Improving QoS of Multi-Layer Networks-on-Chip with Partial and Dynamic Reconfiguration of Routers

Leandro Soares Indrusiak

RTS York

Leandro Möller
Manfred Glesner

TECHNISCHE UNIVERSITÄT DARMSTADT

MES

Fernando Moraes

PUCRS

GAPH
Hardware Design Support Group

Goal: **Present a Multi-Layer NoC which has low area overhead and provides QoS by using partial reconfiguration features and circuit switching networks.**

Motivation: **Use less logic as possible for the NoC; NoC should be mainly wires. Use the right kind of NoC switching, which depends on the data being transferred.**

Circuit Switching x Packet Switching

Circuit Switching

- Reserve before use
- Waste of resources if not used
+Deterministic
+Need small buffer

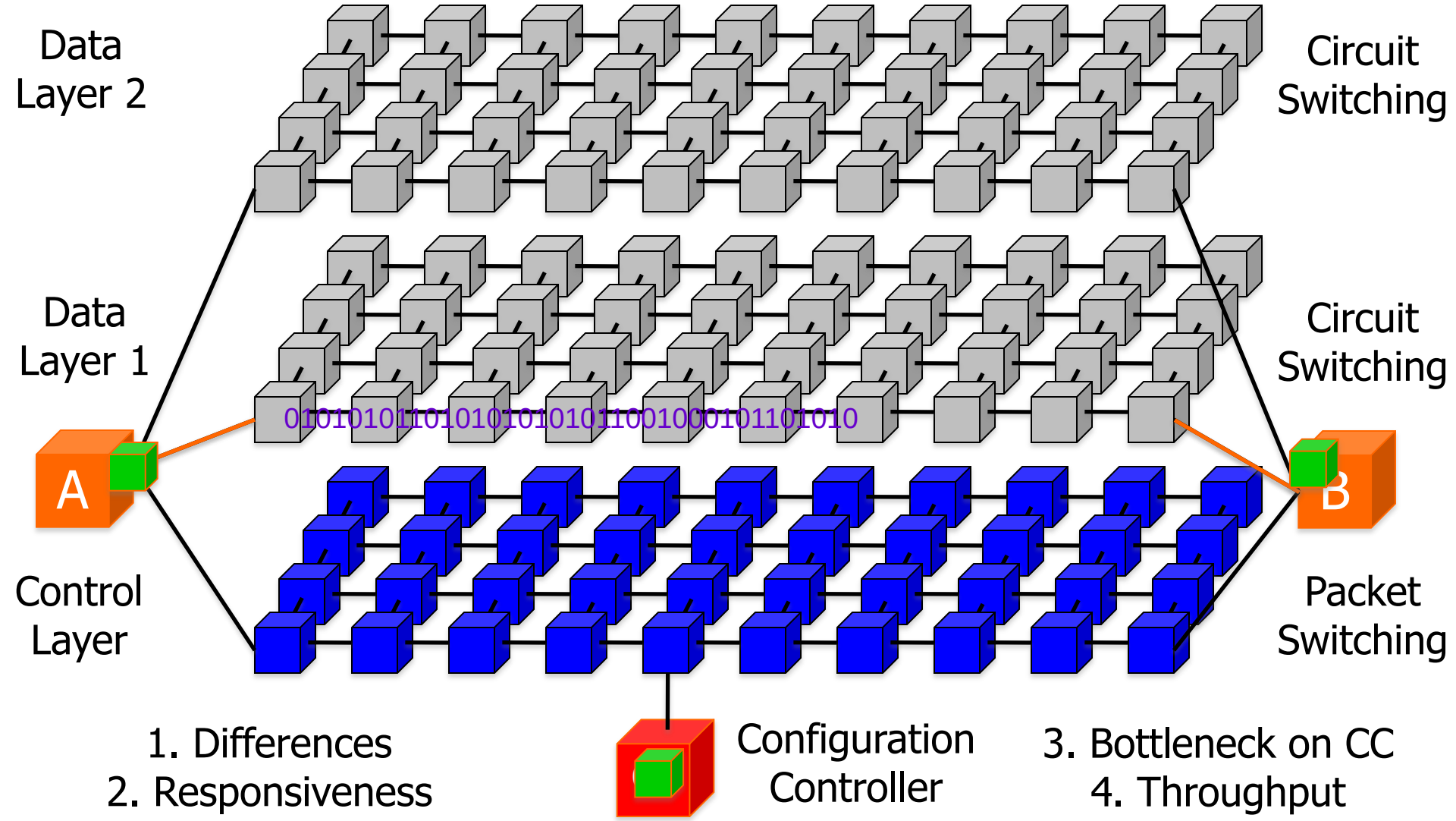
+Good for large data exchange

Packet Switching

+No reserve before use
+No waste of resources
- Unpredictable timing
- Need bigger buffers

+Good for small data exchange (control packets, acknowledges, monitor information, random data request/response)

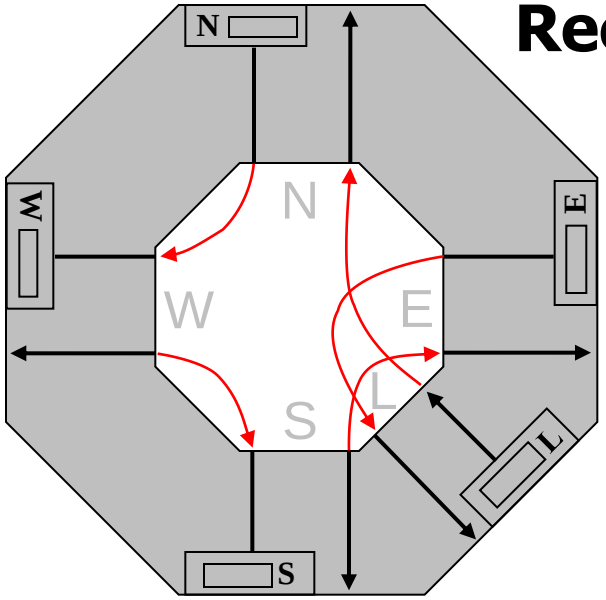
Proposed Multi-Layer NoC Architecture



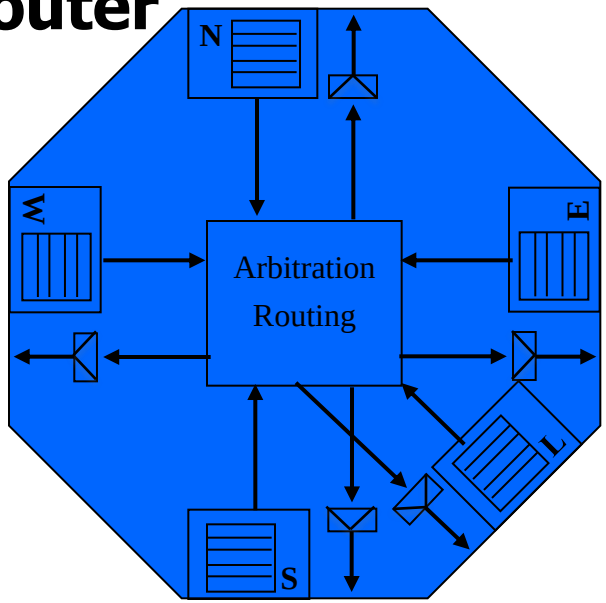
Types of NoC Routers

No Routing Algorithm
No Arbitration
No MUXs (router logic)
Small buffers

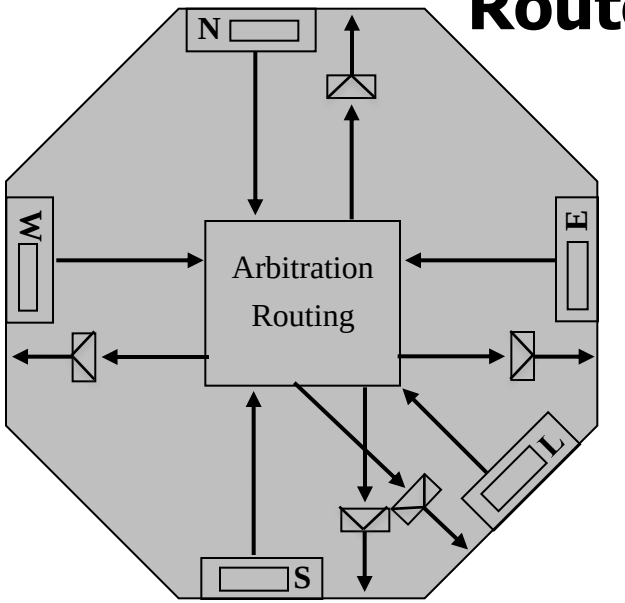
Reconfigurable Circuit Switched Router



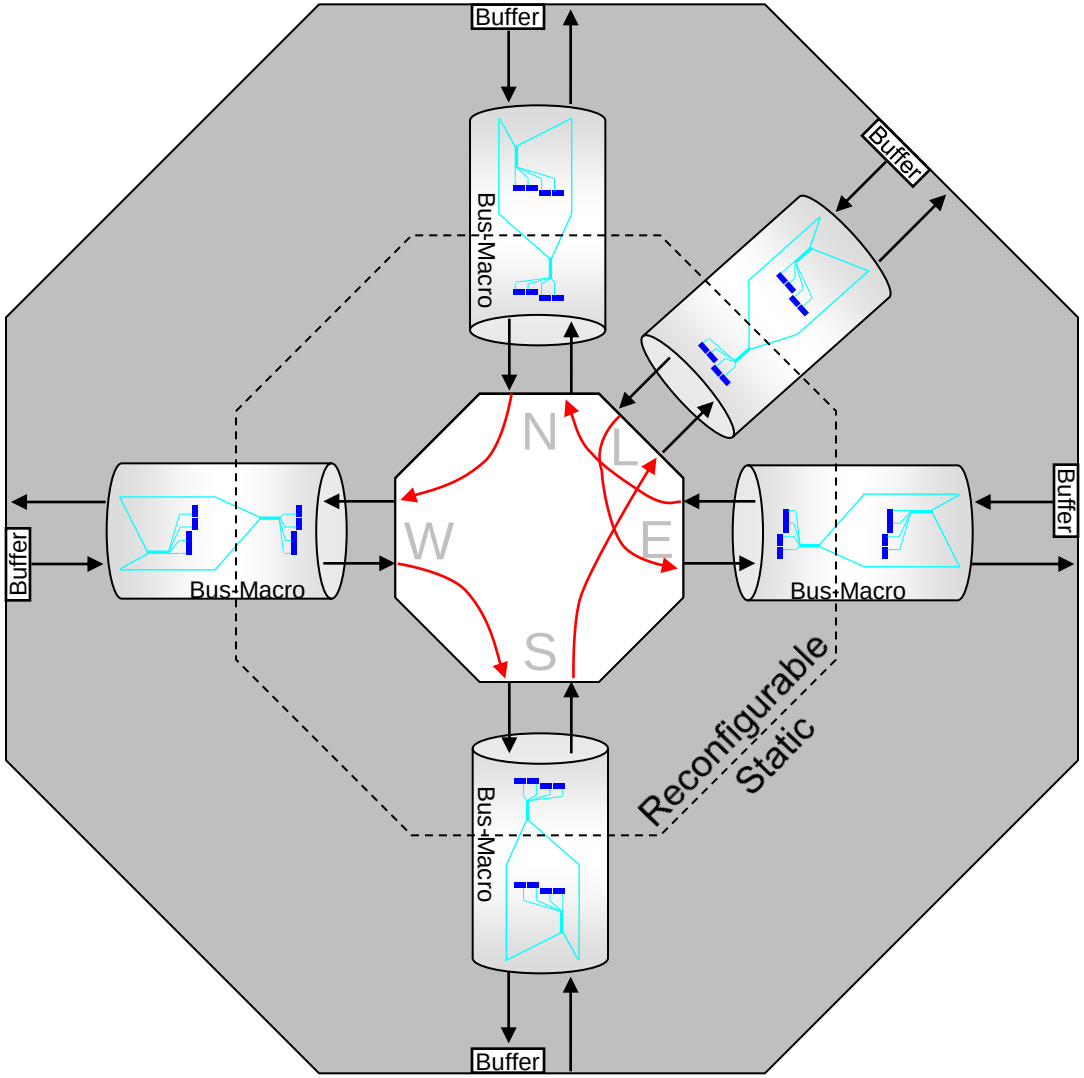
Packet Switched Router



Circuit Switched Router



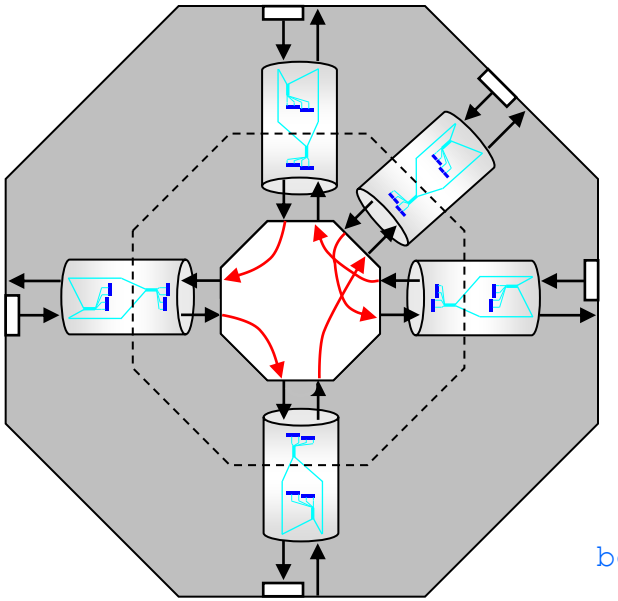
Reconfigurable Router for Xilinx FPGA



120 (5!) Different Fixed Routers

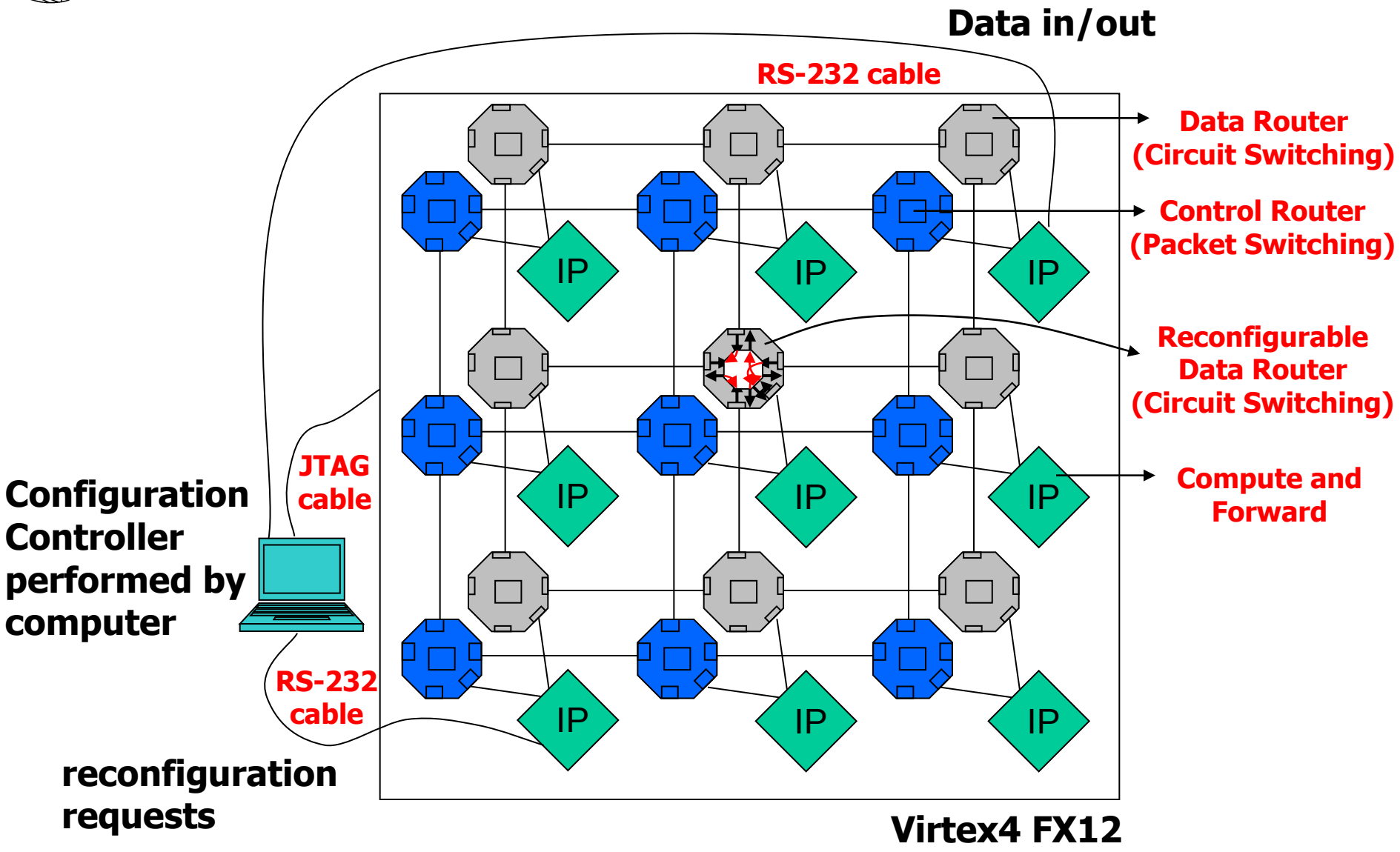
000_EWNSL	024_WENSL	048_NEWSL	072_SEWNL	096_LEWNS
001_EWNLS	025_WENLS	049_NEWLS	073_SEWLN	097_LEWSN
002_EWSNL	026_WESNL	050_NESWL	074_SENWL	098_LENWS
003_EWSLN	027_WESLN	051_NESLW	075_SENLW	099_LENSW
004_EWLNS	028_WELNS	052_NELWS	076_SELWN	100_LESWN
005_EWLSN	029_WELSN	053_NELSW	077_SELNW	101_LESNW
006_ENWSL	030_WNESL	054_NWESL	078_SWENL	102_LWENS
007_ENWLS	031_WNELS	055_NWELS	079_SWELN	103_LWESN
008_ENSWL	032_WNSEL	056_NWSEL	080_SWNEL	104_LWNES
009_ENSLW	033_WNSLE	057_NWSLE	081_SWNLE	105_LWNSE
010_ENLWS	034_WNLES	058_NWLES	082_SWLEN	106_LWSEN
011_ENLSW	035_WNLSE	059_NWLSE	083_SWLNE	107_LWSNE
012_ESWNL	036_WSENL	060_NSEWL	084_SNEWL	108_LNEWS
013_ESWLN	037_WSELN	061_NSELW	085_SNELW	109_LNESW
014_ESNWL	038_WSNEL	062_NSWEL	086_SNWEL	110_LNWES
015_ESNLW	039_WSNLE	063_NSWLE	087_SNWLE	111_LNWSE
016_ESLWN	040_WSLEN	064_NSLEW	088_SNLEW	112_LNSEW
017_ESLNW	041_WSLNE	065_NSLWE	089_SNLWE	113_LNSWE
018_ELWNS	042_WLENS	066_NLEWS	090_SLEWN	114_LSEWN
019_ELWSN	043_WLESN	067_NLESW	091_SLENW	115_LSENW
020_ELNWS	044_WLNES	068_NLWES	092_SLWEN	116_LSWEN
021_ELNSW	045_WLNSE	069_NLWSE	093_SLWNE	117_LSWNE
022_ELSWN	046_WLSEN	070_NLSEW	094_SLNEW	118_LSNEW
023_ELSNW	047_WLSNE	071_NLSWE	095_SLNWE	119_LSNWE

Reconfigurable Router

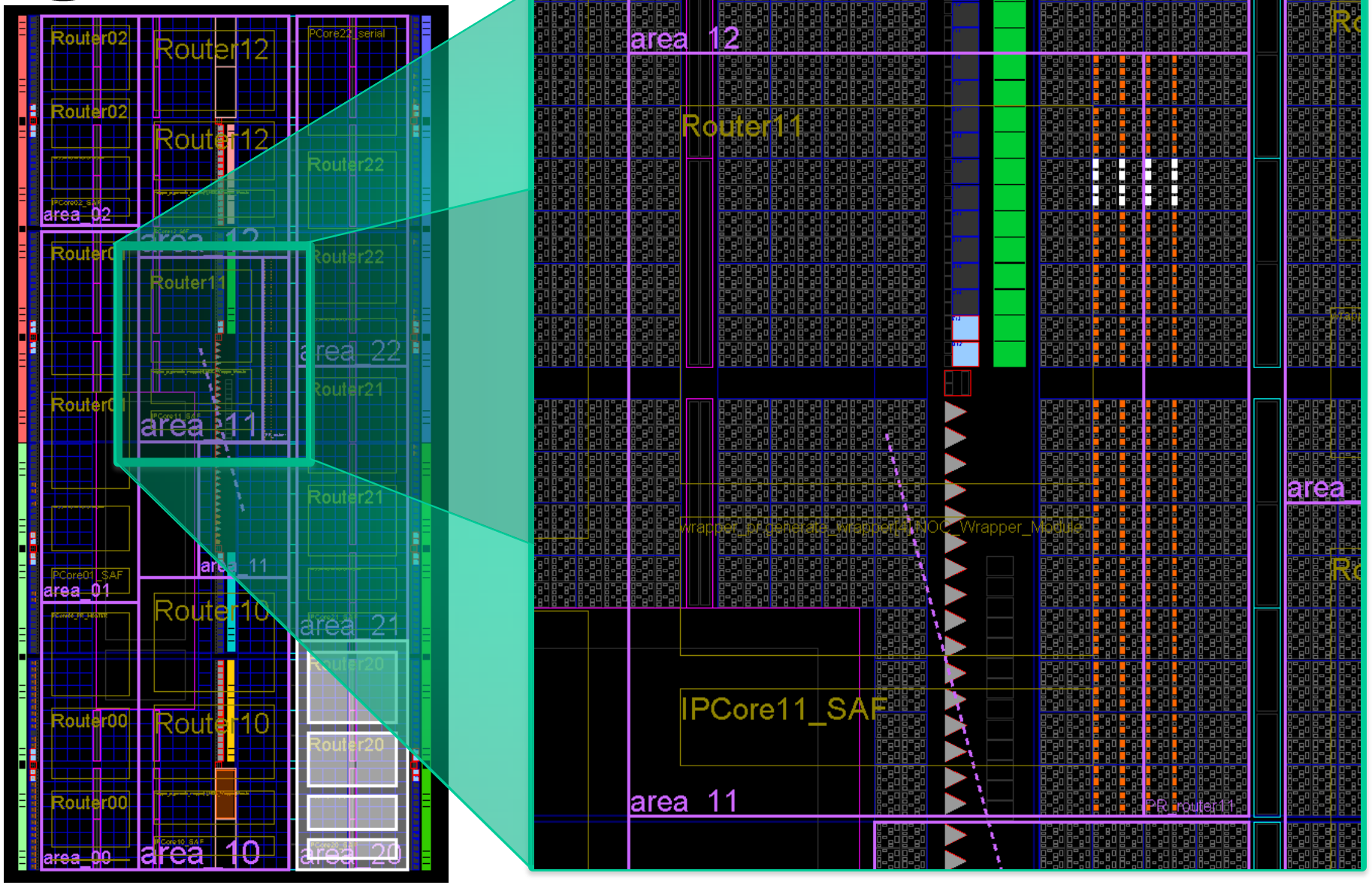


```
architecture Chave_Fixed of Chave_Fixed is
constant route: string := "NSWLE";

   -- convert character (describing direction) to integer (0-4)
   function c2i (c: character) return integer is
   variable i: integer;
   begin
      case(c) is
         when 'E' => i := 0;
         when 'W' => i := 1;
         when 'N' => i := 2;
         when 'S' => i := 3;
         when 'L' => i := 4;
         when others => i := -1;
      end case;
      return i;
   end;

begin

   -- generate connections according to the constant "route"
   generate_connections: process(rx, data_in, ack_tx)
   begin
      for i in 0 to 4 loop
         tx(c2i(route(i+1))) <= not rx(i);
         data_out(c2i(route(i+1))) <= not data_in(i);
         ack_rx(i) <= not ack_tx(c2i(route(i+1)));
      end loop;
   end process;

end Chave_Fixed;
```

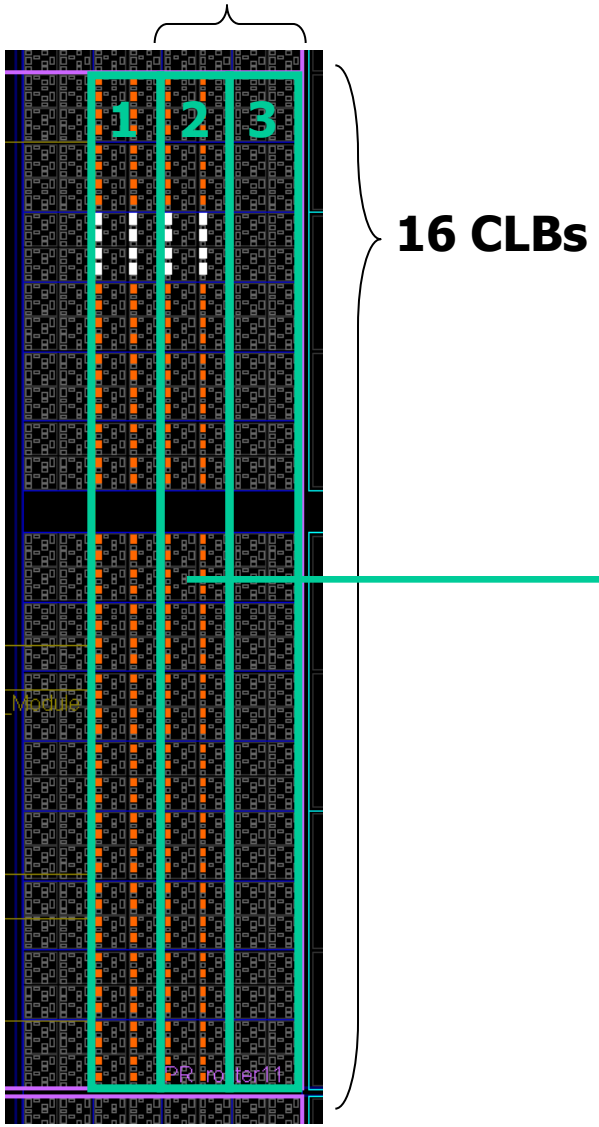
Case Study


Data in/out
RS-232 cable
Data Router
(Circuit Switching)
Control Router
(Packet Switching)
Reconfigurable
Data Router
(Circuit Switching)
Compute and
Forward
IP
JTAG
cable
Configuration
Controller
performed by
computer
RS-232
cable
reconfiguration
requests
Virtex4 FX12

Reconfigurable Router Floorplanning



Area Results



Reconfigurable Router Configuration:

5 in / 5 out ports, 8 bits per port

14 bus-macros (5 data_in, 5 data_out, 2 control_in, 2 control_out)

28 CLBs used, 32 CLBs reserved

256 LUTs and 256 FFs reserved (2% V4FX12)

1 Left side CLB column of Bus-Macro (EAPR)

2 Right side CLB column of Bus-Macro (EAPR)

3 Router wires and inverters (EAPR)

384 LUTs and 384 FFs reserved (3% V4FX12)

Timing Results

≈ 2s

	IP source starts sending a packet to request a new route
2 ck	IP source sends header to control router
6 ck	Delay to pass through each control router
6 ck	IP target (serial core) receives the entire request route packet
≈ 2s	Serial core sends 1 byte requesting a specific partial bitstream by the serial interface; Software running on the PC and connected to the serial interface receives the value, search this value on a table to find the name of the requested partial bitstream and call the Impact tool from Xilinx to trigger the partial reconfiguration; The Impact tool takes 54ms to send the partial bitstream by the JTAG interface running at @6MHz; After reconfiguration a byte is sent back by the serial interface to acknowledge that reconfiguration completed successfully; Finally the byte is received on the Serial core.
2 ck	Serial core sends a confirmation packet to control router
6 ck	Delay to pass through each control router
6 ck	IP source receives the entire confirmation packet
1 ck	IP source can start sending data through the established data path

Conclusions

- A multi-layer NoC was presented, where a packet switching NoC is used for control and one or more circuit switching NoCs can be used for data communication
- Partial reconfiguration is used to establish communications in the data network layer
- Disadvantage is the initial latency for establishing a communication path
- Advantage 1: area saving. Data routers do not need routing algorithms, arbitration and crossbars to connect input ports to output ports.
- Advantage 2: QoS. As each data layer uses circuit switching, maximum throughput is guaranteed between source and target of communication.

Future Works

- Currently 120 partial bitstreams are required for each router. If partial bitstream relocation is employed, only the 120 different combinations need to be stored.
- Implement a Configuration Controller on the FPGA, not in the PC.
- Allow self-reconfiguration by using the ICAP (Internal Configuration Access Port).
- Prototype the multi-layer NoC in a bigger FPGA and improve the case study.

Thanks for your attention

Questions?

Leandro Möller

TU Darmstadt

moller@mes.tu-darmstadt.de

Comparing Interconnection Infrastructures

Time required to establish a communication channel

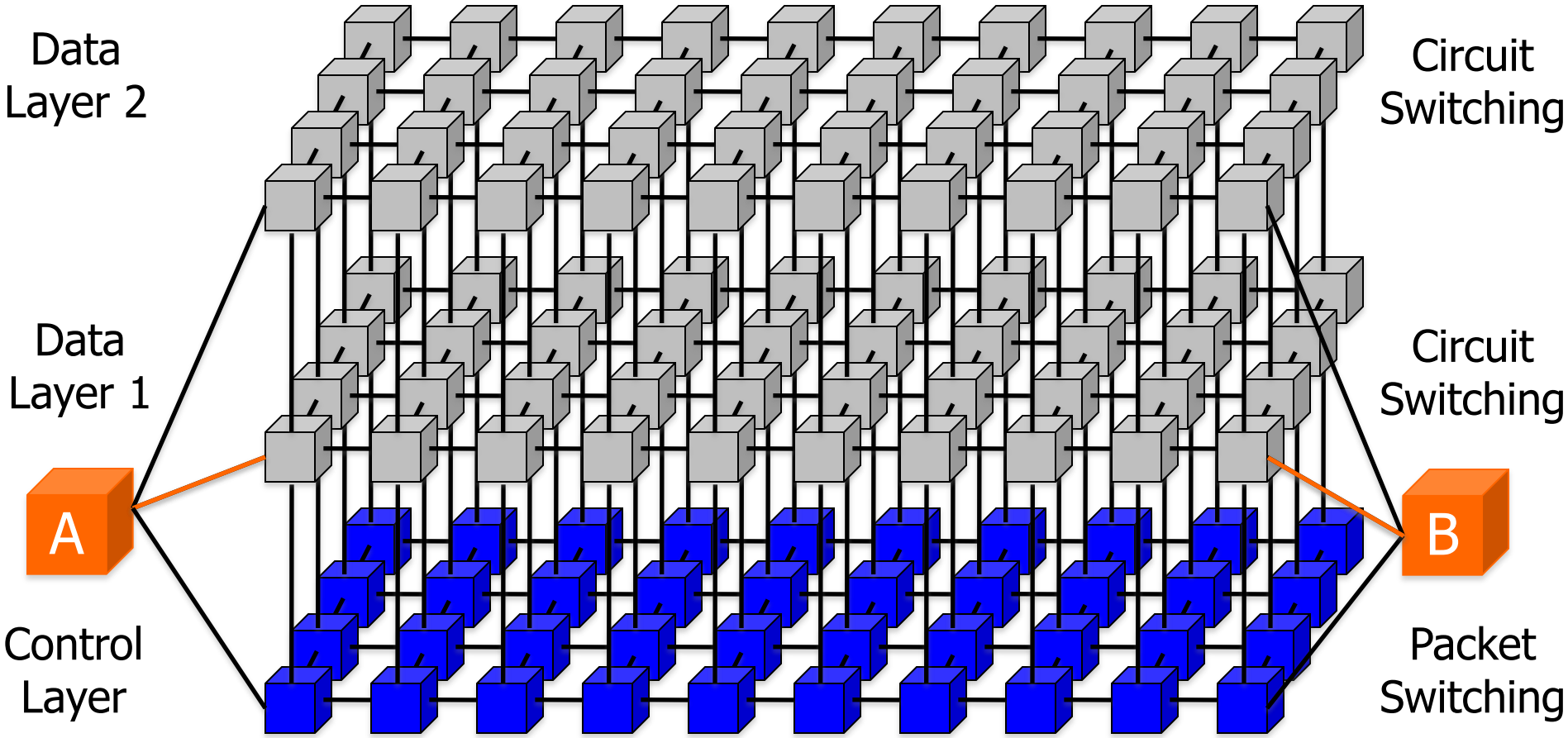
Packet Switched NoC (runtime)	Circuit Switched NoC (runtime)	Multi-Layer NoC (runtime)	Point-to-point (design time)

Area Usage

Point-to-point (direct connection)	Multi-Layer NoC 1 x (routing+arbitration)	Packet+Circuit Switched NoC #Layers x #Routers x (routing+arbitration+mux)

Reduce long wires problem	Point-to-point (no)	Multi-Layer NoC (yes)	Standard NoC (yes)

Multi-Layer NoC without Configuration Controller



Each control router can control the configuration of the data routers in the same XY coordinates