On Computing Power

Jean E. Vuillemin

Digital Equipment Corporation, Paris Research Laboratory
85 Av. Victor Hugo, 92563 Rueil Malmaison, Cedex France.

February 2, 1994

Abstract

We analyze in details some implementations of a challenging, yet simple application: CERN's *calorimeter*. We try both *general* purpose computer architectures (single and multi processors, Simd and Mimd), and *special* purpose electronics (full-custom, gate-array, FPGA) on the problem.

All measures are expressed in a single common *unit* for computing power: the Gbops[1]. It applies to all forms of digital processors, and across technologies. What's more, Noyce's thesis provides a reliable way to extrapolate Gbops benchmarks through *future time*, say up to year 2001.

The quantitative result of our analysis shows that *special* purpose processing is *an order of magnitude* more efficient than *general* purpose processing, on our specific problem. We show how to map the calorimeter on a *programmable active memory* PAM[2], at performance and cost comparable to those of fully dedicated implementations: orders of magnitude better than any general purpose implementation, in 1992. We argue that this current computational power advantage for PAM technology will *increase* with time.

Finally, we discuss how to program such novel *virtual* PAM computers in the **2Z** language, for very large synchronous designs.

[1] 10^9 binary operations per second
[2] large array of configurable logic

1 Noyce's Thesis

Since the advent of modern digital computers, we have lived through 40 years of an exponential growth which is unique in the technical history of mankind.

Thesis 1 (Noyce) *The computing power per unit doubles every year.*

This was first pointed out, in an equivalent form, by R. Noyce in the early sixties. Circuit technology integrates many contributions: they arise from almost every area of modern science and technology, spanning the range from solid state physics to digital computer architecture and programming languages.

Yet, as Noyce observed, the complex combined cumulative effect of all these punctual and discrete advances is *as if*, the feature size of our circuit manufacturing technology was simply shrinking *linearly* with time. As far as experimental evidence goes, and it is plentiful, the average shrink factor per year is about $\alpha \approx 1.25$. As documented further by C. Thacker, the shrink factor has remained statistically steady for over 30 years, and we have every reason to believe that it will keep doing so in the near future, say up to year 2001 (see [14]).

Let G (a natural number) be the number of logic *gates* which one can effectively fit within a unit area by the end of year y, and F (in Hertz) be the *frequency* at which one can reliably operate such gates. The *computing power* is the product of these two figures:

$$P = G \times F. \tag{1}$$

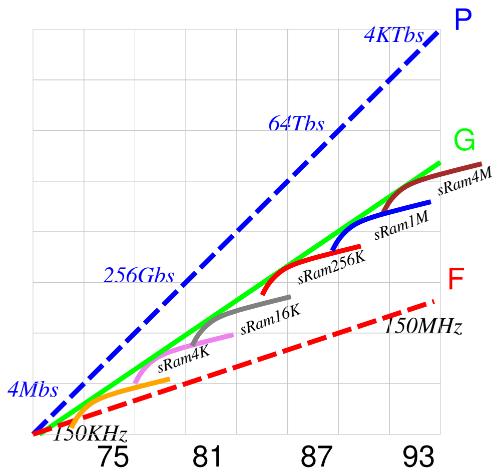


Fig. 1. Plots of the growth rates of G (number of storage bits), F (operating frequency) and P = GF for *static RAM* technology.

By the end of the next technology year $y' = y + 1$, the corresponding figures in (1) are $G' = \alpha^2 G$ and $F' = \alpha F$. Choosing $\alpha = \sqrt[3]{2}$, a good approximation, we find that $P' = 2P$, which is how we stated Noyce's thesis.

Keep in mind that it is just an observation about human science; it is neither a law from physics, nor a theorem from mathematics. It has to do with economics and technology questions, such as: Is GaAs faster than ECL? Will BiCMOS take over CMOS? At what cost?

2 Summary

Noyce's thesis provides a nice and simple model against which we attempt to analyze the impact of time, i.e. technology, on computer architecture. For this purpose, we start from a single application: CERN's *calorimeter*.

- The problem is simple enough to be fully stated in section . Its computing requirement is analyzed, step by step in section 5.
- It is part of a series of benchmarks put forward by CERN[3] [3]. The goal is to measure the performance of various computer architectures, in order to build the electronics required for the *Large Hadron Collider* LHC, before the turn of the millennium.
- It is challenging, and well documented: [3] benchmarks a dozen electronic boxes, including some of the fastest current computers, on the calorimeter and other problems.

Our object is to complement CERN's *experimental* calorimeter benchmarks by a convergent, more *theoretical* analysis; and to use it in order to make some predictions regarding the future.

We try two types of solutions.

The first are representative of the computing power achieved by *general purpose* computer architectures on the calorimeter: this year's fastest computer on a chip; compared to both *massively* and *moderately* parallel implementations. We analyze the cycle time required for such machines, and predict a year when each should become technologically feasible.

The last is representative of the computing power delivered by *special purpose* digital architectures, specifically designed to perform the calorimeter's computation. An implementation in PAM technology is presented in section 7.2. It is

[3]Centre pour l'Etude des Réactions Nucléaires, in Geneva, Switzerland

representative of three related *design methodologies*: full-custom, gate-array and *field programmable gate array* FPGA.

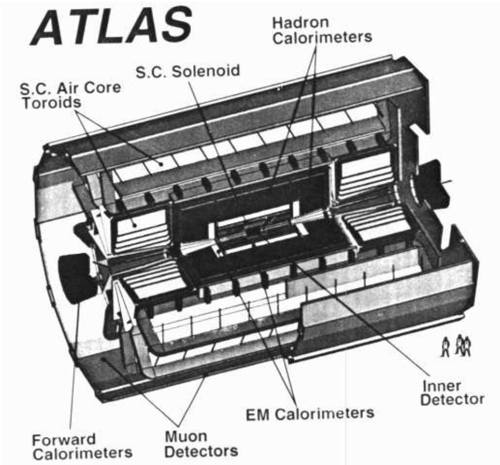


Fig. 2. CERN's view of the LHC.

We apply the same evaluation method to all cases:

1. First assess the *theoretical* computing power of the machine; all are expressed in a common unit, the Gbops.
2. Next, we analyze the *actual* computing power, as measured by running the machine on the calorimeter.

3 CERN's Calorimeter

The function of the calorimeter is to identify the position and most likely nature of a particle that traverses a digitized region of interest, which is a square $\mathcal{S} = \{i, j : 0 \leq i, j < 20\}$. Within the LHC, such traversals occur at the rate of 100KHz, i.e. each 10μsec.

The input is a pair of energy maps ($E', E''[i, j]$ for $i, j \in \mathcal{S}$) providing the line-by-line responses from two analog detectors, digitized down to 16 bits. The average input rate is 160MB/s, presented on two channels (32b, 20MHz each). The analysis of event (E', E'') is done by computing:

E The pixel-wise sum: $$E = E' + E''.$$

S The total energy: $$S = \sum_{\mathcal{S}} E[i, j].$$

M The maximum energy: $M = E[i_m, j_m] = \max_{\mathcal{S}} E[i,j]$.

O The first statistical moment: $O = \sum_{\mathcal{S}} r_m[i,j] \times E[i,j]$,
centered at E's maximum. Here, $r_m[i,j]$ is a tabulated 8b integer approximation of the distance $\sqrt{(i-i_m)^2 + (j-j_m)^2}$.

P The peak energy: $P = \sum_{\mathcal{S}} p_m[i,j] \times E[i,j]$;
here $p_m[i,j]$ is one if $|i-i_m| + |j-j_m| < 2$, zero otherwise.

D The final discriminant: $D = \mathrm{sign}(\alpha \frac{O}{S-M} - \beta \frac{P}{S})$.
The decision between an *electron* and a *hadronic jet* is based on computing the sign of D, for some (experimentally determined) suitable 16b integers α and β.

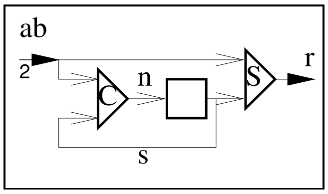
The whole computation may be carried out with 16-bit integers. The output rate is only 100 kb/s, one bit per energy pair (E', E''). The computation of the *maximum* implies that we must buffer a full energy map, between steps **E**, **M** and steps **O**, **P**, **D**.

4 The Gbops

Our only analytical tool so far is definition (1) of the computing power, which is a strictly *digital* measure. The exact *analog* process through which our *mathematical* computing power gets physically delivered does not matter here. What exactly is a *gate* is not important either: it only affects our measure by a constant multiplicative factor (provided that we keep *bounded fan-in*).

Our favorite accounting unit calls *one* any operation which is not more complex than a single bit-serial binary addition. Or subtraction, for that matter; or any gate with at most *three* inputs, and *one* bit of internal state.

Let **1 Gbops** be 10^9 *binary operations per second*, our unit for computing power. It is delivered by any *Bop* circuit, *operating* at 1 GHz.



Each Bop circuit is made of two boolean functions $S, C \in \mathbf{B}^3 \mapsto \mathbf{B}$ and a synchronous register; they are connected as shown in the schema above, or the **2Z** code which follows.

```
Bop(a, b) = (s, r)
where
  n = C(a, b, s); // Next state
  s = reg(n);     // Flip flop
  r = S(a, b, s); // Result
end where;
```

Bop circuits include the *bit-serial binary adder*, which is obtained by choosing:

$$\begin{aligned} S(a,b,s) &= (a+b+s)\cdot|\cdot 2, \\ C(a,b,s) &= (a+b+s) \div 2. \end{aligned}$$

The accounting rules which follow, for arithmetic and logic operations over n bit word inputs, are straightforward:

$+$ One $n+n \mapsto n+1$ bits addition each nano second is worth n Gbops. Subtraction, integer comparison and logical operations are bit-wise equivalent to addition.

$\times$ One $n \times m \mapsto n+m$ bits multiplication each nano second is worth nm Gbops. Division, integer shifts and transitive (see [15]) bit permutations are bit-wise equivalent to multiplication; consequently, so is a $n \mapsto m$ look-up table LUT, or RAM access.

5 Calorimeter Analysis

We now count the number of Gbops required by each step of the calorimeter.

E The input is composed of four digital flows: 4×16b×20MHz. We must add together the first and last two flows: $\begin{aligned} E_t[0] &= E'_t[0]+E''_t[0]; \\ E_t[1] &= E'_t[1]+E''_t[1]. \end{aligned}$
Each addition requires 16 binary operations per cycle: $P_e = 16 \times 2 \times 20M = 0.64$ Gbops.

S The input is : 2×16b×20MHz. We sum all energies from the same map: $P_s = 0.64$ Gbops.

M The maximum can be computed by using the sign of the subtraction to select the proper argument, at a cost of 640 Mbops; together with keeping up to date the 10b index (i_m, j_m): 0.4 Gbops. In addition, the maximum requires to *store a complete map*, in a 400×16b double access RAM. So we charge $2 \times 10 \times 16 \times 40M = 12.8$ Gbops. Total: $P_m = 13.84$ Gbops.

O The first statistical moment is the most complex operation. It requires a 20b$\mapsto$16b look-up table LUT for finding the distance $r_m[i,j]$: 12.8 Gbops. The 8b multiplication accounts for 5.12 Gbops. Total: $P_o = 17.92$ Gbops.

P The peak is the cheapest operation: at 5 additions 16b per *map*, it requires a negligible: $P_p = 8$ Mbops.

D The discriminant is expensive, $2 \times 16 \times 48$ *active bits*; it is executed once per map, so the computing power required here is only: $P_d = 154$ Mbops.

The computing power required by the complete calorimeter computation is the sum $\mathcal{P} = P_e + P_s + P_m + P_o + P_p + P_d$, namely $\boxed{\text{33 Gbops}}$.

Note that our accounting does not take into consideration any of the required data movements: from input to processing unit; from processing unit to output. Such transport operations do not transform values; they do not directly contribute to the final decision D; they get charged here as overhead, exclusively accounted for in the *virtual* computing power of the implementation technology.

6 General Purpose Architectures

To make the analysis simple, we give *general purpose* technology *all benefits from the doubt*: caches are all assumed to be wide enough and fast enough, in order to provide each Cpu with data and valid instructions, at no latency but the minimum feasible.

Ignore the fact that both data and instructions caches would have to be *huge*, by 1992's standards. This permits to perfectly *streamline* the calorimeter's computation: unroll all loops, and take one cycle per fetch or store, on every memory access.

Ignore the fact that, in 1992, none of the general purpose machines by CERN could cope with the calorimeter's external input bandwidth of 160MB/s. So, the input had to be *faked* in the experiments.

6.1 Single Processor

In 1992, the highest performance microprocessor has 64b of data, clocked at 200MHz. The *virtual* computing power of this Cpu64b200MHz is $64 \times 0.2 = 12.8$ Gbops. We know from the calorimeter analysis that this processor is not fast enough.

Let us analyze the number of clock cycles required for running the calorimeter on a *reduced instruction set* Risc processor. In a streamlined code, the number of *cycles* required to process the calorimeter, for each $16b \times 20 \times 20$ energy map, is:

$$\begin{aligned} S &= C_e + C_s + C_m + C_o \\ &= 5 + 2 + 3 + (4 + 16), \\ C &= 400S + C_p + C_d \\ &= 12.1 \text{ K cycles.} \end{aligned}$$

The calorimeter operates at 100KHz; so, the minimum cycle time at which we can expect to run this program is $\boxed{\text{1.2 GHz}}$ Noyce's thesis predicts that this Cpu64b at 1.2GHz will be technologically feasible around year 2000.

The moment **M** gets computed in 4 cycles for the look-up table LUT, and 16 cycles for the actual multiplication. On a machine with a hardware multiplier, C_o gets reduced to 8 cycles. So the clock at which we need operate the calorimeter is only 720 MHz. A Fpu64b720MHz should be feasible by year 1998.

The virtual computing power of Cpu 64b at 1.2GHz is 87 Gbops. The virtual computing power of Fpu64b at 720MHz [4] is 1656 Gbops.

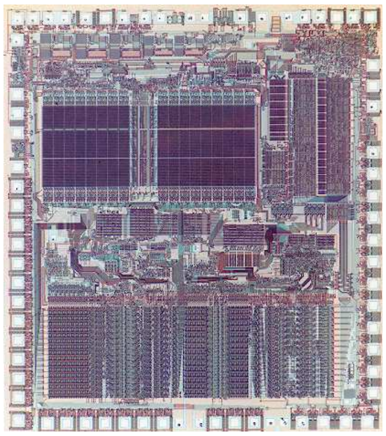


Fig. 3. In this popular micro processor, the area of the actual Cpu is only 1/200-th of the whole.

In both cases, the computing power actually expanded on the calorimeter is only 33 Gbops. The respective virtual to actual power ratios are about 3 and 47. Observe that Fpu delivers at most 4.8 Gbops when it is only computing additions or equivalently cheap operations: a very low utilization of the computing power virtually available in the multiplier.

[4] A 64b floating point unit, with 48b mantissa and 16b exponent, which operates at 100MHz delivers 230 Gbops.

Note that large data paths (32b or 64b) past 16b do not help the calorimeter: the whole computation can be performed on 16b integers, except for the final decision where 48b are convenient. To conclude on single processors, our best fit to the calorimeter are

Cpu16b1.2GHz The ratio between the 43 Gbops (equals 19 for Cpu plus 24 for the LUT and RAM) virtual power and the 35 Gbops actual power *is near one.* We achieve an optimal fit where the 16b computed in each cycle all contribute to the calorimeter's decision.

Fpu16b720MHz The ratio between the 189 Gbops virtual power and the 35 Gbops actual power is near five. The multiplier is used at less than one fifth of its capacity.

Although Cpu16b1.2GHz makes the single processor RISC software solution *optimal* for the data path of the calorimeter, it is hiding a *large* structure (with high Gbops virtual cost) for handling its hierarchical data and instruction memories. There is a *lot more* to Cpu16b1.2GHz than just its Alu16b. If you care to look at Figure 3, the part of interest here is only a *tiny* fraction of the silicon area in the full microprocessor.

6.2 Multi Processors

The class of massively parallel processors fares poorly on CERN's calorimeter. The strong experimental evidence provided in [6] can probably be explained by observing that any attempt to process the calorimeter on a pool of *slow* processors implies a *large* cost: the amount of *memory* required is proportional to the number k of processors used; so is the bandwidth required for communicating the proper data to the proper processing units.

Massively parallel solutions to the calorimeter are ruled out by economics and engineering problems. A 4k=4096 parallel processors 4b Simd operating at 12MHz machine 4kPP4b12MHz has a virtual computing power of 200 Gbops. Yet, the implied cost in memory and communication makes it incapable, in 1992, to compute the calorimeter near real time.

Processing independently the six steps of the algorithm is the best room available for parallelism in the calorimeter. Each processor performs some of the steps (E,S,M,O,P,D).

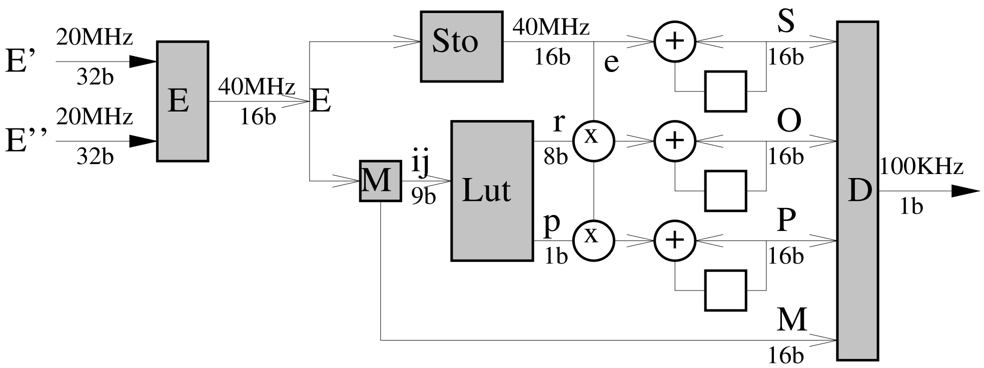
This multiple instruction, multiple data parallel machine operates at the speed of its slowest component, namely the moment unit **M**. Using here both a LUT and a multiplier 16b, we reduce C_o to eight cycles. Each 16b processor is now only required to operate at $40 \times 8 = 320$ MHz in order to process the calorimeter. Such a parallel Mimd processor 6PP16b320MHz should be feasible before 1996.

Note that the bandwidth required between the 6 processing units is $8 \times 80 = 640$ MB/s, a taxing requirement for all general purpose architectures.

Past such a simple six long assembly line organization, there is little more to be gained through parallel processing: the increase in storage and communication is not worth the benefit in effective operations.

7 Special Purpose Architectures

From the nature of the *physical* interface of the calorimeter (input on two HIPPI channels 32b20MHz, output on the host's TURBOchannel 32b25MHz), we know that a minimal size electronics has four printed circuit boards (say 6cm×8cm in 1992): two for input, one for output, and one board for the calorimeter algorithm per se, and connecting the other boards.



The calorimeter algorithm maps directly into eight functional units: schemas above, **2Z** code below.

```
Calorimeter({E', E''} : [32]) = D
where
  E = AddUnit(E', E''); // 100MBs peak output
  // 159 zeroes and a one, period 200 bits
  c159 = Sdd(2**159/(1-2**200));
  (ij, M) = MaxUnit(E, c159);
  (r, p) = Lut(ij, c159);
  e = Sto(E, c159);   // Delay 10mus
  reset c159 do
    S = SumUnit(e);
```

```
    P = PeakUnit(e, p);
    O = MomentUnit(e, r);
  end reset;
  D = OutputUnit(P, S, O, M, c159);
end where;
```

7.1 Hardware Blocks

We present the function of each atomic unit; when it is relevant, we provide the **2Z** code from which the PAM configuration for the corresponding unit can be derived. We also analyze the virtual computing power required by each step of the PAM implementation.

The whole design is synchronized to the 160MB/s input, by a 40MHz clock. Both the period (10μs) and the delay (20μs) are kept at their absolute minimal values. Each arithmetic unit is precisely tailored to its function: 16 bits parallel operators for steps 1 through 5, connected according to the schematics below. Step 6 is implemented in a fully bit-serial manner, to take best advantage of the low bandwidth requirement on this final electron/hadronjet decision.

E The input is composed of four parallel digital flows: 4×16b×20MHz. We first interleave $E'_t[0]$ and $E'_t[1]$ in time, so that $E'_{2t} = E'_t[0]$ and $E'_{2t+1} = E'_t[1]$; similarly, interleave $E''_t[0]$ and $E''_t[1]$ so as to produce E'' at 16b40MHz. In PAM technology, each interleave is realized by a specific column of 16 Pabs[5], at cost: 2×16b40MHz = 1.28 Gbops. We add together the two flows through a 16b adder at 40MHz. The required computing power is 3×16 Pabs: $P_e = 1.92$ Gbops.

M Computes the maximum M of the current map E at 16b100KHz and its index (i_m, j_m) at 10b100KHz. In PAM technology, the maximum is best implemented from *high to low* bits (see [7]). The computing power of this unit is $2 \times (16b + 10b) \times 40$ MHz = 2.08 Gbops.

Sto Double buffers the current $400 \times 16b$ energy map E while reading the previous one e. Both flows are 16b40MHz. In our PAM implementation, we use a $2 \times 400 \times$ 16b40MHz double access RAM: 12.8 Gbops, and 400 Mbops to control the addresses. Total: $P_{sto} = 13.2$ Gbops.

S Sum all energies from the same map, with a 16b accumulator: $P_s = 640$ Mbops.

[5] Programmable Active Bit

```
SumUnit(E:[16])=S:[16]
where
  R = Add(16)(E, S, 0);
  S = Reg(16)(R);
end where;
```

P The peak is the computed at full cost: $P_p = 640$ Mbops. Bit $p = p_m(i,j)$ is produced by the LUT.

```
PeakUnit(E:[16],d)=P:[16]
where
  for k<16 do // sum when d=1
    F[k] = E[k] & d
  end for;
  P = SumUnit(F);
end where;
```

O The first statistical moment is implemented by a 20b$\mapsto$9b LUT for finding the 8b distance $r_m[i,j]$ and the 1b peak $p = p_m(i,j)$. The address control for this RAM uses 800 Mbops. Including the 8b multiplication, we find: $P_o = 18.72$ Gbops.

```
MomentUnit(E:[16],D:[8]) = O:[16]
where
  P = Mul(16, 8)(E, D);
  O = SumUnit(P[0..15]);
end where;
```

D Each of the M,P,S,O units takes inputs at 16b40MHz and produces 16b of output at 100KHz. The four outputs (M,P,S,O) are consumed by the decision unit, 4×16b at 100KHz, in order to produce the final decision D at 1b per 100KHz. The PAM discriminant is detailed in section 8.

Note that the virtual computing power rquired for our PAM calorimeter is only $\mathcal{P} = \boxed{39 \text{ Gbops}}$. The ratio between actual and virtual power is very near one, as for Cpu16b at 1.2GHz. The difference is that here, the whole chip area is devoted to the calorimeter computation. There is no hidden virtual cost for managing PAM data.

From this level of description, we can design and implement a *full-custom* solution in one chip. That makes up for a relatively empty calorimeter board: a single chip and lots of connectors.

An easier solution is to realize all but the LUT stage in a calorimeter *gate-array*; implement the LUT by a RAM; this is a two chips implementation of the calorimeter board.

We can implement the calorimeter on a generic PAM board (same size as all others). It is composed of two RAM banks, one FPGA, two input connectors and one output connector. It can be ready made from off the shelf components.

The only difference between our three boards is their cost per unit. All are functionally equivalent calorimeter implementations. They also have equal performance.

7.2 Programmable Active Memories

As our reader is not assumed to be familiar with this technology, let us survey some of the concepts in this new emerging field. The following is from [5]:

Definition 1 (PAM) *A PAM is a uniform array of identical cells all connected in the same repetitive fashion. Each cell, called a Pab (for Programmable Active Bit) is* configurable *enough so that the following holds true:* **any** *synchronous digital circuit can be realized, through suitable configuration of each Pab, on a large enough grid and for a slow enough clock.*

A Pab is the basic building block out of which FPGAs are built. There are many ways to construct a Pab which has the required generality. The FPGAs from [2], [9], [10] and [17] present four rather *different* implementations of the concept. The Bop circuit from section 4 provides another example of universal Pab.

It should be pointed out that the five Pab structures mentioned so far do not exactly have the same computing power: while it only takes one of either [17] programmable active bit to implement a serial adder, it takes two of [9] and four of [2] or [10] to realize the same function. Such factors must be accounted for in the detailed analysis of their virtual computing power. With our Pab=Bop choice, we simply count one binary operation per Pab.

It takes quite a bit more than our PAM definition to obtain a workable and powerful general purpose digital engine. The most important designs issues involved are thoroughly discussed by P. Bertin in [4]. Besides ours, which were built at INRIA and DEC-PRL, other successful PAM implementations have been reported, in particular at the Universities of Edinburgh [12], Zurich [11], and at Maryland's SRC [1]. Let us also mention [13] which is a large PAM, dedicated to hardware emulation.

The ratio between the theoretically available computing power, and that practically usable for the calorimeter is *much* lower for *dedicated* hardware than for general purpose solutions. PAM technology combines the best from both:

- being a universal virtual machine, the PAM can be configured to a wide class of computing units. As *software*, it is by no means limited to processing a single application.
- being configurable at the *gate and wire* level, a properly dimensioned PAM can emulate efficiently each special purpose hardware. A *fixed* size PAM, say 16×20 Pabs at 40MHz, has some well defined virtual computing power: 12.8 Gbops. With some design effort, it was found on a large number of test cases that such PAM can simulate *in real time*, any specific dedicated synchronous *hardware* whose computing power is less than 12.8 Gbops.
- the benefits derived from processing the calorimeter through special purpose hardware are large; they are representative of a wide class of applications, for which PAM technology provides today an optimal implementation medium.

We demonstrate in [6], through 10 benchmarks which cover a wide range of applications, drawn from arithmetics, algebra, geometry, physics, biology, audio, video and data compression that, our implementation vehicle DECPeRLe$_1$ consistently performs in the 100Gbops range.[6]

8 PAM Programming

By its nature, the PAM has lead us to implement hardware algorithms which are substantially bigger than anything yet attempted on single silicon chips: on the order of 250K gates (or 2M transistors), excluding RAMs. The sheer size of such designs has forced us to aggressively pursue the strictly synchronous design paradigm, throughout the PAM implementations reported in [6].

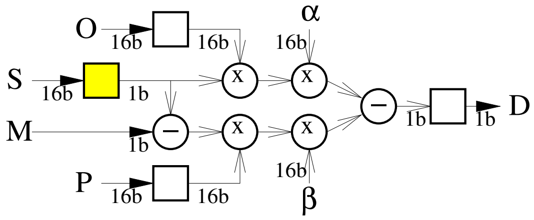
It has quickly become clear that arithmetic circuits are the key to success in this area. Obviously, each of our implementations only requires a finite arithmetic precision. However, any design system which claims to cover the whole spectrum (from 1b to 4Kb!) requires the ability to handle truly arbitrary precision arithmetic.

The natural mathematical domain into which this leads us is that of the 2adic numbers, both discovered and created by K. Hensel around 1900. In [16], we uncover some of the intimate relationships which exist between digital synchronous

[6]Note that the present definition of the Gbops is only one **half** of that used in [5]; the aim is to simplify out useless constant factors, as one serial bit of addition now amounts to one bop, no longer two as in [5].

circuits and 2adic numbers. Capitalizing on these results, [8] attempts to introduce a new programming, named **2Z** , whose main function is to help *concisely* define synchronous circuits.

The most classical features of **2Z** have already been illustrated through examples, since the beginning of this paper. Let us complement them by the source **2Z** code for the decision unit **D**. This fully exploits the facilities for bit serial arithmetic synthesis, which are quite unique to the **2Z** language.



The **2Z** code corresponding to the above schema is:

```
OutputUnit ({P, S, O} : [16], M, c159) = D
where
  s = ParSer(16)(S, c159);
  enable c159 do
    O′ = Reg(16)(O);
    P′ = Reg(16)(P);
  end enable;
  // serial logic synthesis
  u = sMul(16)(s, O′);
  l = sMul(16)(s - m, P′);
  d′ = (u * a) - (l * b);
  // Cern's constants
  a = 134535;
  b = 767665;
  // controls
  enable Fin do
    D = reg d′
  end enable;
  reset c159 do
    Fin = Sdd(2**48);
  end reset;
end where;
```

See [8] and [16] for more details.

9 Conclusion

Under Noyce's thesis, we have established the following.

I The computing power of a single fast Cpu 16b or 64b will grow by a factor *two* each three years. It should reach the 1.2GHz calorimeter frequency before year 2001. By then, the computing power actually delivered on the calorimeter will be 33 Gbops.

II The computing power available in a FPGA will grow by a factor *eight* each three years. Let us pick a for starting point 400 Pabs at 40MHz in 1992. By Noyce's thesis, the corresponding figures by year 2001 should be 25.6K Pabs and 320MHz: 8 Tbops per cm^2!

So, by year 2001, a single chip FPGA will have 200 times the computing power of the fastest sequential Cpu. A small PAM will be three orders of magnitude more powerfull than a single Cpu64b at 1.2GHz. An equivalent way to look at this: in 92, a 16×20 FPGA at 40MHz has the computing power of a vintage 2001 Cpu64b, at 1.2GHz. Two things are clear.

1. PAM technology will inevitably become an important contributor to the high power scientific computations, before the turn of the millennium.

2. General purpose computers will have to become multi-processors, with a relatively large number of processors, in order to sustain the competition.

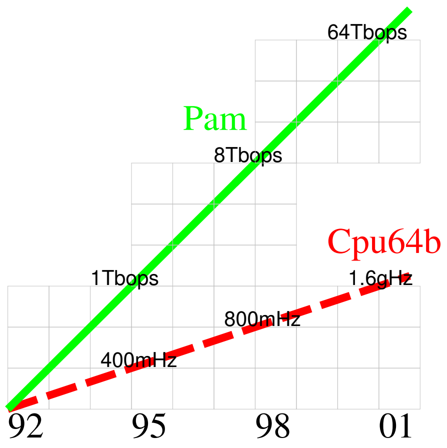


Fig. 4. The future of two technologies?

From our experience, it is clear that the main obstacles to the development of PAM technology come from the current state of computer aided design Cad system: it is much harder to program a PAM that a serial Cpu.

The Cad tools all run on sequential Cpus. The computing power available to run the Cad tools does not scale along with PAM technology.

The **2Z** language is a small step towards meeting such Cad challenges. Orders of magnitude must be gained over the current design techniques, in order to implement the truly huge PAM designs for year 2001. We now deal with 10K Pabs; by then, we shall have 1M gates to design, place and route.

Based on our observations, it is tempting to venture the question:

What computing power will be available in a shoe size box by year 2001?

According to the theories exposed here, and taking 256 Gbops as a reference point for 1993, we predict:

$$\boxed{\text{68 Tbops!}}$$

By then, Niklaus Wirth will also be 68, if God permits.

References

[1] J. Arnold, D. Buell and E. Davis, *Splash II*, in *4th ACM Symposium on Parallel Algorithms and Architectures*, San Diego, California, USA (1992).

[2] Algotronix Ltd., *The Configurable Logic Data Book*, Edinburgh, UK (1990).

[3] J. Badier, R. Bock et al., *Evaluating Parallel Architectures for two Real-Time Applications with 100kHz repetition rate*, EAST note, CERN.

[4] P. Bertin, *Mémoires actives programmables: conception, réalisation et programmation*, Thèse, Université Paris 7, 1993.

[5] P. Bertin, D. Roncin, and J. Vuillemin, *Introduction to Programmable Active Memories*, in *Systolic Array Processors*, J. McCanny, J. McWhirter, E. Swartzlander Jr. editors, pp 301–309, Prentice-Hall (1989). Also as PRL report 3, Digital Equipment Corporation, Paris Research Laboratory, 85, av. Victor-Hugo, 92563 Rueil-Malmaison Cedex, France (1989).

[6] P. Bertin, D. Roncin, and J. Vuillemin, *Programmable Active Memories: a Performance Assessment*, in *Symposium on Integrated Systems*, Seattle, WA, USA, March 1993, MIT Press (1993). Also as PRL report 24, Digital Equipment Corporation, Paris Research Laboratory, 85, av. Victor-Hugo, 92563 Rueil-Malmaison Cedex, France (1993).

[7] P. Boucard, J. Vuillemin, and M. Shand, *Calorimeter Collision Detector on DECPeRLe$_1$*, PRL report 40, Digital Equipment Corporation, Paris Research Laboratory, 85, av. Victor-Hugo, 92563 Rueil-Malmaison Cedex, France (1994).

[8] F. Bourdoncle, J. Vuillemin, and G. Berry, *The 2Z Report*, PRL report 36, Digital Equipment Corporation, Paris Research Laboratory, 85, av. Victor-Hugo, 92563 Rueil-Malmaison Cedex, France (1994).

[9] W. S. Carter, K. Duong, R. H. Freeman, H. C. Hsieh, J. Y. Ja, J. E. Mahoney, L. T. Ngo, and S. L. Sze, *A User Programmable Reconfigurable Logic Array*, in *Proc. IEEE 1986 Custom Integrated Circuits Conference*, 233–235 (1986).

[10] Concurrent Logic, Inc., *Cli6000 Series Field-Programmable Gate Arrays*, Concurrent Logic Inc., 1270 Oakmead Parkway, Sunnyvale, CA94086, USA (1992).

[11] B. Heeb and C. Pfister, *Chameleon, a Workstation of a Different Colour*, in *2nd International Workshop on Field-Programmable Logic and Applications*, paper 5.6, Vienna, Austria (1992).

[12] T. A. Kean and J. P. Gray, *Configurable Hardware: two Case Studies of Micro-Grain Computation*, in *Systolic Array Processors*, J. McCanny, J. McWhirter, E. Swartzlander Jr. editors, pp. 310–319, Prentice-Hall (1989).

[13] Quickturn Systems, Inc., *RPM Emulation System Data Sheet*, Quickturn Systems, Inc., 325 East Middlefield Road, Mountain View, CA 94043, USA (1991).

[14] C. P. Thacker, *Computing in 2001*, Digital Equipment Corporation, Systems Research Center, 130 Lytton, Palo Alto, CA94301, U.S.A.

[15] J. Vuillemin, *A Combinatorial Limit to the Computing Power of VLSI Circuits,* IEEE trans. on Computers, Avril 1983.

[16] J. Vuillemin, *On Circuits and Numbers*, PRL report 25, Digital Equipment Corporation, Paris Research Laboratory, 85, av. Victor-Hugo, 92563 Rueil-Malmaison Cedex, France (1993).

[17] Xilinx, Inc., *The Programmable Gate Array Data Book*, Xilinx, 2100 Logic Drive, San Jose, CA 95124, USA (1991).