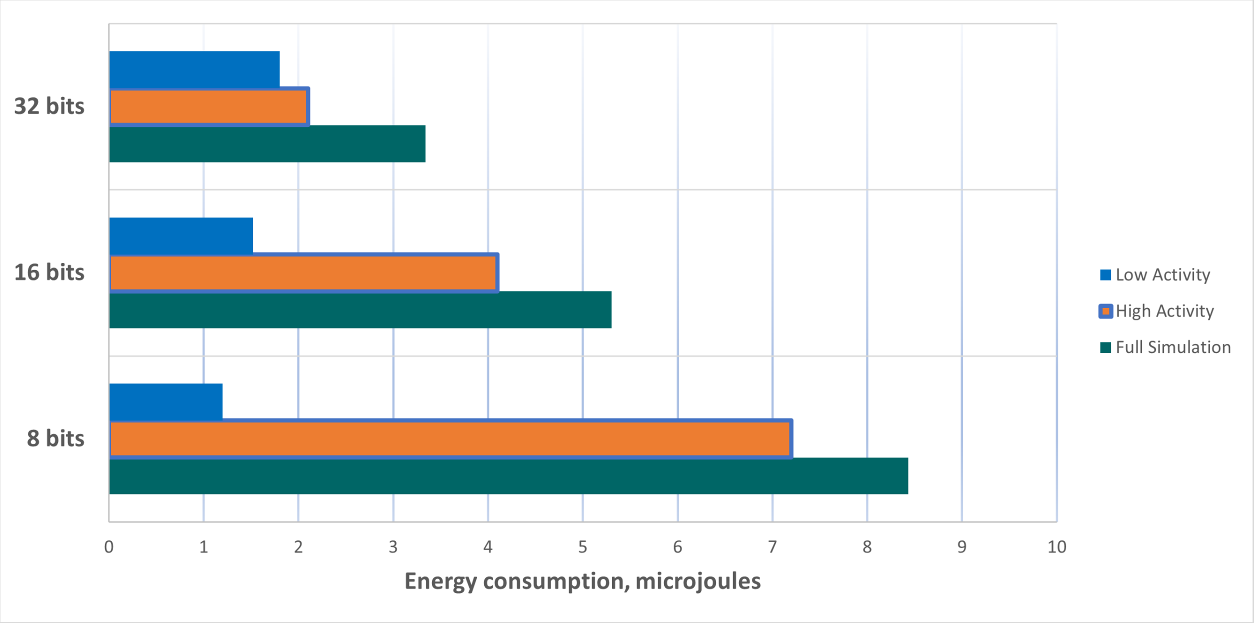


Figure 5.1: Energy consumption values for 8-bit, 16-bit, and 32-bit DMA controllers

rent consumption. If the clock is not stopped the current consumption is 60% all the time from the DMA channel. With the clock gating for the existing DMA, 16-bit DMA, and 32-bit DMA controllers it is possible to stop the clock 50% of the time, 75% of the time, and even more time respectively.

Chapter 6

Conclusion and Future work

Direct Memory Access controllers play a crucial role in chip performance. The DMA controller is used to transfer the data blocks between memory locations and external devices without intererupting the execution flow of the CPU. The introduction of the DMA controller in chips also presented significant benefits in terms of power consumption. This thesis started by analyzing the optimization techniques from the specialization project to identify the applicable modification technique for the existing DMA controller, i.e., the Nordic DMA controller. The modification technique selected for implementation is to increase the buffer width and data bus width in the Nordic DMA controller. The simulation waveforms for existing(8-bit), 16-bit, and 32-bit DMA controllers for different buffer widths are provided are included. Two activity files are developed to calculate the power consumption at different time intervals and the time intervals are defined from simulation waveforms. The power consumption values at different time intervals are calculated for existing 8-bit DMA controller, 16-bit, and 32-bit DMA controllers. Power consumption results for different DMA controllers are generated. A comparison of power consumption for existing 8-bit DMA, 16-bit, and 32-bit DMA controllers for full simulation, high activity, and low activity scenarios has been performed. Further, energy consumption values are calculated for different DMA controllers for these above-mentioned scenarios. The energy consumption was reduced by 37% and 60% for 16-bit and 32-bit DMA controllers respectively compared to 8-bit DMA controller. The energy consumption was reduced for both the full simulation and high activity scenarios whereas the energy consumption was slightly increased during low activity due to the defined low activity time when the buffer width was changed from 8-bit to 16-bit and 32-bit. Overall energy consumption of 16-bit and 32-bit DMA controllers was reduced compared to 8-bit DMA controller.

For future work, the clock-gating technique can be implemented to reduce the power consumption for the DMA controller. The implementation of clock gating can be carried out in a DMA controller with different buffer widths. Another future improvement can be Burst data transfer, instead of sending one byte at a time multiple bytes of data can be transferred in one clock cycle, which can provide interesting results in the power consumption of a DMA controller. The changes in the FIFO depth experiment can also be performed in the future.

Bibliography

[1] A. Ahmed, Mohammed Altaf and Abdullah, “Design and implementation of a direct memory access controller for embedded applications, international journal of technology,” vol. 10, 2019.

[2] R. Bhutada and Y. Manoli, “Complex clock gating with integrated clock gating logic cell,” in *2007 International Conference on Design Technology of Integrated Systems in Nanoscale Era*, 2007, pp. 164–169.

[3] V. Konstantakos, K. Kosmatopoulos, S. Nikolaidis, and T. Laopoulos, “Measurement of power consumption in digital systems.” *IEEE T. Instrumentation and Measurement*, vol. 55, pp. 1662–1670, 01 2006.

[4] H. Morales, C. Duran, and E. Roa, “A low-area direct memory access controller architecture for a risc-v based low-power microcontroller,” in *2019 IEEE 10th Latin American Symposium on Circuits Systems (LASCAS)*, 2019, pp. 97–100.

[5] R. Schrader, T. Ax, C. Röhrig, and C. Fühner, “Advertising power consumption of bluetooth low energy systems,” in *2016 3rd International Symposium on Wireless Systems within the Conferences on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS-SWS)*, 2016, pp. 62–68.

[6] A. Aljumah, Abdullah and Altaf, “Amba based advanced dma controller for soc, international journal of advanced computer science and applications,” vol. 7, 2016, doi = 10.14569/IJACSA.2016.070326.

[7] M. Anumothu, “Design and analysis of dma controller for system on chip based applications,” *International Journal of VLSI and Embedded Systems-IJVES*, vol. 07, pp. 1685–1690, 06 2016.

[8] A. Aljumah and A. Ahmed, “Amba based advanced dma controller for soc,” *International Journal of Advanced Computer Science and Applications*, vol. 7, 03 2016.

[9] F. Z. Harmouch, N. Krami, and N. Hmina, "A multiagent-based decentralized energy management system for power exchange minimization in microgrid cluster," *Sustainable Cities and Society*, vol. 40, pp. 416–427, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2210670717306170

[10] K. Chen, L. Qi, and H. Yu, "Design of two-dimension dma controller in media multiprocessor soc," in *2008 Second International Symposium on Intelligent Information Technology Application*, vol. 2, 2008, pp. 708–711.

[11] A. M and P. M. H, "Design and implementation of power estimation technique for digital circuits, international journal of engineering research technology (ijert)," vol. 3, 2014, issn = 2278-0181, url=https://www.ijert.org/research/design-and-implementation-of-power-estimation-technique-for-digital-circuits-IJERTV3IS041503.pdf.

[12] B. Goel and S. A. McKee, "A methodology for modeling dynamic and static power consumption for multicore processors," in *2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2016, pp. 273–282.

[13] R. Mullins, "Minimising dynamic power consumption in on-chip networks," in *2006 International Symposium on System-on-Chip*, 2006, pp. 1–4.

[14] R. A. A. G. Michael Keating, David Flynn and K. Shi, *"Low Power Methodology Manual For System-on-Chip Design"*, ser. Integrated Circuits and Systems (ICIR). Springer New York, NY, 2008.

[15] P. Jamieson, W. Luk, S. J. Wilton, and G. A. Constantinides, "An energy and power consumption analysis of fpga routing architectures," in *2009 International Conference on Field-Programmable Technology*, 2009, pp. 324–327.

[16] N. Srinivasan, N. S. Prakash, Shalakha D., Sivaranjani D., S. Sri Lakshmi G., and B. B. T. Sundari, "Power reduction by clock gating technique," *Procedia Technology*, vol. 21, pp. 631–635, 2015, sMART GRID TECHNOLOGIES. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2212017315003035

[17] P. Khem, "Physical and silicon measures of low power clock gating success: An apple to apple case study," in *HPCA '05: Proceedings of the 11th International Symposium on High-Performance Computer Architecture*. SNUG, 2007.

[18] F. Agnes Shiny Rachel.N and Akilandeswari.S, "Integration of clock gating and power gating in digital circuits", 5th international conference on advanced computing communication systems (icaccs)," 2019.

[19] H. D. Xiaohui Fan, Yangbo Wu and J. Hu, "A low leakage autonomous data retention flip-flop with power gating technique," *Journal of Electrical and Computer Engineering,*, vol. 2014, no. Article ID 695832, p. 10, 2014.

[20] E. Macii, L. Bolzani, A. Calimera, A. Macii, and M. Poncino, "Integrating clock gating and power gating for combined dynamic and leakage power optimization in digital cmos circuits," in *2008 11th EUROMICRO Conference on Digital System Design Architectures, Methods and Tools*, 2008, pp. 298–303.

[21] H. Mahmoodi-Meimand and K. Roy, "Data-retention flip-flops for power-down applications," in *2004 IEEE International Symposium on Circuits and Systems (IEEE Cat. No.04CH37512)*, vol. 2, 2004, pp. II–677.

[22] P. Coussy, D. D. Gajski, M. Meredith, and A. Takach, "An introduction to high-level synthesis," *IEEE Design Test of Computers*, vol. 26, no. 4, pp. 8–17, 2009.

[23] V. A. Jampala, "Analysis of power benefit from dma," *Specialization Project Report,TFE4580 Electronic Systems Design and Innovation, Norwegian University of Science and Technology(NTNU)*, p. 38, 2021.

[24] X. M. Nikola Vujic, Lluc Alvarez and E. Ayguade, "Dma-circular: an enhanced high level programmable dma controller for optimized management of on-chip local memories, the 9th conference," p. 113, 2012, isbn = 978-1-4503-1215-8, publisher=ACM Press, URL = http://dl.acm.org/citation.cfm?doid=2212908.2212925.

[25] S. Shirur and Aishwarya, "Design and implementation of efficient direct memory access (dma) controller in multiprocessor soc, 2018 international conference on networking, embedded and wireless systems, icnews 2018 - proceedings," 2018, doi = 10.1109/ICNEWS.2018.8903991.

[26] A. Bakshi, A. Burman, and D. chakraborty, "Development of dma controller for real time data processing in fpga based embedded application," *IOSR journal of VLSI and Signal Processing*, vol. 4, pp. 01–08, 01 2014.

[27] T. Enami, K. Kawakami, and H. Yamazaki, "Dma-driven control method for low power sensor node," in *2015 IEEE Topical Conference on Wireless Sensors and Sensor Networks (WiSNet)*, 2015, pp. 53–55.

[28] M. Graphics, *ModelSim® User's Manual*, Mentor Graphics Corporation. [Online]. Available: https://www.microsemi.com/document-portal/doc_view/131619-modelsim-user

[29] S. R. Nesset, “Rtl power estimation flow and its use in power optimization,” Master’s thesis, NTNU, 2018. [Online]. Available: http://hdl.handle.net/11250/2558598

[30] S. P. Ram, “Rtl power estimation methods and strategies,” 2019. [Online]. Available: https://www.linkedin.com/pulse/rtl-power-estimation-methods-strategies-sampath-v-p

[31] S. Ravi, A. Raghunathan, and S. Chakradhar, “Efficient rtl power estimation for large designs,” in *16th International Conference on VLSI Design, 2003. Proceedings.*, 2003, pp. 431–439.

[32] NordicSemiconductor. nrf52832 product specification v1.8. [Online]. Available: https://infocenter.nordicsemi.com/topic/struct_nrf52/struct/nrf52832_ps.html

[33] G. Ma and H. He, “Design and implementation of an advanced dma controller on amba-based soc,” in *2009 IEEE 8th International Conference on ASIC*, 2009, pp. 419–422.

[34] Nordicsemiconductor. nrf51 series reference manual. [Online]. Available: https://infocenter.nordicsemi.com/topic/struct_nrf51/struct/nrf51_refmanual.html