UF REVERSIBLE COMPUTING PROJECT MEMO #M19

Spacetime-Energy Tradeoff for Reversible Computing in the Low-Leakage Limit

Michael P. Frank
University of Florida
CISE Department
mpf@cise.ufl.edu

Started Monday, March 24, 2003
Draft of Wednesday, March 26, 2003

Abstract:

In this memo, we analyze the maximum energy-efficiency boost that can be attained when converting arbitrary irreversible circuits to mostly-reversible ones in the same technology (using known reversiblization algorithms), in a scenario where leakage rates (but not entropy coefficients) are assumed to be negligible.

One important application of this analysis is for the analysis of proposed all-mechanical (as opposed to all-electronic or electromechanical) nanocomputing technologies, many which are predicted to have extremely miniscule leakage rates, such as Drexler's rod logic and Merkle's buckled logic, due to the high energy barriers that must be traversed to overcome steric intermolecular forces enough to cause undesired mechanical transitions. The device capabilities present in these all-mechanical logics must be extended somewhat beyond those original proposals in order to implement the most asymptotically efficient reversible computing hardware algorithms that are currently known, but this is straightforward to do.

The outcome of the study reported in this memo is that worst-case hardware efficiency scales down with roughly the 1.57 power of the energy savings in this scenario. Given the constant factors of rod logic, an energy savings factor of 1,000 can be obtained with a roughly comparable increase in hardware cost per unit of throughput even in the worst case, and perhaps much less than this for an optimized reversible circuit.

1. The Model

The model used in this paper is similar to that discussed in previous memos, #M15 and #M16. The sole device technology parameters we require for the present analysis are:

Device parameters:

- S_{iop} – Entropy generated per irreversible bit-operation
- S_{rdcf} – Entropy coefficient, describing the entropy generated per reversible device-cycle per unit of quickness (reciprocal of cycle time).
- $t_{\text{ic,min}}$ – Minimum time for an irreversible device-cycle (bit erasure event)

The design parameters are:

Design parameters:

- t_{rc} – Time chosen per reversible (adiabatic) device-cycle
- *whichAlg* – Choice of reversiblization algorithm used: currently this may be either BEN89 or FRANK02.
- n – Number of recursive levels in the reversiblization algorithm.
- k – Number of immediate sublevels per higher level.

Here are the important raw cost measures we will be wishing to minimize:

Cost metrics:

- $(dt)_{viop}$ – Spacetime cost (in terms of devices times time) per "virtual irreversible op," *i.e.*, per irreversible op in the original computation emulated in the mostly-reversible computation.
- S_{viop} – Entropy generated per virtual irreversible op.

The following are the important system-level quantities whose tradeoffs we wish to characterize:

Figure of merit:

- $A_S = S_{iop}/S_{viop}$ – Entropic advantage of reversible machine. We wish to maximize this for given D_{dt}.

Figure of demerit:

- $D_{dt} = (dt)_{viop}/(1 \cdot t_{ic,min})$ – Spacetime disadvantage (blowup) on reversible machine. We wish to minimize this for given A_S.

2. Analysis

In this analysis, we consider only the FRANK02 algorithm since it dominates BEN89 in both spacetime and energy (see memo 15). The algorithmic formulas here are also taken from that paper. Consider a complete run of the reversiblization algorithm with given n and k applied to a single irreversible device in the original design.

2.1. Algorithmic analysis.

- **Virtual irreversible ops per run.** $(viop)_{run} = k^n$.
- **Reversible ops per run.** $(rop)_{run} = (2k-1)^n$.
- **Device cycles per run.** $(dcyc)_{run}(k,n) = \{1 \text{ if } n=0, \text{ else } (2k-1)(dcyc)_{run}(k, n-1) + (k^2-3k+2)k^{n-1}/2\}$.
- **Irreversible ops per run.** $(iop)_{run} = 1$.

2.2. Physical analysis.

- **Adiabatic entropy generation per run.** $S_{adia,run} = (rop)_{run} S_{rdcf} / t_{rc} = (2k-1)^n S_{rdcf} / t_{rc}$.
- **Irreversible entropy generation per run.** $S_{irr,run} = (iop)_{run} S_{iop} = S_{iop}$.

- **Total entropy generation per run.** $S_{run} = S_{adia,run} + S_{irr,run}$ $= (2k-1)^n S_{rdcf} t_{rc} + S_{iop}$.
- **Device time per run.** $(dt)_{run} = (dcyc)_{run} t_{rc}$. Recall that *dcyc* is given by the above recurrence.
- **Entropy generation per virtual irreversible op.** $S_{viop} = S_{run} / (viop)_{run}$ $= [(2k-1)^n S_{rdcf} / t_{rc} + S_{iop}] / k^n$.
- **Device time per virtual irreversible op.** $(dt)_{viop} = [(dcyc)_{run} t_{rc}] / k^n$.

2.3. Figures of merit/demerit.

- **Entropic advantage.** $A_S = S_{iop} k^n / [(2k-1)^n S_{rdcf} / t_{rc} + S_{iop}]$ $= k^n / [(2k-1)^n S_{rdcf} / t_{rc} S_{iop} + 1]$
- **Spacetime disadvantage.** $D_{dt} = [(dcyc)_{run} t_{rc}] / k^n t_{ic,min}$.

2.4. Free parameter elimination.

We can reduce the number of free parameters in the figures of merit and demerit by defining a new derived design variable, the dimensionless *slowdown factor* $s = t_{rc} / t_{ic,min}$. This allows us to simplify D_{dt} to $(dcyc)_{run} s / k^n$. Also we can define a new derived technology constant, the dimensionless *entropy disadvantage at irreversible speed* $D_{Sti} = S_{rdcf} / t_{ic,min} S_{iop}$. This lets us simplify A_S to $k^n / [(2k-1)^n D_{Sti}/s + 1]$. With these changes, the only independent technology parameter remaining is D_{Sti}. The design variables are k (integer), n (integer), and s (real).

3. Optimization

We wish to generate an optimized tradeoff curve between the entropic advantage A_S (our figure of merit) and the spacetime disadvantage D_{dt} (our figure of demerit). That is, for any desired minimum value of A_S required, we wish to know the minimum value of D_{dt} that will suffice to yield it, when the design parameters are so optimized, as well as the optimized values of the design parameters. Or, equivalently, for any given maximum value of D_{dt} that can be tolerated within design constraints, we wish to find the maximum value of A_S that can be obtained when the design parameters are so optimized. We wish to graph this tradeoff curve numerically, and find a simplified analytical expression that approximates it as well as the optimized design parameters, in terms of the technology parameter D_{Sti}.

First, let us imagine that k and n are already optimized for the given D_{Sti} and the given maximum D_{dt} or minimum A_S. With k and n fixed, $(dcyc)_{run}$ is fixed also, and can be calculated using the recurrence relation mentioned earlier.

So now, if D_{dt} is given, we can calculate the maximum slowdown that can be tolerated as $s = D_{dt} k^n / (dcyc)_{run}$. We can then plug this into A_S to get $A_S = k^n / [(2k-1)^n (dcyc)_{run} D_{Sti}/D_{dt} k^n + 1] = 1 / [(2k-1)^n (dcyc)_{run} D_{Sti}/D_{dt} k^{2n} + k^{-n}] = 1 / [((2k-1)/k^2)^n (dcyc)_{run} D_{Sti}/D_{dt} + k^{-n}]$. Therefore all we need to do is search the space of possible n and k values to maximize this expression, or in other words to minimize $[(2k-1)/k^2]^n (dcyc)_{run} D_{Sti}/D_{dt} + k^{-n}$. Note we can combine the input variables D_{Sti} and D_{dt} into one dimensionless ratio $R = D_{Sti}/D_{dt}$, a combination of application constraints and technology parameters which is then the only independent parameter needed to determine the maximum value of A_S that we can obtain, by taking the reciprocal of the minimized value

of $[(2k-1)/k^2]^n$ $(dcyc)_{run}$ $R + k^{-n}$. Note that smaller values of R will lead to larger values of A_S. This makes sense, because if either the entropic disadvantage of a reversible gate running at the same speed as an irreversible gate (D_{Sti}) is low, or if the spacetime blowup that can be tolerated (D_{dt}) is high, then this gives a better opportunity for reversible energy savings.

4. Program

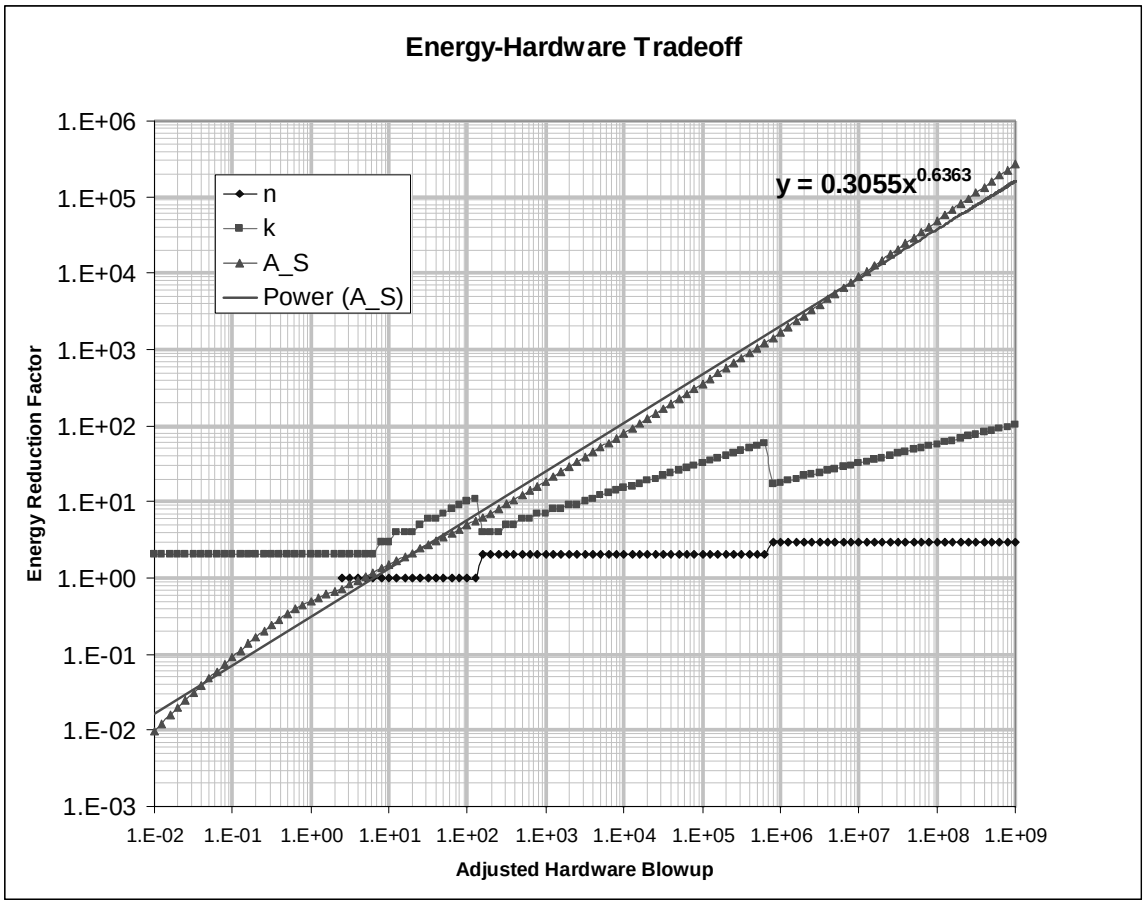
Here is a simple C++ program to generate results according to the above-described optimization procedure.

```
#include <stdio.h>
#include <math.h>

double deviceCycles_per_run_Frank02(int k, int n) {

   if (n==0) return 1; /* No recursive levels?  Just 1 unit. */

   /* Explanation of the below formula:
    - There are 2*k-1 subtriangles, each taking m(k,n-1) devcycles.
    - Plus, there are (k-2)(k-1)/2 "units" of temporary storage
      added by this supertriangle, where each "unit" is a bit
      stored over the total time taken by a subtriangle.
      Each subtriangle takes time k^(n-1) in my algorithm.
      (See my notes dated 3/23/02.) */

   return (2*k-1)*deviceCycles_per_run_Frank02(k,n-1) + (k*k-
3*k+2)*pow(k,(n-1))/2;
}

double entropy_disadvantage(double R, int k, int n) {
  return
    pow((2*k-1)/(double)(k*k),n) * deviceCycles_per_run_Frank02(k,n) *
R
    + pow(k,-n);
}

void minimize_entropy_disadvantage(double R) {
  int best_n = -1;
  int best_k = -1;
  double min_disadvantage = -1;

  for(int n=0; n<10; n++) {
    for(int k=2; k<200; k++) {
      double D = entropy_disadvantage(R,k,n);
      if (min_disadvantage == -1 || D < min_disadvantage) {
      best_n = n;
      best_k = k;
      min_disadvantage = D;
      }
    }
  }

  printf("%g\t%d\t%d\t%g\n", 1/R, best_n, best_k, 1/min_disadvantage);
}
```

```
void sweep_R() {
  double R;
  for ( R = 100; R >= .99e-9; R *= pow(.1,.1) ) {
    minimize_entropy_disadvantage(R);
  }
}

int main(int argc,char**argv){
  sweep_R();
}
```

5. Results

Below is a plot of the above program's output. The horizontal axis is the adjusted hardware blowup $B :\equiv 1/R = D_{dt}/D_{Sti}$, the hardware efficiency disadvantage that can be tolerated within the application, divided by the hardware disadvantage of a reversible device running at a speed at which its adiabatic losses are the same as the losses in an irreversible device, in the given technology. The blue-diamond and green-square lines show the values of n and k, respectively that are optimal in each case. The red-triangle line shows the value of A_S that can be obtained with the given adjusted hardware blowup.



The red line shows a least-squares power-law fit to the A_S data curve. In the range of values studied, the entropy advantage A_S scales roughly as the hardware blowup B to the

~0.64 power. Note that for extremely high values of B, the exponent of the power law (slope of the log-scale curve) increases slightly. Reading off the chart, the below table lists the approximate specific blowup factors that are required to attain various energy reduction factors:

Energy Reduction	Adjusted Hardware Blowup
10	400
100	15,000
1,000	500,000
10,000	12,000,000
100,000	300,000,000

Note that the above analysis assumes that the FRANK02 algorithm, specifically, is used to translate an arbitrarily irreversible computation to an equivalent reversible one. On the other hand, if a more efficient algorithm is used (perhaps specialized to the specific circuit being translated), the adjusted hardware blowup factor can be reduced to be equal to (at best) the energy reduction factor. Note however that if particularly efficient reversible devices are used with D_{Sti} (entropy disadvantage at irreversible speed), that is much less than 1, then the hardware blowup can potentially be less than the energy reduction factor.

6. Application to Rod Logic

In *Nanosystems* (p. 370, §12.7.4), Drexler summarizes the dissipation of reversible and irreversible storage elements as 0.03 zJ and 4.4 zJ respectively with a clock period of 1.2 ns. If this speed is taken to be the maximum speed for an *irreversible* rod logic device, this implies a D_{Sti} value of 0.03/4.4 = 0.0029. Since the real hardware blowup $D_{\mathrm{dt}} = B \cdot D_{\mathrm{Sti}}$, the numbers in the previous table must be multiplied by D_{Sti} to give us a real hardware blowup for the case of rod logic, as shown in the below table:

Energy Reduction	Actual Hardware Blowup
10	1.1
100	43
1,000	1,400
10,000	34,000
100,000	860,000

Note that these calculations assume that the original machine that is being transformed used irreversible devices *exclusively*. That is, we are giving the factor by which the dissipation from irreversible devices may be reduced via replacing them with reversible devices with a given factor reduction in hardware efficiency. If part of the original system's energy dissipation already consists of reversible device operations, our analysis will not necessarily optimize the entire system's dissipation. For that, we would have to add a parameter to the analysis which is the fraction of the device-operations in the original design which were already reversible. However, the overall results are expected to be very similar to these.

6. Conclusion

Using particularly low-leakage devices such as those possible using nanomechanical logic styles such as Drexler's rod logic, the replacement of arbitrary completely-irreversible logic with mostly-reversible logic leads, even in the worst case of automatic translation, where the specific reversible circuit algorithm used is not hand-optimized at all, to reductions in energy per operation that are roughly comparable in magnitude to the amount of increase in circuit complexity per unit of throughput.

So, for example, the dissipation of an all-irreversible rod-logic circuit can be reduced by a factor of 1,000 via a straightforward translation (using the FRANK02 algorithm) while reducing hardware efficiency by only a factor of 1,400. In this particular case, the optimized parameters of the algorithm happen to be n=2, k=55, meaning that every sequence of k^n = 3,025 cycles of the original computation is broken into 55 segments of 55 cycles each. Each segment is done reversibly, accumulating intermediate states, and then undone after reversibly recording its output. The entire sequence is then undone, after irreversibly recording its output. The increase in circuit complexity required to do this is a factor of about 55+55 = 110, but the increase in device-cycles per run is not quite so great as this because not all storage elements are used on every cycle, and so can be shared with other computations as we described in memo 15.

In conclusion, in rod logic computing scenarios (or other low-leakage technologies) where overall throughput is primarily limited by thermal constraints rather than by hardware cost, throughput can be increased through the increased use of reversible logic. The worst-case energy savings scales asymptotically with the 0.64 power of the hardware blowup, or in other words, the worst-case hardware blowup scales as the 1.57 power of the energy savings. Depending on the exact characteristics of the specific device technology, the constant factors may also favor reversible logic by an additional factor. Custom reversible circuits for specific applications should yield even greater benefits.