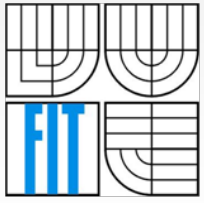
**Annual Doctoral Workshop on Mathematical and Engineering Methods in Computer Science**

**Masaryk University and the Brno University of Technology**

November 14-16, 2008 • Prestige Hotel • Znojmo • Czechia

# Design of FPGA-Based Dependable Systems

**Martin Straka** and **Zdeněk Kotásek**
Department of Computer Systems
Faculty of Information Technology
Brno University of Technology
Czech Republic



# Outline

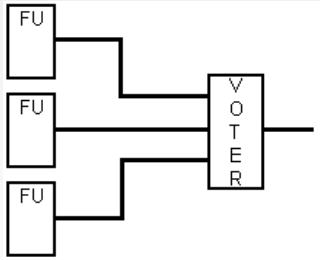
- Introduction
- Motivation and goals of the research
- Methodology for generating on-line checkers
- Technique for generation FT architectures
- Experiments with FT architectures based on checkers
- Results
- Conclusions

# Introduction

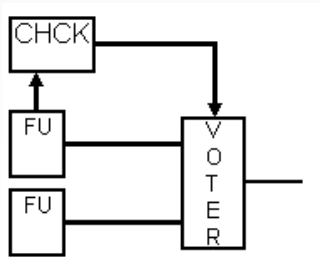
The implementation of electronic systems into FPGA – a challenge to develop new methodologies of Fault Tolerant design.

- The reasoning – why FT systems are implemented into FPGA:
  - The opportunity to re-program the device when fault is present.
  - New possibilities of FT systems design.


- Reconfiguration techniques implemented into FPGA:
  - The reconfiguration is performed dynamically (i.e. at run-time)
  - Partially reconfiguration (i.e. applied only to a portion of the device).
  - Replication of functional unit and other techniques are used.



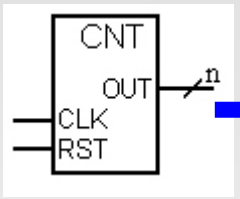

- Technique for improving dependability of systems (FTS):
  - Replication of functional units – like TMR, NMR and Duplex.
  - Redundancy of hardware components (backup).
  - **Checkers.**
  - Security code.
  - 2-wire logic.



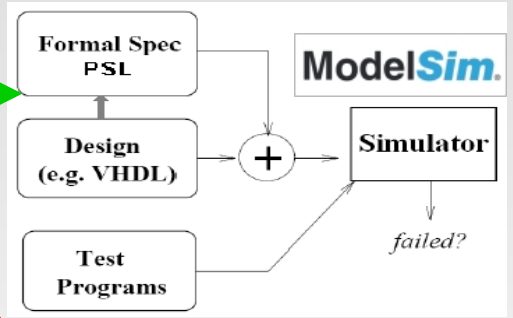
# Introduction

## The idea of on-line checkers

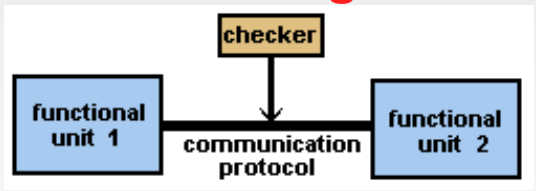
- Checkers in digital system design can be used for several purposes:
  - **Design verification – P**roperty **S**pecification **L**anguage **& FoCs** tool.

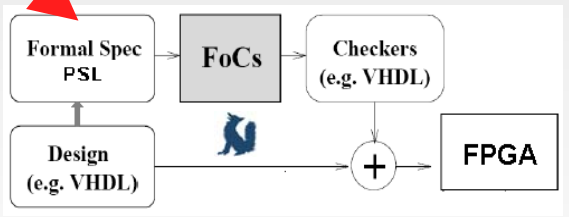



```
vunit checker {
default clock is (clk = '1' and clk'event);
assert always (rst = '0'  &  wr = '1' & full = '1'
                & rd ='0' -> next(wr_error = '1'));
assert never (rst = '0'  -> next(full = '1')); }
```

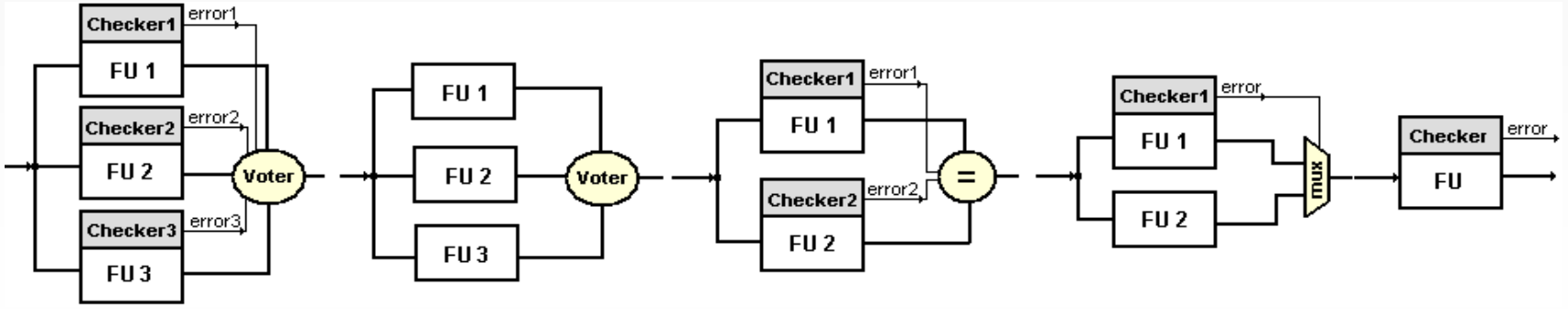



  - **On-line testing** – checkers for diagnostic purpose.



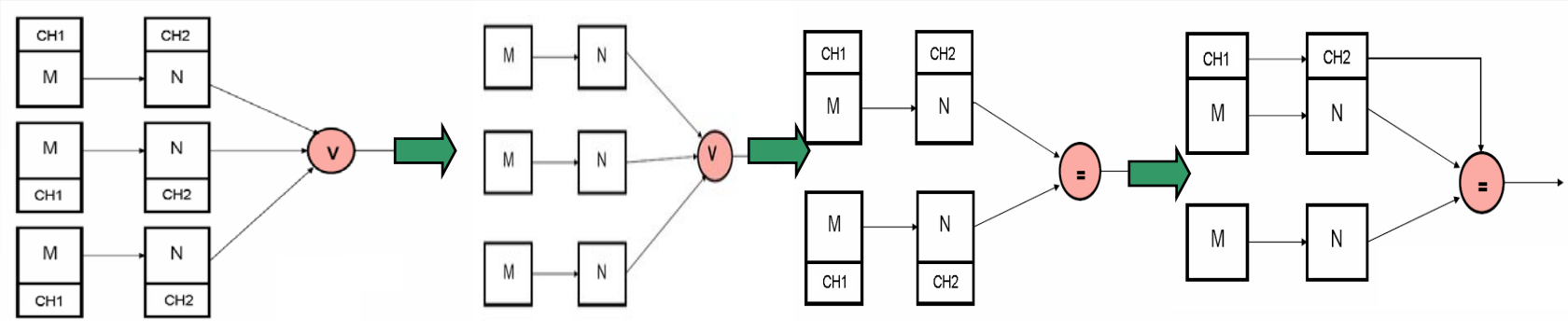




  - **Fault-tolerant system design** – application of checkers in FT architectures.



# Motivation for the research

- The intension to deal with applications in which long lifetime and availability are required, and the resources in FPGA can be possibly exhausted.
- The methodology will allow to develop a sequence of architectures, each architecture covering the required function and equipped with certain diagnostic level.
- The lifetime of these architectures will depend strongly on the area available in FPGA and sources needed to cover the function.
- Creating a methodology of FTS design based on the use of on-line checkers.



# Goals of the research

- To create a methodology, enabling automatic development of on-line checker for diagnostic purposes.

  The goal of the methodology:
  - Checker described in VHDL code for simple circuits synthesizable into FPGA.
  - The use of this approach to increase availability of systems.
  - Comparison our methodology for generating on-line checkers with tools based on PSL.

- To create a technique of FTS design based on the using of checkers and combine this design with TMR and duplex architectures.

  The goal of the technique:
  - The sequence of FT architectures with different level of diagnostic.
  - The use of on-line checkers in this architectures and to increase availability.
  - Evaluation of technique on the basic FT architectures based on TMR and duplex.
  - The evaluation of the results - the number of slices needed to implement of FT architectures and checkers into FPGA.

# Methodology for generating on-line checkers

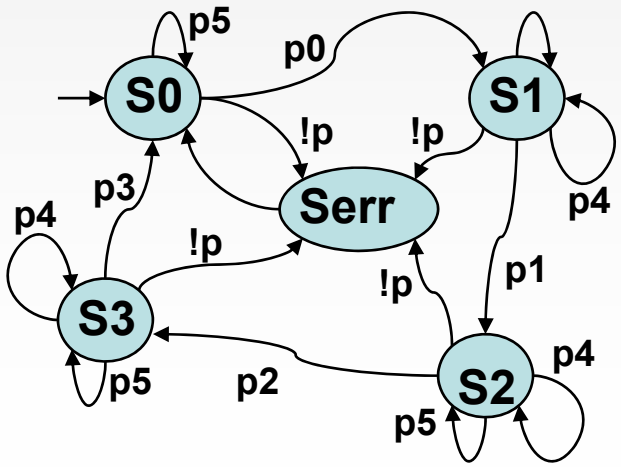
- We are develop formal tool needed to describe functions to be checked by the checker.
- The checker behavior must therefore have features of sequential behavior which can be described by means of FSM.
- For the description of possible faults of digital systems, the dedicated language was defined.
- The definition of language for digital system faults detection therefore arises from the formal description of FSM.

$c0: \; sigA == 1 \text{ and } sigB <> 0 \text{ and } \ldots \text{ and } sigC <> 0;$
$c1: \; (sigC >= 0 \text{ or } sigA < 1) \text{ and } \ldots \text{ and } sigD == 0;$

$P$:
$(S0, c0) : S1;$
$(S1, c1) : S3;$
$(S3, c0) : S0;$
$(S3, c1) : Serr;$
$(Serr) \; : S0;$

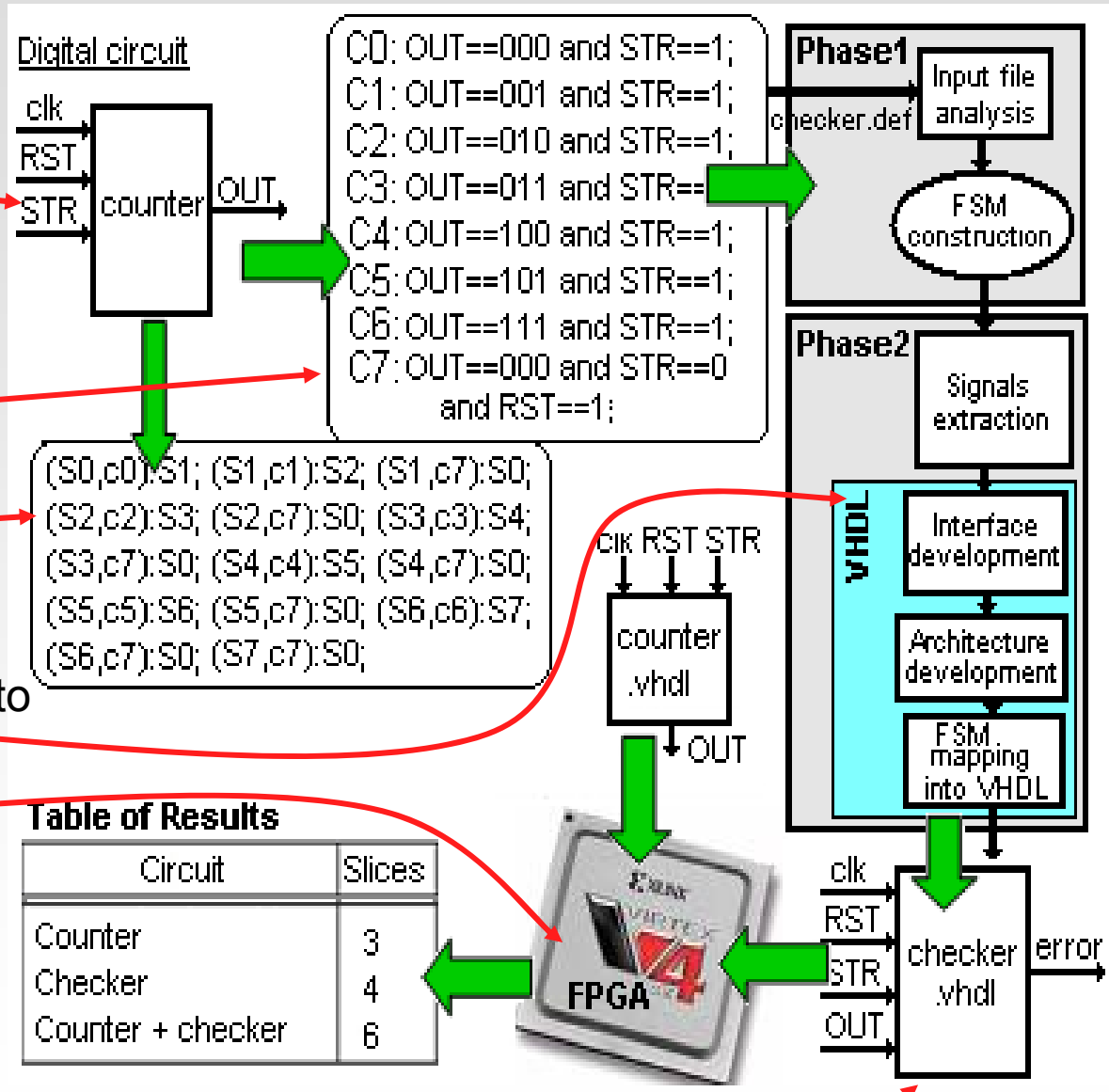
$A = (Q, T, P, S0, Serr)$

$Sn \in Q$
$Cn \in T$
$Serr, S0 \in Q$



# Methodology for generating on-line checkers

1. The component to be checked
2. The conditions to be checked
3. The sequence of states to be checked
4. The conversion of conditions into checker VHDL description
5. Checker description in VHDL
6. The synthesis of the counter and its checker into FPGA



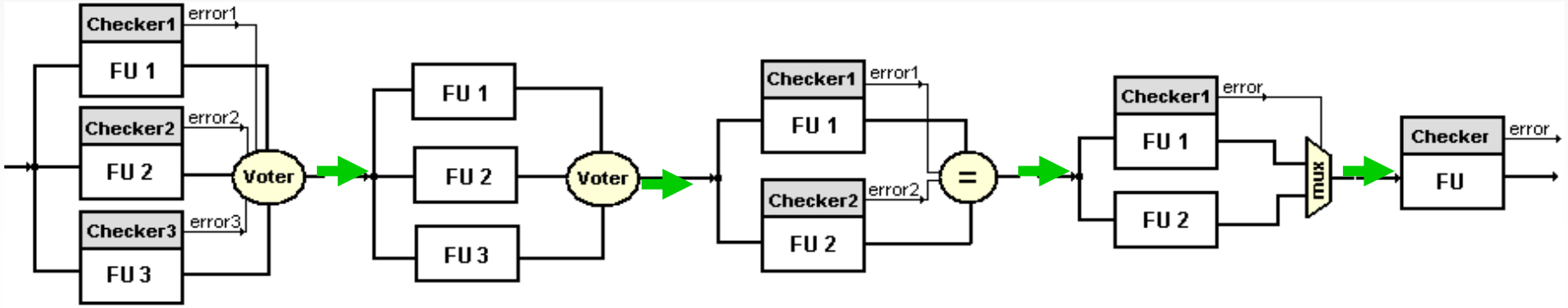
# Technique for generation FT architectures

## Main idea:

- Configuration of architectures with different control level.
- Higher availability and dependability of system, using checkers for this purpose.
- Fault of architecture = the system should switch to the next architecture (lower control level).
- Security voting circuits (voter,comparator).
  - 2-wire logic

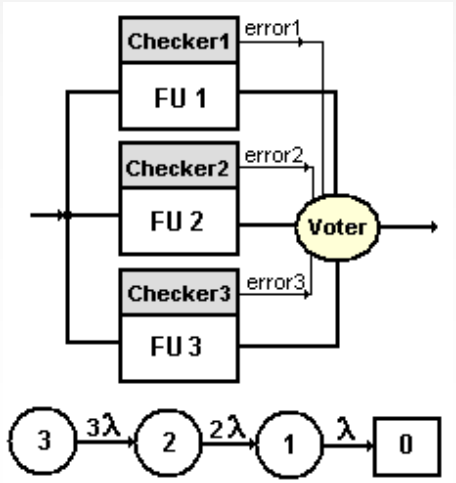
## Two views:

- More architectures in one FPGA and switching between them.
- Partial dynamic reconfiguration – problem with time needed for reconfiguration.

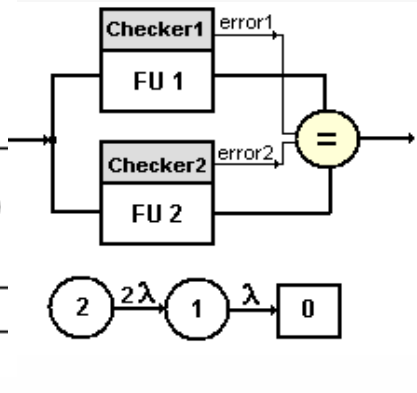


# Technique for generation FT architectures

- For every architecture dependability model must be created.
  (nonrenewable system) – only FU, dependability of voters and comparators not included in dependability model.
- Use of Markov dependability models for this purpose.
- Main parameters:
  - R(t) = probability of fault-tree state of the system
  - Q(t) = failure probability
  - Ts = Mean Times To Failure
  - λ = intensity of failures based on technology



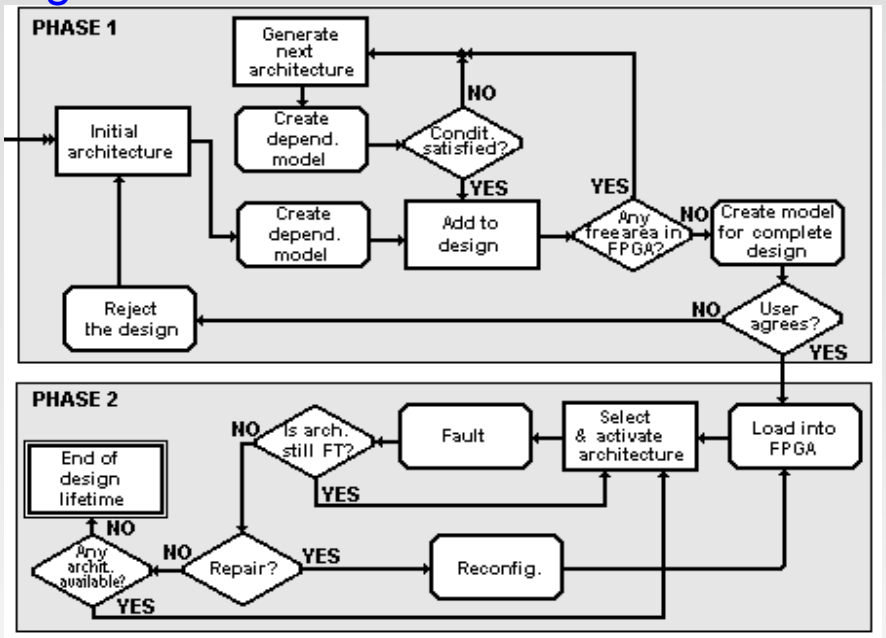

| TMR+3CH | Duplex+2CH |
|---|---|
| $p3'(t) = -3\lambda p3(t)$ | $p2'(t) = -2\lambda p2(t)$ |
| $p2'(t) = 3\lambda p3(t) - 2\lambda p2(t)$ | $p1'(t) = 2\lambda p2(t) - \lambda p1(t)$ |
| $p1'(t) = 2\lambda p2(t) - \lambda p1(t)$ | $p0'(t) = -\lambda p0(t)$ |
| $p0'(t) = -\lambda p0(t)$ | |
| $p3(t) = e^{-3\lambda(t)}$ | $p2(t) = e^{-2\lambda(t)}$ |
| $p2(t) = -3e^{-3\lambda(t)} + e^{-2\lambda(t)}$ | $p1(t) = -2e^{-2\lambda(t)} + e^{-\lambda(t)}$ |
| $p1(t) = 3e^{-3\lambda(t)} - 2e^{-2\lambda(t)} + e^{-\lambda(t)}$ | |
| $R(t) = p3(t) + p2(t) + p1(t)$ | $R(t) = p2(t) + p1(t)$ |
| $Ts = (1/3 + 1/2 + 1) * 1/\lambda$ | $Ts = (1/2 + 1) * 1/\lambda$ |

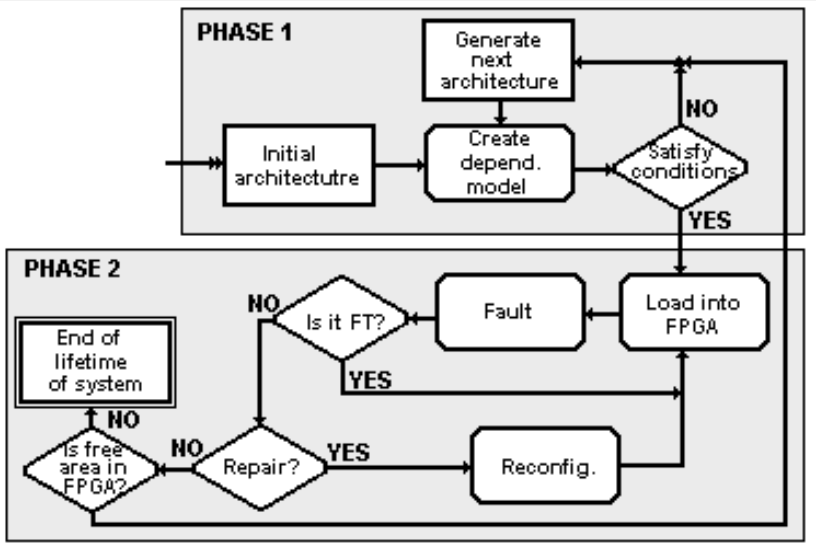


# Technique for generation FT architectures

## More architectures in one FPGA and switching between them.

- User or designer specifies dependability and technology parameters.
- Initial architecture is generated – high diagnostic level.
- Next architectures with different diagnostic level are generated (until complete area in FPGA is not occupied).
- The configuration dependability parameters are compared with the required ones.



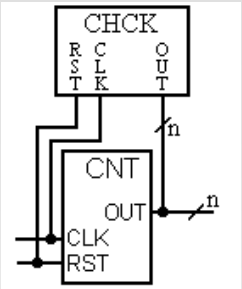



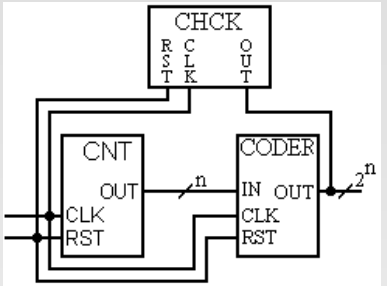

## Partial dynamical reconfiguration

- User or designer specifies dependability and technology parameters.
- Initial architecture is generated – high diagnostic level.
- If fault is detected, the next architecture with different diagnostic is generated.
- The configuration dependability parameters are compared with specified dependability.

# Experiments with FT architectures

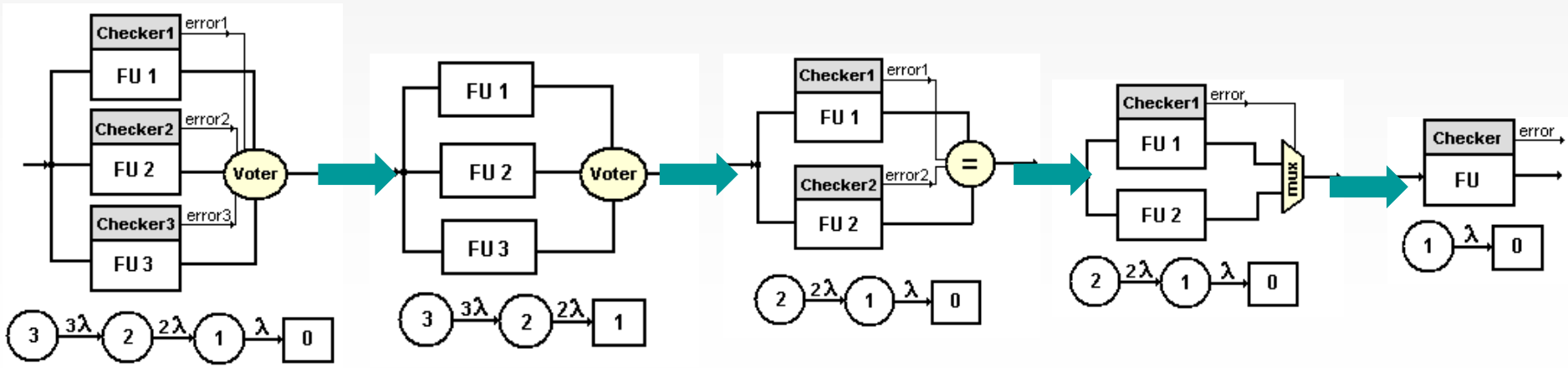
- Our experiments are based on constructing basic FT architectures for the simple digital circuits and their combinations.
  - Counter and decoder
  - Counter and its decoder
  - Serialiser
  - Shift register




- We performed additional experiments to compare FT techniques, namely TMR with duplex techniques based on the use of checkers.

- The availability must be higher.



# Experimental results - checkers

- The experiments with generating of checkers for simple circuits were performed with XILINX FPGA developer platform.
- The components and checkers were synthesized into Virtex5.
- We present the number of slices for checkers based on our methodology.
- We compared the number of slices needed to cover the function and checker implementation.

| Virtex5 - XCV50E | Circuit [slices] | Checker [slices] |
|---|---|---|
| Counter advance | 2 | 7 |
| Counter simple | 2 | 4 |
| Decoder | 4 | 4 |
| Counter+decoder | 5 | 6 |
| Serialiser | 3 | 3 |
| Shift register | 3 | 4 |
| Voter | 7 | - |
| Comparator | 5 | - |

- The comparison of results for checker created by FoCs tool and our methodology for select simple circuits.

| **Virtex5 - XCV50E** circuit | **Checker developed by FoCs** [slices] | **Checker developed by our tool** [slices] |
|---|---|---|
| Counter - full checking | 92 | 7 |
| Counter - only states | 48 | 4 |
| Decoder - full checking | 66 | 5 |
| Shift - only states | 52 | 4 |

# Experimental results for FT architectures

- The experiments for FT architectures were performed with XILINX FPGA platform too.
- The FT architectures were synthesized into Virtex2pro and Virtex5.
- We present the number of slices FT architectures for simple circuits.
- We compared the number of slices needed for FT architectures based on TMR and duplex with checkers.

| Virtex2Pro - XC2VP2 | TMR+3CH | TMR | Duplex+2CH | Duplex+1CH | Simple+1CH |
|---|---|---|---|---|---|
| Circuit | [slices] | [slices] | [slices] | [slices] | [slices] |
| Counter | 36 | 12 | 30 | 18 | 10 |
| Decoder | 28 | 20 | 22 | 17 | 9 |
| Counter+decoder | 31 | 26 | 26 | 25 | 17 |
| Serialiser | 19 | 12 | 15 | 13 | 7 |
| Shift register | 25 | 16 | 20 | 18 | 10 |

| Virtex5 - XCV50E | TMR+3CH | TMR | Duplex+2CH | Duplex+1CH | Simple+1CH |
|---|---|---|---|---|---|
| Circuit | [slices] | [slices] | [slices] | [slices] | [slices] |
| Counter | 21 | 9 | 15 | 11 | 6 |
| Decoder | 27 | 20 | 20 | 18 | 8 |
| Counter+decoder | 30 | 20 | 26 | 22 | 11 |
| Serialiser | 19 | 12 | 14 | 13 | 6 |
| Shift register | 24 | 13 | 18 | 15 | 7 |

# Conclusion

- A methodology for automated design of checkers for verification and diagnostic purposes was presented.
- The checker design methodology is the first step towards the development of the methodology for Fault Tolerant Systems design.
- The basic technique for automated design of FT architectures based on checkers was demonstrated as well.
- Experiments with PSL showed that FoCs tool cannot be used as a tool for the design of on-line checkers for diagnostic purposes to FPGA.

## Future work

- To specify and describe in detail the methodology for generating FT architectures based on on-line checkers.
- To apply the methodology and make experiments on the benchmarks circuit from ISCAS89 set.

# End of presentation

Thank you for your attention