

Quark Routing

Sean T. McCulloch¹ and James P. Cohoon²

¹ Ohio Wesleyan University, Department of Computer Science, Delaware,
OH, 43015 USA

² University of Virginia, Department of Computer Science
Charlottesville, VA 22903, USA

Abstract. With inherent problem complexity, ever increasing instance size and ever decreasing layout area, there is need in physical design for improved heuristics and algorithms. In this investigation, we present a novel routing methodology based on the mechanics of auctions. We demonstrate its efficacy by exhibiting the superior results of our auctionbased FPGA router QUARK on the standard benchmark suite.

1 Introduction

Most typical routers are based on a *sequential* strategy. With this methodology, one of the nets to be routed is chosen first, and is given a path anywhere among the initially unclaimed set of routing resources. Then a second net is routed in the space unused by the first net, and so on, until the last nets must find a path that has not been used by the preceding nets. We feel that the sequential strategy has inherent limitations that impact both solution quality and runtime.

The main contributions of this investigation are a routing methodology based on the mechanics of an auction that supports a simultaneous routing philosophy; and an FPGA router QUARK that implements a version of our methodology. The idea of using an auction for decision-making is not new, and appears with some frequency in the operating systems literature. One particular area is in the allocation of resources in distributed environments. Tasks are given virtual money with which to bid on the various resources in the system. When a resource becomes available, all tasks can bid on that resource. The highest bidder pays for the resource and utilizes it [GAG95].

There are also two similar negotiation-based routers that have some slight resemblance to our methodology—the PathFinder and NC routers [MCMU95, CHAN00]. The routers represent an FPGA as a graph. Initially, each net is assigned an optimal path, even if it conflicts with the paths of other nets. The algorithm iterates until there are no conflicting assignments. The current cost of a shared resource is set to the number of nets that want it. Each net then finds a new shortest path. Because the shared resources become more expensive, nets heuristically end up negotiating an alternate path. Although this process is not dependent on an initial net ordering, it is not a truly simultaneous strategy. In practice, nets are placed down in order of congestion and are typically ripped up in that order if another net overlaps any part of that net's path.

In our methodology, the entire auction of all the routing resources takes place concurrently. This property has led to a different computational model that represents

a novel approach to performing simultaneous routing. In addition, our negotiation processes are both different and more personalized.

2 Basic FPGA Auction Methodology

The concept of our auction-based routing methodology is straightforward—the pins of an FPGA are resources that can be bid upon by the nets. Each net seeks to control a set of pins that realize a complete detailed route for that net. For discussion ease, we define two terms.

- A *pin-auction* is the local auction of a single specific pin. The auction generally consists of several nets bidding for the right to route on that pin.
- A *chip-auction* is the complete auction process over the entire circuit. Thus, this auction comprises all of the various pin-auctions.

Thus, a run of an auction-based router corresponds to a single chip-auction, where a chip auction consists of collection of pin auctions, one for each pin.

Only one net can win a given pin-auction, and thus have the right to be routed upon that pin. The goal of each net, while working independently of the other nets, is to win sufficient pin auctions to realize its detailed routing. The chip auction completes successfully after all nets have achieved a detailed routing. It is important to stress that all of the pin-auctions are taking place simultaneously. The individual pin-auctions start when the chip-auction starts and finish when the chip-auction completes. This requirement gives quick proof to our claim that QUARK is a simultaneous router.

2.1 Income

The algorithm begins with each net being given an initial allocation of funds. These funds are normally the only source of assets available to a net for bidding on various pin-auctions. An alternative method would have been to have periodic “pay periods” for nets. For example, nets that are losing several pin-auctions could be given extra money. However, it is our experience that routinely adding income to the system merely cause the prices in the pin-auctions to increase and hampers the chip-auction's ability to finish. Thus, we recommend limiting the application of additional funds to extreme cases.

After a net has been given its money, the net then places its initial bids. We view this initial bidding as being a distinct process from subsequent bidding actions. The initial bidding process must select the specific path to realize the net. Subsequent bidding processes generally only require checking of pin-auctions and updating of bids if necessary.

Once each net has placed its initial set of bids, the chip-auction enters its iterative main phase. In each iteration of the main phase, all nets are given an opportunity to place or modify bids in the various pin-auctions as they see fit. The nets make bids in such a way as to eventually claim ownership of pins that realize a complete detailed route. If two or more nets enter a pin-auction, the first net processed with a maximal bid has current control of the pin.

A net is considered to have a *complete detailed route* if the set of pin-auctions that it is currently winning comprise a path that would be a legal detailed route for the net. There are no “freeze points” where if a net is winning a pin, it can hold it. Doing so would violate the simultaneous nature of the router. As stated before, there is only one chip-auction for the entire run of the router, and that all nets must constantly have bids on the pins that they need to realize their route.

The chip-auction completes *successfully* on the first iteration in which all nets have a complete detailed route. The chip-auction completes *unsuccessfully* if at some iteration it is determined that it is not possible for a given net to realize a complete detailed route. It is possible for the chip-auction to continue indefinitely, with neither of these criteria being met. However, this situation does not seem to arise in practice.

Figure 1 is an example of the bidding process at work on a sample block. Observe that several nets have active bids on different pins of that block at the current time. It is responsibility of each net to ensure that it is bidding sufficiently high in its various pin-auctions to have control of pins that realize a complete detailed route for itself.

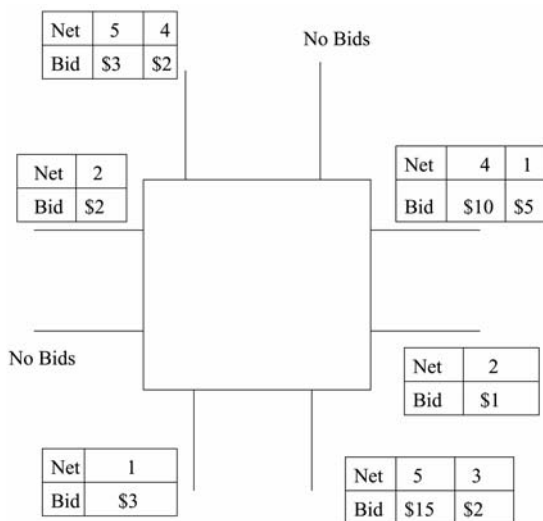


Fig. 1. Hypothetical bidding scenario.

3 Design Details

We now turn our attention to design details. Firstly, there is the issue of the bidding agents. In our model, each net is an active bidding agent, making decisions on which pin-auctions to participate, and once participating, on how much to bid. As a consequence, routing decisions can be made on a low level, which in turn means a priori that there is no general routing policy for deciding which nets gain which pins. Thus, each net individually determines how to best realize its route. Because the nets are simultaneously performing bidding operations, it makes sense to give each net its own view of the state of the various pin-auctions.

An important benefit of delegating these low-level decisions is that a routing tool is then free to allow different nets to use different routing algorithms for determining on which pins to bid. We call these local decision processes the net's *personality*.

While it is important to enable the nets to act individually, and simultaneously, it is also helpful to have a central authority, which we call the *overseer*, for policy decisions that require more information than can be found with regard to an individual net or pin.

3.1 Income and Priorities

When assigning a net an initial amount of money at the beginning of the chip-auction, the amount should be sufficiently large, to give the net some flexibility with regard to the bids of other nets. For example, suppose a net was given only \$10 and was limited to bidding integral amounts. If the net needed ten pins to realize its route, it could only bid \$1 per pin. It would have no excess money in a competition with other nets for its pins.

The initial income given to each net is a very natural way to implement a priority structure. In some designs, there is a need for a *critical net*—a net of high priority that must be given preference in finding its preferred path. In our design, a critical net can be given more initial funds. Heuristically, it is more likely in practice to win the pin auctions along its preferred path. If a net is absolutely critical and must be given its preferred path at the expense of the other nets, then it can be effectively given infinite funds to ensure that it gets that path.

3.2 Overseer

To ensure that the flow of the auction is simultaneous, it is important that there be a rapid flow of information regarding bids amongst the nets. The overseer must process bids quickly, and neither go into a failure state nor allow the router to end prematurely because nets have had an inadequate opportunity to bid against each other.

The overseer can determine whether a chip-auction has successfully completed by simply polling the nets whether they have won a complete detailed route. However, determining whether a chip-auction will be ultimately unsuccessful cannot be answered algorithmically (The problem is a variant of the Halting problem). Thus, the overseer must implement a heuristic to recognize this eventuality.

To decrease the likelihood of an unsuccessful routing, an overseer can be helpful in the case of a net lacking sufficient funds to complete any possible path to realize its detailed route. The routing analogy of eminent domain can be applied in such situations—the overseer steps in and in some manner assists the net in acquiring its path. Care needs to be taken so that the implementation of eminent domain does not effectively turn the router into a sequential one by simply assigning a path outright to the indigent net.

It is the overseer that also extracts the technology files; creates an internal representation of the FPGA that meets the specifications of the technology files specify; and extracts the global routing information for each net.

3.3 Pins

Our methodology gives individual pins the responsibility of processing the pin-auctions associated with them. As a result, the pin reports which net is currently winning its pin-auction. Furthermore, the pin ensures that only the current winner of the pinauction has the right to be routed on it.

3.4 Nets

Our net representation is also markedly different from similar representations found in traditional routers. For example, nets must have the ability to receive money and to bid this money in the pin-auctions. Most importantly, the nets must have the ability to bid on pin-auctions in a effective manner that realized a complete detailed route. This decision making is quite different from that found in previous routers. Routers traditionally have enforced routing decisions at a high-level and have required all nets to all perform homogeneously.

The placing of routing decisions at the net level has several benefits. By allowing different nets to have different personalities, we add great flexibility to the router's functionality. By changing the personality of an individual net, it is possible to optimize that net's routing path goal without impacting the routing decisions of the other nets. In addition, it is also possible to individually change the routing qualities that the nets are trying to optimize. For example, it is possible to route certain nets while trying to minimize delay, and to route other nets to minimize net length.

4 Personalities

The concept of a net personality is a novel one, and leads to many benefits in the routing process. As mentioned earlier, a net personality encapsulates a collection of routing decision processes. Thus allowing the router to give different nets different personalities based on their structure and importance. Our implementation provides three personalities.

The three personalities do have some commonality due to the nature of the routing task. Firstly in our environment, a net must have the ability to receive money from the overseer. One allocation activity always happens at the beginning of the chip-auction. An allocation can also occur as the result of an eminent domain request.

Secondly, a net must have the ability to make a constructive bidding decision in reaction to a request for bid(s) from the overseer. The algorithms that determine these bidding decisions are the central component of a net's personality. Although personalities will make different decisions as to where to bid, they will all share the goal of finding a complete detailed route. A net typically has three main bidding actions: increasing its bids when losing, decreasing its bids when winning, or deciding to bid on an alternate path if it deems its current path is too expensive. Besides being able to perform such actions, a net must be able to determine when they apply. For example, when a net is being outbid in a pin-auction and has sufficient unallocated funds to gain the pin, the net needs a process for determining what bid is sufficient. It will be an aspect of the net's personality to determine that new bid size.

Thirdly, each personality must have a mechanism for finding an alternate path if it deems the current path that it is attempting to acquire as too expensive. In implementing this mechanism, a net must be able to remove bids, modify existing bids, and place new bids.

Fourthly, a net needs a mechanism for determining when there are more affordable possible paths that it can construct. If an eminent domain request is viable in such a case, the request is signaled to the controller.

The personalities have been designed to optimize for the traditional FPGA routing problem of minimizing channel width and total net length. The first personality that we present is a very basic one. As such, we have named it the *baseline personality*. Because the baseline personality makes very simple decisions, it served as a framework for the succeeding personalities. The other two personalities—the *multiple personality* and the *focused personality*—extended the baseline personality in very different directions with regard to the management of resources. Both directions are heuristically promising in practice. The extensions were

- Allowing a net to bid on more than one possible path.
- Allowing a net to concentrate its resources in a small number of important locations.

We next discuss the three individual personalities in more detail.

4.1 Baseline Personality

The primary goal of a baseline personality net is to realize a detailed routing by trying to win the pin-auctions along its current global routing. (The global routing is maintained as the list of blocks that connect the various terminals of the net together.)

At the start of the chip-auction, the net is given the global route as determined by a global router in a previous step of the design process. Using the global route, an arbitrarily-determined legal detailed route is chosen. A minimal bid is placed upon each of these pins in the detailed route.

The personality favors pins that have no interest among the other nets. This bias helps the net to avoid congestion and to minimize overall resource expenditure. Once every pin in the detailed route has a bid of \$1 on it, the net is ready to enter the main phase of the auction.

In the main phase, when a baseline personality net is signaled to react to the current state of the chip-auction, the net first checks the status of each pin-auction along its current detailed route. If a net is winning a pin-auction by a substantial amount, the net heuristically concludes that some other net has lost interest in this pin-auction. The reason being that the net has minimally raised its bid in the past. As a result, the net reduces its bid so that it is only winning by only \$1. The bidding decrease frees up funds to bid in other pin-auctions.

If instead a net is losing a pin-auction and has sufficient funds to gain the pin, the net bids the minimal amount to lead the pin-auction.

If the needed funds are not there, the net enters a re-route state. In this state, the net first tries to make changes that minimally alter the current detailed route. For example, the net might try to switch to a cheaper pin on the same block. However, switch blocks on most conventional FPGAs do not have great flexibility. Therefore, these attempts may not be successful. In such cases, the baseline personality follows a

maze-routing strategy similar in nature to the UPSTART detailed router [McC02]. The UPSTART strategy considers a bounding region around the block that contains the expensive pin for the net. The width and length of the bounding region are specified by parameters. All of the net's bids on pins in the interior of this bounding region are removed. The locations where the detailed routing intersects the bounding box are marked as virtual terminals. An alternate detailed route is calculated that connects these virtual terminals together. If such a path can be found and if the net has sufficient funds to gain the associated pin auctions, then minimal winning bids are placed on these pins.

4.2 Multiple Personality

The baseline personality is clearly a very simple one and definitely has much room for improvement. One interesting way to improve the quality of the routing is to exploit some of the flexibility provided by the auction methodology. For example, we can have a net simultaneously show interest in multiple detailed routes. Sometime later in the course of the chip-auction, the net can commit to some particular routing. We have implemented such a personality and call it our *multiple personality*.

As a way of also demonstrating the fact that different personalities can be designed for different types of nets, we have designed the multiple personality to work on two-terminal nets whose terminals do not lie within the same channel. (If the terminals did lie so, there is a unique shortest path with regard to blocks.)

The key to understanding our multiple personality is to recognize that a two-terminal net with terminal blocks in differing channels has multiple minimum-length paths that are of the dogleg form. A dogleg is a connection in the form of a single stair step, where the step can be oriented either horizontally or vertically. The flat segment can be preceded and/or followed by riser segments of orthogonal orientation. To instantiate a dogleg, we only need to specify its orientation and the length of the risers. Our multiple personality starts by bidding all of its initial allocation as possible in equal amounts on each of these paths.

In subsequent bids, the personality first determines whether there are routes that it is currently bidding on which cannot be won with the available funds. If so, one such route is removed from consideration and the funds are reallocated to the other routes. In particular, the funds are first spent on routes that are most in danger of being lost and then on the other routes.

If the path that was out-bid was the last one being considered by the personality, then the net enters a re-route phase. In rerouting, the personality examines all possible dogleg paths. If there is a path that the net can afford, it is taken. The net then bids all of its funds along this path, in the expectation that the bids will be sufficient enough to retain control. If this action proves unsuccessful, later iterations will try a different dogleg path that the net can afford. The process continues until a net retains the path that was found, or the net realizes that all possible paths cost more money than the net has. In this case, an eminent domain request is considered.

4.3 Focussed Personality

Just as the multiple personality was designed to be used for two-terminal nets, so have we designed the focused personality to be used on particular types of nets. We expect

that a “focus attention on a set of pins that are important to win” strategy would work well if we focus the bidding on certain pins of a multi-terminal net. We define a *branching point* of a multi-terminal net to be a point that has more than two edges incident on it. In QUARK, we treat all blocks where the global route has greater than two edges incident on the block as a branching point. The personality assumes that the global router has intelligently placed these branching points. Therefore, it is important to win the bid on these pins.

The focused personality divides the initial money resources into two main parts. The amount to be used in bidding on branching points, *branching money*; and the amount to be used in bidding on the remainder of the routing, *connection money*. Clearly, the balance between these two amounts is important. If one component does not receive sufficient funds, then the net will be out-bid by some other net and thus have trouble completing a routing.

QUARK allocates enough connection money such that a bid of \$2 can be made on each pin in the given global route. This allows the router to find connections that are longer than the length given by the global route. In doing so, it will be necessary to give some pins \$1 bids. Another benefit of this behavior is that a net can bid more than \$2 on certain pins by reducing the bid of other pins to \$1. The difference gives the router some flexibility in finding a connection path that it can actually keep.

At the beginning of the chip-auction, the division between branching money and connection money is made. The branching money is initially divided evenly among all pins on all branching points.

Once the branching points are bid, the net bids on the connections between branching points. It is important to observe that all effective routing segments leaving branching points either eventually lead to a single terminal, or a single pin on another branching point. Our implementation uses a quick maze-routing strategy to connect the two pins. The resulting routing can be of arbitrary length and go in arbitrary directions. The limitations are based only on the amount of connection money.

When the router signals a focused personality net during the main bidding phase, the net makes two separate checks: one for the connection paths, and one for the branching points. Each of the current connection paths are examined to make sure that the net is winning the pin-auctions along the entire path. If a net is winning, then an additional check is made to see if connection money can be freed. If a net is not winning and there is enough connection money remaining to make the bid higher, then that money is spent to increase the bid. If there still is not enough excess money, the net enters the reroute state.

Note that we do not use branching money to help these connection paths become routed. Doing so would slowly siphon funds away from the branching points into the connection paths. This action would violate the principles of this personality, because it would result in the branching points being more likely to be out-bid.

If a branching pin is being out-bid, then the net examines the bids on all of the other branching pins. The net searches for a branching pin that is winning by a amount that allows for some reduction. The reduction will free up branching money to help the out-bid branching pin to regain control of the pin-auction. The process continues until either the out-bid pin has enough excess money to control the pin-auction again, or it is the case that none of the other branching pins can lower their bid without losing their pin-auction. If this condition results, the net makes an eminent domain request.

The re-routing phase for the connection paths uses a maze-routing algorithm to find a different inexpensive connection path.

5 Experiments

We now analyze a series of experiments that we performed on the Quark router. The experiments were conducted using the standard set of benchmarks maintained by the University of Toronto. The benchmarks have been the basis for the testing of a significant number of tools. Different benchmarks have different numbers of nets, blocks, and block layouts. In our experiments, we use the QUARK tool as a detailed router. The placement and global routing were performed by the SPIFFY tool [SELF]. Because SPIFFY is nondeterministic with regard to its output, five separate runs of SPIFFY were generated for each benchmark. Besides producing a placement and global routing, each run also heuristically specified an expected channel width given complete switchboxes.

Our first set of experiments tested the functionality of the various personalities relative to each other. For these tests, we created four versions of QUARK. The versions differed in their use of various personalities. By using these tests, we are able to make a preliminary decision on which versions of QUARK are best in practice.

- QUARK-BASELINE: every net uses the baseline personality.
- QUARK-MULTIPLE: all nets that qualify for the multiple personality use that personality, and all other nets use the baseline personality.
- QUARK-FOCUSSED: all multi-terminal nets use the focussed personality, and all two-terminal nets use the baseline personality.
- QUARK-ALL: all multi-terminal nets use the focussed personality, all two-terminal nets that qualify for the multiple personality use that personality, and the remaining two-terminal nets use the baseline personality.

A summary of these results are presented in Table 1. This table shows the minimum channel width successfully routed to by the various versions of Quark, along with the average smallest channel width of several runs.

We also note best results tend to be generated by the multiple personality and the baseline personality. Although the baseline personality is slightly better on average, it is significantly slower to run. The results suggest an interesting composite algorithm—run QUARK-MULTIPLE until it discovers a channel width that is too small for it to successfully route. The router then switches to using the slower but more effective QUARK-BASELINE. We call this version QUARK-PRIME. QUARK-PRIME required several minutes to produce its solution for the smaller instances. For the largest instances, it required slightly more than 200 minutes on a Solaris workstation.

Table 2 compares QUARK-PRIME to several state-of-the-art routers: Graph-based router [ALEX98], SEGA tool [LEMI93], CGE tool [BROW92], and VPR tool [BETZ97]. Because the SEGA and CGE tools are designed for different architectures, their results are combined. We note that only VPR betters QUARK-PRIME performance. However, VPR is not running the same instances as QUARK-PRIME. The VPR layout system uses a technology mapper to reduce the number of logic blocks that is unavailable to QUARK-PRIME. This table shows the minimal channel width routed to by each of the routers on these benchmarks.

Table 1. Channel widths successfully routed

| Circuit | QUARK | | QUARK | | QUARK | | QUARK | |
|---------|----------|------|----------|------|----------|------|-------|------|
| | BASELINE | | MULTIPLE | | FOCUSSED | | ALL | |
| | Avg | Best | Avg | Best | Avg | Best | Avg | Best |
| dfsm | 8.8 | 8 | 10.4 | 9 | 11 | 10 | 10.6 | 10 |
| 9sym | 8.0 | 8 | 7.4 | 7 | 7.6 | 7 | 8 | 8 |
| term1 | 5.8 | 5 | 5.8 | 5 | 7 | 5 | 7 | 6 |
| apex7 | 6.2 | 6 | 6.6 | 6 | 6.2 | 6 | 7.4 | 7 |
| alu2 | 8.4 | 7 | 8.6 | 8 | 9.4 | 9 | 9.2 | 9 |
| alu4 | 8.4 | 7 | 9.8 | 9 | 12 | 10 | 11.4 | 11 |
| Total | 45.6 | 41 | 48.6 | 44 | 53.2 | 47 | 53.6 | 51 |

Table 2. Inter-router comparison.

| Circuit | QUARK PRIME | Graph- Based | SEGA/ CGE | VPR |
|---------|----------------|-----------------|--------------|------|
| dfsm | 8 | 9 | 10 | — |
| 9symml | 6 | 8 | 10 | 5 |
| z03 | 9 | 11 | 13 | — |
| term1 | 5 | 8 | 10 | 5 |
| apex7 | 5 | 14 | 10 | 4 |
| alu2 | 7 | 9 | 11 | 6 |
| alu4 | 8 | 11 | 15 | 7 |
| Total | 48 (31) | 70 | 79 | (27) |

References

- [ALEX98] M. Alexander, J. P. Cohoon, J. L. Ganley, and G. Robins, Placement and Routing for High-Performance FPGA layout, *VLSI Design: International Journal of Custom-Chip Design, Simulation, and Testing*, 97-110, January 1998.
- [BETZ97] V. Betz and J. Rose, VPR: a new packing placement and routing tool for FPGA research. International Workshop on Field Programmable Logic and Application, 1997.
- [Brow92] S. D. Brown, J. S. Rose, and Z. G. Vranesic, A detailed router for field programmable gate arrays, *International Conference on Computer-Aided Design*, 382-385, 1990.
- [CHAN00] P. K. Chan and M. D.F. Schlag, New parallelization and convergence results for NC: a negotiation-based FPGA route, *International Symposium on Field Programmable Gate Arrays*, 165-174, 2000.
- [GAG95] R. A. Gagliano, M. D. Fraser, and M. E. Schaefer, Auction allocation of computing resources, *Communications of the ACM*, J88-102, June 1995.
- [Lemi93] G. G. Lemieux and S. D. Brown, A detailed routing algorithm for allocating wire segments in field programmable gate arrays, *ACM-SIGDA Physical Design Workshop*, April 1993.
- [McCu03] Sean T. McCulloch, Auction-based routing for FPGAs, University of Virginia, Doctoral Dissertation, 2002.
- [MCMu95] L. McMurchie and C. Ebeling, Pathfinder: A negotiation-based performance-driven router for FPGAs, *International Symposium on Field Programmable Gate Arrays*, 111-117, 1995.
- [SHRA87] E. Shragowitz and S. Keel, A global router based on a multicommodity flow model, *INTEGRATION: the VLSI Journal*, 3-16, 1987.