Review of:

Rapid Construction of Partial Configuration Datastreams from High-Level Constructs Using JBits

- Paper by:
  - Satnam Singh – Xilinx (University of Glasgow)
  - Philip James-Roxby – Xilinx (University of Birmingham)
- Published in:
  - Field-Programmable Logic and Applications (FPL) 2001
  - August 2001
- Original copy on-line as:
  - http://link.springer.de/link/service/series/0558/papers/2147/21470346.pdf
- Survey by:
  - Christopher Zuver

The Challenge

- Provide ability to
  - Rapidly generate configuration datastreams
  - Ability to generate full or partial datastreams
- Expand on the JBits
  - Server-Client model for adaptively
    - Server and Client may run on same host
      - Generally faster then the common design flow
    - Abstracts away Java language level interface with standard network Sockets
  - No “real” support for partial configuration

Style of the Paper

- A Survey of previous JBits work and limitations
  - Broad - assumes familiarity with JBits from previous FCCM'01 publication on JBits.
  - No overview of JBits (In previous work [1], a…)
- A survey of prior work
  - Cites 3 efforts for providing high-level support for run-time reconfiguration.
- Proposal of JBits Server-Client Model
  - Constructing the configuration datastream
  - API description of high-level constructs
  - Translation of API to datastream
  - Support for partial configuration
  - Performance

What's new in JBits Server-Client

- Prior model
  - Lava produces EDIF file
  - JBits parses EDIF and creates datastream
  - Much of the processing time spent on parsing.
  - No "real" support for partial configurations
- Client-Server Model
  - Client
    - generates constructs (nets and instances) using Lava (hardware description language)
  - Server
    - translates high-level constructs into low-level configuration calls using the JBits API.
  - Partial Configuration
    - Support for partial configuration achieved through support for reconfiguration state graphs (RSG)

Previous Work

- Brenbner – Swappable Logic Units (SLU)
  - "The Swappable Logic Unit: A Paradigm for Virtual Hardware"
  - http://dlib.computer.org/conferen/fccm/8159/pdf/81590077.pdf
  - Provide operating system like support – calls to load SLUs
  - FPGA is pre-loaded with a communications harness
    - Allows SLUs to interact
- Luk (et al) – RC_Mux
  - "Compilation Tools for Run-Time Reconfigurable Desings"
  - http://dlib.computer.org/conferen/fccm/8159/pdf/81590056.pdf
  - Components are selected at run time by a mux.
  - Each component programmed individually into FPGA
  - Component's simulation is independent
- Woods – Combining SFGs and CDGs
  - Produce reconfigurable designs by define algorithm as a SFG (Signal Flow Graph)
  - Combine SFGs to create overall algorithm in the CDG (Configuration Data graph)
  - Created designs stored in RSG (Reconfiguration State Graph) data structure.

Construction of Configuration Datastreams

- Client-Server removes need for physical file (EDIF)
  - Interact through high-level constructs
  - Support for RSG
    - multiple configurations produced

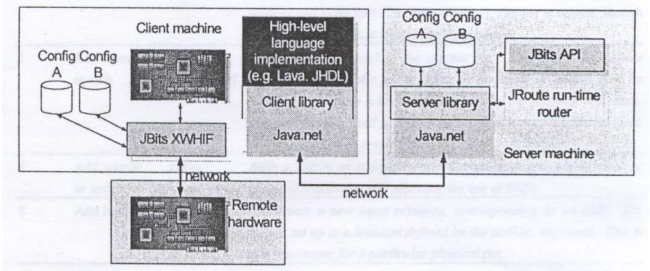


Figure 1 : Client-server architecture

Construction of Configuration Datastreams (cont)

- Architecture
  - Client
    - Transform high-level language to constructs in client library functions.
    - Lava front-end (was EDIF creator) – creates a network connection with a socket to the JAVA-based server.
      - Lava language is embedded in the Haskell lazy functional programming language
      - Haskell compiler produced at Microsoft Research
    - Lava sends high-level constructs (in client library) to server instead of producing a EDIF

What is Lava?

- Provides high level abstractions and specifies the layout of circuits
  - high level abstraction
    - Ability to use higher order functions.
    - Circuits are represented as functions and may be passed as arguments to other functions
    - Functions can also return circuits as results
    - **Circuit combinators** - composing circuits together to form larger systems.
      - Combine not only behavior but also layout.
      - Lays out circuits in rows, columns, triangles, trees etc..
    - Easier to produce parameterized circuits than VHDL
- Circuits produced at Xilinx with Lava
  - high speed constant coefficient multipliers
  - finite impulse response filters (1D and 2D)
  - adder tree networks
  - sorting butterfly networks.

Example of Lava

- Consider the definition of a three input AND-gate
- Define a function in Haskell that performs the AND3 operation:

```
and3function :: Bool -> Bool -> Bool -> Bool
and3function a b c = a & b & c
```

  - The first line gives the type of and function as something that takes three boolean values and returns a boolean value.
  - The next line defines the and3function in terms of the Haksell binary & function.
- Define a circuit that ANDs three inputs:

```
and3 = lut3 and3function
```

- Generated VHDL might look like this
  - instantiates a LUT with the appropriate programming information.

```
lut3_1 : lut3 generic map (init => "1000") portmap (i0 =>
lava(5), i1 => lava (6), i2 => lava (7), o => lava(4)) ;
```


Construction of Configuration Datastreams (cont)

- Server
  - Controls dialog with client (Only one Client)
  - Creates two ports for communications
    - Used for textual acknowledgment and partial configuration datastreams respectively.
  - Translation from client high-level constructs to datastream occurs
  - Secondary Design
    - User can now create another design based on RSG and the initial design
  - Uses standard byte-handling to support wide range of clients, not just Java based.

API Description

- Contains 10 instructions
  - examples
    - New design (designName, deviceType)
    - New partial (partialName, baseName
    - New instance (name, type, row, col, slice,Init)
      - type is primitive from unisim library
      - LUT1,LUT2,LUT3,LUT4,XORCY,MUXCY, RAM16x1
      - Can take initialization value
    - New net (name)
      - Add Source (Name, instanceRef)
      - Add Sink (Name, instanceRef)
      - instanceRef = null for top-level IOB
    - Add input (name, padLoc)

Translation Details

- Four Phases
  - mapping, placement, routing, and bitstream generation.
- Constant Values
  - Generated without additional logic
    - Uses pull-up mechanism of Virtex parts when inputs to CLB are not driven
    - Use inverters to produce “0”.
    - MUXCY can be turned into constants by leaving BX unconnected
  - Relies on Reference 1 for describing translation

“Anyone know the whereabouts of figure 2?

Support for Partial Configuration

- Uses RSG (reconfigurable state graphs)
  - Nodes correspond to actual configuration states
  - Arcs represent partial configurations required to transition between configurations

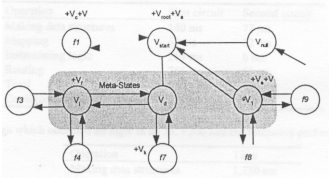


Figure 3 : Reconfiguration state graph from [4]

-- Meta-states

* Common configurations in multiple configurations

* Not implemented however could speed reconfiguration

Established API in JBits for partial reconfiguration [JRTR]

--Tracks frames (atomic unit of configuration) that become 'dirty' from configuration changes.

-- Dirty frames become the partial reconfiguration

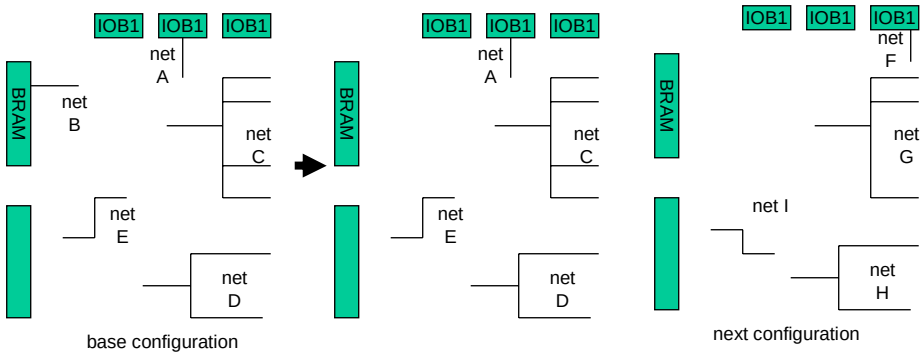
* Minimizes the total number of dirty frames

* Unused frames are disconnect but left in-tact so returning to the previous design is efficient

Support for Partial Configuration (cont)

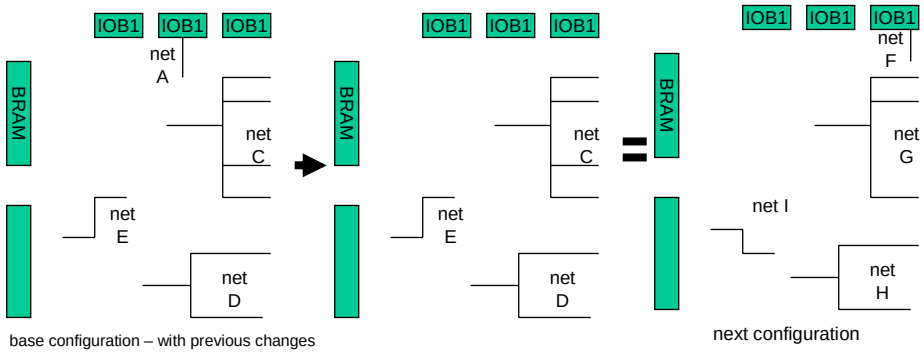
- Server is provided the base configuration and the target configuration
  - calculates the minimal disabling of the base configuration
  - removes all interaction (connects) to BlockRAM and IOBs using constructs in JRoute API.
  - Shared logical resources are checked for needed values before replacement
    - update from original JBitsDiff tool
  - Routing
    - all routing in base configuration left intact initially
    - if the existing source and sinks are identical route remains intact else they are removed

Example of partial reconfiguration



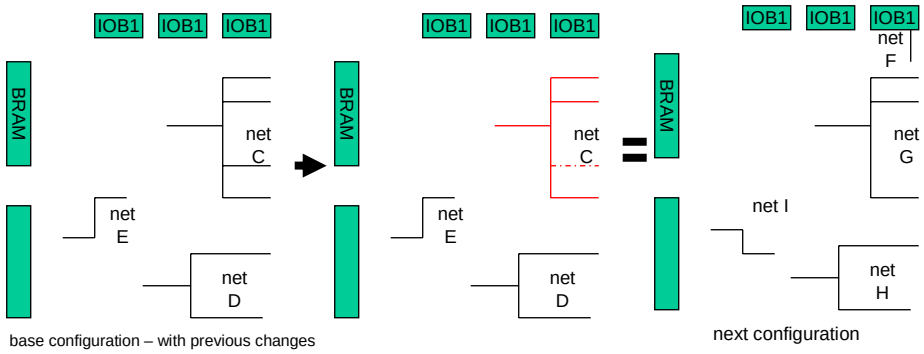
•Remove routes to BLKRAMS

Example of partial reconfiguration



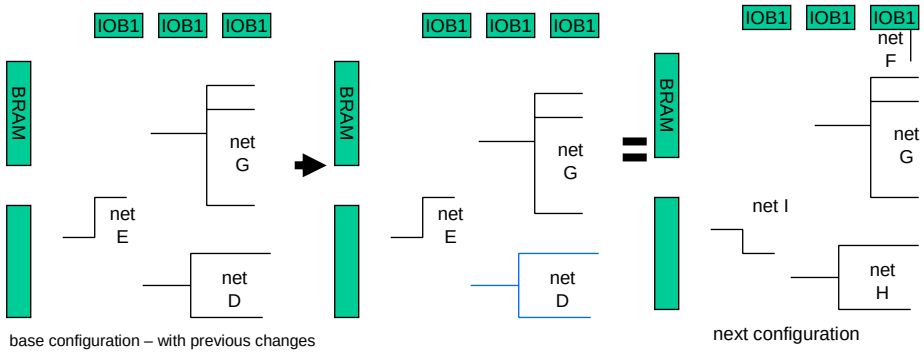
•Remove routes to IOBs

Example of partial reconfiguration



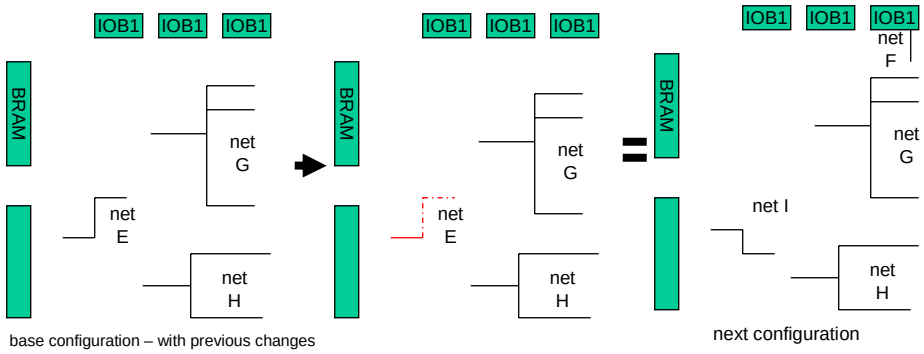
- Route "net G"
  - Doesn't match "net C"
  - "Net C" unrouted and "net G" is routed

Example of partial reconfiguration



- Need to route "net H"
  - Identical to "net D" so stays intact

Example of partial reconfiguration



- Need to route "net I"
  - Doesn't match "net E"
  - "Net E" unrouted and "Net I" is routed
- Need to route "net F"

Performance

- Results from changing one gate (in ms)

Operation	First Circuit	Second Circuit
Making data structures	140	77
Mapping	63	32
Instantiating logic	265	0
Routing	766	63
Total	1,234	172
Bytes produces	218,980	1,464

Performance

- Results from changing design using all logic avaliable (in ms)

Operation	First Circuit	Second Circuit
Making data structures	1,750	??
Mapping	969	??
Instantiating logic	1,149	??
Routing	4,875	??
Total	8,734	??
Bytes produces	218,980	??

Performance

- Results from butterfly network sorter fully placed in Lava(in ms)

Operation	First Circuit	Second Circuit
Transmission of Constructs	422	??
Making data structures	641	
Mapping	281	??
Instantiating logic	250	??
Routing	2,125	??
Total	3,297	??
Bytes produces	218,980	??

Performance

- Results from Original Lava → JBits conversion using EDIF intermediate (From FCCM'01)

Operation	First Circuit	Second Circuit
Parse EDIF	62.8s	Generate a partial configuration 94ms
Making data structures	7.23s	
Mapping	4.6s	
Instantiating logic	XX	
Routing	15.4s	
Total	90.25s	
Bytes produces	218,980	

Conclusions and Future Work

- Client-Server models allows flexibility in front-end tools
- Once client recieves configuration (full / partial) they can be use in actual hardware
- Future Work
  - add interfaces for other popular languages
  - improved interface for Fully-mapped, fully-placed EDIF
    - Only place except Abstract where EDIF are considered input…
    - Abstract not reflective of paper.

Questions and Discussion

- Abstract states server accepts EDIF files, however not stated again in the paper until the conclusion.
- Partial reconfigurable results only given for a one gate reconfiguration.
  - Where are the results for larger circuits?
- Concerns with paper
  - Little background information presented
  - References to lacking figures
  - Information lacking on role of server