An FPGA Family Optimized for High Densities and Reduced Routing Delay

Mike Ahrens, Abbas El Gamal, Doug Galbraith, Jonathan Greene, Sinan Kaptanoglu
K.R. Dharmarajan, Lynn Hutchings, Sifuei Ku, Phil McGibney, John McGowan, Amer Samie
Kitty Shaw, Norma Stiawalt, Telle Whitney, Tom Wong, Wayne Wong, Bortay Wu

Actel Corporation
955 E. Arques Ave.
Sunnyvale, California 94086

ABSTRACT: The Act-2 family of CMOS Field-Programmable Gate Arrays uses an electrically programmable "antifuse" and new architectural and circuit features to obtain higher logic densities while increasing speed and routability. Improvements include: two new logic modules, novel IO and clock driver circuitry, and more flexible and faster routing paths. New addressing circuitry shortens programming time and speeds complete testing for shorts, opens and stuck-at faults. Fully automatic placement and complete routing are retained. Special software tools used for architectural exploration and layout generation are noted.

1. Introduction.

Previous papers described an architecture for field-programmable gate arrays (FPGAs) [1], and its implementation in the Act-1 FPGA circuits [2]. These demonstrate that user programmability can be obtained without sacrificing the application flexibility of a channeled gate array architecture.

This paper describes new architectural features, circuit techniques and software that approximately double system speeds, and are capable of extending the architecture to logic densities of 8,000 gates in 1.2 micron technology and to approximately 16,000 gates for 0.8 micron. (Note that these gate counts are based on the capacity of an equivalent mask-programmed gate array. Other measures would yield higher values.) The circuits employ a one-time electrically programmable "antifuse" offering small area and capacitance, and low resistance once programmed [3].

As before, the architecture consists of rows of logic modules separated by horizontal channels. This organization is similar to that of a channeled gate array, except that instead of an area for custom metallization the channels contain wiring segments of various lengths which can be connected by antifuses.

A key goal was to insure complete automatic placement and routing with acceptable routing delays. This is facilitated by the inherent flexibility of the channeled architecture and the integration of large numbers of antifuses (700,000 or more) on a single chip.

2. Logic Module

The choice of the logic module is critical to an FPGA architecture. The module must be simple enough to permit a compact and high-speed circuit layout. Yet it must also be flexible enough to accommodate the most frequently used logic functions (macros) with several choices of routing. Our approach is to evaluate many candidate modules against macro usage statistics from actual applications. (The philosophy is similar to that used to define the instruction set of a RISC microprocessor. It has also recently been applied to BiCMOS gate arrays [4].) To assist in this task, a program has been developed that can enumerate all macros accommodated by a given module in minutes [5].

The Act-1 family uses one general-purpose module, which implements all combinational functions of 2 inputs, many of 3 or 4 inputs, and others ranging up to 8 inputs [1]. Any sequential macro can be configured from one or more modules using appropriate feedback routings.

At higher logic densities, the law of averages makes designs begin to adhere more closely to typical macro usage statistics (see, e.g., [4]). This motivates the use of a mix of two new modules, each of which is most efficient for a different set of macros. The "C-module" is a modified version of the Act-1 module reoptimized to better accommodate high-fan-in combinational macros, e.g. wider AND gates, though with some loss in ability to accommodate sequential functions. The S-module, on the other hand, is optimized for configuring sequential macros. It can accommodate a latch or flip-flop and/or many combinational macros of one to seven inputs. Both transparent-high and -low latches and rising- and falling-edge-triggered flip-flops are possible.

The two-module scheme can reduce the number of modules required for a block of logic by up to a factor of 3. On average, logic density per module is increased by over 50%. Furthermore, because the density is increased, the number of routed nets in a typical critical path is reduced. This significantly improves speed. Fig. 1 shows how a typical critical path in a state machine can be implemented to take advantage of the wide fan-in of the C-module, and the capabilities of the S-module. The delay paths include only two routed nets. Performance data is summarized in Table 1.

Since the fan-in of each module is no larger than that of a typical gate-array macro, the two-module scheme maintains

CH2860-5/90/0000-0166 © 1990 IEEE

the generality of a "fine-grained" architecture. Significantly larger and more specialized modules would risk a sharp loss of efficiency for applications that deviate from typical usage statistics. Using a larger module, or more types of modules, also adds constraints to the placement and routing problem, making automatic solution more difficult and ultimately increasing net delay.

3. Input/Output

Of particular importance to system performance is the delay between the time a clock signal changes at an input pad and when data appears on an output pad, referred to as T_{clk-Q}. (Memory bus interface applications are a good example). The goal is to gain maximum speed without sacrificing flexibility.

This is accomplished by providing a dedicated transparent-high latch in each output path. If desired, the dedicated latch can be combined with a transparent-low latch configured from a logic module to form a rising edge-triggered flip-flop. (Note that the net connecting the two latches is not in the critical path, so T_{clk-Q} is not increased relative to having a dedicated flip-flop in each IO.) If flow-through operation is desired, the output latch gate is simply tied off to make the latch transparent.

To limit set-up time requirements, a dedicated transparent-low latch is provided on each input path. The polarities of the input and output latches are chosen so they can be combined with each other, and possibly with other internal latches or flip-flops, to form a path that is functionally equivalent to a chain of rising-edge flip-flops. (See Fig. 2.).

Chips with many simultaneously switching outputs require some form of slew rate control to avoid noise problems; several alternatives are possible. Sequencing the operation of several parallel drivers limits the slope of the current ramp when driving a passive load, but large di/dt can occur in bus contention situations when the contending driver suddenly shuts off. Feedback remedies this problem, but can still allow large di/dt in asynchronous systems where the logic state changes before a transition is complete. Instead a current mirror circuit was used to limit the drive current. This results in lower di/dt noise in worst case situations, a simple way to implement programmable slew rate, and 90% power efficiency. The output buffer meets the 4mA HCT buss driver specification for AC, and the 6mA specification in steady state when the current limit shuts off. ESD protection is >2000V.

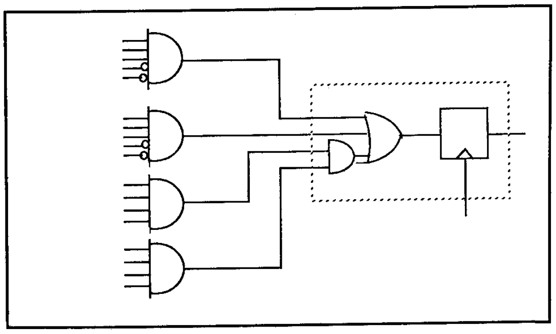

Figure 1: part of a state machine implemented in four C-modules and one S-module.

Connections between the array and the IO pads are made via special IO modules interspersed with the logic modules. The IO module has inputs for data, slew control, tristate, and separate gates for the input and output latches. The gate inputs are not restricted to a dedicated clock signal, but may each be driven from any pad or internal net.

4. Clock Distribution

Clock distribution is a problem in most large chips. In an FPGA, where the load capacitance may be changed or redistributed to suit each application, it is a greater challenge.

Special distribution networks are provided to deliver high-fanout clock signals to the inputs of any logic or IO module with minimal skew. Each network may be driven directly from an input pad for high speed, or from user-defined internal logic. High speed and low power are obtained by a distributed driver with 90% power efficiency.

Skew is further reduced by automatic placement algorithms that balance the loading on each branch of the distribution tree.

All clock inputs may also be routed in the normal way instead, allowing many local asynchronous clock signals if desired.

parameter	nsec
module input to module input (critical net):	7-8
setup+hold time (module used as flip-flop):	< 6
input pad to IO module output:	5-7
IO module input to output pad:	8
clock distribution net skew:	< 5
in-circuit probe delay (module to pad):	15-20

Table 1: Performance Estimates.
(1.2 micron CMOS, typical process, 5 volts, 25 C).

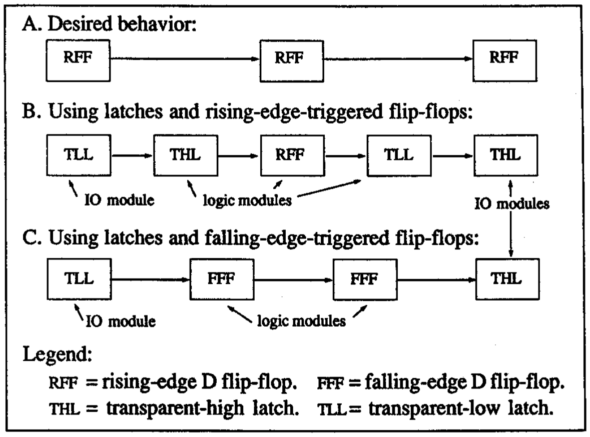
A. Desired behavior:
RFF
RFF
RFF
B. Using latches and rising-edge-triggered flip-flops:
TLL
THL
RFF
TLL
THL
IO module
logic modules
IO modules
C. Using latches and falling-edge-triggered flip-flops:
TLL
FFF
FFF
THL
IO module
logic modules
Legend:
RFF = rising-edge D flip-flop. FFF = falling-edge D flip-flop.
THL = transparent-high latch. TLL= transparent-low latch.

Figure 2: IO clocking--three equivalent implementations.

5. Routing Architecture

Each routing channel contains horizontal tracks divided into segments of various lengths [1]. Surprisingly, the restriction to segments of predefined length does not greatly increase the number of tracks beyond what would be required in the unrestricted case of mask-programmed channels [6].

In an efficient architecture it is inevitable that some nets' routings will be slower than others. The use of a low resistance switch, such as the antifuse, helps to narrow the resulting delay distribution. Further improvements have been obtained by a reduction in the maximum number of antifuses in the worst delay paths, as follows.

In the vertical direction, most nets are routed using a short dedicated segment connected to the module's output driver through an "isolation" transistor (Fig. 3). (The transistor isolates the module circuitry from programming voltages present on the segments). In this case, there are only two antifuses plus the isolation device in the path from the buffer to each input (input A in the figure).† Though this favorable routing can be assured for speed critical nets, generally some 5-10% of the other nets must be placed with an input in some channel beyond the span of the dedicated segment (input B, Fig. 3). In the past, this required use of an uncommitted vertical segment and 4 antifuses.

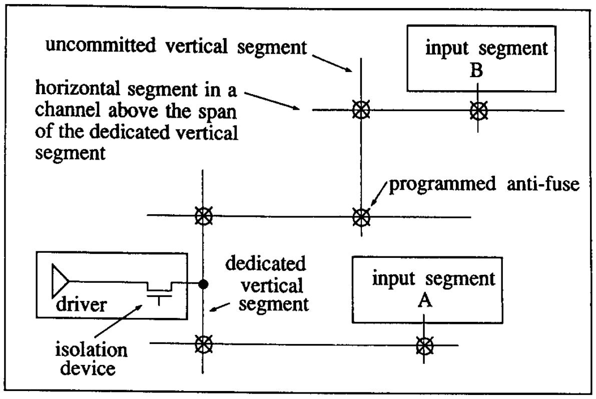
uncommitted vertical segment
input segment B
horizontal segment in a channel above the span of the dedicated vertical segment
programmed anti-fuse
dedicated vertical segment
input segment A
driver
isolation device

Figure 3: Act-1 Routing

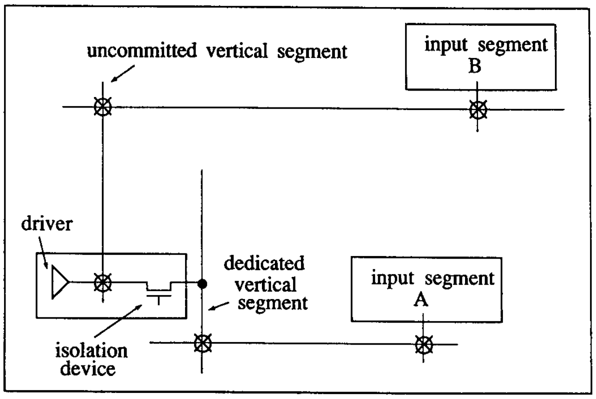

Figure 4: Act-2 Routing

Alteration of the order in which antifuses are programmed and a robust driver circuit permit limited programming of antifuses on the node connecting the driver to the isolation device, without risk of device breakdown [7]. This allows direct connection of the driver to any of several uncommitted vertical segments, as shown in Fig. 4. Since the additional antifuse presents little more resistance than that of the bypassed isolation device, the delay of these nets is not much greater than those using dedicated segments. Prediction of delays prior to placement (when it is not yet known which nets require uncommitted segments) becomes more accurate as well.

Segmented channels represent an unusual layout challenge. They are as dense and large as a memory array, yet not repetitive. (A carefully chosen but irregular mix of segment lengths is provided for good routability.) For this reason, a layout generation program was developed that assembles the channels and modules automatically from the same database used by the routing software. This permits rapid layout of a family of arrays of various sizes by simply rerunning the generator with the appropriate input files.

6. Placement and Routing Software

Several new complexities are added to the placement optimization problem. Macros must be placed in modules of the appropriate type (C or S). Macros hooked to a clock network should be distributed so as to balance the load on the network's branches. There should not be excess demand for uncommitted vertical segments within the same column. Speed critical nets should be routed using only short horizontal segments and dedicated vertical segments.

Nevertheless, new algorithms make it possible to satisfy all these constraints. Nearly all designs with module utilization under 85%, and most designs with utilization under 95%, route without manual intervention. Table 2 summarizes results for several applications. Time for complete placement and routing is about 45-60 minutes on a 68030-based workstation.

7. Programming and Testing

The time required to program an antifuse falls exponentially with the applied voltage. To keep programming time under 5-10 minutes for a chip with nearly a million antifuses, new circuit designs were developed that eliminate the threshold voltage drop along the path from the chip's supply pad to the antifuse being programmed.

Changes have also been made in the addressing circuits. The pass transistor scheme described in [1] is appropriate for cases where there are many short segments in a track.

† Two adjacent horizontal segments in the same track may be connected end-to-end by an antifuse to form a longer segment [1]. For good routability it is necessary to route some small percentage of the nets in this way [6], which adds an antifuse to the path. However, speed critical nets are routed without this additional antifuse.

However in larger chips the number of horizontal segments per unit area decreases to the point that it is possible to address each individual segment directly using only a small proportion of area for the addressing circuitry [7]. The reduction in the number of pass devices in the programming path improves the programming current and lowers the resistance of programmed antifuses, improving performance. The pass transistor scheme is still used in the vertical direction where tracks are highly segmented.

Direct addressing also reduces the time required to test for breakdown of defective unselected antifuses during programming. A complete test for unintended connections between any two segments can be done after the conclusion of programming (despite the fact that it is not possible to uniquely address each individual antifuse once programming commences). The number of vectors required is only logarithmic in the number of nets. Previously, the test for shorts required one or more vectors after each antifuse is programmed.

Proper closure of a programmed antifuse is confirmed by the passage of the programming current. Note that this complete testing for shorts and opens, combined with exhaustive testing of each logic and IO module prior to programming, is more thorough than even a so-called "100% stuck-at fault coverage" test done on a conventional gate array.

Once programming and testing are complete, no increase in resistance of a programmed antifuse or false programming of an unprogrammed antifuse have been observed in 1.8 million accelerated burn-in device-hours [8].

8. Other Circuit Improvements

The Act-1 and Act-2 architectures allow user selection of any internal logic signal for presentation at a "probe" pad. This allows real-time external observation of each net as the chip operates in a system (similar to an in-circuit emulator for a microprocessor). Use of a sense amp circuit greatly increases the speed of the in-circuit probe path.

Another challenge is to keep the gates of thousands of isolation devices pumped to a high voltage during normal operation. A rapid, high-power pump operates when the chip turns on. It is then shut down when the desired voltage is reached and a low-power sustainer pump takes over. The required standby current is under 300uA.

Acknowledgements

The authors acknowledge contributions by: Rick Wilkenson (layout); Sam Beal, Andy Haines, Dennis McCarty, Bob Osann (applications); Gregg Bakker, Steve Chiang, Shafy Eltoukhy, Esmat Hamdy, John McCollum, (technology); Sanko Lan, Justin Reyneri (logic) and Jeff Schlageter for his support. Sample designs were contributed by Texas Instruments, Data General, many Actel customers, and the EE 218 class at Stanford University.

References

[1] A. El Gamal, J. Greene, J. Reyneri, E. Rogoyski, K. El-Ayat, and A. Mohsen. "An Architecture for Electrically Configurable Gate Arrays." IEEE J. Solid-State Circuits, Vol. 24, No. 2, April, 1989, pp. 394-398.

[2] K. El Ayat, et. al. "A CMOS Electrically Configurable Gate Array." IEEE J. Solid-State Circuits, Vol. 24, No. 3, June, 1989, pp. 752-762.

[3] E. Hamdy, et. al. "Dielectric Based Antifuse for Logic and Memory ICs." IEDM Tech. Digest, San Francisco, CA, 1988, pp. 786-789.

[4] A. El Gamal, J. Kouloheris, D. How, M. Morf. "BiNMOS: A Basic Cell for BiCMOS Sea-of-Gates." Proc. IEEE 1989 Custom Integrated Circuits Conf., page 8.3.1.

[5] S. Lan, J. Reyneri, J. Greene, A. El Gamal. "An Automatic Function Generator for Field Programmable Gate Arrays." In preparation.

[6] J. Greene, V. Roychowdhury, S. Kaptanoglu, A. El Gamal. "Segmented Channel Routing." Submitted for publication.

[7] A. El Gamal, J. Greene, J. Reyneri. "Programmable Interconnect Architecture." US Patent 4,873,549.

[8] S. Chiang, R. Wang, J. Chen, K. Hayes, J. McCollum, E. Hamdy, C. Hu. "Oxide-Nitride-Oxide Antifuse Reliability." Int'l Reliability Physics Symp., March 1990.

design	logic module utilization	pins per logic module
a) design done in an 8K gate TI gate array	99.5 %	4.28
b) 32 bit datapath, 16x16 mult, state machine	99.4	4.30
c) 2901 ALU (x4)	98.1	4.57
d) DMA controller (x3)	97.1	3.99
e) asynchronous serial ECC	97.0	4.78
f) pipelined fixed point mult, div, sqrt	94.5	3.37
g) state mach, mult/add, datapath, counter	92.7	4.34
h) color crt controller (x3)	87.3	3.83
i) 32 bit datapath w/ sum, compare (x3)	86.8	5.25
j) 40 bit floating point add/sub	86.7	4.33
k) 16 bit datapath, 16x16 mult, state machine	98.1	4.86
l) 2901 (x2)	93.2	4.68
m) DRAM, DMA &SCSI controllers, UART	92.6	4.73

Table 2: Place and Route Examples.

Examples routed in possible implementations of the architecture with 1232 (a-j) and 649 (k-m) logic modules. The notation "(xN)" means the block was replicated N times.