CMOS VLSI Correlator Design for Radio-Astronomical Signal Processing

A thesis presented in partial fulfilment of the requirements for the degree of

Doctor of Philosophy
in
Engineering

at Massey University, Auckland, New Zealand

Stepan Lapshev

2018

Abstract

Multi-element radio telescopes employ methods of indirect imaging to capture the image of the sky. These methods are in contrast to direct imaging methods whereby the image is constructed from sensor measurements directly and involve extensive signal processing on antenna signals. The Square Kilometre Array, or the SKA, is a future radio telescope of this type that, once built, will become the largest telescope in the world. The unprecedented scale of the SKA requires novel solutions to be developed for its signal processing pipeline one of the most resource-consuming parts of which is the correlator. The SKA uses the FX correlator construction that consists of two parts: the F part that translates antenna signals into frequency domain and the X part that cross-correlates these signals between each other. This research focuses on the integrated circuit design and VLSI implementation issues of the X part of a very large FX correlator in 28 nm and 130 nm CMOS. The correlator's main processing operation is the complex multiply-accumulation (CMAC) for which custom 28 nm CMAC designs are presented and evaluated. Performance of various memories inside the correlator also affects overall efficiency, and input-buffered and output-buffered approaches are considered with the goal of improving upon it. For output-buffered designs, custom memory control circuits have been designed and prototyped in 130 nm that improve upon eDRAM by taking advantage of sequential access patterns. For the input-buffered architecture, a new scheme is proposed that decreases the usage of the input-buffer memory by a third by making use of multiple accumulators in every CMAC. Because cross-correlation is a very data-intensive process, high-performance SerDes I/O is essential to any practical ASIC implementation. On the I/O design, the 28 nm full-rate transmitter delivering 15 Gbps per lane is presented. This design consists of the scrambler, the serialiser, the digital VCO with analog fine-tuning and the SST driver including features of a 4-tap FFE, impedance tuning and amplitude tuning.

Acknowledgements

I would like to acknowledge my supervisor Rezaul Hasan without whose support this work would not have been possible.

Contents

List of figures

List of tables

List of acronyms and abbreviations

AC	Alternating current
ASIC	Application-specific integrated circuit
CLA	Carry-lookahead adder
CMAC	Complex multiplier-accumulator
CML	Current-mode logic
CMOS	Complementary metal-oxide semiconductor
CSA	Carry-save adder
CV	Coefficient of variation
DAC	Digital-to-analog converter
DC	Direct current
DET	Dual-edge-triggered
DFT	Discrete Fourier Transform
DRAM	Dynamic random-access memory
DTSCR	Diode-triggered silicon controlled rectifier
EDA	Electronic design automation
eDRAM	Embedded dynamic random-access memory
ESD	Electrostatic discharge
FA	Full adder
FF	Flip-flop
FFE	Feed-forward equaliser
GF	GlobalFoundries
HA	Half adder
IC	Integrated circuit
IDDQ	Leakage current
I/O	Input/output
IP	Intellectual property

LFSR	Linear-feedback shift register
LVDS	Low-voltage differential signalling
MC	Monte Carlo
MOSFET	Metal-oxide-semiconductor field-effect transistor
MSB	Most significant bit
MUX	Multiplexer
PDP	Power-delay product
PFA	Partial full adder
PLL	Phase-locked loop
PRNG	Pseudorandom number generator
PVT	Process, voltage and temperature
RAM	Random-access memory
RSD	Relative standard deviation
SAM	Sequential-access memory
SCR	Silicon controlled rectifier
SD	Standard deviation
SerDes	Serialiser/deserialiser
SET	Single-edge-triggered
SI	Sub-integration
SST	Source-series termination
VCO	Voltage-controlled oscillator
VLSI	Very-large-scale integration

Chapter 1

Introduction

This work deals primarily with the design of integrated circuits. The introductory chapter provides background information about radio astronomy, multi-element radio telescopes, radio interferometry and correlators. The chapter also presents motivations and the overview of this work.

1.1 Telescopes with multiple antennas

Radio astronomy is a field of science that deals with observations of space objects such as stars and galaxies in radio spectrum. There are different types of radio telescopes that can image these objects. Telescopes employ different imaging techniques depending on the number of antennas that they consist of: single-antenna telescopes measure the image directly and telescopes with multiple antennas calculate the image from the measured signals.

The two types of telescopes are related. For a single-dish telescope, the maximum angular resolution θ is limited by the effects of diffraction and is given by the following relationship between the wavelength of the observed signal λ and the diameter of the dish D [1, Ch. 6]:

$$\theta = \frac{\lambda}{D}. \tag{1.1}$$

Multi-element telescopes use signal processing techniques that allow multiple antennas to act as a single imaging instrument with the maximum resolution equivalent to that of a single-dish telescope the size of the entire antenna array [1, Ch. 9]. For example, antennas separated 10 000 km apart can produces images as large as a single dish that is 10 000 km in diameter. Thus, only a few antennas are enough to be able to generate a high-resolution image. Although increasing the number of antennas while keeping the same maximum separation does not increase the maximum resolution, using more antennas increases the total collecting area, which improves many other important characteristics of the telescope such as its sensitivity and the survey speed [2, Sec. 6.5].

The process of combining signals from multiple antennas to generate a single image is called aperture synthesis [1, Ch. 9], [2]. One of the key techniques of aperture synthesis is interferometry. The use of interferometry can be explained with the help of Figure 1.1 which shows a diagram of two antennas receiving signals from different sources. Although each antenna measures the sum of all signals, the distance that separates these antennas ensures that phase differences between these signals are different for different antennas. Interfering signals for every

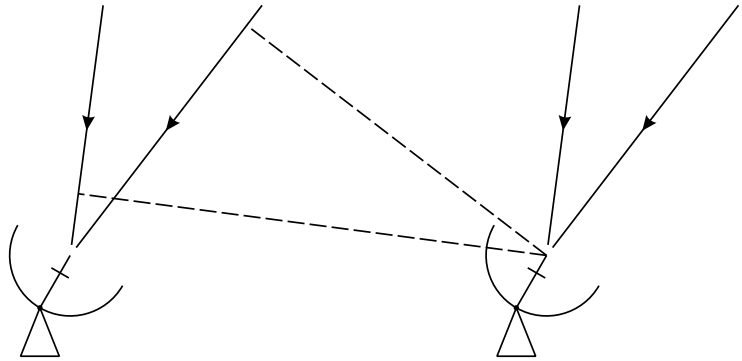


Figure 1.1: Diagram of two antennas receiving two signals. Although both antennas receive the sum of the two signals, the phase difference between these signals is different for different antennas.

antenna pair allows the interferometer to extract this information and ultimately reconstruct the image. Modern telescopes use the cross-correlation operation for this purpose, which is implemented inside the correlator.

1.2 The correlator

The correlator produces a cross-correlation signal for every distinct pair of antenna signals. Every such pair is called a baseline. Every signal is also correlated with itself to produce an auto-correlation baseline. Since dual-polarisation antennas produce two signals, an array of N such antennas produces $2N$ input signals for the correlator. This amounts to $(2N^2 - N)$ cross-correlation baselines and $2N$ auto-correlation baselines making the size of the correlator scale at least quadratically with N.

The cross-correlation operation measures how similar the correlated signals are as a function of time displacement or lag. Time-domain correlation of signals s_1 and s_2 can be written as follows [3, p. 401]:

$$(s_1 \star s_2)[t] = \sum_{a=-\infty}^{\infty} s_1^*[a] \cdot s_2[a+t], \tag{1.2}$$

where s_1^* is the complex conjugate of s_1 and the indices are time instances. In practice, sequence lengths and summation bounds are chosen to achieve a specific frequency resolution after the frequency transform. When correlating in frequency domain, the output for each frequency channel c can be written as a complex multiplication [4, pp. 46 and 243], [5, p. 219]:

$$(s_1 \star s_2)[c] = s_1^*[c] \cdot s_2[c]. \tag{1.3}$$

Frequency channelisation itself is performed by the DFT; however, modern telescopes use more complicated frequency transformations [6].

The input into the correlator is in time domain, but for the purposes of imaging the output is required to be in frequency domain [7]. The correlator can then be thought to consist of two parts: the X part that performs cross-correlations and the F part that translates its input into frequency domain. Changing the order of X and F creates two basic correlator architectures: the XF correlator that cross-

correlates in time domain and the FX correlator that cross-correlates in frequency domain. The idea for the FX correlator was proposed in [8]. When N is large, the FX structure is computationally more efficient than the XF structure, which was explained in [9] and [10] for its first implementation. Whereas both X and F are proportional to N^2 in an XF correlator, only the X part is proportional to N^2 in an FX correlator while the F part is proportional to just N. This follows from the fact that the F part processes individual antenna signals in the FX correlator rather than the correlation signals for each baseline as in the XF correlator.

For each baseline, the correlator output consists of integrated data samples for all C frequency channels. These data samples are called visibilities. Integration is performed by accumulating the correlated samples T times for each frequency channel of each baseline. T is called the integration count and is related to the dump time of the correlator τ and the per-signal sample rate B with $T = \tau B$.

The planned Square Kilometre Array radio telescope [11] will consist of several imaging instruments using multiple correlators with various parameters. For example, one of the modes of the SKA Mid correlator will process 8-bit data and will have the following parameters: $C \approx 250\,000$, $N \approx 254$, $B \approx 1.4$ Gs/s and $\tau_{\text{min}} \approx 0.1$ s. Other modes will use 8-bit, 4-bit and 3-bit data at different rates. The first construction phase will see a smaller version of the SKA Mid built with $N \approx 190$ and $C \approx 65\,000$ [12].

This work focuses on the architecture and implementation issues of the X part of very large FX correlators when using very-large-scale integration (VLSI) technologies. These apply to correlators the size of the SKA and larger.

1.3 Motivation

Advances in computing technologies are essential to advances in science. Radio telescope interferometers are an example of this as their scientific capabilities are often limited by the processing ability of their supercomputers. The future telescope Square Kilometre Array is envisioned to be considerably larger in size and science output than any of the existing telescopes such as the ALMA [13], JVLA [14], ASKAP [15], MWA [16] or LOFAR [17]. This scale is set to create challenges for the implementation of the telescope's signal processing pipeline and especially its correlator.

A comparison can be made between the SKA and, for example, the LOFAR. The LOFAR consists of about 50 antenna stations, which makes it one of the largest telescopes currently in operation. The SKA Low instrument will be making observations in a similar frequency range but with about 1,000 antenna stations and with higher sensitivity and survey speed [11], [12]. The large number of stations alone will create more than a million baselines for the SKA Low's correlator versus only a few thousand for the LOFAR. This increase is a significant challenge for the signal processing hardware as incremental improvements in computing technology will not allow existing solutions to be simply scaled up by orders of magnitude [18]. As the number of antennas increases, the correlator's processing requirements experience quadratic growth and new methods and more efficient technologies are needed to be able to meet the technical requirements of such future telescopes.

VLSI technologies deal with the creation of integrated circuits (ICs). Efficient implementation of custom signal processing circuits and tight integration between their various components can be achieved with application-specific ICs (ASICs).

The desired functionality can be implemented exactly and with few compromises on an ASIC. For example, custom circuits can be made to efficiently handle large-scale 4-bit arithmetic, which may not be possible without purpose-built hardware. Moreover, custom circuits can incorporate specific optimisations that improve their performance characteristics in a particular use case when compared against more universal designs.

High research and development costs of ASIC design are often a concern. With the SKA expected to be in operation for many decades, most of the SKA's costs are likely to be incurred during the operation of telescope rather than its construction. The higher costs of ASIC development are thus likely to be offset by the ASICs' reduced power dissipation and increased performance. The combination of efficiency and long-term value makes ASICs a good fit for long-term projects with demanding technical requirements, yet little effort has gone into IC design for modern correlators in modern CMOS technologies. This makes ASIC design of correlator circuits a relevant research topic.

1.4 Thesis outline

The thesis is organised into six chapters. Chapter 2 presents custom 4-bit and 8-bit CMACs that have been designed in a 28 nm CMOS technology. The presented designs are thoroughly optimised high-performance circuits suitable for use in an ASIC correlator. The designs merge the carry-save adder trees of the multipliers comprising each CMAC to minimise the use of expensive carry-lookahead adders. The circuits employ the dual-edge-triggered clocking scheme to reduce the clock power dissipation.

Chapter 3 considers the common input-buffered FX correlator construction and proposes incorporating multiple accumulators in every CMAC to reduce the utilisation of the input-buffer memory by a third. In this chapter the CMACs from Chapter 2 are adapted to include three accumulators per CMAC. The performance characteristics of such CMACs are evaluated and compared to conventional one-accumulator CMACs and are found to be at least as good. The three-accumulator 8-bit CMAC exhibits reduced power dissipation within its multiplier circuits due to reduced input switching activity, which more than makes up for the increased power dissipation within its accumulator circuits. This reduction in the CMAC power dissipation is in addition to the reduction in the memory.

Chapter 4 continues on the topic of design of dual-edge-triggered storage elements that were used in the CMAC designs in Chapter 2 and adapted for the multiple-accumulator approach in Chapter 3. Five novel DET flip-flop designs are presented. The new designs are extensively evaluated in simulation and compared against a number of existing designs. The common features of these designs are the use of C-elements to reduce the energy dissipation due to glitches at the flip-flops' data inputs.

Chapter 5 discusses the output-buffered FX correlator construction and presents the design of custom memory control circuits for the correlator's output buffer implemented as a dynamic memory. The chapter includes the test report of their prototype that has been fabricated using 130 nm CMOS technology. This type of memory can be used in place of embedded DRAM to achieve much better timing and energy performance characteristics.

Chapter 6 presents the design of the full-rate 15 Gbps transmitter circuit in 28 nm CMOS. The design uses the 64b/66b line encoding scheme. The circuit includes the scrambler, the serialiser, the digital VCO with analog fine-tuning, the driver and the ancillary circuits. The driver is of the Source Series Terminated type. The driver consists of five segments for each of the four taps of the Feed Forward Equaliser and the amplitude-tuning segment. The replica biasing circuit takes as its input the FFE, impedance and amplitude parameters and sets bias voltages for all driving segments. The impedance is additionally controlled by adjusting the termination resistor to achieve better linearity. DTSCR-based ESD protection is employed with an asymmetric T-coil. The design of the LVDS 315.25 MHz reference clock input amplifier is also presented.

Finally, Chapter 7 concludes the thesis and gives an overview of potential future work. The appendix contains two sections: Appendix A details the netlists of the multiplier designs from Chapter 2 and Appendix B includes the list of publications that have been based on the work presented in this thesis.

Chapter 2

Digital CMAC design

2.1 Complex multiplication and CMAC

Before the correlator circuits can be designed, their exact function needs to be ascertained. The key processing operation that the correlator performs is the complex multiplication as per the cross-correlation formula in (1.3). The complex multiplication of two numbers $G = a + ib$ and $H = c + id$ can be written as follows:

$$G \cdot H = (a + ib)(c + id) = (ac - bd) + i(ad + bc). \tag{2.1}$$

The complex conjugate is applied to one of the numbers before the multiplication as per (1.3). This results in the following expression that the correlator circuits must implement:

$$G \cdot H^* = (a + ib)(c - id) = (ac + bd) + i(bc - ad). \tag{2.2}$$

Although expressions (2.1) and (2.2) may look different at first glance, they are actually the same from the point of view of the implementation. This can be demonstrated by swapping the real and imaginary parts of one of the numbers before the multiplication:

$$(b + ia)(c + id) = (bc - ad) + i(ac + bd), \tag{2.3}$$

which yields the same result as in (2.2) but with the swapped real and imaginary parts. Thus, the cross-correlation operation can be implemented with a regular complex multiplier with the outputs and some of its inputs relabelled as illustrated in Figure 2.1 in order to implement the complex conjugate.

The output from the complex multiplier is integrated using an accumulator. Strictly speaking, only the implementation of (2.1) along with an accumulator is a complex multiply-accumulator (CMAC). In this work the term CMAC is used to refer to the implementation of (2.2). This is justified because although (2.1) is mathematically different to (2.2), their implementations result in identical circuits only with different input and output port names.

The functional diagram of the CMAC processing circuits following (2.2) and including accumulators is shown in Figure 2.2. The CMAC design deals with the design of multipliers, adders and accumulator circuits with the multipliers being a major of part of the overall design.

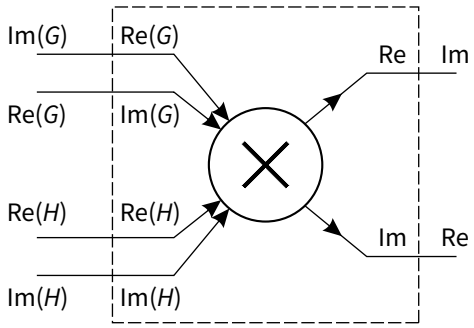


Figure 2.1: A complex multiplier can calculate the cross-correlation without performing the complex conjugate operation explicitly. This can be achieved by simply relabelling the ports for one input and relabelling the outputs as explained by (2.3). In this case, the real and imaginary parts of the *G* input are swapped.

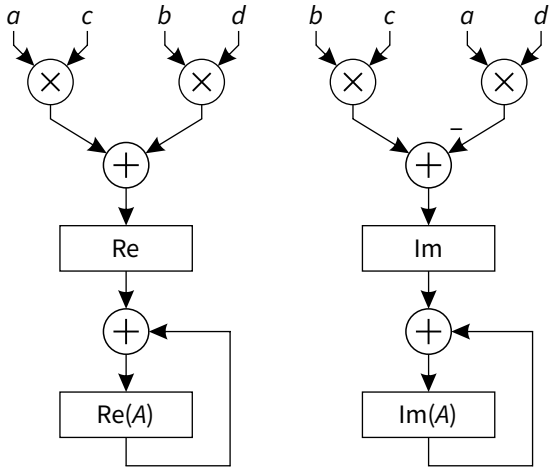


Figure 2.2: CMAC functional diagram following (2.2) and including accumulators.

```
        1000000000000
      0011010001000
    1000000000000
  1100101111011
1     1           1
-------------------
1 0001110001101000  = 7272_10
```

Figure 2.3: Example of how a signed 4b×12b multiplication can be performed without sign extensions. The two multiplication operands are −6, which is 1010 in 2's complement binary, and −1212, which is 101101000100. The result is truncated to 16 bits.

2.2 CMAC multiplier design

A multiplier generally operates on two input numbers, or operands, to produce the result of their multiplication at its output. Common parallel multipliers work by generating partial products which are then added up with an adder tree. Partial products themselves are 1-bit multiplications of each bit of the first operand by the entire second operand. The correlator implements signed arithmetic and thus the design of 2's complement multipliers and adder trees is of interest. One of the features of naive 2's complement additions is that all addition operands need to be extended to the bit-length of the final result, which in the case of a multiplier adder tree is a significant overhead. This can be avoided by using the method first introduced in [19] which is now called the Baugh and Wooley multiplication method.

Figure 2.3 shows an example of how an adder tree of a signed multiplication can be done without sign-extending the partial products. The sign extension for each partial product is dealt with by adding a 1 at its MSB position, which has the effect of inverting the MSB and zeroing out the sign extension. These 1s are subtracted from the result, which explains the two leftmost 1s in the diagram. Adding 1s at MSB positions and subtracting them from the final result effectively removes the sign extension for the second multiplication operand. In order to remove the sign extension for the first operand (i.e., in order to not produce more partial products than there are bits in the first operand), the last partial product is subtracted rather than added. As the diagram illustrates it, handling partial products this way works out to the correct result. Although this exact method is not the same as the Baugh and Wooley multiplication algorithm, methods that use extra 1s and subtract some partial products are generally credited to them.

After partial products are generated, they are added up with the help of an adder tree. Common parallel adder tree construction methods include the Wallace tree [20] and the Dadda tree [21], which are both carry-save adder (CSA) tree variants. CSA trees work by adding up more than two numbers down to just two which are then added with a normal adder. Carry-save addition is attractive for adder trees because it is one of the constant-time addition methods whose delay depends only on the number of the addition operands rather than their bit-length [22]. The caveat is that the final result from such a tree is in the "carry-save" encoding which is a redundant encoding that is converted to the normal binary encoding using a standard adder. This work uses carry-lookahead adders (CLAs)

at the output of CSA trees to resolve the carry signals of the carry-save encoding instead of the more common but slower ripple-carry adders. The CLA structure was chosen not only because of its speed but also its versatility in designing adders of various bit-lengths using a common set of CLA circuits [23].

A CSA tree is constructed from full adders and half adders that reduce the tree in sequential stages. A full adder (FA) is a circuit that adds up three bits of the same weight and outputs their 2-bit sum in the form of the sum bit and the carry-out bit. A FA is also known as the 3:2 compressor since it reduces its 3 inputs to just 2 outputs. A half adder (HA) is similar to a FA but it only adds up 2 bits of the same weight to produce 2 bits at the output. HAs do not reduce the tree since they produce the same number of outputs as they have inputs. HAs can thus be thought to only redistribute signals between weights.

The Wallace and Dadda tree construction algorithms are the most famous methods of determining exact positions of FAs and HAs, yet numerous other methods exist. This work uses the method described in [24] and [25], which is essentially a variant of the Wallace tree method that strives to minimise the use of HAs in favour of using more FAs. This achieves an overall reduction in hardware without compromising on anything else except perhaps the regularity of the tree. The comparison of this method to the Wallace tree is shown in Figure 2.4 which contains dot diagrams of adder trees for the signed 8-bit multiplication. The notation that is used in this work is similar to notations from [26] and [27]: each dot represents a single bit; a hollow dot denotes an inversion on the given bit, which only happens for some bits in the first stage of the tree to avoid sign extensions; encirclements of three bits represent placements of FAs and that of two bits represent HAs. Each two- and three-bit encirclement creates a sum bit, or a single dot, at the same weight and a carry-out bit for the higher weight in the next tree stage.

Based on the theory introduced above, multiplier adder trees for the 4-bit and 8-bit CMACs have been constructed. As per (2.2), the CMAC produces two numbers that are the real and imaginary parts of the complex-valued result. Rather than constructing separate trees for each of the CMAC's four multiplications, which would produce a regular CMAC design with four multipliers, the two multiplications for each of the real and imaginary parts are combined together into a single adder tree. In this way, the two adder trees calculate the result of the complex multiplication directly without explicitly producing the results of any of the individual multiplications. This is more efficient considering area, timing and energy performance since fewer carry-resolution circuits of CLA adders have to be employed. Thus, each CMAC consists of only two adder trees, one for each of the real and imaginary parts of the output.

Figures 2.5 and 2.6 show the designed adder tree diagrams of the 4-bit and 8-bit CMACs. The presented adder trees are human-readable illustrations. The netlists detailing the internal tree signals and FA and HA placements for both CMACs are given in Appendix A. The following sections present the implementation of these CMACs including the details of their digital circuits.

2.3 Design of digital circuit cells

All circuits have been designed using the high-performance GF 28HPP CMOS technology. The development has mostly followed the approach of building larger

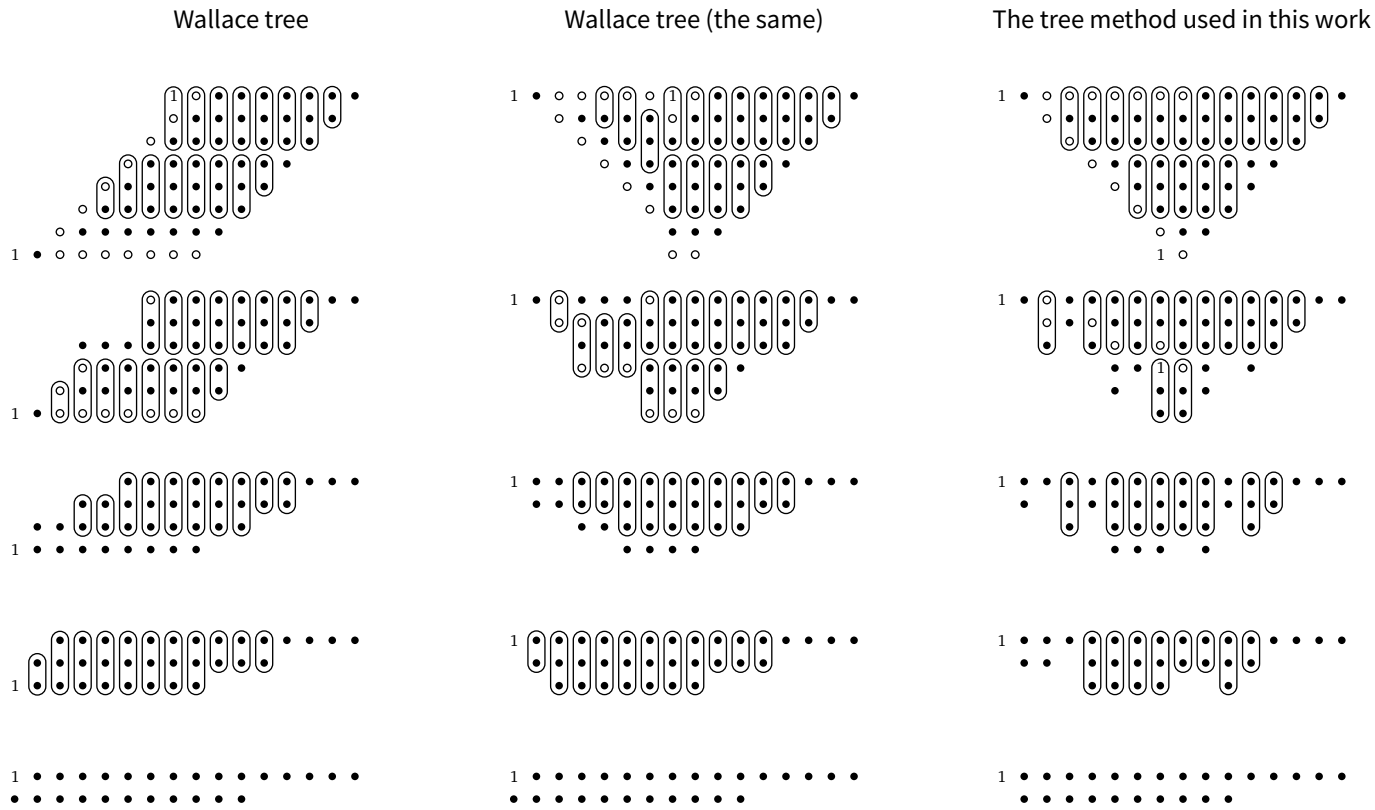


Figure 2.4: An example of the adder tree design for the signed 8-bit multiplication using the Wallace tree (left and centre columns) and the tree method from [25] that is used in this work (right column). The Wallace tree uses 38 FAs and 15 HAs while the design method uses 39 FAs and 7 HAs. Symbol "(1 • •)" indicates the addition of two bits with a 1, which is a special case of a HA, rather than a FA.

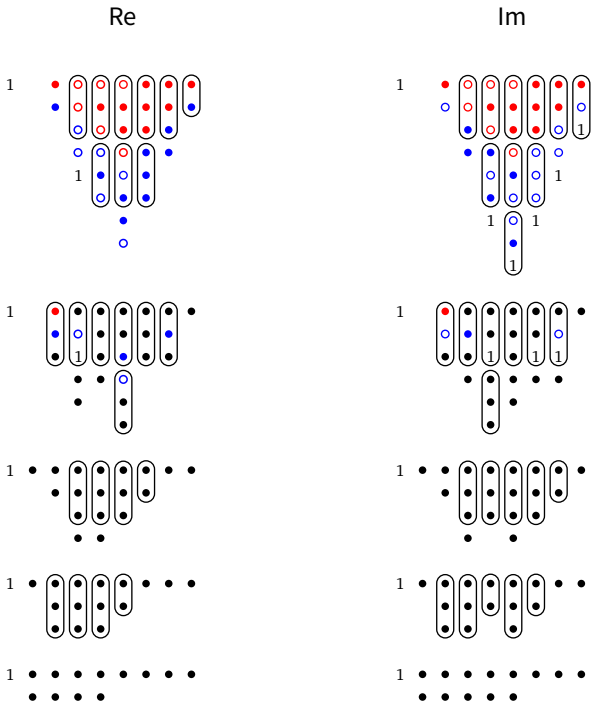


Figure 2.5: Adder tree diagrams for the 4-bit CMAC for the real (left tree) and imaginary (right tree) parts of the result of the cross-correlation operation. Red and blue colour denote partial product bits from different multiplications.

circuits from smaller circuit cells. Although a collection of standard cells is provided with this technology by a third party organisation, these cells have not been used in final versions of any design.

2.3.1 Adder circuits

The key building blocks of the adders used in this work are FAs and HAs. Their efficient implementation is thus key to the overall efficiency of any design that uses them. This is especially true for the FA cell, which is the most important cell of the adder trees.

Partial products are generated right before the adder tree. Their individual bits are 1b×1b multiplications of each bit of one multiplication operand with each bit of the other operand. A 1-bit multiplication can be implemented as an AND gate. However, the static CMOS circuit design style does not have an AND gate, only NAND. Simply inverting the output of a NAND to make an AND incurs a significant power overhead as the number of partial product bits is proportional to the square of the operands' bit-length. Although due to De Morgan's laws, the AND gate can be implemented with a single NOR gate as follows:

$$A \cdot B = \overline{\overline{A} + \overline{B}}, \tag{2.4}$$

the NOR gate is slightly less energy efficient than the NAND gate for the same timing performance, which is due to its use of larger p-type transistors. Thus, the simple NAND gate was chosen to generate "active low", or inverted, partial products. Accordingly, all of the adder tree circuits have been designed to have

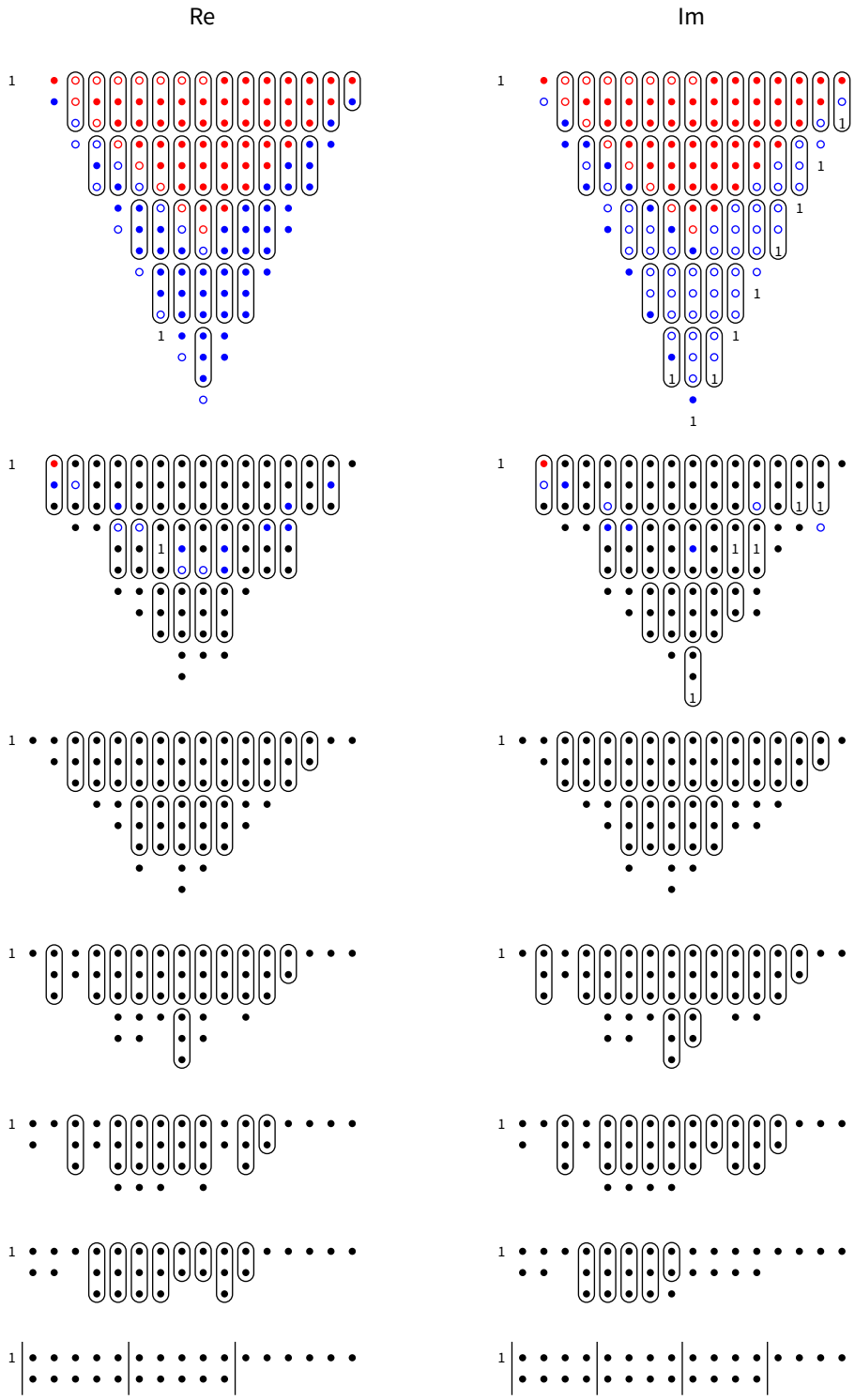


Figure 2.6: Adder tree diagrams for the 8-bit CMAC for the real (left tree) and imaginary (right tree) parts of the result of the cross-correlation operation. Red and blue colour denote partial product bits from different multiplications. The vertical lines for the output of the tree denote the placements of 4-bit and 5-bit CLA sections.

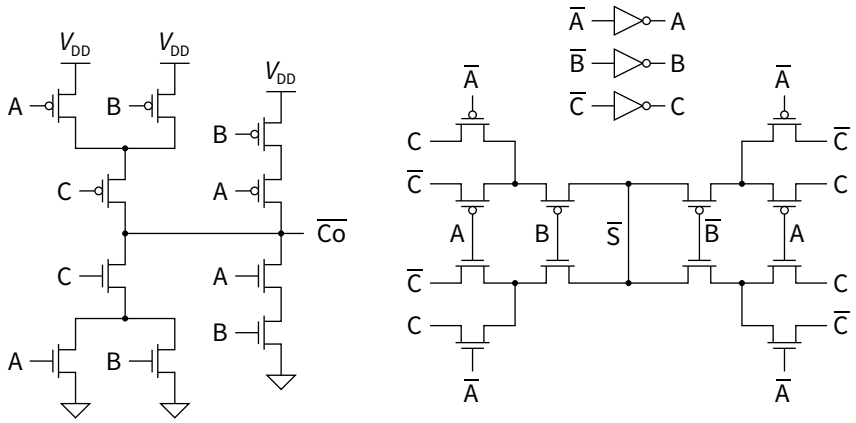


Figure 2.7: Transistor-level schematic diagram of the FA circuit that has been used throughout most of this work. The sum output is S and the carry-out output is Co.

"active low" inputs and outputs. It can be noted that there is no difference between the "active low" and "active high" versions of the FA as simply inverting all values in its truth table [28, p. 119] yields the same table. The "active low" notation is still kept for the FA for the sake of consistency.

The schematic diagram of the FA that has been used throughout most of the multiplier circuits in this work is shown in Figure 2.7. The circuit uses the standard static carry generation circuit and a pass-transistor version of the 3-input XOR circuit shown in [29]. The XOR3 circuit construction is similar to CMOS pass-transistor and double pass-transistor circuit construction style of [30] and [31]. The use of unbuffered outputs from pass-transistor circuits actually complicates adder tree construction, because such an output should not be used as the input into the pass-transistor circuit again. Compared to the conventional FA circuit [32] shown in Figure 2.8, for the same number of transistors this design is inherently slightly faster as its critical path has fewer inverters.

The provided standard cell library includes several FA implementations that use two circuit topologies shown in Figures 2.8 and 2.9. Compared to the provided cells, the designed FA has been found to reduce the power dissipation of the adder trees in which it is used by more than 30% for the same timing performance. This has been found to be mostly due to the different transistor sizing approaches. The provided cells use transistors that are much larger than the optimal size from the point of view of the energy-delay trade-off in order to reduce circuit area. Even though transistors in these cells are large, certain routing techniques become possible with large transistors that reduce the space between n-type and p-type transistors thus reducing the overall area.

The designed HA circuit is shown in Figure 2.10 along with the provided standard cells. The designed HA has a shorter critical path when compared to the provided circuit. Just as the designed FA circuit, the HA uses a pass-transistor XOR to generate the sum signal.

Another important adder circuit is the CLA, which is used at the output of every adder tree and also in places where just two numbers need to be added. The CLA is the parallel adder structure introduced in [33] to speed up the addition of two numbers. The implemented CLA circuits include the partial full-adder (PFA) and carry-lookahead circuits constructed from NAND gates. Both the standard and

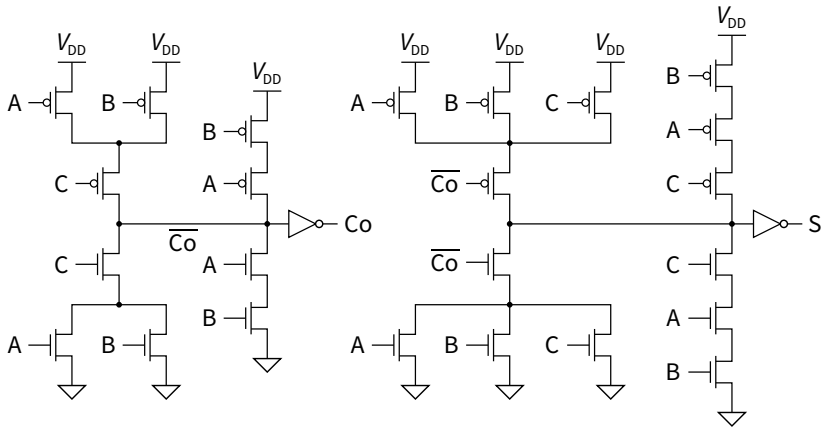


Figure 2.8: The conventional static FA circuit [32]. This is one of the structures that is used in the provided standard cell library.

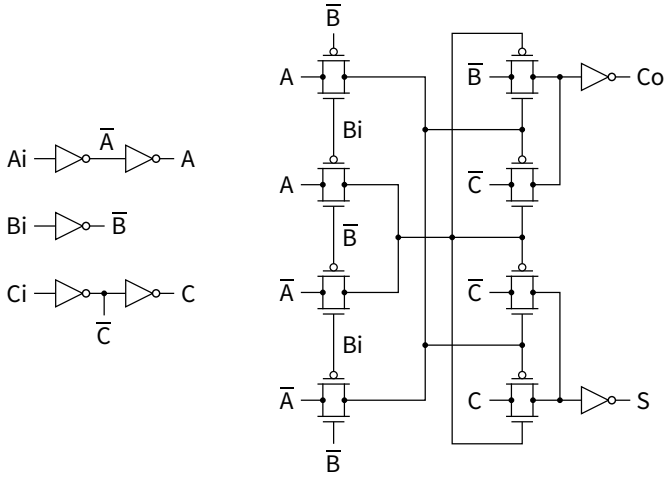


Figure 2.9: A pass-transistor implementation of a FA. This is one of the structures that is used in the provided standard cell library.

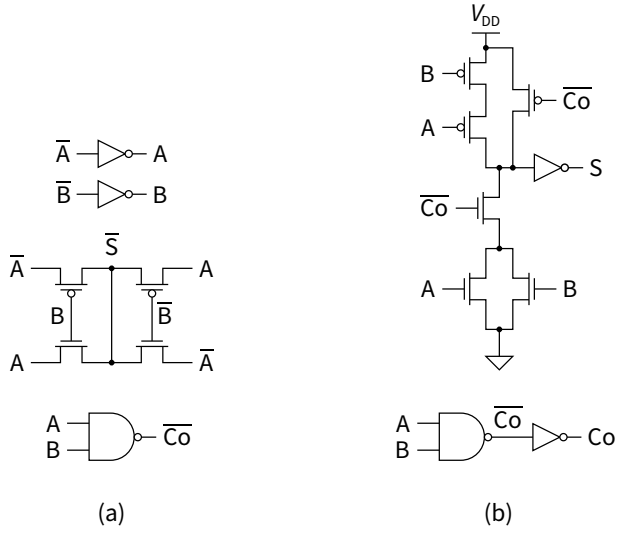


Figure 2.10: The HA schematic diagrams of (a) the circuit that is used in the work and (b) the circuit from the standard cell library.

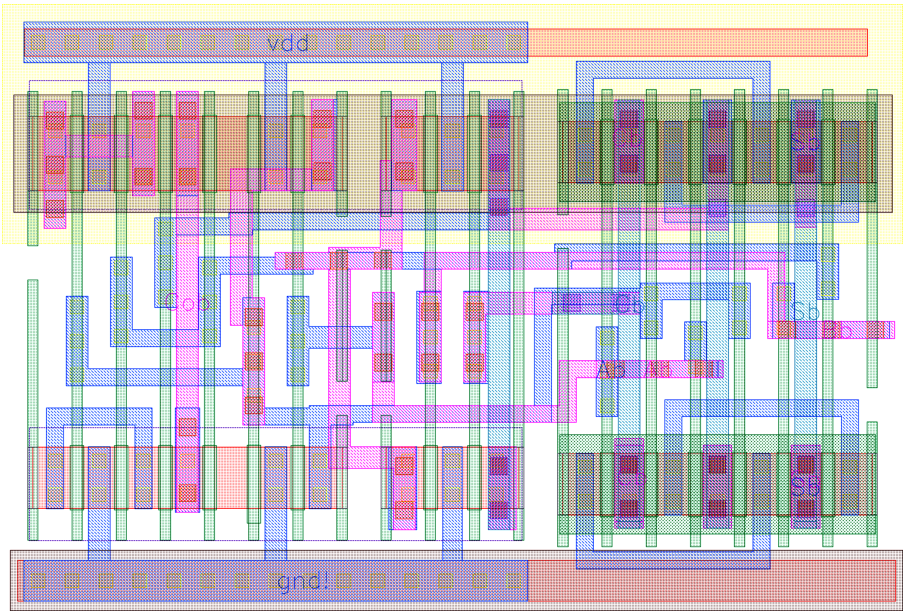


Figure 2.11: The layout implementation of the FA circuit from Figure 2.7.

hierarchical CLA adder sections have been designed for the 4-bit and 5-bit CLAs. Although less common, 5-bit CLA sections give more freedom in the implementation of adders of various bit-lengths, for example, making it possible to implement a 20-bit adder as a two-level hierarchical CLA adder.

Transistors of the adder tree cells were sized to achieve the best power efficiency given the timing requirements. The cell layouts were created while following the technology manufacturability guidelines such as the recommended minimums for the sizes of various layout features, which is important for design robustness in advanced process nodes such as 28 nm. The example of a layout of an individual cell is shown in Figure 2.11 for the FA circuit of Figure 2.7.

2.3.2 Memory registers

Each CMAC is implemented as a two-stage pipeline. The first stage is the computation of the cross-correlation and the second stage is the accumulation. Registers are used in between the pipeline stages and inside accumulator circuits. Registers have been implemented using dual-edge-triggered flip-flops (DET FFs).

DET FFs achieve the same data rate as the more common single-edge-triggered (SET) FFs at half the clock frequency, which usually leads to reduced power dissipation of synchronous logic circuits [34], [35]. Although DET FFs are more complicated to handle in EDA tools than SET FFs, their use can easily be adapted to existing design flows [36]. CMAC circuits use a common design called the Latch-MUX (LM) DET FF [34], [37] which consists of two input latches multiplexed to one output. The LM FF is essentially a DET version of the common SET Master-Slave FF. A pulsed latch design similar to the ep-DSFF (EP) DET FF of [38] is also used on some critical paths to allow for dynamic time-borrowing as introduced in [39]. Time-borrowing is also known as clock stretching.

The LM FF is the main flip-flop that is used throughout the design. The circuit diagram and layout for the variant that is used inside accumulators are shown in Figures 2.12 and 2.13. Multiple implementations of this FF have been made

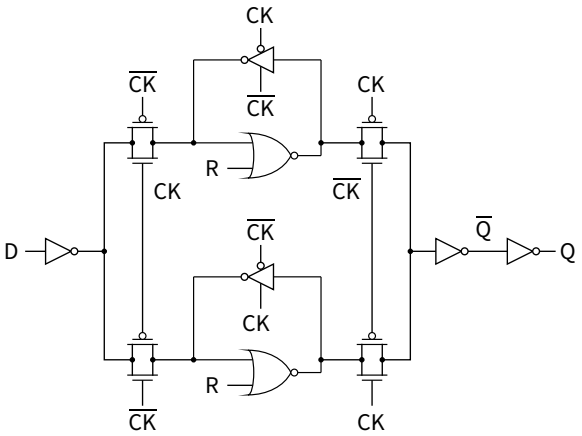


Figure 2.12: Schematic diagram of the LM DET flip-flop as used inside the CMACs' accumulators.

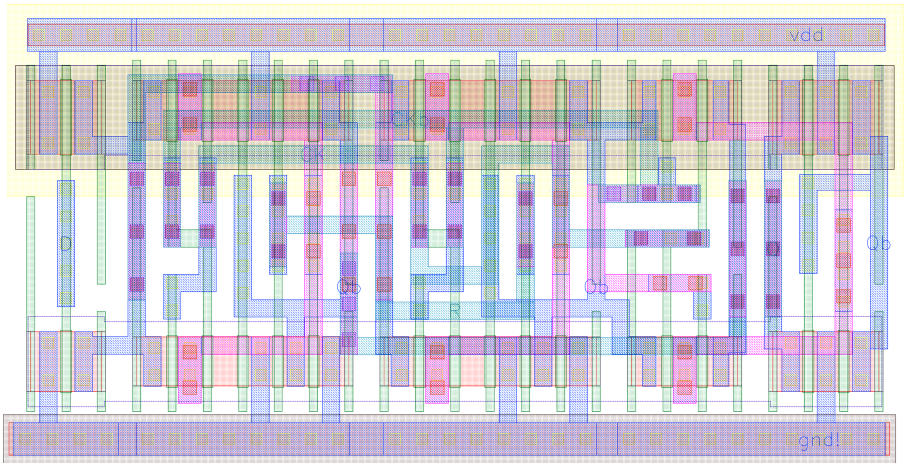


Figure 2.13: Layout view of the LM DET flip-flop from Figure 2.12 as used inside the CMACs' accumulators.

to fit in various circuits. These implementations have different driving strengths, set/reset configurations and different data input and clock signal buffering.

The other FF type that is used in the CMACs is the pulsed latch FF. This FF is essentially a latch whose transparency is controlled by the pulse signal. This FF possesses the "soft-edge" property, which allows it to capture its data input up to a certain amount of time after the clock transition. This additional time is used to effectively extend the evaluation time of critical circuit paths beyond the duration of the clock cycle at the expense of the following pipeline stage. In the case of CMACs, this technique extends the evaluation time for the complex multiplication at the expense of the accumulation, which increases the maximum pipeline frequency without having to increase the number of pipeline stages.

The pulsed latch FF design consists of two parts: the latch and the pulse generator. One feature of this FF is that the pulse generator can be shared among several latches, which improves the overall energy characteristics of the group of such FFs. Circuit diagrams for the pulse generator and the latch that are used in the 8-bit CMAC are shown in Figures 2.14 and 2.15 respectively. The design of

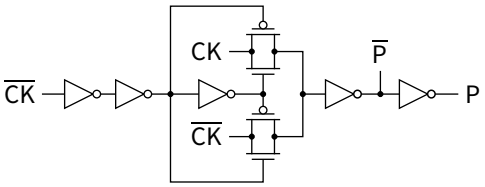


Figure 2.14: The schematic diagram of the pulse generator circuit of the pulsed-latch flip-flop that is shared among several latches.

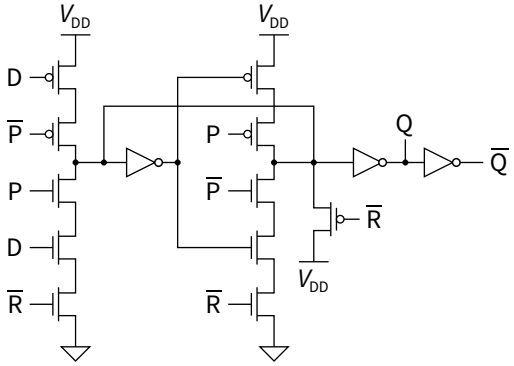


Figure 2.15: Schematic diagram of the reset version of the pulsed latch.

storage elements for the CMACs has spawned an effort to create more advanced DET flip-flops that are presented in Chapter 4.

2.4 CMAC implementation

Two CMACs have been created for the 4-bit and 8-bit input data. For the 4-bit design, the input data sample format is complex 8 bits (4 bits for each of the real and imaginary parts) and the output accumulation is complex 40 bits (20 bits for each part of the complex number) as is used in [40]. In the 8-bit design, the input is complex 16 bits and the output is complex 64 bits as is used in the SKA. The netlists for the adder tree circuits that detail internal tree signals and the placements of FAs and HAs inside the tree are given in Appendix A.

Figures 2.16 and 2.17 show the layouts of the implemented CMAC circuits for the 4-bit and 8-bit CMACs respectively. The circuit area for both designs is 1421 μm^2 and 3950 μm^2 respectively. Figures 2.18 and 2.19 show adder tree implementations for just the imaginary part of the cross-correlation operation of the two CMACs. In both cases the adder trees were designed in the shape of a triangle, which decreases local interconnect lengths despite potentially increasing the overall circuit area.

Post-layout analog simulations were performed to measure the performance parameters of the presented designs. Simulation parameters were chosen to simulate realistic worst-case performance. The simulation temperature was set to 60 °C. The input data was generated randomly to have 35% switching activity on every bit. In terms of timing performance, the two designs could safely operate at 2 GHz and 1.25 GHz for the 4-bit and 8-bit CMACs respectively. At 2 GHz, the 4-bit CMAC was found to dissipate 1.94 mW. At 1.25 GHz, the 8-bit CMAC dissipates 3.27 mW.

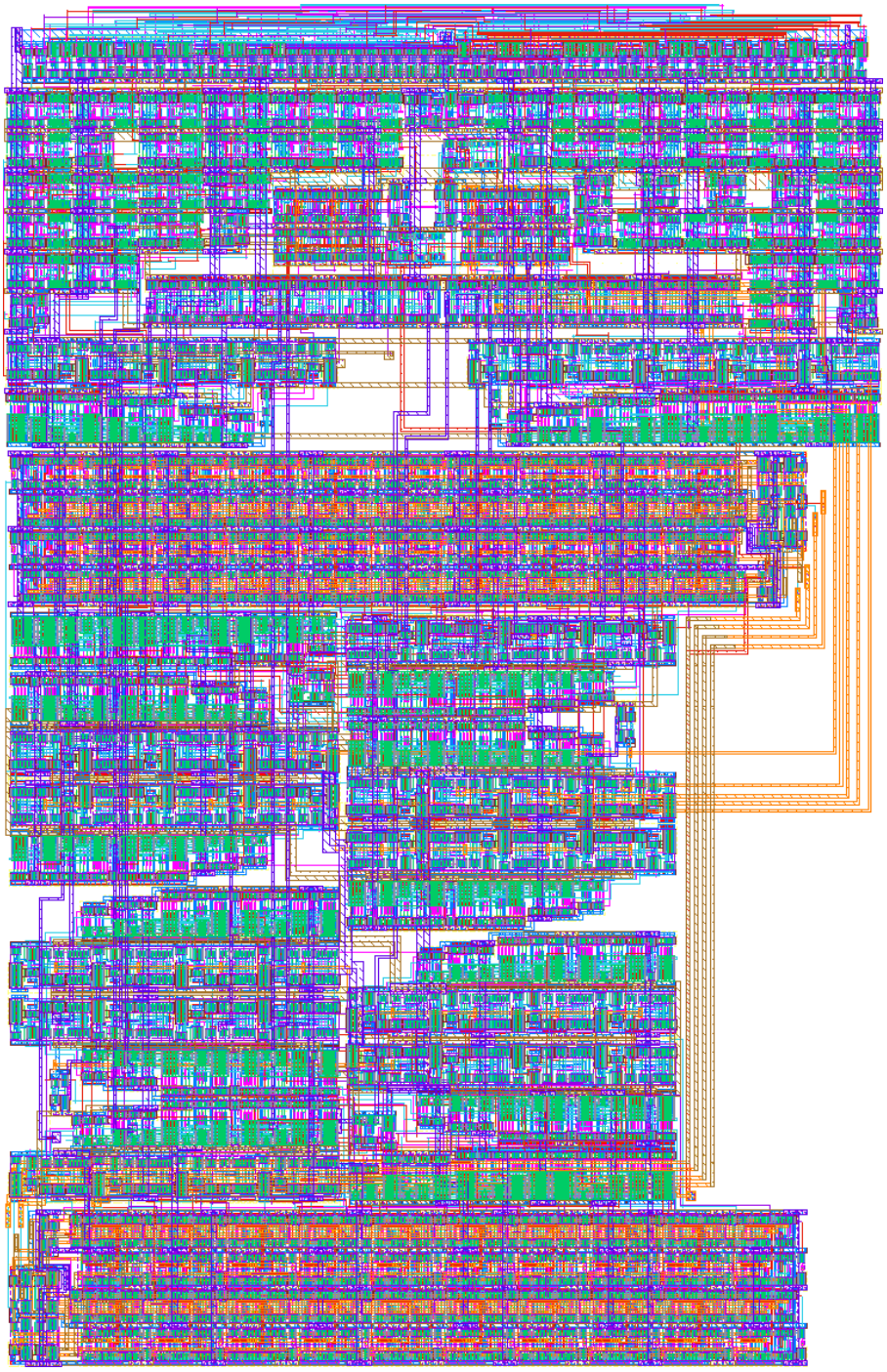


Figure 2.16: Layout of the 4-bit CMAC.

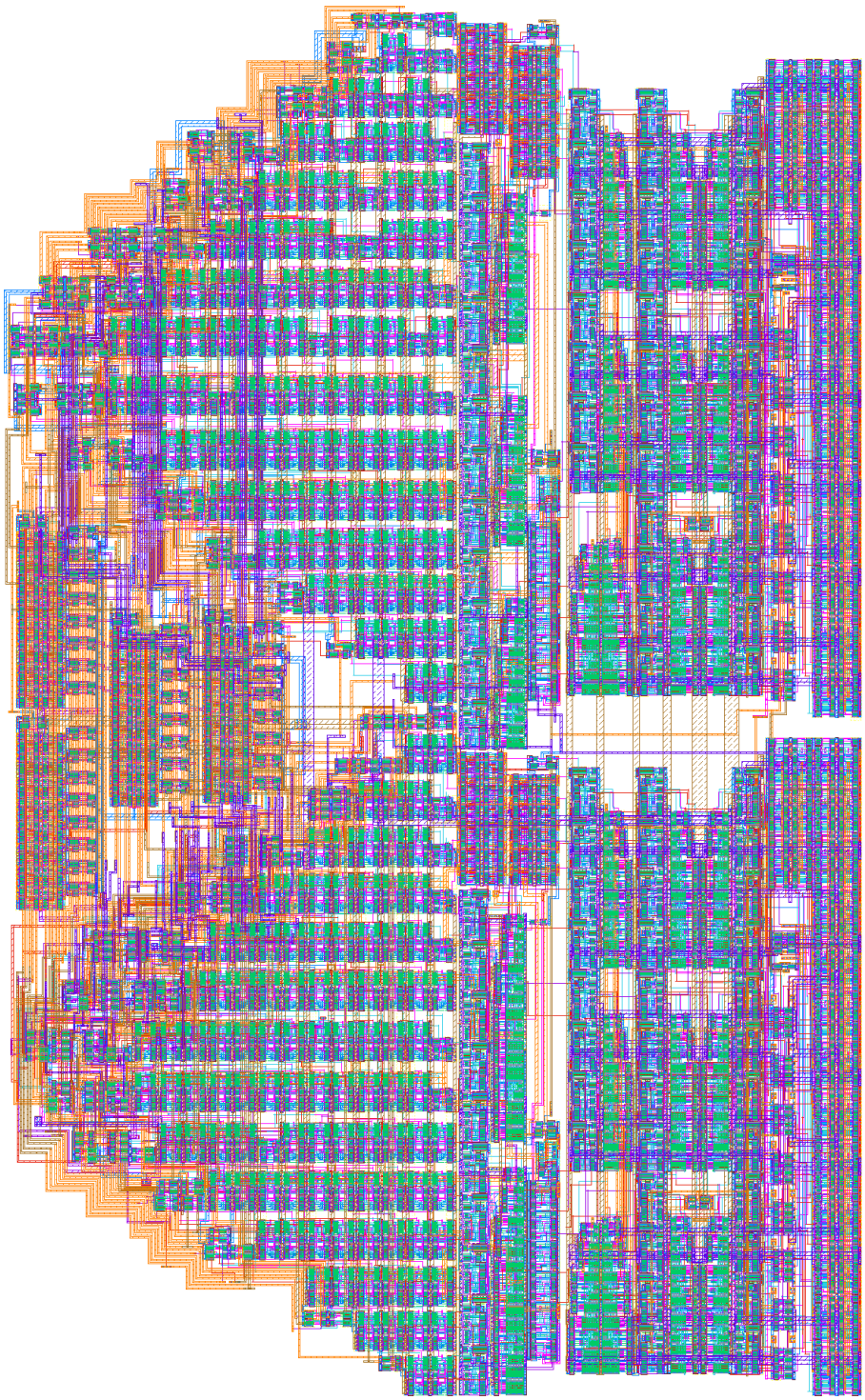


Figure 2.17: Layout of the 8-bit CMAC.

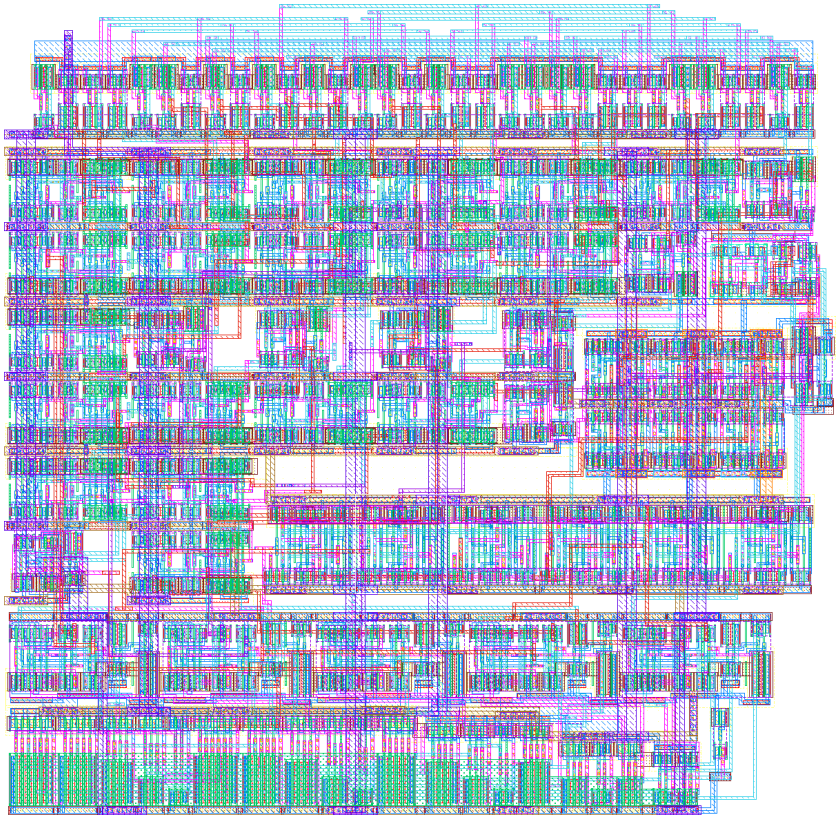


Figure 2.18: Layout for the imaginary part of the 4-bit CMAC.

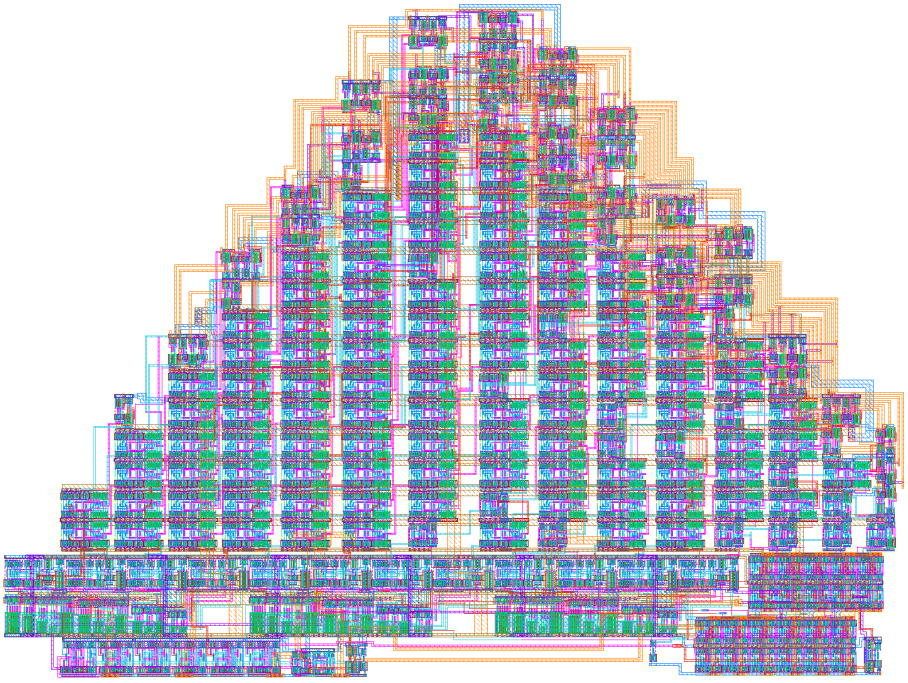


Figure 2.19: Layout for the imaginary part of the 8-bit CMAC.

Chapter 3

Multiple-accumulator CMACs for the input-buffered FX correlator

CMACs are used to implement the X part of an FX correlator. A popular architecture uses the input-buffered approach that reorders data from the F part for the convenience of processing in the X part. The F part naturally produces the entire spectrum of frequency channels for the time window on which it performs channelisation, yet a normal CMAC with a single accumulator can only integrate cross-correlations across time for the same frequency channel. This may create a need for an input buffer that can hold data samples for the entire period of the integration time and for all signals and the entire spectrum. This input buffer is also called a corner turner. The design of the corner turner itself is outside the scope of this work mostly because there is little that an ASIC implementation can bring to it since it only reorders data without performing any processing on it.

This chapter presents a new approach of using multiple-accumulator CMACs to reduce the utilisation of the input-buffer memory that is internal to the processing unit that contains such CMACs. In this chapter $\lfloor x \rfloor$ and $\lceil x \rceil$ denote the floor and ceiling operations respectively.

The chapter consists of five sections. Section 3.1 introduces the details of the considered architecture. Section 3.2 introduces the rules of grouping sub-integrations for multiple accumulators and explains the rationale behind reduced memory access. Section 3.3 quantifies memory access improvements. Section 3.4 presents and compares the designs of multiple-accumulator CMACs against the single-accumulator CMACs from Chapter 2. Section 3.5 concludes this chapter.

3.1 Architecture overview

The architecture for the X part of an FX correlator that is considered in this work was introduced in [18] as Architecture 2 and is further detailed in [40]. The architecture is intended for an ASIC implementation. The internal structure of one processing unit is shown in Figure 3.1. The main components are the input memory, the CMAC array and the two processing buffers. The input memory holds $2NT$ data samples for all $2N$ input signals, for one frequency channel and for all T time instances that occur over the period of the integration time. The key parameters of this architecture are n and w. Every n input signals are grouped into

The content of this chapter appeared in a published article. Adapted by permission from Springer Nature: Springer Experimental Astronomy, "Using multiple-accumulator CMACs to improve efficiency of the X part of an input-buffered FX correlator", S. Lapshev and S. M. R. Hasan, doi: 10.1007/s10686-017-9527-4, ©2017.

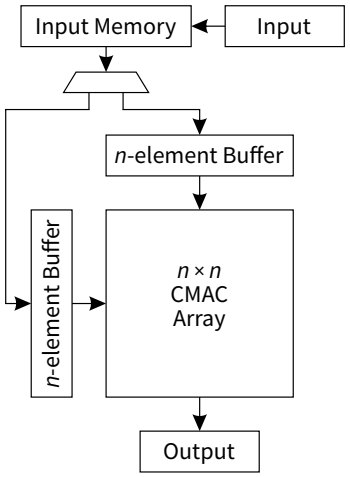


Figure 3.1: Simplified diagram of the internal structure of one processing unit of Architecture 2 in [18].

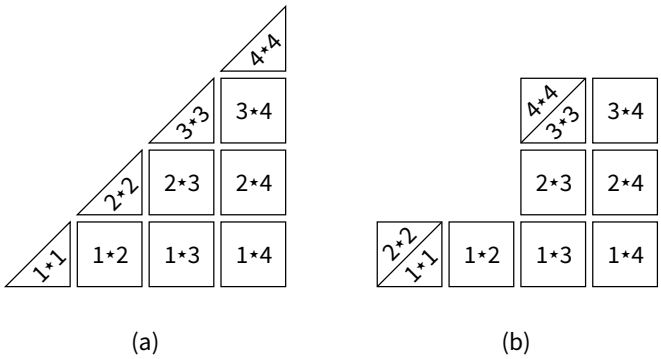


Figure 3.2: The example of how a correlation triangle is split into SIs for the case of $w = 4$. Numbers represent the four signal sets. (a) shows how a full integration is first split into sections. (b) shows the final pattern of SIs after auto-correlation sections are paired together into full SIs.

a signal set, which amounts to $w = \lceil 2N/n \rceil$ signal sets. Every n data samples read from or written to the memory for one signal set are referred to as a sample set in this work. Each of the two processing buffers (n-element buffers) holds one sample set. Each signal set is correlated with all w signal sets by using the $n \times n$ array of CMACs which can perform n^2 cross-correlations in parallel. The CMAC array integrates correlations for T processing cycles. Every integration performed by the CMAC array is called a sub-integration (SI). An example of how a full integration for all signals can be split into SIs is shown in Figure 3.2. Auto-correlation half-SIs are paired together because the CMAC array can be processing two such half-SIs at the same time. In this architecture all SIs are processed in the same chip and by the same CMAC array.

The downside of processing correlations in this way is that each data sample is read w times from the memory because it is used in the computation of w SIs. My work [41] introduced the idea of using two accumulators per CMAC to reduce the number of memory operations in this architecture by pairing SIs that share at least one signal set. This chapter generalises and expands on this approach.

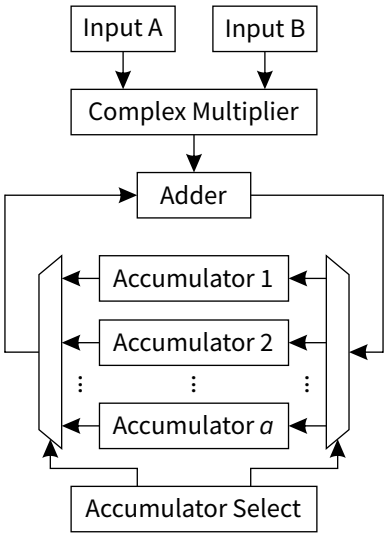


Figure 3.3: Architecture of one multiple-accumulator CMAC with a accumulators. The "Accumulator Select" circuit chooses which accumulators are read from and written into in the current processing cycle.

The following sections detail how multiple accumulators per CMAC can be used to further reduce the number of memory operations and achieve larger power savings.

3.2 Grouping SIs for multiple-accumulator CMACs

During integration of a particular SI, in every processing cycle a one-accumulator array reads two sample sets which belong to the same time instance. For every new processing cycle, two new sample sets are read from the memory for the same two signal sets but for a different time instance. The array cannot switch to processing a different SI until it finishes the current SI and it thus has to read the same data samples again for all other SIs that correlate the same two signal sets with all other signal sets. The multiple-accumulator array can keep intermediate accumulations for multiple SI and can thus switch between different SIs in every processing cycle. The architecture of one multiple-accumulator CMAC with a accumulators is shown in Figure 3.3. Regardless of the number of accumulators per CMAC, the CMAC array can only process one SI in each cycle. The reduction in the number of memory read operations is achieved when the same data samples that have been read for the current SI in one cycle are reused for a different SI in the next cycle. Grouping SIs enables fewer memory operations per processing operation without increasing or decreasing the number of processing cycles that need to be performed.

With a accumulators per CMAC, SIs need to be grouped into groups of a. Each group of SIs is a sequence of several SIs. Each SI in the sequence has to share at least one signal set with the following SI in the sequence. In this way, for every time instant of the input data samples, the multiple-accumulator array reads two sample sets for the first SI in the sequence and then sequentially processes all the other SIs in the group while updating at most one sample set in the processing buffers in every cycle.

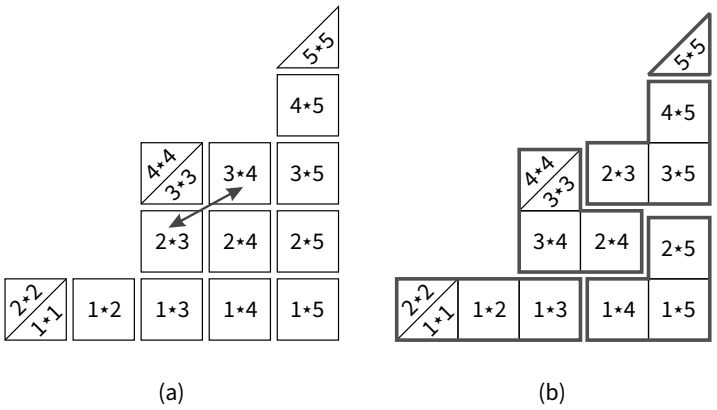


Figure 3.4: The illustration showing how (a) SIs for the case of $w = 5$ can be grouped into (b) groups of 3 following the rules in Section 3.2. The arrow indicates the swapping of SIs that is performed before grouping.

For every SI that is composed of two auto-correlation half-SIs there is the cross-correlation SI that processes the exact same signal sets. Every paired auto-correlation SI and its cross-correlation counterpart should be in the same group and should be processed in sequence. Then, no memory read operations are required for the processing of one such SI because the same sample sets would have already been read for the previous SI in the processing sequence. This is in contrast to the one-accumulator array where every processing cycle requires the update of the two sample sets in the processing buffers. Depending on the implementation, this has the potential advantage of making it easier to update the memory with new data samples for the next full integration.

Figure 3.4a shows a pattern of SIs and Figure 3.4b shows the example of grouping these into groups of 3. Essentially, all SIs are aligned in one sequence that is split into a new SI group after every a SIs. The sequence starts from the bottom left of the correlation triangle and continues row by row left-to-right for the odd rows and right-to-left for the even rows. In this sequence every paired auto-correlation SI is always followed by its cross-correlation counterpart, but depending on w and a the cross-correlation SI may need to be swapped with the SI preceding the paired auto-correlation SI for the two SI to be in the same group. After such swapping, every SI still shares at least one signal set with the following SI in the sequence. The swapping is done once in this example as is indicated in Figure 3.4a. The same grouping strategy applies to any valid combination of w and a. Numerous other ways of grouping SIs are possible and the most convenient way depends on the implementation.

Different architectures with a similar idea can be developed to use larger processing buffers that can each hold more than one sample set, in which case the grouping strategy can be different. If every memory read operation brings m data sample sets into the processing buffers each holding m sample sets as opposed to just one, the SIs can be grouped into groups of m^2. Then for every two memory read operations, the multiple-accumulator array with $a = m^2$ can perform m^2 cross-correlations compared to just m if a one-accumulator array is used. Thus, the multiple-accumulator array of size n is equivalent in input memory access efficiency to a one-accumulator array of size mn. A previous work [42] on a similar architecture with $m = 16$ and $a = 256$ reported this result.

3.3 Quantifying memory access improvements

The CMAC array has two data inputs for the two sample sets that it correlates in each cycle. In the case of the memory operating on one sample set per operation, every change of the array's inputs corresponds to one memory read operation. As per the grouping approach presented in the previous section, the average number of times the array's inputs change per $2N$ input data samples can be expressed as

$$R(a,w) = (a+1)A + (B+1)\operatorname{sgn} B - \left\lfloor \frac{w}{2} \right\rfloor \operatorname{sgn}(a-1) + (w \bmod 2)(1-\operatorname{sgn} B), \tag{3.1}$$

where

$$A = \left\lfloor \frac{\lfloor w^2/2 \rfloor}{a} \right\rfloor \tag{3.2}$$

and

$$B = \left\lfloor \frac{w^2}{2} \right\rfloor \bmod a. \tag{3.3}$$

In (3.1)–(3.3), a is the number of accumulators per CMAC and w is the total number of signal sets as per the definitions in previous sections; A is the number of full SI groups each with a SIs and B is the number of remaining SIs that form an incomplete group. In these relationships $\lfloor w^2/2 \rfloor$ is the total number of full SIs including the paired auto-correlation half-SIs. The number of these half-SIs is w which are paired to form $\lfloor w/2 \rfloor$ full SIs. There is an unpaired half-SI when w is odd.

The expression defining $R(a,w)$ in (3.1) is composed of several terms. The first term is the number of memory read operations normally needed for full groups: There are A such groups each normally contributing $(a+1)$ memory reads. The second term is the number of memory reads for the remaining B full SIs which normally account for $(B+1)$ memory read operations. When B is 0, the second term is also 0 rather than $(B+1)$. The third term takes into account the paired auto-correlation SIs and subtracts their number from the first and second terms. For this to be correct, the groups that include auto-correlation SIs also have to include the cross-correlation SIs that process the exact same signal sets, which reduces the number of required memory reads from $(a+1)$ and $(B+1)$ by 1 for every pair of auto-correlation and the corresponding cross-correlation SIs in the same group. The third term is 0 when a is 1, because in this case it is not possible to have the auto-correlation and the corresponding cross-correlation SIs in the same group. The fourth term takes into account the remaining unpaired auto-correlation half-SI by adding an additional memory operation only when w is odd and B is 0: When w is even, there is no unpaired half-SI; when w is odd and B is non-zero, the unpaired half-SI is included with the incomplete group that also includes the B full SIs. In the latter case, the signal set processed by the unpaired half-SI must be one of the signal sets processed by at least one of the B full SIs so that its processing does not require an additional memory operation.

The total number of data samples read from and written to the memory per one full integration (i.e., per $2NT$ data samples) is thus

$$M(n,T,w,a) = nT\,(R(a,w)+w). \tag{3.4}$$

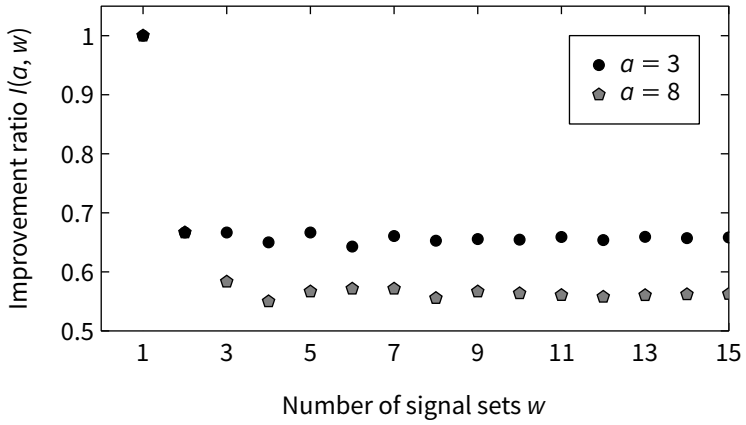


Figure 3.5: Plot of the improvement ratio I over the one-accumulator array against w for two cases of a.

Equation (3.4) generalises the required memory access per one full integration using the multiple-accumulator CMAC array. The value of $M(n, T, w, 2)$ represents both values derived in [41] for the two-accumulator array. The improvement ratio over the one-accumulator CMAC array can be expressed as

$$I(a, w) = \frac{M(n, T, w, a)}{M(n, T, w, 1)} = \frac{R(a, w) + w}{w^2 + w}. \tag{3.5}$$

The larger w is, the less $I(a, w)$ depends on w. For w larger than a, I can be approximated with

$$I \simeq \frac{a + 1}{2a}. \tag{3.6}$$

The improvement ratio represents the reduction in the number of input memory operations. Consequently, it also represents the reduction in the memory's dynamic power dissipation. The inverse of I may also represent the increase in performance for memory-limited implementations. Figure 3.5 shows the value of the improvement ratio I against w for two cases of a: the reduction in the number of memory operations is more than 30% and 40% when a is 3 and 8 respectively.

The case of the input memory operating on more than sample set per operation (i.e. when $m > 1$) is more complex. The specific quantification for such an architecture can be derived from equations (3.1)–(3.4) but depends on the implementation details such as the design of the processing buffers which would now be holding m sample sets each as opposed to just one. Continuing the example introduced in the previous section for the memory operating on m sample sets for m signal sets per operation, with $a = m^2$ the number of data samples that are read from and written to the memory per one full integration is $M(mn, T, \lceil 2N/(mn) \rceil, 1)$, which is equivalent to the reduction on the order of m when compared to $M(n, T, w, 1)$. With $2m$ sample sets read into the processing buffers with two operations, the buffers can present m^2 combinations of these sample sets to the CMAC array so that it can compute m^2 correlations. In this case each change of the array's inputs no longer necessarily corresponds to a new input memory operation. Applying the approach of grouping SIs to grouping the SI groups themselves, every two m^2 SI groups sharing m or $2m$ signal sets can be paired into larger $2m^2$ SI super-groups and processed with an array of $a = 2m^2$ achieving the memory sample rate of

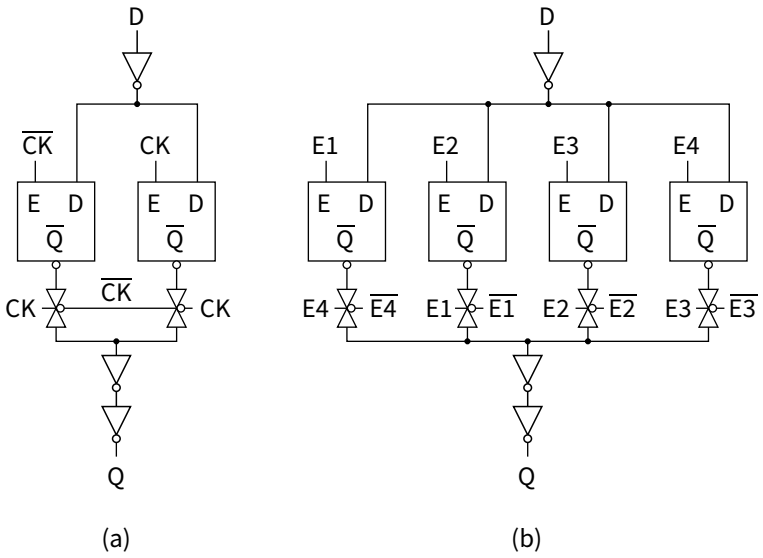


Figure 3.6: Comparison of the schematic diagrams of the two data storage circuits employed in the four CMAC designs: (a) standard Latch-MUX DET FF from Section 2.3.2 for one-accumulator designs and (b) custom 3-bit variant of the same circuit for three-accumulator designs. The 3-bit circuit is only twice the size of the 1-bit circuit. E1 through E4 are the locally-generated enable signals.

$M(mn, T, \lceil 2N/(mn) \rceil, 2)$. Depending on specific numbers, this would result in at least a further 25% reduction for a large number of such SI super-groups.

3.4 Design and evaluation of multiple-accumulator CMACs

It was estimated for a proposed ASIC implementation of a one-accumulator architecture in [40] that the input memory is responsible for about 20% of the total chip power with most of the rest (about 51%) being dissipated in the CMAC array. It is thus important to consider the impact of the multiple-accumulator approach on the CMAC array.

In addition to the two one-accumulator CMACs from Chapter 2, two more designs have been made to evaluate the multiple-accumulator approach in terms of its benefits and costs to performance parameters such as the energy dissipation, leakage, circuit area and circuit delay.

A naive design approach to multiple accumulators would be to simply use as many FFs as there are bits that need to be stored. Better designs are possible because the multiple values stored in the accumulator are never accessed at the same time. In general, the exact choice of design depends on the value of a and the implementation technology. Figure 3.6 shows the schematic diagrams of the static data storage circuits that are used in the CMACs designed for this evaluation. The one-accumulator storage uses a standard DET FF design (Figure 3.6a) as was explained in Section 2.3.2. The three-accumulator designs use the storage element shown in Figure 3.6b that is a variation of the DET FF circuit with the same Latch-MUX construction. These storage circuits use inverters, transmission gates and D latches. Compared to the standard FF that uses two D latches per 1 bit of storage, the multiple-accumulator storage element uses $(a+1)$ D latches to store a bits. Instead of using the clock signal directly, this storage element uses $(a+1)$

Table 3.1: Summary of the design information and simulation results of the 4-bit and 8-bit CMACs with one and three accumulators.

Parameter or Metric	4-bit CMACs		8-bit CMACs	
Temperature, °C	60			
Cycle time, ns	0.5		0.8	
Accumulator count	1	3	1	3
E_{total}, pJ	0.972	1.031	2.614	2.578
E_{CMUL}, pJ	0.400	0.347	1.619	1.373
E_{ACC}, pJ	0.572	0.684	0.995	1.205
I_{DDQ_total}, µA	88.8	96.5	228.8	243.1
I_{DDQ_ACC}, µA	43.5	51.2	60.3	74.6

locally-generated enable signals which sequentially switch in every clock cycle and only one of which is active at the same time. In this case the three-accumulator storage circuit is only twice the size of the standard one-accumulator FF which is a significant improvement over naive scaling.

Only the data storage circuits in the accumulators are different between the one- and three-accumulator CMACs of the same data wordlength. A specific implementation of the circuits shown in Figure 3.6 presents many trade-offs including the ones between energy and delay as is investigated for similar circuits in Chapter 4. In this case these circuits have been designed so that the timing performance of one- and three-accumulator CMACs is equivalent.

Per-cycle energy dissipation of the CMAC circuits was measured precisely using post-layout analog simulations on netlists that include the extracted RC parasitics. The simulated energies include the energy costs of local clock distribution. Input data pattern was generated pseudo-randomly with each bit having a 35% probability of switching in every cycle. Simulations were carried out for the typical process corner with the temperature set to 60 °C. Leakage currents were simulated using the dedicated IDDQ models provided with the technology design kit. Table 1 summarizes the relevant design information and simulation results of the four CMAC designs. In the table E_{total} is the total average per-cycle energy dissipation of a CMAC circuit; E_{CMUL} is part of E_{total} that is dissipated in the multiplier part of the CMAC; E_{ACC} is the difference between E_{total} and E_{CMUL} that represents the per-cycle energy dissipation of the accumulator part of the CMAC including the adder performing accumulations; I_{DDQ_total} is the total leakage current of the CMAC and I_{DDQ_ACC} is that of the accumulator part of the circuit including the adder.

Because the high-speed adders dominate the circuit area of the CMACs' accumulator parts, the impact of using three accumulators compared to just one is small in the metrics of area and leakage. The increase in the leakage is about 8.7% and 6.3% overall and about 18% and 24% for only the accumulator parts in the 4-bit and 8-bit cases respectively. The increase in circuit area is about 5.3% for the 8-bit case. For the 4-bit case, the designed one-accumulator layout is not as area-optimized as the other layouts and only a 5.5% reduction in area was achieved when compared to the three-accumulator design.

It has been found that the multiple-accumulator approach decreases E_{CMUL} and increases E_{ACC}. The reduction in E_{CMUL} is due to reduced switching activity within the circuit when at least one of the processed data sample sets stays unchanged

between clock cycles. The increase in E_{ACC} has been found to be mostly due to increased switching activity in the adder and to a lesser extent due to a more complex data storage circuit. The increase in the adder is due to processing of accumulations for different baselines in different clock cycles.

The reduction in E_{CMUL} has been found to outweigh the increase in E_{ACC} for the 8-bit CMACs. For the 4-bit CMACs the net effect has been found to be an increase in the overall energy dissipation. E_{total} is decreased by 1.4% for the 8-bit case and is increased by 6.1% for the 4-bit case. Larger overall power savings can be expected for larger wordlengths as the multiplier becomes more significant than the accumulator part.

Considering only multipliers, the reduction in the switching activity results in the decrease of E_{CMUL} by 13% and 15% in the 4-bit and 8-bit cases respectively. For the processing cycles when one of the data inputs stays unchanged between clock cycles, the decrease is about 20% and 25% for the 4-bit and 8-bit CMACs respectively. This presents an incentive of using a larger number of accumulators per CMAC to increase the number of such cycles and decrease the average power dissipation.

Increasing a results in diminishing returns for the input memory savings as per (3.5) and (3.6). If only memory savings are considered, there may be little point in values of a greater than 4. On the other hand, using larger a is shown to decrease the power dissipation of the CMAC array which could be dominating the overall power dissipation as discussed in the beginning of this section.

3.5 Conclusion

The approach of grouping sub-integrations for efficient processing by the multiple-accumulator CMAC array has been presented. With such CMACs being able to switch in every cycle between different SIs sharing some data inputs, significant reduction in the number of memory read operations is achieved for the same number of processing operations and the same number of CMACs. With only three accumulators per CMAC, the total number of memory operations and thus the memory power is reduced by more than a third for an arbitrarily large number of input signals when compared to a one-accumulator CMAC array.

In addition to the two CMACs presented in Chapter 2, two more CMACs with multiple accumulators have been designed and evaluated. The multiple-accumulator approach has been found to result in the 1.4% decrease in the overall CMAC power dissipation in the case of three-accumulator 8-bit CMACs, which is in addition to a more than 30% reduction in the memory. Larger power savings can be expected for larger input data wordlengths and a larger number of accumulators.

Chapter 4

DET flip-flops based on C-elements

CMAC designs of Chapter 2 used of dual-edge-triggered storage elements which then had to be adapted for the multiple-accumulator approach of Chapter 3. This has led to the effort of creating better circuit structures for DET storage elements. Compared to the more common single-edge-triggered designs, fewer DET structures have been presented in literature, which gives more opportunity for creating novel designs.

The common DET flip-flop design that has been used throughout the CMAC circuits is the Latch-MUX flip-flop [34], [37] which consists of two latches whose outputs are multiplexed to one output. The two latches are level-triggered by opposite clock levels so that there is always a transparent latch that follows every change at the input. As a result of this transparency, glitches at the input have an adverse effect on the flip-flop's power dissipation. Glitches often happen in circuits such as multipliers that have deep combinational logic. It was estimated in [35] that Latch-MUX DET flip-flops dissipate less power than SET flip-flops only when glitches are rare. Other DET flip-flop designs include pulsed DET flip-flops [34], [43], [38]. Generally, a pulsed DET flip-flop works by making its output latch transparent to the input signal after every clock edge for a short time interval. Power dissipation of such flip-flops is less dependent on input signal changes at the cost of increased power dissipation due to clock activity.

This chapter presents new static DET flip-flop circuits that use C-elements. A C-element, introduced in [44], is normally a three-terminal device with two inputs and one output. When all of its inputs are the same, the output switches to the value of the inputs; when the inputs are different, the previous output value is preserved. This device acts as a latch which can be set and reset with combinations of signal levels at the input. Figure 4.1 shows the two transistor-level implementations of C-elements that are used in this work. C-elements and variations of their circuit topologies are the building blocks of the new DET flip-flops presented below.

The chapter consists of seven sections. Sections 4.1 to 4.4 present five novel DET designs including the low-glitch LG_C flip-flop, implicit-pulsed IP_C flip-flop, floating-node FN_C flip-flop and two high-performance conditional-toggle CT_C and CTF_C flip-flops. Section 4.5 describes simulation setup and the comparison methodology that is used to compare the presented flip-flops against each other and against six previously reported DET flip-flop designs. Section 4.6 presents and

The content of this chapter appeared in a published article. ©2016 IEEE. Reprinted, with permission, from S. Lapshev and S. M. R. Hasan, "New Low Glitch and Low Power DET Flip-Flops Using Multiple C-Elements", IEEE Transactions on Circuits and Systems I: Regular Papers, doi: 10.1109/TCSI.2016.2587282, 10/2016.

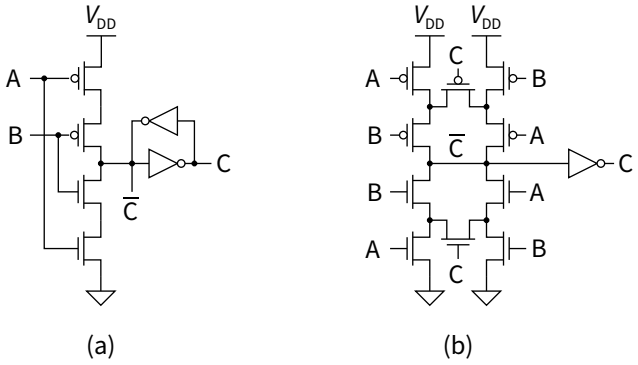


Figure 4.1: Transistor-level implementations of a C-element that are used in this work: (a) the weak-feedback and (b) the symmetric [45] implementations.

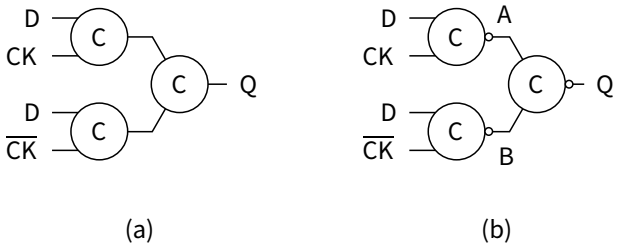


Figure 4.2: Gate-level schematic of the new LG_C DET flip-flop using (a) non-inverting and (b) inverting C-elements.

discusses the results of extensive Monte Carlo and voltage scaling simulations. The comparison includes metrics of energy and power delay products and also additional metrics of energy dissipation due to glitches, where the presented flip-flops excel. Finally, Section 4.7 concludes the work presented in this chapter.

4.1 Low-glitch LG_C flip-flop

The new low-glitch LG_C DET flip-flop consists of three C-elements that are connected as shown in Figure 4.2. Although both inverting and non-inverting C-elements can be used for this DET flip-flop, only the inverting topology shown in Figure 4.2b is used due to the transistor-level implementations in Figure 4.1 being faster in the inverting configuration. The logic waveforms depicting the operation of the LG_C flip-flop are shown in Figure 4.3. This flip-flop has two internal latches A and B and an output latch Q belonging to the three C-elements. The output latch switches only when the two internal nodes A and B are at the same signal level. The latches at A and B can only switch to $\overline{CK}$ and CK when D becomes equal to CK and $\overline{CK}$ respectively. Thus, in between clock transitions, at least one of A or B is latching $\overline{D}$. And once there is a transition in the clock signal, the internal node that is not at $\overline{D}$ will toggle to have both A and B at $\overline{D}$, which results in the output C-element switching Q to D.

The LG_C flip-flop bears some resemblance to Latch-MUX designs and, in particular, the LM_C flip-flop presented in [46], yet its operation is different. The diagram of a generic Latch-MUX flip-flop and its operational waveforms are shown in Figures 4.4 and 4.5. Figure 4.4 shows that in a Latch-MUX flip-flop every change

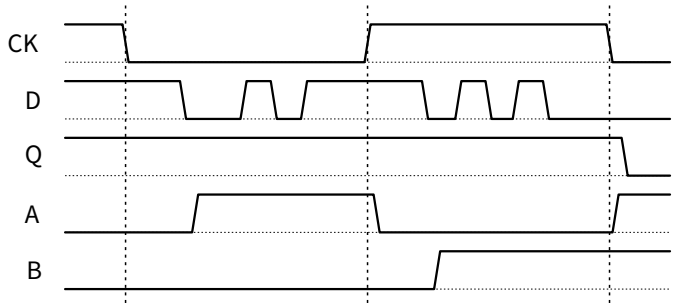


Figure 4.3: Operational waveforms showing the behaviour of the LG_C flip-flop.

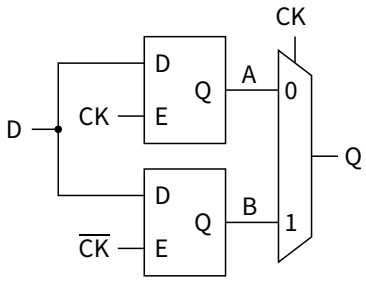


Figure 4.4: Schematic diagram of a generic Latch-MUX flip-flop.

at D leads to switching in one of the two latches. In the novel LG_C flip-flop, whose operational waveforms were shown in Figure 4.3 for the same CK and D signal patterns, C-elements are used to reduce switching activity due to input signal transitions so that none of the flip-flop's latches follow the input signal at any point during the operation. The value of D only determines which signal level latches A and B switch to at the next clock transition. No matter how many times input D toggles in between the clock edges, LG_Cs internal state only changes once with the first signal change at the input. This leads to much lower power dissipation in the presence of glitches at the flip-flop's input, which is evaluated using simulation in Section IV.

One of the features of the LG_C flip-flop is its immunity to overlap between CK and $\overline{\text{CK}}$ signals. A change at the flip-flop's output is triggered by a change at one of A or B which happens due to a signal transition in either CK or $\overline{\text{CK}}$ respectively. That is, every change at the output is triggered by a transition in only one of CK and $\overline{\text{CK}}$ signals, not both. Thus, although clock overlap has an impact on the timing

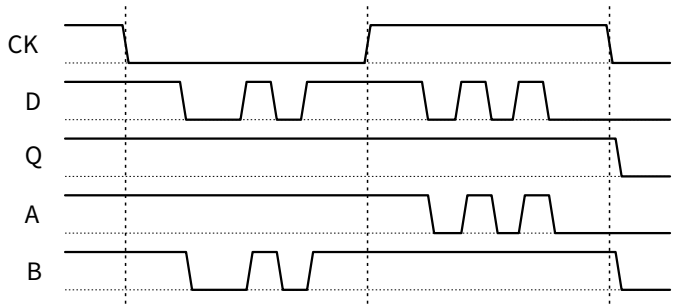


Figure 4.5: Operational waveforms for a generic Latch-MUX flip-flop.

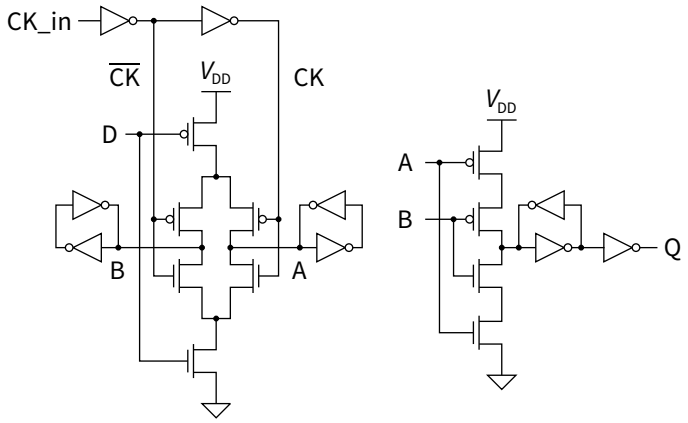


Figure 4.6: Proposed transistor-level design of the LG_C flip-flop based on weak-feedback C-elements shown in Figure 4.1a.

of output transitions, it cannot cause the flip-flop to fail in latching a correct value. While there are many implementations of C-elements that can be used in the LG_C flip-flop, this work analyses the design shown in Figure 4.6. This design is based on the weak-feedback C-element of Figure 4.1a. The two C-elements at the input are merged by sharing transistors connected to the D input. This configuration leads to low input loading and a lower transistor count than what is otherwise possible.

4.2 Implicit-pulsed IP_C flip-flop

The LG_C flip-flop can be modified to reduce the total dynamic power dissipation at the expense of somewhat increased power dissipation due to clock signal transitions. Since there is never a moment when D is neither CK nor $\overline{CK}$, only one latch at either A or B is required at any given moment while the other latch plays no role other than to increase switching power dissipation. The purpose of latches at A and B is to be able to keep these nodes at $\overline{CK}$ and CK respectively and to allow them to be switched to $\overline{D}$ at clock transitions. This behaviour can be achieved without latches.

Figures 4.7 and 4.8 show the transistor-level and gate-level schematic diagrams of the implicit-pulsed IP_C flip-flop. In the transistor-level schematic, two additional weak C-elements (inner C-elements) are merged with the two input C-elements by sharing clocked transistors. The advantages of the IP_C design over the LG_C design are reduced delays and reduced switching power dissipation as is shown in Section IV. The operation of the IP_C flip-flop is illustrated using logic waveforms shown in Figure 4.9. Although this design may bear superficial resemblance to a common Latch-MUX flip-flop in [47], its circuit operation is quite different. The IP_C design has no static storage latches at nodes A and B. The input C-elements are essentially inverters for the CK and $\overline{CK}$ signals respectively. The D input influences the timing of these inversions in such a way that whenever there is a change in the CK signal, one of A or B nodes that is not at $\overline{D}$ toggles using strong transistors of an input C-element. This event then makes the other node toggle using weaker transistors of an inner C-element. Compared to the LG_C flip-flop, the C-elements in the IP_C flip-flop circuit have no weak feedback to overpower.

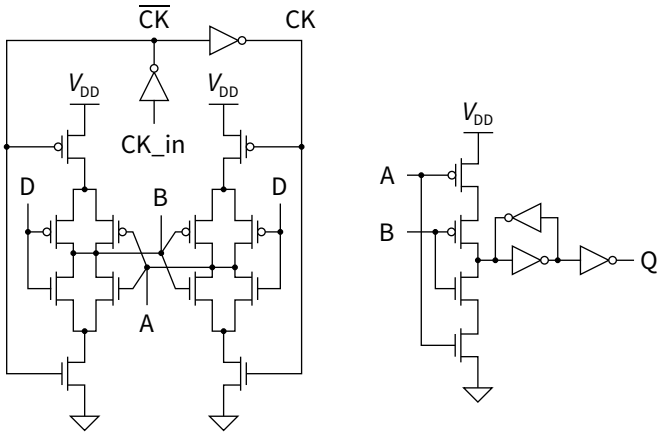


Figure 4.7: Transistor-level schematic diagram of the implicit-pulsed IP_C DET flip-flop.

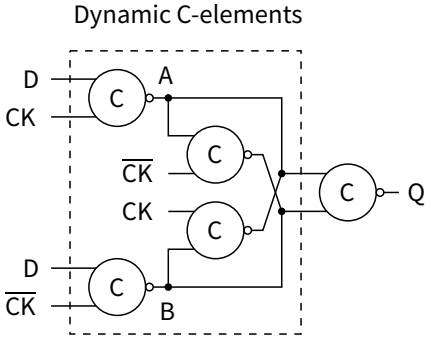


Figure 4.8: Gate-level schematic diagram of the implicit-pulsed IP_C DET flip-flop.

The two additional inner C-elements in the IP_C flip-flop are needed to reinforce the voltage levels at nodes A and B in order to make the circuit static. Considering the flip-flop's schematic diagrams, only the output C-element uses a latch. All the other C-elements do not have latches at their outputs and can thus be considered dynamic. However, the overall circuit is static due to the cross-connection of the inner C-elements reinforcing the signal levels at nodes A and B. The IP_C flip-flop works by creating a brief moment after every clock edge when both A and B are at $\overline{\text{D}}$, which results in the output switching to D. Thus, this design belongs to the category of implicit-pulsed DET flip-flops.

4.3 Floating-node FN_C flip-flop

In the IP_C flip-flop, signal levels at A and B toggle after every clock transition regardless of D and Q, which leads to higher power dissipation due to clock signal transitions. Figures 4.10 and 4.11 show the transistor-level and gate-level schematic diagrams of the improved FN_C flip-flop that solves this issue. In the case of the LG_C, IP_C and FN_C flip-flops, for the output C-element to work correctly, the requirements for the flip-flop's input stage are as follows: in between the clock edges, at least one of A and B has to be kept at $\overline{\text{Q}}$ to avoid flipping at the output, and once the clock toggles, both nodes have to be at $\overline{\text{D}}$ for the output to switch to D. As long as one of A and B is kept at $\overline{\text{Q}}$, the signal level at the other node

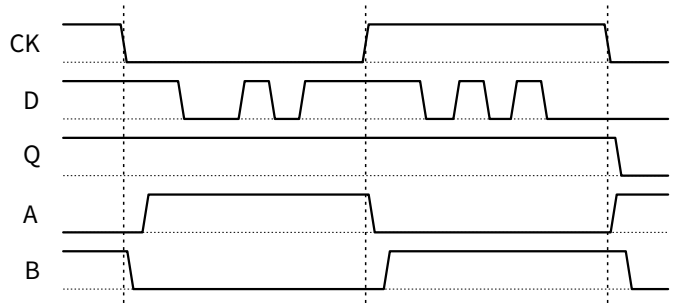


Figure 4.9: Logic waveforms showing the behaviour of the IP_C DET flip-flop.

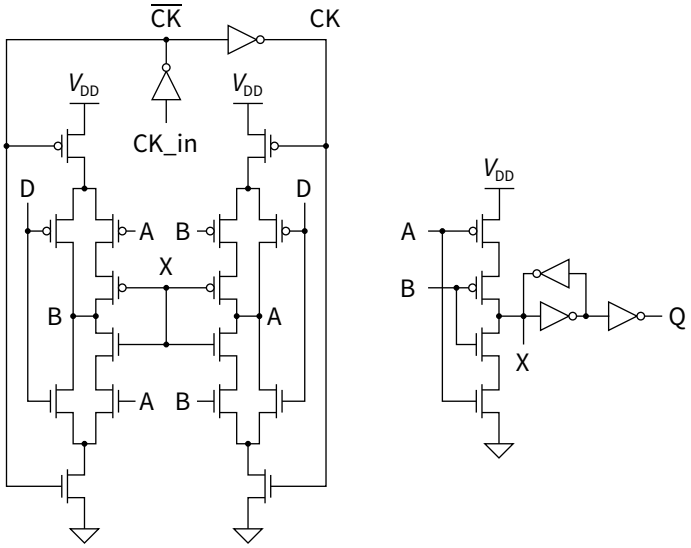


Figure 4.10: Transistor-level diagram of the improved floating-node FN_C DET flip-flop that uses 5 C-elements including weak devices for the inner C-elements.

is irrelevant for the correct operation of the output stage. With implementations of C-elements shown in Figure 4.1, one of A and B that is not kept at $\overline{Q}$ can be at any voltage level without affecting the output. The LG_C flip-flop satisfies these requirements by employing two independent latches at A and B, one of which is always redundant (which one depends on D and CK). The implicit-pulsed IP_C flip-flop ensures correct operation through the cross-connection of the weak inner C-elements. This cross-coupling ensures that nodes A and B are at opposite signal levels and that one of them is at $\overline{Q}$. The improved floating-node FN_C flip-flop design does not cater for the node not at $\overline{Q}$ in between clock transitions and does not reinforce its signal level. This behaviour is implemented through the feedback of X, which Q follows, into the inner cross-coupled weak C-elements. If the signal combination of D and CK is such that the $\overline{Q}$ level at one of A or B is maintained by strong transistors of an input C-element, the other node is left floating. This does not compromise the overall static behaviour of the circuit, since as it was mentioned above, the signal level at the floating node cannot affect the output as long as the other node is at $\overline{Q}$. Compared to the IP_C flip-flop, this flip-flop's power dissipation due to clock signal transitions is reduced without increasing power dissipation due to input switching.

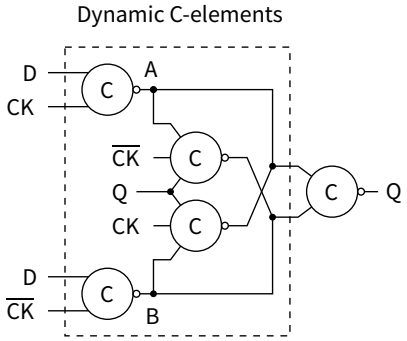


Figure 4.11: Gate-level schematic of the FN_C DET flip-flop shown in Figure 4.8 with 2 inner 3-input weak C-elements exhibiting a floating-node behaviour.

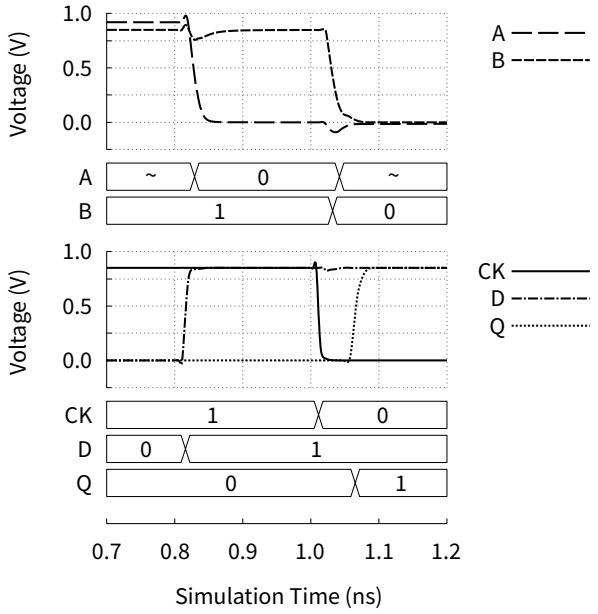


Figure 4.12: Simulated signal levels of the implemented FN_C flip-flop. Floating states are denoted as "~".

Figure 4.12 shows the FN_C's simulated signal levels for one clock transition. The flip-flop was designed in the 28 nm GF 28HPP CMOS technology. Initially in the simulation, Q is 0, CK is 1 and D is 0. Since D is equal to $\overline{CK}$, B is kept at 1 with strong transistors of its input C-element, which ensures that X (and Q that follows X) stays at 0. As a result, node A is floating. Due to the switching of parasitic capacitances, the voltage level at node A is slightly higher than the technology V_{DD} voltage of 0.85 V. Before the next clock edge, input D switches to 1. Since D is now equal to CK, A switches to 0. Now $\overline{CK}$, A and X are 0, which keeps node B at 1 through the weak transistor branch connected to its input C-element. The next clock edge comes when CK switches to 0 and $\overline{CK}$ switches to 1. Since $\overline{CK}$ is now equal to D, node B switches to 0. Both A and B are now at 0, which causes the output C-element to switch Q to 1. Node A is floating again and B is kept at 0, which ensures that Q stays at 1 in the new cycle. In this example, the combinations of D and CK are such that only node A is seen floating. If either D or CK were inverted, node B would have been the floating node.

Because of the switching of parasitic capacitances before floating node events, floating voltage levels at A and B can be above the V_{DD} voltage of 0.85 V or below 0 V. Analog simulations indicate that the increase above V_{DD} and the decrease below 0 V do not exceed 10% of the nominal V_{DD} and that no significant dynamic current is flowing through the transistors' channels during this time. Thus, floating node events do not affect circuit reliability.

4.4 Conditional-toggle CT_C and CTF_C flip-flops

The novel conditional-toggle CT_C DET flip-flop design is shown in Figure 4.13. The CT_C flip-flop circuit uses only 20 transistors including transistors for the input, output and clock buffering. The flip-flop consists of a dynamic C-element at the output and a latch that provides static behavior to the circuit. The distinguishing feature of the CT_C flip-flop is that the state of its latch does not change when the flip-flop's output switches after a clock transition, which leads to low switching energy dissipation. The circuit for the output C-element is based on the weak-feedback implementation shown in Figure 4.1a but with the feedback inverter eliminated. The inputs to it are input D and the signal that mirrors Q in between clock transitions. When D is equal to Q, the C-element keeps its inverted output X at the level of $\overline{Q}$. When D is not equal to Q, the $\overline{Q}$ level at X is kept by the latch. The latch part of the circuit is responsible for toggling the signal level at X after clock transitions and for keeping it at $\overline{Q}$ in between clock transitions when D is not equal to Q. The latch part consists of two inverters connected back-to-back and a bi-directional 2-to-1 multiplexer with its output connected to node X. The operation of the CT_C flip-flop is illustrated using simulated voltage traces shown in Figure 4.14. The clock signal makes the multiplexer alternate between the two ends of the latch A and B after every clock transition. In between clock transitions, the end of the latch that is connected to X is at $\overline{Q}$. After a clock transition, the multiplexer switches to the opposite end of the latch, which makes the signal levels at X and Q toggle if D was not equal to Q before the clock edge. If D is equal to Q when the clock edge arrives, it is the latch that toggles its stored value and not node X, because the C-element forces node X to the level of $\overline{D}$. Toggling at the output is not done by changing the value stored by the latch but rather by multiplexing to the inverse of it, which achieves low switching activity within the flip-flop. The latch toggles its stored value after a clock edge only when D is equal to Q.

Implementation of the CT_C flip-flop presents trade-offs between power dissipation at high and low switching activities and circuit delay. Strong inverters in the latch make the output switching faster with little impact on the switching energy. However, the stronger these inverters are, the more energy it takes to change the state of the latch, i.e., the higher the power dissipation at low switching activity becomes. Figure 4.15 shows the CTF_C flip-flop, a modification of the CT_C flip-flop, that relaxes these trade-offs.

During operation one of A or B nodes is at D while the other is at $\overline{D}$. In the CTF_C flip-flop, the strengths of signal levels at these nodes depend not only on transistor sizing but also on D: Whichever node is at $\overline{D}$ is kept strongly while the other node is kept weakly with always-on transistors. If at a clock transition D is not equal to Q, node X is multiplexed to a strong $\overline{D}$ that quickly changes X to $\overline{D}$ and Q to D. When D is equal to Q, logic level at node X is kept strongly at $\overline{D}$ by

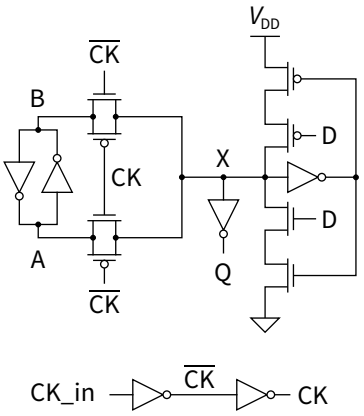


Figure 4.13: Transistor-level schematic diagram of the conditional-toggle CT_C DET flip-flop.

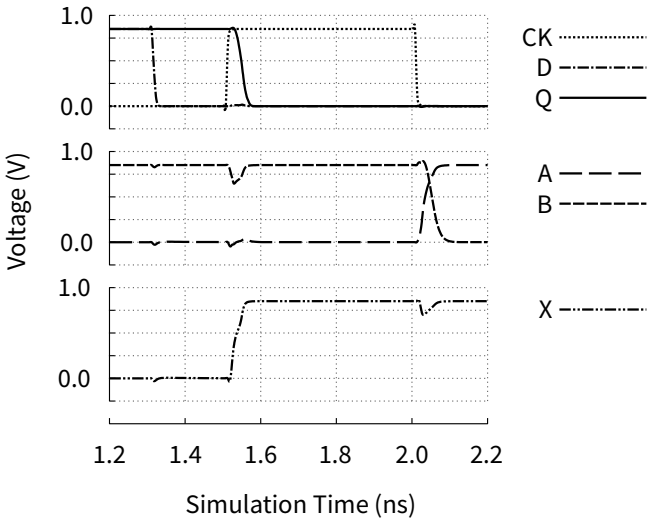


Figure 4.14: Simulated signal levels of the CT_C flip-flop implemented in the GF 28HPP technology.

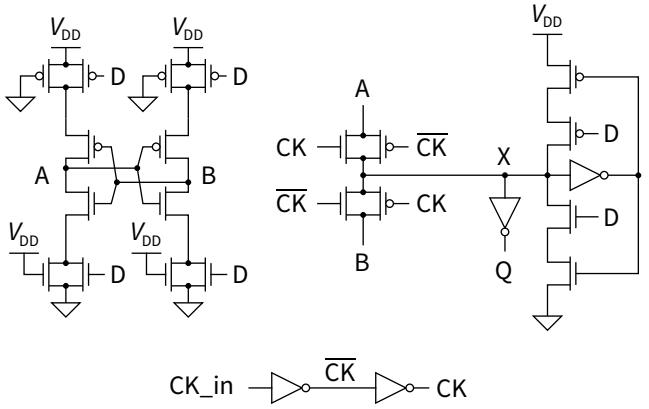


Figure 4.15: Transistor-level schematic diagram of the improved conditional-toggle CTF_C DET flip-flop.

the C-element. The next clock transition multiplexes node X to a weak D which the C-element overpowers quickly and at a low energy cost. Thus, CTF_C achieves faster switching compared to CT_C and reduces energy dissipation of idle cycles.

4.5 Simulation methodology

Extensive simulations have been performed to compare the five presented DET flip-flops against each other and also against six previously reported DET flip-flop designs. Two versions of the novel FN_C flip-flop have been considered: the version presented in Figure 4.10 and the version with the symmetric C-element of Figure 4.1b replacing the weak-feedback output C-element. The latter version is denoted as FN_C (sym) in the comparison. For a fair comparison, all flip-flops include input, output and clock buffering.

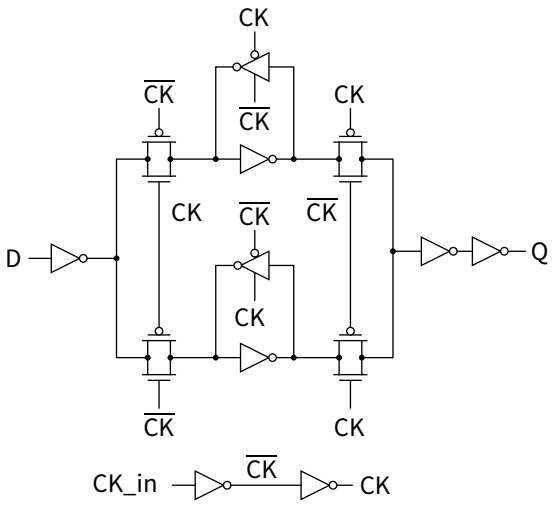
Figure 4.17 shows transistor-level schematic diagrams of the six previously reported DET flip-flop designs that are considered in this work for comparison. The designs are as follows:

(a) LM, described in [37], is a variant of the common Latch-MUX DET design;
(b) EP is a variant of the common Explicit-Pulsed DET flip-flop in [38];
(c) LM_C is a Latch-MUX design, introduced in [46], that uses a C-element at the output to perform the function of a MUX;
(d) TSP, presented in [48], is the True-Single-Phase Clock DET flip-flop design that follows the Latch-MUX approach but does not use the inverted clock;
(e) CP, introduced in [34], is the Conditional Precharge DET flip-flop;
(f) IP, described in [43], is the Implicit-Pulsed DET flip-flop.

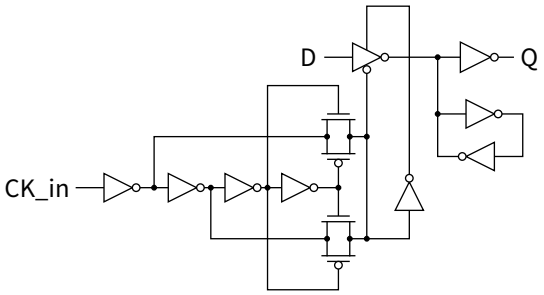
The flip-flops were implemented in the same 28 nm GF 28HPP CMOS technology. Implementations were optimised for minimum energy-delay product. For the optimisation step, the delay metric was the maximum CK–Q delay because it is straightforward to measure. Optimisations were performed by the simulation tool in an automated fashion: The tool varied transistor sizes within the specified bounds and chose the best sizes for each flip-flop after a number of iterations. The search bounds were chosen so that resultant designs would meet recommended design rules most of the time. Weak transistors were allowed to use minimum width rather than the recommended minimum width as it would otherwise result in poor circuit performance.

Simulations were performed on schematic designs. Conservative estimates of layout parasitics were included in the simulation models at both the optimisation and final simulation stages. These estimates were provided by one of the features of the design kit: The kit can automatically include its own estimation of the RC parasitic interconnect network into schematic simulation models. Parasitic extraction and post-layout simulations were also performed on selected designs and were compared to schematic simulations that used automatic estimation of parasitics. Post-layout simulations showed that the kit's estimates for small designs are often conservative and that compact circuits often perform slightly faster in post-layout simulations than in schematic simulations with the automatic parasitic network estimation turned on.

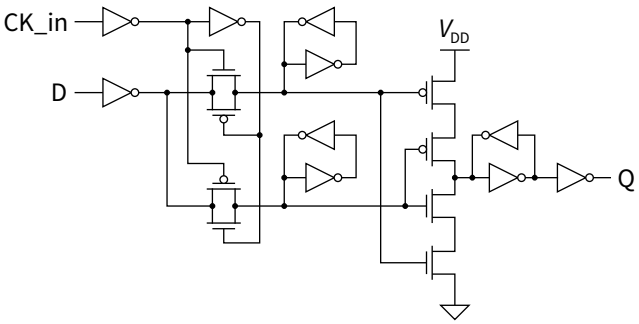
The simulation test bench that is used in this comparison is very similar to the ones used in [43], [49], [50]. The Q output of a simulated flip-flop is connected to a load of four symmetric inverters with their n-type transistors sized at minimum



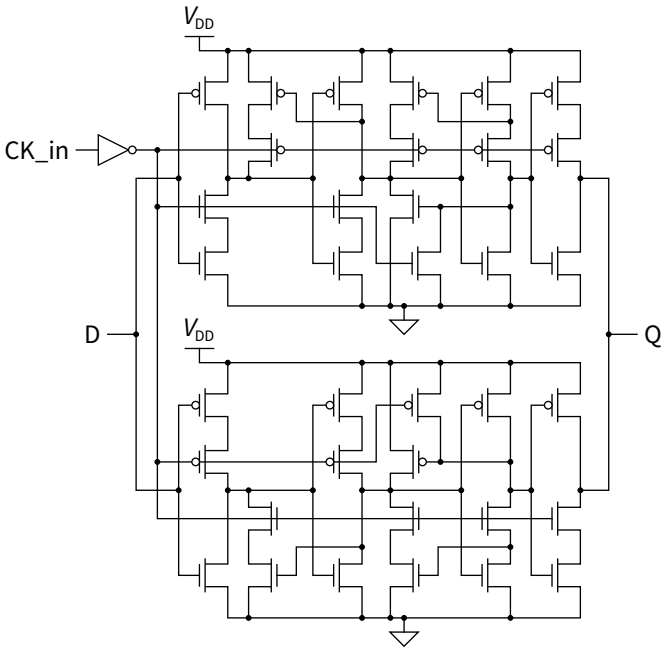
(a) LM



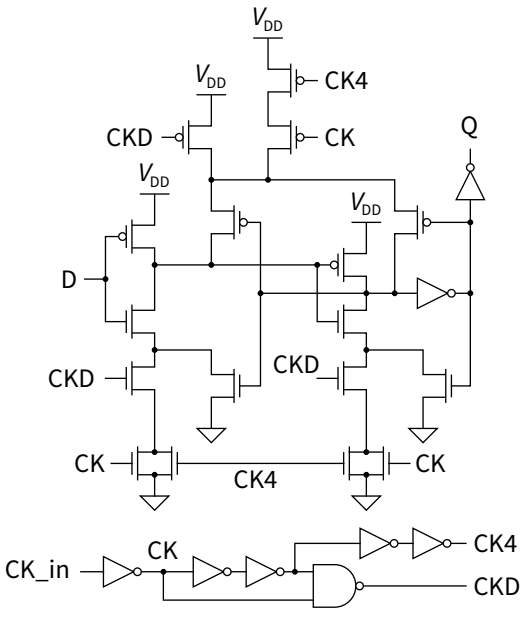
(b) EP



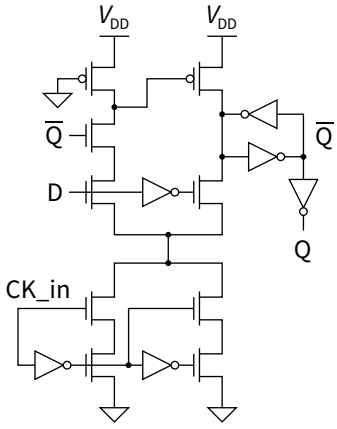
(c) LM_C



(d) TSP



(e) CP



(f) IP

Figure 4.17: Transistor-level schematic diagrams of the six previous DET flip-flop designs that are considered in this work for comparison with the new DET flip-flops. All circuits include input, output and clock buffering. The flip-flops are (a) LM [37], (b) EP [38], (c) LM_C [46], (d) TSP [48], (e) CP [34] and (f) IP [43].

recommended width. The generated data and clock signals are connected to the flip-flop's inputs through two inverters. The clock frequency is 1 GHz, which results in a 0.5 ns cycle time.

Most of the measurements relating to energy and delays were taken from Monte Carlo simulations with all process variations enabled. 2000 MC points were simulated for each flip-flop. The simulation temperature was set to 70 °C. Variation for a number of metrics is reported as coefficient of variation (CV). The CV is also known as the relative standard deviation (RSD) which is defined as the ratio between the standard deviation (SD) and mean. In this technology, simulation models make conservative assumptions about sources of variation when performing MC analysis on schematic designs. Variations in physical implementations are expected to be lower than what is reported from these simulations. The following parameters were evaluated from MC simulations:

- Power at 10%, 50% and 100% switching activities $P_{0.1}$, $P_{0.5}$ and P_1 respectively;
- Power-delay products $\mathrm{PDP}_{0.1}$, $\mathrm{PDP}_{0.5}$ and PDP_1 for each of the three power values where the delay is the D–Q delay;
- Maximum CK–Q delay t_{cq};
- Worst-case minimum D–Q delay t_{dq}.

The power is measured from the calculated $E(t)$ curve of the total dissipated energy versus the simulation time. This curve is calculated by integrating simulated power supply, data input and clock input currents for each simulated flip-flop in the following way:

$$E(t) = \int \left(I_{\mathrm{DD}} + \frac{1}{2} \left(I_{\mathrm{D}} + |I_{\mathrm{D}}| + I_{\mathrm{CK\_in}} + \left| I_{\mathrm{CK\_in}} \right| \right) \right) \mathrm{d}t. \tag{4.1}$$

In (4.1), the power supply current I_{DD} is integrated along with the positive currents flowing into the flip-flop's D and CK_in inputs. Currents flowing out of the flip-flop's inputs are discarded. These negative currents are either the result of an internal weak-feedback inverter working against the D input driver, in which case the current is supplied by the flip-flop and is thus already included in I_{DD}, or the result of a driver sinking the voltage at the input's parasitic capacitance, in which case this energy was already accounted for when the driver previously charged this capacitance with a positive current.

Part of the measured energy is dissipated outside of the flip-flop's circuit. This includes energies dissipated by the input drivers on driving the flip-flop's inputs and the energy dissipated by the flip-flop on driving the output load. Although the latter depends solely on the size of the load, the comparison is fair in that the load is the same for all flip-flops.

The CK–Q delay is measured as the maximum delay between the CK and Q transitions for the four possible combinations of transitions of CK and Q. For each case, the D transition happens sufficiently early so as not to affect the timing of the Q transition.

The D–Q delay of a flip-flop is generally considered to be a more important metric than just the CK–Q delay [51]: some flip-flops (e.g. the EP design) allow input changes to be captured well after a clock transition whereas others do not. In this sense, the D–Q delay is the time the flip-flop takes out of the clock cycle, which is the reason why the D–Q delay is used for PDP calculations in this work.

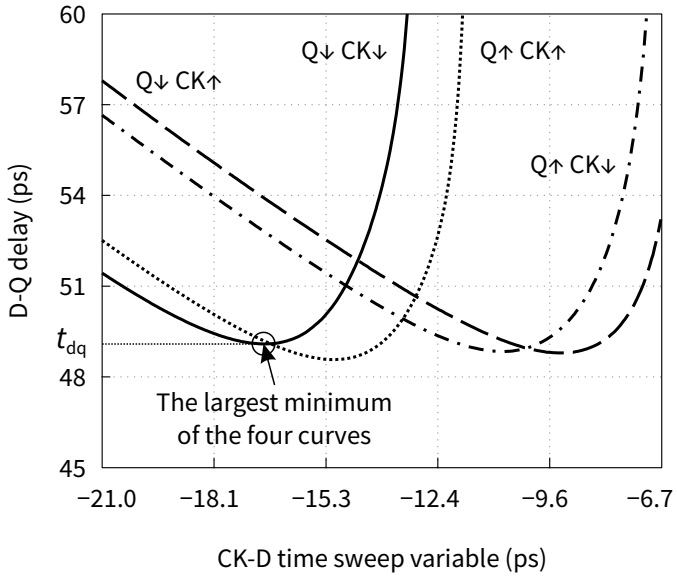


Figure 4.18: Illustration of the procedure for measuring the worst-case minimum D–Q delay. This plot is for a particular Monte Carlo point of the LM flip-flop. The curves are for the four cases of CK and Q transitions. The worst-case minimum D–Q delay is marked on the plot and also on the y-axis as t_{dq}.

For every MC point, multiple simulations were performed in order to measure the worst-case minimum D–Q delay for that point. There are four possible cases of CK and Q transitions. For each case, a parametric sweep is run with the sweep variable being the timing of the D transition relative to the CK transition. The minimum D–Q delay is found for each of the four cases. For every MC point, the worst-case minimum D–Q delay is then the maximum of these four minimums. Figure 4.18 illustrates this procedure for a particular MC point of the LM flip-flop. For illustration purposes, the step in the D–CK transition sweep variable was set to 0.04 ps. In final simulations, the step was set to 1.25 ps to reduce the number of simulations. For this example, the difference in delays measured with 0.04 ps and 1.25 ps resolutions of the D–CK sweep variable is less than 0.06 ps. Such a small difference is due to the steepness of D–Q vs D–CK curves reducing to 0 around their minimums.

Independent simulations were performed to measure the hold times t_h of the flip-flops for the typical process corner. For each flip-flop, for all four cases of CK and Q transitions, t_h was measured as the minimum amount of time D needs to be unchanged after a clock transition so as not to result in the flip-flop failing to latch the correct value, increase the CK–Q delay by more than 40% percent, or result in a glitch of more than half the V_{DD} voltage. Timings of D transitions were swept with a 0.05 ps step to record the hold time values with a 0.1 ps precision.

A separate set of simulations were performed to assess the impact of glitches on the flip-flops' power dissipation. In these simulations, the Q output is steady across clock cycles. The Q-to-$\overline{\text{Q}}$ and $\overline{\text{Q}}$-to-Q transitions for one glitch are introduced at the D input in between clock transitions. The total dissipated energy is then recorded after the next clock transition including the energy of restoration of the flip-flop's internal states. Energy dissipation due to one and three glitches is measured. The

energies are denoted as G_1 and G_3 respectively. The recorded numbers are averages for the four possible cases of CK and Q.

All energy measurements include leakage currents. More precise leakage simulations were performed using dedicated IDDQ models that are provided with the technology kit. The result is reported as I_{DDQ}. The results are averaged across eight cases of D, Q and CK. The reported values include leakage through D and CK_in inputs and exclude the leakage through the Q output.

Voltage scaling simulations were performed to assess the impact of supply voltage on the CK–Q delay. In these simulations, the supply voltage is scaled from 110% to 60% of the nominal V_{DD} of 0.85 V. Only the new flip-flops and two previous LM and EP designs are evaluated in voltage scaling simulations.

Simulations of flip-flops at various temperatures have been performed. Temperature variations have been found to have a negligible effect on the performance of any flip-flop in any metric except the leakage current. This is consistent with findings in [52].

4.6 Simulation results and comparison

Table 4.1 summarizes the results of extensive Monte Carlo, glitch energy and leakage simulations. All flip-flops operated correctly in all Monte Carlo simulation points. The figure of merit in this comparison is the power-delay product at 50% switching activity.

According to the results reported in Table 4.1, the two best DET flip-flops are the new CTF_C and CT_C flip-flops. The CT_C design has the lowest transistor count, lowest glitch energies and the lowest leakage of all the surveyed flip-flops. The improved CTF_C design achieves the lowest PDP values at high switching activities. Compared to the common LM flip-flop, the CT_C design improves on transistor count, power and PDP at 50% and 100% switching activities, D–Q delay, leakage and glitch energies. The improved CTF_C design is additionally better in $\mathrm{PDP}_{0.1}$.

In terms of $P_{0.1}$, the best performing flip-flops are the LM_C, LG_C and FN_C designs. The proposed FN_C (sym) design has the lowest $P_{0.1}$. In this case, LM shows higher power than LM_C because it uses more clocked transistors. LM_C has fewer such transistors because it uses a C-element to perform the function of the MUX.

Considering energy dissipation due to glitches, the four worst performing flip-flops are the LM, LM_C, TSP and IP designs. In these four flip-flops, every change at the input leads to switching within internal circuits. The measured G_1 and G_3 glitch energies for these flip-flops are such that the presence of multiple glitches at the input leads to glitch energies dominating the overall power consumption. All five novel flip-flops have significantly lower glitch energies.

The new LG_C and FN_C flip-flops are the only surveyed flip-flops that have both low $P_{0.1}$ and low G_1 and G_3. The FN_Cs glitch energies are lower than that of LG_C due to the absence of static latches at nodes A and B. Energies due to the first glitch in both LG_C and FN_C are high due to switching at one of A or B, yet the energies of subsequent glitches in the same cycle are considerably lower due to the flip-flops' internal states staying unchanged. This behaviour distinguishes the proposed LG_C and FN_C flip-flops from Latch-MUX flip-flops.

Table 4.1: Simulation results of the new and previously reported DET flip-flops.

DET FF	LM [37]	LM_C [46]	EP [38]	IP [43]	CP [34]	TSP [48]	LG_C	IP_C	FN_C	FN_C (sym)	CT_C	CTF_C
Transistor count	26	28	24	21	35	38	28	26	30	34	20	28
$P_{0.1}$, µW	4.69	3.81	9.28	4.39	6.15	5.57	3.86	7.68	3.85	3.74	6.09	5.93
$P_{0.5}$, µW	8.34	8.68	11.84	8.36	8.63	8.41	8.80	9.98	7.54	7.30	6.83	7.02
P_{1}, µW	12.91	14.78	15.04	13.31	11.74	11.97	14.97	12.85	12.15	11.76	7.76	8.40
$PDP_{0.1}$, fJ	0.240	0.267	0.258	0.169	0.265	0.357	0.321	0.419	0.300	0.269	0.290	0.202
$PDP_{0.5}$, fJ	0.428	0.609	0.330	0.322	0.372	0.540	0.731	0.545	0.587	0.526	0.325	0.239
PDP_{1}, fJ	0.662	1.036	0.419	0.513	0.506	0.768	1.244	0.702	0.947	0.847	0.369	0.285
CV($P_{0.5}$)	0.017	0.019	0.022	0.026	0.018	0.018	0.017	0.017	0.019	0.018	0.025	0.021
CV($PDP_{0.5}$)	0.070	0.074	0.092	0.079	0.080	0.084	0.082	0.095	0.079	0.070	0.111	0.081
t_{cq}, ps	41.2	51.6	44.9	42.6	57.7	41.8	62.7	59.1	64.2	59.1	57.4	43.3
t_{dq}, ps	51.3	70.1	27.9	38.6	43.1	64.1	83.1	54.6	77.9	72.1	47.6	34.1
CV(t_{cq})	0.077	0.078	0.077	0.087	0.077	0.086	0.081	0.077	0.077	0.077	0.096	0.083
CV(t_{dq})	0.074	0.078	0.088	0.086	0.078	0.087	0.084	0.094	0.080	0.074	0.109	0.087
t_{h}, ps	0.5	27.9	81.0	52.4	57.8	3.5	37.8	33.8	35.9	34.2	38.3	26.5
I_{DDQ}, nA	146	147	130	141	126	133	169	153	124	113	89	118
G_{1}, fJ	5.12	5.73	4.78	4.66	3.66	6.06	5.08	3.97	3.72	3.92	3.29	3.91
G_{3}, fJ	11.55	14.53	6.19	9.99	5.14	11.69	6.05	5.40	5.25	5.46	4.07	6.07

Of the three LG_C, IP_C and FN_C designs, IP_C is the fastest while FN_C shows the best overall performance: FN_C and FN_C (sym) designs show the lowest power, $PDP_{0.5}$, variations and glitch energies. Comparing the FN_C and FN_C (sym) designs, the latter shows reduced power and reduced delays. The PDP values of FN_C (sym) are 10% better than that of FN_C, which is attributable to the symmetric implementation of the output C-element having a lower PDP in itself when compared to the weak-feedback implementation [53]. Although not shown in Table 4.1, similar performance gains are also seen when using symmetric C-elements in LG_C and IP_C flip-flops.

Considering the Monte Carlo analysis, LM and FN_C (sym) are the two flip-flops that exhibit the lowest variation in their parameters. Relative variations in the CT_C design are the highest, which is due to its weak transistors playing a key role in its energy and timing performance parameters. Generally, performance of weak transistors is more susceptible to variations than that of stronger and wider transistors.

Considering variations in CK–Q delays, 7 out of 12 DET flip-flops show practically the same CV of 7.7%. Although the absolute values of SDs for these flip-flops are quite different, the fact that these work out to the same RSD shows that these differences have less to do with particular circuit topologies and more to do with the technology itself. Monte Carlo simulations show a clear trend that higher mean values have higher absolute variations. This trend is also evident in a previous Monte Carlo analysis of DET flip-flops in [52]. The work in [52] judged the EP design as the best in terms of parameter variations, but it only considered absolute variations. While it is true that the absolute value of the EP's SD for the D–Q delay is the lowest, its RSD is actually one of the highest.

The hold time was found to be positive for all novel flip-flops with the CTF_C achieving the lowest value among them. The hold time was also found to be positive for most of the previously reported DET flip-flop designs except the TSP as shown in Table 4.1.

The results of voltage scaling simulations are shown in Figure 4.19. The CK–Q delays of all flip-flops scale in a very similar way. The only two flip-flops that scale slightly differently from others are the CT_C and CTF_C flip-flops. The CK–Q delays of these two flip-flops increase faster with decreased supply voltage when compared to other flip-flops. The opposite is true as well: increasing the supply voltage decreases delays in these flip-flops faster than in other designs. Had the traces of these two flip-flops not been plotted, there would have been no line crossings on the plot.

4.7 Conclusion

Five new DET flip-flop designs have been developed. The new designs were compared to previous DET flip-flops using circuit simulation. The new LG_C design and its variants were shown to significantly improve on Latch-MUX DET flip-flop designs in the metric of energy dissipation due to glitches at the input, which makes them useful for designs with large logic depth that are prone to glitching. The new CT_C and CTF_C designs can be used in high-performance scenarios as they were found to have superior power and power-delay products during periods of high switching activity.

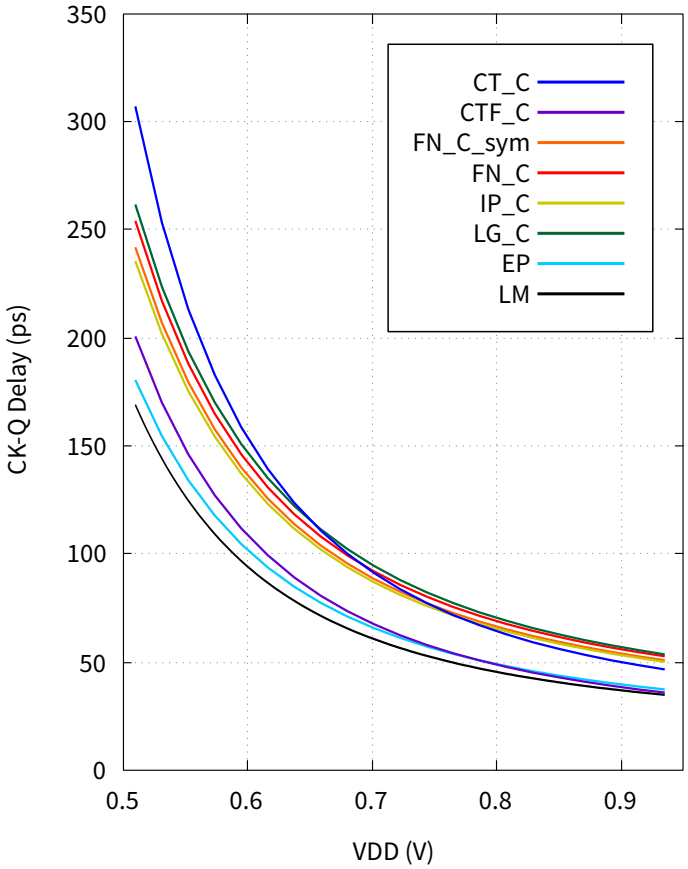


Figure 4.19: Plot of the CK–Q delay versus the supply voltage for new flip-flops and two previous LM and EP designs.

Extensive Monte Carlo simulations were carried out to demonstrate that the new flip-flops are robust under process variations. The new FN_C design was found to be one of designs least susceptible to process variations. Voltage scaling simulations were performed that show that the performance of the presented flip-flops scales very similarly to that of previous DET flip-flops.

Chapter 5

Memory design for the output-buffered FX correlator

An alternative to input-buffered FX correlator architectures, one of the most promising of which was discussed in Chapter 3, is an output-buffered architecture. An output-buffered design was proposed for the SKA in [54]. The main advantage of such a design for the X part is that it can process the data from the F part in the same order as it is produced and thus does not require any corner turning between the F part and the X part. Disadvantages of this approach include much higher requirements for the performance of the output buffer memory: in the input-buffered architecture, the memory bandwidth in the input buffer is proportional to the number of antennas N, whereas in the output buffer it is proportional to the number of baselines which in turn is proportional to N^2.

In the fully output-buffered design, there is a CMAC for every baseline. The output buffer consists of smaller output buffers for each of the CMACs. These smaller buffers store intermediate accumulations for every frequency channel. For example, for a design with $C = 265\,000$, each CMAC effectively includes 265 000 accumulators. For such a large number of accumulators, space-efficient eDRAM becomes the only implementation option as far as conventional technologies are concerned. Although versatile, eDRAM has a number of disadvantages that can be detrimental to the chip's overall performance. Firstly, eDRAM's maximum clock rate is significantly lower than what is achievable in the processing circuits in the same technology. Secondly, the memory becomes responsible for a significant share of the overall power dissipation. This work aims to develop a new space-efficient memory based on eDRAM that achieves the same maximum clock rate as the processing circuits while dissipating less energy.

Not all processing tasks require the memory to have all features DRAM has to offer. DRAM's circuits can be tailored to a particular memory access pattern at the expense of the memory's general versatility. This chapter proposes a new design of a high-performance, low-power and space-efficient memory for applications where processing circuits process the stored data sequentially by repeatedly cycling through the memory at a high clock rate. Because of sequential access patterns, this is a Sequential Access Memory (SAM). The new memory design reduces energy dissipation and increases the processing throughput without compromising on the design area. The focus of this work is on the implementation of a proof-of-concept prototype. The 130 nm GF CMRF8SF CMOS technology was used as the prototyping technology.

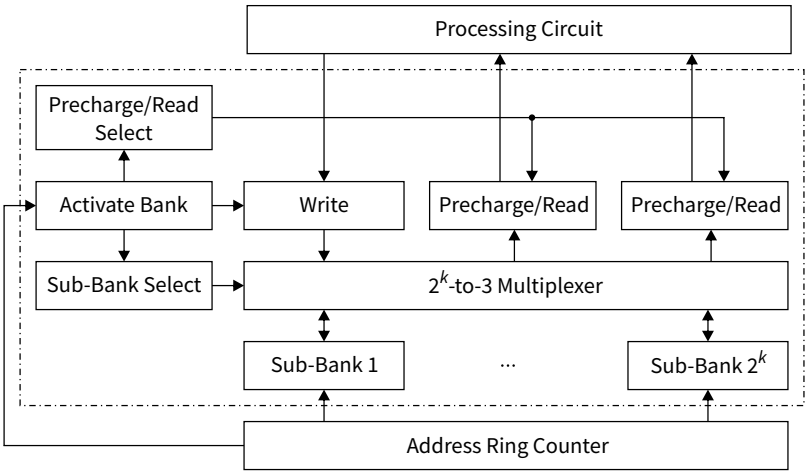


Figure 5.1: Architecture of the SAM memory.

5.1 Sequential access memory

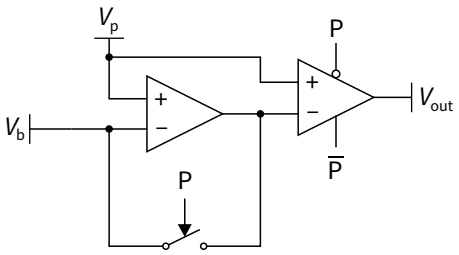
The architecture of this memory is similar to the previously reported SAM architectures [55], [56], [57]. The main difference to the previous designs is the usage of open-loop sensing.

The memory consists of several memory banks each consisting of several identical sub-banks. The architecture of one bank is shown in Figure 5.1. Sub-banks consist of bitlines with memory capacitors. Each bank has one writing circuit and two precharging/reading circuits that are shared among all sub-banks in this bank. The multiplexer in a bank consists of three smaller multiplexers that connect each sub-bank to one or none of the writing and precharging/reading circuits.

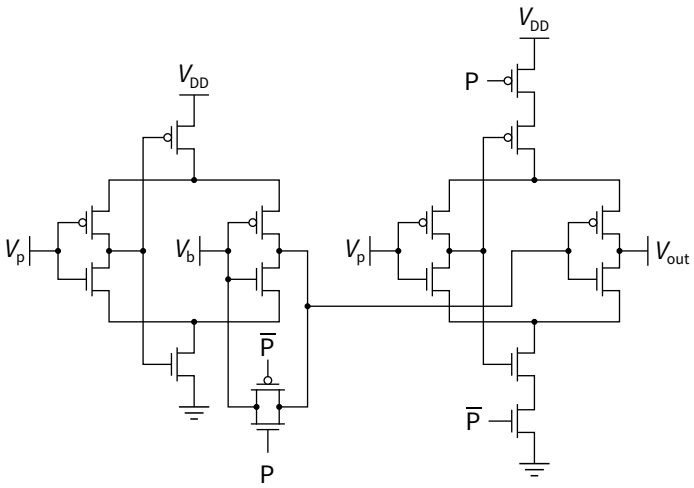
Due to the full predictability of memory access, the access rate is increased by essentially eliminating address decoding delay and pipelining memory operations. The pipelining is such that both read and write operations are allowed to take two full clock cycles. At any one moment during the operation, the memory is precharging the bitlines two address positions ahead, sensing the data one address position ahead and writing the data one and two address positions behind the current processing position. Because the sequence of memory operations always includes a write operation after a read operation for every address, the use of open-loop sense amplifiers that do not write the data back into the memory becomes possible.

In Figure 5.1, one Precharge/Read circuit consists of sense amplifiers for every bitline in a sub-bank that are used for both precharging and sensing. The schematic diagram of one amplifier is shown in Figure 5.2. The amplifier consists of two stages. Each stage is a self-biased differential amplifier [58]. During precharging, the first stage of the sense amplifier acts as a voltage buffer and the second stage is powered down to turn its output into a high-impedance state and also to reduce overall power dissipation. Since there is no need to recharge memory cells as part of reading, the sense amplifier switches to an open-loop topology during reading and acts as a voltage comparator.

The advantages of an open-loop design are higher sensitivity, faster sensing and lower power dissipation. The disadvantage is the increased transistor count when compared to the DRAM sense amplifier designs. Higher sensitivity of this



(a)



(b)

Figure 5.2: Schematic diagrams of the sense amplifier: (a) the general diagram and (b) its transistor-level circuit. V_b is the voltage on the bitline and V_p is the precharge voltage equal to $0.5\,V_{DD}$.

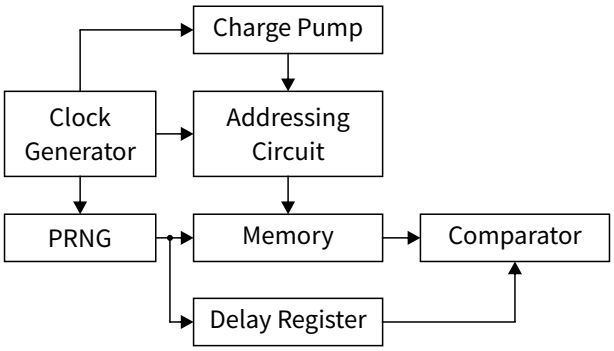


Figure 5.3: Top-level architecture of the fabricated chip.

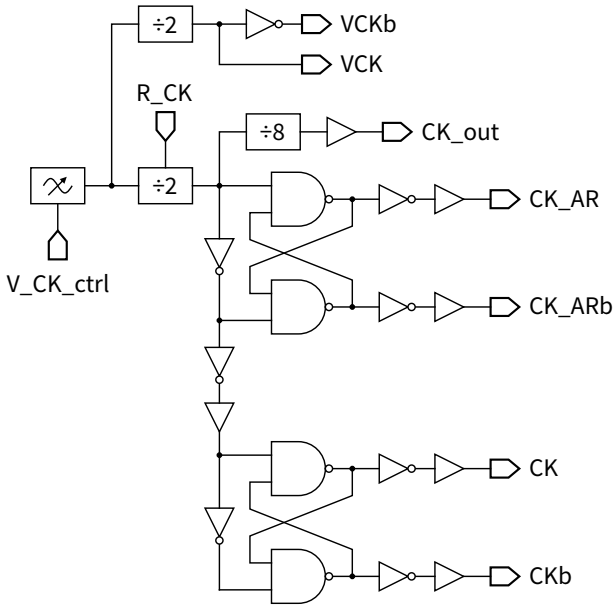


Figure 5.4: The schematic diagram of the Clock Generator circuit.

circuit is due to a reduced effect of the amplifier's random offset voltage, which is significant in RAM circuits [59], [60],[61]. In DRAM, sense amplifiers use positive feedback to both sense and recharge memory cells. With positive feedback, the change of the bitline's voltage due to the charge on a memory capacitor must be higher than the offset voltage in order for the bit to be sensed correctly. In the open-loop design, the effect of the offset voltage is reduced by the gain of the sense amplifier. Another difference to the DRAM sensing is that DRAM circuits wait until memory cells are recharged before producing the output. In the case of the open-loop sensing circuit, bitline voltage changes are amplified to a full voltage without recharging, which leads to faster sensing and reduced power dissipation.

5.2 Design description

The top-level architecture of the fabricated chip is shown in Figure 5.3. The implemented dynamic memory consists of one bank with 8 sub-banks. Each sub-bank consists of 8 bitlines with 128 memory cells on each one. This corresponds to the total of 8 Kib of memory. In the absence of space-efficient DRAM memory

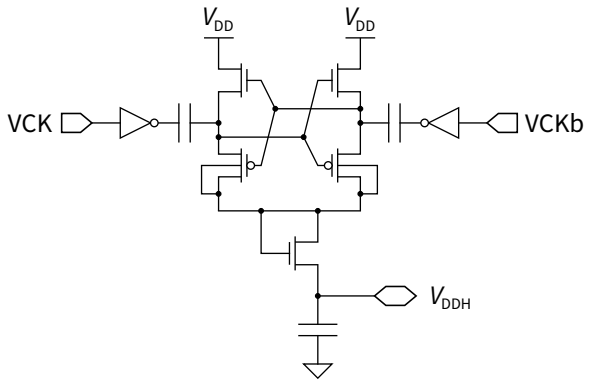


Figure 5.5: The schematic diagram of the Charge Pump circuit.

capacitors, low-leakage thick-oxide varactors with 80 fF of capacitance are used for memory capacitors in this prototype. The dimensions of each varactor are such that a bitline with 128 memory cells on it is about 0.9 mm long. The total area of this memory is about 0.61 mm^2.

The circuit diagram for the Clock Generator circuit is shown in Figure 5.4. The Clock Generator's VCO frequency is controlled by an external signal. The VCO is a common current-starved ring oscillator. The VCO generates clock signals for the memory testing circuits, the charge pump, the addressing circuit, and the chip output. The clock signal for the testing circuits is delayed relative to that for the addressing circuit by the addressing circuit's delay. The chip output clock is one eighth of the frequency of the chip processing clock. Some of the circuits in the memory are positive-edge-triggered, some are negative-edge-triggered and others are dual-edge-triggered. The overall memory circuit is dual-edge-triggered and the testing circuits are thus also dual-edge-triggered.

The implemented Charge Pump circuit is shown in Figure 5.5. The design uses a voltage doubler [62], [63], [64] and a diode-connected MOSFET. The Charge Pump is built using 2.5 V transistors while the rest of the circuit uses 1.2 V transistors. The doubler portion of the circuit produces about 1.8 V, and a diode-connected MOSFET is used to reduce it to about 1.65 V. The diode-connected MOSFET also makes it possible to supply V_{DDH} externally without shorting it with any of the internal signals.

The Addressing Circuit is implemented as a shift register similar to [65]. This circuit is powered by the V_{DDH} voltage generated in the Charge Pump. This voltage is higher than the chip's V_{DD} voltage to allow charging of memory capacitors to full V_{DD} through n-type transistors. Given that the Addressing Circuit operates at a higher voltage than the rest of the chip, logic level converters are used for interfacing between the two voltage domains. The only signals that are interfaced in this way are the clock signals and the reset signal. Clock gating is employed to only clock the active portions of the shift register.

The memory is tested using the pseudorandom number generator (PRNG) implemented as a linear-feedback shift register (LFSR). The LFSR generates a repeating pattern of 8-bit numbers with the repetition period of 255 cycles using the feedback polynomial $x^8+x^6+x^5+x^4+1$. During the operation the Addressing Circuit continuously traverses the entire memory address space in a cyclical manner. In every clock cycle the output from the LFSR is written into the current memory

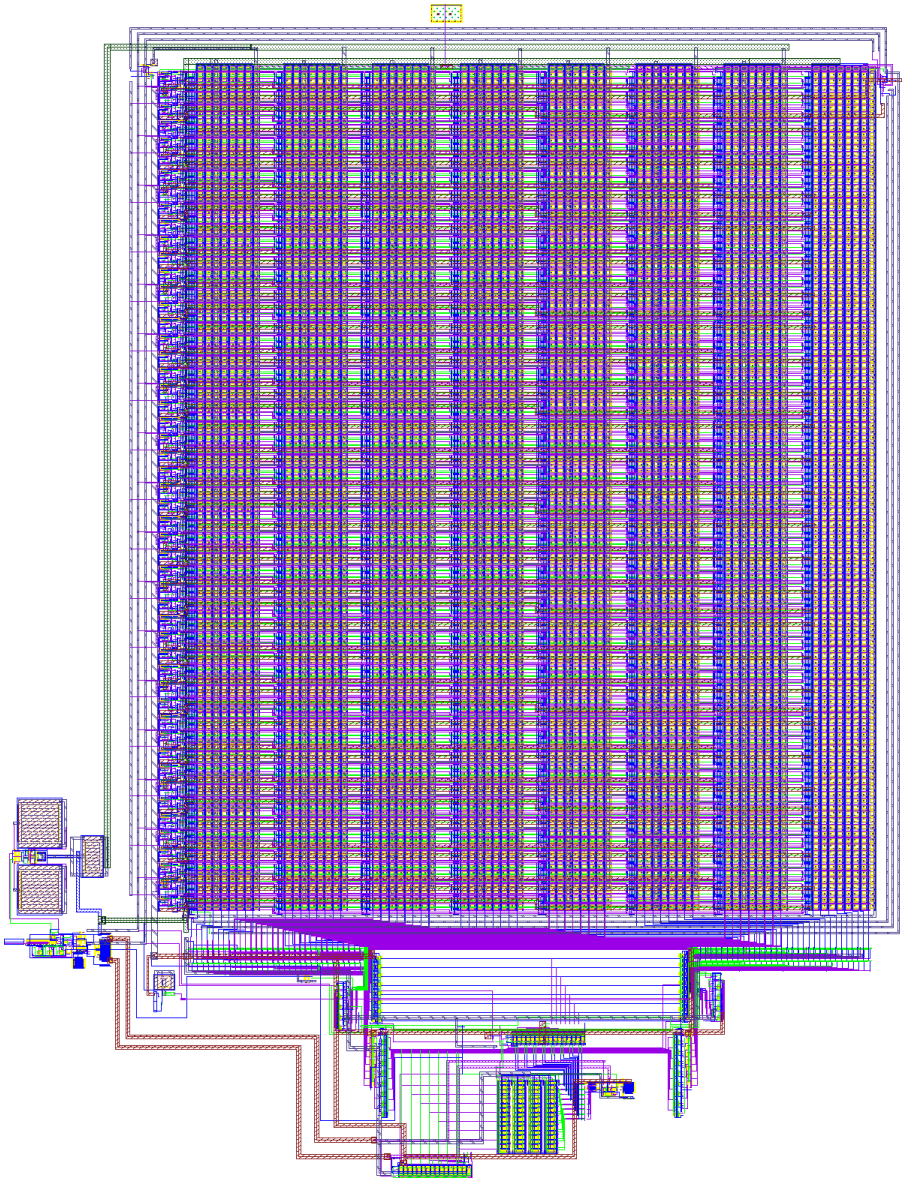


Figure 5.6: The layout of the designed memory prototype.

address. Given that the memory holds 1024 8-bit numbers and that $1024 \bmod 255 = 4$, the current value that is read from the memory is 4 cycles behind the LFSR output. This is exploited for verification purposes: every LFSR output is delayed in the Delay Register by 4 cycles and compared to the current value from the memory in the Comparator. The Comparator's output is provided at the chip's output. Thus, logic 1 at the Comparator's output corresponds to the correct operation of the memory.

The designed layout of the overall circuit is shown in Figure 5.6. The corresponding photo of the fabricated chip is shown in Figure 5.7.

5.3 Testing setup

The schematic diagram of the testing circuit with this chip is shown in Figure 5.8. The IC was designed to implement all the necessary testing circuits internally so

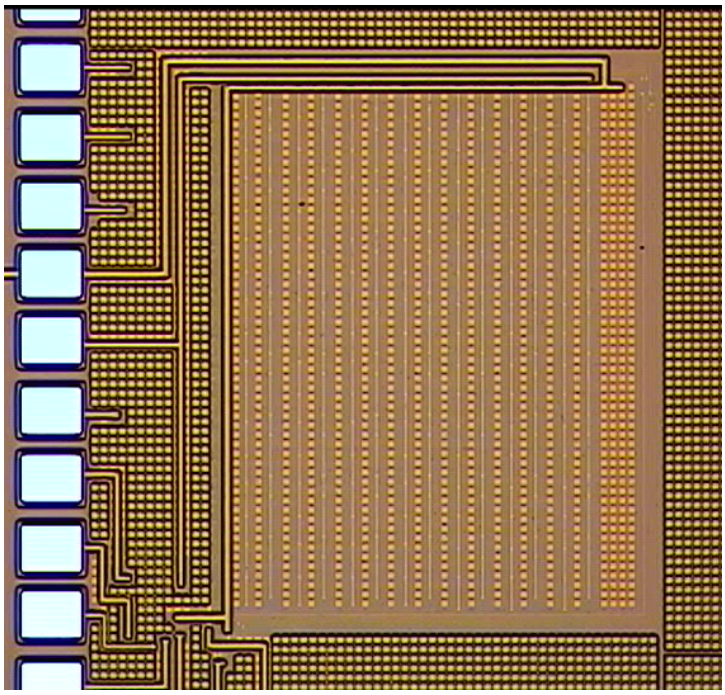


Figure 5.7: The chip photo of the fabricated memory prototype.

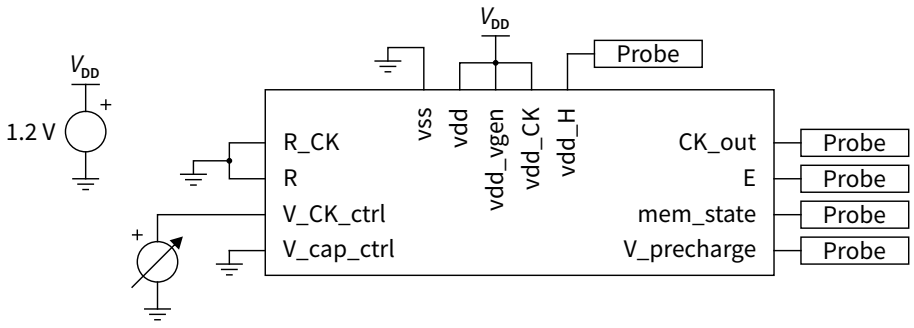


Figure 5.8: Schematic of the test circuit with the fabricated chip.

that no complex external circuitry would be required to test its operation. The external circuit includes a 1.2 V power supply, the variable voltage supply for the Clock Generator's VCO frequency control and an oscilloscope with 100 MHz of analog bandwidth. The IC's self-test circuits begin operating as soon as the chip is powered on. The chip uses multiple VDD pins to allow independent power supply and separate verification of the internal testing circuits, the charge pump and the clock generator.

The inputs to the chip are the two reset signals "R" and "R_CK" for the internal testing circuits and the clock generator respectively, the VCO frequency control pin "V_CK_ctrl," and the "V_cap_ctrl" pin. The latter is connected to the n-wells of all the memory cells' varactors. Thus, the voltage at the "V_cap_ctrl" pin could be used to influence the capacitance of memory cells. However, for thick-oxide varactors the capacitance tunability range is low especially for the control voltages in the range from 0 V to 1.2 V. The "V_cap_ctrl" pin is set to ground during testing.

The outputs from this chip are the following:

- The "CK_out" signal which is one eighth of the frequency of the internal clock signal;
- The "vdd_H" pin is internally connected to the output of the Charge Pump; this voltage can also be set externally;
- The "E" output from the Comparator which is "1" when the memory operates correctly and "0" otherwise;
- The "mem_state" signal which is one of the control signals for the internal multiplexers; during the correct operation this signal has a 25% duty cycle at one fourth of the clock frequency;
- "V_precharge" which is the analog precharge voltage for the bitlines; this pin is nominally at $0.5V_{\mathrm{DD}}$ generated internally with an active voltage divider.

5.4 Test results

The Charge Pump was tested first. Figure 5.9 shows the oscilloscope trace of the V_{DDH} voltage at the chip's output during steady-state operation. The trace shows steady voltage at about 1.65 V which is higher than the power supply's 1.2 V as is shown in the figure. It was found that the charge pump does not maintain its high voltage well when the VCO frequency is low which is illustrated in Figures 5.10 and 5.11 that show the V_{DDH} versus the VCO control voltage. For this reason V_{DDH} was supplied externally when testing low-frequency performance of the memory.

Figure 5.12 shows the "E" signal indicating the correctness of operation versus the VCO control voltage. The figure shows that the memory needs a certain minimum clock frequency to operate correctly. When the frequency is reduced, the chip reports that the memory operates inconsistently with some values being read correctly and some incorrectly. The memory was also observed to take a few seconds to a few minutes to begin failing when the clock frequency is low.

Figure 5.13 shows the correct "CK_out" and "mem_state" signal traces when the clock frequency is low indicating correct operation of the clock generator and memory state circuits. Figure 5.14 shows the "CK_out" and "E" signals for the case of about 8 MHz of internal clock frequency. As shown in the figure, the chip reports correct operation. Figure 5.15 shows the chip reporting correct operation for the case of about 387 MHz of internal clock frequency. Given that this is a

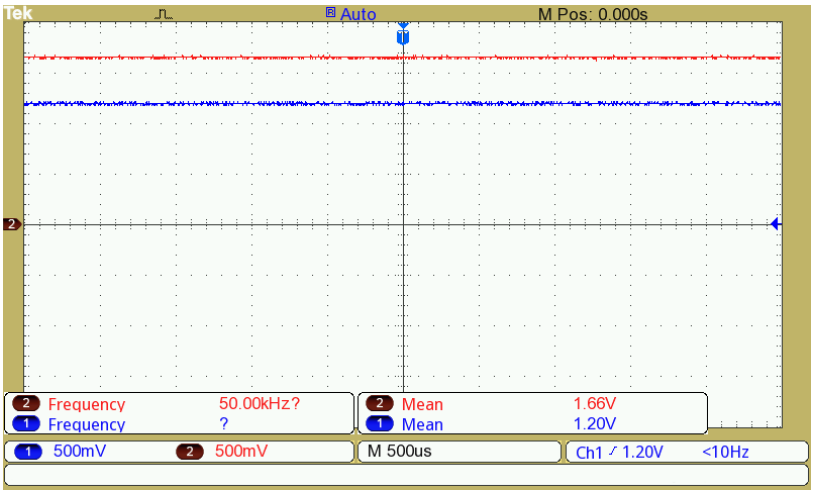


Figure 5.9: Oscilloscope traces of (CH1, blue) the power supply voltage and (CH2, red) the output voltage of the charge pump during operation at high clock frequencies.

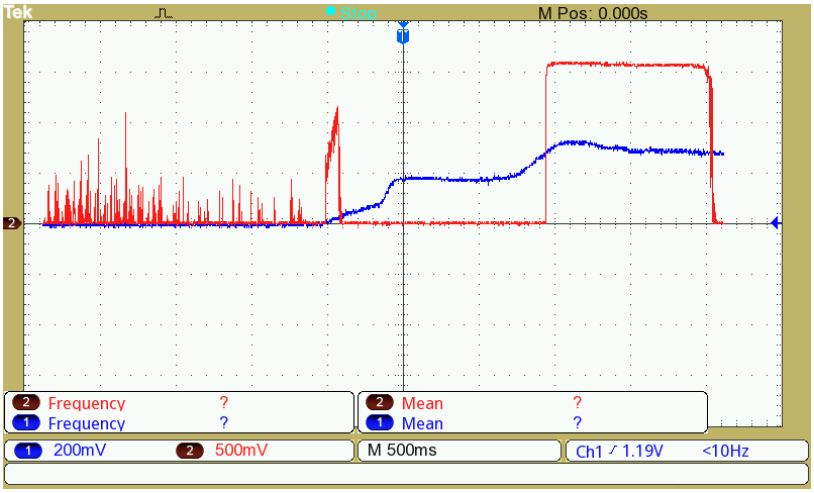


Figure 5.10: Oscilloscope traces of (CH1, blue) the VCO control voltage and (CH2, red) the output voltage of the charge pump when the VCO control voltage is varied manually. Note the different voltage scales for the two traces.

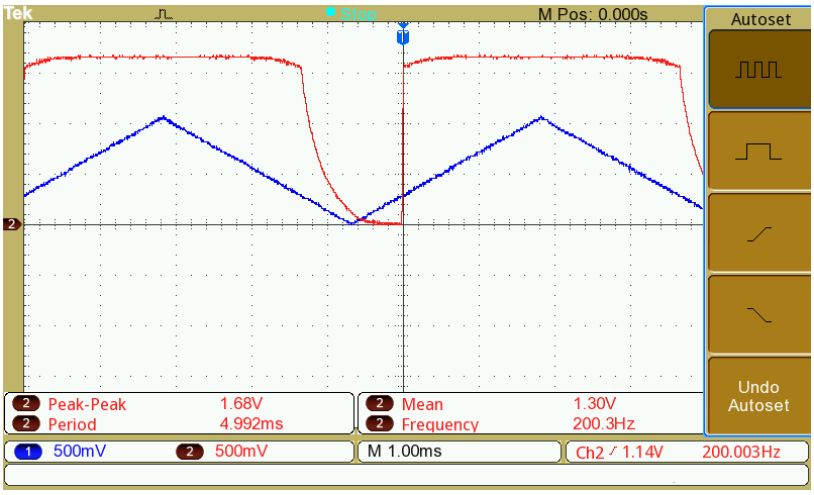


Figure 5.11: Oscilloscope traces of (CH1, blue) the VCO control voltage and (CH2, red) the output voltage of the charge pump when the VCO control voltage is generated externally to be a low-frequency sawtooth wave.

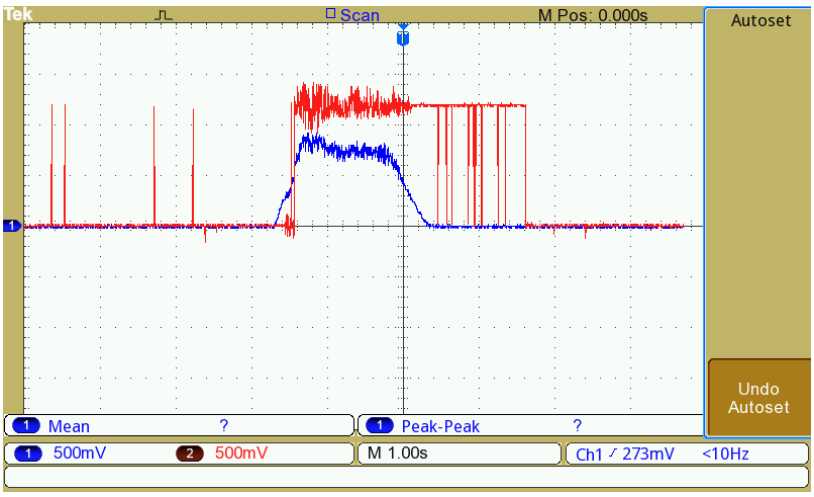


Figure 5.12: Voltage traces of (CH1, blue) the VCO control voltage and (CH2, red) the “E” signal which indicates correctness of operation as reported by the built-in self-test circuits.

Figure 5.13: Correct traces of (CH1, blue) the "CK_out" and (CH2, red) the "mem_state" signals which are respectively one eighth and one fourth of the internal clock frequency.

dual-edge-triggered circuit, this performance is equivalent to a 773 MHz clock for a single-edge-triggered case.

Figure 5.16 shows that the circuit operates correctly at 77 MHz of output clock frequency. The memory was found to be begin failing at about 78 MHz of output clock frequency as shown in Figure 5.17. This corresponds to about 625 MHz of internal clock frequency. Given that the memory and testing circuits are dual-edge-triggered, this performance is equivalent to the clock rate of more than 1.2 GHz for single-edge-triggered circuits.

5.5 Conclusion

The 130 nm prototype of the new dynamic memory suitable for CMACs with a large number of accumulators was fabricated and tested. The memory implements a sequential access scheme and novel open-loop sensing to increase both its timing and power performance. The memory was found to work correctly at cycle times as low as 0.8 ns, which is equivalent to the 1.2 GHz performance for single-edge-triggered circuits. Achieving the high clock frequency for which the memory begins to fail was crucial to its verification. Rather than seeing the built-in self-test output stuck at 1, the existence of the upper limit on the memory's performance effectively proves the correct operation of the memory and the internal testing circuits.

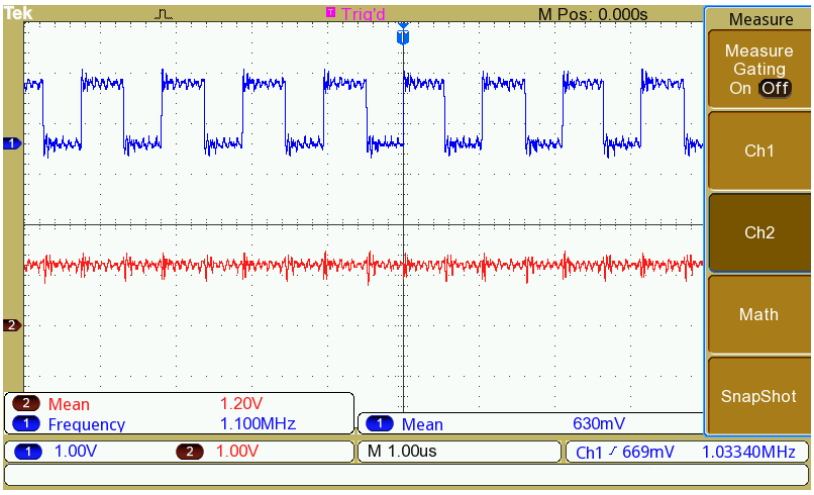


Figure 5.14: Voltage traces of (CH1) the “CK_out” and (CH2) the correctness “E” signals as reported by the chip at a low clock frequency. The memory operates correctly.

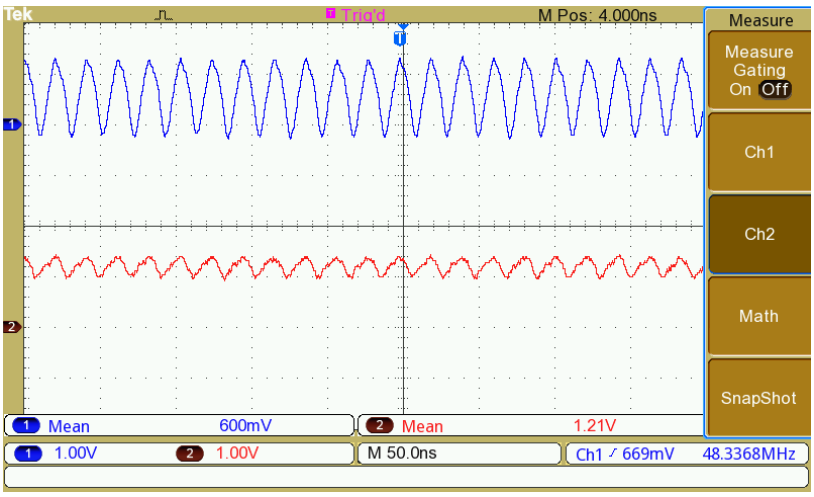


Figure 5.15: Voltage traces of (CH1) the “CK_out” and (CH2) the correctness “E” signals at 773 MHz of internal VCO frequency. The “CK_out” trace does not appear to be square wave because of the 100 MHz bandwidth limit of the oscilloscope.

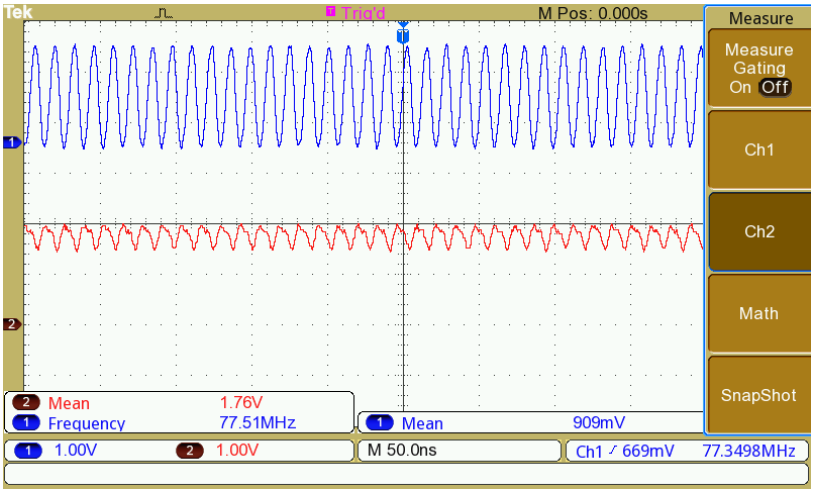


Figure 5.16: Voltage traces of (CH1) the “CK_out” and (CH2) the correctness “E” signals when the VCO is at about 1.2 GHz. The memory works correctly. The “CK_out” trace does not appear to be a square wave because of the 100 MHz bandwidth limit of the oscilloscope.

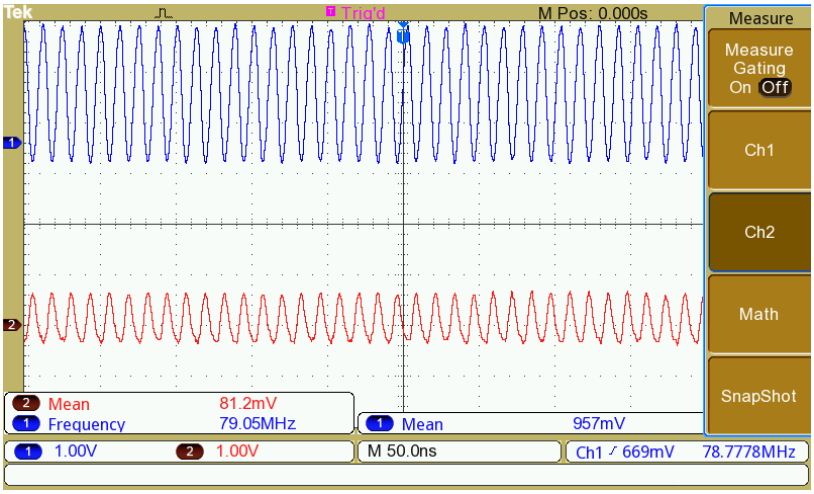


Figure 5.17: Voltage traces of (CH1, blue) the “CK_out” and (CH2, red) the correctness “E” signals as reported by the chip when the VCO is at above 1.2 GHz. This is the clock frequency for which the memory begins to fail. The “CK_out” trace does not appear to be a square wave because of the 100 MHz bandwidth limit of the oscilloscope.

Chapter 6

SerDes I/O design

Some SKA antennas are envisioned to be generating about 1 Tbps of data per antenna. This creates a very data- and I/O-intensive signal processing task, where the challenge is not only to be able to process the data but also to be able to deliver it to places where it is processed. This challenge applies to the design of large and efficient memories that can support very high bandwidths while being power-efficient and also to the design of I/O circuits.

It was estimated in [40] for a proposed implementation of the input-buffered FX correlator architecture that the performance of each processing unit will be limited by the available chip I/O resources, which is mostly due to the cost concerns of ASIC IP blocks for high-speed SerDes I/O circuits. This example demonstrates that performant I/O circuits are key to the effective implementation of a modern correlator architecture. The design of SerDes I/O represents a greater challenge than the design of mostly digital circuits that have been presented in this work so far. From the point of view of IC design, the most complicated part of an ASIC correlator implementation is the I/O.

This chapter presents the design of a prototype 15 Gbps transmitter circuit. This is a full-rate design of the source-series terminated type. A high-level diagram of the presented design is shown in Figure 6.1. The chapter introduces the designed circuits including the LVDS clock reference amplifier, the PLL, the serialiser, the driver and ancillary circuits such as the biasing circuits.

6.1 Analog design

The transmitter design uses analog circuits to a greater extent than the designs presented in previous chapters do. Because of the high clock rate at which some transmitter circuits operate, some digital circuits have to be designed with the care of analog ones.

Some of the features that high-rate circuits operating close to the limits of the underlying technology have to additionally consider throughout the design cycle include PVT variations and parasitic effects. An example of a circuit implemented

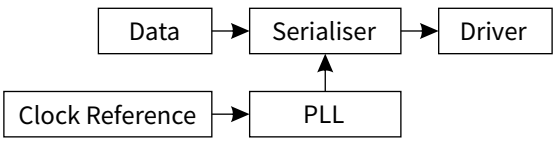


Figure 6.1: Block-level diagram of the transmitter circuit.

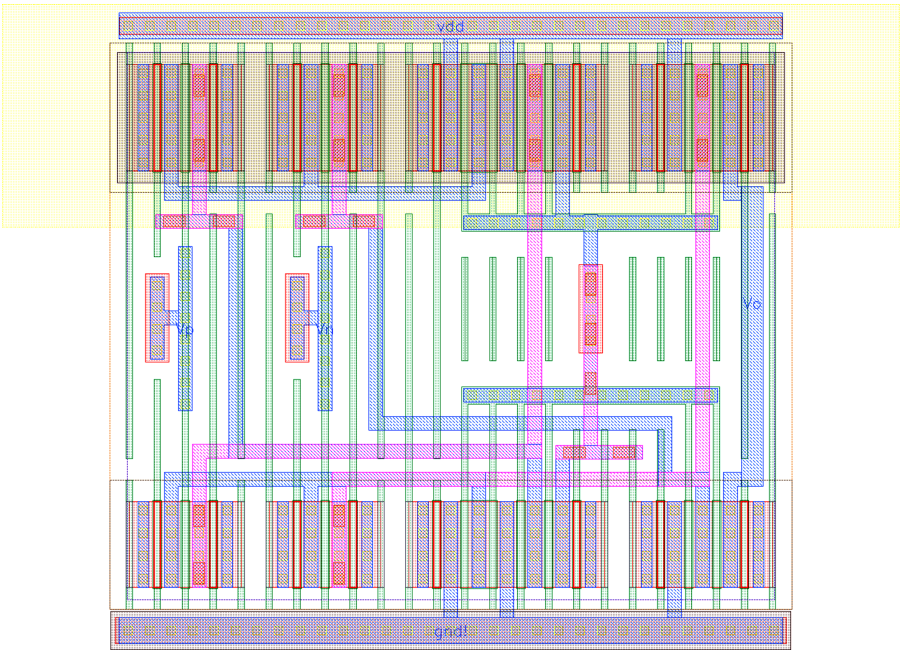


Figure 6.2: The layout of the self-biased folded cascode differential amplifier.

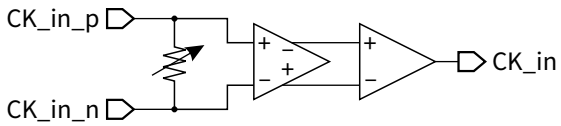


Figure 6.3: Diagram of the LVDS clock receiver.

with the minimisation of these effects in mind is a self-biased folded cascode amplifier from [58] whose designed layout is shown in Figure 6.2. Transistors of this circuit were designed to be multiples of the same width, making it possible to align the edges of diffusion areas and maintain a constant separation between n-type and p-type transistors. Transistors are stacked and every transistor stack is terminated with a dummy transistor on both sides to reduce the impact of the lateral proximity of each gate to isolation. This includes single transistors that are created as a stack of three transistors including two dummy ones. In the figure, dummy transistors are marked with red. Transistor gates are oriented in the same direction and maintain a constant pitch. In this circuit, the direction of matched currents is the same. Matched transistors also match antenna ratios. One of the features of this 28 nm technology is that metal interconnects are subject to corrections termed catastrophic optical proximity corrections that may change widths and placements of some metal wires thus having an impact on the modelled parasitics. In this case, layouts were designed to avoid being subject to such corrections.

6.2 LVDS clock reference receiver

The LVDS [66] clock reference receiver is one of the simplest circuits of this transmitter. The circuit accepts a 312.5 MHz clock from an external oscillator such as [67]. The diagram of the implemented circuit is shown in Figure 6.3. The 100 Ω

resistor is implemented using parallel resistor stripes that can be turned in or out to control the the value of this resistor similarly to the approach of [68]. Such simple resistor tuning scheme is possible because of the relatively low input signal frequency of 312.5 MHz.

The LVDS receiver amplifier is implemented as a two-stage amplifier consisting of a differential pair and a self-biased folded cascode amplifier. These circuits are designed using 1.8 V transistors that are capable of operating at 2.2 V in line with the LVDS standard. The layout of the designed amplifiers is shown in Figure 6.4.

6.3 Phased-locked loop design

The PLL design includes the LC-tank VCO, the frequency divider and the charge pump circuits. Initially, the PLL was created as a conventional type II PLL design [69, Ch. 5]. Later, the digital control of the VCO frequency was incorporated along with the analog fine-tuning.

The oscillator is the LC-tank oscillator, which is common in implementations of I/O circuits [70], [71]. The schematic diagram of the implemented VCO is shown in Figure 6.5 and is similar to the design in [72] except that the biasing transistor is removed. The capacitor is implemented as a parallel connection of several varactors. The voltage on the varactors controls the capacitance of these devices, which in turn controls the oscillation frequency. The designed VCO nominally oscillates at 15 GHz and can to be tuned within 0.5 GHz of its nominal value. Although multiple sources include recommendations on how to optimise the designs of such oscillators [73], [74], most of the published SerDes designs that are similar to this work have not followed it and have found the quality of their oscillators to be acceptable. In this case, the sizes of passive devices such as inductors were changed so as to have a smaller circuit footprint at some expense of oscillator quality.

The VCO frequency is controlled digitally with a 4-bit number. A fifth “bit” is the analog control that partially overlaps in control strength with the two least significant bits of the digital control. This analog control enables fine tuning of the VCO frequency. This digital control scheme reduces the oscillator noise as was found in [75] and [72]. It is interesting that once the digital control has been incorporated into the VCO, the mismatch between the p-type and n-type transistor currents of the VCO’s charge pump became largely irrelevant. The initial version of the charge pump used a current reference and current mirrors to match the two currents, which were removed in the final version. The simulated design was found to achieve a correct lock state in cases of a 10x mismatch in both n-type and p-type transistors.

The frequency divider implements a fixed divide-by-48 circuit. This is achieved using a divide-by-16 circuit constructed with rail-to-rail logic flip-flops from [76] and a divide-by-3 Johnson counter, which is usually a divide-by-6 circuit only in this case it is dual-edge-triggered. This PLL is thus a frequency multiplier, which takes a 312.5 MHz reference signal and outputs 15 GHz.

6.4 Serialiser

The serialiser implements a 66-to-1 serialisation. Each transmitted frame is 66 bits including the 2 frame bits and the 64 data bits as is used in the 64b/66b line

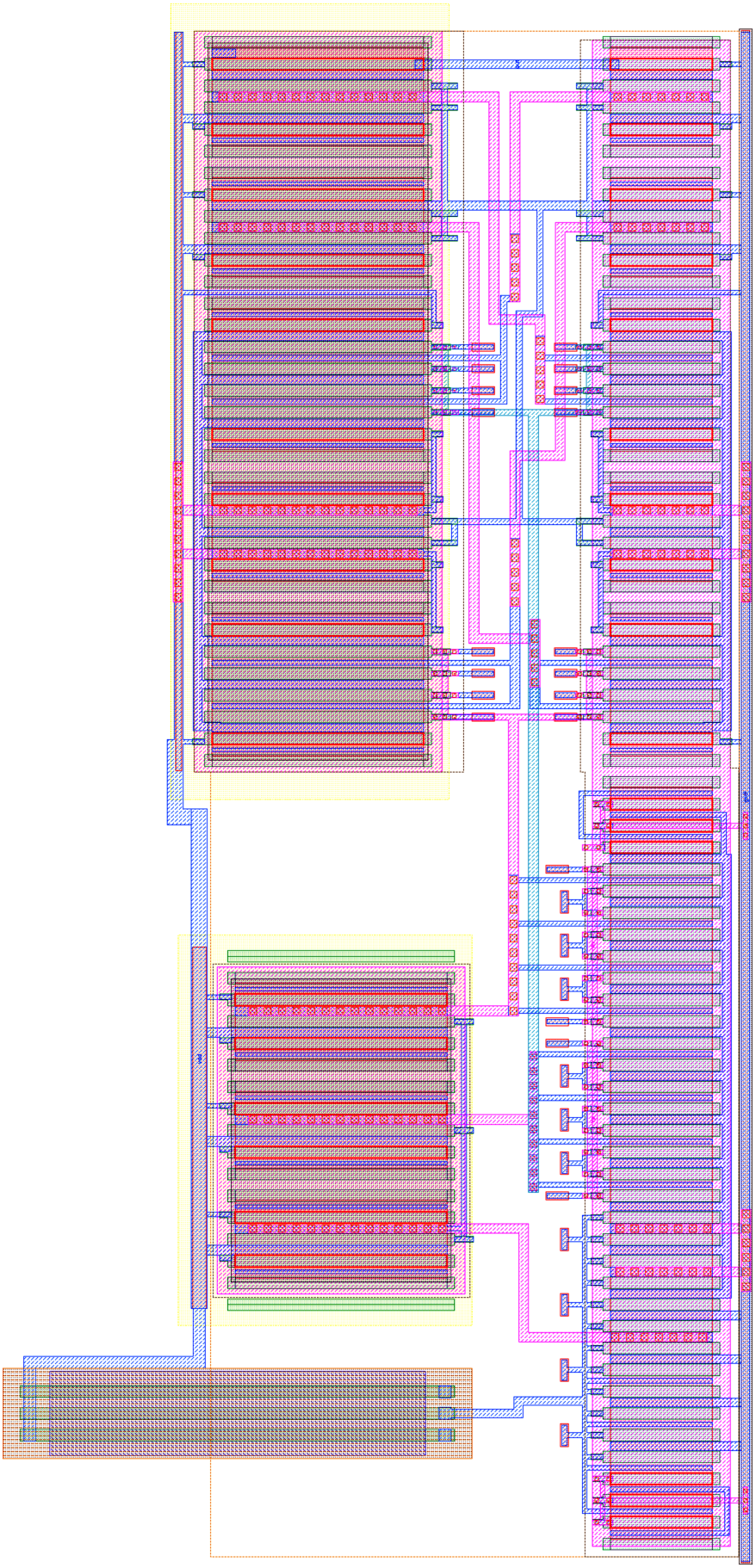


Figure 6.4: The layout of the designed clock reference input amplifier.

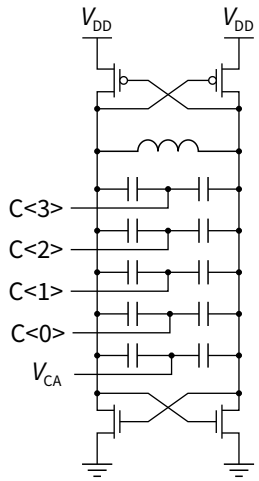


Figure 6.5: Schematic diagram of the implemented LC-tank oscillator.

encoding. Initially, the serialiser structure was designed to follow [77] and [78] as a 64-to-1 serialiser. In this design the framing bits were inserted into the stream by pausing the serialiser clock. This approach was found to be worse than the direct 66-to-1 serialisation that has ultimately been used in these circuits.

The 66-to-1 serialiser consists of two 33-to-1 and one 2-to-1 serialiser. The two 33-to-1 serialisers take at their inputs odd and even bits of the 66-bit frame, which are then serialised with a 2-to-1 multiplexer circuit of [79] into a single stream. Each 33-to-1 serialiser consists of three 11-to-1 and one 3-to-1 serialisers. The 11-to-1 and 3-to-1 serialisers use a token-ring structure whereby only one element outputs its value onto a common bus.

Layouts of the 3-to-1, 11-to-1 and combined 66-to-1 serialisers are shown in Figures 6.6, 6.7 and 6.8 respectively. The combined design was simulated and was found to work across the entire space of PVT variations in the presence of RLC parasitics at above 20 GHz.

6.5 Scrambler

The scrambler is essential in the design of the high-speed I/O. The purpose of the scrambler is to remove the DC component of the output signal and ensure frequent signal transitions. Multiple scrambler designs are possible [80]. The common scrambler design which would have some degree of compatibility with other SerDes designs is the LFSR design of the Ethernet standard specified in [81, Sec. 49.2.6]. The specified LFSR is a self-synchronous scrambler design [82], which would have to be converted into a parallel structure for high-speed operation using methods such as [83], [84]. For the purposes of this prototype, a parallel scrambler design was created based on the “xorshift” PRNG [85]. The implemented version of the “xorshift” generator was chosen to be the recommended version in [86].

6.6 Predriver

The predriver circuit amplifies the digital stream from the serialiser to drive the output driver. One of the issues of full-rate designs is that the serialised signal includes the delay of the flip-flop that generates it, which is not the case with

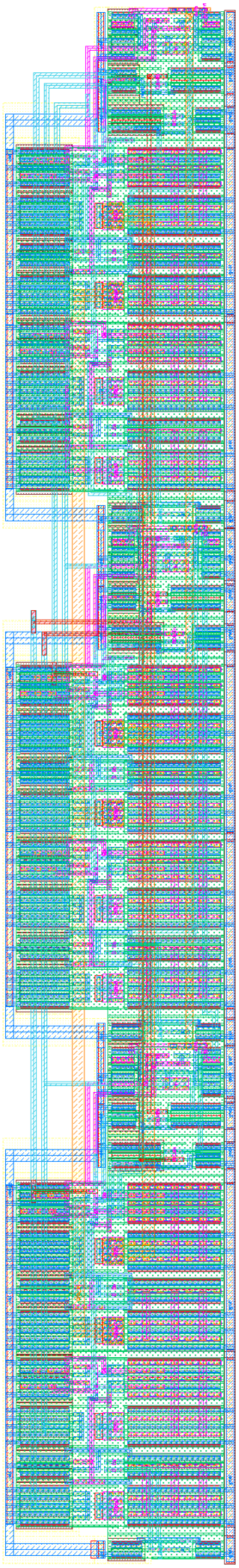


Figure 6.6: The layout of the 3-to-1 serialiser.

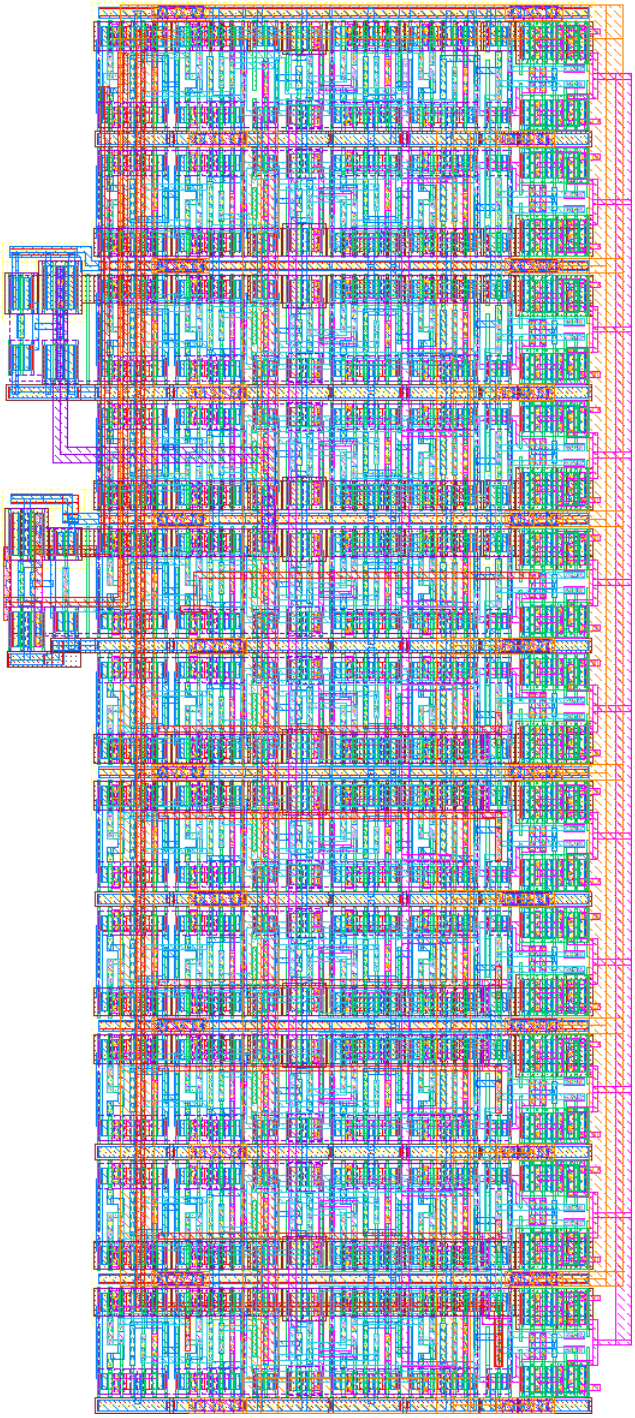


Figure 6.7: The layout of the 11-to-1 serialiser.

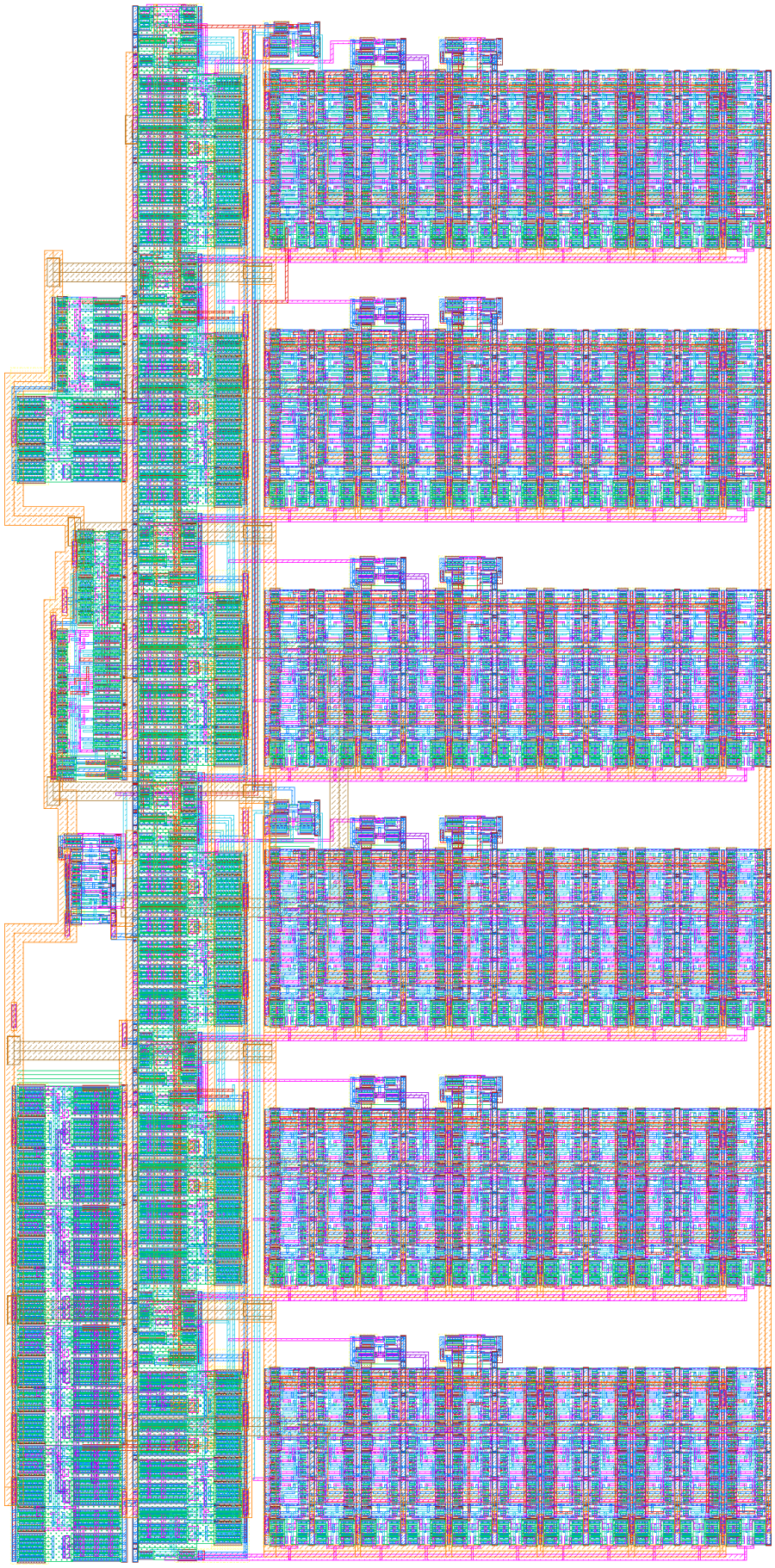


Figure 6.8: The layout of the 66-to-1 serialiser.

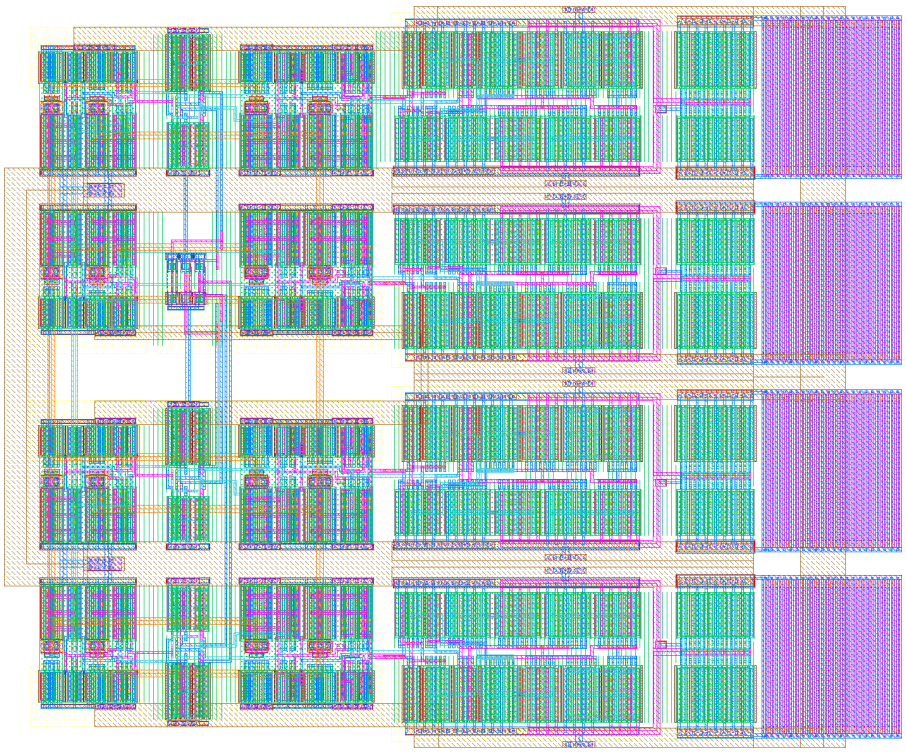


Figure 6.9: The layout of the predriver circuit.

half-rate designs. The issue is that the timings of flip-flop output transitions is not the same for 1-to-0 and 0-to-1 transitions. This design incorporates special buffer circuits from such flip-flops that use C-elements to even-out the differences in the timings of 1-to-0 and 0-to-1 transitions. The C-elements also play a role in the signal amplification. The layout of the designed predriver is shown in Figure 6.9.

6.7 Output driver

The output driver is a voltage-mode driver of the source-series terminated type [87], [88]. Advantages of this driver type over current-mode logic (CML) drivers include lower power consumption, larger swing and support for multiple termination voltages [89]. The schematic diagram of the designed driver is shown in Figure 6.10. This SST driver is structured after the recommended design in [90] and [91]. The overall layout of the driver is shown in Figure 6.11.

6.7.1 Feed-forward equaliser

The driver implements a 4-tap FFE with one post-emphasis and two pre-emphasis taps. The driver consists of five driving segments for the 4 taps of the FFE and for the amplitude tuning segment. The driver essentially drives four signal bits at once with their relative driving strengths controlled by the FFE settings. The driving strength of each segment is controlled by an analog signal generated by the biasing circuit.

Most of the previously reported SST designs construct the driver from a large number of smaller slices, each of which has its own predriver, e.g. [92], [93]. This

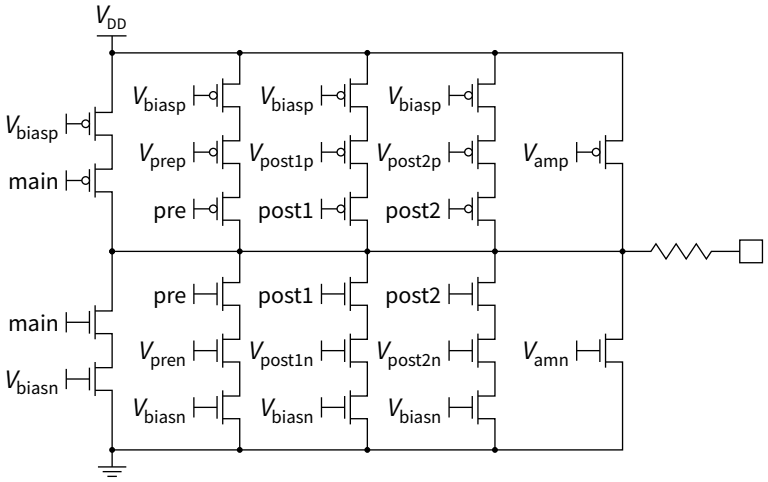


Figure 6.10: The schematic diagram of the designed SST driver.

precludes the use of higher resolution FFE tap values and complicates predriver construction.

6.7.2 Impedance tuning

The impedance of the driver is tuned using three different methods that work together. The first method uses the adjustment of the driver's bias voltages. The second and third methods adjust the termination resistor. The combination of these methods ensures that the output impedance is 50 Ω and that the driver's transistors are always in the triode region for high linearity.

The replica biasing circuit of Section 6.8 controls bias voltages. The second methods controls the value of the termination resistor by turning in or out 16 transmission gates connected in parallel with the resistor. The third method controls voltages that are supplied to transmission gates implementing finer control over the impedance of the transmission gates. Figure 6.12 shows the layout of the impedance-tuning transmission gates.

6.7.3 ESD protection

Diode-triggered SCR ESD [94] protection is employed. This ESD protection circuit includes the use of a T-coil [95]. The ESD protection circuit serves not only to protect internal circuitry from external ESD events, but also to significantly extend the bandwidth of the circuit as explained in [96], [97], [98].

A version of the same ESD protection circuit was designed with an asymmetric T-coil. In this design, the inductor's centre tap is moved away from a symmetric position to further improve the bandwidth characteristics of the transmitter. It was found that the asymmetric design may not be necessary for a 15 Gbps circuit but may be necessary for faster designs.

6.8 Driver biasing

The biasing circuit for this design was inspired by the designs in [99] and [100]. Several replicas of the entire driver are implemented inside the biasing circuit. Replicas use the same transistor widths as the driver but reduce the number of

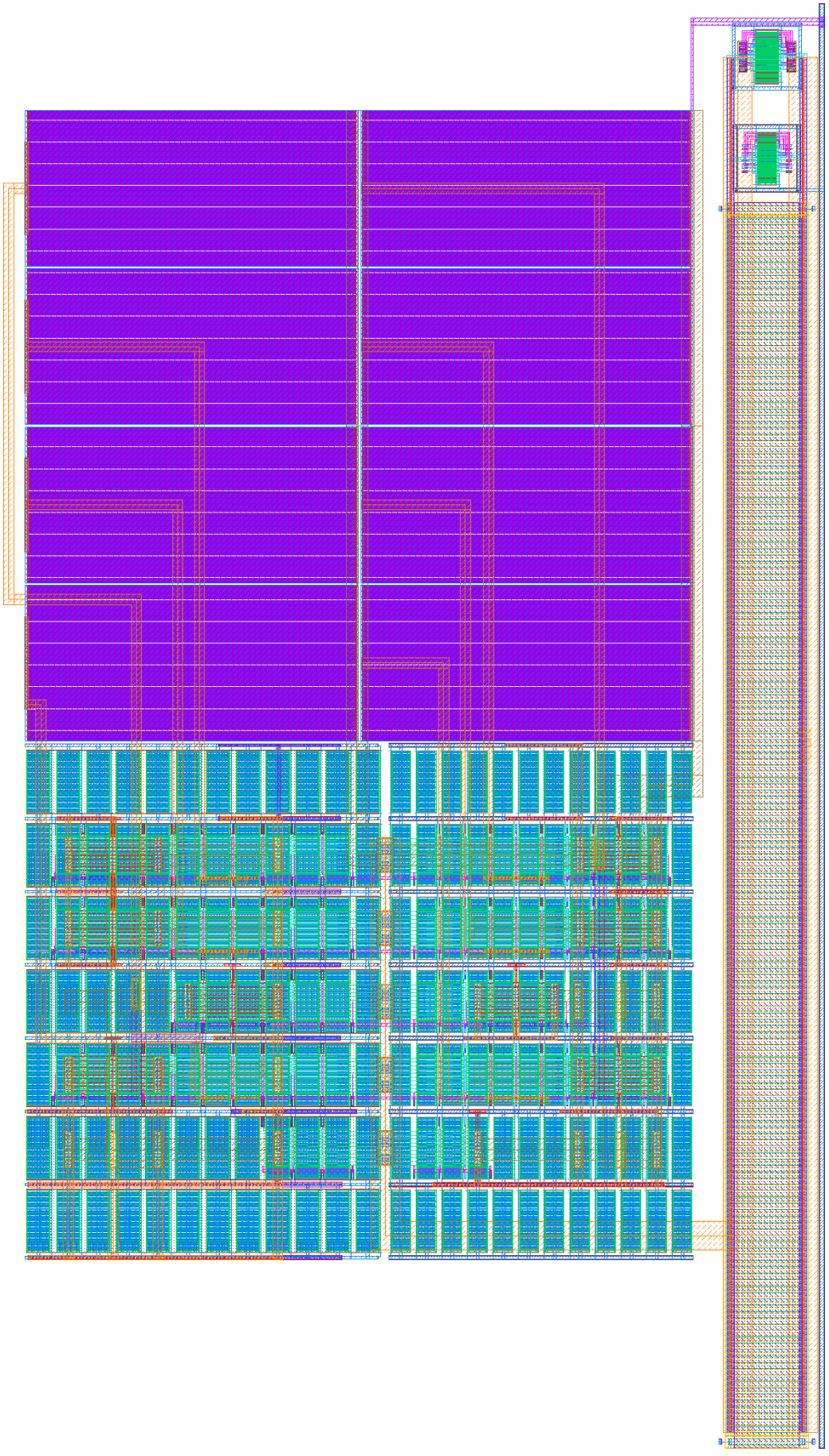


Figure 6.11: The layout of the implemented SST driver. The image shows one half of the pseudo-differential driver design.

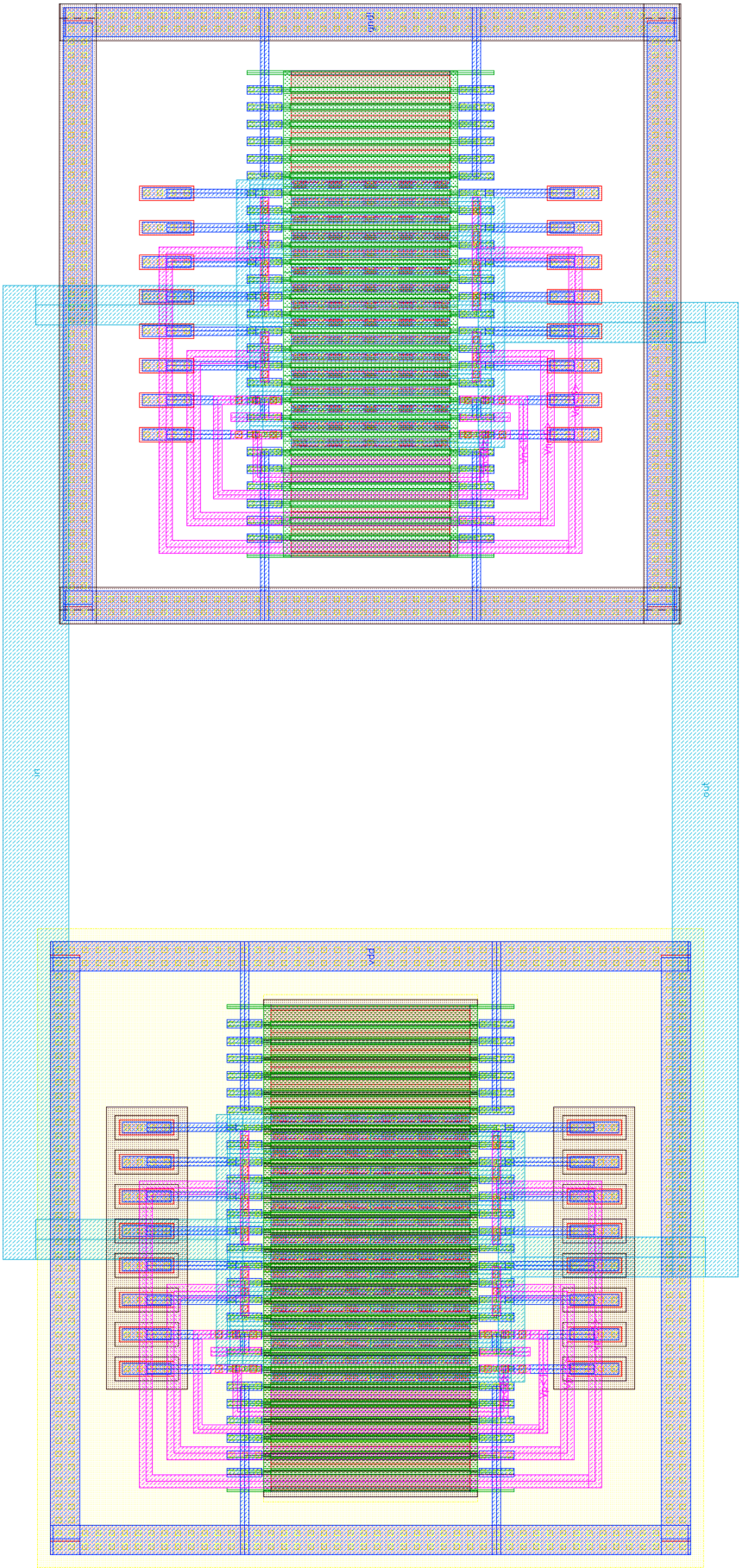


Figure 6.12: The layout of the transmission gates that can be used to regulate the driver's impedance.

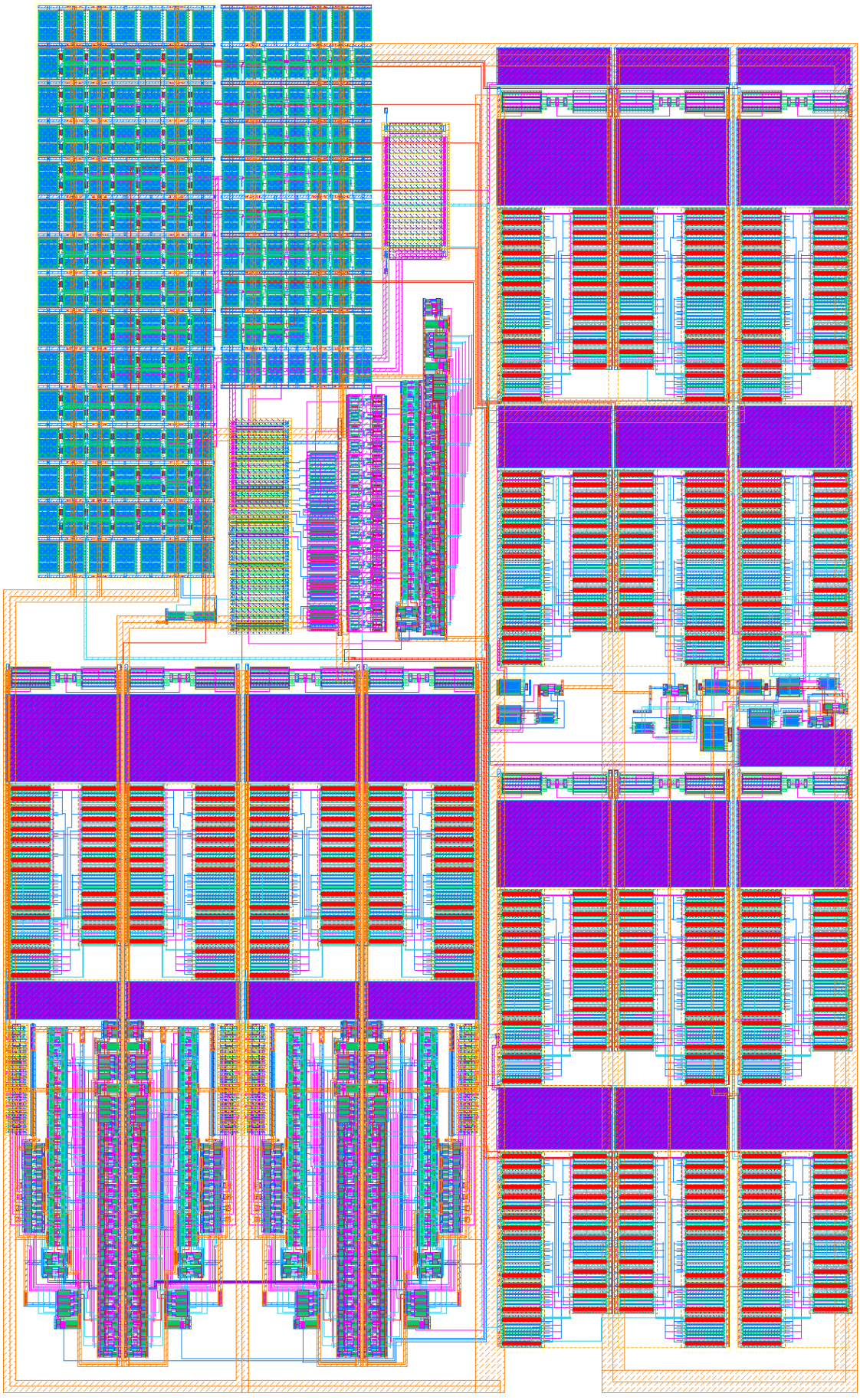


Figure 6.13: The layout of the bias generator circuit.

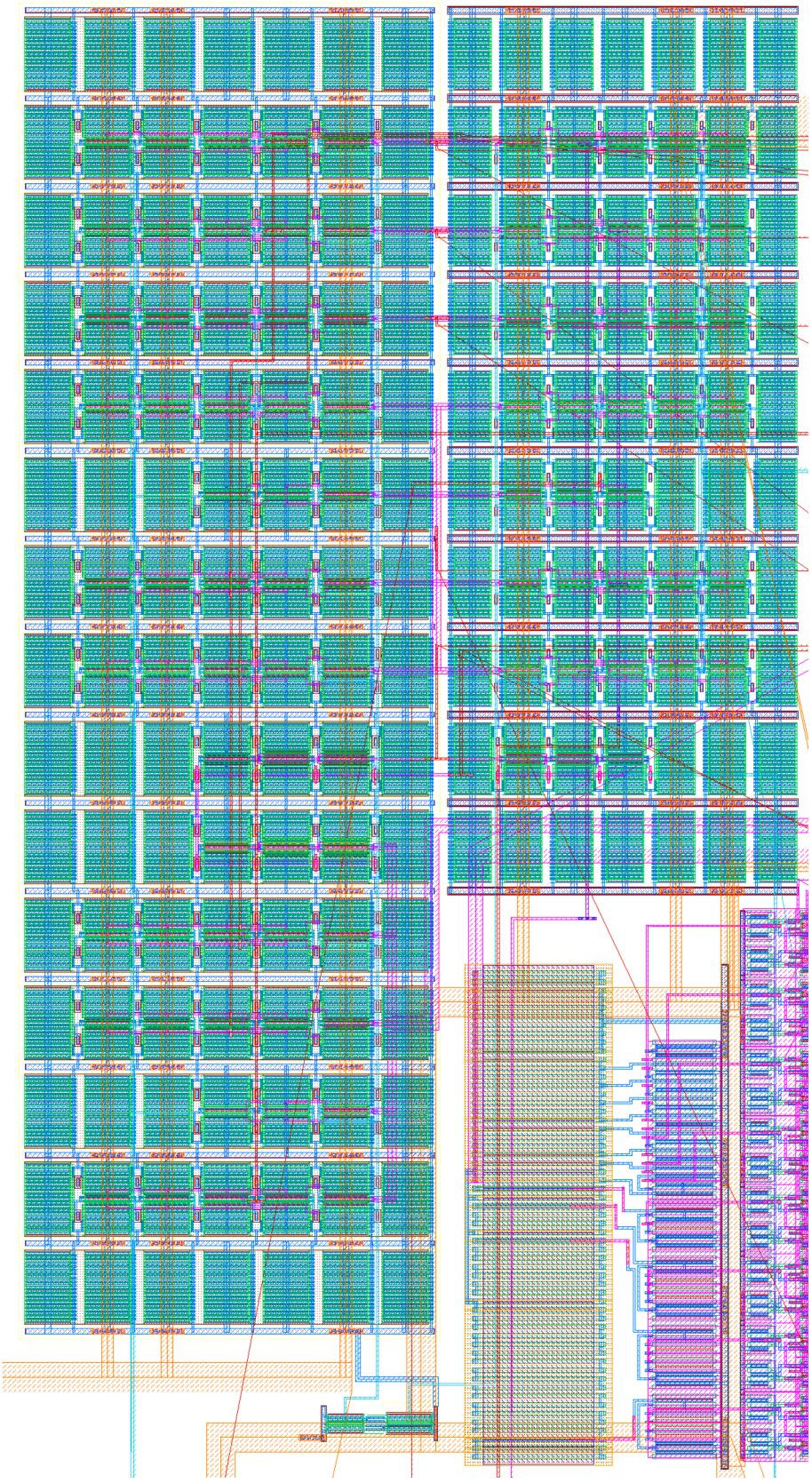


Figure 6.14: The layout view of the replica biasing transistors within the biasing circuit.

transistor fingers by a factor of 20. These biasing circuits ensure that for each of the driver's segments n-type and p-type circuit parts have equal impedance. They also match the overall driving strength of these segments to a tunable resistor, which is the first method of tuning the driver's impedance given in Section 6.7.2.

All FFE, impedance and amplitude tuning parameters are 8-bit numbers, which is higher than usual for SST driver designs. Several R-2R ladder DACs [101] were implemented to generate analog voltages from these numbers.

It was found from previous fabrications in 130 nm that standard voltage dividers are not always precise, thus an active voltage divider of [102] was implemented to provide a precise $0.5V_{DD}$ voltage reference for the voltage comparators.

The layout of the overall biasing circuit is shown in Figure 6.13. The view of the replica bias segments is shown in Figure 6.14. There are about two orders of magnitude more dummy transistors than actual replica biasing transistors for precision and good matching among the replica transistors.

6.9 Conclusion

High-performance SerDes transmitter design was implemented in the 28 nm CMOS technology. The implemented circuits include the voltage-mode source-series terminated driver. The driver of this type is generally more efficient than the common current-mode logic driver. The driver was designed to include a number of features including the 4-tap FFE with 8-bit taps, output signal amplitude tuning and impedance tuning. The impedance is tuned by adjusting the driving strength of the driver and by adjusting the termination resistor. The termination resistor can be controlled by connecting transmission gates in parallel with it and also by adjusting their controlling analog voltages, which enables very fine control of the driver's behaviour. The designed digital circuits include the serialiser whose circuits are capable of delivering higher performance and thus can be used in a more advanced future design.

Chapter 7

Conclusion

7.1 Summary

The thesis presented the work that stemmed from the research into the implementation of the X part of very large FX correlators. The main processing circuit of the correlator, namely the CMAC, was designed in Chapter 2 using the high-performance 28 nm CMOS technology. The designs achieved 1.94 mW and 3.27 mW at the clock rates of 2 GHz and 1.2 GHz for the 4-bit and 8-bit CMACs respectively. The design effort for Chapter 2 led to the creation of numerous circuit cells that then became the building blocks of other digital designs in this work.

CMACs have to be incorporated into an implementation of a correlator architecture in order to be useful. Chapter 3 considered the input-buffered correlator approach that can be implemented using standard design techniques and available IP blocks such as the RAM memory. After researching input-buffering, it was noticed that the standard CMAC can be improved to include more than one accumulator, which achieves a significant reduction in the number of input-buffer memory operations in an input-buffer architecture that was previously proposed for an ASIC implementation. More specifically, using 3 accumulators per CMAC was shown to reduce the utilisation of the internal memory of the processing unit by more than a third while reducing the CMAC's own power dissipation due to reduced switching activity within its multiplier circuits.

The research into the efficient design of digital circuits led to the effort of designing better accumulator circuits implemented as static flip-flops. Chapter 4 presented the designs of novel dual-edge triggered flip-flops that use C-elements to reduce their power dissipation due to glitches at their inputs. These glitches often happen in deep combinational logic circuits such as multipliers and large adder trees.

The idea of using multiple accumulators in every CMAC was extended to its maximum in Chapter 5 where each CMAC was considered to include as many accumulators as there are frequency channels. This approach removes the need for any input buffering yet it imposes high technical requirements for the memory that implements these accumulators. The proposed memory control circuits were implemented and prototyped in a 130 nm CMOS technology. These circuits can be used as control circuits for the eDRAM memory cells to improve upon eDRAM's performance and power dissipation.

The experience of designing and analysing circuit topologies gained in the design of CMACs and dual-edge-triggered flip-flops has been fully utilised and improved upon in the design of I/O circuits as given in Chapter 6. In a high per-

formance correlator with very data-intensive processing, the I/O implementation is just as important as the implementation of the processing circuits, and would usually be considerably more complicated. The chapter presented full-rate transmitter design achieves 15 Gbps and includes numerous analog circuits that were necessary to implement such a design.

7.2 Suggestions for future work

The presented work can be improved upon in several different ways. Chapters 2 and 3 implemented CMACs and proposed an improved input-buffered architecture that can utilise these CMACs. An implementation of the considered architecture could be used in the design of a general correlator which would be useful in a variety of present and future multi-antenna radio telescopes. The next logical step would be to design an actual implementation of this architecture, which would involve more digital design and more integration among the already designed circuits.

The work on the output-buffer memory in Chapter 5 could be promising mainly because it removes the complicated input buffer from the correlator implementation and replaces it with a per-CMAC output buffer. The presented memory control circuits can be implemented in a modern CMOS technology using actual DRAM cells to evaluate its performance and suitability for this task.

The I/O can be improved on in several ways. Firstly, 15 Gbps is far from the limit of this 28 nm technology. One relatively simple way of improving the I/O's performance is to create a half-rate version of the presented full-rate circuit which would double the achieved line data rate. Many of the existing I/O circuits were designed with this in mind as, for example, the serialiser, which already consists of two half-rate 33-to-1 serialisers. These smaller serialisers can be used on their own in a half-rate design. In a half-rate I/O circuit, the data is serialised to the line rate at the driver itself, which complicates the driver construction. Yet the overall SST driver architecture stays the same in this case.

Many of the presented circuits including the I/O were designed conservatively while following technology recommendations and manufacturability guidelines. It is possible that better practical circuit characteristics and performance can be achieved when a less stringent approach is adopted.

Bibliography

[1] T. L. Wilson, K. Rohlfs and S. Hüttemeister. *Tools of Radio Astronomy*. Astronomy and Astrophysics Library. Springer Berlin Heidelberg, 2013. ISBN: 9783642399503.

[2] A. R. Thompson, J. M. Moran and G. W. Swenson. *Interferometry and Synthesis in Radio Astronomy*. Astronomy and Astrophysics Library. Springer International Publishing, 2017. ISBN: 9783319444314. DOI: 10.1007/978-3-319-44431-4.

[3] L. R. Rabiner and B. Gold. *Theory and application of digital signal processing*. Signal Processing Series. Prentice-Hall, 1975. ISBN: 9780139141010.

[4] A. Papoulis. *The Fourier Integral and Its Applications*. Electronic Sciences Series. McGraw-Hill, 1962. ISBN: 9780070484474.

[5] S. B. Damelin and W. Miller. *The Mathematics of Signal Processing*. Cambridge Texts in Applied Mathematics. Cambridge University Press, 2012. ISBN: 9781107013223.

[6] J. Bunton. "Multi-resolution FX Correlator". In: *ALMA Memo Series*. 447. Feb. 2003. URL: http://library.nrao.edu/public/memos/alma/main/memo447.pdf.

[7] J. D. Romney. "Cross Correlators". In: *Synthesis Imaging in Radio Astronomy II*. Vol. 180. Astronomical Society of the Pacific Conference Series. 1999, p. 57. URL: http://adsabs.harvard.edu/abs/1999ASPC..180...57R.

[8] J. L. Yen. "The Role of Fast Fourier Transform Computers in Astronomy". In: *Astronomy and Astrophysics Supplement Series* 15 (June 1974), pp. 483–484.

[9] Y. Chikada et al. "A 6 × 320-MHz 1024-channel FFT cross-spectrum analyzer for radio astronomy". In: *Proceedings of the IEEE* 75.9 (Sept. 1987), pp. 1203–1210. ISSN: 0018-9219. DOI: 10.1109/PROC.1987.13873.

[10] Y. Chikada et al. "A Digital FFT Spectro-Correlator for Radio Astronomy". In: *Indirect Imaging. Measurement and Processing for Indirect Imaging.* (1984). Ed. by J. A. Roberts, p. 387.

[11] P. E. Dewdney. *SKA1 System Baseline Design*. SKA Organisation, Mar. 2013. URL: https://www.skatelescope.org/wp-content/uploads/2014/11/SKA-TEL-SKO-0000002-AG-BD-DD-Rev01-SKA1_System_Baseline_Design.pdf.

[12] P. E. Dewdney. *SKA1 System Baseline V2 Description*. SKA Organisation, Nov. 2015. URL: https://www.skatelescope.org/wp-content/uploads/2014/03/SKA-TEL-SKO-0000308_SKA1_System_Baseline_v2_DescriptionRev01-part-1-signed.pdf.

[13] NRAO. *Atacama Large Millimeter/submillimeter Array*. URL: https://public.nrao.edu/telescopes/alma/ (visited on 14/06/2018).

[14] NRAO. *Very Large Array*. URL: https://public.nrao.edu/telescopes/vla/ (visited on 14/06/2018).

[15] CSIRO. *ASKAP Radio Telescope*. URL: http://www.atnf.csiro.au/projects/askap/index.html (visited on 14/06/2018).

[16] *Murchison Widefield Array*. URL: http://www.mwatelescope.org/telescope (visited on 14/06/2018).

[17] ASTRON. *The Low-Frequency Array*. URL: http://www.astron.nl/general/lofar/lofar (visited on 14/06/2018).

[18] L. D'Addario. "Low-Power Correlator Architecture for the Mid-Frequency SKA". In: *SKA Memo Series*. 133. Mar. 2011. URL: https://www.skatelescope.org/uploaded/46888_133_memo_D'Addario.pdf.

[19] C. R. Baugh and B. A. Wooley. "A Two's Complement Parallel Array Multiplication Algorithm". In: *IEEE Transactions on Computers* C-22.12 (Dec. 1973), pp. 1045–1047. ISSN: 0018-9340. DOI: 10.1109/T-C.1973.223648.

[20] C. S. Wallace. "A Suggestion for a Fast Multiplier". In: *IEEE Transactions on Electronic Computers* EC-13.1 (Feb. 1964), pp. 14–17. ISSN: 0367-7508. DOI: 10.1109/PGEC.1964.263830.

[21] L. Dadda. "Some Schemes for Parallel Adders". In: *Alta Frequenza* 34 (1965), pp. 349–356.

[22] D. S. Phatak, T. Goff and I. Koren. "Constant-time addition and simultaneous format conversion based on redundant binary representations". In: *IEEE Transactions on Computers* 50.11 (Nov. 2001), pp. 1267–1278. ISSN: 0018-9340. DOI: 10.1109/12.966499.

[23] C. Nagendra, M. J. Irwin and R. M. Owens. "Area-time-power tradeoffs in parallel adders". In: *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing* 43.10 (Oct. 1996), pp. 689–702. ISSN: 1057-7130. DOI: 10.1109/82.539001.

[24] K. A. C. Bickerstaff, M. Schulte and E. E. Swartzlander. "Parallel reduced area multipliers". In: *Journal of VLSI signal processing systems for signal, image and video technology* 9.3 (Apr. 1995), pp. 181–191. ISSN: 0922-5773. DOI: 10.1007/BF02407084.

[25] K. A. C. Bickerstaff, M. Schulte and E. E. Swartzlander. "Reduced area multipliers". In: *Proceedings of International Conference on Application-Specific Array Processors*. Oct. 1993, pp. 478–489.

[26] H. Eriksson et al. "Multiplier reduction tree with logarithmic logic depth and regular connectivity". In: *2006 IEEE International Symposium on Circuits and Systems*. May 2006, pp. 5–8. DOI: 10.1109/ISCAS.2006.1692508.

[27] L. Dadda and V. Piuri. “Pipelined adders”. In: *IEEE Transactions on Computers* 45.3 (Mar. 1996), pp. 348–356. ISSN: 0018-9340. DOI: 10.1109/12.485573.

[28] M. Mano. *Digital Logic and Computer Design*. Prentice-Hall, 1979. ISBN: 9780132145107.

[29] Kuo-Hsing Cheng and Ven-Chieh Hsieh. “High efficiency 3-input XOR for low-voltage low-power high-speed applications”. In: *AP-ASIC ’99. The First IEEE Asia Pacific Conference on ASICs*. 1999, pp. 166–169. DOI: 10.1109/APASIC.1999.824054.

[30] M. Suzuki et al. “A 1.5-ns 32-b CMOS ALU in double pass-transistor logic”. In: *IEEE Journal of Solid-State Circuits* 28.11 (Nov. 1993), pp. 1145–1151. ISSN: 0018-9200. DOI: 10.1109/4.245595.

[31] D Markovi, B Nikoli and V.G Oklobdija. “A general method in synthesis of pass-transistor circuits”. In: *Microelectronics Journal* 31.11 (2000), pp. 991–998. ISSN: 0026-2692. DOI: 10.1016/S0026-2692(00)00088-4.

[32] R. Zimmermann and W. Fichtner. “Low-power logic styles: CMOS versus pass-transistor logic”. In: *IEEE Journal of Solid-State Circuits* 32.7 (July 1997), pp. 1079–1090. ISSN: 0018-9200. DOI: 10.1109/4.597298.

[33] G. B. Rosenberger. *Simultaneous carry adder*. US Patent 2,966,305. Dec. 1960.

[34] N. Nedovic and V. G. Oklobdzija. “Dual-edge triggered storage elements and clocking strategy for low-power systems”. In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 13.5 (May 2005), pp. 577–590. ISSN: 1063-8210. DOI: 10.1109/TVLSI.2005.844302.

[35] A. G. M. Strollo, E. Napoli and C. Cimino. “Analysis of power dissipation in double edge-triggered flip-flops”. In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 8.5 (Oct. 2000), pp. 624–629. ISSN: 1063-8210. DOI: 10.1109/92.894168.

[36] Andrea Bonetti et al. “Automated Integration of Dual-Edge Clocking for Low-Power Operation in Nanometer Nodes”. In: *ACM Transactions on Design Automation of Electronic Systems (TODAES)* 22.4 (May 2017), 62:1–62:20. ISSN: 1084-4309. DOI: 10.1145/3054744.

[37] R. Hossain, L. D. Wronski and A. Albicki. “Low power design using double edge triggered flip-flops”. In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 2.2 (June 1994), pp. 261–265. ISSN: 1063-8210. DOI: 10.1109/92.285754.

[38] J. Tschanz et al. “Comparative delay and energy of single edge-triggered and dual edge-triggered pulsed flip-flops for high-performance microprocessors”. In: *Low Power Electronics and Design, International Symposium on, 2001.* 2001, pp. 147–152. DOI: 10.1109/LPE.2001.945391.

[39] V. G. Oklobdzija. “Clocking in multi-GHz environment”. In: *2002 23rd International Conference on Microelectronics. Proceedings (Cat. No.02TH8595)*. Vol. 2. 2002, pp. 561–568. DOI: 10.1109/MIEL.2002.1003320.

[40] Larry R. D'Addario and Douglas Wang. "An Integrated Circuit for Radio Astronomy Correlators Supporting Large Arrays of Antennas". In: *Journal of Astronomical Instrumentation* 05.02 (2016). DOI: 10.1142/S2251171716500021.

[41] Stepan Lapshev and S. M. Rezaul Hasan. "On the architecture for the X part of a very large FX correlator using two-accumulator CMACs". In: *Experimental Astronomy* 41.1 (Feb. 2016), pp. 259–270. ISSN: 1572-9508. DOI: 10.1007/s10686-015-9489-3.

[42] L. de Souza et al. "A Radio Astronomy Correlator Optimized for the Xilinx Virtex-4 SX FPGA". In: *2007 International Conference on Field Programmable Logic and Applications*. Aug. 2007, pp. 62–67. DOI: 10.1109/FPL.2007.4380626.

[43] P. Zhao et al. "Low-Power Clock Branch Sharing Double-Edge Triggered Flip-Flop". In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 15.3 (Mar. 2007), pp. 338–345. ISSN: 1063-8210. DOI: 10.1109/TVLSI.2007.893623.

[44] D. E. Muller. *Theory of asynchronous circuits*. Internal Report no. 66. Digital Computer Laboratory. University of Illinois at Urbana-Champaign, Dec. 1955.

[45] Kees van Berkel. "Beware the isochronic fork". In: *Integration, the VLSI Journal* 13.2 (June 1992), pp. 103–128. ISSN: 0167-9260. DOI: 10.1016/0167-9260(92)90001-F.

[46] S. V. Devarapalli, P. Zarkesh-Ha and S. C. Suddarth. "A robust and low power dual data rate (DDR) flip-flop using c-elements". In: *2010 11th International Symposium on Quality Electronic Design (ISQED)*. Mar. 2010, pp. 147–150. DOI: 10.1109/ISQED.2010.5450403.

[47] A. Gago, R. Escano and J. A. Hidalgo. "Reduced implementation of D-type DET flip-flops". In: *IEEE Journal of Solid-State Circuits* 28.3 (Mar. 1993), pp. 400–402. ISSN: 0018-9200. DOI: 10.1109/4.210012.

[48] A. Bonetti, A. Teman and A. Burg. "An overlap-contention free true-single-phase clock dual-edge-triggered flip-flop". In: *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*. May 2015, pp. 1850–1853. DOI: 10.1109/ISCAS.2015.7169017.

[49] M. Alioto, E. Consoli and G. Palumbo. "Analysis and Comparison in the Energy-Delay-Area Domain of Nanometer CMOS Flip-Flops: Part I—Methodology and Design Strategies". In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 19.5 (May 2011), pp. 725–736. ISSN: 1063-8210. DOI: 10.1109/TVLSI.2010.2041376.

[50] M. Alioto, E. Consoli and G. Palumbo. "Analysis and Comparison in the Energy-Delay-Area Domain of Nanometer CMOS Flip-Flops: Part II—Results and Figures of Merit". In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 19.5 (May 2011), pp. 737–750. ISSN: 1063-8210. DOI: 10.1109/TVLSI.2010.2041377.

[51] V. Stojanovic and V. G. Oklobdzija. "Comparative analysis of master-slave latches and flip-flops for high-performance and low-power systems". In: *IEEE Journal of Solid-State Circuits* 34.4 (Apr. 1999), pp. 536–548. ISSN: 0018-9200. DOI: 10.1109/4.753687.

[52] M. Alioto, E. Consoli and G. Palumbo. "Analysis and comparison of variations in double edge triggered flip-flops". In: *2014 5th European Workshop on CMOS Variability (VARI)*. Sept. 2014, pp. 1–6. DOI: 10.1109/VARI.2014.6957076.

[53] M. Shams, J. C. Ebergen and M. I. Elmasry. "Modeling and comparing CMOS implementations of the C-element". In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 6.4 (Dec. 1998), pp. 563–567. ISSN: 1063-8210. DOI: 10.1109/92.736128.

[54] B. Carlson. "The Giant Systolic Array (GSA) Straw-man Proposal for a Multi-Mega Baseline Correlator for the SKA". In: *SKA Memo Series*. 127. Aug. 2010. URL: https://www.skatelescope.org/uploaded/14974_127_Memo_Carlson.pdf.

[55] K. Yamanaka and M. Kimura. *Sequential access memory and its operation method*. US Patent 5,444,660. Aug. 1995.

[56] Y. Imamura et al. *Sequential access memory that can have circuit area reduced*. US Patent 5,535,170. July 1996.

[57] R. Zimmerman. *Sequential access memory elements*. US Patent 8,208,314. June 2012.

[58] M. Bazes. "Two novel fully complementary self-biased CMOS differential amplifiers". In: *IEEE Journal of Solid-State Circuits* 26.2 (Feb. 1991), pp. 165–168. ISSN: 0018-9200. DOI: 10.1109/4.68134.

[59] Sanghoon Hong et al. "Low-voltage DRAM sensing scheme with offset-cancellation sense amplifier". In: *IEEE Journal of Solid-State Circuits* 37.10 (Oct. 2002), pp. 1356–1360. ISSN: 0018-9200. DOI: 10.1109/JSSC.2002.803052.

[60] J. F. Ryan and B. H. Calhoun. "Minimizing Offset for Latching Voltage-Mode Sense Amplifiers for Sub-Threshold Operation". In: *9th International Symposium on Quality Electronic Design (isqed 2008)*. Mar. 2008, pp. 127–132. DOI: 10.1109/ISQED.2008.4479712.

[61] R. Singh and N. Bhat. "An offset compensation technique for latch type sense amplifiers in high-speed low-power SRAMs". In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 12.6 (June 2004), pp. 652–657. ISSN: 1063-8210. DOI: 10.1109/TVLSI.2004.827566.

[62] R. Pelliconi et al. "Power efficient charge pump in deep submicron standard CMOS technology". In: *IEEE Journal of Solid-State Circuits* 38.6 (June 2003), pp. 1068–1071. ISSN: 0018-9200. DOI: 10.1109/JSSC.2003.811991.

[63] P. Favrat, P. Deval and M. J. Declercq. "A high-efficiency CMOS voltage doubler". In: *IEEE Journal of Solid-State Circuits* 33.3 (Mar. 1998), pp. 410–416. ISSN: 0018-9200. DOI: 10.1109/4.661206.

[64] D. Baderna et al. “Efficiency comparison between doubler and Dickson charge pumps”. In: *2005 IEEE International Symposium on Circuits and Systems*. Vol. 2. May 2005, pp. 1891–1894. DOI: 10.1109/ISCAS.2005.1464981.

[65] R. Thewes et al. *Pointer circuit with low surface requirement high speed and low power loss*. US Patent 6,097,661. Aug. 2000.

[66] “Electrical Characteristics of Low Voltage Differential Signaling (LVDS) Interface Circuits”. In: *TIA-644* (Feb. 2001).

[67] *Low Jitter VCXO*. RVX2520R. Rakon. URL: http://www.rakon.com/products/families/download/file?fid=39.269.

[68] J. Cao et al. “A 500 mW ADC-Based CMOS AFE With Digital Calibration for 10 Gb/s Serial Links Over KR-Backplane and Multimode Fiber”. In: *IEEE Journal of Solid-State Circuits* 45.6 (June 2010), pp. 1172–1185. ISSN: 0018-9200. DOI: 10.1109/JSSC.2010.2047473.

[69] K. Shu and E. Sanchez-Sinencio. *CMOS PLL Synthesizers: Analysis and Design*. The Springer International Series in Engineering and Computer Science. Springer US, 2005. ISBN: 9780387236698.

[70] T. Seong, J. J. Kim and J. Choi. “Analysis and Design of a Core-Size-Scalable Low Phase Noise *LC* -VCO for Multi-Standard Cellular Transceivers”. In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 62.3 (Mar. 2015), pp. 781–790. ISSN: 1549-8328. DOI: 10.1109/TCSI.2014.2382191.

[71] X. Tang et al. “A wideband 0.13 μm CMOS LC-VCO for IMT-Advanced and UWB applications”. In: *2012 IEEE MTT-S International Microwave Workshop Series on Millimeter Wave Wireless Technology and Applications*. Sept. 2012, pp. 1–4. DOI: 10.1109/IMWS2.2012.6338194.

[72] K. Kanda et al. “A Single-40 Gb/s Dual-20 Gb/s Serializer IC With SFI-5.2 Interface in 65 nm CMOS”. In: *IEEE Journal of Solid-State Circuits* 44.12 (Dec. 2009), pp. 3580–3589. ISSN: 0018-9200. DOI: 10.1109/JSSC.2009.2031030.

[73] M. D. M. Hershenson et al. “Design and optimization of LC oscillators”. In: *1999 IEEE/ACM International Conference on Computer-Aided Design. Digest of Technical Papers (Cat. No.99CH37051)*. Nov. 1999, pp. 65–69. DOI: 10.1109/ICCAD.1999.810623.

[74] D. Martynenko, G. Fischer and O. Klymenko. “Implementation of the ultra-low power load-independent LC VCO”. In: *2012 IEEE International Conference on Circuits and Systems (ICCAS)*. Oct. 2012, pp. 27–31. DOI: 10.1109/ICCircuitsAndSystems.2012.6408299.

[75] N. Nedovic et al. “A 3 Watt 39.8—44.6 Gb/s Dual-Mode SFI5.2 SerDes Chip Set in 65 nm CMOS”. In: *IEEE Journal of Solid-State Circuits* 45.10 (Oct. 2010), pp. 2016–2029. ISSN: 0018-9200. DOI: 10.1109/JSSC.2010.2057970.

[76] J. W. Jung and B. Razavi. “A 25-Gb/s 5-mW CMOS CDR/Deserializer”. In: *IEEE Journal of Solid-State Circuits* 48.3 (Mar. 2013), pp. 684–697. ISSN: 0018-9200. DOI: 10.1109/JSSC.2013.2237692.

[77] L. Henrickson et al. “Low-power fully integrated 10-Gb/s SONET/SDH transceiver in 0.13- mu;m CMOS”. In: *IEEE Journal of Solid-State Circuits* 38.10 (Oct. 2003), pp. 1595–1601. ISSN: 0018-9200. DOI: 10.1109/JSSC.2003.817586.

[78] M. Ida, N. Kato and T. Takada. “A 4 Gbits/s GaAs 16:1 multiplexer/1:16 demultiplexer LSI chip”. In: *IEEE Journal of Solid-State Circuits* 24.4 (Aug. 1989), pp. 928–932. ISSN: 0018-9200. DOI: 10.1109/4.34073.

[79] W. Y. Tsai et al. “A Novel Low Gate-Count Pipeline Topology With Multiplexer-Flip-Flops for Serial Link”. In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 59.11 (Nov. 2012), pp. 2600–2610. ISSN: 1549-8328. DOI: 10.1109/TCSI.2012.2206494.

[80] C. Wortman, C.H. Lee and H. Ngo. *Configurable multi-lane scrambler for flexible protocol support*. US Patent 9,367,509. June 2016.

[81] “ISO/IEC/IEEE International Standard for Ethernet”. In: *ISO/IEC/IEEE 8802-3:2014(E)* (Apr. 2014). DOI: 10.1109/IEEESTD.2014.6781545.

[82] J. E. Savage. “Some simple self-synchronizing digital data scramblers”. In: *The Bell System Technical Journal* 46.2 (Feb. 1967), pp. 449–487. ISSN: 0005-8580. DOI: 10.1002/j.1538-7305.1967.tb01066.x.

[83] J. Jung et al. “Efficient Parallel Architecture for Linear Feedback Shift Registers”. In: *IEEE Transactions on Circuits and Systems II: Express Briefs* 62.11 (Nov. 2015), pp. 1068–1072. ISSN: 1549-7747. DOI: 10.1109/TCSII.2015.2456294.

[84] Chih-Hsien Lin et al. “Parallel scrambler for high-speed applications”. In: *IEEE Transactions on Circuits and Systems II: Express Briefs* 53.7 (July 2006), pp. 558–562. ISSN: 1549-7747. DOI: 10.1109/TCSII.2006.875316.

[85] George Marsaglia. “Xorshift RNGs”. In: *Journal of Statistical Software, Articles* 8.14 (2003), pp. 1–6. ISSN: 1548-7660. DOI: 10.18637/jss.v008.i14.

[86] Sebastiano Vigna. “An Experimental Exploration of Marsaglia’s Xorshift Generators, Scrambled”. In: *ACM Trans. Math. Softw.* 42.4 (June 2016), 30:1–30:23. ISSN: 0098-3500. DOI: 10.1145/2845077.

[87] M. Kossel et al. “A T-Coil-Enhanced 8.5 Gb/s High-Swing SST Transmitter in 65 nm Bulk CMOS With $<$ 16 dB Return Loss Over 10 GHz Bandwidth”. In: *IEEE Journal of Solid-State Circuits* 43.12 (Dec. 2008), pp. 2905–2920. ISSN: 0018-9200. DOI: 10.1109/JSSC.2008.2006230.

[88] Y. Lu et al. “Design and Analysis of Energy-Efficient Reconfigurable Pre-Emphasis Voltage-Mode Transmitters”. In: *IEEE Journal of Solid-State Circuits* 48.8 (Aug. 2013), pp. 1898–1909. ISSN: 0018-9200. DOI: 10.1109/JSSC.2013.2258790.

[89] K. Kaviani et al. “A 0.4-mW/Gb/s Near-Ground Receiver Front-End With Replica Transconductance Termination Calibration for a 16-Gb/s Source-Series Terminated Transceiver”. In: *IEEE Journal of Solid-State Circuits* 48.3 (Mar. 2013), pp. 636–648. ISSN: 0018-9200. DOI: 10.1109/JSSC.2013.2242714.

[90] J. F. Bulzacchelli et al. "A 28-Gb/s 4-Tap FFE/15-Tap DFE Serial Link Transceiver in 32-nm SOI CMOS Technology". In: *IEEE Journal of Solid-State Circuits* 47.12 (Dec. 2012), pp. 3232–3248. ISSN: 0018-9200. DOI: 10.1109/JSSC.2012.2216414.

[91] J. F. Bulzacchelli. "Design techniques for CMOS backplane transceivers approaching 30-Gb/s data rates". In: *Proceedings of the IEEE 2013 Custom Integrated Circuits Conference*. Sept. 2013, pp. 1–8. DOI: 10.1109/CICC.2013.6658405.

[92] W. D. Dettloff et al. "A 32mW 7.4Gb/s protocol-agile source-series-terminated transmitter in 45nm CMOS SOI". In: *2010 IEEE International Solid-State Circuits Conference - (ISSCC)*. Feb. 2010, pp. 370–371. DOI: 10.1109/ISSCC.2010.5433825.

[93] R. A. Philpott et al. "A 20Gb/s SerDes transmitter with adjustable source impedance and 4-tap feed-forward equalization in 65nm bulk CMOS". In: *2008 IEEE Custom Integrated Circuits Conference*. Sept. 2008, pp. 623–626. DOI: 10.1109/CICC.2008.4672163.

[94] M. P. J. Mergens et al. "Speed optimized diode-triggered SCR (DTSCR) for RF ESD protection of ultra-sensitive IC nodes in advanced technologies". In: *IEEE Transactions on Device and Materials Reliability* 5.3 (Sept. 2005), pp. 532–542. ISSN: 1530-4388. DOI: 10.1109/TDMR.2005.853510.

[95] B. Razavi. "The Bridged T-Coil [A Circuit for All Seasons]". In: *IEEE Solid-State Circuits Magazine* 7.4 (Fall 2015), pp. 9–13. ISSN: 1943-0582. DOI: 10.1109/MSSC.2015.2474258.

[96] M. S. Keel and E. Rosenbaum. "CDM-Reliable T-Coil Techniques for a 25-Gb/s Wireline Receiver Front-End". In: *IEEE Transactions on Device and Materials Reliability* 16.4 (Dec. 2016), pp. 513–520. ISSN: 1530-4388. DOI: 10.1109/TDMR.2016.2594281.

[97] H. M. Cheema et al. "A 40 GHz, broadband, highly linear amplifier, employing T-coil bandwidth extension technique". In: *2008 IEEE Radio Frequency Integrated Circuits Symposium*. June 2008, pp. 645–648. DOI: 10.1109/RFIC.2008.4561520.

[98] E. Pillai and J. Weiss. "Novel T-Coil Structure and Implementation in a 6.4-Gb/s CMOS Receiver to Meet Return Loss Specifications". In: *2007 Proceedings 57th Electronic Components and Technology Conference*. May 2007, pp. 147–153. DOI: 10.1109/ECTC.2007.373789.

[99] S. Chen et al. "A novel SST transmitter with mutually decoupled impedance self-calibration and equalization". In: *2011 IEEE International Symposium of Circuits and Systems (ISCAS)*. May 2011, pp. 173–176. DOI: 10.1109/ISCAS.2011.5937529.

[100] Y. Lu et al. "A 10Gb/s 10mW 2-tap reconfigurable pre-emphasis transmitter in 65nm LP CMOS". In: *Proceedings of the IEEE 2012 CCIC*. Sept. 2012, pp. 1–4. DOI: 10.1109/CICC.2012.6330581.

[101] M. P. Kennedy. "On the robustness of R-2R ladder DACs". In: *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications* 47.2 (Feb. 2000), pp. 109–116. ISSN: 1057-7122. DOI: 10.1109/81.828565.

[102] M.A. Wyatt. *Precision voltage divider*. US Patent 5,030,848. July 1991.

Appendix A

CMAC multiplier netlists

This appendix includes the netlists of the adder tree implementations for the 4-bit and 8-bit CMAC designs. The implemented trees were given in illustrations in Figures 2.5 and 2.6 for the 4-bit and 8-bit CMACs respectively. The input into the circuits consists of the four numbers a, b, c and d that correspond to the four variables in (2.2). Throughout the designs, "b" after the signal name is used to indicate inversion on that signal.

There are several types of signals within adder trees. Partial products have the format PX_Y, where X differentiates partial products from different multiplications and Y denotes the weight of the given bit. Bit weights start from 0 for the least significant weight. Within the tree, sum signals are denoted as SX_Y, where X and Y denote the tree stage and the signal weight respectively. Tree stages are numbered sequentially and start from 0. The carry signals are denoted as CX_Y_Z, where X is the tree stage, Y is the bit weight at which the carry was generated and Z is the actual weight of the carry. R<X> stands for a bit of the final result of weight X. In case there are multiple signals for the same tree stage and weight, postfix <X> is used to differentiate between such signals.

A.1 4-bit CMAC

```
********************************************************************************
* Library Name: CMACs
* Cell Name:    DETFF_Re_9_CMAC
* View Name:    schematic
********************************************************************************

.SUBCKT DETFF_Re_9_CMAC CK_in D<0> D<1> D<2> D<3> D<4> D<5> D<6> D<7> D<8>
+ Q<0> Q<1> Q<2> Q<3> Q<4> Q<5> Q<6> Q<7> Q<8> Qb<0> Qb<1> Qb<2> Qb<3> Qb<4>
+ Qb<5> Qb<6> Qb<7> Qb<8> Rb_in vdd
*.PININFO CK_in:I D<0>:I D<1>:I D<2>:I D<3>:I D<4>:I D<5>:I D<6>:I D<7>:I
*.PININFO D<8>:I Rb_in:I Q<0>:O Q<1>:O Q<2>:O Q<3>:O Q<4>:O Q<5>:O Q<6>:O
*.PININFO Q<7>:O Q<8>:O Qb<0>:O Qb<1>:O Qb<2>:O Qb<3>:O Qb<4>:O Qb<5>:O
*.PININFO Qb<6>:O Qb<7>:O Qb<8>:O vdd:B
XI23 D<3> P Pb Q<3> Qb<3> R vdd / pulsed_latch_S
XI27 D<8> P Pb Q<8> Qb<8> R vdd / pulsed_latch_S
XI42 Rb_in Rb R vdd / buffer
XI20 CK_in CK CKb vdd / buffer
XI22 CK CKb P Pb vdd / pulse_generator
XI30 D<7> P Pb Q<7> Qb<7> Rb vdd / pulsed_latch
XI24 D<4> P Pb Q<4> Qb<4> Rb vdd / pulsed_latch
XI25 D<6> P Pb Q<6> Qb<6> Rb vdd / pulsed_latch
XI26 D<5> P Pb Q<5> Qb<5> Rb vdd / pulsed_latch
```

```
XI2 CK CKb D<2> Q<2> Qb<2> R vdd / DETFF_S
XI1 CK CKb D<1> Q<1> Qb<1> R vdd / DETFF_S
XI0 CK CKb D<0> Q<0> Qb<0> R vdd / DETFF_S
.ENDS

********************************************************************************
* Library Name: CMACs
* Cell Name:    DETFF_Im_9_CMAC
* View Name:    schematic
********************************************************************************

.SUBCKT DETFF_Im_9_CMAC CK_in D<0> D<1> D<2> D<3> D<4> D<5> D<6> D<7> D<8>
+ Q<0> Q<1> Q<2> Q<3> Q<4> Q<5> Q<6> Q<7> Q<8> Qb<0> Qb<1> Qb<2> Qb<3> Qb<4>
+ Qb<5> Qb<6> Qb<7> Qb<8> Rb_in vdd
*.PININFO CK_in:I D<0>:I D<1>:I D<2>:I D<3>:I D<4>:I D<5>:I D<6>:I D<7>:I
*.PININFO D<8>:I Rb_in:I Q<0>:O Q<1>:O Q<2>:O Q<3>:O Q<4>:O Q<5>:O Q<6>:O
*.PININFO Q<7>:O Q<8>:O Qb<0>:O Qb<1>:O Qb<2>:O Qb<3>:O Qb<4>:O Qb<5>:O
*.PININFO Qb<6>:O Qb<7>:O Qb<8>:O vdd:B
XI27 D<8> P Pb Q<8> Qb<8> R vdd / pulsed_latch_S
XI42 Rb_in Rb R vdd / buffer
XI20 CK_in CK CKb vdd / buffer
XI22 CK CKb P Pb vdd / pulse_generator
XI30 D<7> P Pb Q<7> Qb<7> Rb vdd / pulsed_latch
XI23 D<3> P Pb Q<3> Qb<3> Rb vdd / pulsed_latch
XI24 D<4> P Pb Q<4> Qb<4> Rb vdd / pulsed_latch
XI25 D<6> P Pb Q<6> Qb<6> Rb vdd / pulsed_latch
XI26 D<5> P Pb Q<5> Qb<5> Rb vdd / pulsed_latch
XI2 CK CKb D<2> Q<2> Qb<2> R vdd / DETFF_S
XI1 CK CKb D<1> Q<1> Qb<1> R vdd / DETFF_S
XI0 CK CKb D<0> Q<0> Qb<0> R vdd / DETFF_S
.ENDS

********************************************************************************
* Library Name: CMACs
* Cell Name:    CMUL_4bit_Re
* View Name:    schematic
********************************************************************************

.SUBCKT CMUL_4bit_Re CK_in Resetb Y<8> Y<7> Y<6> Y<5> Y<4> Y<3> Y<2> Y<1> Y<0>
+ Yb<8> Yb<7> Yb<6> Yb<5> Yb<4> Yb<3> Yb<2> Yb<1> Yb<0> a<3> a<2> a<1> a<0>
+ ab<3> ab<2> ab<1> ab<0> b<3> b<2> b<1> b<0> bb<3> bb<2> bb<1> bb<0> c<3>
+ c<2> c<1> c<0> cb<3> cb<2> cb<1> cb<0> d<3> d<2> d<1> d<0> db<3> db<2> db<1>
+ db<0> vdd
*.PININFO CK_in:I Resetb:I a<3>:I a<2>:I a<1>:I a<0>:I ab<3>:I ab<2>:I ab<1>:I
*.PININFO ab<0>:I b<3>:I b<2>:I b<1>:I b<0>:I bb<3>:I bb<2>:I bb<1>:I bb<0>:I
*.PININFO c<3>:I c<2>:I c<1>:I c<0>:I cb<3>:I cb<2>:I cb<1>:I cb<0>:I d<3>:I
*.PININFO d<2>:I d<1>:I d<0>:I db<3>:I db<2>:I db<1>:I db<0>:I Y<8>:O Y<7>:O
*.PININFO Y<6>:O Y<5>:O Y<4>:O Y<3>:O Y<2>:O Y<1>:O Y<0>:O Yb<8>:O Yb<7>:O
*.PININFO Yb<6>:O Yb<5>:O Yb<4>:O Yb<3>:O Yb<2>:O Yb<1>:O Yb<0>:O vdd:B
XI37 d<3> b<3> P1_6 vdd / nand2
XI33 d<2> b<2> P1_4<1> vdd / nand2
XI30 d<2> b<1> P1_3<2> vdd / nand2
XI29 d<2> b<0> P1_2<2> vdd / nand2
XI28 d<1> b<2> P1_3<1> vdd / nand2
XI27 d<1> b<1> P1_2<1> vdd / nand2
XI26 d<1> b<0> P1_1<1> vdd / nand2
XI24 d<0> b<2> P1_2<0> vdd / nand2
XI23 d<0> b<1> P1_1<0> vdd / nand2
XI22 d<0> b<0> P1_0 vdd / nand2
XI21 a<3> c<3> P0_6 vdd / nand2
XI11 a<2> c<2> P0_4<1> vdd / nand2
XI10 a<2> c<1> P0_3<2> vdd / nand2
```

```
XI9 a<2> c<0> P0_2<2> vdd / nand2
XI7 a<1> c<2> P0_3<1> vdd / nand2
XI6 a<1> c<1> P0_2<1> vdd / nand2
XI5 a<1> c<0> P0_1<1> vdd / nand2
XI2 a<0> c<2> P0_2<0> vdd / nand2
XI1 a<0> c<1> P0_1<0> vdd / nand2
XI0 a<0> c<0> P0_0 vdd / nand2
XI64 CK_in Rb<0> Rb<1> Rb<2> Rb<3> R<4> R<5> R<6> R<7> Rb<8> Yb<0> Yb<1> Yb<2>
+ Yb<3> Y<4> Y<5> Y<6> Y<7> Yb<8> Y<0> Y<1> Y<2> Y<3> Yb<4> Yb<5> Yb<6> Yb<7>
+ Y<8> Resetb vdd / DETFF_Re_9_CMAC
XI58 C2_2_3b S2_3b C3_3_4b Rb<3> vdd / HA_pt
XI54 C1_1_2b S1_2b C2_2_3b Rb<2> vdd / HA_pt
XI38 P0_0 P1_0 C0_0_1b Rb<0> vdd / HA_pt
XI61 S1_6b C2_5_6b C1_5_6b C3_6_7b S3_6b vdd / FA_pt
XI60 S2_5b C2_4_5b C1_4_5b C3_5_6b S3_5b vdd / FA_pt
XI59 S2_4b C2_3_4b C1_3_4b<1> C3_4_5b S3_4b vdd / FA_pt
XI57 S1_5b C0_4_5b<0> C0_4_5b<1> C2_5_6b S2_5b vdd / FA_pt
XI56 S1_4b C1_3_4b<0> C0_3_4b<1> C2_4_5b S2_4b vdd / FA_pt
XI55 S1_3b<0> S1_3b<1> C1_2_3b C2_3_4b S2_3b vdd / FA_pt
XI53 C0_5_6b P0_6 P1_6 C1_6_7b S1_6b vdd / FA_pt
XI51 S0_4b<0> S0_4b<1> C0_3_4b<0> C1_4_5b S1_4b vdd / FA_pt
XI50 P1_3<3> C0_2_3b<0> C0_2_3b<1> C1_3_4b<1> S1_3b<1> vdd / FA_pt
XI49 S0_3b<0> S0_3b<1> P1_3<2> C1_3_4b<0> S1_3b<0> vdd / FA_pt
XI48 S0_2b<0> S0_2b<1> C0_1_2b C1_2_3b S1_2b vdd / FA_pt
XI47 S0_1b C0_0_1b P1_1<1> C1_1_2b Rb<1> vdd / FA_pt
XI46 P0_5<0> P0_5<1> P1_5<0> C0_5_6b S0_5b vdd / FA_pt
XI45 P1_4<0> P1_4<1> P1_4<2> C0_4_5b<1> S0_4b<1> vdd / FA_pt
XI44 P0_4<0> P0_4<1> P0_4<2> C0_4_5b<0> S0_4b<0> vdd / FA_pt
XI43 P0_3<3> P1_3<0> P1_3<1> C0_3_4b<1> S0_3b<1> vdd / FA_pt
XI42 P0_3<0> P0_3<1> P0_3<2> C0_3_4b<0> S0_3b<0> vdd / FA_pt
XI41 P1_2<0> P1_2<1> P1_2<2> C0_2_3b<1> S0_2b<1> vdd / FA_pt
XI40 P0_2<0> P0_2<1> P0_2<2> C0_2_3b<0> S0_2b<0> vdd / FA_pt
XI39 P0_1<0> P0_1<1> P1_1<0> C0_1_2b S0_1b vdd / FA_pt
XI52 P1_5<1> S0_5b C1_5_6b S1_5b vdd / HA_add_1_pt
XI62 C3_3_4b C3_4_5b C3_5_6b C3_6_7b S3_4b S3_5b S3_6b C1_6_7b LL Rb<8> R<4>
+ R<5> R<6> R<7> vdd / CLA4
MP3 net014 net014 vdd vdd pfet m=1 w=269n l=30n nf=1.0 pccrit=1 as=26.9f
+ ad=26.9f ps=738n pd=738n sa=115n sb=115n sd=100n ptwell=0 ngcon=1 p_la=0
MN1 LL net014 gnd! gnd! nfet m=1 w=180n l=30n nf=1.0 pccrit=1 as=18f ad=18f
+ ps=560n pd=560n sa=115n sb=115n sd=100n ptwell=0 ngcon=1 p_la=0
XI36 db<3> bb<2> P1_5<1> vdd / nor2_alt
XI35 db<3> bb<1> P1_4<2> vdd / nor2_alt
XI34 db<2> bb<3> P1_5<0> vdd / nor2_alt
XI32 db<1> bb<3> P1_4<0> vdd / nor2_alt
XI31 db<3> bb<0> P1_3<3> vdd / nor2_alt
XI25 db<0> bb<3> P1_3<0> vdd / nor2_alt
XI17 ab<3> cb<0> P0_3<3> vdd / nor2_alt
XI19 ab<3> cb<2> P0_5<1> vdd / nor2_alt
XI18 ab<3> cb<1> P0_4<2> vdd / nor2_alt
XI12 ab<2> cb<3> P0_5<0> vdd / nor2_alt
XI8 ab<1> cb<3> P0_4<0> vdd / nor2_alt
XI4 ab<0> cb<3> P0_3<0> vdd / nor2_alt
.ENDS

********************************************************************************
* Library Name: CMACs
* Cell Name:    CMUL_4bit_Im
* View Name:    schematic
********************************************************************************

.SUBCKT CMUL_4bit_Im CK_in Resetb Y<8> Y<7> Y<6> Y<5> Y<4> Y<3> Y<2> Y<1> Y<0>
+ Yb<8> Yb<7> Yb<6> Yb<5> Yb<4> Yb<3> Yb<2> Yb<1> Yb<0> a<3> a<2> a<1> a<0>
```

```
+ ab<3> ab<2> ab<1> ab<0> b<3> b<2> b<1> b<0> bb<3> bb<2> bb<1> bb<0> c<3>
+ c<2> c<1> c<0> cb<3> cb<2> cb<1> cb<0> d<3> d<2> d<1> d<0> db<3> db<2> db<1>
+ db<0> vdd
*.PININFO CK_in:I Resetb:I a<3>:I a<2>:I a<1>:I a<0>:I ab<3>:I ab<2>:I ab<1>:I
*.PININFO ab<0>:I b<3>:I b<2>:I b<1>:I b<0>:I bb<3>:I bb<2>:I bb<1>:I bb<0>:I
*.PININFO c<3>:I c<2>:I c<1>:I c<0>:I cb<3>:I cb<2>:I cb<1>:I cb<0>:I d<3>:I
*.PININFO d<2>:I d<1>:I d<0>:I db<3>:I db<2>:I db<1>:I db<0>:I Y<8>:O Y<7>:O
*.PININFO Y<6>:O Y<5>:O Y<4>:O Y<3>:O Y<2>:O Y<1>:O Y<0>:O Yb<8>:O Yb<7>:O
*.PININFO Yb<6>:O Yb<5>:O Yb<4>:O Yb<3>:O Yb<2>:O Yb<1>:O Yb<0>:O vdd:B
XI68 d<3> a<0> P1_3<3> vdd / nand2
XI67 d<3> a<1> P1_4<2> vdd / nand2
XI69 d<2> a<3> P1_5<0> vdd / nand2
XI73 d<1> a<3> P1_4<0> vdd / nand2
XI66 d<3> a<2> P1_5<1> vdd / nand2
XI77 d<0> a<3> P1_3<0> vdd / nand2
XI21 b<3> c<3> P0_6 vdd / nand2
XI11 b<2> c<2> P0_4<1> vdd / nand2
XI10 b<2> c<1> P0_3<2> vdd / nand2
XI9 b<2> c<0> P0_2<2> vdd / nand2
XI7 b<1> c<2> P0_3<1> vdd / nand2
XI6 b<1> c<1> P0_2<1> vdd / nand2
XI5 b<1> c<0> P0_1<1> vdd / nand2
XI2 b<0> c<2> P0_2<0> vdd / nand2
XI1 b<0> c<1> P0_1<0> vdd / nand2
XI0 b<0> c<0> P0_0 vdd / nand2
XI64 CK_in Rb<0> Rb<1> Rb<2> R<3> R<4> R<5> R<6> R<7> Rb<8> Yb<0> Yb<1> Yb<2>
+ Y<3> Y<4> Y<5> Y<6> Y<7> Yb<8> Y<0> Y<1> Y<2> Yb<3> Yb<4> Yb<5> Yb<6> Yb<7>
+ Y<8> Resetb vdd / DETFF_Im_9_CMAC
XI88 C2_1_2b S2_2b C3_2_3b Rb<2> vdd / HA_pt
XI86 C0_0_1b S1_1b C2_1_2b Rb<1> vdd / HA_pt
XI90 C2_3_4b S2_4b C3_4_5b S3_4b vdd / HA_pt
XI61 S1_6b C1_5_6b C2_5_6b C3_6_7b S3_6b vdd / FA_pt
XI60 S2_5b C1_4_5b<1> C2_4_5b C3_5_6b S3_5b vdd / FA_pt
XI89 S2_3b C1_2_3b C2_2_3b C3_3_4b S3_3b vdd / FA_pt
XI57 S1_5b C0_4_5b<1> C1_4_5b<0> C2_5_6b S2_5b vdd / FA_pt
XI56 S1_4b<0> S1_4b<1> C1_3_4b C2_4_5b S2_4b vdd / FA_pt
XI55 S0_3b<2> S1_3b C0_2_3b<1> C2_3_4b S2_3b vdd / FA_pt
XI87 S1_2b S0_2b<1> C1_1_2b C2_2_3b S2_2b vdd / FA_pt
XI53 P0_6 P1_6 C0_5_6b C1_6_7b S1_6b vdd / FA_pt
XI85 S0_5b P1_5<1> C0_4_5b<0> C1_5_6b S1_5b vdd / FA_pt
XI50 S0_4b<1> C0_3_4b<1> C0_3_4b<2> C1_4_5b<1> S1_4b<1> vdd / FA_pt
XI49 S0_3b<0> S0_3b<1> C0_2_3b<0> C1_3_4b S1_3b vdd / FA_pt
XI46 P0_5<0> P0_5<1> P1_5<0> C0_5_6b S0_5b vdd / FA_pt
XI45 P1_4<0> P1_4<1> P1_4<2> C0_4_5b<1> S0_4b<1> vdd / FA_pt
XI44 P0_4<0> P0_4<1> P0_4<2> C0_4_5b<0> S0_4b<0> vdd / FA_pt
XI43 P0_3<3> P1_3<0> P1_3<1> C0_3_4b<1> S0_3b<1> vdd / FA_pt
XI42 P0_3<0> P0_3<1> P0_3<2> C0_3_4b<0> S0_3b<0> vdd / FA_pt
XI41 P1_2<0> P1_2<1> P1_2<2> C0_2_3b<1> S0_2b<1> vdd / FA_pt
XI40 P0_2<0> P0_2<1> P0_2<2> C0_2_3b<0> S0_2b<0> vdd / FA_pt
XI39 P0_1<0> P0_1<1> P1_1<0> C0_1_2b S0_1b vdd / FA_pt
XI81 P1_3<2> P1_3<3> C0_3_4b<2> S0_3b<2> vdd / HA_add_1_pt
XI84 C0_3_4b<0> S0_4b<0> C1_4_5b<0> S1_4b<0> vdd / HA_add_1_pt
XI83 C0_1_2b S0_2b<0> C1_2_3b S1_2b vdd / HA_add_1_pt
XI82 P1_1<1> S0_1b C1_1_2b S1_1b vdd / HA_add_1_pt
XI38 P0_0 P1_0 C0_0_1b Rb<0> vdd / HA_add_1_pt
XI65 db<3> ab<3> P1_6 vdd / nor2_alt
XI72 db<2> ab<0> P1_2<2> vdd / nor2_alt
XI71 db<2> ab<1> P1_3<2> vdd / nor2_alt
XI70 db<2> ab<2> P1_4<1> vdd / nor2_alt
XI76 db<1> ab<0> P1_1<1> vdd / nor2_alt
XI75 db<1> ab<1> P1_2<1> vdd / nor2_alt
XI74 db<1> ab<2> P1_3<1> vdd / nor2_alt
```

```
XI80 db<0> ab<0> P1_0 vdd / nor2_alt
XI79 db<0> ab<1> P1_1<0> vdd / nor2_alt
XI78 db<0> ab<2> P1_2<0> vdd / nor2_alt
XI17 bb<3> cb<0> P0_3<3> vdd / nor2_alt
XI19 bb<3> cb<2> P0_5<1> vdd / nor2_alt
XI18 bb<3> cb<1> P0_4<2> vdd / nor2_alt
XI12 bb<2> cb<3> P0_5<0> vdd / nor2_alt
XI8 bb<1> cb<3> P0_4<0> vdd / nor2_alt
XI4 bb<0> cb<3> P0_3<0> vdd / nor2_alt
MP3 net014 net014 vdd vdd pfet m=1 w=269n l=30n nf=1.0 pccrit=1 as=26.9f
+ ad=26.9f ps=738n pd=738n sa=115n sb=115n sd=100n ptwell=0 ngcon=1 p_la=0
MN1 LL net014 gnd! gnd! nfet m=1 w=180n l=30n nf=1.0 pccrit=1 as=18f ad=18f
+ ps=560n pd=560n sa=115n sb=115n sd=100n ptwell=0 ngcon=1 p_la=0
XI62 C3_2_3b C3_3_4b C3_4_5b C3_5_6b C3_6_7b S3_3b S3_4b S3_5b S3_6b C1_6_7b
+ LL Rb<8> R<3> R<4> R<5> R<6> R<7> vdd / CLA5
.ENDS
```

A.2 8-bit CMAC

```
********************************************************************************
* Library Name: CMACs
* Cell Name:    DETFF_Re_17_CMAC
* View Name:    schematic
********************************************************************************

.SUBCKT DETFF_Re_17_CMAC CK_in D<0> D<1> D<2> D<3> D<4> D<5> D<6> D<7> D<8>
+ D<9> D<10> D<11> D<12> D<13> D<14> D<15> D<16> Q<0> Q<1> Q<2> Q<3> Q<4> Q<5>
+ Q<6> Q<7> Q<8> Q<9> Q<10> Q<11> Q<12> Q<13> Q<14> Q<15> Q<16> Qb<0> Qb<1>
+ Qb<2> Qb<3> Qb<4> Qb<5> Qb<6> Qb<7> Qb<8> Qb<9> Qb<10> Qb<11> Qb<12> Qb<13>
+ Qb<14> Qb<15> Qb<16> Rb_in vdd
*.PININFO CK_in:I D<0>:I D<1>:I D<2>:I D<3>:I D<4>:I D<5>:I D<6>:I D<7>:I
*.PININFO D<8>:I D<9>:I D<10>:I D<11>:I D<12>:I D<13>:I D<14>:I D<15>:I
*.PININFO D<16>:I Rb_in:I Q<0>:O Q<1>:O Q<2>:O Q<3>:O Q<4>:O Q<5>:O Q<6>:O
*.PININFO Q<7>:O Q<8>:O Q<9>:O Q<10>:O Q<11>:O Q<12>:O Q<13>:O Q<14>:O Q<15>:O
*.PININFO Q<16>:O Qb<0>:O Qb<1>:O Qb<2>:O Qb<3>:O Qb<4>:O Qb<5>:O Qb<6>:O
*.PININFO Qb<7>:O Qb<8>:O Qb<9>:O Qb<10>:O Qb<11>:O Qb<12>:O Qb<13>:O Qb<14>:O
*.PININFO Qb<15>:O Qb<16>:O vdd:B
XI39 CK CKb D<8> Q<8> Qb<8> Rb vdd / DETFF_R
XI38 CK CKb D<7> Q<7> Qb<7> Rb vdd / DETFF_R
XI36 CK CKb D<9> Q<9> Qb<9> Rb vdd / DETFF_R
XI40 CK CKb D<10> Q<10> Qb<10> Rb vdd / DETFF_R
XI37 CK CKb D<6> Q<6> Qb<6> Rb vdd / DETFF_R
XI42 Rb_in Rb R vdd / buffer
XI22 CK CKb P Pb vdd / pulse_generator
XI30 D<15> P Pb Q<15> Qb<15> Rb vdd / pulsed_latch
XI23 D<11> P Pb Q<11> Qb<11> Rb vdd / pulsed_latch
XI24 D<12> P Pb Q<12> Qb<12> Rb vdd / pulsed_latch
XI25 D<14> P Pb Q<14> Qb<14> Rb vdd / pulsed_latch
XI26 D<13> P Pb Q<13> Qb<13> Rb vdd / pulsed_latch
XI20 CK_in CK CKb vdd / buffer_strong
XI33 CK CKb D<5> Q<5> Qb<5> R vdd / DETFF_S
XI32 CK CKb D<4> Q<4> Qb<4> R vdd / DETFF_S
XI31 CK CKb D<3> Q<3> Qb<3> R vdd / DETFF_S
XI2 CK CKb D<2> Q<2> Qb<2> R vdd / DETFF_S
XI1 CK CKb D<1> Q<1> Qb<1> R vdd / DETFF_S
XI0 CK CKb D<0> Q<0> Qb<0> R vdd / DETFF_S
XI27 D<16> P Pb Q<16> Qb<16> R vdd / pulsed_latch_S
.ENDS

********************************************************************************
* Library Name: CMACs
```

```
* Cell Name:    DETFF_Im_17_CMAC
* View Name:    schematic
********************************************************************

.SUBCKT DETFF_Im_17_CMAC CK_in D<0> D<1> D<2> D<3> D<4> D<5> D<6> D<7> D<8>
+ D<9> D<10> D<11> D<12> D<13> D<14> D<15> D<16> Q<0> Q<1> Q<2> Q<3> Q<4> Q<5>
+ Q<6> Q<7> Q<8> Q<9> Q<10> Q<11> Q<12> Q<13> Q<14> Q<15> Q<16> Qb<0> Qb<1>
+ Qb<2> Qb<3> Qb<4> Qb<5> Qb<6> Qb<7> Qb<8> Qb<9> Qb<10> Qb<11> Qb<12> Qb<13>
+ Qb<14> Qb<15> Qb<16> Rb_in vdd
*.PININFO CK_in:I D<0>:I D<1>:I D<2>:I D<3>:I D<4>:I D<5>:I D<6>:I D<7>:I
*.PININFO D<8>:I D<9>:I D<10>:I D<11>:I D<12>:I D<13>:I D<14>:I D<15>:I
*.PININFO D<16>:I Rb_in:I Q<0>:O Q<1>:O Q<2>:O Q<3>:O Q<4>:O Q<5>:O Q<6>:O
*.PININFO Q<7>:O Q<8>:O Q<9>:O Q<10>:O Q<11>:O Q<12>:O Q<13>:O Q<14>:O Q<15>:O
*.PININFO Q<16>:O Qb<0>:O Qb<1>:O Qb<2>:O Qb<3>:O Qb<4>:O Qb<5>:O Qb<6>:O
*.PININFO Qb<7>:O Qb<8>:O Qb<9>:O Qb<10>:O Qb<11>:O Qb<12>:O Qb<13>:O Qb<14>:O
*.PININFO Qb<15>:O Qb<16>:O vdd:B
XI39 CK CKb D<8> Q<8> Qb<8> Rb vdd / DETFF_R
XI38 CK CKb D<7> Q<7> Qb<7> Rb vdd / DETFF_R
XI36 CK CKb D<9> Q<9> Qb<9> Rb vdd / DETFF_R
XI40 CK CKb D<10> Q<10> Qb<10> Rb vdd / DETFF_R
XI33 CK CKb D<5> Q<5> Qb<5> Rb vdd / DETFF_R
XI32 CK CKb D<4> Q<4> Qb<4> Rb vdd / DETFF_R
XI37 CK CKb D<6> Q<6> Qb<6> Rb vdd / DETFF_R
XI42 Rb_in Rb R vdd / buffer
XI22 CK CKb P Pb vdd / pulse_generator
XI30 D<15> P Pb Q<15> Qb<15> Rb vdd / pulsed_latch
XI23 D<11> P Pb Q<11> Qb<11> Rb vdd / pulsed_latch
XI24 D<12> P Pb Q<12> Qb<12> Rb vdd / pulsed_latch
XI25 D<14> P Pb Q<14> Qb<14> Rb vdd / pulsed_latch
XI26 D<13> P Pb Q<13> Qb<13> Rb vdd / pulsed_latch
XI20 CK_in CK CKb vdd / buffer_strong
XI31 CK CKb D<3> Q<3> Qb<3> R vdd / DETFF_S
XI2 CK CKb D<2> Q<2> Qb<2> R vdd / DETFF_S
XI1 CK CKb D<1> Q<1> Qb<1> R vdd / DETFF_S
XI0 CK CKb D<0> Q<0> Qb<0> R vdd / DETFF_S
XI27 D<16> P Pb Q<16> Qb<16> R vdd / pulsed_latch_S
.ENDS

********************************************************************************
* Library Name: CMACs
* Cell Name:    CMUL_8bit_Re
* View Name:    schematic
********************************************************************************

.SUBCKT CMUL_8bit_Re CK_in Resetb Y<0> Y<1> Y<2> Y<3> Y<4> Y<5> Y<6> Y<7> Y<8>
+ Y<9> Y<10> Y<11> Y<12> Y<13> Y<14> Y<15> Y<16> Yb<0> Yb<1> Yb<2> Yb<3> Yb<4>
+ Yb<5> Yb<6> Yb<7> Yb<8> Yb<9> Yb<10> Yb<11> Yb<12> Yb<13> Yb<14> Yb<15>
+ Yb<16> a<0> a<1> a<2> a<3> a<4> a<5> a<6> a<7> ab<0> ab<1> ab<2> ab<3> ab<4>
+ ab<5> ab<6> ab<7> b<0> b<1> b<2> b<3> b<4> b<5> b<6> b<7> bb<0> bb<1> bb<2>
+ bb<3> bb<4> bb<5> bb<6> bb<7> c<0> c<1> c<2> c<3> c<4> c<5> c<6> c<7> cb<0>
+ cb<1> cb<2> cb<3> cb<4> cb<5> cb<6> cb<7> d<0> d<1> d<2> d<3> d<4> d<5> d<6>
+ d<7> db<0> db<1> db<2> db<3> db<4> db<5> db<6> db<7> vdd
*.PININFO CK_in:I Resetb:I a<0>:I a<1>:I a<2>:I a<3>:I a<4>:I a<5>:I a<6>:I
*.PININFO a<7>:I ab<0>:I ab<1>:I ab<2>:I ab<3>:I ab<4>:I ab<5>:I ab<6>:I
*.PININFO ab<7>:I b<0>:I b<1>:I b<2>:I b<3>:I b<4>:I b<5>:I b<6>:I b<7>:I
*.PININFO bb<0>:I bb<1>:I bb<2>:I bb<3>:I bb<4>:I bb<5>:I bb<6>:I bb<7>:I
*.PININFO c<0>:I c<1>:I c<2>:I c<3>:I c<4>:I c<5>:I c<6>:I c<7>:I cb<0>:I
*.PININFO cb<1>:I cb<2>:I cb<3>:I cb<4>:I cb<5>:I cb<6>:I cb<7>:I d<0>:I
*.PININFO d<1>:I d<2>:I d<3>:I d<4>:I d<5>:I d<6>:I d<7>:I db<0>:I db<1>:I
*.PININFO db<2>:I db<3>:I db<4>:I db<5>:I db<6>:I db<7>:I Y<0>:O Y<1>:O Y<2>:O
*.PININFO Y<3>:O Y<4>:O Y<5>:O Y<6>:O Y<7>:O Y<8>:O Y<9>:O Y<10>:O Y<11>:O
*.PININFO Y<12>:O Y<13>:O Y<14>:O Y<15>:O Y<16>:O Yb<0>:O Yb<1>:O Yb<2>:O
```

```
*.PININFO Yb<3>:O Yb<4>:O Yb<5>:O Yb<6>:O Yb<7>:O Yb<8>:O Yb<9>:O Yb<10>:O
*.PININFO Yb<11>:O Yb<12>:O Yb<13>:O Yb<14>:O Yb<15>:O Yb<16>:O vdd:B
XI177 d<0> b<4> P1_4<0> vdd / nand2
XI176 d<0> b<5> P1_5<0> vdd / nand2
XI175 d<0> b<6> P1_6<0> vdd / nand2
XI174 d<0> b<1> P1_1<0> vdd / nand2
XI173 d<0> b<2> P1_2<0> vdd / nand2
XI172 d<0> b<3> P1_3<0> vdd / nand2
XI171 d<0> b<0> P1_0 vdd / nand2
XI169 d<1> b<3> P1_4<1> vdd / nand2
XI168 d<1> b<1> P1_2<1> vdd / nand2
XI167 d<1> b<2> P1_3<1> vdd / nand2
XI166 d<1> b<0> P1_1<1> vdd / nand2
XI165 d<1> b<4> P1_5<1> vdd / nand2
XI164 d<1> b<5> P1_6<1> vdd / nand2
XI163 d<1> b<6> P1_7<1> vdd / nand2
XI161 d<2> b<3> P1_5<2> vdd / nand2
XI160 d<2> b<1> P1_3<2> vdd / nand2
XI159 d<2> b<0> P1_2<2> vdd / nand2
XI158 d<2> b<2> P1_4<2> vdd / nand2
XI157 d<2> b<4> P1_6<2> vdd / nand2
XI156 d<2> b<5> P1_7<2> vdd / nand2
XI155 d<2> b<6> P1_8<1> vdd / nand2
XI153 d<3> b<3> P1_6<3> vdd / nand2
XI152 d<3> b<1> P1_4<3> vdd / nand2
XI151 d<3> b<0> P1_3<3> vdd / nand2
XI150 d<3> b<2> P1_5<3> vdd / nand2
XI149 d<3> b<4> P1_7<3> vdd / nand2
XI148 d<3> b<5> P1_8<2> vdd / nand2
XI147 d<3> b<6> P1_9<1> vdd / nand2
XI145 d<4> b<5> P1_9<2> vdd / nand2
XI144 d<4> b<4> P1_8<3> vdd / nand2
XI143 d<4> b<2> P1_6<4> vdd / nand2
XI142 d<4> b<0> P1_4<4> vdd / nand2
XI141 d<4> b<1> P1_5<4> vdd / nand2
XI140 d<4> b<3> P1_7<4> vdd / nand2
XI139 d<4> b<6> P1_10<1> vdd / nand2
XI137 d<5> b<3> P1_8<4> vdd / nand2
XI136 d<5> b<1> P1_6<5> vdd / nand2
XI135 d<5> b<0> P1_5<5> vdd / nand2
XI134 d<5> b<2> P1_7<5> vdd / nand2
XI133 d<5> b<4> P1_9<3> vdd / nand2
XI132 d<5> b<5> P1_10<2> vdd / nand2
XI131 d<5> b<6> P1_11<1> vdd / nand2
XI129 d<6> b<6> P1_12<1> vdd / nand2
XI128 d<6> b<5> P1_11<2> vdd / nand2
XI127 d<6> b<4> P1_10<3> vdd / nand2
XI126 d<6> b<2> P1_8<5> vdd / nand2
XI125 d<6> b<0> P1_6<6> vdd / nand2
XI124 d<6> b<1> P1_7<6> vdd / nand2
XI123 d<6> b<3> P1_9<4> vdd / nand2
XI115 d<7> b<7> P1_14 vdd / nand2
XI109 a<6> c<3> P0_9<4> vdd / nand2
XI108 a<6> c<1> P0_7<6> vdd / nand2
XI107 a<6> c<0> P0_6<6> vdd / nand2
XI106 a<6> c<2> P0_8<5> vdd / nand2
XI105 a<6> c<4> P0_10<3> vdd / nand2
XI104 a<6> c<5> P0_11<2> vdd / nand2
XI103 a<6> c<6> P0_12<1> vdd / nand2
XI101 a<5> c<6> P0_11<1> vdd / nand2
XI100 a<5> c<5> P0_10<2> vdd / nand2
XI99 a<5> c<4> P0_9<3> vdd / nand2
```

```
XI98 a<5> c<2> P0_7<5> vdd / nand2
XI97 a<5> c<0> P0_5<5> vdd / nand2
XI96 a<5> c<1> P0_6<5> vdd / nand2
XI95 a<5> c<3> P0_8<4> vdd / nand2
XI92 a<4> c<3> P0_7<4> vdd / nand2
XI91 a<4> c<1> P0_5<4> vdd / nand2
XI90 a<4> c<0> P0_4<4> vdd / nand2
XI89 a<4> c<2> P0_6<4> vdd / nand2
XI88 a<4> c<4> P0_8<3> vdd / nand2
XI87 a<4> c<5> P0_9<2> vdd / nand2
XI86 a<4> c<6> P0_10<1> vdd / nand2
XI84 a<3> c<6> P0_9<1> vdd / nand2
XI83 a<3> c<5> P0_8<2> vdd / nand2
XI82 a<3> c<4> P0_7<3> vdd / nand2
XI81 a<3> c<2> P0_5<3> vdd / nand2
XI80 a<3> c<0> P0_3<3> vdd / nand2
XI79 a<3> c<1> P0_4<3> vdd / nand2
XI78 a<3> c<3> P0_6<3> vdd / nand2
XI77 a<2> c<3> P0_5<2> vdd / nand2
XI76 a<2> c<1> P0_3<2> vdd / nand2
XI75 a<2> c<0> P0_2<2> vdd / nand2
XI73 a<1> c<3> P0_4<1> vdd / nand2
XI72 a<1> c<1> P0_2<1> vdd / nand2
XI71 a<1> c<2> P0_3<1> vdd / nand2
XI74 a<2> c<2> P0_4<2> vdd / nand2
XI69 a<1> c<0> P0_1<1> vdd / nand2
XI68 a<0> c<0> P0_0 vdd / nand2
XI67 a<0> c<3> P0_3<0> vdd / nand2
XI66 a<0> c<2> P0_2<0> vdd / nand2
XI65 a<0> c<1> P0_1<0> vdd / nand2
XI21 a<7> c<7> P0_14 vdd / nand2
XI11 a<2> c<6> P0_8<1> vdd / nand2
XI10 a<2> c<5> P0_7<2> vdd / nand2
XI9 a<2> c<4> P0_6<2> vdd / nand2
XI7 a<1> c<6> P0_7<1> vdd / nand2
XI6 a<1> c<5> P0_6<1> vdd / nand2
XI5 a<1> c<4> P0_5<1> vdd / nand2
XI2 a<0> c<6> P0_6<0> vdd / nand2
XI1 a<0> c<5> P0_5<0> vdd / nand2
XI0 a<0> c<4> P0_4<0> vdd / nand2
XI179 P0_4<0> P0_4<3> P1_4<1> P0_4<1> P0_4<4> P1_4<2> P0_4<2> P1_4<0> P1_4<3>
+ C0_4_5b<0> C0_4_5b<1> C0_4_5b<2> S0_4b<0> S0_4b<1> S0_4b<2> vdd / FA_pt_3
XI185 P0_10<0> P0_10<3> P1_10<1> P0_10<1> P0_10<4> P1_10<2> P0_10<2> P1_10<0>
+ P1_10<3> C0_10_11b<0> C0_10_11b<1> C0_10_11b<2> S0_10b<0> S0_10b<1>
+ S0_10b<2> vdd / FA_pt_3
XI197 S0_7b<0> S0_7b<2> S0_7b<4> S0_7b<1> S0_7b<3> C0_6_7b<1> P1_7<7>
+ C0_6_7b<0> C0_6_7b<2> C1_7_8b<0> C1_7_8b<1> C1_7_8b<2> S1_7b<0> S1_7b<1>
+ S1_7b<2> vdd / FA_pt_3
XI196 S0_6b<0> S0_6b<2> C0_5_6b<0> S0_6b<1> S0_6b<3> C0_5_6b<1> P1_6<5>
+ P1_6<6> C0_5_6b<2> C1_6_7b<0> C1_6_7b<1> C1_6_7b<2> S1_6b<0> S1_6b<1>
+ S1_6b<2> vdd / FA_pt_3
XI198 S0_8b<0> S0_8b<2> C0_7_8b<0> S0_8b<1> S0_8b<3> C0_7_8b<1> P1_8<5>
+ P1_8<6> C0_7_8b<2> C1_8_9b<0> C1_8_9b<1> C1_8_9b<2> S1_8b<0> S1_8b<1>
+ S1_8b<2> vdd / FA_pt_3
XI253 C4_7_8b S4_8b C5_8_9b S5_8b vdd / HA_pt
XI252 C3_6_7b S4_7b C5_7_8b S5_7b vdd / HA_pt
XI250 C4_4_5b S4_5b C5_5_6b Rb<5> vdd / HA_pt
XI242 C3_3_4b S3_4b C4_4_5b Rb<4> vdd / HA_pt
XI229 C2_2_3b S2_3b C3_3_4b Rb<3> vdd / HA_pt
XI210 C1_1_2b S1_2b C2_2_3b Rb<2> vdd / HA_pt
XI38 P0_0 P1_0 C0_0_1b Rb<0> vdd / HA_pt
XI257 S3_12b C3_11_12b C4_11_12b C5_12_13b S5_12b vdd / FA_pt
```

```
XI256 S4_11b C3_10_11b C4_10_11b C5_11_12b S5_11b vdd / FA_pt
XI255 S4_10b C3_9_10b C4_9_10b C5_10_11b S5_10b vdd / FA_pt
XI254 S4_9b C3_8_9b<1> C4_8_9b C5_9_10b S5_9b vdd / FA_pt
XI251 S3_6b C3_5_6b C4_5_6b C5_6_7b S5_6b vdd / FA_pt
XI249 S2_13b C2_12_13b C3_12_13b C4_13_14b S4_13b vdd / FA_pt
XI248 S3_11b C2_10_11b<0> C2_10_11b<1> C4_11_12b S4_11b vdd / FA_pt
XI247 S3_10b C2_9_10b<0> C2_9_10b<1> C4_10_11b S4_10b vdd / FA_pt
XI246 S3_9b C2_8_9b<1> C3_8_9b<0> C4_9_10b S4_9b vdd / FA_pt
XI245 S3_8b<0> S3_8b<1> C3_7_8b C4_8_9b S4_8b vdd / FA_pt
XI244 S3_7b C2_6_7b<0> C2_6_7b<1> C4_7_8b S4_7b vdd / FA_pt
XI243 S3_5b C2_4_5b C3_4_5b C4_5_6b S4_5b vdd / FA_pt
XI241 S1_14b C1_13_14b C2_13_14b C3_14_15b S3_14b vdd / FA_pt
XI240 S2_12b C1_11_12b<1> C2_11_12b C3_12_13b S3_12b vdd / FA_pt
XI239 S2_11b C1_10_11b<0> C1_10_11b<1> C3_11_12b S3_11b vdd / FA_pt
XI237 S2_10b<0> S2_10b<1> C1_9_10b<2> C3_10_11b S3_10b vdd / FA_pt
XI236 S2_9b<0> S2_9b<1> C2_8_9b<0> C3_9_10b S3_9b vdd / FA_pt
XI235 C2_7_8b<0> C2_7_8b<1> C1_7_8b<2> C3_8_9b<1> S3_8b<1> vdd / FA_pt
XI234 S2_8b<0> S2_8b<1> C1_7_8b<1> C3_8_9b<0> S3_8b<0> vdd / FA_pt
XI233 S2_7b<0> S2_7b<1> C1_6_7b<2> C3_7_8b S3_7b vdd / FA_pt
XI232 S2_6b<0> S2_6b<1> C2_5_6b C3_6_7b S3_6b vdd / FA_pt
XI231 S2_5b C1_4_5b<0> C1_4_5b<1> C3_5_6b S3_5b vdd / FA_pt
XI230 S2_4b C2_3_4b C1_3_4b<1> C3_4_5b S3_4b vdd / FA_pt
XI227 S1_13b C1_12_13b C0_12_13b<1> C2_13_14b S2_13b vdd / FA_pt
XI226 S1_12b C1_11_12b<0> C0_11_12b<1> C2_12_13b S2_12b vdd / FA_pt
XI224 S1_11b<0> S1_11b<1> C0_10_11b<2> C2_11_12b S2_11b vdd / FA_pt
XI223 C1_9_10b<0> C1_9_10b<1> C0_9_10b<3> C2_10_11b<1> S2_10b<1> vdd / FA_pt
XI222 S1_10b<0> S1_10b<1> C0_9_10b<2> C2_10_11b<0> S2_10b<0> vdd / FA_pt
XI221 S1_9b<0> S1_9b<1> C1_8_9b<0> C2_9_10b<0> S2_9b<0> vdd / FA_pt
XI220 S1_9b<2> C1_8_9b<1> C1_8_9b<2> C2_9_10b<1> S2_9b<1> vdd / FA_pt
XI219 S1_8b<2> C1_7_8b<0> C0_7_8b<4> C2_8_9b<1> S2_8b<1> vdd / FA_pt
XI218 S1_8b<0> S1_8b<1> C0_7_8b<3> C2_8_9b<0> S2_8b<0> vdd / FA_pt
XI217 S1_7b<0> S1_7b<1> C0_6_7b<3> C2_7_8b<0> S2_7b<0> vdd / FA_pt
XI216 S1_7b<2> C1_6_7b<0> C1_6_7b<1> C2_7_8b<1> S2_7b<1> vdd / FA_pt
XI215 S1_6b<2> C1_5_6b<0> C1_5_6b<1> C2_6_7b<1> S2_6b<1> vdd / FA_pt
XI214 S1_6b<0> S1_6b<1> C0_5_6b<3> C2_6_7b<0> S2_6b<0> vdd / FA_pt
XI213 S1_5b<0> S1_5b<1> C0_4_5b<2> C2_5_6b S2_5b vdd / FA_pt
XI212 S1_4b<0> S1_4b<1> C1_3_4b<0> C2_4_5b S2_4b vdd / FA_pt
XI211 S1_3b<0> S1_3b<1> C1_2_3b C2_3_4b S2_3b vdd / FA_pt
XI208 C0_13_14b P0_14 P1_14 C1_14_15b S1_14b vdd / FA_pt
XI207 S0_13b C0_12_13b<0> P1_13<1> C1_13_14b S1_13b vdd / FA_pt
XI206 S0_12b<0> S0_12b<1> C0_11_12b<0> C1_12_13b S1_12b vdd / FA_pt
XI205 C0_10_11b<0> C0_10_11b<1> P1_11<3> C1_11_12b<1> S1_11b<1> vdd / FA_pt
XI204 S0_11b<0> S0_11b<1> P1_11<2> C1_11_12b<0> S1_11b<0> vdd / FA_pt
XI203 S0_10b<2> C0_9_10b<0> C0_9_10b<1> C1_10_11b<1> S1_10b<1> vdd / FA_pt
XI202 S0_10b<0> S0_10b<1> P1_10<4> C1_10_11b<0> S1_10b<0> vdd / FA_pt
XI194 S0_5b<2> S0_5b<3> C0_4_5b<1> C1_5_6b<1> S1_5b<1> vdd / FA_pt
XI195 S0_5b<0> S0_5b<1> C0_4_5b<0> C1_5_6b<0> S1_5b<0> vdd / FA_pt
XI51 S0_4b<0> S0_4b<1> P1_4<4> C1_4_5b<0> S1_4b<0> vdd / FA_pt
XI50 P1_3<3> C0_2_3b<0> C0_2_3b<1> C1_3_4b<1> S1_3b<1> vdd / FA_pt
XI49 S0_3b<0> S0_3b<1> P1_3<2> C1_3_4b<0> S1_3b<0> vdd / FA_pt
XI48 S0_2b<0> S0_2b<1> C0_1_2b C1_2_3b S1_2b vdd / FA_pt
XI47 S0_1b C0_0_1b P1_1<1> C1_1_2b Rb<1> vdd / FA_pt
XI192 S0_4b<2> C0_3_4b<0> C0_3_4b<1> C1_4_5b<1> S1_4b<1> vdd / FA_pt
XI43 P0_3<3> P1_3<0> P1_3<1> C0_3_4b<1> S0_3b<1> vdd / FA_pt
XI42 P0_3<0> P0_3<1> P0_3<2> C0_3_4b<0> S0_3b<0> vdd / FA_pt
XI41 P1_2<0> P1_2<1> P1_2<2> C0_2_3b<1> S0_2b<1> vdd / FA_pt
XI40 P0_2<0> P0_2<1> P0_2<2> C0_2_3b<0> S0_2b<0> vdd / FA_pt
XI39 P0_1<0> P0_1<1> P1_1<0> C0_1_2b S0_1b vdd / FA_pt
XI187 P0_11<3> P1_11<0> P1_11<1> C0_11_12b<1> S0_11b<1> vdd / FA_pt
XI190 P1_12<0> P1_12<1> P1_12<2> C0_12_13b<1> S0_12b<1> vdd / FA_pt
XI188 P0_12<0> P0_12<1> P0_12<2> C0_12_13b<0> S0_12b<0> vdd / FA_pt
XI200 S0_9b<1> S0_9b<2> C0_8_9b<1> C1_9_10b<1> S1_9b<1> vdd / FA_pt
```

```
XI186 P0_11<0> P0_11<1> P0_11<2> C0_11_12b<0> S0_11b<0> vdd / FA_pt
XI191 P0_13<0> P0_13<1> P1_13<0> C0_13_14b S0_13b vdd / FA_pt
XI201 S0_9b<3> C0_8_9b<2> C0_8_9b<3> C1_9_10b<2> S1_9b<2> vdd / FA_pt
XI193 C0_8_9b<0> S0_9b<0> C1_9_10b<0> S1_9b<0> vdd / HA_add_1_pt
XI258 C5_5_6b C5_6_7b C5_7_8b C5_8_9b C5_9_10b S5_6b S5_7b S5_8b S5_9b S5_10b
+ LL CR_10_11 R<6> R<7> R<8> R<9> R<10> vdd / CLA5
XI259 C5_10_11b C5_11_12b C5_12_13b C4_13_14b C3_14_15b S5_11b S5_12b S4_13b
+ S3_14b C1_14_15b CR_10_11 Rb<16> R<11> R<12> R<13> R<14> R<15> vdd / CLA5
MP3 net014 net014 vdd vdd pfet m=1 w=269n l=30n nf=1.0 pccrit=1 as=26.9f
+ ad=26.9f ps=738n pd=738n sa=115n sb=115n sd=100n ptwell=0 ngcon=1 p_la=0
MN1 LL net014 gnd! gnd! nfet m=1 w=180n l=30n nf=1.0 pccrit=1 as=18f ad=18f
+ ps=560n pd=560n sa=115n sb=115n sd=100n ptwell=0 ngcon=1 p_la=0
XI178 db<0> bb<7> P1_7<0> vdd / nor2_alt
XI170 db<1> bb<7> P1_8<0> vdd / nor2_alt
XI162 db<2> bb<7> P1_9<0> vdd / nor2_alt
XI154 db<3> bb<7> P1_10<0> vdd / nor2_alt
XI146 db<4> bb<7> P1_11<0> vdd / nor2_alt
XI138 db<5> bb<7> P1_12<0> vdd / nor2_alt
XI130 db<6> bb<7> P1_13<0> vdd / nor2_alt
XI122 db<7> bb<5> P1_12<2> vdd / nor2_alt
XI121 db<7> bb<4> P1_11<3> vdd / nor2_alt
XI120 db<7> bb<3> P1_10<4> vdd / nor2_alt
XI119 db<7> bb<1> P1_8<6> vdd / nor2_alt
XI118 db<7> bb<2> P1_9<5> vdd / nor2_alt
XI117 db<7> bb<0> P1_7<7> vdd / nor2_alt
XI116 db<7> bb<6> P1_13<1> vdd / nor2_alt
XI113 ab<7> cb<0> P0_7<7> vdd / nor2_alt
XI112 ab<7> cb<2> P0_9<5> vdd / nor2_alt
XI111 ab<7> cb<1> P0_8<6> vdd / nor2_alt
XI110 ab<7> cb<3> P0_10<4> vdd / nor2_alt
XI102 ab<6> cb<7> P0_13<0> vdd / nor2_alt
XI94 ab<5> cb<7> P0_12<0> vdd / nor2_alt
XI93 ab<4> cb<7> P0_11<0> vdd / nor2_alt
XI85 ab<3> cb<7> P0_10<0> vdd / nor2_alt
XI17 ab<7> cb<4> P0_11<3> vdd / nor2_alt
XI19 ab<7> cb<6> P0_13<1> vdd / nor2_alt
XI18 ab<7> cb<5> P0_12<2> vdd / nor2_alt
XI12 ab<2> cb<7> P0_9<0> vdd / nor2_alt
XI8 ab<1> cb<7> P0_8<0> vdd / nor2_alt
XI4 ab<0> cb<7> P0_7<0> vdd / nor2_alt
XI180 P0_5<0> P0_5<3> P1_5<0> P1_5<3> P0_5<1> P0_5<4> P1_5<1> P1_5<4> P0_5<2>
+ P0_5<5> P1_5<2> P1_5<5> C0_5_6b<0> C0_5_6b<1> C0_5_6b<2> C0_5_6b<3> S0_5b<0>
+ S0_5b<1> S0_5b<2> S0_5b<3> vdd / FA_pt_4
XI184 P0_9<0> P0_9<3> P1_9<0> P1_9<3> P0_9<1> P0_9<4> P1_9<1> P1_9<4> P0_9<2>
+ P0_9<5> P1_9<2> P1_9<5> C0_9_10b<0> C0_9_10b<1> C0_9_10b<2> C0_9_10b<3>
+ S0_9b<0> S0_9b<1> S0_9b<2> S0_9b<3> vdd / FA_pt_4
XI181 P0_6<0> P0_6<3> P0_6<6> P1_6<2> P0_6<1> P0_6<4> P1_6<0> P1_6<3> P0_6<2>
+ P0_6<5> P1_6<1> P1_6<4> C0_6_7b<0> C0_6_7b<1> C0_6_7b<2> C0_6_7b<3> S0_6b<0>
+ S0_6b<1> S0_6b<2> S0_6b<3> vdd / FA_pt_4
XI183 P0_8<0> P0_8<3> P0_8<6> P1_8<2> P0_8<1> P0_8<4> P1_8<0> P1_8<3> P0_8<2>
+ P0_8<5> P1_8<1> P1_8<4> C0_8_9b<0> C0_8_9b<1> C0_8_9b<2> C0_8_9b<3> S0_8b<0>
+ S0_8b<1> S0_8b<2> S0_8b<3> vdd / FA_pt_4
XI182 P0_7<0> P0_7<3> P0_7<6> P1_7<1> P1_7<4> P0_7<1> P0_7<4> P0_7<7> P1_7<2>
+ P1_7<5> P0_7<2> P0_7<5> P1_7<0> P1_7<3> P1_7<6> C0_7_8b<0> C0_7_8b<1>
+ C0_7_8b<2> C0_7_8b<3> C0_7_8b<4> S0_7b<0> S0_7b<1> S0_7b<2> S0_7b<3>
+ S0_7b<4> vdd / FA_pt_5
XI260 CK_in Rb<0> Rb<1> Rb<2> Rb<3> Rb<4> Rb<5> R<6> R<7> R<8> R<9> R<10>
+ R<11> R<12> R<13> R<14> R<15> Rb<16> Yb<0> Yb<1> Yb<2> Yb<3> Yb<4> Yb<5>
+ Y<6> Y<7> Y<8> Y<9> Y<10> Y<11> Y<12> Y<13> Y<14> Y<15> Yb<16> Y<0> Y<1>
+ Y<2> Y<3> Y<4> Y<5> Yb<6> Yb<7> Yb<8> Yb<9> Yb<10> Yb<11> Yb<12> Yb<13>
+ Yb<14> Yb<15> Y<16> Resetb vdd / DETFF_Re_17_CMAC
.ENDS
```

```
********************************************************************************
* Library Name: CMACs
* Cell Name:    CMUL_8bit_Im
* View Name:    schematic
********************************************************************************

.SUBCKT CMUL_8bit_Im CK_in Resetb Y<0> Y<1> Y<2> Y<3> Y<4> Y<5> Y<6> Y<7> Y<8>
+ Y<9> Y<10> Y<11> Y<12> Y<13> Y<14> Y<15> Y<16> Yb<0> Yb<1> Yb<2> Yb<3> Yb<4>
+ Yb<5> Yb<6> Yb<7> Yb<8> Yb<9> Yb<10> Yb<11> Yb<12> Yb<13> Yb<14> Yb<15>
+ Yb<16> a<0> a<1> a<2> a<3> a<4> a<5> a<6> a<7> ab<0> ab<1> ab<2> ab<3> ab<4>
+ ab<5> ab<6> ab<7> b<0> b<1> b<2> b<3> b<4> b<5> b<6> b<7> bb<0> bb<1> bb<2>
+ bb<3> bb<4> bb<5> bb<6> bb<7> c<0> c<1> c<2> c<3> c<4> c<5> c<6> c<7> cb<0>
+ cb<1> cb<2> cb<3> cb<4> cb<5> cb<6> cb<7> d<0> d<1> d<2> d<3> d<4> d<5> d<6>
+ d<7> db<0> db<1> db<2> db<3> db<4> db<5> db<6> db<7> vdd
*.PININFO CK_in:I Resetb:I a<0>:I a<1>:I a<2>:I a<3>:I a<4>:I a<5>:I a<6>:I
*.PININFO a<7>:I ab<0>:I ab<1>:I ab<2>:I ab<3>:I ab<4>:I ab<5>:I ab<6>:I
*.PININFO ab<7>:I b<0>:I b<1>:I b<2>:I b<3>:I b<4>:I b<5>:I b<6>:I b<7>:I
*.PININFO bb<0>:I bb<1>:I bb<2>:I bb<3>:I bb<4>:I bb<5>:I bb<6>:I bb<7>:I
*.PININFO c<0>:I c<1>:I c<2>:I c<3>:I c<4>:I c<5>:I c<6>:I c<7>:I cb<0>:I
*.PININFO cb<1>:I cb<2>:I cb<3>:I cb<4>:I cb<5>:I cb<6>:I cb<7>:I d<0>:I
*.PININFO d<1>:I d<2>:I d<3>:I d<4>:I d<5>:I d<6>:I d<7>:I db<0>:I db<1>:I
*.PININFO db<2>:I db<3>:I db<4>:I db<5>:I db<6>:I db<7>:I Y<0>:O Y<1>:O Y<2>:O
*.PININFO Y<3>:O Y<4>:O Y<5>:O Y<6>:O Y<7>:O Y<8>:O Y<9>:O Y<10>:O Y<11>:O
*.PININFO Y<12>:O Y<13>:O Y<14>:O Y<15>:O Y<16>:O Yb<0>:O Yb<1>:O Yb<2>:O
*.PININFO Yb<3>:O Yb<4>:O Yb<5>:O Yb<6>:O Yb<7>:O Yb<8>:O Yb<9>:O Yb<10>:O
*.PININFO Yb<11>:O Yb<12>:O Yb<13>:O Yb<14>:O Yb<15>:O Yb<16>:O vdd:B
XI211 d<1> a<7> P1_8<0> vdd / nand2
XI219 d<0> a<7> P1_7<0> vdd / nand2
XI160 d<3> a<7> P1_10<0> vdd / nand2
XI203 d<2> a<7> P1_9<0> vdd / nand2
XI157 d<4> a<7> P1_11<0> vdd / nand2
XI154 d<5> a<7> P1_12<0> vdd / nand2
XI172 b<0> c<4> P0_4<0> vdd / nand2
XI171 b<0> c<1> P0_1<0> vdd / nand2
XI170 b<0> c<2> P0_2<0> vdd / nand2
XI169 b<0> c<3> P0_3<0> vdd / nand2
XI168 b<0> c<0> P0_0 vdd / nand2
XI167 b<1> c<3> P0_4<1> vdd / nand2
XI166 b<1> c<1> P0_2<1> vdd / nand2
XI165 b<1> c<2> P0_3<1> vdd / nand2
XI164 b<1> c<0> P0_1<1> vdd / nand2
XI163 b<2> c<1> P0_3<2> vdd / nand2
XI162 b<2> c<0> P0_2<2> vdd / nand2
XI161 b<2> c<2> P0_4<2> vdd / nand2
XI159 b<3> c<1> P0_4<3> vdd / nand2
XI158 b<3> c<0> P0_3<3> vdd / nand2
XI155 b<4> c<0> P0_4<4> vdd / nand2
XI119 d<7> a<1> P1_8<6> vdd / nand2
XI121 d<7> a<4> P1_11<3> vdd / nand2
XI117 d<7> a<0> P1_7<7> vdd / nand2
XI120 d<7> a<3> P1_10<4> vdd / nand2
XI146 b<0> c<5> P0_5<0> vdd / nand2
XI145 b<0> c<6> P0_6<0> vdd / nand2
XI143 b<1> c<4> P0_5<1> vdd / nand2
XI142 b<1> c<5> P0_6<1> vdd / nand2
XI141 b<1> c<6> P0_7<1> vdd / nand2
XI139 b<2> c<3> P0_5<2> vdd / nand2
XI138 b<2> c<4> P0_6<2> vdd / nand2
XI137 b<2> c<5> P0_7<2> vdd / nand2
XI136 b<2> c<6> P0_8<1> vdd / nand2
XI134 b<3> c<3> P0_6<3> vdd / nand2
```

```
XI133 b<3> c<2> P0_5<3> vdd / nand2
XI132 b<3> c<4> P0_7<3> vdd / nand2
XI131 b<3> c<5> P0_8<2> vdd / nand2
XI130 b<3> c<6> P0_9<1> vdd / nand2
XI128 b<4> c<5> P0_9<2> vdd / nand2
XI127 b<4> c<4> P0_8<3> vdd / nand2
XI126 b<4> c<2> P0_6<4> vdd / nand2
XI125 b<4> c<1> P0_5<4> vdd / nand2
XI124 b<4> c<3> P0_7<4> vdd / nand2
XI123 b<4> c<6> P0_10<1> vdd / nand2
XI95 b<5> c<3> P0_8<4> vdd / nand2
XI96 b<5> c<1> P0_6<5> vdd / nand2
XI97 b<5> c<0> P0_5<5> vdd / nand2
XI98 b<5> c<2> P0_7<5> vdd / nand2
XI99 b<5> c<4> P0_9<3> vdd / nand2
XI100 b<5> c<5> P0_10<2> vdd / nand2
XI101 b<5> c<6> P0_11<1> vdd / nand2
XI103 b<6> c<6> P0_12<1> vdd / nand2
XI104 b<6> c<5> P0_11<2> vdd / nand2
XI105 b<6> c<4> P0_10<3> vdd / nand2
XI106 b<6> c<2> P0_8<5> vdd / nand2
XI107 b<6> c<0> P0_6<6> vdd / nand2
XI108 b<6> c<1> P0_7<6> vdd / nand2
XI109 b<6> c<3> P0_9<4> vdd / nand2
XI122 d<7> a<5> P1_12<2> vdd / nand2
XI151 d<6> a<7> P1_13<0> vdd / nand2
XI118 d<7> a<2> P1_9<5> vdd / nand2
XI116 d<7> a<6> P1_13<1> vdd / nand2
XI91 b<7> c<7> P0_14 vdd / nand2
MN1 LL net0110 gnd! gnd! nfet m=1 w=180n l=30n nf=1.0 pccrit=1 as=18f ad=18f
+ ps=560n pd=560n sa=115n sb=115n sd=100n ptwell=0 ngcon=1 p_la=0
XI226 P0_9<0> P0_9<3> P1_9<0> P1_9<3> P0_9<1> P0_9<4> P1_9<1> P1_9<4> P0_9<2>
+ P0_9<5> P1_9<2> P1_9<5> C0_9_10b<0> C0_9_10b<1> C0_9_10b<2> C0_9_10b<3>
+ S0_9b<0> S0_9b<1> S0_9b<2> S0_9b<3> vdd / FA_pt_4
XI300 P0_6<0> P0_6<3> P0_6<6> P1_6<2> P0_6<1> P0_6<4> P1_6<0> P1_6<3> P0_6<2>
+ P0_6<5> P1_6<1> P1_6<4> C0_6_7b<0> C0_6_7b<1> C0_6_7b<2> C0_6_7b<3> S0_6b<0>
+ S0_6b<1> S0_6b<2> S0_6b<3> vdd / FA_pt_4
XI229 P0_5<0> P0_5<3> P1_5<0> P1_5<3> P0_5<1> P0_5<4> P1_5<1> P1_5<4> P0_5<2>
+ P0_5<5> P1_5<2> P1_5<5> C0_5_6b<0> C0_5_6b<1> C0_5_6b<2> C0_5_6b<3> S0_5b<0>
+ S0_5b<1> S0_5b<2> S0_5b<3> vdd / FA_pt_4
XI223 P0_8<0> P0_8<3> P0_8<6> P1_8<2> P0_8<1> P0_8<4> P1_8<0> P1_8<3> P0_8<2>
+ P0_8<5> P1_8<1> P1_8<4> C0_8_9b<0> C0_8_9b<1> C0_8_9b<2> C0_8_9b<3> S0_8b<0>
+ S0_8b<1> S0_8b<2> S0_8b<3> vdd / FA_pt_4
XI317 C3_5_6b S3_6b C4_6_7b S4_6b vdd / HA_pt
XI315 C3_2_3b S3_3b C4_3_4b Rb<3> vdd / HA_pt
XI314 C2_6_7b<1> C1_6_7b<2> C3_7_8b<1> S3_7b<1> vdd / HA_pt
XI288 C3_7_8b<1> S4_8b C5_8_9b S5_8b vdd / HA_pt
XI312 C2_1_2b S2_2b C3_2_3b Rb<2> vdd / HA_pt
XI310 P1_1<1> S1_1b C2_1_2b Rb<1> vdd / HA_pt
XI307 C0_4_5b<2> S0_5b<3> C1_5_6b<2> S1_5b<2> vdd / HA_pt
XI292 S3_12b C3_11_12b C4_11_12b C5_12_13b S5_12b vdd / FA_pt
XI291 S4_11b C3_10_11b C4_10_11b C5_11_12b S5_11b vdd / FA_pt
XI290 S4_10b C3_9_10b C4_9_10b C5_10_11b S5_10b vdd / FA_pt
XI289 S4_9b C3_8_9b<1> C4_8_9b C5_9_10b S5_9b vdd / FA_pt
XI284 S2_13b C2_12_13b C3_12_13b C4_13_14b S4_13b vdd / FA_pt
XI283 S3_11b C2_10_11b<0> C2_10_11b<1> C4_11_12b S4_11b vdd / FA_pt
XI282 S3_10b C2_9_10b<0> C2_9_10b<1> C4_10_11b S4_10b vdd / FA_pt
XI281 S3_9b C2_8_9b<1> C3_8_9b<0> C4_9_10b S4_9b vdd / FA_pt
XI280 S3_8b<0> S3_8b<1> C3_7_8b<0> C4_8_9b S4_8b vdd / FA_pt
XI279 S3_7b<0> S3_7b<1> C3_6_7b C4_7_8b S4_7b vdd / FA_pt
XI278 S3_5b C2_4_5b C3_4_5b C4_5_6b S4_5b vdd / FA_pt
XI316 S3_4b C2_3_4b C3_3_4b C4_4_5b S4_4b vdd / FA_pt
```

```
XI276 C1_7_8b<2> C1_7_8b<3> C2_7_8b<1> C3_8_9b<1> S3_8b<1> vdd / FA_pt
XI275 S1_14b C1_13_14b C2_13_14b C3_14_15b S3_14b vdd / FA_pt
XI274 S2_12b C1_11_12b<1> C2_11_12b C3_12_13b S3_12b vdd / FA_pt
XI273 S2_11b C1_10_11b<0> C1_10_11b<1> C3_11_12b S3_11b vdd / FA_pt
XI261 S1_7b<2> S1_7b<3> C1_6_7b<1> C2_7_8b<1> S2_7b<1> vdd / FA_pt
XI311 S0_2b<1> S1_2b C1_1_2b C2_2_3b S2_2b vdd / FA_pt
XI231 P0_11<3> P1_11<0> P1_11<1> C0_11_12b<1> S0_11b<1> vdd / FA_pt
XI272 S2_10b<0> S2_10b<1> C1_9_10b<2> C3_10_11b S3_10b vdd / FA_pt
XI264 C1_9_10b<0> C1_9_10b<1> C0_9_10b<3> C2_10_11b<1> S2_10b<1> vdd / FA_pt
XI263 S1_9b<2> C1_8_9b<1> C1_8_9b<2> C2_9_10b<1> S2_9b<1> vdd / FA_pt
XI262 S1_8b<2> C1_7_8b<0> C1_7_8b<1> C2_8_9b<1> S2_8b<1> vdd / FA_pt
XI270 S2_8b<0> S2_8b<1> C2_7_8b<0> C3_8_9b<0> S3_8b<0> vdd / FA_pt
XI260 S1_6b<2> C1_5_6b<1> C1_5_6b<2> C2_6_7b<1> S2_6b<1> vdd / FA_pt
XI259 S1_13b C1_12_13b C0_12_13b<1> C2_13_14b S2_13b vdd / FA_pt
XI258 S1_12b C1_11_12b<0> C0_11_12b<1> C2_12_13b S2_12b vdd / FA_pt
XI257 S1_11b<0> S1_11b<1> C0_10_11b<2> C2_11_12b S2_11b vdd / FA_pt
XI248 S0_3b<2> S1_3b C0_2_3b<1> C2_3_4b S2_3b vdd / FA_pt
XI246 C0_10_11b<0> C0_10_11b<1> P1_11<3> C1_11_12b<1> S1_11b<1> vdd / FA_pt
XI245 S0_10b<2> C0_9_10b<0> C0_9_10b<1> C1_10_11b<1> S1_10b<1> vdd / FA_pt
XI238 C0_13_14b P0_14 P1_14 C1_14_15b S1_14b vdd / FA_pt
XI237 S0_13b C0_12_13b<0> P1_13<1> C1_13_14b S1_13b vdd / FA_pt
XI236 S0_12b<0> S0_12b<1> C0_11_12b<0> C1_12_13b S1_12b vdd / FA_pt
XI235 S0_11b<0> S0_11b<1> P1_11<2> C1_11_12b<0> S1_11b<0> vdd / FA_pt
XI234 S0_10b<0> S0_10b<1> P1_10<4> C1_10_11b<0> S1_10b<0> vdd / FA_pt
XI233 S0_5b<0> S0_5b<1> C0_4_5b<0> C1_5_6b<0> S1_5b<0> vdd / FA_pt
XI42 P0_3<0> P0_3<1> P0_3<2> C0_3_4b<0> S0_3b<0> vdd / FA_pt
XI40 P0_2<0> P0_2<1> P0_2<2> C0_2_3b<0> S0_2b<0> vdd / FA_pt
XI39 P0_1<0> P0_1<1> P1_1<0> C0_1_2b S0_1b vdd / FA_pt
XI222 P0_12<0> P0_12<1> P0_12<2> C0_12_13b<0> S0_12b<0> vdd / FA_pt
XI221 P0_11<0> P0_11<1> P0_11<2> C0_11_12b<0> S0_11b<0> vdd / FA_pt
XI256 S1_10b<0> S1_10b<1> C0_9_10b<2> C2_10_11b<0> S2_10b<0> vdd / FA_pt
XI252 S1_6b<0> S1_6b<1> C1_5_6b<0> C2_6_7b<0> S2_6b<0> vdd / FA_pt
XI267 S1_5b<2> S2_5b C1_4_5b<1> C3_5_6b S3_5b vdd / FA_pt
XI251 S1_5b<0> S1_5b<1> C1_4_5b<0> C2_5_6b S2_5b vdd / FA_pt
XI254 S1_8b<0> S1_8b<1> C0_7_8b<4> C2_8_9b<0> S2_8b<0> vdd / FA_pt
XI250 S1_4b<0> S1_4b<1> C0_3_4b<1> C2_4_5b S2_4b vdd / FA_pt
XI269 S2_7b<0> S2_7b<1> C2_6_7b<0> C3_7_8b<0> S3_7b<0> vdd / FA_pt
XI266 S2_4b C0_3_4b<2> C1_3_4b C3_4_5b S3_4b vdd / FA_pt
XI253 S1_7b<0> S1_7b<1> C1_6_7b<0> C2_7_8b<0> S2_7b<0> vdd / FA_pt
XI255 S1_9b<0> S1_9b<1> C1_8_9b<0> C2_9_10b<0> S2_9b<0> vdd / FA_pt
XI271 S2_9b<0> S2_9b<1> C2_8_9b<0> C3_9_10b S3_9b vdd / FA_pt
XI268 S2_6b<0> S2_6b<1> C2_5_6b C3_6_7b S3_6b vdd / FA_pt
XI313 S2_3b C1_2_3b C2_2_3b C3_3_4b S3_3b vdd / FA_pt
XI43 P0_3<3> P1_3<0> P1_3<1> C0_3_4b<1> S0_3b<1> vdd / FA_pt
XI220 P0_13<0> P0_13<1> P1_13<0> C0_13_14b S0_13b vdd / FA_pt
XI230 P1_12<0> P1_12<1> P1_12<2> C0_12_13b<1> S0_12b<1> vdd / FA_pt
XI51 S0_4b<0> S0_4b<1> P1_4<4> C1_4_5b<0> S1_4b<0> vdd / FA_pt
XI49 S0_3b<0> S0_3b<1> C0_2_3b<0> C1_3_4b S1_3b vdd / FA_pt
XI41 P1_2<0> P1_2<1> P1_2<2> C0_2_3b<1> S0_2b<1> vdd / FA_pt
XI193 db<3> ab<4> P1_7<3> vdd / nor2_alt
XI218 db<0> ab<0> P1_0 vdd / nor2_alt
XI214 db<0> ab<6> P1_6<0> vdd / nor2_alt
XI217 db<0> ab<3> P1_3<0> vdd / nor2_alt
XI215 db<0> ab<1> P1_1<0> vdd / nor2_alt
XI201 db<2> ab<5> P1_7<2> vdd / nor2_alt
XI206 db<1> ab<2> P1_3<1> vdd / nor2_alt
XI183 db<4> ab<4> P1_8<3> vdd / nor2_alt
XI195 db<3> ab<6> P1_9<1> vdd / nor2_alt
XI202 db<2> ab<6> P1_8<1> vdd / nor2_alt
XI198 db<2> ab<0> P1_2<2> vdd / nor2_alt
XI216 db<0> ab<2> P1_2<0> vdd / nor2_alt
XI212 db<0> ab<4> P1_4<0> vdd / nor2_alt
```

```
XI204 db<1> ab<3> P1_4<1> vdd / nor2_alt
XI177 db<5> ab<3> P1_8<4> vdd / nor2_alt
XI156 db<4> ab<6> P1_10<1> vdd / nor2_alt
XI208 db<1> ab<4> P1_5<1> vdd / nor2_alt
XI213 db<0> ab<5> P1_5<0> vdd / nor2_alt
XI207 db<1> ab<0> P1_1<1> vdd / nor2_alt
XI197 db<2> ab<1> P1_3<2> vdd / nor2_alt
XI152 db<5> ab<5> P1_10<2> vdd / nor2_alt
XI185 db<4> ab<0> P1_4<4> vdd / nor2_alt
XI199 db<2> ab<2> P1_4<2> vdd / nor2_alt
XI205 db<1> ab<1> P1_2<1> vdd / nor2_alt
XI209 db<1> ab<5> P1_6<1> vdd / nor2_alt
XI191 db<3> ab<0> P1_3<3> vdd / nor2_alt
XI186 db<4> ab<1> P1_5<4> vdd / nor2_alt
XI196 db<2> ab<3> P1_5<2> vdd / nor2_alt
XI210 db<1> ab<6> P1_7<1> vdd / nor2_alt
XI190 db<3> ab<1> P1_4<3> vdd / nor2_alt
XI179 db<5> ab<0> P1_5<5> vdd / nor2_alt
XI153 db<5> ab<6> P1_11<1> vdd / nor2_alt
XI184 db<4> ab<2> P1_6<4> vdd / nor2_alt
XI200 db<2> ab<4> P1_6<2> vdd / nor2_alt
XI192 db<3> ab<2> P1_5<3> vdd / nor2_alt
XI178 db<5> ab<1> P1_6<5> vdd / nor2_alt
XI187 db<4> ab<3> P1_7<4> vdd / nor2_alt
XI188 db<3> ab<3> P1_6<3> vdd / nor2_alt
XI180 db<5> ab<2> P1_7<5> vdd / nor2_alt
XI175 db<6> ab<0> P1_6<6> vdd / nor2_alt
XI176 db<6> ab<1> P1_7<6> vdd / nor2_alt
XI149 db<6> ab<6> P1_12<1> vdd / nor2_alt
XI173 db<6> ab<3> P1_9<4> vdd / nor2_alt
XI115 db<7> ab<7> P1_14 vdd / nor2_alt
XI148 db<6> ab<4> P1_10<3> vdd / nor2_alt
XI150 db<6> ab<5> P1_11<2> vdd / nor2_alt
XI147 bb<0> cb<7> P0_7<0> vdd / nor2_alt
XI144 bb<1> cb<7> P0_8<0> vdd / nor2_alt
XI140 bb<2> cb<7> P0_9<0> vdd / nor2_alt
XI135 bb<3> cb<7> P0_10<0> vdd / nor2_alt
XI129 bb<4> cb<7> P0_11<0> vdd / nor2_alt
XI114 bb<5> cb<7> P0_12<0> vdd / nor2_alt
XI102 bb<6> cb<7> P0_13<0> vdd / nor2_alt
XI181 db<5> ab<4> P1_9<3> vdd / nor2_alt
XI182 db<4> ab<5> P1_9<2> vdd / nor2_alt
XI194 db<3> ab<5> P1_8<2> vdd / nor2_alt
XI174 db<6> ab<2> P1_8<5> vdd / nor2_alt
XI94 bb<7> cb<5> P0_12<2> vdd / nor2_alt
XI93 bb<7> cb<4> P0_11<3> vdd / nor2_alt
XI110 bb<7> cb<3> P0_10<4> vdd / nor2_alt
XI111 bb<7> cb<1> P0_8<6> vdd / nor2_alt
XI112 bb<7> cb<2> P0_9<5> vdd / nor2_alt
XI113 bb<7> cb<0> P0_7<7> vdd / nor2_alt
XI92 bb<7> cb<6> P0_13<1> vdd / nor2_alt
XI241 S0_7b<0> S0_7b<2> S0_7b<4> S0_7b<1> S0_7b<3> P1_7<7> C0_6_7b<0>
+ C0_6_7b<1> C0_6_7b<2> C1_7_8b<0> C1_7_8b<1> C1_7_8b<2> S1_7b<0> S1_7b<1>
+ S1_7b<2> vdd / FA_pt_3
XI224 P0_10<0> P0_10<3> P1_10<1> P0_10<1> P0_10<4> P1_10<2> P0_10<2> P1_10<0>
+ P1_10<3> C0_10_11b<0> C0_10_11b<1> C0_10_11b<2> S0_10b<0> S0_10b<1>
+ S0_10b<2> vdd / FA_pt_3
XI228 P0_4<0> P0_4<3> P1_4<1> P0_4<1> P0_4<4> P1_4<2> P0_4<2> P1_4<0> P1_4<3>
+ C0_4_5b<0> C0_4_5b<1> C0_4_5b<2> S0_4b<0> S0_4b<1> S0_4b<2> vdd / FA_pt_3
XI309 S0_9b<0> S0_9b<2> C0_8_9b<2> S0_9b<1> S0_9b<3> C0_8_9b<3> C0_8_9b<0>
+ C0_8_9b<1> C0_8_9b<4> C1_9_10b<0> C1_9_10b<1> C1_9_10b<2> S1_9b<0> S1_9b<1>
+ S1_9b<2> vdd / FA_pt_3
```

```
XI240 S0_6b<0> S0_6b<2> S0_6b<4> S0_6b<1> S0_6b<3> C0_5_6b<2> C0_5_6b<0>
+ C0_5_6b<1> C0_5_6b<3> C1_6_7b<0> C1_6_7b<1> C1_6_7b<2> S1_6b<0> S1_6b<1>
+ S1_6b<2> vdd / FA_pt_3
XI239 S0_8b<0> S0_8b<2> S0_8b<4> S0_8b<1> S0_8b<3> C0_7_8b<2> C0_7_8b<0>
+ C0_7_8b<1> C0_7_8b<3> C1_8_9b<0> C1_8_9b<1> C1_8_9b<2> S1_8b<0> S1_8b<1>
+ S1_8b<2> vdd / FA_pt_3
XI227 P0_7<0> P0_7<3> P0_7<6> P1_7<1> P1_7<4> P0_7<1> P0_7<4> P0_7<7> P1_7<2>
+ P1_7<5> P0_7<2> P0_7<5> P1_7<0> P1_7<3> P1_7<6> C0_7_8b<0> C0_7_8b<1>
+ C0_7_8b<2> C0_7_8b<3> C0_7_8b<4> S0_7b<0> S0_7b<1> S0_7b<2> S0_7b<3>
+ S0_7b<4> vdd / FA_pt_5
XI308 C0_6_7b<3> C0_6_7b<4> C1_7_8b<3> S1_7b<3> vdd / HA_add_1_pt
XI302 P1_8<5> P1_8<6> C0_8_9b<4> S0_8b<4> vdd / HA_add_1_pt
XI301 P1_6<5> P1_6<6> C0_6_7b<4> S0_6b<4> vdd / HA_add_1_pt
XI298 P1_3<2> P1_3<3> C0_3_4b<2> S0_3b<2> vdd / HA_add_1_pt
XI297 P0_0 P1_0 C0_0_1b Rb<0> vdd / HA_add_1_pt
XI306 C0_4_5b<1> S0_5b<2> C1_5_6b<1> S1_5b<1> vdd / HA_add_1_pt
XI303 C0_0_1b S0_1b C1_1_2b S1_1b vdd / HA_add_1_pt
XI304 C0_1_2b S0_2b<0> C1_2_3b S1_2b vdd / HA_add_1_pt
XI305 C0_3_4b<0> S0_4b<2> C1_4_5b<1> S1_4b<1> vdd / HA_add_1_pt
MP3 net0110 net0110 vdd vdd pfet m=1 w=269n l=30n nf=1.0 pccrit=1 as=26.9f
+ ad=26.9f ps=738n pd=738n sa=115n sb=115n sd=100n ptwell=0 ngcon=1 p_la=0
XI319 C4_7_8b C5_8_9b C5_9_10b C5_10_11b S5_8b S5_9b S5_10b S5_11b CR_7_8
+ CR_11_12 R<8> R<9> R<10> R<11> vdd / CLA4
XI318 C4_3_4b C4_4_5b C4_5_6b C4_6_7b S4_4b S4_5b S4_6b S4_7b LL CR_7_8 R<4>
+ R<5> R<6> R<7> vdd / CLA4
XI320 C5_11_12b C5_12_13b C4_13_14b C3_14_15b S5_12b S4_13b S3_14b C1_14_15b
+ CR_11_12 Rb<16> R<12> R<13> R<14> R<15> vdd / CLA4
XI295 CK_in Rb<0> Rb<1> Rb<2> Rb<3> R<4> R<5> R<6> R<7> R<8> R<9> R<10> R<11>
+ R<12> R<13> R<14> R<15> Rb<16> Yb<0> Yb<1> Yb<2> Yb<3> Y<4> Y<5> Y<6> Y<7>
+ Y<8> Y<9> Y<10> Y<11> Y<12> Y<13> Y<14> Y<15> Yb<16> Y<0> Y<1> Y<2> Y<3>
+ Yb<4> Yb<5> Yb<6> Yb<7> Yb<8> Yb<9> Yb<10> Yb<11> Yb<12> Yb<13> Yb<14>
+ Yb<15> Y<16> Resetb vdd / DETFF_Im_17_CMAC
.ENDS
```

Appendix B

List of publications

The following list contains publications that have been based on the work presented in this thesis:

- S. Lapshev and S. M. R. Hasan, “New Low Glitch and Low Power DET Flip-Flops Using Multiple C-Elements”. In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 63.10 (Oct. 2016), pp. 1673–1681.
- S. Lapshev and S. M. R. Hasan, “On the architecture for the X part of a very large FX correlator using two-accumulator CMACs”. In: *Experimental Astronomy* 41.1 (Feb. 2016), pp. 259–270.
- S. Lapshev and S. M. R. Hasan, “Using multiple-accumulator CMACs to improve efficiency of the X part of an input-buffered FX correlator”. In: *Experimental Astronomy* 43.2 (Apr. 2017), pp. 177–187.

SPRINGER NATURE LICENSE
TERMS AND CONDITIONS

Jun 14, 2018

This Agreement between Mr. Stepan Lapshev ("You") and Springer Nature ("Springer Nature") consists of your license details and the terms and conditions provided by Springer Nature and Copyright Clearance Center.

License Number	4367570871035
License date	Jun 14, 2018
Licensed Content Publisher	Springer Nature
Licensed Content Publication	Experimental Astronomy
Licensed Content Title	Using multiple-accumulator CMACs to improve efficiency of the X part of an input-buffered FX correlator
Licensed Content Author	Stepan Lapshev, S. M. Rezaul Hasan
Licensed Content Date	Jan 1, 2017
Licensed Content Volume	43
Licensed Content Issue	2
Type of Use	Thesis/Dissertation
Requestor type	academic/university or research institute
Format	print and electronic
Portion	full article/chapter
Will you be translating?	no
Circulation/distribution	<501
Author of this Springer Nature content	yes
Title	CMOS VLSI Correlator Design for Radio-Astronomical Signal Processing
Instructor name	Rezaul Hasan
Institution name	Massey University
Expected presentation date	Jul 2018
Total	0.00 USD