Conversion Driven Design of Binary to Mixed Radix Circuits

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

***Abstract*—A conversion driven design approach is described. It takes the outputs of mature and time-proven EDA synthesis tools to generate mixed radix datapath circuits in an endeavour to investigate the added relative advantages or disadvantages. An algorithm underpinning the approach is presented and formally described together with m-of-n encoded gate-level implementations. The application is found in a wide variety and overlapping areas of circuit design, here a subset are analysed where the method finds the strongest application: arithmetic circuits and hardware security. The obtained results are reported showing an increase in power consumption but with considerable improvement in resistance to differential power analysis (DPA).**

I. Introduction

Single-rail circuits, traditionally used and adopted in conventional EDA flows, have a number of drawbacks with respect to security applications, asynchronous design and network-on-chip communication. These types of circuits have no power balancing, no completion detection and prone to hazards; *m-of-n codes* are known and often cited as a solution [1].

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 is used to separate data using dummy signals (spacers) and called *return-to-zero* (RTZ) or *spacer protocol*. M-of-n codes with RTZ protocol have data independent (balanced) switching of wires. Circuits based on m-of-n codes, typically 1-of-4 or 1-of-2, over the years have been used in a number of areas of electronics. Some specific examples but certainly not exhaustive include: network-on-chip [2], FPGA fabric [3], low power circuits [4], security based circuits [5] and clockless circuits.

1-of-2 (dual-rail) is widely used due to its simple theory and component implementation. However, we can observe 1-of-4 has a halved switching factor compared to that of 1-of-2, which makes it highly desirable if the goal is to minimise switching power, variability[1] (e.g. cross talk) and at the wire-level to have a constant power consumption. For security based circuits, this means the benefit of constant power consumption will be still present but lowered. The encoding of binary data in dual-rail and 1-of-4 is shown in Table I.

[1]Submicron effects, without indepth silicon experimentation this added advantage cannot be validated, however it is an often cited benefit of radix based circuits [6] and this work has been conducted under this presumption.

TABLE I
DUAL-RAIL AND 1-OF-4 ENCODINGS

multi-valued	single-rail (binary)	dual-rail	1-of-4
0	00	01 01	0001
1	01	01 10	0010
2	10	10 01	0100
3	11	10 10	1000
NULL	spacer (NULL)	00 00	0000

Since the synthesis of 1-of-4 circuits is based on multi-valued logic (MVL) synthesis, it is a rather complex task with little tool support. A number of forward-thinking efforts dedicated to the MVL synthesis have been made in recent years, in particular [7] and [8], but an effective methodology for MVL design is still an open challenge.

Prior to our work a review of the literature revealed a lack of a straightforward means or design flow to construct MVL circuits; on this premise this work was initiated. Our justification and reasoning for the research stems from the following facts:

- Moving away from the RTL design flow is frequently frowned upon by industry;
- Existing EDA tools are mature, known and time-proven;
- MVL synthesis methods employ computationally expensive algorithms instead of reusing the computational power of existing tools.

Having recognised a novel property of binary datapath circuits to facilitate conversion to a mixed radix circuit using a mixture of 1-of-4 and dual-rail, we now suggest a conversion driven design (CDD) approach. The goal of this approach is to achieve a tight integration with the conventional EDA flows with low algorithmic complexity.

This article presents the first steps and investigation of a CDD. Section II gives the background to conversion of a given binary circuit. Sections III and IV introduce the conversion method and supporting gate level components. The method implies an algorithm formalised and implemented in Section V. The final sections report results on an applicable set of security based benchmarks and suggest further optimisations.

II. Conversion basics

The problem addressed in this article can be characterised as follows. The original single-rail datapath is given as a structural HDL netlist; where datapath is defined as logic gates without registers or combinational loops. The goal of the conversion is to produce an equivalent higher radix circuit.

978-1-4244-2658-4/08/$25.00 ©2008 IEEE

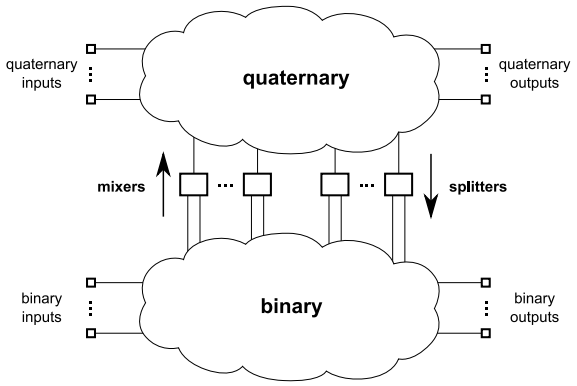


Fig. 1. Mixed encoding in a converted combinational logic circuit.

Since dual-rail is based on binary computations, there is a transparent correspondence between initial specification and the resultant circuit. Conversion to dual-rail can be performed using direct mapping of single-rail gates to dual-rail counterparts [9]. The method was implemented earlier in a set of software tools which interface to conventional EDA tools and form a coherent design flow [10].

Conversion from binary to multi-valued logic can employ grouping of data signals. However, a grouping of all signals in the circuit is not globally efficient, because the original structure usually causes restructuring and splitting of higher radix data. This leads to the use of mixed radix encoding, which means that the circuit becomes partially binary and partially multi-valued (heterogeneous) [8].

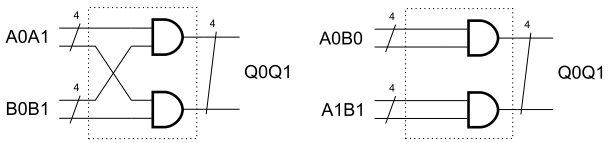
In terms of CDD, quaternary logic is of more interest compared to other types of multi-valued logic due to the simplicity of signal grouping by two bits. A result of the conversion is shown in Figure 1 and consists of binary and quaternary blocks connected through a row of signal converters: *splitters* and *mixers*. A splitter is an element which divides quaternary signal into two binary ones. A mixer is an element which merges two binary signals into one quaternary.

A generic outline for the proposed conversion technology can be described as follows. The algorithm starts by "transferring" gates from the binary block to the quaternary block by grouping them into pairs. During this phase the conversion uses technology independent (abstract) binary and quaternary components thus adding a component level of abstraction to the design flow. After all possible grouping is done the circuit can be mapped into a gate-level netlist replacing components with real cells using specific encoding and library.

The way the gates are grouped determines the efficiency of the conversion, therefore the conversion problem corresponds directly to the gate grouping problem described in the following sections.

III. Types of Gate Grouping

For $2n$-bit binary circuits there is an intuition to group higher and lower bits of each signal pair, as shown in Figure 2(a), to form a n-signal quaternary circuit. Certain gates which violate bitwise regularity of the original circuit will remain ungrouped forming a binary part of the resultant mixed radix circuit.



(a) bitwise (b) operandwise

Fig. 2. Types of gate groupings (considering the same original circuit).

TABLE II
Bitwise and operandwise quaternary operations example

x	y	bitwise AND	operandwise AND-AND
0	0	$0 \wedge 0 = 0$	$\langle 0 \wedge 0, 0 \wedge 0 \rangle_4 = \langle 0, 0 \rangle_4 = 0$
0	1	$0 \wedge 1 = 0$	$\langle 0 \wedge 0, 0 \wedge 1 \rangle_4 = \langle 0, 0 \rangle_4 = 0$
0	2	$0 \wedge 2 = 0$	$\langle 0 \wedge 0, 1 \wedge 0 \rangle_4 = \langle 0, 0 \rangle_4 = 0$
0	3	$0 \wedge 3 = 0$	$\langle 0 \wedge 0, 1 \wedge 1 \rangle_4 = \langle 0, 1 \rangle_4 = 1$
1	0	$1 \wedge 0 = 0$	$\langle 0 \wedge 1, 0 \wedge 0 \rangle_4 = \langle 0, 0 \rangle_4 = 0$
1	1	$1 \wedge 1 = 1$	$\langle 0 \wedge 1, 0 \wedge 1 \rangle_4 = \langle 0, 0 \rangle_4 = 0$
1	2	$1 \wedge 2 = 0$	$\langle 0 \wedge 1, 1 \wedge 0 \rangle_4 = \langle 0, 0 \rangle_4 = 0$
1	3	$1 \wedge 3 = 1$	$\langle 0 \wedge 1, 1 \wedge 1 \rangle_4 = \langle 0, 1 \rangle_4 = 1$
...	...	...	...
3	0	$3 \wedge 0 = 0$	$\langle 1 \wedge 1, 0 \wedge 0 \rangle_4 = \langle 1, 0 \rangle_4 = 2$
3	1	$3 \wedge 1 = 1$	$\langle 1 \wedge 1, 0 \wedge 1 \rangle_4 = \langle 1, 0 \rangle_4 = 2$
3	2	$3 \wedge 2 = 2$	$\langle 1 \wedge 1, 1 \wedge 0 \rangle_4 = \langle 1, 0 \rangle_4 = 2$
3	3	$3 \wedge 3 = 3$	$\langle 1 \wedge 1, 1 \wedge 1 \rangle_4 = \langle 1, 1 \rangle_4 = 3$

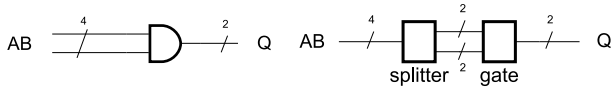
Although the intuition behind the bitwise approach is straight-forward, our investigation [11] revealed that automatically distinguishing even and odd bit parts of the given netlist is computationally complex or, in certain cases, infeasible. For example, S-box circuits [12], [13] tend to reshuffle input data, thus input signals have no bitwise meaning. Natural for the synthesis from functional specification, bitwise grouping is not suitable for structural netlist transformations, especially when the original structure requires data "shifting" between bitwise parts of the circuit. This necessitates finding a more efficient way of gate grouping with respect to the CDD approach.

Let's assume that the original circuit consists of two-input standard cells (AND, OR, XOR gates) and inverters. Considering this, it is possible to group inputs of any gate into one quaternary signal. Outputs of a given pair of gates can also be grouped into a quaternary signal. This produces an *operandwise grouping* of gates illustrated in Figure 2(b).

Due to the nature of operandwise grouping, any quaternary signal x can be rewritten as a pair of its original signals, $x = \langle s_1, s_0 \rangle_4$. For two quaternary signals $x = \langle s_1, s_0 \rangle_4$ and $y = \langle t_1, t_0 \rangle_4$, two binary functions A and B can form a quaternary operandwise operation $\langle \mathrm{A}, \mathrm{B} \rangle_4$ shown in (1) and clarified in Table II.

$$Q_{\mathrm{AB}}(x, y) = \langle \mathrm{A}(s_1, s_0), \mathrm{B}(t_1, t_0) \rangle_4 \qquad (1)$$

In a case when the output of a quaternary gate has to



(a) Q/B gate as an incomplete operandwise group

(b) structural decomposition

Fig. 3. Understanding Q/B gates.

be split again into binary signals an insertion of a splitter can be avoided using incomplete operandwise grouping, i.e. a grouping when gate inputs are grouped, but the output remains binary as shown in Figure 3(a). A *Q/B gate* is a gate with a quaternary input and a binary output. Due to their semantical meaning clarified in Figure 3(b), Q/B gates can eliminate unnecessary splitters during the conversion.

IV. Component implementation

Moving from abstract binary and quaternary signals to specific encodings, dual-rail and 1-of-4 correspondingly, CDD requires proper definition of mixed radix components.

The resultant circuit should be consistent with RTZ protocol, i.e. its components should satisfy the *spacer condition*, which means that the output of a component must go to a spacer value (NULL) if any of arguments have spacer value.

From (1) for each wire q_i of 1-of-4 encoded $Q_{AB} = \{q_3, \ldots, q_0\}$ w.r.t. spacer condition we have:

$$q_i = \begin{cases} 0, & \text{if } x = \text{NULL or } y = \text{NULL} \\ \mathrm{A}^{<i_1>}(x) \cdot \mathrm{B}^{<i_0>}(y), & \text{otherwise} \end{cases} \tag{2}$$

Here i_0 and i_1 are the bits of binary representation of integer i, in other words $i_{10} = \langle i_1, i_0 \rangle_2$. Notation $\mathrm{F}^{<k>}(x)$ stands for polarity representation of a binary function $\mathrm{F}(s_1, s_0)$ with quaternary x representing binary arguments s_1, s_0, i.e. $\mathrm{F}^{<k>}(x) = \left\{\overline{\mathrm{F}}(s_1, s_0), \mathrm{F}(s_1, s_0)\right\}$ for $k = \{0, 1\}$ correspondingly.

Let

$$f_{\mathrm{F}}^{<k>}(x) = \begin{cases} 0, & \text{if } x = \text{NULL} \\ \mathrm{F}^{<k>}(x), & \text{otherwise} \end{cases} \tag{3}$$

then (2) becomes:

$$q_i = f_{\mathrm{A}}^{<i_1>}(x) \cdot f_{\mathrm{B}}^{<i_0>}(y) \tag{4}$$

According to Table I, binary signals s_1, s_0 can be expressed using certain wires $x_3, \ldots, x_0$ of 1-of-4 encoded quaternary $x = \langle s_0, s_1 \rangle_4$ as follows:

$$\begin{array}{ll} s_0 = x_3 + x_1 & \overline{s}_0 = x_2 + x_0 \\ s_1 = x_3 + x_2 & \overline{s}_1 = x_1 + x_0 \end{array} \tag{5}$$

Applying (5) and the spacer condition (w.r.t. 1-of-4 code) to (3) we have:

$$f_{\mathrm{F}}^{<k>}(x_3, \ldots, x_0) = \mathrm{F}^{<k>}(x_3, \ldots, x_0)(x_3 + \ldots + x_0)$$

Total set of functions $f_{\mathrm{F}}^{<k>}(x)$ for the standard functions $\mathrm{F} = \{\mathrm{AND}, \mathrm{OR}, \mathrm{XOR}\}$:

$$\begin{array}{ll} f_{\mathrm{AND}}^{<0>}(x) = x_0 + x_1 + x_2 & f_{\mathrm{AND}}^{<1>}(x) = x_3 \\ f_{\mathrm{OR}}^{<0>}(x) = x_0 & Typesof f_{\mathrm{OR}}^{<1>}(x) = x_1 + x_2 + x_3 \\ f_{\mathrm{XOR}}^{<0>}(x) = x_0 + x_3 & f_{\mathrm{XOR}}^{<1>}(x) = x_1 + x_2 \end{array} \tag{6}$$

For security application the circuit components should have balanced switching to guarantee data independent power consumption. This can be done using identical gates with certain inputs connected to the ground. Thus $f_{\mathrm{F}}^{<k>}(x)$ should be symmetric with regard to k, and representations $f_{\mathrm{OR}}^{<0>}(x) = x_0 + 0 + 0$ and $f_{\mathrm{AND}}^{<1>}(x) = x_3 + 0 + 0$ should be used instead.

Example 1: From (4) and (6), considering balanced switching, one can derive a set of equations (7) defining an operandwise 1-of-4 AND-XOR group.

$$\begin{array}{rcl} q_0 & = & (x_0 + x_1 + x_2)(y_0 + y_3) \\ q_1 & = & (x_0 + x_1 + x_2)(y_1 + y_2) \\ q_2 & = & (x_3 + 0 + 0)(y_0 + y_3) \\ q_3 & = & (x_3 + 0 + 0)(y_1 + y_2) \end{array} \tag{7}$$

If the design flow requires negative logic decomposition, NOR expansion of (7) is also possible:

$$\begin{array}{rcl} q_0 & = & \neg(\neg(x_0 + x_1 + x_2) + \neg(y_0 + y_3)) \\ q_1 & = & \neg(\neg(x_0 + x_1 + x_2) + \neg(y_1 + y_2)) \\ q_2 & = & \neg(\neg(x_3 + 0 + 0) + \neg(y_0 + y_3)) \\ q_3 & = & \neg(\neg(x_3 + 0 + 0) + \neg(y_1 + y_2)) \end{array} \tag{8}$$

A *Q/B gate* of a function F can be implemented as (9) where $f_{\mathrm{F}}^{<k>}(x)$ is an equation from (6) and x is 1-of-4 argument.

$$\begin{array}{rcl} q_0 & = & f_{\mathrm{F}}^{<0>}(x) \\ q_1 & = & f_{\mathrm{F}}^{<1>}(x) \end{array} \tag{9}$$

Equations (4) and (9) in their turn imply a function for a mixer component (10) for two dual-rail signals $a = \{a_1, a_0\}$ and $b = \{b_1, b_0\}$.

$$q_i = a_{i_1} \cdot b_{i_0} \tag{10}$$

Mixers and *splitters* have a rather straight-forward positive logic implementations derived from (10) and (5) respectively. The splitter is the only component having two switching output wires.

Dual-rail gates also use positive logic decomposition. Negative logic optimisations [10] potentially can be performed, but require more sophisticated analysis of the circuit structure and are not considered within this article.

It is known that *inversion* in dual-rail can be done using "wire-crossing" [10], and inverters do not affect complexity of the conversion result. Inversion in 1-of-4 is similar, but mutual independence of operandwise signals requires separate inversions for each of them. Half inversions, when only one bit of a pair is inverted, should be defined as shown in Table III.

TABLE III
INVERSION IN 1-OF-4

	full inversion	half inversion	
		lower bit	higher bit
$x_0 =$	x_3	x_1	x_2
$x_1 =$	x_2	x_0	x_3
$x_2 =$	x_1	x_3	x_0
$x_3 =$	x_0	x_2	x_1

As can be seen from the specification, described mixed radix components exhibit early propagation [14], i.e. are weakly indicating. Since m-of-n codes imply input complete gates, a completion detection mechanism is required [15], [16]. However, completion detection in heterogeneous circuits is a different issue. We restrict ourselves with early propagation gates leaving other forms of speed independent solutions outside the scope of the article.

The full set of library items contains all possible 1-of-4 operandwise groupings (nine components), dual-rail and Q/B gates for basic binary functions and signal converters.

V. CONVERSION ALGORITHM

The operandwise grouping suggests a binary trees approach considering gates of the circuit to be tree-nodes and their inputs to be child branches. As it was mentioned before, the given datapath circuit contains no loops. However, a pure tree-like structure can be blocked by gates with multiple fanout. The tree size can be reduced by recursive operandwise grouping of child nodes for each gate in binary trees within the circuit. This grouping causes all signals in tree-like structures to become 1-of-4 encoded, but "blocked" parts of the circuit remain binary and go to dual-rail.

Formally, a circuit is considered to be a set of entities E; each entity has a type of *input*, *output*, or *gate*. Input and output entities are circuit ports. Each non-input entity $e \in E$ has a preset $I(e) = \{i_0(e), \ldots, i_{n-1}(e)\}$, i.e. entities connected to the inputs of e. Due to declared constraints $n = 2$ for gates, and $n = 1$ for outputs. Set P is a set of gate groups (pairs). It represents 1-of-4 encoded part of the circuit. In addition there is a parameter θ_e that stands for encoding of entity $e \in E$ and can be either dual-rail or 1-of-4. It is used for automated insertion of signal converters (splitters and mixers) into the final circuit.

The proposed algorithm contains three phases as shown in Algorithm 1. To avoid recursion the grouping is done in two search passes through E (phase 1 and 2). The first phase ignores gate fanouts and groups all signals considering the whole circuit as a binary tree. This leads to duplication of certain gates. The second phase analyses the duplicates and discards the groups which lead to duplication. The last phase reconstructs correctness of links between gates by inserting signal converters where appropriate. Splitters are reduced to Q/B gates.

There is no simple solution to estimate optimal grouping of outputs but to search through all possibilities. However, due to the nature of binary trees approach the grouping of outputs has minor influence to the structure comparing with the whole circuit. Therefore a suggestion to group any pair of outputs is accepted.

Algorithm 1 Conversion based on binary trees

```
// Phase 1: group all gate inputs.
    for each gate g in E:
      if i0(g) and i1(g) are of the same type:
         group {i0(g), i1(g)} and add to P
    end for
// Phase 2: discard groups containing gates with fanout > 1.
    for each non-output entity e in E:
      u = how many groups share e
      if u > 1: remove groups containing e
      else if u = 1: set θe to 1-of-4
      else set θe to dual-rail
      end if
      if e is gate and group {i0(g), i1(g)} ∉ P:
          remove all groups containing i0(g) or i1(g)
    end for
// Phase 3: insert mixers and Q/B gates.
    for each gate g in E:
      if θi(g) is 1-of-4 and θg is dual-rail:
         set type of g to Q/B gate
      else if θi(g) is dual-rail and θg is 1-of-4:
         insert mixer before g
    end for
```

The algorithm exploits structural isomorphism of operandwise grouping, i.e. the resultant circuit can be converted back to the original by substituting gate groups with Figure 2(b) and Q/B-gates with Figure 3(a). Thus it is correct by construction.

The computational complexity of the algorithm is linear, $O(3N)$, where N is the size of E. The algorithm is highly modular; one can add more passes to the algorithm to increase efficiency of the conversion. However it can produce significant "fractioning" of dual-rail and 1-of-4 parts of the circuit increasing the number of signal converters required.

Example 2: Consider a 2-bit adder shown in Figure 4. Preset gates of the gate g40 can form the group {g30, g10}; similarly ports A0 and B0 are grouped as inputs to g00 or g10. Considering all gates we can make the following grouping: {A0, B0}, {A1, B1}, {g30, g10}, {g31, g11}, {g40, g01}. Preset of gates g20 and g30 cannot be grouped because of the different entity types (input C cannot be grouped with the gate g00). The second phase should cancel signal duplicating gate groups, but in this case nothing is to be cancelled. Indeed, in spite of the fact that some gates and ports have fanout > 1, they are not shared between different groups and do not lead to signal duplication. Finally, randomly selected output grouping {Q0, Q1} leads to the grouping {g20, g21}. Resultant circuit is shown in Figure 5. Gates g00 and g41 are Q/B gates.

The described algorithm was implemented as a plug-in for the Workcraft framework [17]. This tool is dedicated to asynchronous design and has a support for gate level

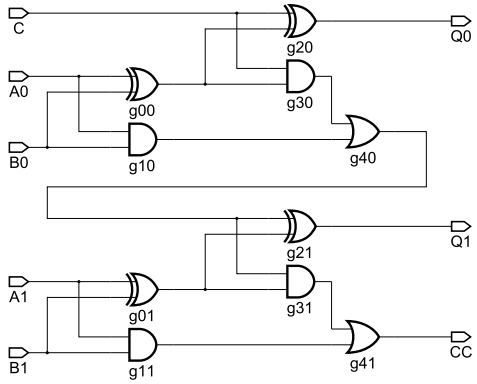


Fig. 4. Example original single-rail circuit: 2-bit adder.

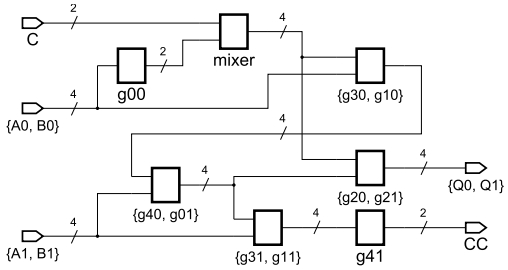


Fig. 5. 2-bit adder converted using binary trees approach; components are shown as "black boxes".

modelling including reading and writing of structural Verilog files.

VI. Benchmark results

To date as far as the authors are aware no security based circuits have been proposed which use an encoding other than dual-rail, thus making them the logical choice for benchmarks. A number of arithmetic based examples were added to give a wider range for comparison. A component implementation is still a subject of improvement, therefore these tests were made for the purpose of estimation of the approach, and they do not represent final results. Since during a test we compare two circuits designed using the same library, the choice of the library doesn't make a big difference. AMS 0.35μm library was applied because of the availability of technical information.

a) Verilog simulation: Converted circuits were compared with pure dual-rail equivalents using gate level Verilog simulation in synchronous mode. Initial circuits were mapped using complex gates as specified in (7). The results are shown in Table IV. Switching activity was measured per RTZ protocol period. In order to discover data dependent variations as a result of imperfect balancing the switching energy was calculated separately for each gate output as a sum of documented values; a standard deviation of switching energy values was calculated for each benchmark circuit.

The 16-bit full adder was ideally converted (78 of 80 gates are grouped into 1-of-4) but still has only 12% power savings due to the implementation of 1-of-4 gates. From (4), (9) and (10) one can conclude that in fact any operandwise quaternary group consists of two Q/B gates and a mixer, and there are always two switching wires inside a 1-of-4 gate. On the other hand, Kasumi S-boxes [13] have more switching activity in 1-of-4 than in dual-rail, but consume approximately the same power. The reason is that the AMS implementation of 1-of-4 XOR-XOR component uses OA22 complex gate, when dual-rail XOR uses AO22 which consumes more power.

Original single-rail circuits for 4-bit multiplier and two S-boxes were synthesised using the complete design flow from Synopsys toolkit; in particular area and power optimisations were applied leading to negative logic decomposition. The results show these optimisations cause worse conversion results and should be taken into account when developing a design flow.

However, the switching energy of gates has minor impact on the total energy consumption of the circuits, thus the number of switching wires is of greater importance. Consequently the most significant reason oTypes off extra power cost is a large number of mixers in the converted circuits that in certain cases exceeds the number of quaternary gates.

b) SPICE simulation: AES S-box [12] implementations in single-rail, dual-rail and mixed radix were simulated in Synopsys. NOR decompositions (8) were used for mixed radix components. During the test the circuits were fed with random data inputs producing variations in the supply current over time shown in Figure 6. Similar shapes of the current peaks on the graphs show that dual-rail and mixed radix circuits are power balanced unlike the single-rail one. An average of correlations between power signatures of different input signals was used as a numeric estimation parameter. The numbers show that both dual-rail and mixed radix solutions have significant protection parameters, but the mixed radix example has rather negligible advantage.

VII. Conclusions

The use of existing EDA tools for higher radix and heterogeneous circuits design was suggested. A method of conversion was described. For the proposed algorithm and developed library a number of benchmarks were tested. The results have shown a number of advantages and disadvantages with respect to different optimisation criteria and depending on types of circuits. The tests revealed power and area overhead. However, an improvement in power balancing and a flexible tradeoff between power consumption and balancing makes a considerable advance for security applications.

The use of mixed radix contributes advantageously to security due to its intrinsic diffusion of binary values between signals representing a mixed radix value. A data signal passing through the mixed radix circuit is "unpredictably" reformed due to the mixing with other signals and splitting again for multiple times. Therefore a tracing of the initial data

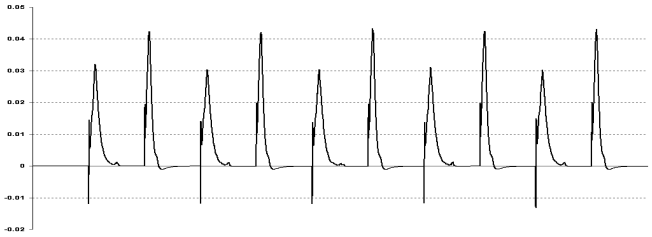
TABLE IV
CONVERSION RESULTS IN COMPARISON WITH PURE DUAL-RAIL CIRCUITS

Circuit	Dual-rail				1-of-4 and dual-rail mixed				
	Gates	Switching	Switching energy, pJ		Gates	Switching	Switching energy, pJ		
	binary	wires	ave.	std. dev.	q-ary; binary; Q/B; conv.	wires	ave.	std. dev.	save, %
2-bit adder	10	20	11.13	0.2661	4; 0; 2; 1	14	10.66	0.0000	4.3
16-bit ripple carry adder	80	160	86.31	0.9803	39; 0; 2; 1	84	75.72	0.0000	12.3
4-bit multiplier*	28	56	30.08	0.5247	10; 4; 4; 11	58	38.72	0.1074	–28.7
Kasumi S-box 7	125	250	139.13	3.4908	39; 44; 4; 45	264	144.98	0.6993	–4.2
Kasumi S-box 9	128	256	146.51	3.7947	34; 59; 1; 38	264	137.12	1.2070	6.4
Kasumi S-box 9*	150	300	169.56	1.3448	44; 53; 9; 57	326	187.96	0.7809	–10.8
AES S-box*	797	1594	818.56	1.1914	252; 274; 19; 275	1640	1116.03	0.4870	–36.3

* original single-rail circuits were optimised in Synopsys.

(a) single-rail circuit

(b) dual-rail circuit, average correlation: 0.997



(c) mixed radix circuit, average correlation: 0.998

Fig. 6. SPICE simulation results for AES S-box.

becomes more complex, thereby increasing the resistance to DPA attacks.

The proven advantage of the CDD approach in general is the fact that conversion algorithms can have a negligible computational cost in comparison with the developing a completely new synthesis; and the approach fits within the standard EDA flow.

The future work can include transistor level component optimisations, investigation of other grouping solutions and different types of heuristics.

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

REFERENCES

[1] T. Verhoeff, "Delay-insensitive codes – an overview," *Distributed Computing*, no. 3, pp. 1–8, 1988.

[2] W. Bainbridge, W. Toms, D. Edwards, and S. Furber, "Delay-insensitive, point-to-point interconnect using m-of-n codes," in *Proceedings of the Ninth International Symposium on Asynchronous Circuits and Systems (ASYNC'03)*, 2003.

[3] J. Teifel and R. Manohar, "An asynchronous dataflow FPGA architecture," *IEEE Trans. Comput.*, vol. 53, no. 11, pp. 1376–1392, 2004.

[4] T. Takahashi and T. Hanyu, "Multiple-valued multiple-rail encoding scheme for low-power asynchronous communication," in *ISMVL '04: Proceedings of the 34th International Symposium on Multiple-Valued Logic*, pp. 20–25, 2004.

[5] S. Moore, R. Anderson, P. Cunningham, R. Mullins, and G. Taylor, "Improving smart card security using self-timed circuits," *Asynchronous Circuits and Systems, 2002. Proceedings. Eighth International Symposium on*, pp. 211–218, 8-11 April 2002.

[6] US Patent 6202194, "Method and apparatus for routing 1 of N signals."

[7] M. Gao, J.-H. Jiang, Y. Jiang, Y. Li, S. Sinha, and R. Brayton, "MVSIS," in *Notes of the International Workshop on Logic Synthesis*, June 2001.

[8] W. B. Toms, D. A. Edwards, and A. Bardsley, "Synthesising heterogeneously encoded systems," in *Proceedings of the 12th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC '06)*, 2006.

[9] A. Kondratyev and K. Lwin, "Design of asynchronous circuits using synchronous CAD tools," *IEEE Des. Test*, vol. 19, no. 4, pp. 107–117, 2002.

[10] 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.

[11] A. Rafiev, J. Murphy, D. Sokolov, and A. Yakovlev, "Investigating gate grouping algorithms for mixed radix conversion," tech. rep., Newcastle University, May 2008.

[12] *Specification for the Advanced Encryption Standard (AES)*, Nov 26, 2001. Federal Information Processing Standards Publication 197.

[13] *3GPP Technical Specification 35.202*, 2001. v3.1.1.

[14] Y. Zhou, D. Sokolov, and A. Yakovlev, "Cost-aware synthesis of asynchronous circuits based on partial acknowledgement," in *ICCAD '06: Proceedings of the 2006 IEEE/ACM international conference on Computer-aided design*, pp. 158–163, 2006.

[15] C. Seitz, "System timing," in *Introduction to VLSI Systems* (C. Mead and L. Conway, eds.), ch. 7, Addison-Wesley, 1980.

[16] C. Jeong and S. Nowick, "Block-level relaxation for timing-robust asynchronous circuits based on eager evaluation," in *Proceedings of the 14th International Symposium on Asynchronous Circuits and Systems (ASYNC'08)*, 2008.

[17] I. Poliakov, D. Sokolov, and A. Mokhov, "Workcraft: A static data flow structure editing, visualisation and analysis tool," in *ICATPN*, pp. 505–514, 2007.