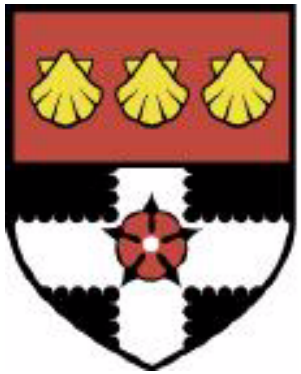
Perspex Machine
SPIE 2002

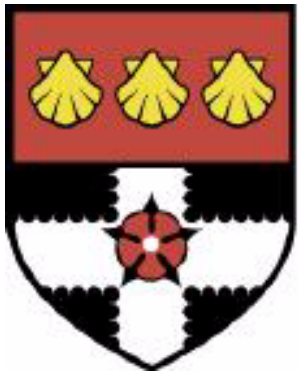
Dr. James A.D.W. Anderson
Computer Science
The University of Reading
England



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Aims

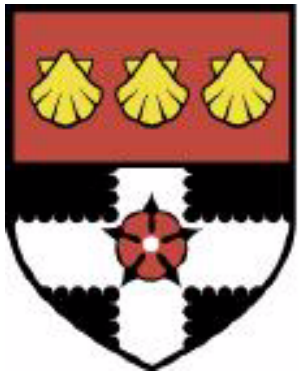
- To unify projective geometry and the Turing machine.
- To specify a fast, optical computer.
- To develop an anisotropic spacetime.
- To modularise perspex programs in spacetime.
- To suggest how to measure physical time-flow.



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Introduction

- **Unification:** Definition of Perspex machine.
  Simulation of Turing machine.
  Rational machine Turing equivalent.
  Irrational machine is supra-Turing.
- **Optics:** Petahertz machine?
- **Anisotropy:** Spacetime.
  Modularisation of perspex programs.
  Physical Experiment.
- **Conclusion**
- **Questions**



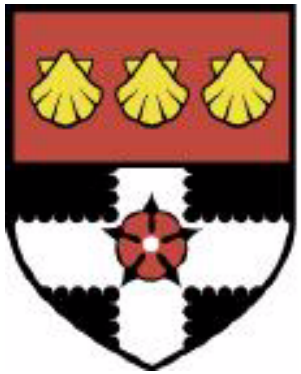
© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Unification: Perspex

- A *perspex space* or *program space* is a rational or real 4D space of Cartesian co-ordinates, optionally augmented by the point at nullity.

- Perspexes can be constructed in any whole-numbered dimension, but here we define that a *perspex* is an active, pre-multiplying, 4×4 matrix of homogeneous co-ordinates with column vectors x, y, z, and t.

$$\begin{bmatrix} x_1 & y_1 & z_1 & t_1 \\ x_2 & y_2 & z_2 & t_2 \\ x_3 & y_3 & z_3 & t_3 \\ x_4 & y_4 & z_4 & t_4 \end{bmatrix}$$



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

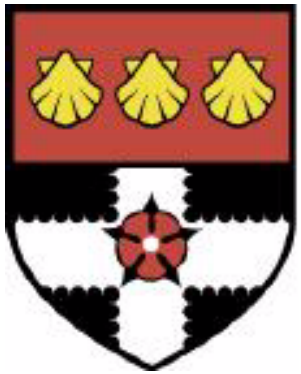
Unification: Canonical Form

- The canonical form of a perspex is defined as follows.

$$kA \equiv A \text{ with } k = \begin{cases} 1/a_{44}, a_{44} \neq 0 \\ 1, \text{otherwise} \end{cases}$$

- This is *not* sufficient to define the canonical form of perspective transformations directly, but it is sufficient to obtain a supra-Turing machine and then to obtain a canonical perspective transformation indirectly.

- Notice that the canonical value $a_{44} = 0, 1$. This causes anisotropy.



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Unification: Instruction

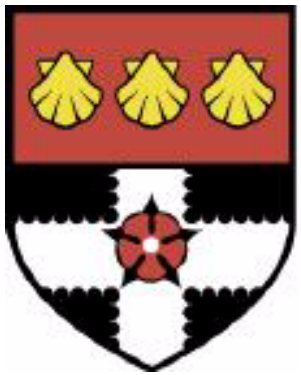
- A perspex machine has one instruction.

$$\vec{x}\vec{y} \rightarrow \vec{z}$$

$$\text{jump}(\vec{z}_{11}, t)$$

- The instruction is isomorphic to a perspex with column vectors x, y, z, and t.

- $\vec{x}\vec{y} \rightarrow \vec{z}$ obtains the perspexes pointed to by the vectors x and y, performs a matrix multiplication on the perspexes, then reduces the result to canonical form. The canonical result is written into the location in program space pointed to by z.



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

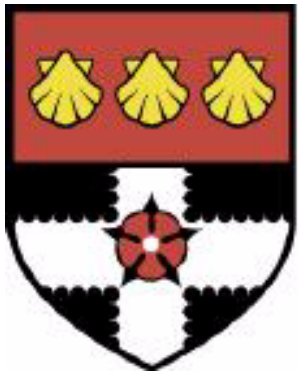
Unification: Instruction

- jump$(\vec{z}_{11}, t)$ causes control to jump by the relative vector j to a new location.

$$j_1 = \begin{cases} t_1, \vec{z}_{11} < 0 \\ 0, \text{otherwise} \end{cases} \qquad j_2 = \begin{cases} t_2, \vec{z}_{11} = 0 \\ 0, \text{otherwise} \end{cases}$$

$$j_3 = \begin{cases} t_3, \vec{z}_{11} > 0 \\ 0, \text{otherwise} \end{cases} \qquad j_4 = t_4$$

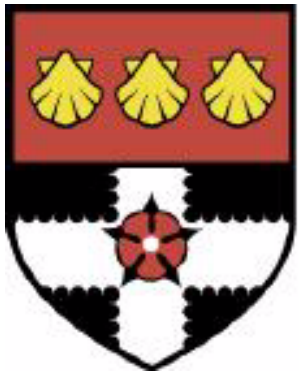
- In canonical form $t_4 = 0, 1$.



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Unification: Execution

- A perspex program starts execution at the Euclidean origin, $\begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T$.
- A perspex program halts when it is instructed to write to nullity, or to jump to nullity, or to jump by nullity. Nullity is denoted by the zero vector.
- The zero perspex is used as the universal halting instruction.
- The paper describes how to implement a null and an identity perspex machine.

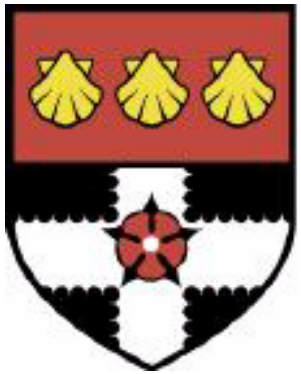


© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Unification: URM

- The Unlimited Register Machine, URM, is known to be turing equivalent. It has four instructions: ground (G), count (C), transpose (T), and advance (A). We will show how to implement these as perspective transformations.

- The paper shows how to use perspex space and perspexes to model the program and registers in a URM.

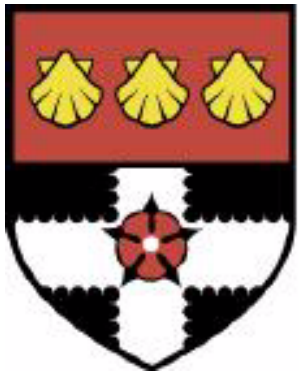


© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Unification: G

- Writing G into the location z zeros the element z_{11} as required by the URM instruction $G(z)$. Hence it is sufficient for the perspex to be of the form $GI \rightarrow \vec{z}$; jump$(\vec{z}_{11}, t)$.

$$G = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

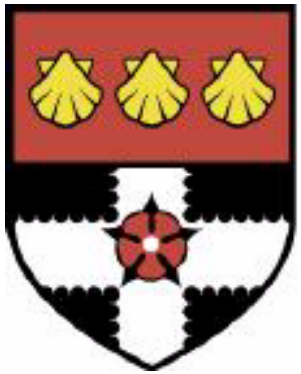


© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Unification: C

$$C = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

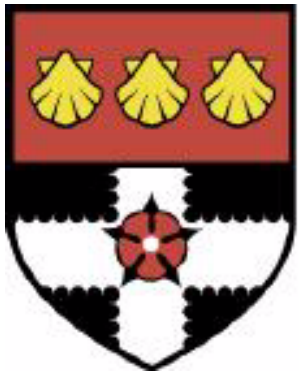
- The URM instruction $C(z)$ increments the contents of register z. The perspex transformation C has the same effect, as can be proved by induction on the following series.



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Unification: C

$$Z^{(0)} = G = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \Rightarrow z^{(0)}{}_{11} = 0$$

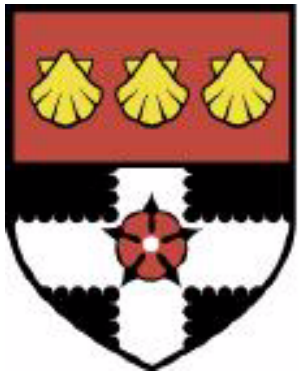


© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Unification: C

$$Z^{(1)} = CG = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} =$$

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \Rightarrow z^{(1)}{}_{11} = 1$$

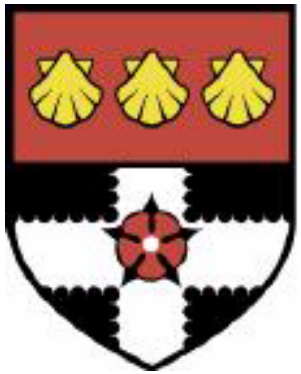


© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Unification: C

$$Z^{(2)} = CCG = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} =$$

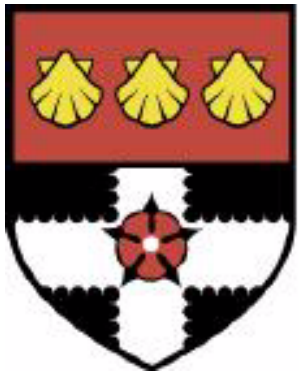
$$\begin{bmatrix} 2 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \Rightarrow z^{(2)}{}_{11} = 2$$



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Unification: T

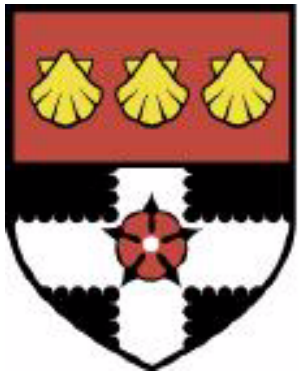
- The URM instruction $T(m, n)$ writes the contents of register m into register n. A sufficient perspex is of the form $\vec{m}I \rightarrow \vec{n}$; jump$(\vec{n}_{11}, t)$.



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Unification: A

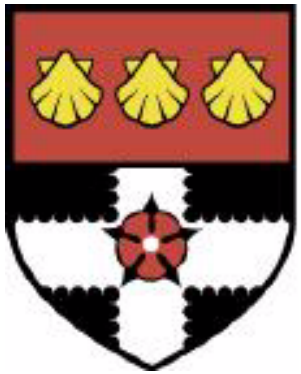
- The URM instruction $A(m, n, p)$ compares registers m and n then jumps to instruction p if $m = n$, but otherwise advances to the next instruction q in the URM program. It is sufficient for a perspex to jump depending on the value of $m_{11} - n_{11}$. This can be done by using a subtraction operation S to compute an intermediate perspex R with the element $r_{11} = m_{11} - n_{11}$.



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Unification: A

$$S = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \end{bmatrix}$$

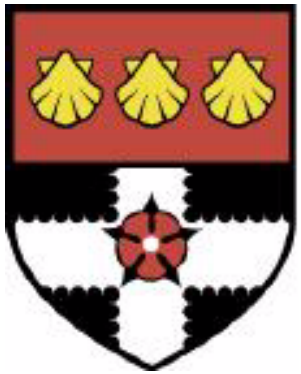


© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Unification: A

$$R = \begin{bmatrix} m_{11} & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} n_{11} & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} =$$

$$\begin{bmatrix} m_{11} - n_{11} & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$



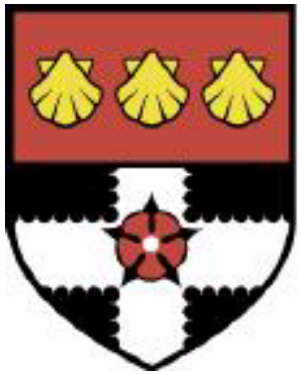
© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Unification: A

- Hence a sufficient perspex program computes R and then performs the jump:

$$RI \to \vec{z}; \text{jump}\left(\vec{z}_{11}, \left[q' \;\; p' \;\; q' \;\; 0\right]^T\right)$$

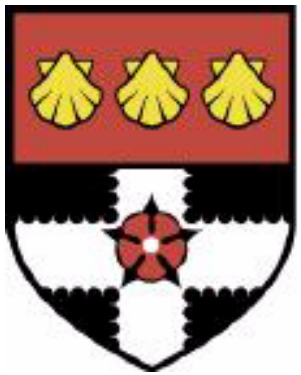
- Here p' and q' are, respectively, the relative jumps from the current location to p and q.



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Unification: Turing

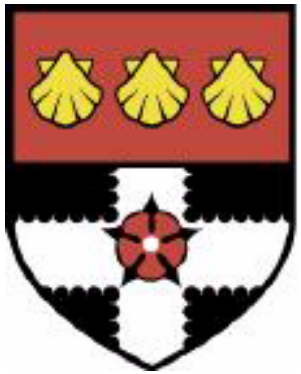
- We have just shown that a perspex machine can do everything that a Turing machine can do. But can it do more?
- Rational matrix algebra is known to be Turing computable.
- The perspex jump instruction can be implemented as a change of state in a Turing machine.
- Therefore the rational perspex machine is Turing computable. Hence it is Turing equivalent.



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Unification: Supra-Turing

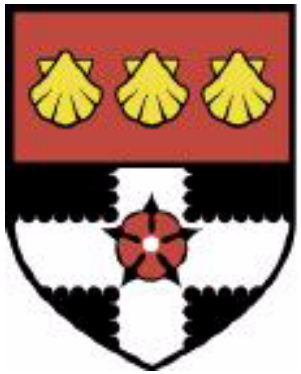
- Operations on irrational numbers are, in general, not Turing computable, but the perspex machine operates on all real numbers, so it is a supra-Turing machine.
- The perspex machine operates as a Turing machine with an irrational number oracle.



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Unification: Projective Geometry

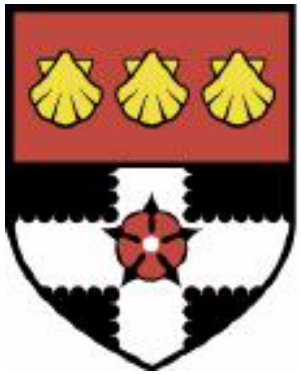
- The rational perspex machine can compute all projective geometry in the homogeneous model using Tarski's algorithm.
- The real perspex machine can compute all projective geometry.
- This completes the unification.



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Optics: Petahertz Computation

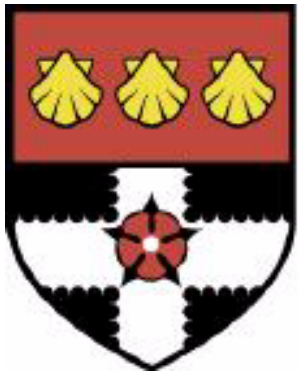
- The perspex machine can be implemented as a pin-hole performing the multiplications and reduction to canonical form, and a stop, mirror, or wave-guide performing the jump.
- Using ultraviolet light, we should be able to use of order 10^7 computing elements per metre.
- Light travels at order 10^8 metres per second.
- This gives $10^7 \times 10^8 = 10^{15}$ instructions per second.



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Anisotropy: Spacetime

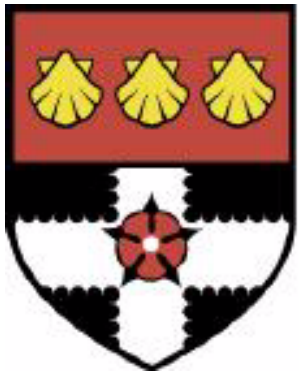
- Perspex programs with $t_4 = 0$ are reversible in the current hyperplane.
- Perspex programs with $t_4 = 1$ are *not* reversible.
- Perspex programs with perspexes in non-canonical form are, generally, continuously variable and reversible.
- So perspex space provides a rich model of anisotropic spacetime.



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Anisotropy: Modularisation

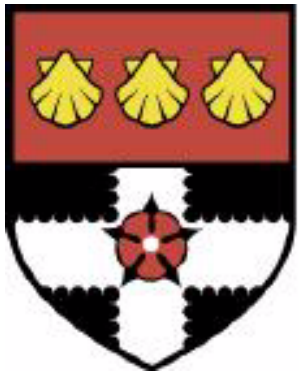
- Execute sub-programs in the current hyperplane, then jump to the successor hyperplane to prevent re-entering the subprogram.



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Anisotropy: Spacetime

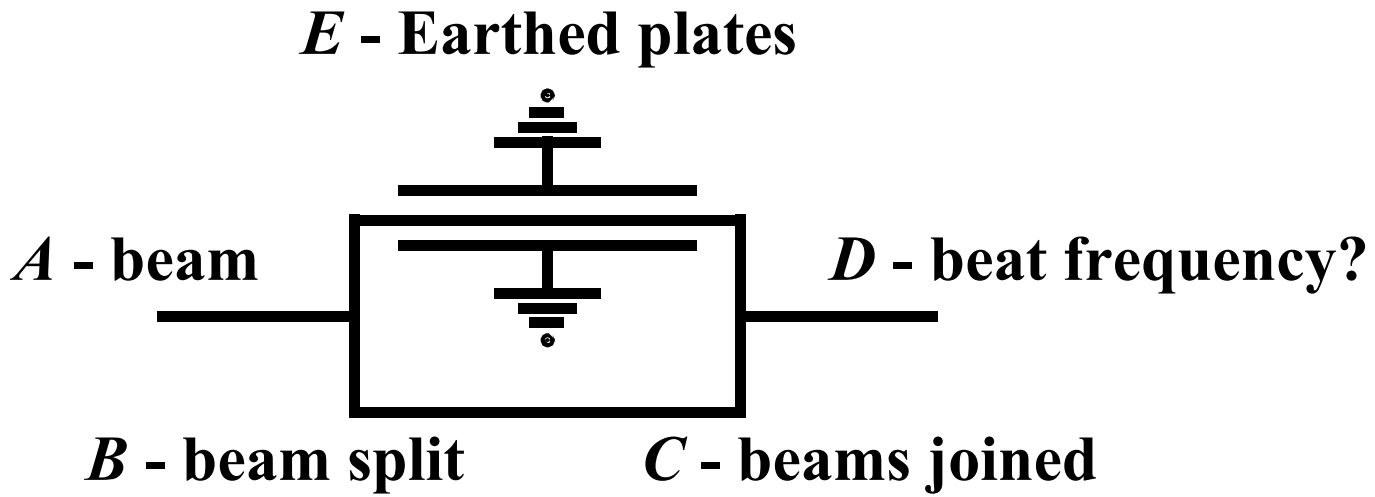
- Hypothesise that the direction of time-flow oscillates in a continuum, analogous to the reversibility of perspex operations within a hyperplane.
- By definition *random* events are not affected by the outcome at any earlier or later time, so random events are not reversible in time.
- By definition, *events* are whole numbered.
- Quantum events are random, so they are whole numbered and not reversible.
- Hence quantum events are analogous to perspex hyperplanes at whole numbered locations.



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Instantaneous Computation

- Test the above hypothesis in a modified version of the Casimir apparatus.
- If the experiment works then we should be able to construct a computer that has inputs and outputs simultaneously present in the elapsed time of the vacuum.

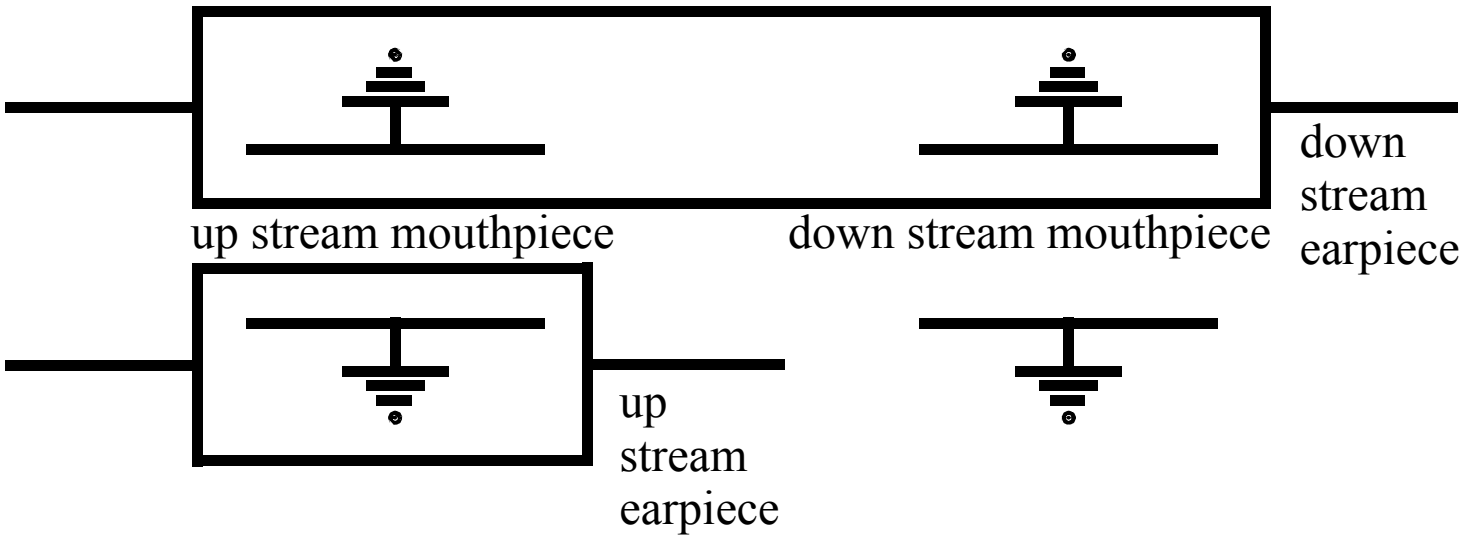


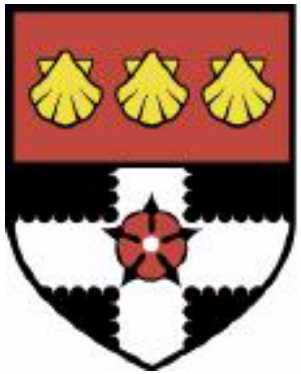
© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Time Telephone

- In this configuration the up stream telephone talks to the down stream telephone in the future. Modulating the plates as a mouthpiece sends a signal in elapsed time.

- The downstream telephone sends a signal to the upstream one via time-flow oscillations.

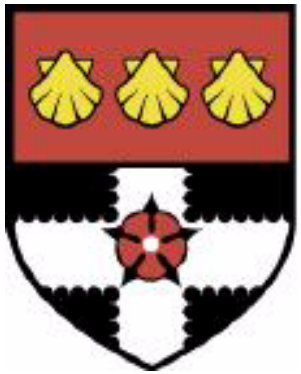




© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com

Conclusion

- Unified projective geometry and the Turing machine.
- Specified a fast, optical computer.
- Developed an anisotropic spacetime.
- Modularised perspex programs in spacetime.
- Suggested how to measure physical time-flow.



© James A.D.W. Anderson, 2002. All rights reserved. Home: http://www.bookofparagon.com