Performance Metrics of a Reconfigurable Fabric

Mua’ad Abu-Faraj

1 Department of Computer Information Systems, The University of Jordan
Aqaba, 77110, Jordan

Abstract
Performance metrics of reconfigurable systems have largely focused on the relative performance of specific algorithms on the fabric compared against the same algorithm on a general-purpose processor or ASIC device. Here we introduce a number of metrics, which relate purely to the act and cost of reconfiguration, independent of implementation or algorithm. We specifically introduce the notation of total and relative gate efficiency, power, and granularity.

Keywords: Dynamic Reconfigurable Fabric (DRF), Granularity, Power, complexity, and Efficiency.

1. Introduction
A Dynamically Reconfigurable Fabric (DRF) is a hardware system capable of being reconfigured under software control dynamically. Several reconfigurable architectures have been proposed [1]-[4]. The number of application fields in which reconfigurable computing has been applied is massive, including embedded systems, network security applications, and multimedia applications [5]. Some studies have addressed analysis and exploration of analytical models of reconfigurable architectures [6]-[9].

Most performance analysis focuses on how well an algorithm performs on a reconfigurable system. In this paper a performance analysis for different reconfigurable systems is explained. In [6], a unified area model for RP-space has been studied. This model is used to estimate the peak computational density as a function of granularity and on-chip instruction store size, which is also used to characterize the computational efficiency.

The Chimaera system was evaluated with several benchmarks. In [1] , three different algorithms were used for the system analysis: Compress/SPEC92, with a speedup of 1.11; Eqntott/SPEC92, with a speedup of 1.8; and Conway’s Game of Life with a speedup of 1.34. By replacing the kernels with reconfigurable unit instructions, it is possible to get a speedup of 2.06. A speed up of 160 times can be achieved using careful hand mapping of bit parallel optimization opportunities. In [10], some applications of MediaBench [11] were evaluated: MPEG Encoder, G.721 encoder and decoder, ADPCM compression and decompression, Pegwit (public key encryption), as well as applications taken from the Honeywell benchmark: image compression and decompression. In [1], [12], DES (encryption/decryption), Simple Gaussian Blur, RGB-Scale Conversion were also tested on Chimaera.

Simulations were performed in order to gather results for Garp, since no actual hardware existed. It was compared against a Sun Ultra-SPARC 1/170, a 4-way superscalar 64-bit processor with 16 KB each of on-chip instruction and data caches [13].

REMARC executes MPEG2 decoding, optimizing two kernels: IDCT and MC. It also executes MPEG2 encoding and DES. A high-level simulation of the system demonstrated speedups ranging from a factor of 2.3 to 21.2 in the aforementioned applications [14].

The difficulty with such benchmarks is that they are algorithm dependent rather than quantifying more general aspects of the DRF. In this short paper our interest is to start a conversation about the development of more general performance metrics for DRFs, which are (for the most part) algorithm-independent. We begin by examining a simple case of a DRF by way of motivation.

2. Motivating Example
As a motivating example of a dynamically reconfigurable system, we consider the SAXPY processor we introduced in [15]. A Saxpy coprocessor can perform numerical operation:

\[ Z = \alpha x + y \]  

where \( \alpha \) is a scalar and \( x \) and \( y \) real or complex vectors, which we take here to be length-1 for simplicity. The general Saxpy coprocessor is shown in Fig.1.
The quad Saxpy coprocessor shown in Fig. 2 can be used (with N-fold replication) to perform a single N-way complex single Saxpy operation or, using the same hardware, 4N-way real single Saxpy operations.

The addition of multiplexers to the coprocessor in Fig. 2 results in the reconfigurable Saxpy coprocessor as shown in Fig. 3, which can now easily switch between 4 real SAXPY operations or 1 complex SAXPY operation.

Note that only 4 bits are needed to control the reconfiguration of this system, and that the reconfigurability is driven by relatively simple multiplexors. It is obvious that there is a loss in this system due both to the area occupied by the reconfiguration multiplexors and to the delay paths introduced by them, and it would seem reasonable that $T_{ASIC}^{SAXPY} > T_{SAXPY}^{ASIC}$ (where $T$ is total compute time for the SAXPY on an ASIC or in the DRF). On the other hand, the simple assertion that $C_{SAXPY}^{ASIC} < C_{SAXPY}^{DRF}$, where $C$ is some cost metric, is not so obvious. Certainly, if $C$ is (VLSI) area, then the assertion would hold (providing constant technology etc.). If $C$ is power, the assertion also probably holds. If $C$ measures reuse potential, then more likely $C_{SAXPY}^{DRF} < C_{SAXPY}^{ASIC}$.

What interests us here is not the computation accomplished by the reconfigurable system above, but rather the derivation of metrics which relate the cost of reconfigurability to the compute power of the system, the degree of complexity of the reconfigurable units themselves, and the difficulty of using or defining the reconfigurable system.

3. Efficiency

We have previously defined [15] an efficiency in terms of the ratio of the gate counts of the gates used in the functional part of the circuit (i.e., that part of the circuit used for computation; in the case of the example in Section 2, these would be the multipliers/adders) and the gates used to control the reconfigurability of the system (in
our example in Section 2, the multiplexors). We call this the relative efficiency $E_R$, given by:

$$E_R = \frac{N_F}{N_F + N_R}$$

(2)

where $N_F$ is the gate count associated with functional components in the DRF and $N_R$ is the gate count associated with components in the DRF whose only role is enabling reconfiguration. Note that an equivalent definition using area rather than gate count was given by [6], [16]-[17].

This definition of efficiency captures the relative cost of the reconfigurability but, in a sense, downplays the power of the reconfigurable system by ignoring the fact that without reconfiguration, multiple circuits would have to be built to accommodate the reconfigurable options. Our second definition of efficiency captures this more precisely, and is called the total efficiency $E_T$ given by:

$$E_T = 1 - \frac{N_F + N_R}{\sum N_F}$$

(3)

where $N_F^i$ is the number of functional gates needed to compute computational option $i$ and $N_F$ and $N_R$ are as given above. Thus, if functions $p$ and $q$ can be computed individually using $N_p$ and $N_Q$ gates (respectively) or as options in a reconfigurable system using $N_F + N_R$ gates (in total), then $E_T$ would be given by $1 - (N_F + N_R)/(N_F + N_Q)$. We expect:

$$\sum N_F^i > N_F + N_R$$

(4)

and thus $0 \leq E_C, E_T \leq 1$

4. Power

We define the power $P$ of a DRF as the number of distinct functions the DRF can compute. Sometimes power is trivially computed (as in the example below) and sometimes (as with an FPGA, for example) power may be less easy to determine. One particularly simple class of circuits with surprisingly wide applications are the cascade circuits comprising of $n$-input functions $f_i(x_0, x_1, ... , x_{n-1})$ cascaded in sequence and in parallel with the inputs to each successive function multiplexed from the outputs of the preceding level. Fig. 4 shows an example of such a circuit.

We refer to such circuits in general as $L^pF^qI^r$ circuits, where $p$ is the number of levels in the circuit, $q$ is the number of distinct function blocks at each level, and $r$ is the number of inputs to each block at the first level. (Note that at each stage the purely reconfiguration elements of the circuit consists of two 2-1 $n$-bit multiplexors.)

We choose this kind of circuit as an example for computing power here because it leads naturally to an interpretation of power that is not obvious at first glance; that is, we believe there are circumstances where it is appropriate to distinguish the computation $f(x=a,y=b)$ from $f(x=a,y=a)$ even when $f$ itself is not changing. (This odd way of counting arises in part due to our definition of total efficiency $E_T$). This is tantamount to arguing that $2a$, $a + b$ and $2b$ in an adder constitute different computations no matter what values are actually assigned to a and b. There are circumstances for such circuits where we assume that an input set (here $\{a,b\}$) arrives at the input to the cascade and (for the example shown) we would like the output of the first function pair to be any of possible outputs of $f$ and $g$ given the inputs (multiplexed), and equivalently thereafter throughout the cascade. Then if $f_i(x,y) = f_i(y,x)$ throughout the cascade the power of this circuit (assuming no equivalent results and counted in this admittedly controversial way) is 42, and when carried out to $n$ levels, in general by the recurrence:

$$P(n) = P(n - 1) \times (P(n - 1) + 1)$$

(5)

Power, used alone, is a weak measure of the compute power of the DRF (at least in the usual algorithmic complexity sense), since it does not convey what is being computed. Assessing the degree of complexity of the DRF relates to the individual complexity of the elements, which constitute the DRF, encapsulated by granularity.
5. Granularity

In [6], the granularity is defined as the number of resources controlled by each instruction. In this paper we define the granularity as the number of gates in a reconfigurable unit against the number in the entire system. If we call the smallest reconfigurable unit the Least Reconfigurable Unit (LRU), then we can define the relative granularity \( G_R \) as follows:

\[
G_R = \frac{N_{LRU}}{N_{SYS}}
\]

(6)

where \( N_{LRU} \) is the number of functional gates in the LRU, and \( N_{SYS} \) is the total number of gates in the system (including gates in the reconfigurable path i.e., those only used in reconfiguration). If we assume the die area of a reconfigurable system is given by \( A_D \) (in \( \text{cm}^2 \)) then the gate density \( D_G \) is:

\[
D_G = \frac{N_{SYS}}{A_D}
\]

(7)

We need a normalizing constant \( N_{MAX} \), which we take to be the maximum gate density (for a given technology) in any unit area. Then the relative gate density \( D_R \) is:

\[
D_R = \frac{D_G}{N_{MAX}}
\]

(8)

Then we define the absolute granularity \( G_A \) as:

\[
G_A = \frac{N_{LRU}}{N_{SYS}} \times \frac{N_{SYS}}{N_{MAX} \times A_D} = \frac{N_{LRU}}{N_{MAX} \times A_D}
\]

(9)

Fig. 5 shows an example of the behavior of the granularities (\( G_R \rightarrow \text{red}, G_A \rightarrow \text{blue} \) and \( G \rightarrow \text{green} \)) as \( N_{LRU} \) varies from \( 0 \) to \( N_{SYS}/2 \) for \( N_{SYS} = N_{MAX}/2 \) and \( A_D = 4 \).

6. Complexity

Complexity is a more difficult metric to capture; while granularity (above) captures complexity in the sense of gate count, it does not capture complexity in any functional sense. One approach to building a complexity metric which might also capture some sense of difficulty of design (in addition to the more usual notion of algorithmic complexity) might relate eventually to the underpinning complexity of the VHDL [see [18]] used to define the DRF LRUs, combined with the complexity of the VHDL which dynamically reconfigures the LRUs into functional systems.
6. Conclusions

In this paper we have taken some initial steps in defining some basic general metrics, which quantify DRFs in terms of reconfigurability rather than in terms of any specific algorithm. In particular, we have offered two variants of efficiency, three of granularity and one of power as an initial step in quantifying this important area.

References


Mua’ad M. Abu-Faraj received the B.Eng. degree in computer engineering from Mu’tah University, Mu’tah, Jordan, in 2004, the M.Sc. degree in computer and network engineering from Sheffield Hallam University, Sheffield, UK, in 2005, and the M.Sc. and Ph.D. degrees in computer science and engineering from the University of Connecticut, Storrs, Connecticut, USA, in 2012. He is, at present, assistant professor at the University of Jordan, Aqaba, Jordan. He is currently serving as reviewer for the IEEE Micro, IEEE Transactions on Computers, Journal of Supercomputing, and International Journal of Computers and Their Applications (IJA).

His research interests include computer architecture, reconfigurable hardware, cryptography, and wireless networking. Dr. Abu-Faraj is a member of the IEEE, ISCA (International Society of Computers and their Applications), and JEA (Jordan Engineers Association).