8-2017

Energy and Performance Balancing Architecture for Asynchronous Data Processing Platforms

Chien-Wei Lo
University of Arkansas, Fayetteville

Energy and Performance Balancing Architecture for Asynchronous Data Processing Platforms

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy in Engineering

by

Chien-Wei Lo
The Pennsylvania State University
Bachelor of Science in Electrical Engineering, 2010
University of Bridgeport
Master of Science in Electrical Engineering, 2012

August 2017
University of Arkansas

This dissertation is approved for recommendation to the Graduate Council.

Dr. Jia Di
Dissertation Director

Dr. Dale Thompson
Committee Member

Dr. Jingxian Wu
Committee Member

Dr. James Parkerson
Committee Member

ABSTRACT

The semiconductor industry has been increasingly focused on the energy consumption and heat generation in CMOS-based integrated circuits (ICs) for its dominating impact on the system performance and reliability. Without clock-related timing constraints, asynchronous circuits have demonstrated unique flexibility in performance-energy tradeoffs compared to synchronous designs. This dissertation work presents the architecture capable of balancing energy and performance for asynchronous digital signal processing circuits using the Multi-Threshold NULL Convention Logic (MTNCL). Architecture implementing user-configurable adaptive dynamic voltage scaling (DVS) and data processing core disabling based on the detection and parameterization of system throughput are developed for MTNCL parallel homogeneous and heterogeneous platforms to optimally balance performance and energy efficiency. Simulation results and comparison with previously designed MTNCL homogeneous and heterogeneous platforms implementing only DVS show enhanced coherency between energy consumption and performance, and the improved effectiveness of DVS with core disabling in balancing the energy and performance of both platforms.

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to my advisor, Dr. Jia Di. I could not have completed my Ph.D. studies without his enlightenment and support. His deep knowledge and meticulous attitude toward research have been a great inspiration for me during my studies at the University of Arkansas and will continue to be throughout my professional career.

I am deeply thankful to my committee members: Dr. Jingxian Wu, Dr. Dale Thompson, and Dr. James Parkerson for their precious opinions and guidance towards my research work.

The great love and encouragement from my wife Siqi Ma, my mother Lin-Hwa Wang, and my father Kuo-Cheng Lo has always been the cornerstone of my Ph.D. studies. I am very much grateful of their support.

DEDICATION

This dissertation is dedicated to my wife, Siqi Ma, who always has love and faith in me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Introduction

Energy consumption and heat generation in CMOS-based integrated circuits (ICs) are dominating factors affecting the system performance and reliability. Many design techniques, e.g., parallel architecture, supply voltage scaling, and limiting switching activity, have been developed to optimally balance performance and energy consumption. Such balance is one of the primary aspirations and obstacles of future digital processors [1]. Asynchronous circuits have demonstrated unique flexibility during performance-energy tradeoffs compared to synchronous designs [2]. Such flexibility is also extended to near-threshold operating voltage regions [3]. Other notable precursors, such as security [4-6], scalability [7, 8], and extended temperature operations [9, 10], have expanded the utilization of asynchronous circuits into various technological areas.

Parallel computing, realized in multi-core processors [11], was developed in an attempt to enhance computing performance. The approach of parallelism is to utilize a number of processing cores with lower frequency to replace a single core with higher frequency. Parallel architecture can attain maximum speed permitted by the Amdahl's law through sending input data to the process cores and merging the outputs. Previous research [12] demonstrated that parallelism can be implement to Multi-Threshold NULL Convention Logic (MTNCL) systems for improved performance and power.

Dynamic Voltage Scaling (DVS) is the accordance use of power based on different external circumstances. For multi-processors systems, a variation-aware technique is proposed in [13]. Synchronous designs implementing DVS, restricted by overhead investment, commonly have a very limited set of voltage-frequency pairs and have to sacrifice performance for the process, temperature, and timing variation introduced. For delay-insensitive (DI) style

asynchronous designs like MTNCL, data propagation is amply time-independent and the circuit is correct-by-construction. Preceding research [14] indicated that DVS implemented with MTNCL design has mitigated issues present in synchronous designs and achieved a broad application of DVS across a wide range of voltages with improved reliability.

For MTNCL systems implementing parallelism and DVS, the supply voltage of each processing cores is scaled from the maximum allowed by the process to the minimum that transistors can still operate based on a one-to-one mapping relationship between supply voltage and system workload [14]. However, leakage power is still being consumed during idle periods as long as there is voltage supplied to the system. To avoid such leakage power consumption, the circuit block will need to be switched off by disconnecting the voltage supply. Disabling the power supply of processing cores presents itself as an extension to the supply voltage and workload mapping. Such extension can further enhance the accordance use of power based on system workload. Furthermore, processing cores with different functionalities will introduce variation in system workload. User configurable mapping of supply voltage and system workload provides improved coherency in the mapping relationship which results in better balancing of system power and performance. This dissertation work is to realize optimal balancing between energy and performance of MTNCL parallel platforms through the accurate control and modeling of the relationship between supply voltage and platform throughput.

1.1 Proposed Research and Approach

The proposed research is to develop an architecture capable of balancing the energy and performance of MTNCL parallel homogeneous and heterogeneous platforms across different application scenarios. DVS and core power supply disabling are incorporated in this architecture. The main aspects of the architecture using DVS and core-disabling are the capability to provide fine-grain control of power usage based on throughput and accommodate platform workload variation. The proposed architecture provides fine-grain power and throughput balancing through introducing processing core disabling as an addition to DVS. DVS proposed in [14] mapped 7 core supply voltages to 7 platform workloads. With the introduction of core-disabling, since each supply voltage can be paired with 4 different number of core-disabled for the platform with 4 cores, 20 core-voltage and core-on pairs (CVCOP) are now available to be mapped to 20 platform throughputs. This methodology has further enhanced the linearity between power usage and throughput.

The architecture is also capable of accommodating throughput variation introduced by implementing parallel cores of different functionalities through allowing the circuit designer to provide input on the maximum (max) and minimum (min) throughput of the core. The architecture derives the range of throughput based on these max and min numbers, parameterizes the range, and maps each to a CVCOP. In other words, the mapping relationship between throughput and CVCOP will always be updated by the architecture when different parallel cores are implemented. For MTNCL platforms, with increased input data rate, the core supply voltage and the number of cores enabled are raised to improve performance. On the other hand, when input data rate decreased, the core supply voltage is lowered and a number of cores are disabled to reduce power consumption. The scaling of supply voltage and disabling of

cores in accordance to throughput-based platform workload status detection ensures the platform only used the necessary amount of energy to sustain the required performance.

1.2 Dissertation Organization

Chapter 2 details the background information regarding DVS, asynchronous paradigm, and MTNCL homogeneous and heterogeneous platforms employed by this work. Chapter 3 presents the theory and design of energy and performance balancing controller. Chapter 4 contains the architecture of the MTNCL homogeneous platform with controller and the analysis of the platform's performance and energy consumption simulation results across different scenarios. Chapter 5 presents the architecture of the MTNCL heterogeneous platform with controller and the analysis of the platform's performance and energy consumption. Chapter 6 details the performance and energy consumption comparison between homogeneous and heterogeneous platforms implemented with either DVS or the controller. This chapter also includes a guideline to aid in the decision if the controller should be incorporated when a new platform is designed. Chapter 7 provides a summary of the innovations and achievements detailed in this dissertation, and outlines future work.

2 Background

2.1 Dynamic Voltage Scaling

DVS is the cornerstone for adaptive energy and performance balancing in digital systems. Digital systems using CMOS logic gates has active power dissipation characterized by $P_{dyn} = C_L V^2 f$. The P_{dyn} equation indicated that if the system supply voltage V scales down, the active power dissipation can reduce quadratically. This approach was first proposed in [15] and implemented on self-timed circuits. Low-power operation of the circuit was enabled by state detection and dynamic voltage scaling with the utilization of FIFO buffers.

DVS applied to synchronous circuits will have reduced the dynamic range to ensure proper functioning of the circuits under process, voltage, and temperature (PVT) variations. In [16], the paper demonstrates that for an 18×18 multiplier operating at 90 MHz with supply voltage scaled from 1.8V to 1.38V, the data error rate is 1.3% with 35% power reduction. As a variation of DVS, Adaptive Voltage Scaling (AVS) proposed in [17] is to scale the supply voltage to aid in timing closure, since frequency decrease with supply voltage. In [18], the research proposes a variation-aware technique for chip multiprocessors (CMPs) and the author of [19] assess various multi-core voltage-frequency island (VFI) methodologies. Panoptic Dynamic Voltage Scaling (PDVS) is introduced in [20], which utilized Local Voltage Dithering (LVD) when circuit operates in sub-threshold region for increased energy savings. Machine learning techniques is implemented in learning based DVS as demonstrates in [21] for adaptive circuit heat dissipation, performance, and energy balancing. Asynchronous data paths proposed in [22] are implemented between voltage areas for multi-rate signal processing applications since the voltage-frequency pairs of DVS implemented in synchronous circuits are constrained by hardware cost and extra control required to reduce energy. Furthermore, the timing variation and PVT variation introduced by DVS will also need to be accommodated for synchronous circuits to function

correctly. Activity detection proposed in [23] is designed to perform voltage scaling and reduce static power consumption for asynchronous network-on-chip (ANOC) nodes.

2.2 NULL Convention Logic (NCL)

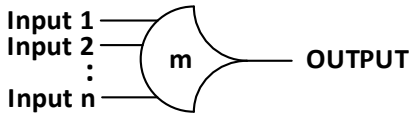


Figure 1: NCL THmn Threshold Gate

NULL Conventional Logic (NCL) is a DI style asynchronous paradigm. Unlike synchronous design subject to timing constraints, NCL circuits operate correctly whenever the transistors switch properly. NCL consists of 27 fundamental gates [24]. A generic gate with n inputs and a threshold of m is shown in Figure 1. NCL gates are able to hold states through hysteresis [24]. The condition for the output to be asserted is that at least *m* of the *n* inputs are asserted. Once an output is asserted, all inputs must be deasserted in order for the output to be deasserted. NCL designs employ dual-rail or quad-rail signals to attain delay-insensitivity. Dual-rail signals, as the name suggested incorporate two wires D^0 and D^1. D^0 and D^1 are used to represent four types of Boolean logic: D^0=1 and D^1=0 represent FALSE state in Boolean logic and DATA0 state in NCL. D^0=0 and D^1=1 represent TRUE in Boolean logic and DATA1 state in NCL. D^0=0 and D^1=0 represent NULL state in NCL, meaning that the data is not yet available. D^0=1 and D^1=1 represents an invalid state in NCL. The DATA/NULL cycle is controlled by handshaking protocol utilizing request and acknowledge signals, denoted as *Ki* and *Ko*.

2.3 Multi-Threshold NCL (MTNCL)

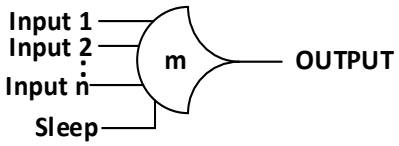


Figure 2: MTNCL Gate

For Multi-Threshold NULL Conventional Logic (MTNCL), sleep transistors are added to NCL gates for power gating and NULL state generation. When an MTNCL gate's sleep signal is high, the gate is asleep, and the output will be pulled low to generate a NULL state. The NULL state is inserted in between the current and previous DATA state to avoid overlapping. In addition to generating NULL state, the sleep transistors are used to reduce leakage power by shutting down the current flow from V_{DD} when the circuit is in NULL state. Similar to NCL, the handshaking signals *Ki* and *Ko* are utilized to control signals and no additional logic is required for sleep. MTNCL is easier to design compare to NCL and is much more energy efficient [25]. MTNCL gates are represented by an *m* inside the gates, indicating the implementation of sleep transistor, as shown in Figure 2.

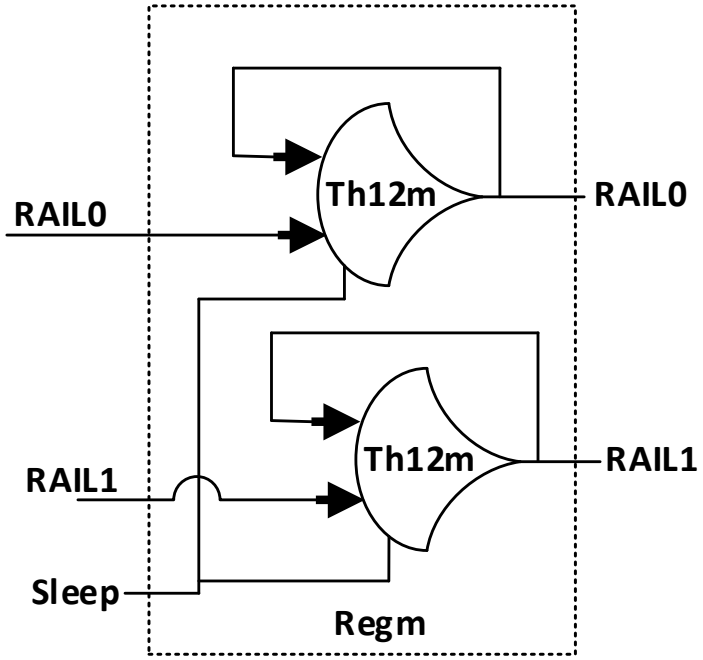


Figure 3: 1-bit Register in MTNCL

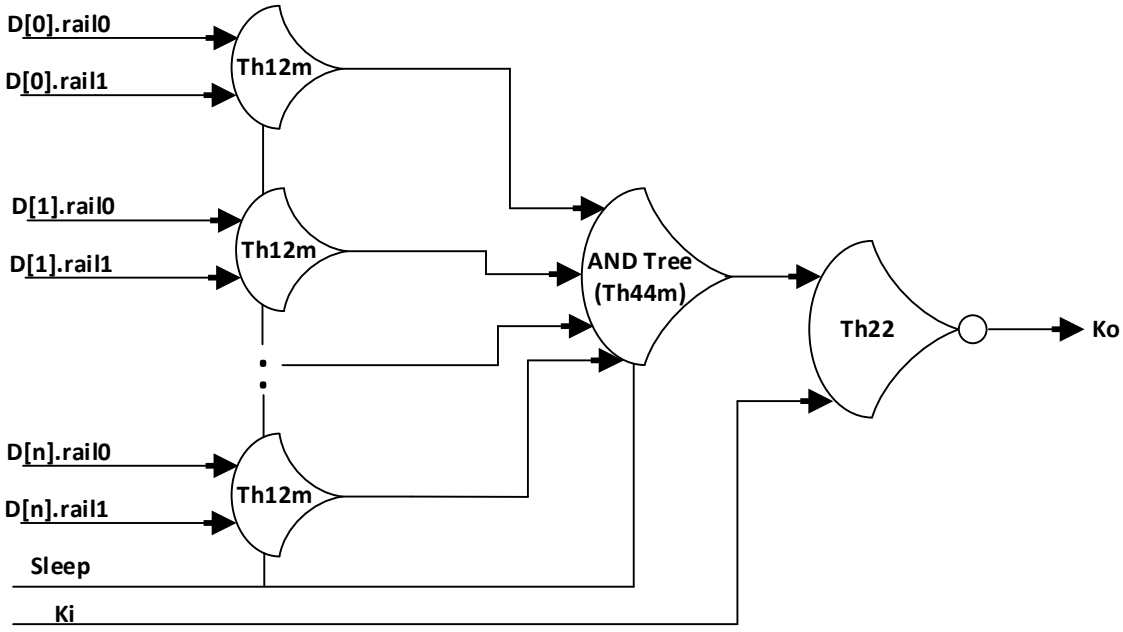


Figure 4: Completion Logic in MTNCL

The MTNCL pipeline architecture has DI combinational logic in between DI registers [26]. To avoid the previous DATA state from being overridden by the current DATA state, handshaking signals, *Ki* and *Ko* are incorporated with an exclusive NULL state inserted in

between the two DATA states. DI registers (*Regm*) are consisted of TH12m gates, as shown in Figure 3. When sleep is deasserted, DATA is registered and when sleep is asserted, output becomes NULL. Completion logic (*Comp*), as shown in Figure 4, will generate Data request signal (*Ko*), which becomes the data acknowledge signal (*Ki*) of the previous pipeline stage. When the next pipeline stage is requesting for data (*rfd*), meaning *Ki* is active, and all the dual-rail inputs are in valid DATA states, the DATA wavefront is going to be propagated to the next pipeline stage. Meanwhile, the output of the completion logic *Ko*, is deasserted to request for NULL *(rfn*), which assures a NULL wavefront always follow the DATA. Furthermore, *Ko* will be asserted to *rfd* if the next pipeline stage is requesting for NULL and a NULL wavefront is facilitated by the active sleep signal.

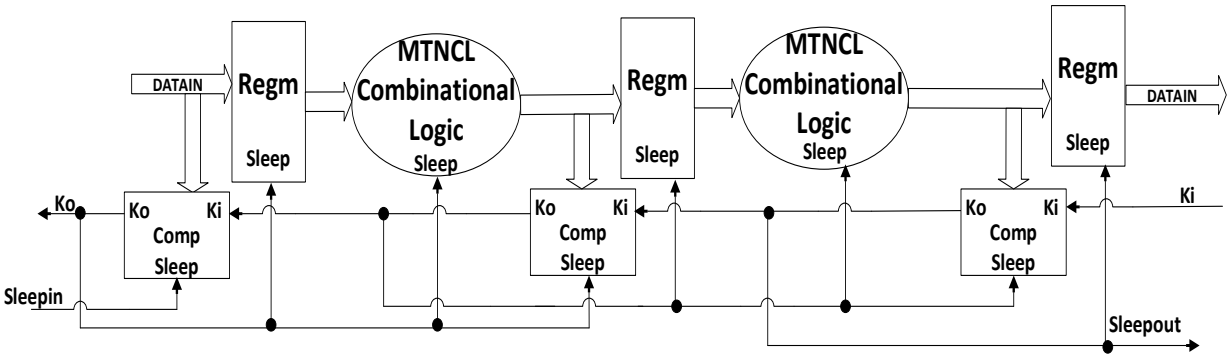


Figure 5: MTNCL Pipeline Architecture

The handshaking signals in the MTNCL pipeline architecture are used as the sleep control signals, no additional logic is needed. As shown in Figure 5, the completion logic output, *Ko*, serves as the sleep signal of next stage combinational MTNCL logic, the DI register, and the completion logic. At the beginning, all the components in the pipeline are in NULL state and all the *Ko* signals are in *rfd*. After the first DATA wavefront appears at the input ports, the completion logic will deassert *Ko* to *rfn*, which activates the following register and

combinational logic to propagate the input DATA. The deasserted *Ko* will maintain its value until subsequent NULL wavefront appears at the input ports and the completion logic is slept by the *sleepin* signal. The following register and combinational logic will be slept when *Ko* is asserted to *rfd* and generate a NULL wavefront. The DATA/NULL cycle will repeat to fill all the pipeline stages until the first DATA appears at the output ports. T_{dd}, the DATA-to-DATA cycle time, has a comparable aspect to the clock period in synchronous designs. MTNCL designs have demonstrated improved energy reduction as presented in [27]. A pipelined 8-bit×8-bit MTNCL multiplier has 150× and 1.8× leakage power and active energy reductions comparing to the NCL counterpart.

2.4 MTNCL Platform with Homogeneous Cores

A MTNCL platform with four homogeneous cores was previously designed to utilize parallelism and DVS in DI asynchronous circuits as shown in Figure 6 [14]. Demultiplexer (DEMUX) and Input Sequence Generator (ISG) are used to dispatch input data sequentially to each parallel core while multiplexer (MUX) and Output Sequence Generator (OSG) ensure the data exits the platform properly. When 4 parallel cores are active, the first 4 data are sequentially sent to each core for processing. The processed data will then go through the multiplexer and appear at the outputs in the original order. After the first 4 data, the platform will request another 4 data for processing.

The peripheral components, including the sequence generators and multiplexers, are required to operate at the maximum speed to ensure the functionality of the platform. Therefore, the peripheral components will have a fixed maximum voltage supply while the 4 parallel cores' supply voltages can be adjusted during run-time from the minimum supply voltage guaranteeing the operation of transistors to the maximum supply voltage specified by the technology.

In addition to the parallel architecture, a Voltage Control Unit (VCU) was designed to adjust the supply voltage dynamically based on the platform workload status. The VCU is consisted of a Pipeline Fullness Detector (PFD) for workload estimation, a Reference Voltage (*Vref*) Generator, and a Voltage Regulator for producing the dynamically scaled voltage for the cores.

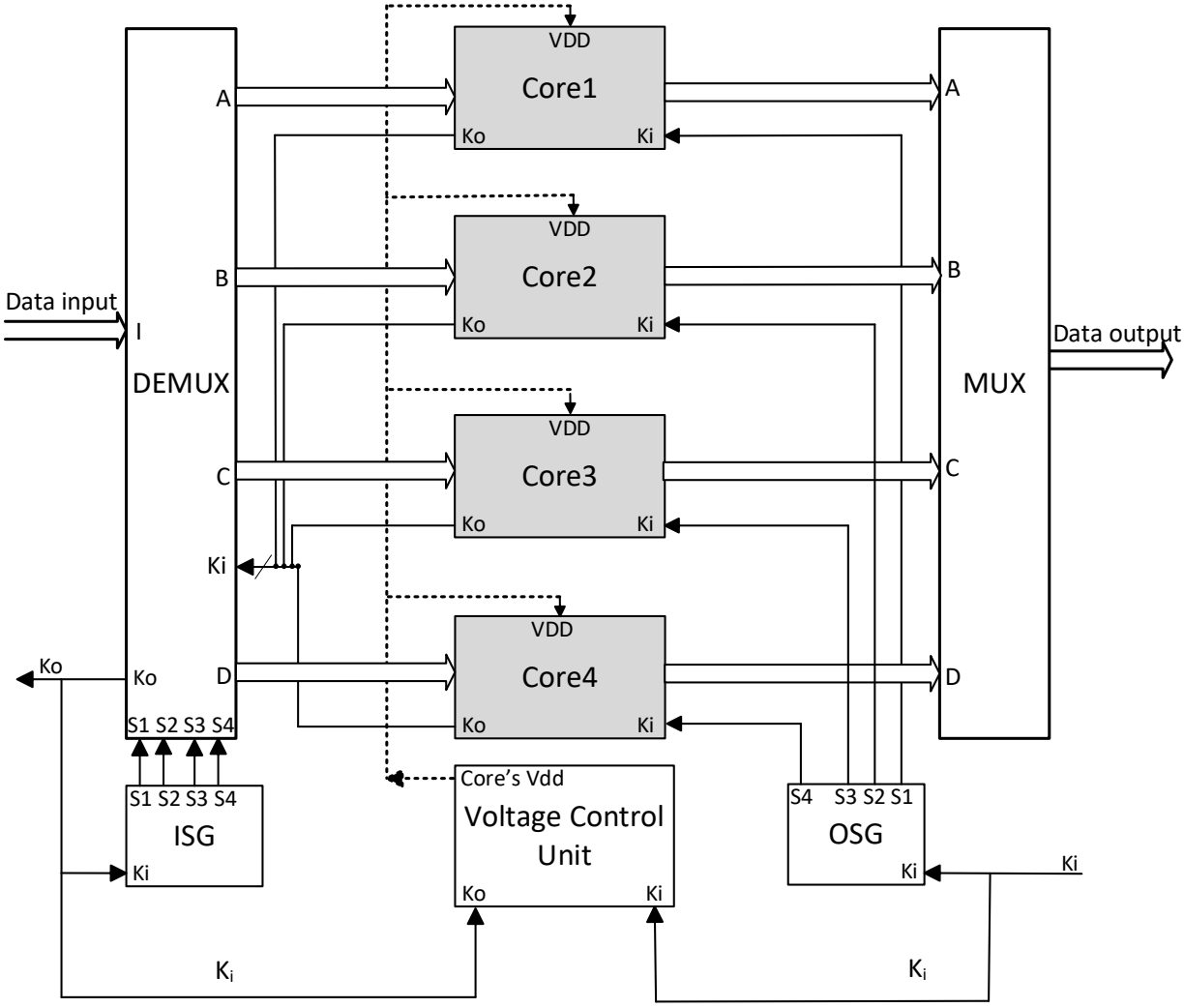


Figure 6: Homogeneous Platform Architecture

2.5 MTNCL Platform with Heterogeneous Cores

A MTNCL platform with four heterogeneous cores is shown in Figure 7 [14]. The heterogeneous platform was previously designed to mitigate the problem that when cores of

different functionalities are included, the overall performance will be dictated by the slowest core since all the faster cores need to wait for the slowest core to finish before the platform can request the next set of data. The platform will send data to a core once it requests for data, and the slowest core always has priority over other faster cores if two cores happen to request data within a short time period. A heterogeneous platform is designed with a generic and cascaded structure. Figure 7 shows the platform top-view including 4 cores. At the beginning, all the cores are at *rfd* states, and only one core will be granted data by the arbiter while others hold the same states. After the demultiplexer sends data to the granted core, its *Ko* signal will be de-asserted to *rfn*. Once this round is finished, the arbiter will then grant another core's request for input data.

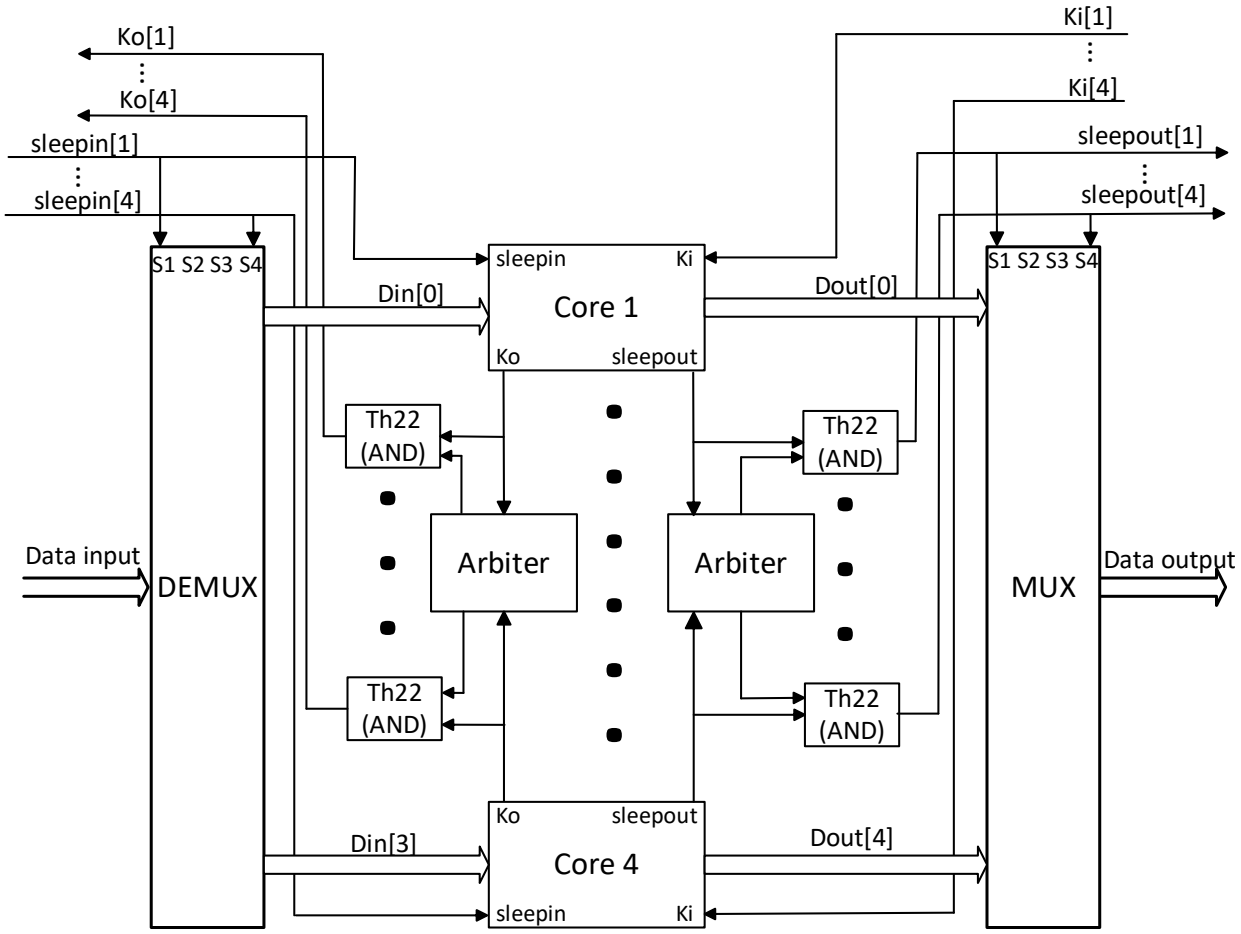


Figure 7: Heterogeneous Platform Architecture

3 Controller Enabling Adaptive Core Voltage Scaling and Core-Disabling

3.1 Design of the Controller

The energy and performance balancing architecture incorporates a controller as the major component responsible for adjusting core supply voltage and switching cores on/off based on platform throughput. The controller is designed to perform three kinds of tasks:

(a) Parameterize core T_{dd} range configured by the circuit designer, and map each T_{dd} to a CVCOP.

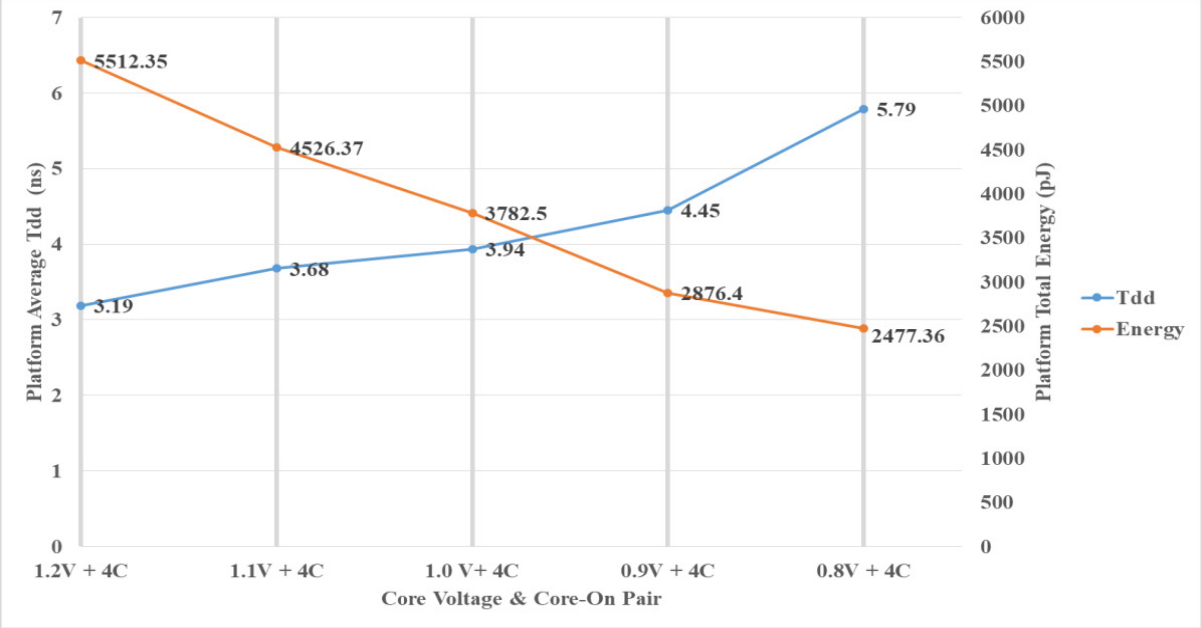
(b) Monitor platform throughput and decides which CVCOP best balance energy consumption and performance.

(c) Adjust core supply voltage and/or disabling cores based on the decision made in (b).

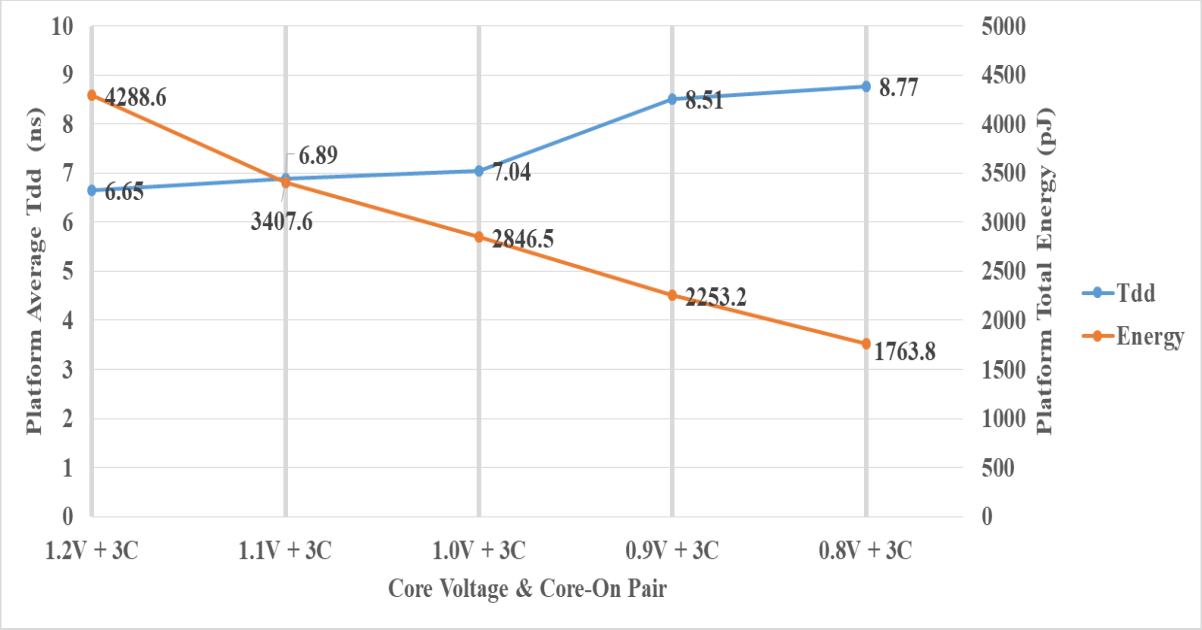
3.1.1 User Configurable Voltage-Workload Mapping

Previous research [14] incorporating DVS is to scale the core supply voltage from the maximum allowed by the process to the minimum that transistors can still operate based on a one-to-one mapping relationship between the core's supply voltage and the platform throughput, which mapped 7 different core supply voltage to 7 platform throughputs. The addition of core-disabling mechanism to DVS has extended the range and resolution of this one-to-one mapping relationship to totally 20 CVCOP available to map to 20 platform throughputs. Combing DVS and core-disabling therefore enables fine-grain control of power usage based on the platform throughput. Furthermore, the 4 FIR filter cores implemented in the homogeneous platform can be swapped in the future with cores of different functionalities which will introduce variation in the platform throughput, implying that cores implemented with different functionalities requires different mapping between the platform throughput and CVCOP. By enabling the user to be able to configure the maximum and minimum T_{dd} of the parallel cores to the controller, the mapping

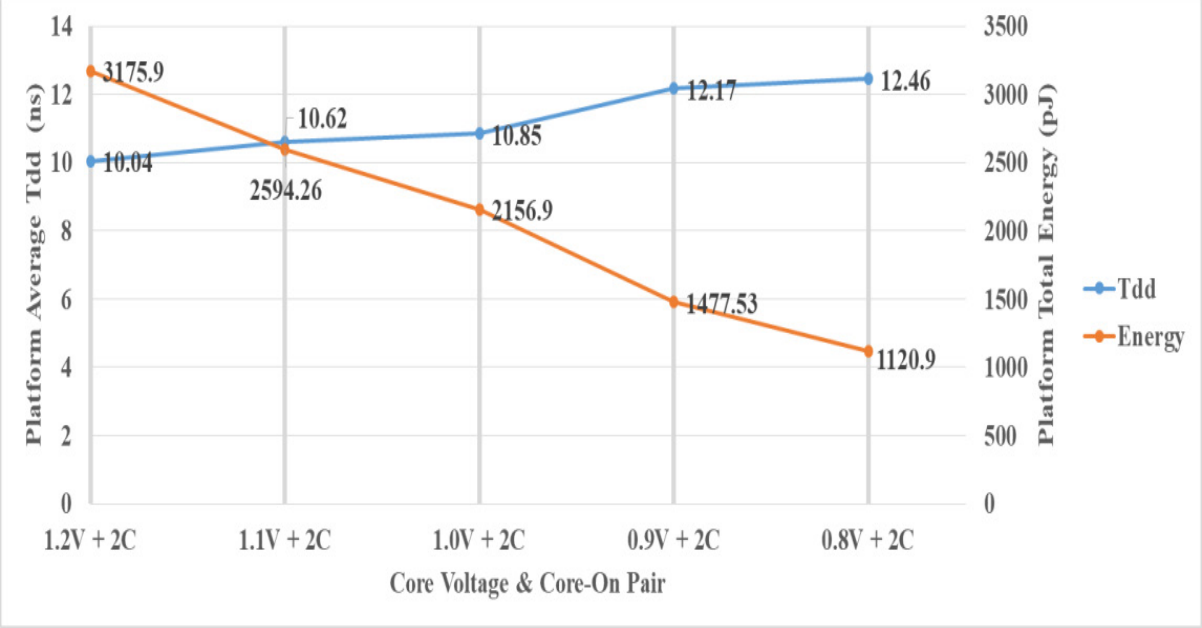
relationship between platform throughput and CVCOP is always closely correlated. To develop the mapping methodology between platform throughput and CVCOP, a case study is conducted on MTNCL homogeneous platform with 4 FIR filter cores realized at IBM 130nm technology node [14]. Each of the 20 CVCOP is applied to the platform with 40 input data patterns so platform total energy and platform average throughput can be measured.



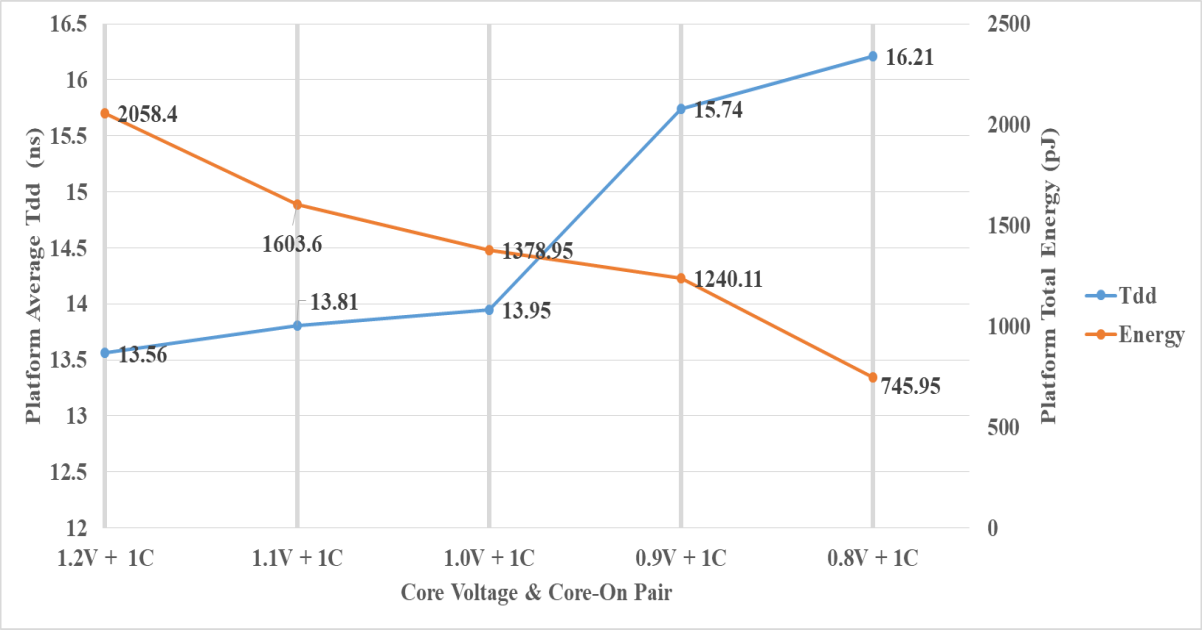
(a) 4 Cores On



(b) 3 Cores On



(c) 2 Cores On



(d) 1 Core On

Figure 8: Homogeneous Platform T_{dd} and Energy Trend across Different CVCOP

Figure 8 demonstrates the average T_{dd} and the total energy consumption measured under 20 different CVCOP of homogeneous platform. Figure 8 (a) indicated that the platform throughput becomes slower with reduce in supply voltage with all 4 cores turned on. Ideally, the platform throughput should have an unidirectional relationship with the platform total energy, the slower the throughput, the lower the energy. However, the data in Figure 8 representing platform total energy consumption indicate a non-unidirectional mapping relationship between platform energy and platform T_{dd}. For example, at the points 1.2V+3C and 1.1V+3C, the energy consumption is higher than neighboring points 0.8V+4C and 1.0V+3C with less than 1ns T_{dd} difference. This implies that when T_{dd} is in the range of 6ns to 7ns, instead of choosing 1.2V+3C and 1.1V+3C, 0.8V+4C or 1.0V+3C should be chosen for lower energy consumption and minimal impact on platform throughput.

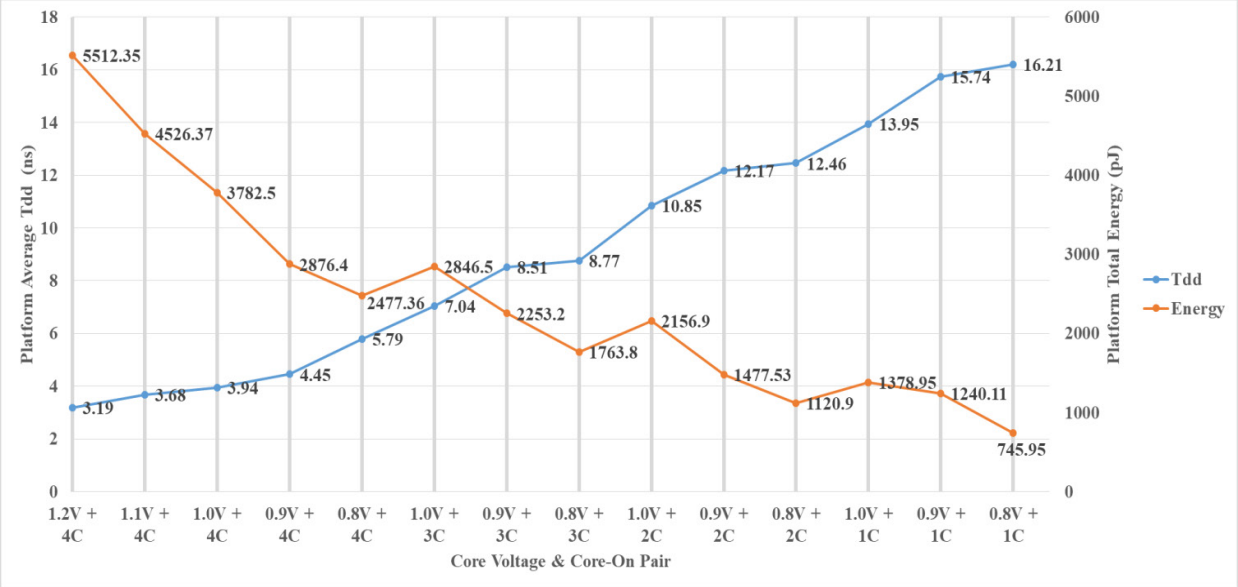


Figure 9: Controller Throughput – CVCOP Mapping Strategy

Applying the above-mentioned methodology, the mapping relationship between platform energy and platform T_{dd} is inversely proportional as demonstrates in Figure 9. Figure 9 indicates that instead of using 20 different CVCOP to map to 20 platform T_{dd}, 6 CVCOP: 1.2V+3C, 1.1V+3C, 1.2V+2C, 1.1V+2C, 1.2V+1C, 1.1V+1C are removed since these combinations has similar throughput but higher energy consumption compares to other CVCOP. The rest 14 pairs provide improved coherency between platform energy and platform T_{dd} as shown in Figure 9. Better energy and T_{dd} coherency implies that energy is used accordingly to throughput. The relationship between throughput and CVCOP observed in Figure 9 can be used to develop the throughput parametrization methodology and subsequently create a mapping between throughput and CVCOP. Controller is designed to allow the circuit designer of the FIR filter cores to configure the maximum and minimum T_{dd} range of the parallel cores, which is 16ns and 3ns. Controller performs a calculation of subtracting 3 from 16 and dividing the value by 14. Each divided value will be rounded-up then be assigned to the 14 CVCOP, creating the mapping table between platform T_{dd} and CVCOP as shown in Table 1. After the mapping table is created, the

controller continuously monitors the platform T_{dd} and choose CVCOP for the platform based on the already derived table.

Table 1: Platform Throughput-CVCOP Mapping

CVCOP	1.2V+4C	1.1V+4C	1.0V+4C	0.9V+4C	0.8V+4C
T_{dd} (ns)	3	4	5	6	7
CVCOP	1.0V+3C	0.9V+3C	0.8V+2C		
T_{dd} (ns)	8	9	10		
CVCOP	1.0V+2C	0.9V+2C	0.8V+2C		
T_{dd} (ns)	11	12	13		
CVCOP	1.0V+1C	0.9V+1C	0.8V+1C		
T_{dd} (ns)	14	15	16		

3.1.2 Throughput Detection and Core-Voltage Core-on Pair Selection

Detection of platform throughput is critical to the correct functioning of the controller since controller relies on the throughput detected to choose the corresponding CVCOP in the mapping. Pipeline Fullness Detector (PFD) proposed in [14] is utilized to detect the platform throughput. Its working principle is based on the understanding that *Ko* signal at platform inputs represents the data entering the pipeline, whereas *Ki* signal at platform outputs represents the data leaving the pipeline. By subtracting the number of *Ko*'s rising edge from the number of *Ki*'s rising edge, the amount of data within the pipeline during the platform latency time can be calculated and the throughput of the platform can be quantified.

Throughput of the platform is constantly changing during real-world operation. When the input data rate is low, the platform does not need high throughput to process data; instead, throughput needs to be lowered to save energy, implying that the core supply voltage can be lowered and the number of cores processing data can be reduced. On the contrary, when input data rate is high, the platform throughput will need to be increased, which requires higher core supply voltage and increased number of processing cores. Increased throughput means the time

data enter the platform till the time data exit the platform is short, which can also be defined as short T_{dd}. Controller implements the throughput and CVCOP mapping shown in Table 1 as a lookup table (LUT), and detects real-time platform throughput through the PFD. The detected throughput is compared with the throughput in the LUT by a comparator, the comparison results allow the controller to know the core voltage and number of cores it needs to provide to the platform to maintain similar throughput but at lower energy consumption.

3.2 Core Supply Voltage Scaling and Core Disabling

Once the controller knows the core voltage and number of cores it needs to provide to the platform, it will initialize different sequences to either adjust core supply voltage or disable cores. To only adjust supply voltage, the controller waits for the current 4 data to exit the platform and platform *Ko* becomes *rfd*. Once *Ko rfd* is detected, the controller instructs the reference voltage generator to generate a reference value for the voltage regulator to produce the desired supply voltage to all 4 parallel cores

3.2.1 Core Disabling and Enabling Sequence

Sequences are developed to enable the controller to enable or disable cores. Initially, all 4 parallel cores are active, and the first 4 data are sent to each core for processing. The processed data will then go through the multiplexer and appeared at the outputs in the original order. Once T_{dd} observed by the Pipeline Fullness Detector has moved to one of the 14 CVCOP described in Table 1, the controller will initialize the core disabling sequence as shown in Figure 10. Core disabling starts in the order of Core1, Core2, and Core3. To disable Core1, controller waits for the current 4 data to exit the platform and platform *Ko* becomes *rfd*. Once platform *Ko* becomes *rfd*, the controller first enables the isolation cells at Core1's inputs and outputs and then switches off the power switches of Core1. At the same time, the controller instructs the Input Sequence

Generator to only dispatch 3 data to Core2, Core3, and Core4. Output Sequence Generator is also instructed to only take data from Core2, Core3, and Core4.

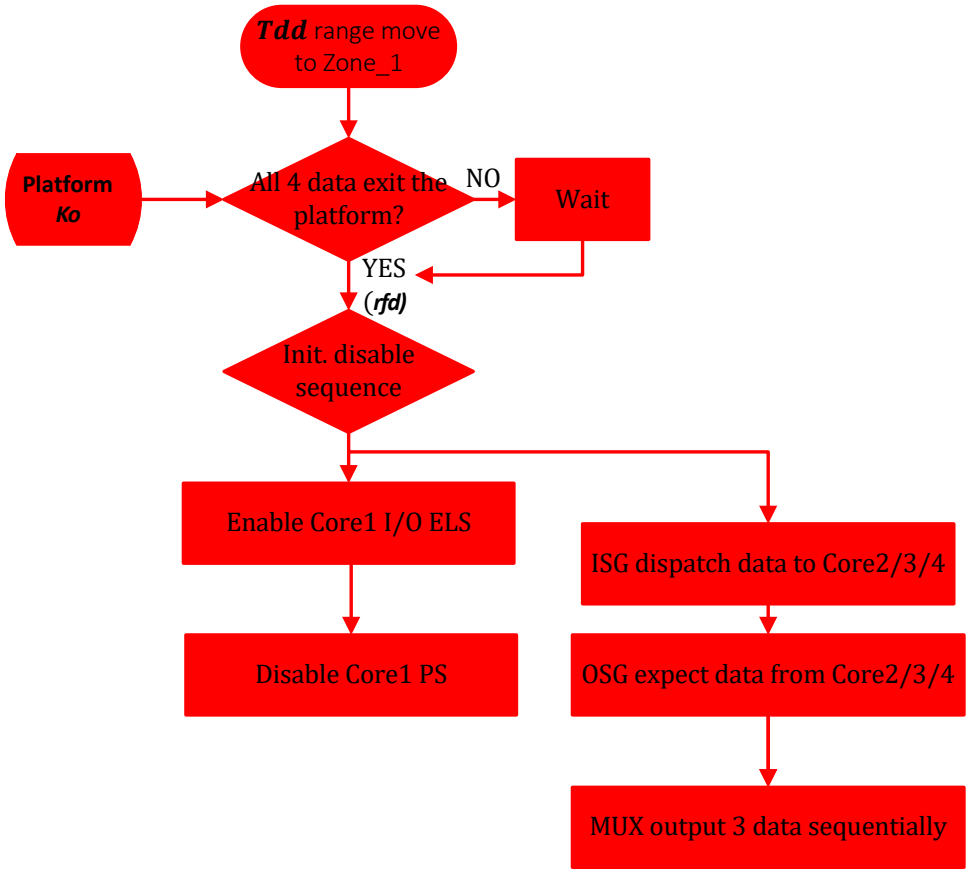


Figure 10: Core-Disabling Sequence

For Core1 enabling sequence as shown in Figure 11, controller waits for the current 3 data to exit the platform and platform *Ko* becomes *rfd*. Once platform *Ko* becomes *rfd*, controller first switches on the power switches of Core1 and sends a reset signal to Core1, which makes all the components in the core pipeline goes into NULL state and all the *Ko* signals become *rfd*. Controller monitors if Core1's *Ko* signals becomes *rfd*; if yes, the controller disables the isolation cells at Core1's inputs and outputs at the same time. Meanwhile, the controller instructs the Input Sequence Generator and Output Sequence Generator to dispatch and receive 4 data from Core1, Core2, Core3, and Core4.

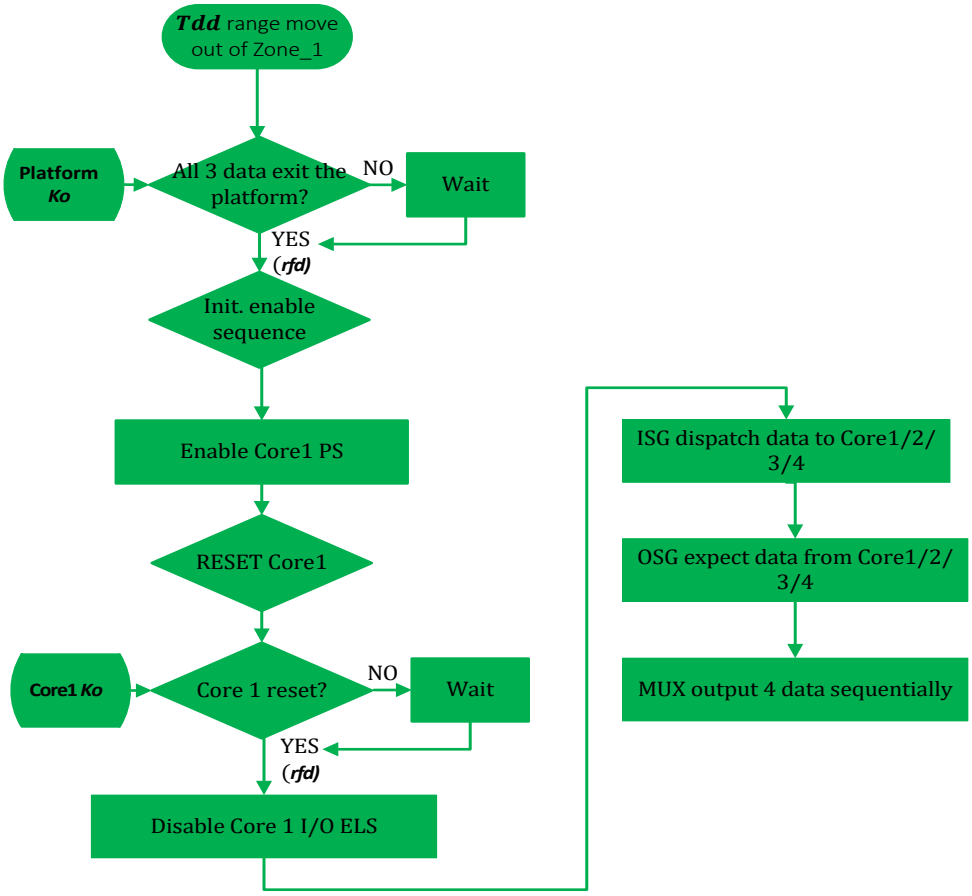


Figure 11: Core-Enabling Sequence

3.2.2 Design of Power Switch, Isolation Cell, and Enable Level Shifter

Power switches, isolation cells, and enable level shifters are important components utilized by the controller when enabling or disabling cores. Power switches are utilized to supply power for cores that can be powered down. High threshold voltage PMOS transistor is used to enable/disable VDD supply to the core, while high threshold voltage NMOS transistor is used to enable/disable VSS supply as shown in Figure 12.

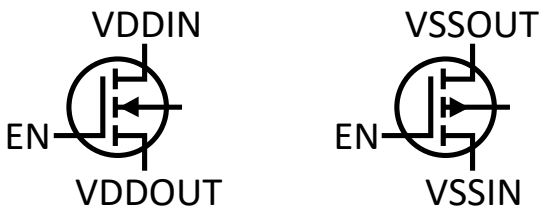


Figure 12: VDD and VSS Power Switch

Isolation cells are utilized where signals enter from peripheral components to a disabled core or leave from a disabled core to peripheral components that are powered on. An isolation cell, as shown in Figure 13, provides a constant logic value to the powered-on block when the core is disabled and has no power, thus preventing unknown or intermediate value that could lead to short circuit current which occurs when both PMOS and NMOS are partially on and there is direct path between VDD and GND. When *EN* signal is high, signal can propagate from A to Y and the isolation cell acts like a buffer; when *EN* signal is low, output Y is held at a constant value through powering the cell by the peripheral component power supply VDDB.

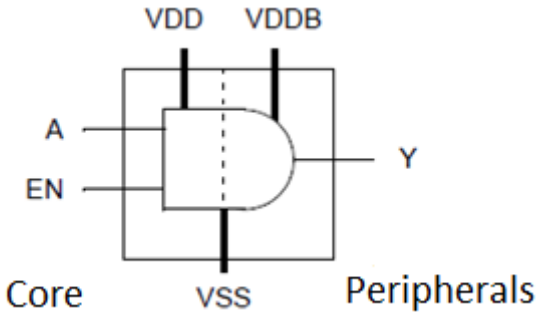


Figure 13: Isolation Cell

A cell with the functionality of both level shifting and isolation is called enable level-shifter (ELS), as shown in Figure 14. ELS is used in between the parallel cores and the peripheral components where the two voltage levels are different and the cores can be powered down. While *EN* signal is high, signal propagates from *IN* to *OUT* and the voltage level is being

shifted. While *EN* signal is low, *OUT* is held at a constant value through peripheral components power supply VDD2.

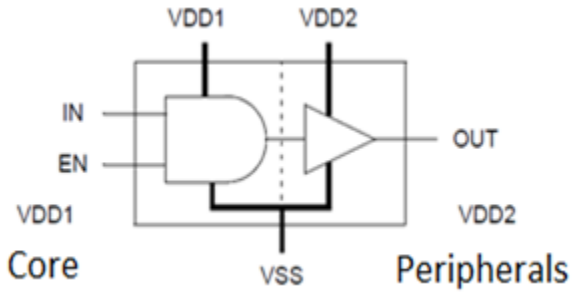


Figure 14: Enable Level-Shifter

4. Homogeneous Platform with Adaptive Core Voltage Scaling and Core-Disabling

A MTNCL homogeneous platform with 4 FIR filter cores [14] is utilized to implement the controller designed in Chapter 3. The controller enables adaptive core-disabling as an addition to the DVS realized in previous research [14]. The homogeneous platform with DVS and core-disabling (CD) is synthesized and placed-and-routed (PnR) in IBM 130nm 8RF-DM technology to acquire three kinds of data: platform layout area, total energy consumption, and performance. The three kinds of data will then be compared with data from previous research to demonstrate the improvements made and the limitation of the platform with DVS and CD.

4.1 Architecture of the Homogeneous Platform with Controller

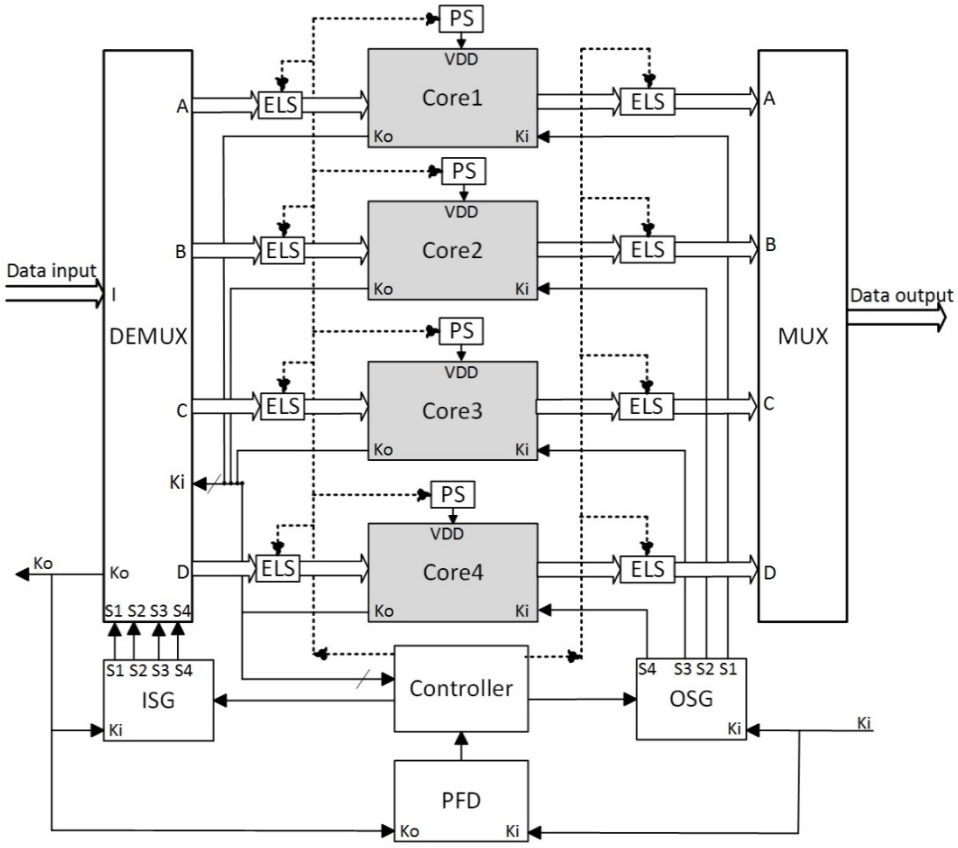


Figure 15: Homogeneous Platform Architecture with Controller

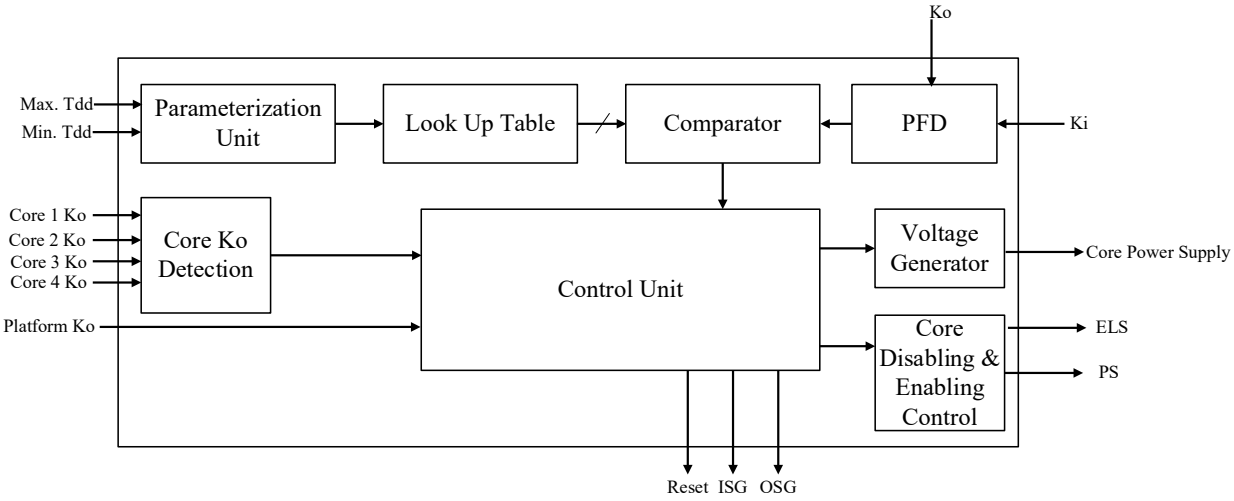


Figure 16: Controller for Homogeneous Platform

Figure 15 presents the architecture of the homogeneous platform with DVS and CD. The

controller is implemented in the platform and serves as the decision-making unit to adjust supply voltage and/or disable cores according to platform throughput as shown in Figure 16. PFD realized in [14] is implemented to continuously provide real-time platform throughput to the controller by subtracting the number of *Ko*'s rising edge from the number of *Ki*'s rising edge and calculates the amount of data within the pipeline during the platform latency time. Once the controller receives the throughput information from the PFD, a comparator will compare it with throughput already populated in the LUT when user configured the controller with max and min core T_{dd}. The comparison results will then yield a CVCOP so the Control Unit knows what voltage it needs to set for supply voltage and how many cores it needs to disable. For the case that supply voltage needs to be lowered and 1 core needs to be disabled, the controller first makes sure the current 4 data have exited the platform by monitor the platform *Ko*, which needs to be *rfd.* The controller will then enable the ELS at Core1's inputs and outputs and then turn off the PS of Core1 through Core Disabling & Enabling Control. Once Core1 ELS and PS are set, the controller instructs the Voltage Generator to generate a value for the voltage regulator to produce the desired supply voltage to Core2, Core3, and Core4. At the same time, the controller instructs the Input Sequence Generator to only dispatch 3 data to Core2, Core3, and Core4, respectively. Output Sequence Generator is also instructed to only take data from Core2, Core3, and Core4. For the case 1 core needs to be enabled, controller again checks if platform *Ko* is *rfd,* implying current 3 data have exited the platform. The controller will then turn on the PS of Core1 and reset Core1, which will make Core1 *Ko rfd.* Once Core *Ko* Detection Unit detects Core1 *Ko rfd*, the Control Unit will disable the ELS at Core1 inputs and outputs. At the same time, the controller instructs the Input Sequence Generator and Output Sequence Generator to dispatch and receive 4 data from Core1, Core2, Core3, and Core4, respectively.

4.2 Simulation of the Homogeneous Platform with Controller

The homogeneous platform with controller is implemented at the transistor-level with the IBM 130nm 8RF-DM technology and simulated in Cadence UltraSim environment. Input Pulse Time (*IPT*) from 15ns to 1ns is utilized to vary platform throughput in different ranges and is defined as the interval between DATA/NULL patterns presenting at the input rails. *IPT* stepping is defined as the time from one *IPT* to the immediate next *IPT*. For example, *IPT* ranging from 15ns to 1ns with 1ns stepping means *IPT* goes from 15ns, 14ns, 13ns, …, to 1ns. A stepping of 5ns means *IPT* goes from 15ns, 10ns, 5ns, to 1ns. Two *IPT* scenarios, down ramp and up ramp, each with a range of different stepping, are simulated for 40 patterns. For down ramp scenario, *IPT* is ranging from 1ns to 15ns, implying that the interval between DATA/NULL patterns get longer and longer, and less and less data arrives at the input ports in a fixed period of time. Nine stepping are applied to the down ramp scenario: 0.1ns, 0.25ns, 0.35ns, 0.5ns, 1ns, 2ns, 3ns, 4ns, and 5ns. Stepping of 0.1ns to 0.5ns implying *IPT* is going down ramp slower, takes longer time to reach 15ns from 1ns. Stepping of 2ns to 5ns means *IPT* is going down ramp faster and takes shorter time to reach 15ns. For up ramp scenario, *IPT* is ranging from 15s to 1ns, implying that the interval between DATA/NULL patterns get shorter and shorter, and more data arrives at the input rails in a fixed period of time. Nine stepping are applied to the up ramp scenario: 0.1ns, 0.25ns, 0.35ns, 0.5ns, 1ns, 2ns, 3ns, 4ns, and 5ns. Stepping of 0.1ns to 0.5ns implies that *IPT* is going up ramp slower and takes longer time to reach 1ns. Stepping of 2ns to 5ns means *IPT* is going up ramp faster and takes shorter time to reach 1ns.

4.3 Homogeneous Platform Energy and Performance Analysis

Layout area, energy consumption, and performance data of homogeneous platform with controller are collected, analyzed, and compared with homogeneous platform with DVS

presented in previous research [14].

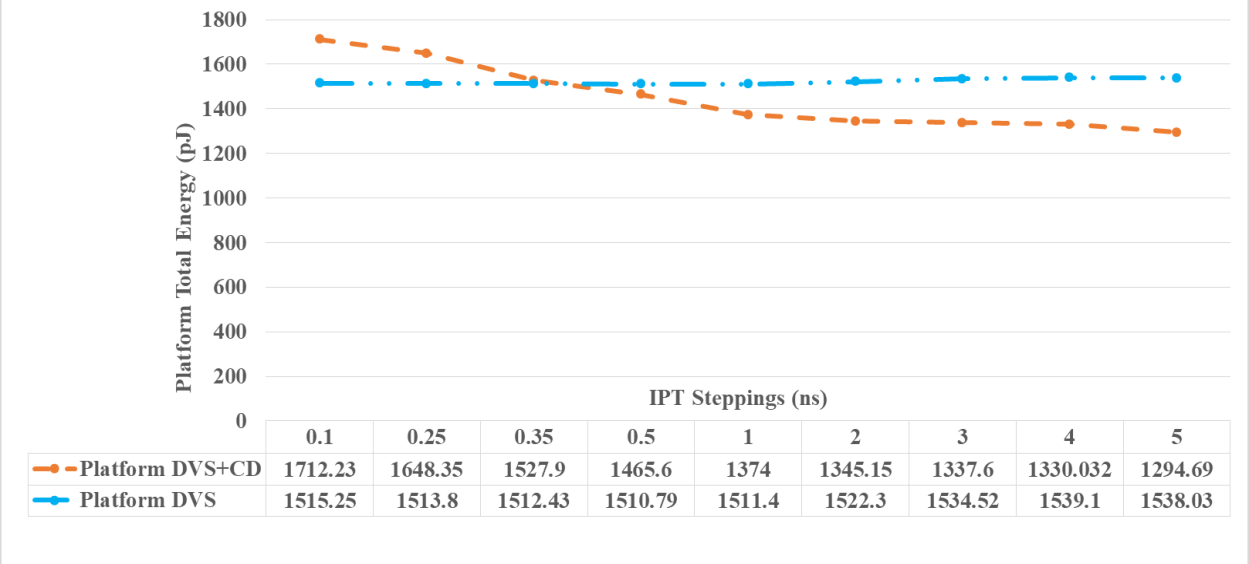
For the platform with DVS proposed in previous research and implemented with IBM 130nm 8RF-DM technology, the layout created with Cadence Encounter with 70% standard cell utilization rate is 0.992mm^2. For the platform with the controller, which is implemented with the same technology and utilization rate, has layout area of 0.95mm^2, 5% smaller compare to platform with only DVS. The addition of controller, PS and ELS network to the platform results in 5% smaller area compare to previous research with only DVS, reason being the Cadence Encounter PnR used for implementing platform with DVS did not include optimization commands to aid in layout area reduction. For PnR of platform with controller, multiple common optimization commands aimed to reduce cell count and wire length are used. Without taking advantage of the layout optimization commands, the area of platform with controller will increase to 1.02mm^2 which is 4% larger than the area of platform with DVS. The area increase is expected since the newly added controller consumes about 6% of total platform energy based on Table 2, implying extra logics and control signals are being added to the platform which results in an increase in area compare to platform with DVS.

The energy consumption of the controller to enable DVS and core-disabling of the homogeneous platform is measured to make sure the controller network's energy overhead is small enough not to overshadow the energy saving introduced. Table 2 presents the energy consumption breakdown of the controller network, FIR filter cores, and the peripheral components such as DEMUX, MUX, ISG, and OSG in the platform for down ramp and up ramp scenarios with *IPT* between 15ns and 1ns with 1ns stepping.

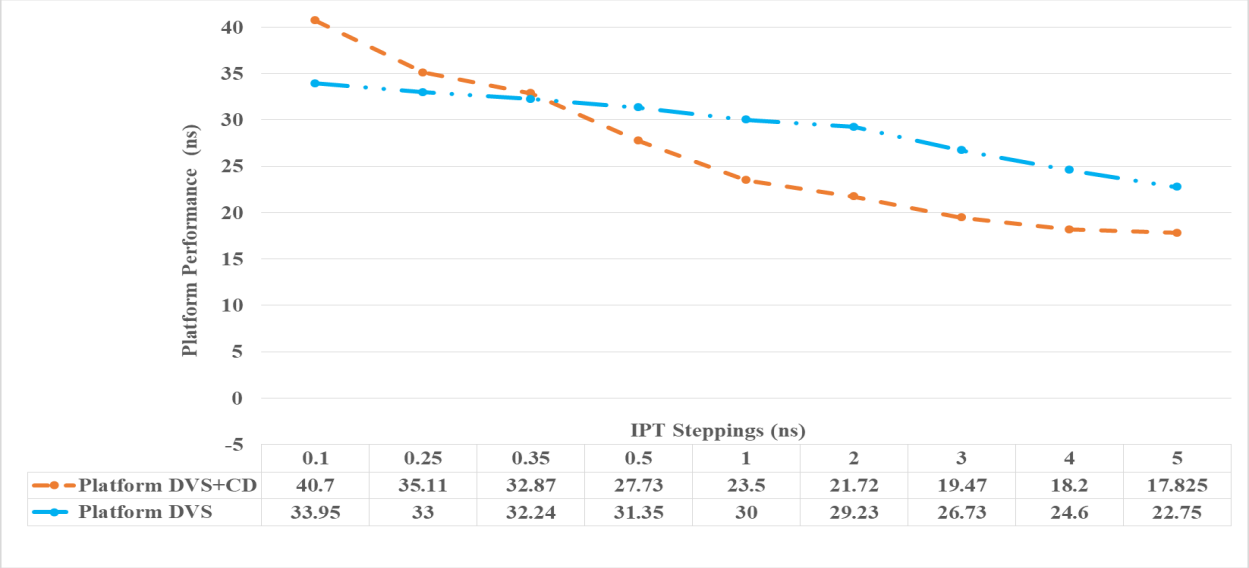
Table 2: Controller Network Energy Breakdown

	Down Ramp (1374 pJ)	Up Ramp (1365.2 pJ)
Controller network	6% (82.44pJ)	6% (81.912pJ)
Platform	4% (54.96pJ)	5% (68.26pJ)
FIR Cores	90% (1236.6pJ)	89% (1215.028pJ)
Look up table registers	27.48pJ	27.304pJ

The platform total energy consumption and platform performance is measured with *IPT* down ramp and up ramp scenarios, each with 9 different stepping and are simulated for 40 data patterns. Down ramp scenario is presented in Figure 17 and up ramp scenario is presented in Figure 18.



(a) Platform Total Energy Consumption



	0.1	0.25	0.35	0.5	1	2	3	4	5
Platform DVS+CD	40.7	35.11	32.87	27.73	23.5	21.72	19.47	18.2	17.825
Platform DVS	33.95	33	32.24	31.35	30	29.23	26.73	24.6	22.75

(b) Platform Performance

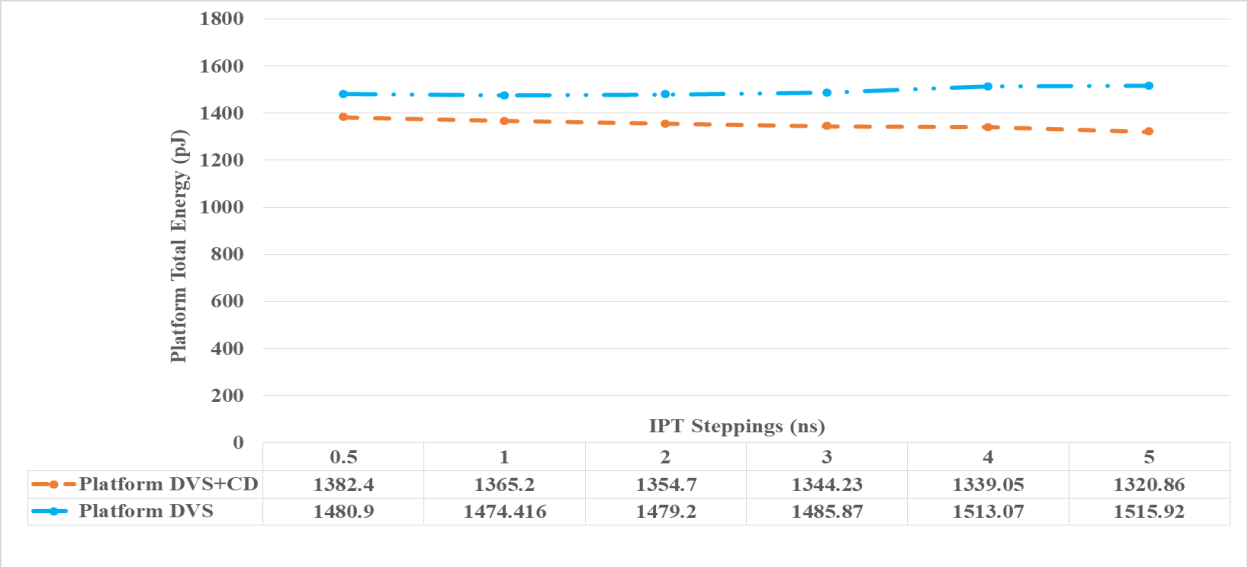
Figure 17: Down Ramp Scenario

Figure 17 (a) and Figure 17 (b) indicates that for down ramp scenario going down slower or faster, the platform with controller (denoted as Platform DVS+CD) has 10% to 15% less energy consumption and 22% to 28% better performance compare to previous research (denoted as Platform DVS) when IPT stepping are from 1ns to 5ns (down ramp going down faster). Once the stepping is between 0.5ns and 0.1ns (down ramp going down slower), energy consumption increased by 2% to 13% and performance is 2% to 17% worse than previous design.

For down ramp going down faster, meaning the *IPT* will increase and reach 15ns in shorter period of time, implying the platform *IPT* change less and less often, signifying less and less platform T_{dd} change. Therefore, with less T_{dd} change, the controller is spending less and less time in comparing platform T_{dd} with LUT T_{dd}, choosing CVCOP, adjusting supply voltage and disabling cores. With less controller activities, and fewer cores being turned off, the energy consumption is reduced and performance is improved. For platform with DVS,

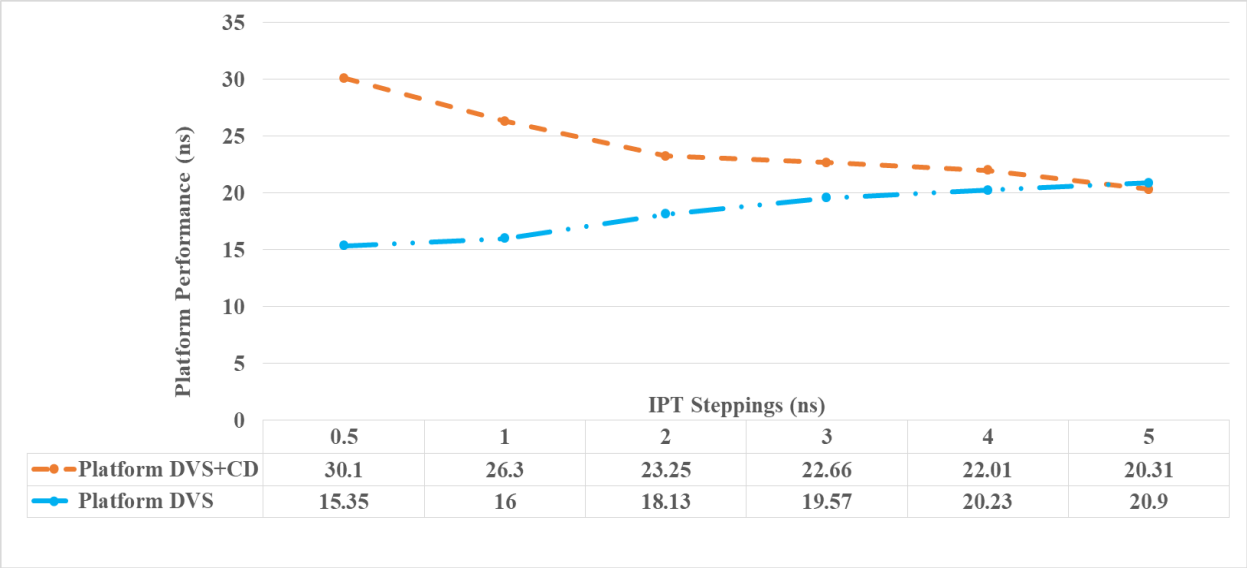
there is no controller in place to disable cores. When down ramp is going down faster, the energy saving by only dropping voltage is outweighed by disabling cores. Furthermore, platform with DVS lowered voltage to 0.7V and 0.6V [14]. The long platform T_{dd} at these voltages contribute negatively to the platform performance.

Same argument applies to down ramp going down slower, meaning the *IPT* will increase and reach 15ns in longer period of time, implying the *IPT* and platform T_{dd} change more and more often. In this case, platform DVS+CD exhibits worse performance and energy consumption which result from the controller keep comparing T_{dd} and choosing CVCOP, and thus spending more time before turning off cores. For platform with DVS, since down ramp is going down slower, the platform reaches supply voltage of 0.7V and 0.6V slower, hence T_{dd} is not greatly affected. The energy consumption becomes better than platform DVS+CD since the controller spend more time making decisions before turning off cores, the longer the time the cores stays on, the higher the energy consumption.



	0.5	1	2	3	4	5
Platform DVS+CD	1382.4	1365.2	1354.7	1344.23	1339.05	1320.86
Platform DVS	1480.9	1474.416	1479.2	1485.87	1513.07	1515.92

(a) Platform Total Energy Consumption



	0.5	1	2	3	4	5
Platform DVS+CD	30.1	26.3	23.25	22.66	22.01	20.31
Platform DVS	15.35	16	18.13	19.57	20.23	20.9

(b) Platform Performance

Figure 18: Up Ramp Scenario

Figure 18 (a) and Figure 18 (b) indicates that for up ramp scenario going up slower/faster, the Platform DVS+CD has 7% to 14% less energy consumption compares to Platform DVS when IPT stepping are between 0.5ns and 5ns. In terms of performance, once the stepping is between 0.5ns and 4ns, the Platform DVS+VD performance become 16% to 48% worse than previous design whereas stepping between 4ns and 5ns will produce 3% to 9% better performance.

Comparing Platform DVS+CD with Platform DVS, Platform DVS+CD has better energy consumption for all stepping, reason being all 4 parallel cores will not be on at the same time for the majority of the stepping, since the *IPT* started from 15ns which is very long and the platform will not need all 4 cores to maintain throughput. Platform DVS+CD performance becomes better with up ramp going up faster (stepping between 4ns and 5ns). This is due to the controller is spending less and less time in comparing $T_{dd,}$ choosing CVCOP and enabling cores. Furthermore, since the sequence of enabling cores take more time and energy than disabling cores, the less

controller core–enabling activity the better in terms of performance and energy efficiency. Therefore, Platform DVS+CD performance gradually outperform Platform DVS with up ramp going up faster.

For up ramp going up slower, Platform DVS+CD is outperformed by Platform DVS in terms of performance since enabling cores already takes more time and now controller is spending extra time in making decisions as well. Even though the Platform DVS+CD energy consumption is still less than Platform DVS, the extra time spent by the controller contribute negatively to the performance.

In summary, for both down ramp and up ramp scenarios, the platform with controller will have different *IPT* stepping that yield better energy efficiency and performance compare to platform with DVS. For down ramp scenario, going down faster with *IPT* stepping between 1ns and 5ns produce better energy efficiency and performance. For up ramp scenario, going up faster with *IPT* stepping between 4ns and 5ns also yields improved energy efficiency and performance. In general, for platform with controller to have advantage over platform with DVS, less *IPT* change of both down ramp and up ramp scenario is preferred.

5 Heterogeneous Platform with Adaptive Core Voltage Scaling and Core-Disabling

The controller designed in Chapter 3 is implemented in a MTNCL heterogeneous platform with 4 different cores: 8-tap Fully Pipelined FIR Filter (FP FIR), 8-tap Non-Pipelined FIR Filter (NP FIR), 8-bit Multiplier (MULT), and 16-bit Adder (ADD) as the one in [14]. The heterogeneous platform with DVS and CD is synthesized and placed-and-routed in IBM 130nm 8RF-DM technology so that the platform layout area, total energy consumption, and performance can be measured and compared with previous research to evaluate the energy efficiency advancement contributed by the controller.

5.1 Architecture of the Heterogeneous Platform with Controller

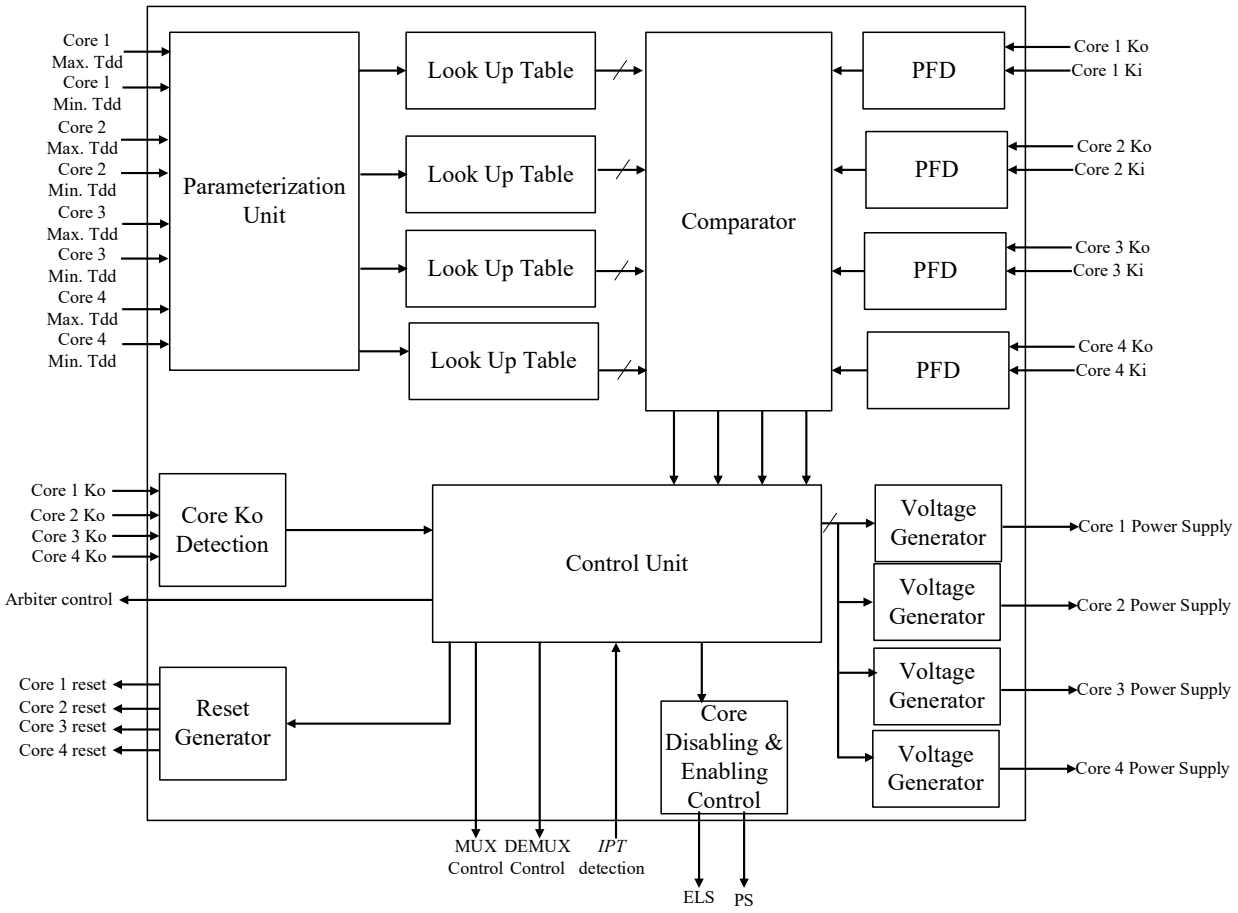


Figure 19: Controller for Heterogeneous Platform

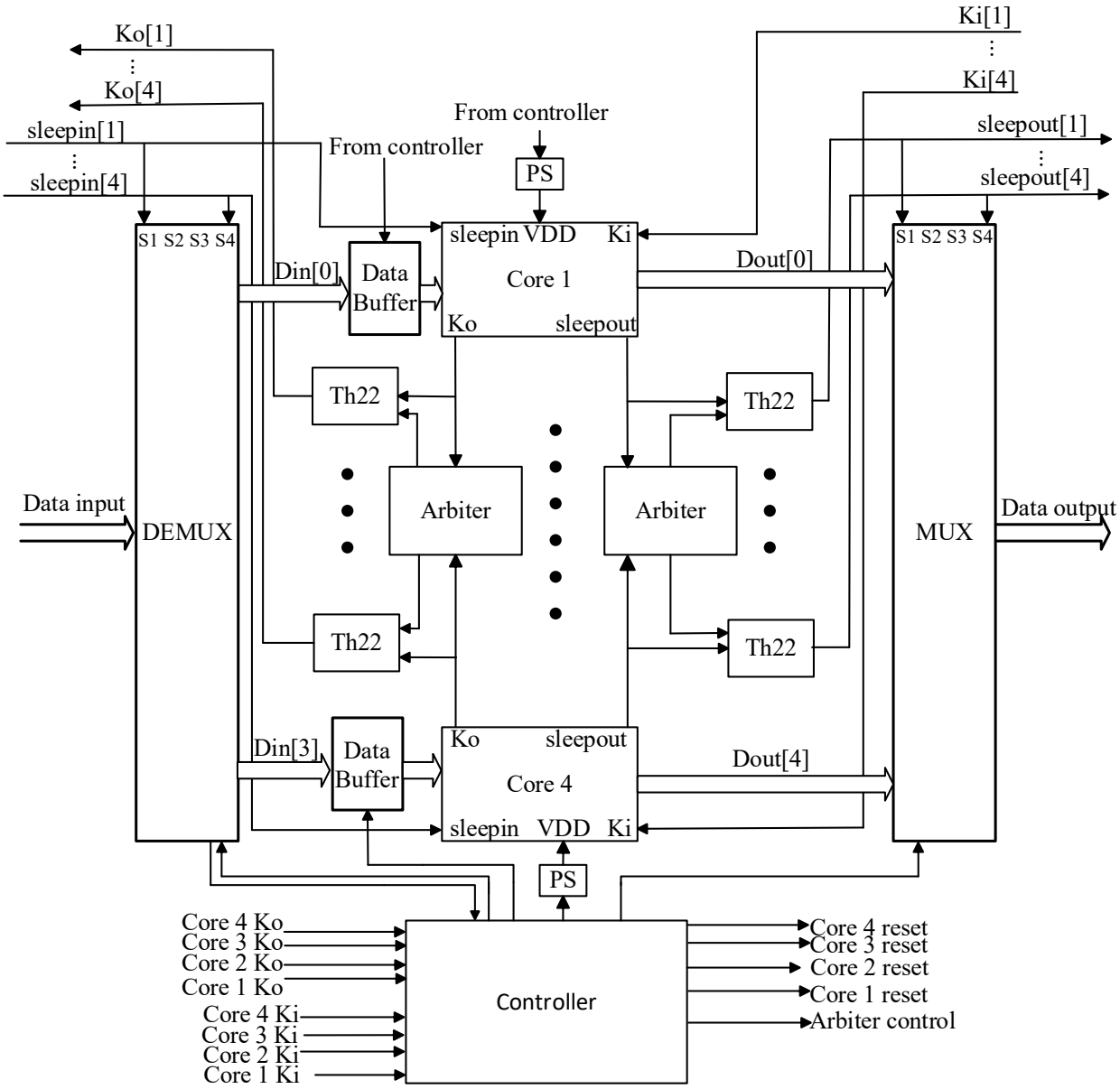


Figure 20: Heterogeneous Platform Architecture with Controller

Heterogeneous platform incorporates 4 cores with different functionalities is to extend the capabilities of the platform. Homogeneous platform incorporating 4 exactly same cores only has one kind of functionality. For example, platform with 4 FIR filter cores can only perform FIR filter specific calculation. If different cores are incorporated, such as multiplier and adder, the heterogeneous platform will have extended capability to perform different calculations.

Furthermore, the controller architecture design should be capable of balancing energy and performance of not only homogeneous platform. With some modifications, it should be able to balance energy and performance for the heterogeneous platform with extended capabilities compared to homogeneous platform.

The controller needs to be redesigned to enable DVS and core-disabling for individual cores in the platform. Since each core has different functionality, the throughput will be different, meaning that the controller is required to handle 4 different throughputs simultaneously during real-time operation and needs to adjust each core's supply voltage and perform core-disabling individually.

Figure 19 shows the architecture of the controller for heterogeneous platform. To adjust core supply voltage, each core has a designated PFD implemented in the controller to detect the change in throughput and provide it to the controller. Controller for homogeneous platform only requires 1 LUT as all the cores incorporated are the same, however, 4 LUTs are required for the heterogeneous platform controller to store the throughput and core voltage mapping for 4 cores of different functionalities. The realization of core disabling and enabling in heterogeneous platform is different compare to that of homogeneous platform. For homogeneous platform, core-disabling is based on the CVCOP chosen by the controller according to platform throughput. However, since each core in the heterogeneous platform is responsible for processing a unique set of data, the core can not be disabled even if its throughput becomes slower. If a core is disabled due to slow throughput, the next set of input data, which needs to be processed by the now-disabled core, will end up not being taken by the core and cause platform deadlock. To prevent this issue and take advantage of the energy saving by disabling cores, the criterion has changed from monitoring platform throughput to the detection of presence of data at

input rails.

Two scenarios are considered when using the presence of data as core-disabling criteria. First scenario is *IPT* being longer than the core's T_{dd} for all 4 cores or only part of the cores. Second scenario is *IPT* being shorter than the core's T_{dd} for all cores or part of the cores. For the first scenario, when *IPT* is longer than core T_{dd}, even a core finished processing data and the arbiter has granted it access to the common data bus, the next set of data may not be immediately available due to the long *IPT.* To take advantage of this observation, the controller is designed to check if data is available at platform input right after arbiter grants the core data: if no data is available, controller can disable the core. The problem induced by this methodology is that the controller does not know if the next input data will arrive soon or not, if data arrives when controller is disabling core, the core will not be available to process data which will negatively impact platform performance. To resolve this problem, a data buffer is inserted between each core and the DEMUX. If data arrives when the controller is disabling/enabling cores, the buffer can hold the data and allow the controller extra time to finish core disabling/enabling. The controller also needs to communicate with arbiter to keep track of which core should receive data once the previous set of data is already in the buffer.

The second scenario is when *IPT* is shorter than part of the core's T_{dd}. For example, if 2 cores are now turned off due to core T_{dd} shorter than current *IPT*, the other 2 slower cores will need to run at full speed to process data. The arbiter will need to be configured to only dispatch data to these 2 cores. The buffers for these two cores are used to hold data temporarily before the core supply voltage is adjusted to 1.2V and the arbiter is configured. The controller is designed to monitor if the current *IPT* is shorter than the two core's T_{dd} since each core's T_{dd} information is provided by the designer, and the controller can constantly monitor the period between two

consecutive sets of data send to each core by the DEMUX. If the period is shorter than the already known core T_{dd}, implying the *IPT* is shorter than core T_{dd}, then the data is hold at the buffer immediately by the controller while the core supply voltage is tuned to 1.2V and the data path is adjusted. The buffer for each core is designed to hold 1 set of data. Holding two sets of data in the buffer would allow extra time for the controller to either disable/enable core or configure the arbiter; however, this would increase area overhead, power consumption, and circuit complexity as extra control is needed to dispatch two sets of data. Since the buffer can hold 1 set of data, for the scenario that core T_{dd} is longer than *IPT*, if the time needed to configure arbiter and adjust core voltage is longer than 1 *IPT*, the buffer will not be able to hold the next data set. Therefore, the controller is designed that the time needed to configure arbiter and tune core voltage is lower than the shortest *IPT*. The DEMUX is redesigned to be controllable by the controller to stop sending input data to the buffer if the buffer is still full and the arbiter is still being configured when next data arrives.

Figure 20 presents the architecture of the heterogeneous platform with modified controller enabling DVS and CD. Four PFDs are implemented in the controller to provide real-time core throughput to the controller. Throughput from the PFD of each of the four cores will be compared with the throughput in the LUT of the corresponding core. 4 LUTs are populated when user configured the controller with max and min T_{dd} of all 4 cores as shown in Figure 19. The comparison result will produce a core voltage so the control unit knows the supply voltage to set for the individual core. For the heterogeneous platform, adjusting core voltage and disabling core are mutually exclusive. The reason is that if the core voltage needs adjustment, there are data available for the core to process and thus the core can not be disabled.

For the case that MULT and ADD need to be disabled due to core T_{dd} shorter than current

IPT and FP FIR and NP FIR supply voltages need to be increased, the controller first checks if the *Ko* of both MUTL and ADD are *rfd.* If yes, the controller will first activate the ELS at the inputs and outputs of MULT and ADD, and then turn off the PS of MULT and ADD through Core Disabling & Enabling Control. During the disabling sequence, the controller continuously monitors the platform data input to see if any new input data arrives. If a new input data arrives before the disabling sequence is finished, the controller will first hold the input data at the MULT and ADD buffer and then turn on MULT and ADD PS. The controller will then send a reset signal to MULT and ADD. Once MUTL and ADD *Ko* signals are *rfd*, the controller will deactivate the ELS. As soon as the ELS is deactivated, the arbiter is able to detect the MULT and ADD *Ko rfd,* and the input data being held in the buffer will then be dispatched by the arbiter to MULT and ADD. On the other hand, if no new data arrives at platform input, the controller will finish the disabling sequence of MULT and ADD. While MULT and ADD are being disabled, the controller already knows the *IPT* is shorter than FP FIR and NP FIR T_{dd}. The controller will first hold the input data at FP FIR and NP FIR buffer, and instruct the voltage generator to generate 1.2V for both FIRs. The controller will then configure the arbiter to only dispatch data to the two FIRs, given there are no new input data arrive for MULT and ADD. Once the 1.2V voltage is generated and the arbiter is configured, the controller will enable the data path between the buffers and the FIRs so the arbiter can dispatch data to them.

5.2 Simulation of the Heterogeneous Platform with Controller

The heterogeneous platform with controller is implemented at the transistor-level with IBM 130nm 8RF-DM technology and simulated in Cadence UltraSim environment. Input Pulse Time (*IPT*) from 30ns to 5ns is utilized to vary platform throughput in different ranges. The 4 cores tested: FP FIR, NP FIR, MULT, and ADD have minimum T_{dd} ranging from 11ns to 27ns. This

simulation setup ensures that *IPT* covers the following cases: *IPT* shorter than 4 core's T_{dd}, *IPT* shorter than part of the core's T_{dd}, *IPT* longer than part of the core's T_{dd}, and *IPT* longer than all core's T_{dd}. Two *IPT* scenarios, down ramp and up ramp, each with a range of different stepping, are simulated for 20 patterns. For down ramp scenario, *IPT* is ranging from 5ns to 30ns, implying that the interval between DATA/NULL patterns get longer and longer, and less and less data arrive at the input rails in a fixed period of time. Five stepping are applied to the down ramp scenario: 1ns, 1.5ns, 2.5ns, 5ns, and 10ns. For up ramp scenario, *IPT* is ranging from 30s to 5ns, implying that the interval between DATA/NULL patterns get shorter and shorter, and more data arrive at the input rails in a fixed period of time. Five stepping are applied to the up ramp scenario: 1ns, 1.5ns, 2.5ns, 5ns, and 10ns.

5.3 Heterogeneous Platform Energy and Performance Analysis

Layout area, total energy consumption, and performance data of the heterogeneous platform with controller are analyzed, and compared with the same heterogeneous platform with DVS presented in previous research [14]. For the platform with DVS which is also implemented with IBM 130nm 8RF-DM technology, the layout created with Cadence Encounter with 70% standard cell utilization rate is 1.158mm^2. For the platform with the controller, which is realized with the same technology and utilization rate, has layout area of 1.286mm^2. Platform with controller is 11% larger compare to platform with DVS. The larger area is expected since 4 data buffers are added to the platform and 3 LUTs are added to the controller. Extra control logics and signals are added to the controller as well since the DVS and core-disabling enablement of the heterogeneous platform are more complex to account for processing cores with different capabilities.

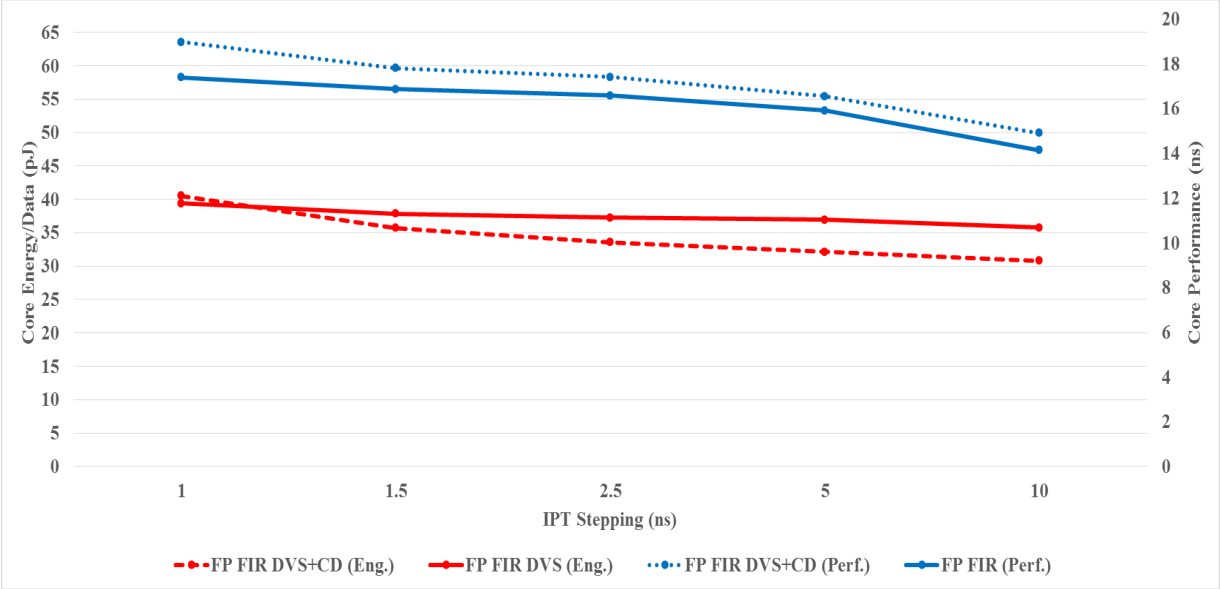
The energy consumption per data of the controller network and data buffers for enabling

DVS and core-disabling of the heterogeneous platform is measured to confirm the energy overhead is small enough to not outweigh the energy saving introduced. Table 3 presents the energy consumption breakdown of the controller network, data buffers, NP FIR, FP FIR, MULT, ADD, and the peripheral components such as DEMUX, MUX, and arbiter in the platform for random scenario with *IPT* randomly varies between 30ns and 5ns for 20 data patterns.

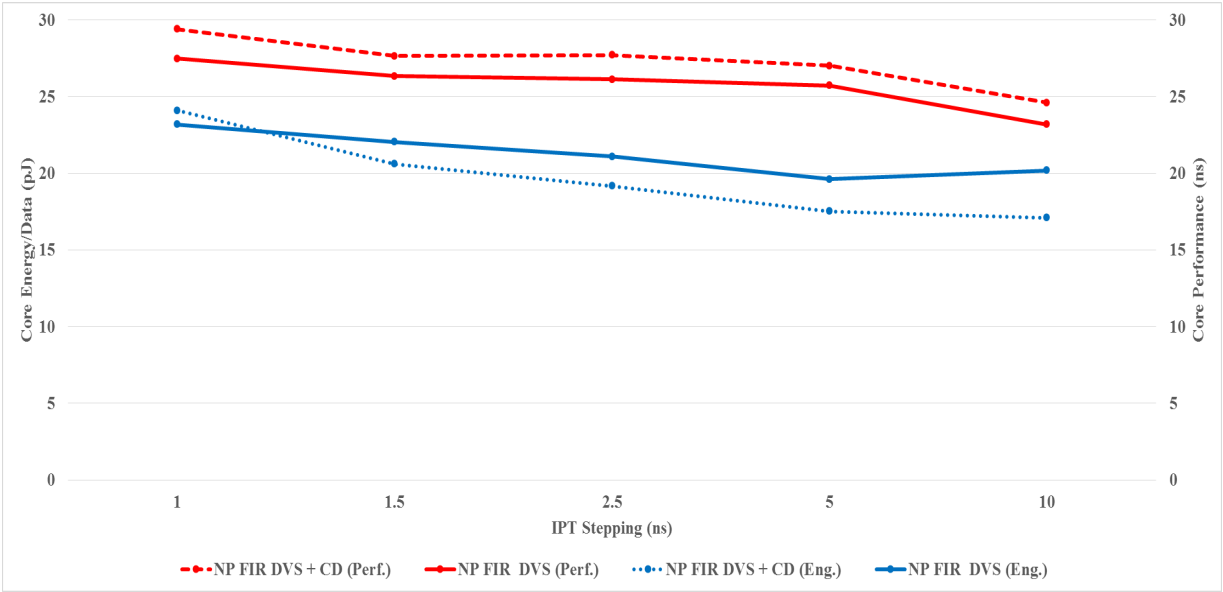
Table 3 Energy Consumption Breakdown

Controller Network	10% (18.867pJ)
Platform	5% (9.43pJ)
FP FIR	38% (71.7pJ)
NP FIR	28% (52.8pJ)
MULT	11% (20.75pJ)
ADD	6% (11.32pJ)
Data Buffers	2% (3.8pJ)

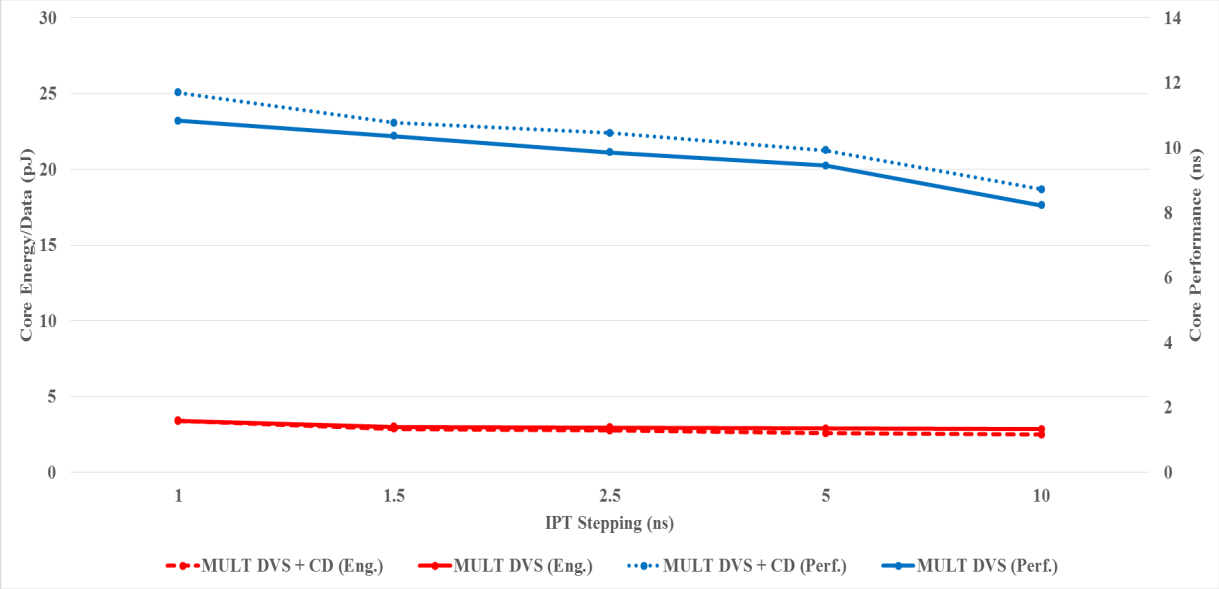
The energy consumption per data and performance of FP FIR, NP FIR, MULT, and ADD are measured with *IPT* down ramp and up ramp scenarios, each with 5 different stepping, and are simulated for 20 data patterns. Down ramp scenario is presented in Figure 21 and up ramp scenario is presented in Figure 22.



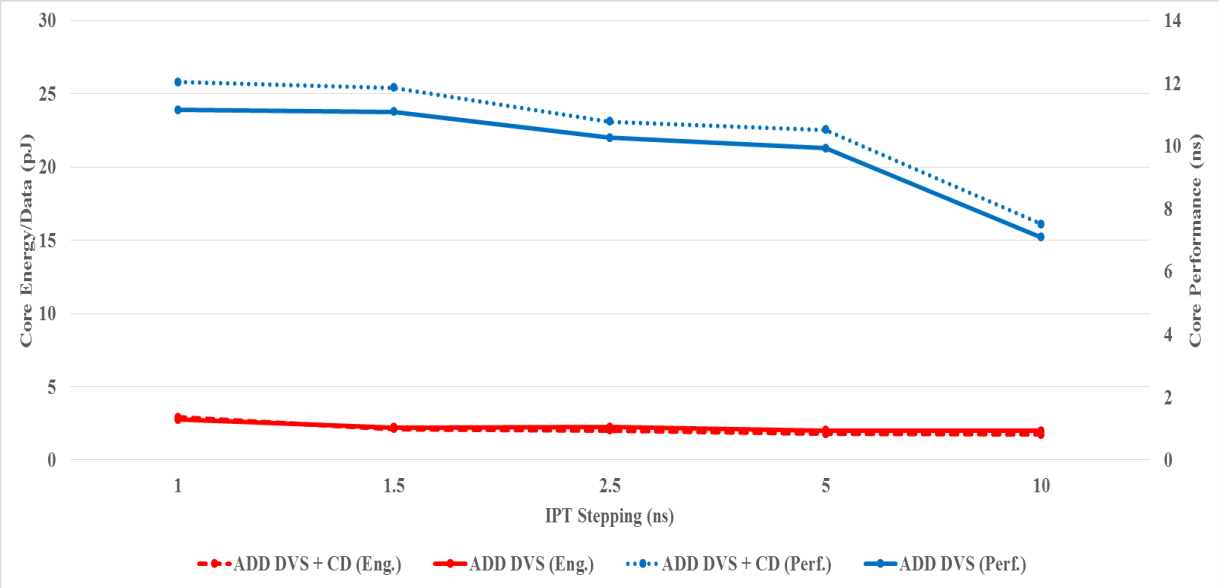
(a) FP FIR



(b) NP FIR



(c) MULT



(d) ADD

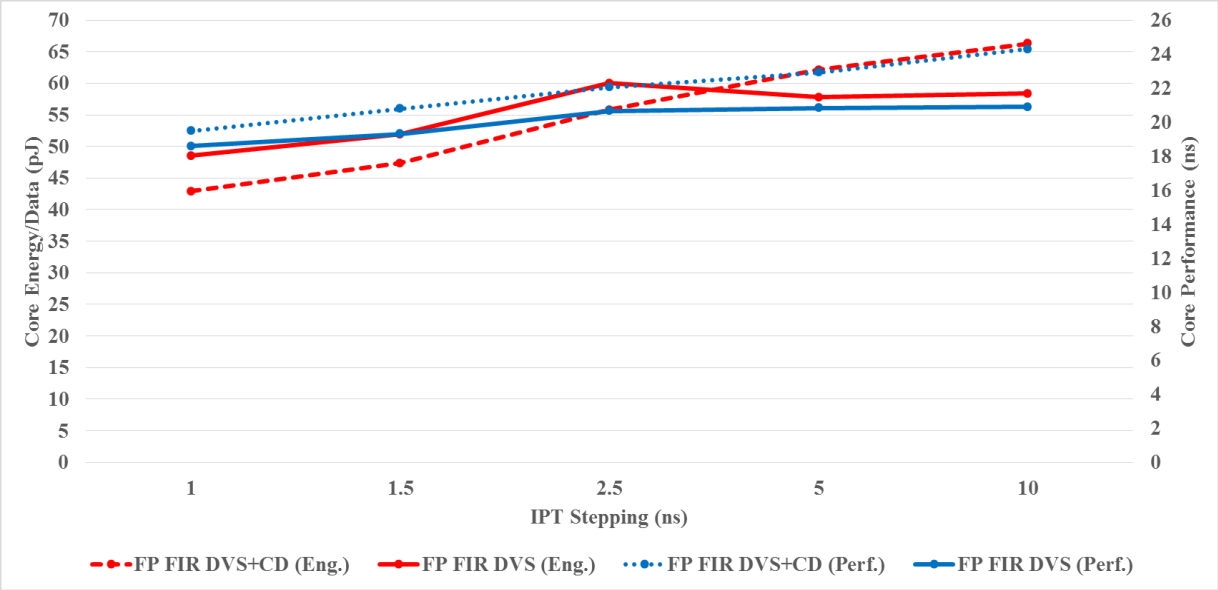
Figure 21: Down Ramp Scenario

Figure 21 (a), (b), (c), and (d) indicate that for down ramp scenario, when *IPT* stepping is ≥2.5ns and ≤10ns, all cores show improved energy efficiency with slightly degraded performance. FP FIR and NP FIR in Figure 21 (a) and (b) show more energy saving compare to MULT and ADD in Figure 21 (c) and (d) since FP FIR and NP FIR are complex designs

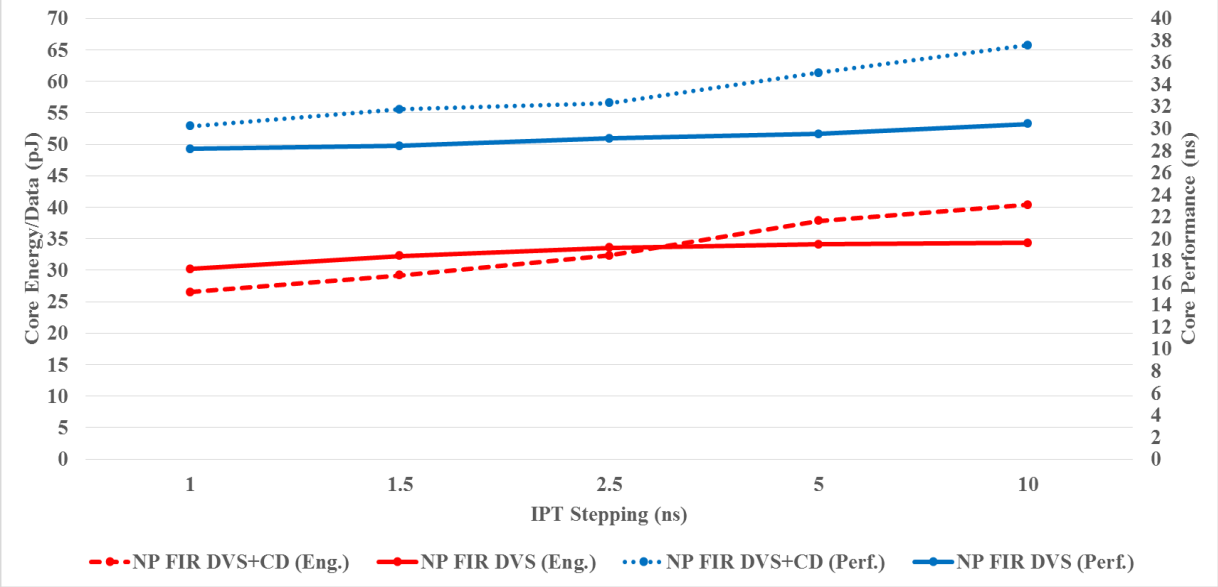
compare to MULT and ADD which consumes more energy when active and will have more energy saving when disabled, implying the controller is suitable for improving energy efficiency of complex designs. For *IPT* stepping ≥2.5ns and ≤10ns, the energy consumption per data of each core is 10% to 18% less compare to platform with DVS (denoted as core type DVS, for example, ADD DVS), while each core's performance degraded 5% to 6%. The reason is that for down ramp scenario with *IPT* from 5ns to 30ns, once *IPT* is longer than the T_{dd} of ADD and MULT, the controller is starting to have time to turn off ADD or MULT cores. ADD and MULT have shorter T_{dd} compared to FP FIR and NP FIR, implying they can be disabled first when *IPT* becomes longer. With *IPT* becomes longer and longer, it will eventually be longer than FP FIR's T_{dd} and NP FIR's T_{dd}, enabling these two cores to be turned off. The longer the *IPT*, the more time is allowed for the controller to turn off cores and for the cores to stay off, implying improved energy consumption. Furthermore, for the case that down ramp *IPT* is going down faster, the platform will receive more input data with long *IPT* compare to IPT going down slower. More input data with long *IPT* also contributes to energy saving since more time between data is allocated to turn off cores and buffers are less utilized to temporarily store data. T_{dd} of each core increased slightly due to the added data buffers and extra control signals. In general, the longer the *IPT,* the more time is available for the controller to disable cores. The 10ns *IPT* stepping is the longest stepping simulated for 20 patterns while avoiding extra-long simulation time.

For down ramp scenario with *IPT* stepping ≤1ns, each core consumes 3% to 5% more energy and has 4% to 9% degrade in performance compare to platform with DVS. The reason is that since the platform has more input data with *IPT* faster than all 4 core's T_{dd}, less time is allocated for the controller to turn off cores to save energy and the 4 buffers are constantly active

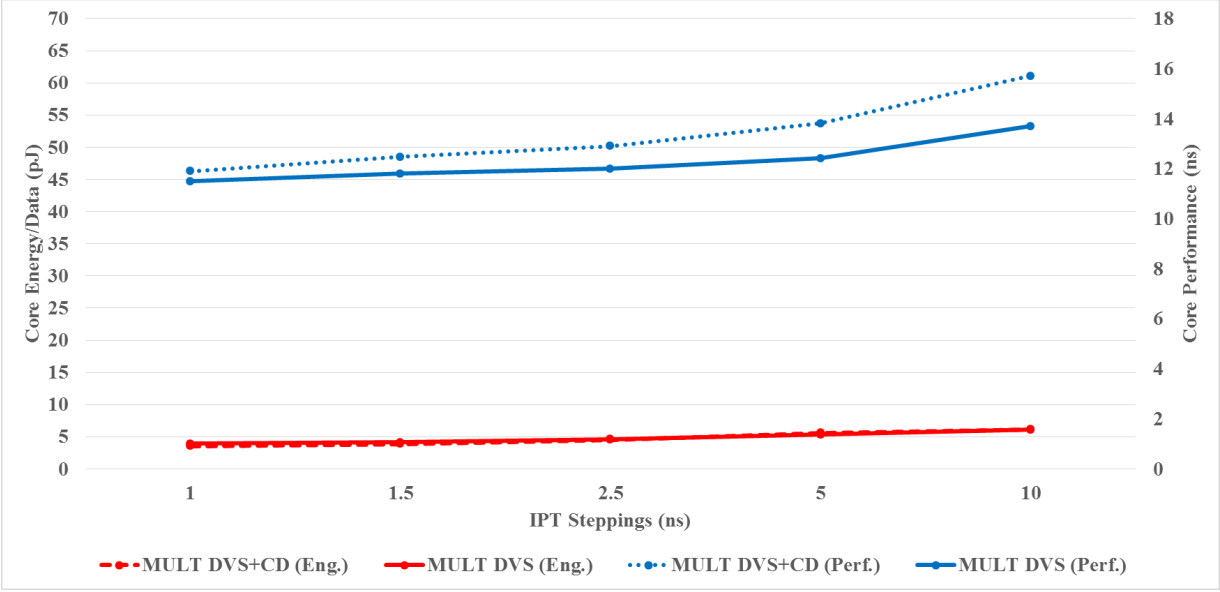
to store data. The energy saving contributed by the data input that allow controller to turn off cores is less than the energy consumption induced by data input that requires all 4 cores and buffers to be active all the time, resulting in diminished energy efficiency.



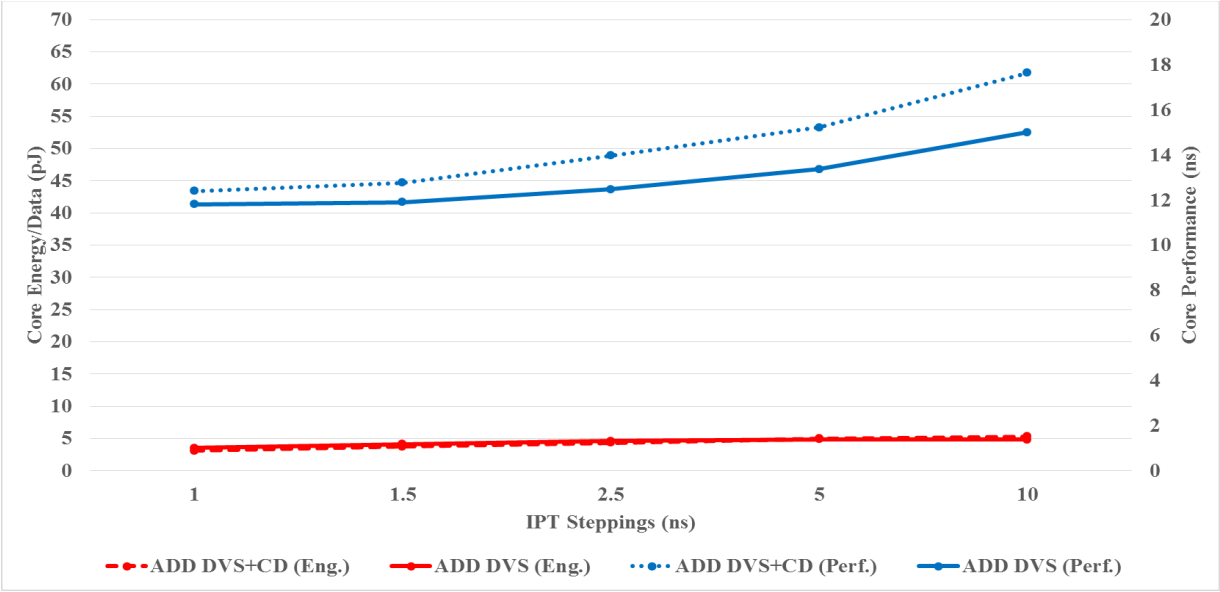
(a) FP FIR



(b) NP FIR



(c) MULT



(d) ADD

Figure 22: Up Ramp Scenario

Figure 22 (a), (b), (c), and (d) indicate that for up ramp scenario, the *IPT* stepping exhibits improved energy efficiency is ≤2.5ns. FP FIR and NP FIR in Figure 22 (a) and (b) show more energy saving compare to MULT and ADD in Figure 22 (c) and (d), suggesting the controller is

suitable for improving energy efficiency of complex designs. The energy consumption per data of each core is 5% to 14% less compare to previous research, while each core's performance degraded by 3% to 10%. The reason is that for up ramp scenario with *IPT* from 30ns to 5ns, since *IPT* is longer than all 4 cores at the beginning, the controller has more time to turn off all cores and keep all cores off. With *IPT* getting shorter, FP FIR and NP FIR will start to have less time to stay off with *IPT* approaching their T_{dd}. FP FIR and NP FIR will eventually have to always stay on when *IPT* becomes shorter than their T_{dd}. ADD and MULT will have more time to stay off compare to FP FIR and NP FIR since their T_{dd} are shorter in comparison. However, with *IPT* keep getting shorter, ADD and MULT will start to have less time to be disabled and eventually must stay always on when *IPT* is shorter than their T_{dd}. Similar as down ramp scenario, the longer the *IPT*, the more time is allowed for controller to disable cores and for the cores to stay off. For the case that up ramp *IPT* is going up faster, the platform will receive more input data with short *IPT* compare to when *IPT* going up slower. More input data with short *IPT* contribute negatively to energy saving since less time between data is allocated to turn off cores and buffers are utilized increasingly to temporarily store data. T_{dd} increased more compared to down ramp scenario since extra time is needed to enable cores compare to disable cores.

Up ramp scenario with *IPT* stepping ≥5ns has degraded energy efficiency and performance. Each core consumes 6% to 15% more energy and has 10% to 19% performance degrade compare to platform with DVS. The reason is that since the platform has more input data with *IPT* shorter than all 4 core's T_{dd}, less time is allocated for controller to disable cores to save energy and the 4 buffers are constantly active to store data. Furthermore, the controller has to carry out increased number of core-enabling sequence which consumes more energy compared to down ramp scenario which controller is carrying out increased number of core-disabling

sequence. With up ramp scenario going up faster, the energy saving contributed by the data input that allow controller to disable cores is being offset by the energy consumption of data input that requires all 4 cores and buffers to be active, hence degrading energy efficiency.

In summary, for down ramp scenario, the platform with controller consumes 10% ~to 18% less energy with 5% to 6% degrade in each core's performance compare to the platform with DVS if *IPT* stepping is ≥2.5ns. When *IPT* is ≤1ns, the platform with controller consumes 3% to 5% more energy with 4% to 9% degrades in each core's performance. For up ramp scenario, the platform with controller consumes 5% to 14% less energy with 3% to 10% degrades in each core's performance compare to the platform with DVS if *IPT* stepping is ≤2.5ns. When *IPT* is ≥5ns, the platform with controller consumes 6% to 15% more energy and has 10% to 19% degrades in core performance. Platform with controller has advantage over platform with DVS in energy consumption and minimal performance impact for two scenarios:

(a) *IPT* is down ramping and the stepping is ≥2.5ns.

(b) *IPT* is up ramping and the stepping is ≤2.5ns.

IPT is defined as a time delay between DATA/NULL patterns, which can be interpreted as input data rate. The smaller the delay the more data will arrive within a fixed time frame and vice versa. *IPT* down ramping and up ramping can be interpreted as constant changing data rate. In real-world application, constantly changing data rate is common in mobile communication, where data rate is slower when streaming a standard definition video to a mobile device from a base transceiver station compared to higher data rate when streaming high definition video [28].

6 Core-Disabling Scheme Application Strategy

6.1 Energy and Performance Comparison of Core-Disabling Scheme and Dynamic Voltage Scaling Scheme

The homogeneous platform and the heterogeneous platform with DVS [14] have been utilized to implement the core-disabling scheme. The energy consumption and performance data collected from both platforms with varying *IPT* scenarios indicate that the core-disabling scheme is advantageous in balancing energy and performance for certain scenarios with limitations in others.

For homogeneous platform, since 4 cores have the same functionality, the T_{dd} of each core is not a factor affecting the implementation of core-disabling scheme. Instead, *IPT* is the dominating factor since it determines the amount of time controller compares the platform throughput with LUT throughput, chooses CVCOP, and disables cores. The controller's decision making frequency has direct impact over energy consumption and platform T_{dd}. When *IPT* stepping is between 1ns to 5ns for down ramp scenario and 4ns to 5ns for up ramp scenario, the platform total energy consumption and performance are better than that of the platform with DVS, suggesting that to implement and benefit from the core-disabling scheme, the *IPT* stepping will need to be within 1ns to 5ns for both scenarios. To further lower energy consumption without negative impact on platform performance, the percentage of input data going down ramp will need to be more than the data going up ramp since down ramp scenario has less energy consumption and better performance across a wider *IPT* stepping range (1ns to 5ns) compare to up ramp scenario (4ns to 5ns).

For heterogeneous platform, implementing core-disabling scheme when *IPT* stepping is ≥2.5ns for down ramp scenario and ≤2.5ns for up ramp scenario improves energy consumption with minimal performance impact. To benefit from the implementation of core-disabling

scheme, the shorter the core T_{dd} compared to *IPT*, the more time the controller will have to disable the core and thus saves more energy. Furthermore, down ramp scenario going down at a stepping ≥2.5ns and up ramp scenario going up at a stepping ≤2.5ns will also contribute positively to energy saving without significantly increase in each core's T_{dd}.

In summary, whether or not to utilize the core-disabling scheme is based on different *IPT* scenarios for the homogeneous platform. For heterogeneous platform, the criteria are based on both *IPT* scenarios and the T_{dd} of different cores incorporated.

6.2 Core Disabling Scheme Implementation Flow

A flow is constructed to determine if the core-disabling scheme is needed when designing a new platform. The decision flow for a homogeneous platform is different from that of a heterogeneous platform since each platform has different criteria to utilize core-disabling scheme.

For homogeneous platforms:

1) Characterize the maximum and minimum T_{dd} of the parallel cores to be implemented.

   The energy and performance balancing controller relies on the max/min T_{dd} of the cores configured by the user to create LUT.

2) Determine the CVCOP that provides best coherency between platform energy and platform T_{dd}.

   For the platform with *X* cores and *Y* different supply voltage levels for adjustment, there will be totally $X*Y = Z$ CVCOP available. Apply each CVCOP to the platform with the parallel cores and plot the platform average T_{dd} and platform total energy trend across all CVCOP. After plotting, identify the *N* amount of CVCOP that has similar throughput but higher energy consumption than other CVCOP, remove the identified *N* CVCOP from *Z*.

Platform energy and platform T_{dd} should now be inversely proportional with totally $Z - N = M$ CVCOP.

3) T_{dd} range (TR) derived from max/min T_{dd} is mapped to the M CVCOP.

   Max T_{dd} - Min $T_{dd} = TR$.

   Max T_{dd} is defined as the longest time the platform needs to process data assuming the data from the previous stage is immediately available when requested. *TR* will be parameterized and mapped to M CVCOP. The controller divides the *TR* by *M*, creates *P* amount of platform T_{dd}, and maps each T_{dd} from *P* to one of the *M* CVCOP. *P* represents the resolution of the platform throughput to the controller. The larger the *M* is, the finer *P* becomes, and throughput will be more closely mapped to CVCOP, implying better coherency between platform energy and platform T_{dd}. The mapping between *P* and *M* populates the LUT and provides a reference for the controller to adjust supply voltage and disable cores based on platform throughput.

4) Determine the *IPT* stepping of input data the platform will process.

   For homogeneous platform controller, the best effectiveness in reducing energy consumption and improving performance is when *IPT* is having less change of both down ramp and up ramp scenario. Assuming the *IPT* range for down ramp scenario is from D_{min} to D_{max}, and the stepping is *S*. The larger the *S* is, the faster D_{min} will reach D_{max}, implying less platform throughput variation induced by *IPT* change and less controller activities. Furthermore, if the percentage of input data going down ramp is more than the data going up ramp, energy consumption can be further lowered without negative impact on platform performance.

For heterogeneous platforms:

1) Characterize the maximum and minimum T_{dd} of each core to be implemented.

   The energy and performance balancing controller relies on the max/min T_{dd} of each core configured by the user to populate 4 LUTs.

2) Determine the core voltage (CV) that provides best coherency between platform energy and platform T_{dd}.

   For the platform with X cores and Y different supply voltage levels for adjustment, there will be totally $X*Y = Z$ CV available. Apply each CV to the platform with the parallel cores and plot each core's T_{dd} and energy/data trend across all CV. After plotting, identify the N amount of CV that has similar throughput but higher energy consumption than other CV, remove the identified N CV from Z. Core energy and T_{dd} should now be inversely proportional with totally $Z - N = M$ CV.

3) T_{dd} range (TR) derived from max/min T_{dd} is mapped to the M CV.

4) Determine the *IPT* range of input data

   For heterogeneous platform controller, the shorter each core's T_{dd} compare to *IPT*, the more time the controller will have to disable the core which improves energy saving.

7 Conclusion

This dissertation work focuses on the MTNCL parallel computing platforms which incorporate homogeneous and heterogeneous cores with Dynamic Voltage Scaling (DVS). Core-disabling is introduced as an advancement of DVS for optimal energy and performance balancing. Core-disabling is realized with the design of a controller architecture capable of accurate control and modeling of the relationship between supply voltage and platform throughput. The controller enables the user to configure the throughput of the cores implemented, parameterize the throughput, and map to various core-voltage and core-disabled

combinations. The mapping creates a unidirectional relationship between energy consumption and performance. For the homogeneous platform controller, the platform throughput is observed and compared with the mapping to determine the core-voltage and core-disabled combination that provides lowest energy consumption while maintaining performance. For heterogeneous platform controller, each core's throughput is monitored individually and mapped to a certain supply voltage. Core-disabling has been made possible by observing the input pulse time (*IPT)* of the input data and disable cores while there are no data presented at platform input rails.

Both platforms with controllers are implemented using IBM 130nm 8RF-DM technology. Transistor-level simulation results for the homogeneous platform indicates that for *IPT* down ramp scenario with stepping between 1ns and 5ns, the platform has 10% to 15% less energy consumption and 22% to 28% better performance compare to platform with DVS. For *IPT* up ramp scenario with stepping between 4ns and 5ns, the platform has 7% to 14% less energy consumption and 3% to 9% better performance compare to platform with DVS. Heterogeneous platform simulation results show that for *IPT* down ramp scenario, the platform has 10% to 18% less energy consumption and 5% to 6% degrade in each core's performance compare to platform with DVS if *IPT* stepping is ≥2.5ns. For *IPT* up ramp scenario, platform has 5% to 14% less energy consumption with 3% to 10% degrade in each core's performance compare to platform with DVS if *IPT* stepping is ≤2.5ns. Both platforms with controllers have been demonstrated to be capable of best balancing energy efficiency and performance with DVS and core-disabling under a wide range of application scenarios.

This research demonstrates the significant advantage of core-disabling in balancing energy and performance of large scale parallel computing platforms. For future work, the

available core-voltage and core-disable combinations can be expanded by increasing the scale of the supply voltage for finer energy consumption and throughput mapping and better usage of power based on workload.

8 References

[1] Brooks, David M., et al. "Power-aware microarchitecture: Design and modeling challenges for next-generation microprocessors." *IEEE Micro* 20.6 (2000): 26-44.

[2] Linder, Michael, Jia Di, and Scott C. Smith. "Multi-threshold dual-spacer dual-rail delay-insensitive logic (MTD3L): A low overhead secure IC design methodology." *Journal of Low Power Electronics and Applications* 3.4 (2013): 300-336.

[3] Beerel, Peter A., and Marly E. Roncken. "Low power and energy efficient asynchronous design." *Journal of Low Power Electronics* 3.3 (2007): 234-253.

[4] Alarcón, Louis P., et al. "Exploring very low-energy logic: A case study." *Journal of Low Power Electronics* 3.3 (2007): 223-233.

[5] Hinds, Michael, et al. "An asynchronous advanced encryption standard core design for energy efficiency." *Journal of Low Power Electronics* 9.2 (2013): 175-188.

[6] Moore, Simon, et al. "Improving smart card security using self-timed circuits." *Asynchronous Circuits and Systems, 2002. Proceedings. Eighth International Symposium on*. IEEE, 2002.

[7] Choi, Myungsu, and Minsu Choi. "Scalability of globally asynchronous QCA (quantum-dot cellular automata) adder design." *Journal of Electronic Testing* 24.1-3 (2008): 313-320.

[8] Apperson, Ryan W., et al. "A scalable dual-clock FIFO for data transfers between arbitrary and haltable clock domains." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 15.10 (2007): 1125-1134.

[9] Hollosi, Brent, et al. "Delay-insensitive asynchronous ALU for cryogenic temperature environments." *Circuits and Systems, 2008. MWSCAS 2008. 51st Midwest Symposium on*. IEEE, 2008.

[10] Karaki, Nobuo, et al. "A flexible 8b asynchronous microprocessor based on low-temperature poly-silicon TFT technology." *Solid-State Circuits Conference, 2005. Digest of Technical Papers. ISSCC. 2005 IEEE International*. IEEE, 2005.

[11] Asanovic, Krste, et al. *The landscape of parallel computing research: A view from berkeley*. Vol. 2. Technical Report UCB/EECS-2006-183, EECS Department, University of California, Berkeley, 2006.

[12] Men, Liang, Brent Hollosi, and Jia Di. "Framework of an adaptive delay-insensitive asynchronous platform for energy efficiency." *VLSI (ISVLSI), 2014 IEEE Computer Society Annual Symposium on*. IEEE, 2014.

[13] Herbert, Sebastian, and Diana Marculescu. "Variation-aware dynamic voltage/frequency

scaling." *High Performance Computer Architecture, 2009. HPCA 2009. IEEE 15th International Symposium on*. IEEE, 2009.

[14] Men, Liang, and Jia Di. "Asynchronous Parallel Platforms with Balanced Performance and Energy." *Journal of Low Power Electronics* 10.4 (2014): 566-579.

[15] Nielsen, Lars Skovby, et al. "Low-power operation using self-timed circuits and adaptive scaling of the supply voltage." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 2.4 (1994): 391-397.

[16] Ernst, Dan, et al. "Razor: A low-power pipeline based on circuit-level timing speculation." *Microarchitecture, 2003. MICRO-36. Proceedings. 36th Annual IEEE/ACM International Symposium on*. IEEE, 2003.

[17] Amrutur, Bharadwaj, et al. "Adaptative techniques to reduce power in digital circuits." *Journal of Low Power Electronics and Applications* 1.2 (2011): 261-276.

[18] Herbert, Sebastian, and Diana Marculescu. "Variation-aware dynamic voltage/frequency scaling." *High Performance Computer Architecture, 2009. HPCA 2009. IEEE 15th International Symposium on*. IEEE, 2009.

[19] Herbert, Sebastian, and Diana Marculescu. "Analysis of dynamic voltage/frequency scaling in chip-multiprocessors." *Low Power Electronics and Design (ISLPED), 2007 ACM/IEEE International Symposium on*. IEEE, 2007.

[20] Putic, Mateja, et al. "Panoptic DVS: A fine-grained dynamic voltage scaling framework for energy scalable CMOS design." *Computer Design, 2009. ICCD 2009. IEEE International Conference on*. IEEE, 2009.

[21] Shen, Hao, Jun Lu, and Qinru Qiu. "Learning based DVFS for simultaneous temperature, performance and energy management." *Quality Electronic Design (ISQED), 2012 13th International Symposium on*. IEEE, 2012.

[22] Li, Yee William, et al. "Asynchronous datapath with software-controlled on-chip adaptive voltage scaling for multirate signal processing applications." *Asynchronous Circuits and Systems, 2003. Proceedings. Ninth International Symposium on*. IEEE, 2003.

[23] Thonnart, Yvain, et al. "Power reduction of asynchronous logic circuits using activity detection." *IEEE transactions on very large scale integration (VLSI) systems* 17.7 (2009): 893-906.

[24] S. C. Smith and J. Di, Designing Asynchronous Circuits using NULL Convention Logic

(NCL). Morgan Claypool Publishers, 2009.

[25] Bailey, Andrew, et al. "Multi-threshold asynchronous circuit design for ultra-low power." *Journal of Low Power Electronics* 4.3 (2008): 337-348.

[26] Zhou, Liang, Scott C. Smith, and Jia Di. "Bit-Wise MTNCL: An ultra-low power bit-wise pipelined asynchronous circuit design methodology." *Circuits and Systems (MWSCAS), 2010 53rd IEEE International Midwest Symposium on*. IEEE, 2010.

[27] Bailey, Andrew, et al. "Multi-threshold asynchronous circuit design for ultra-low power." *Journal of Low Power Electronics* 4.3 (2008): 337-348.

[28] Dahlman, Erik, et al. *3G evolution: HSPA and LTE for mobile broadband*. Academic press, 2010.