<https://dl.acm.org/doi/10.5555/19899.19958>

[Journal of FORTH Application and ResearchVolume 4Issue 2](https://dl.acm.org/toc/jfar/1986/4/2)June 1986pp 349–352

1986 Rochester Forth Conference

**Signal Space, Address Space, & Symbol Space**

James C. Brakefield

Technology Incorporated, Life Sciences Division

300 Breesport, San Antonio, Texas 78216

Abstract:

Three major engineering disciplines (computer or hardware, software, and knowledge) can be abstracted so that they deal with vector spaces. The three abstract spaces can be mapped inot one another such that concepts in one take another form in one of the others. This leads to insights into the nature of various problems.

Boolean signal space is a variation of the binary hyper-cube. It can be used as a framework for logic design.

Address space is a variation of Forth. It can be used as a framework for the programming of computers.

Symbol space is an approach to the artificial intelligence paradigm. Its intuitive appeal is as a model of thought.

The talk will describe each of these in greater detail, show the mapping from one to another, and show some problems that benefit from the recasting of their context.

1. Binary Signal Space

Binary signal space is a collection of binary inputs, outputs, and intermediate signals considered as a whole. Usually each is consider an orthogonal axis giving rise to the hyper-cube. This is the usual starting point for logic circuit design. My concern is with the entire system, such as a complete computer with all its memory modeled as gates. Thus the area of interest is thousands, millions, or even billions of signals.

Some questions can be asked in the presence of such large numbers: is it possible to fully specify the complete truth table with so many signals? How does one know a particular circuit is correct? What happens to the circuit as truth table entries are added? What happens to the circuit when signals are added?

My interest is mapping the pattern recognition and learning situations into logic design. Each pattern or learning example is a single truth table entry. Any mechanism that “learns” is then doing circuit design where entries and/or signals are added one at a time to the truth table. The truth table entries can be thought of as labels of the vertices of the hyper-cube.

In such a circuit design environment the truth table is extremely sparse. With only one hundred sitnals there are 2\*\*100 or 10\*\*30 entries. Evan a million entries (say 10 by ten binary rester images) is an insignificant portion of the hyperspace. These unfilled entries are essentially “don’t cares”. The circuit design software or hardware assigns them so as to simplify the circuit.

1. Address Space

Address space is an abstraction of the programming paradigm. Forth is very suitable for this. Programs consist of address strings referencing either other address strings or primitives. The interpretation of the strings is sequential whereas the signal space merely computes a function.