A Novel Implementation Style for Teaching VLSI.

Tom Kean	Genbao Feng	Irene Buchanan	John Gray
Algotronix	Algotronix	Algotronix	United Silicon Structures (US2)

Abstract

VLSI technology presents a dilemma for Universities: while it is vital that students are educated in it use it can be both difficult and expensive to provide this training. Most of the problems are caused by practical difficulties with access to silicon. This paper presents an electrically programmed cellular structure whose properties at the systems level are very similar to those of silicon and which can be used to implement both simple and complex designs. This allows inexpensive designs with immediate turnaround and automatic conversion to more established implementation styles, such as full-custom silicon.

1 Introduction.

VLSI technology presents a dilemma for Universities: while it is vital that students are trained in its use it can be both difficult and expensive to provide this training. There are many reasons for this: a few of the most important are outlined below.

1. Fabrication Expense and Availability. Chip fabrication is generally expensive and unavailable in the format and timescale required by educational institutions (although there are some notable exceptions, for example, ES2 in Europe and MOSIS in the US).

2. High Cost of CAD Tools. The high cost of commercial CAD tools and the computers to run them, coupled with the relatively long learning curves required to master them pose a major problem in the educational context. Typically, insufficient workstation 'seats' will be available or inadequate design software will be used according to where the price/performance compromise has been made.

3. Simulation. Simulation of even moderately complex VLSI designs is extremely computer intensive but is absolutely necessary if chips are to have a reasonable chance of being functional after fabrication. Provision of adequate computer resource for even minimal simulations by a large number of students will often be impractical for all but the smallest designs.

4. Short Time for Practical Exercises. Normally, practical exercises involving chip design cannot be assigned until relatively late in a VLSI course since considerable information must be assimilated by the students before they can begin the project. It is also often necessary for the project to be completed before the end of the course; this may leave only 2 or 3 weeks for

the student to complete the practical. Thus interesting projects of reasonable complexity cannot be assigned.

5. Long Delay in Receiving Fabricated Chips. It is not uncommon for it to take several months for fabricated chips to be returned and this timing is often unreliable (although, again, there are exceptions to this rule and direct write electron beam technology can allow rapid turnaround). The students will have moved on to other courses by this time and it may be impossible to find time in their schedules to achieve design closure by testing.

6. Small Silicon Areas. Multiproject chips are often area limited to such an extent that only trivial functions are possible.

7. Foundry Interface. There is a large administrative load in organising design files for the foundry given CIF for the student's design. Packaging, layer naming (students usually design with a simplified process model without implant or well layers which must be generated prior to fabrication and may use different layer names from those required by the foundry), design rule checking (normally a final DRC is run on a flattened version of the whole chip after implant layers have been added) and pattern generation all complicate this handover.

Given these problems it is probably time to reexamine the educational goals of VLSI design courses (and perhaps hardware design courses in general). Geometric level design is only one facet of the subject and while it is important that students understand the physical medium, it is also important not to unbalance the syllabus by an over-reliance on physical design and chip fabrication. In many VLSI courses the primary aim is to communicate high-level concepts such as timing methodologies and hierarchical design. However, the level of 'practical' problems encountered in real-life VLSI design is such that student projects must be stripped of educationally valuable conceptual problems if they are to be completed in the available time.

This paper proposes a technical solution to the problems of teaching digital VLSI design: substitute a novel EPLD architecture for silicon as the target technology. The EPLD is sufficiently close to silicon to allow silicon design techniques and CAD tools to be used but is electrically reprogrammable allowing instant turnaround. The greatest advantage of this technique is that since the students no longer need to waste time on less relevant details they can design systems of realistic complexity in their practical exercise and can achieve immediate design closure including testing. Section 5 of this paper contains a small but detailed design example: a digital stopwatch. More complex design examples including DES encryption, image pattern matching, a cellular automaton machine for fuid-flow simulation and systolic string matching are presented in [1],[3],[4],[5].

2 Configurable Logic.

The proposed EPLD architecture is termed Configurable Array Logic (CAL) and is a classic cellular array design [2] (figure 1). The array structure is extensible over chip boundaries transparently to user designs so multi-chip 2D computing surfaces can be built to emulate the largest silicon designs. Each cell in the array communicates with its nearest neighbours and provides one two-input logic gate or latch as its function.

As well as the local interconnect 3 global signals are provided: the first two (G1 and G2) are inputs to all the cells in the array and are intended to be used as a two-phase non-overlapping clock. These signals are often used to clock pipeline registers implemented using the cell latch function. The third global signal (FTEST) is an output which can selectively be connected to the output of any cell's function block allowing internal signals within user designs to be monitored.

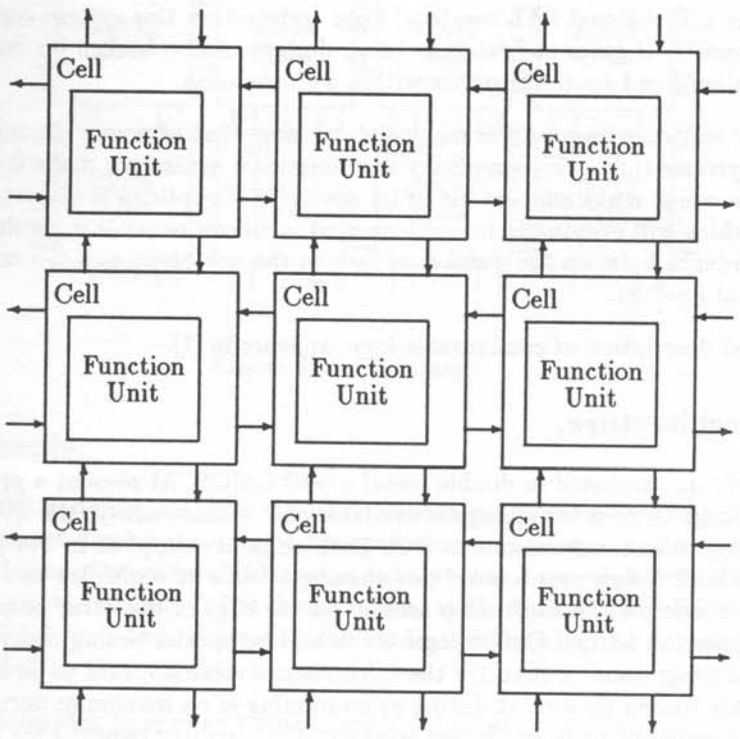

Figure 1: Basic Structure.

The low cost of latches and global clocks make the technology particularly suitable for the implementation of pipelined and systolic array algorithms. Therefore, from a teaching point of view, CAL's can be used in a range of courses from simple digital circuits to advanced parallel processing architectures ([1],[3],[4],[5]).
This architecture has several advantages over current dynamically programmable logic families in the educational context.

1. The system has an interface to VLSI design tools [7] allowing designs done using the cellular array to be converted automatically to silicon implementations. This allows the electrically programmable technology to be used to prototype systems which will eventually be implemented in silicon.

2. Unlike the conventional PAL two level logic architecture this system supports arbitrary interconnection of gates and latches. Large designs can be broken up into subunits which can then be placed and routed together within a single array.

3. The basic cell's function unit is much simpler than that of recent electrically programmable cellular systems ([9]). This simplicity and functional symmetry make it easier to write higher level tools which make efficient use of its resources. Simplicity is important when prototyping designs which will eventually be implemented in silicon because it results in a reasonable correspondence between the number of cells in the prototype and the area of silicon required in the final product.

A more detailed description of configurable logic appears in [1].

3 Chip Architecture.

The CAL chip is implemented in double metal n-well CMOS. At present a prototype chip with a 16x16 array of cells in $2\mu m$ technology is available and a 32x32 array (the CAL1024 or kCAL) in $1.5\mu m$ technology is under development [10]. Both chips are supplied in 144 pin Pin Grid Array packages, the 32x32 design uses a novel pad sharing scheme to multiplex two array inputs and outputs onto a single pad for multi-chip arrays. At the edge of the array communication with the CAL chips happens at normal CMOS logic levels and no special timing constraints are imposed. To the external programming circuitry the CAL control store appears to be a long static shift register, the only timing constraint during programming is on maximum data rate. This interface minimises the number of pads and board level support circuitry required for programming: typically configuration data will be downloaded using an RS232 or Centronix interface. A future version of the CAL chip will have a 'memory-like' programming interface allowing random access to the configuration store.

4 The System Architecture.

A configurable array logic system consists of one or more CAL chips connected together along their boundaries to produce a two dimensional surface (figure 2).
With this arrangement a 4x4 array of CAL1024 chips yields a 128x128 array of cells i.e. 16K cells. With a system of this size it is possible to implement a number of interesting parallel algorithms. A smaller 2x2 array of CAL chips will fit on a single PC/AT card and can implement dmedium complexity designs at relatively low cost.

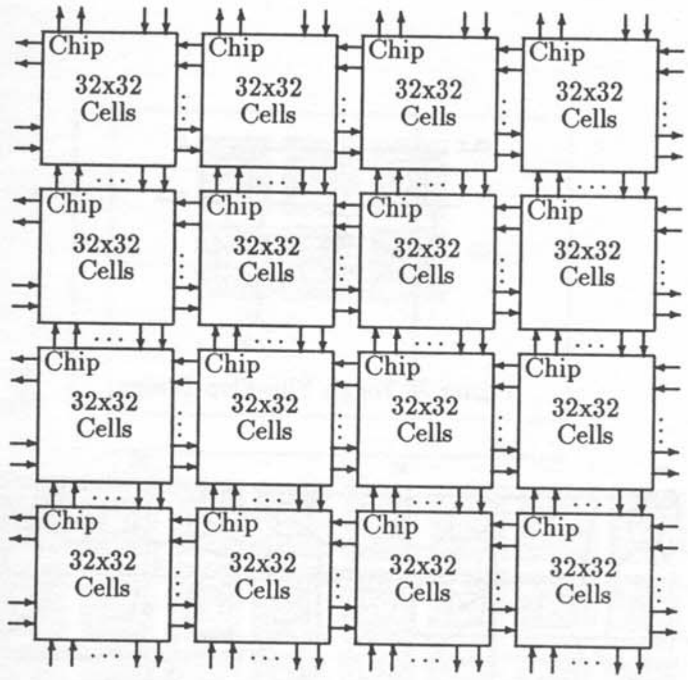

Figure 2: CAL System.

5 Design Example.

This section covers the design of a simple digital stopwatch designed to count up in tenths of a second to one minute and display the current time on three seven segment displays (tenths of seconds, seconds and tens of seconds). This design example has been used for several years as the practical exercise in an introductory VLSI design course at the University of Edinburgh [6]; student designs are fabricated and returned to the students for testing the following term.
The watch is controlled by three signals:

INIT Clears the stopwatch to zero and puts it in the 'stop' state.

SS Start/Stop A high going edge in the 'stop' state starts the watch counting. A high going edge in the 'start' counting state puts the watch in the 'stop' state.

CLOCK 10Hz clock signal.

The seven segment decoders provide a good example of 'random' combinational logic while the counters provide a good example of classical sequential logic. This design uses three separate units for the three displays although the design complexity and area could be reduced by implementing it as a single state machine with multiplexed displays (since the large seven segment decoder logic would not need to be duplicated). This is necessary to comply with the design specification given to the students in order to make a fair comparison between design styles.

The Counter. The counter is built from 4 toggle flip-flops (Figure 3). Each large box on the diagram represents one cell. The small box in the centre represents the logic gate. Its two inputs

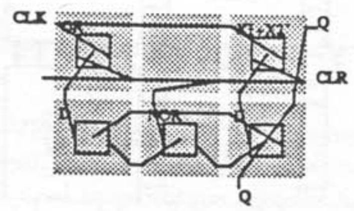

Figure 3: Toggle Flip-Flop Design.

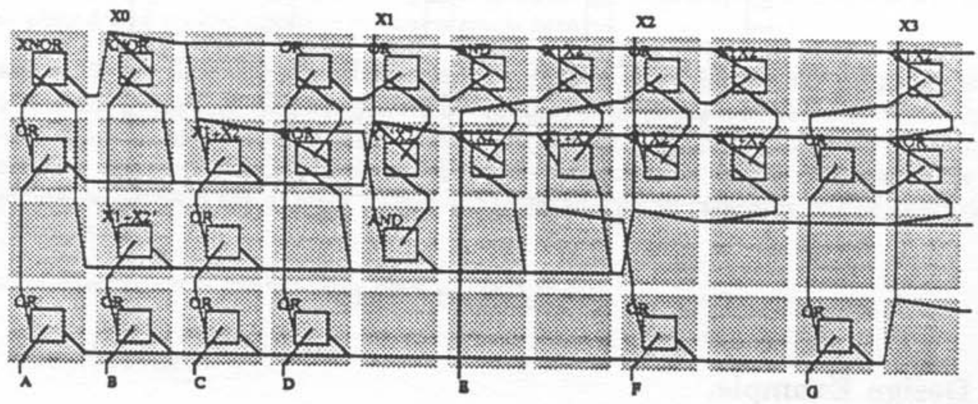

Figure 4: Seven Segment Decoder Design.

are half way down the left (X1 input) and right (X2 input) sides of the box and its output comes from the centre of the box. The particular logic function being implemented is printed in the top left. The lines within the box represent connections. The basic 4 bit ripple counter is converted to a decade counter by a gate which sets the clear line when the counter gets to ten. The output of this gate is also used as the clock for the succeeding counter. The counter can also be cleared by the user's $INIT$ signal.

Providing the clear capability was the only real problem in the design of the toggle flip-flop (the basic D latches have only clock and D inputs). The clear is provided by extra logic gates which force 0's onto the D inputs and 1's onto the clock inputs. A basic master-slave flip-flop can be built with only two D latch cells whereas this clearable implementation requires six cells.

The Decoder. The decoder (figure 4) takes advantage of the ability to produce any function of two boolean variables within one cell and uses several levels of logic rather than the two level logic normally used to implement such functions.

The Control Logic. The controller function (at the bottom left of figure 6) is implemented using a toggle flip-flop. The design is the same as those within the counters. The toggle flip-flop is clocked by the start-stop input and its output determines whether the counter should be stopped or run freely. The initialise signal clears the control flip-flop and stops the counter. The counters

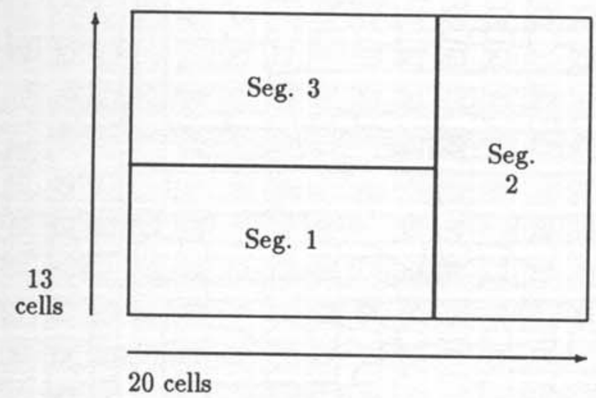

Figure 5: Stopwatch Floorplan.

are stopped by AND'ing the 10Hz clock with the output of the control flip-flop: thus when the output of the control flip-flop is 0 the counters' clock input is held low and when it is 1 the counters receive the 10Hz clock input.

Floorplan. The floorplan of the full watch circuit is given below as figure 5 and the layout in figure 6. It requires a 20 by 13 array of cells.

Size Comparisons. The digital stopwatch example has been designed using several different styles: CAL, PLA and multiple mask change 'optimised' (because wiring channels and logic areas are only as large as they need to be - the detailed chip floorplan is specific to a particular design) gate array. The optimised array designs were done by European Silicon Structure's SOLO-1000 system [7] using an input specification (in the MODEL language) generated automatically from the cell based design. This compiler has had many years of development and produces very high quality layouts for this kind of design: in this case it can also take advantage of the efficient gate level design done for the configurable logic implementation. The figures for this design are, therefore, better than average for this implementation style. The PLA designs were done by students as a practical exercise and have been fabricated using lambda rules with $\lambda = 2\mu m$ (i.e. $4\mu m$ technology). They are composed of four PLA's one for each of the segment units and one for the controller. The CAL and gate array designs were done using the same commercial $2\mu m$ rules. The areas given are core symbol sizes (no pads), the CAL figures are for a 20 by 13 (logical) chip. The results are given in Table 1, in this case there is a factor of about 15 in area between the CAL implementation and the optimised array: special processing in the CAL design could significantly reduce this gap. Although a factor of 15 seems high it suggests that any gate-level design which will fit on a single ASIC could be emulated using a board full of configurable chips.
It is interesting to note that a single 32x32 CAL chip can contain about 2.5 times as much logic as the multiproject chip designed by the students: the reason for this is that the much larger silicon area available and the better processing technology have more than compensated for the overhead of the programming circuitry. Typically between 4 and 16 CAL chips will be used in an array allowing large systems to be implemented. If necessary standard memory or analogue parts can be

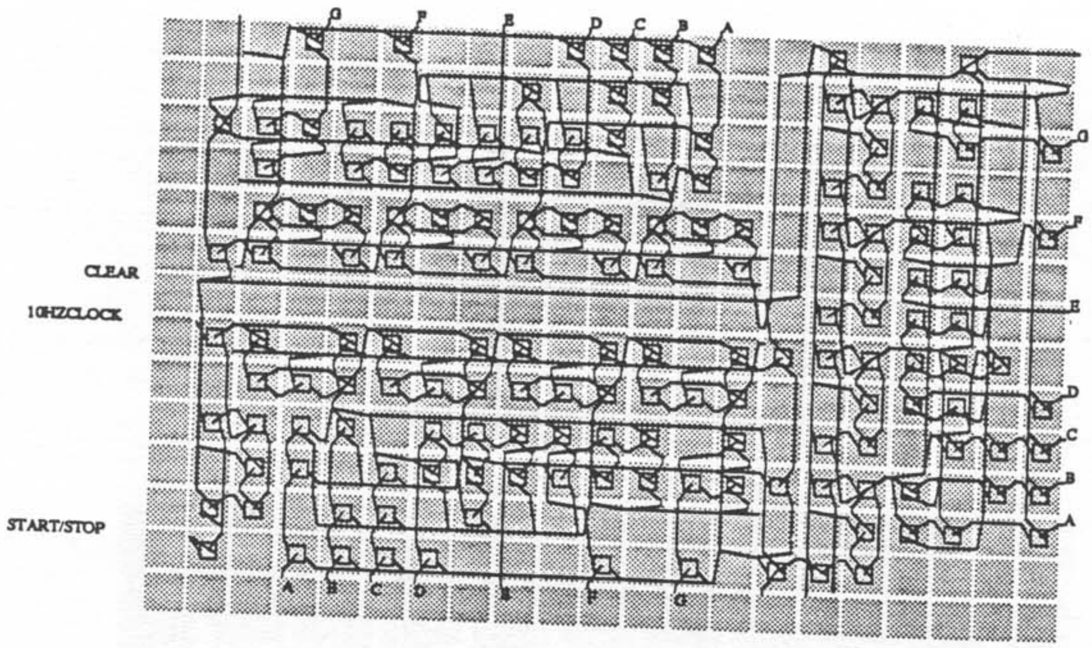

Figure 6: Full Stopwatch Design.

Design.	*Technology*	*Dimensions*
CAL	$2\mu m$	$5777 \times 3786\mu m$
PLA	$4\mu m$	$1432 \times 1625\mu m$
Gate Array	$2\mu m$	$1544 \times 952\mu m$

Table 1: Design Size Comparison.

added to a board level prototyping system to allow designs with special requirements to be emulated.

6 CAD Tools for Configurable Logic.

The CAL chips regular layout structure means that CAD tools are straightforward to implement. A CAD system for Configurable Logic has been written in Standard ML [8] and has been used to design several large systems [3, 5, 4]. The system consists of the kind of components familiar to users of silicon design tools: for example, logic synthesisers, channel routers and graphical editors. This system is currently being ported to C to support user designs on the commercial CAL chips.

7 Conclusions.

The CAL is a novel cellular array architecture which can provide an attractive alternative to silicon implementation in VLSI design practical courses. The CAL is not only cheaper and easier to use but also allows more significant projects to be undertaken. A CAD system for CAL is under development which provides a similar environment to standard silicon CAD systems.

References

[1] Tom Kean. *Configurable Logic: A Dynamically Programmable Cellular Architecture and its VLSI Implementation.* PhD Thesis. University of Edinburgh, Dept. of Computer Science, 1989.

[2] R.C. Minnick. *A Survey of Microcellular Research.* J. ACM, 14(2):203–241, April 1967.

[3] J.P. Gray and T.A. Kean. *Configurable Hardware: A New Paradigm for Computation.* To Appear in Proc. Decennial Caltech Conference on VLSI, Pasadena CA, March 1989.

[4] Jouko Viitanen and Tom Kean. *Image Pattern Recognition using Configurable Logic Cell Arrays.* To Appear in Proc. Computer Graphics International, Leeds UK, June 1989.

[5] T.A. Kean and J.P. Gray. *Configurable Hardware: Two Case Studies of Micrograin Computation.* To Appear in Proc. International Conference on Systolic Arrays, Killarney, May 1989.

[6] David Rees. *MSc VLSI Design Course Lecture Notes.* University of Edinburgh, Dept. of Computer Science, 1987.

[7] European Silicon Structures Ltd. *SOLO 1000 User Manual.* Bracknell, UK., 1988.

[8] Robert Harper, David MacQueen and Robin Milner. *Standard ML.* Technical Report CSR-209-86, University of Edinburgh,Dept. of Computer Science, 1986.

[9] Xilinx Inc. *The Programmable Gate Array Design Handbook,* San Jose, CA., 1986.

[10] Algotronix Ltd. *CAL1024 Preliminary Data Sheet.* Edinburgh UK, 1989.