**QUEEN'S
UNIVERSITY
BELFAST**

Secure Design Flow for Asynchronous Multi-valued Logic Circuits

Rafiev, A., Murphy, J., & Yakovlev, A. (2010). *Secure Design Flow for Asynchronous Multi-valued Logic Circuits*. 264-269. Paper presented at 40th IEEE International Symposium on Multiple-Valued Logic, Spain.
<https://doi.org/10.1109/ISMVL.2010.56>

Queen's University Belfast - Research Portal:

[Link to publication record in Queen's University Belfast Research Portal](#)

General rights

Copyright for the publications made accessible via the Queen's University Belfast Research Portal is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy

The Research Portal is Queen's institutional repository that provides access to Queen's research output. Every effort has been made to ensure that content in the Research Portal does not infringe any person's rights, or applicable UK laws. If you discover content in the Research Portal that you believe breaches copyright or violates any law, please contact openaccess@qub.ac.uk.

Secure Design Flow for Asynchronous Multi-Valued Logic Circuits

Ashur Rafiev, Julian P. Murphy, Alex Yakovlev
School of Electrical, Electronic & Computer Engineering
Newcastle University, UK
{ashur.rafiev, j.p.murphy, alex.yakovlev}@ncl.ac.uk

Abstract—The purpose of secure devices such as smartcards is to protect secret information against software and hardware attacks. Implementation of the appropriate protection techniques often implies non-standard methods that are not supported by the conventional design tools. In the recent decade the designers of secure devices have been working hard on customising the workflow. The presented research aims to collect the up-to-date experiences in this area and create a generic approach to the secure design flow that can be used as guidance by engineers.

In the presented paper the emphasis is put on multi-valued logic synthesis and asynchronous system design. The proposed flow employs the tool based on higher radix and mixed radix Reed-Muller expansions [1], power-balanced logic component libraries and TiDE design environment [2]. The challenge of the research here is interfacing between different EDA tools and technologies. An example is also presented and described from the view of the system designer.

I. INTRODUCTION

Smartcards are highly complex devices combining state of the art technologies in order to protect secret information against possible attacks at both software and hardware levels [3]. Non-invasive hardware attacks, also called side-channel attacks, represent higher threat due to their availability. During this type of attacks the attacker uses weaknesses of certain hardware implementations of cryptographic algorithms to reveal a secret key. In particular such information can be obtained by observing the data-dependent power consumption of a device (power analysis) or analysing the number of clock cycles used between operations (timing analysis).

In order to develop reliable hardware with the required level of security, engineers have to introduce special techniques altering the standard design environment. This article has presents analysis of designers' experience in this area. The most appropriate countermeasures are discussed in Section II, they include such techniques as multi-valued and switching balanced logic [4], [5] and clockless design [6].

Multi-valued logic (MVL) synthesis has been developing for many years, nevertheless it has not been applied in commonly used multi-purpose industrial design environments yet. MVL examples appeared in practice as separate custom-design projects weakly linked with the traditional EDA flow. Similar problems happen with the asynchronous approach. Although a number of design environments have been developed and are used in industry [7], [2], they are based on specific protocols and sometimes lack in flexibility.

As a result, each secure system takes a lot of effort to be designed, and the whole design process from the high-level description to the physical floor-planning implies a lot of customisation and has many bottlenecks. However in most cases designers struggle against similar tasks, so it appears essential to combine existing methodologies in a generic forward-looking philosophy of the secure design flow.

The goal of the paper is to bind MVL synthesis approach together with other security-related techniques within a single workflow, which is assumed to include such features as conformity with the security requirements, coherence with industrial tools where possible, modularity and flexibility. The key idea, intended to optimise the design results, is to apply the security features at the initial pre-synthesis stage of the workflow rather than to modify (convert) the final netlist.

Modular approach of the design flow implies independent consideration of the data path and the control path development; consequently there is a possibility to use various frameworks if it improves the results. Hence the difficulty being worked on is interfacing between tools from different design environments. Although all EDA tools tend to use compatible input and output formats, many of them imply certain technologies and protocols not acceptable by others.

The proposed design flow promises to help smartcard engineers in their work and to put more practical effort into the MVL research.

This paper is organised as follows. Section II discusses types of side-channel attacks and countermeasures. Section III analyses existing design environments and systematises collected knowledge with respect to the proposed secure design flow. Section IV describes the technical features and capabilities of the tool used for MVL synthesis. Section V presents an example of design using the proposed flow. Section VI concludes the work.

II. TYPES OF ATTACKS AND COUNTERMEASURES

Hardware attacks are divided into two major categories: invasive and non-invasive (side-channel). Based on reverse engineering, invasive attacks require special laboratory equipment and destroy packaging in the process while side-channel attacks do not require in-depth knowledge of the technology and use simple equipment [3].

During the *power analysis* the hacker monitors data-dependent power consumption of the device during the normal

operation [8]. The method relies on the following fundamental hypothesis: there exists an intermediate variable that appears during the computation of the algorithm, such that knowing a few key bits allows the attacker to decide whether two inputs (respectively two outputs) give or not the same value for this variable [9]. Consequently, splitting such variables and combining with random values can protect against power analysis. This method is called *masking*, and its major advantage with respect to this work is that it can be implemented using standard EDA software. However the recent research discovered a mathematical modification of power analysis that can break the masking approach [10].

Another countermeasure to the power analysis is to make power consumption of the device data-independent, it is called *power balancing*. A number of methods to equalise power signatures of the logic using specific representation of data signals over physical wires, e.g. m-of-n codes, were proposed in [4], [6], [11], [12]. M-of-n codes are an encoding scheme in which data is represented using n wires and where m of them are set to an active level (usually high). A protocol separating data using dummy symbols (spacers) is called a *spacer protocol*. A particular emphasis has so far only been put on dual-rail (1-of-2) codes for binary radix and 1-of-4 for quaternary logic. As a price for the improved resistance to power attacks the approach results in an overhead with respect to the overall power consumption of the system.

Electromagnetic analysis is similar to power analysis but uses data-dependent electromagnetic emission instead of power consumption [13]. The method of equalising switching activity of the logic can work in this case as well.

Another way to make the hacker's life harder is to randomise timing properties of the circuits, so it would become difficult to perform correct sampling of physical parameters during operation. The simplest method is to insert random delays in the clock cycles. The most reliable countermeasure however is an asynchronous logic design (i.e. self-timed circuits) [6]. Asynchronously working device modules cause overlays in power consumption, thus making it practically impossible to distinguish between single operations. This can help against the *timing analysis*, which uses the amount of time required for running non constant cryptographic algorithm to retrieve information about the data processed [14].

Fault analysis is an attack which uses abnormal environment conditions, e.g. glitches on power or clock signals, so malfunction of the device can create a window for vulnerabilities [15]. A known countermeasure for this attack is using fault-tolerant protocols. M-of-n codes are fault-tolerant as they imply relatively simple fault detection logic [16].

Since m-of-n codes and clockless design approach appear to be the most universal countermeasures, the following discussion on the secure design is presented in relation to these ideas.

III. SECURE DESIGN FLOW

Considering trends in countermeasure methodology described in Section II, one can conclude that security requirements affect the design flow as follows.

- Data signal representations other than traditional binary (single-rail) signals are required, this may include multi-valued signals. This requirement implies advanced logic synthesis and affects the data path.
- Control path has to be synthesised using asynchronous system design techniques. For asynchronous designs it is possible to develop the data path and the control path concurrently since in general the control path does not rely on timing properties of the combinational logic.
- Physical layout can also impact on the security properties, since (i) it affects wire capacitance and consequently power balancing and (ii) floor-planning defines electromagnetic emission patterns.

Independent consideration of latter approaches allows using more efficient combinations of countermeasures. The whole design environment can be split into three parts: data path synthesis, control path synthesis and layout. Traditionally design environments consist of multiple tools, each is used for a different task. Thus partitioning the flow should not be a problem. However certain compatibility issues take place, so it is often not possible to create direct connection between tools from different design environments.

Another important point of view on the secure system design is the conversion driven design (CDD) approach, which suggests conversion of existing insecure circuits applying security features on top of the previously designed netlists. In this case the design environment structure is similar to the described above but uses conversion tools instead of synthesis tools. This may include tools for desynchronising clocked circuits and tools for converting signal representation in existing binary (single-rail) data paths.

A. Data path synthesis

The only signal representation supported by the industrial logic synthesis tools is traditional binary encoding, which is also called single-rail since one wire represents one bit of data. Consequently, this lack of support often discourages designers from using power balanced protocols. In order to make the proposed design flow more attractive to engineers we suggest using RTL-implemented power-balanced logic components.

As it was mentioned in the previous section, there are two approaches to the security applied design flow: conversion and synthesis. Conversion approach is believed to be more convenient for industrial development as it requires minimal modification to the design environment. The simplest way to apply power balancing is to map previously synthesised binary logic directly into power-balanced dual-rail components. This can be done using Verimap tool [11].

A significant drawback of dual-rail versus other m-of-n codes is the increased power consumption. An efficient solution to this problem is to use multi-valued logic (MVL) instead of binary. Due to the properties of m-of-n codes, and especially 1-of-n codes, higher radix signals produce less switching activity of wires reducing the power consumption. An attempt to use the conversion approach for MVL synthesis has been made in [17]. The presented tool groups pairs of binary gates into quaternary gates. However, due to the structure of the

Figure 1. CDD-based design flow

initial circuit, there are usually a number of gates that cannot be grouped and remain binary. Signal conversion components are used to connect parts of different radices, and the amount of this additional logic can be significantly large producing less efficient results.

In terms of the design flow, as can be observed from Figure 1, the approach has a number of drawbacks. First, a design compiler has to be used twice: to synthesise the original circuit and to merge the converted data path with the control path. This complication may negatively impact on the testability of the design, since it is very difficult to reflect possible errors back to the original high-level description. Second, since conversion at this stage is applied only to the data path, there should be some method to extract it from the flat gate-level netlist. Normally this is done by a conversion tool, but the very idea is rather weak.

All these bottlenecks can be avoided if MVL is applied not at the post-synthesis stage, but at the pre-synthesis stage. Certain modules in the initial HDL description can be replaced with the modules preprocessed (synthesised) using special tools. The paradigm of accommodating different radices within a circuit has been evolved in the work presented in [1], which uses Reed-Muller expansions based on Galois field arithmetic.

Since it considers the notion of mixed radices from the highest level of mathematical representation, the synthesised logic does not require internal signal conversion and shows significant improvement in power and area. A detailed description of the tool and its features is given in Section IV. In

Figure 2. Reed-Muller + Haste synthesis flow

terms of the design flow, the tool allows the designer to choose which modules to synthesise using Reed-Muller approach, i.e. to use power balancing locally if needed. It can be emerged into the flow as shown in Figure 2, the rest of the tools used in the illustrated flow are described further in Section III-B.

B. Control path synthesis

The flexibility of the secure design flow implies an “unconstrained” choice of the security countermeasures, so they can be used in different combinations if needed. It is important to take into consideration the case of power balanced but synchronous designs that may be sufficient for achieving certain level of security. Most of custom logic synthesis tools can output Verilog format, thus customised combinational logic can be merged with the control path using standard EDA tools. However, as the proposed design flow implies the security at all levels of the design, asynchronous methods should be used instead.

Desynchronised circuits: A conversion method for the control path has been described in [18]. The principal idea is to replace clocked registers in synchronous designs with handshake registers. Elastix tool [19] implements this approach. The input of the tool is a finalised synchronous design. Hence the step of merging customised combinational logic with a clocked control path is to be done before the stage of desynchronisation. The possibility to achieve the required secure properties without redesigning the whole system “from

scratch” motivates designers to use the desynchronisation approach. However, with respect to the conversion flow shown in Figure 1, the drawbacks of the data path conversion approach nullify all the benefits of the Elastix flow.

Handshake-based asynchronous circuits: Asynchronous circuits use the request-acknowledge interaction between components to control the system. This way of signalling is called *handshake protocol*, and it can be implemented in different ways. The tools for asynchronous control path synthesis use netlists of handshake components as an intermediate state of the design.

The most widely known toolkits for the asynchronous development are Balsa toolkit [7] and TiDE [2].

Balsa-based environment uses *balsa-c* compiler, which compiles a Balsa description into the netlist of handshake components, and *balsa-netlist* gate-level mapper. The major drawback of this environment with respect to the secure design flow is that it uses a predefined set of components, so it is impossible to use custom data representations in its standard workflow. The only solution is to build the control path in a separate module and then merge modules using third-party software.

The TiDE design environment, developed by Handshake Solutions, is based on the Haste description language and uses *htcomp* compiler to synthesise handshake component netlists; *htmap* substitutes abstract handshake components with gate-level modules; *htlink* is used to merge the control path with presynthesised (custom) combinational logic provided in a form of gate-level netlist.

At the moment TiDE appears to be the most flexible and industry-aware asynchronous design environment, and it has already been used in security applications. Figure 2 illustrates how the data path synthesis can be interfaced into TiDE.

Unfortunately, in terms of power balancing there is a conflict that has to be resolved. It is known that an asynchronous system can be designed using a single-rail bundled data protocol or m-of-n encoded spacer protocol [20]. In terms of security, the spacer protocol is essential for power balancing as it guarantees equal number of wires switching per period. Indeed, since an m-of-n encoded signal implies m wires to be set to 1, exactly m wires switch from 0 to 1 when the data arrives and reset back to 0 on spacer. Equalised switching of wires is fundamental for power balanced circuits, and exclusion of the spacer from the protocol also removes this property.

Based on the single-rail data format, TiDE uses bundled data protocol and cannot be directly used for m-of-n encoded power balanced designs. However, in systems which employ bundled data protocol the spacer may be emulated (enforced) by explicitly alternating data and spacers at the inputs. Simple for pipelined systems, this approach becomes rather difficult for the systems with loops in the structure. However the known solution is to use master-slave registers [11], [21], so one stores spacer, while another stores data and vice versa. This kind of behaviour can be described in the high-level code, but this requires in-depth knowledge of asynchronous design technology.

In contrast, the presented research attempts to minimise

+	0	1	A	B	×	0	1	A	B
0	0	1	A	B	0	0	0	0	0
1	1	0	B	A	1	0	1	A	B
A	A	B	0	1	A	0	A	B	1
B	B	A	1	0	B	0	B	1	A

Figure 3. Addition and multiplication over GF(4)

the designer’s effort and automate the process. Our proposal is to adjust the final bundled data asynchronous circuits by directly replacing memory cells with custom registers that inject spacers into existing paths. For example, the principle of “wagging” registers [5] implements spacer and data alternation using two flip-flops connected in parallel via multiplexer (in contrast to sequentially connected master-slave registers).

C. Physical design

Layout issue is highly important for power balancing since it defines the capacitance of wires and global electromagnetic emission patterns. In m-of-n codes, considering that each data signal is transferred using a set of wires, balanced switching activity makes sense only if the switching energies of wires in each bus are equal. Consequently the secure design flow should guarantee equalised wire lengths and fanout, i.e. the parameters affecting the switching energy of gates.

In general this can be done using advanced scripting in standard industrial layout tools. However the best way to implement power balanced layout is to route buses as parallel sets of wires, which is not supported by common tools. Pulsic software [22] specialising in a layout for custom designs can perform such routing of wires, so it is advised for security applications.

IV. MIXED RADIX REED-MULLER SYNTHESIS TOOL

Computation of the quaternary Reed-Muller (RM) expansions over Galois fields of radix 4 has a long research history [23], [24], [25], [26], [27], [28]. These expansions are popular due to the efficiency of their hardware implementations and testability. The research presented in [1] goes a step further and proposes to consolidate the arithmetic of different radices within a single circuit in such a way that the benefit seen in mathematical representation can be propagated down to silicon level.

Current implementation covers binary and quaternary Galois fields, namely GF(2) and GF(4). In GF(2) the operation of addition refers to the binary XOR operation, and the operation of multiplication refers to the binary AND. Addition and multiplication over GF(4) differ from bitwise operations and, denoting elements of GF(4) as 0, 1, A, and B, can be defined as shown in Figure 3. Ternary radix over GF(3) has also been added for the experimental purposes but currently has limited support.

Quaternary RM expansions have a form of the sum of products in Galois field arithmetic. They also may include different inversion (polarity) forms of function arguments. If each argument appears in the expansion in the same polarity, the expansion is called *fixed polarity* expansion. For n -valued

quaternary function there are 4^n possible fixed polarity forms. Formal definitions and a method of computation can be found in [29].

The principle of the mixed radix RM expansions is based on providing all or some function arguments in lower radix [1]. Due to the mathematical properties of Galois fields this produces mathematical forms partially “enclosed” within a lower radix domain. For example, for quaternary functions of all binary arguments the product part consists of all GF(2) multiplications, while all operations of addition are still quaternary.

The reason for such unusual mix of radices is as follows. The efficient mapping from mathematical equations into a gate level netlist becomes a significant problem since concrete gate level implementations of Galois field arithmetic components in different radices, encodings and trade-offs between balancing and power costs have different merits and demerits [30]. For example, efficient for data transfer higher radix signals may introduce considerable overhead in the corresponding logic implementation. Hence it appears impossible to find a globally optimal choice for the radix with respect to security context, and a possible solution to the problem is to combine arithmetic over GF(2) and GF(4) within a scope of one expansion to uncover an area for further optimisations.

Considering actual physical properties of the arithmetic components the synthesis tool can arrange different encodings to produce maximum efficiency. In order to maximally use the proposed potential for optimisation the tool must know the real (or estimated) physical properties of arithmetic components. Consequently the library should be provided explicitly. During the search through all possible polarities and argument radices the tool maps the synthesised equations into physical cells from the library and estimates the total power and area costs of the circuit using them as optimisation criteria.

Current version of the tool supports the following features:

- Uniform radix RM expansions in binary, ternary and quaternary radices. Fixed polarity expansions are used; the tool searches for the optimal solution through all fixed polarity forms.
- Mixed radix approach as quaternary expansions of binary arguments (binary-to-quaternary) and quaternary expansions of mixed radix arguments (mixed-to-quaternary) with optional automated search for the best combination of argument radices.
- Minimisation and preliminary estimation of actual physical properties of the synthesised circuit. If the input specifies more than one function, cross-function optimisation is applied, i.e. commonly used terms and subterms are shared between functions.
- Mapping arithmetic operations into library components. Binary signals are replaced with dual-rail buses, quaternary signals – with 1-of-4.
- Automatic conversion of the input and output port radices to match global (system) radix. The latter feature is more important for mixed-to-quaternary expansions. Additional power and area costs of radix conversion logic are also considered during the optimisation process.

This is a cross-platform command-line tool. It requires Java

runtime environment (JRE) version 1.5 or higher.

Supported input and output formats are listed below:

- Input of the tool is a list of truth vectors in a text format.
- Library of components should be specified in Verilog format. Each arithmetic component is defined as a separate module and built using generic cells or runtime library components. In case if the library is implemented using technology independent cells the tool is unable to estimate power and area of the circuit, so the only possible optimisation criterion is the number of switching wires.
- Output of the tool is a Verilog file containing the synthesised circuit as a single module.

V. WORKFLOW EXAMPLE

We have chosen DES as a classic example of cryptographic algorithm since it is simple enough to fit into a short discussion. It was processed through the design flow illustrated in Figure 2. Basic commands are listed in Algorithm 1.

The first step of the design process is the synthesis of S-boxes using Reed-Muller expansions. Specified command line options mean that the synthesised modules have a 1-of-4 encoded quaternary external interface (*-iq*), a mixed radix approach is applied in a form of quaternary function of binary arguments [1] (*-rb*), and the applied optimisation minimises wire switching (*-ow*). A library of power-balanced components is provided in the file `gflib_relaxed_generic.v`. Time taken to synthesise all S-boxes is 2.24s.

After the modules have been presynthesised, Synopsys design compiler is used to elaborate all Verilog files into a single gate-level netlist. An important option at this stage is that we use exact technology mapping (*-exact_map*), otherwise compiler would reduce redundant logic paths used for power-balancing.

The next step is to process the design in a sequence of TiDE tools. Finally, in order to enforce spacers in the circuit, the UNIX stream editor (*sed*) replaces all occurrences of D flip-flops (HDDFFPQ1) with instances of the register SpacerDFF illustrated in Figure 4. When the Enable signal from the

Algorithm 1 DES design flow commands

```

java rmmixed.ReedMullerCalculator -iq -rb -ow -lib
gflib_relaxed_generic.v -in des/des1_q -out des1.v
...
java rmmixed.ReedMullerCalculator -iq -rb -ow -lib
gflib_relaxed_generic.v -in des/des8_q -out des8.v

analyze -format verilog gflib_relaxed_generic.v
des1.v ... des8.v descore.v
elaborate descore
compile -exact_map
write -format verilog -hierarchy -output
descore_syn.v

htcomp -input des_control.ht
htmap -input des_control.hcl des
htlink -input descore_syn.v -top des_htmap.v des
htpost -input des_htlink.v des
sed 's/HDDFFPQ1/SpacerDFF/' des_htpost.v >
des_htpost_sp.v

```

Figure 4. Simple spacer injecting register

handshake control unit is high, a spacer is generated; when Enable is low data is passed. The cycles are also asymmetric, so the spacer is generated for only a small percentage of the cycle (20 – 30%). The addition of 20ps delay gate at the front of the register is trivial and does not affect the timing constraints. A big disadvantage of this approach is that flip-flops never get reset, i.e. they never capture the spacer value. Consequently, although they give balancing to the logic, they are not power-balanced themselves. However, compromised balancing can be accepted in order to reduce power consumption as long as the number of memory cells is significantly smaller than the number of logic gates. The number of replaced memory cells in our example is 5.3% of the total number of gates.

The final file `des_htpost_sp.v` contains an asynchronous power balanced m-of-n encoded circuit. Layout has not been applied in this example; this is a subject for future work.

VI. CONCLUSIONS

M-of-n codes in asynchronous circuits have been considered as an efficient countermeasure to side-channel attacks, particularly to power analysis. In scope of these techniques a number of existing design environments and synthesis tools have been analysed for their capabilities.

Logic synthesis based on mixed radix Reed-Muller expansions over Galois field arithmetic is stated to be efficient for optimising power consumption in power balanced circuits. Previously this method has never been used in a practically applied design flow. TiDE design environment has been proposed as a toolkit for the control path synthesis. It easily fits into the customised design flow, however has no direct support for the spacer protocol. This problem has been solved using power-efficient spacer injecting registers.

The proposed flow has been applied in a typical example of a cryptographic algorithm (DES), and has been recognised to be easy to follow and acceptable for the industrial use.

Future work is to finalise the design example using the Pulsic layout tool and to produce more testing and benchmark results comparing them to other works in this area.

Acknowledgement: This work is supported by EPSRC GR/F016786/1.

REFERENCES

- [1] A. Rafiev, J. P. Murphy, and A. Yakovlev, "Quaternary Reed-Muller expansions of mixed radix arguments in cryptographic circuits," in *Proc. 39th International Symposium on MVL*, 2009.
- [2] www.handshakesolutions.com.
- [3] M. Renaudin and F. Bouesse, "High security smartcards," in *Proc. the Design, Automation and Test in Europe Conference and Exhibition (DATE'04)*, 2004.
- [4] A. Bystrov, D. Sokolov, A. Yakovlev, and A. Koelmans, "Balancing power signature in secure systems," in *Proc. 14th UK Asynchronous Forum*, 2003.
- [5] UK Patent No. 0719455.8, "Cryptographic processing and processors." Newcastle University, Oct 2008.
- [6] S. Moore, R. Anderson, P. Cunningham, R. Mullins, and G. Taylor, "Improving smart card security using self-timed circuits," *Proc. of Asynchronous Circuits and Systems*, pp. 211–218, 2002.
- [7] D. A. Edwards and A. Bardsley, "Balsa: An asynchronous hardware synthesis language," *The Computer Journal*, vol. 45 (1), pp. 12–18, Jan 2002.
- [8] P. Kocher, J. Jaffe, and B. Jun, "Introduction to differential power analysis and related attacks," 1998.
- [9] J.-S. Coron and L. Goubin, "On Boolean and arithmetic masking against differential power analysis," *Lecture Notes in Computer Science*, vol. 1965, pp. 231–??, 2001.
- [10] J. Pan, J. den Hartog, and J. Lu, "You cannot hide behind the mask: Power analysis on a provably secure s-box implementation," in *Proc. the 10th International Workshop on Information Security Applications (WISA2009)*, 2009.
- [11] D. Sokolov, J. Murphy, A. Bystrov, and A. Yakovlev, "Design and analysis of dual-rail circuits for security applications," *IEEE Trans. Comput.*, vol. 54, no. 4, pp. 449–460, 2005.
- [12] K. J. Kulikowski, V. Venkataraman, Z. Wang, and A. Taubin, "Power balanced gates insensitive to routing capacitance mismatch," in *Proc. the Design, Automation and Test in Europe Conference and Exhibition (DATE'08)*, pp. 1280–1285, 2008.
- [13] K. Gandolfi, D. Naccache, C. Paar, K. G. C. Mourtel, and F. Olivier, "Electromagnetic analysis: Concrete results," 2001.
- [14] P. C. Kocher, "Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems," pp. 104–113, Springer-Verlag, 1996.
- [15] E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," *Lecture Notes in Computer Science*, vol. 1294, pp. 513–??, 1997.
- [16] W. Bainbridge, W. Toms, D. Edwards, and S. Furber, "Delay-insensitive, point-to-point interconnect using m-of-n codes," in *Proc. of ASYNC'03*, 2003.
- [17] A. Rafiev, J. Murphy, D. Sokolov, and A. Yakovlev, "Conversion driven design of binary to mixed radix circuits," in *Proc. ICCD*, 2008.
- [18] J. Cortadella, A. Kondratyev, L. Lavagno, and C. P. Sotiriou, "Desynchronization: Synthesis of asynchronous circuits from synchronous specifications," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 25, no. 10, pp. 1904–1921, 2006.
- [19] E. Tuncer, J. Cortadella, and L. Lavagno, "Enabling adaptability through elastic clocks," in *Proc. 46th Design Automation Conference (DAC)*, July 2009.
- [20] J. Sparso and S. Furber, eds., *Principles of asynchronous circuit design - A systems Perspective*. Kluwer Academic Publishers, December 2001.
- [21] J. Teifel, "Asynchronous cryptographic hardware design," in *Proc. to Carnahan Conferences Security Technology*, 2006.
- [22] www.pulsic.com.
- [23] B. Falkowski and S. Rahardja, "Efficient computation of quaternary fixed polarity Reed-Muller expansions," *Computers and Digital Techniques, IEE Proc.*, vol. 142, pp. 345–352, 1995.
- [24] B. J. Falkowski and C. C. Lozano, "Quaternary fixed-polarity Reed-Muller expansion computation through operations on disjoint cubes and its comparison with other methods," *Computers & Electrical Engineering*, vol. 31, pp. 112–131, 2005.
- [25] D. Green, "Reed-Muller expansions with fixed and mixed polarities over GF(4)," in *IEE Proc., Part E*, vol. 137, 1990.
- [26] D. Jankovic and R. S. Stankovic, "Efficient calculation of fixed-polarity polynomial expressions for multiple-valued logic functions," in *Proc. of ISMVL '02*, p. 76, IEEE Comp. Soc., 2002.
- [27] D. Jankovic, R. S. Stankovic, and C. Moraga, "Optimization of GF(4) expressions using the extended dual polarity property," in *Proc. of ISMVL '03*, p. 50, IEEE Comp. Soc., 2003.
- [28] S. Rahardja and B. Falkowski, "Efficient algorithm to calculate Reed-Muller expansions over GF(4)," *Circuits, Devices and Systems, IEE Proc.*, vol. 148, pp. 289–295, 2001.
- [29] D. Green and I. Taylor, "Multiple-valued switching circuit design by means of generalized Reed-Muller expansions," *Digital Processes*, vol. 2, pp. 63–81, 1976.
- [30] A. Rafiev, J. Murphy, and A. Yakovlev, "RTL implementations of GF(2) and GF(4) arithmetic components," tech. rep., Newcastle University, 2008.