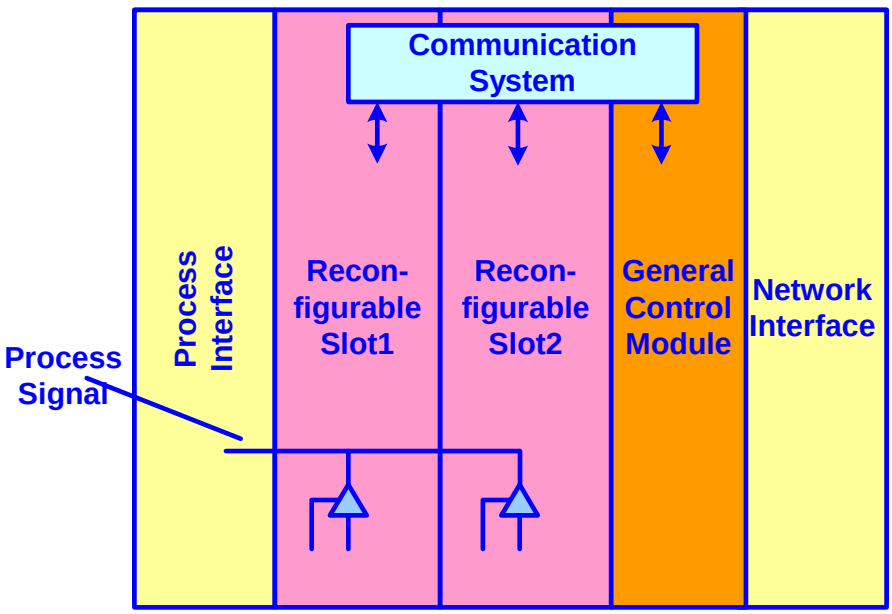
Dynamic Reconfiguration of Mechatronic Real-Time Systems Based on Configuration State Machines

Steffen Toscher; Roland Kasper; Thomas Reinemann
Institute of Mechatronics and Drive Systems
University of Magdeburg, Germany

A reconfigurable controller for mechatronic control systems

- a new type of **adaptable real-time system** on FPGAs using dynamic reconfiguration thus reducing the amount of logic resources
- **function replacement** realizes switching between operating states of the controller
- specification is based on signal flow and Finite State Machines (FSMs)
- **distributed FSMs implemented directly in hardware** control the reconfiguration
- data flow implementation makes use of **bit serial algorithms**



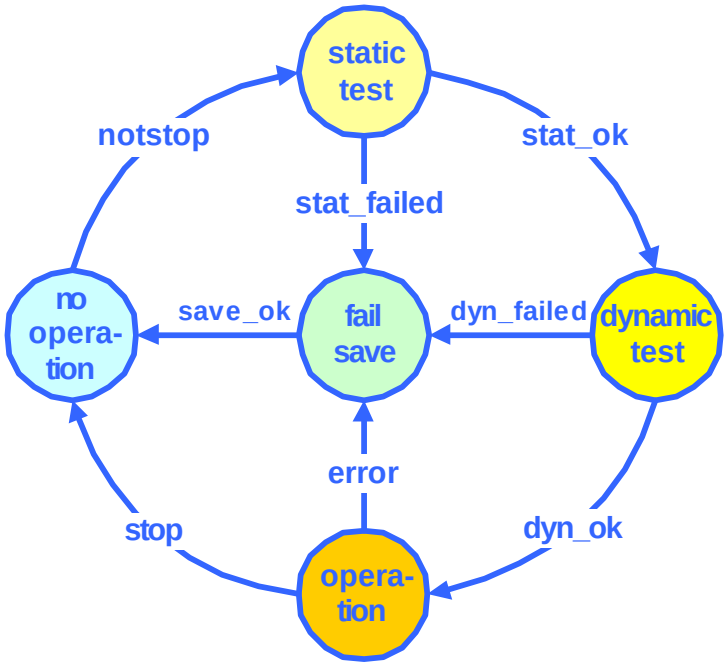
Structure of the reconfigurable controller

imat

Specification of mechatronic control systems

Data Flow ←→ Control Flow

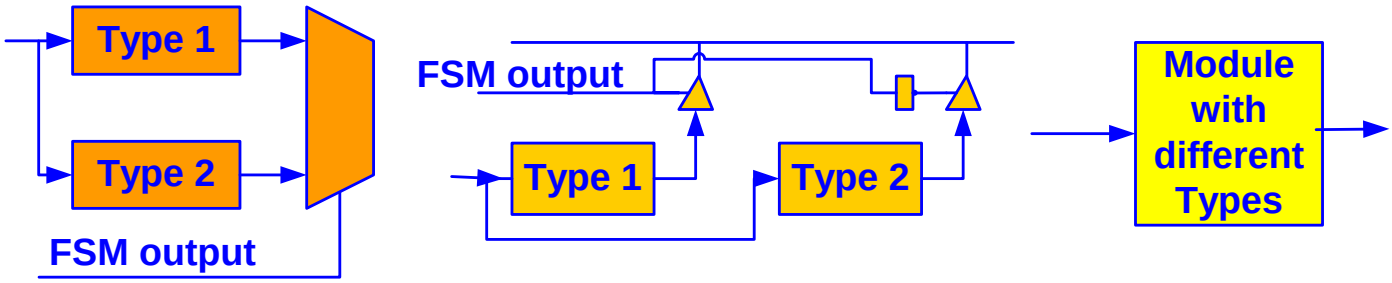
- specification of data flow: **signal diagrams**
- specification of control flow: **FSMs**
- example of structured control flow specification: run-up and operation FSM
- FSMs can be integrated into data flow and vice versa



Run-up and operation FSM

- ***only one active state in a FSM at a given time → good way to determine parts of a complex hardware system that are to be loaded and ready for operation***

Implementing dynamic reconfigurable systems I



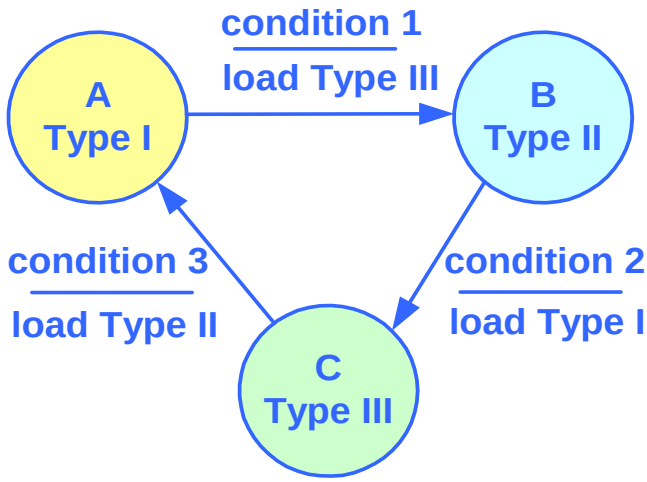
Implementation methods for type switching/function replacement

- implementation uses Xilinx Virtex-II FPGA and Xilinx Modular Design methodology
- a function **block** specified by the data flow can be implemented as a **module** in a slot of the FPGA
- blocks are reconfigurable if they offer the same interface → these blocks represent different **types** of the same module
- ***FSMs** realize the **switching of types** because there can be only one active state/type at a given time*

Implementing dynamic reconfigurable systems II

Module-FSM

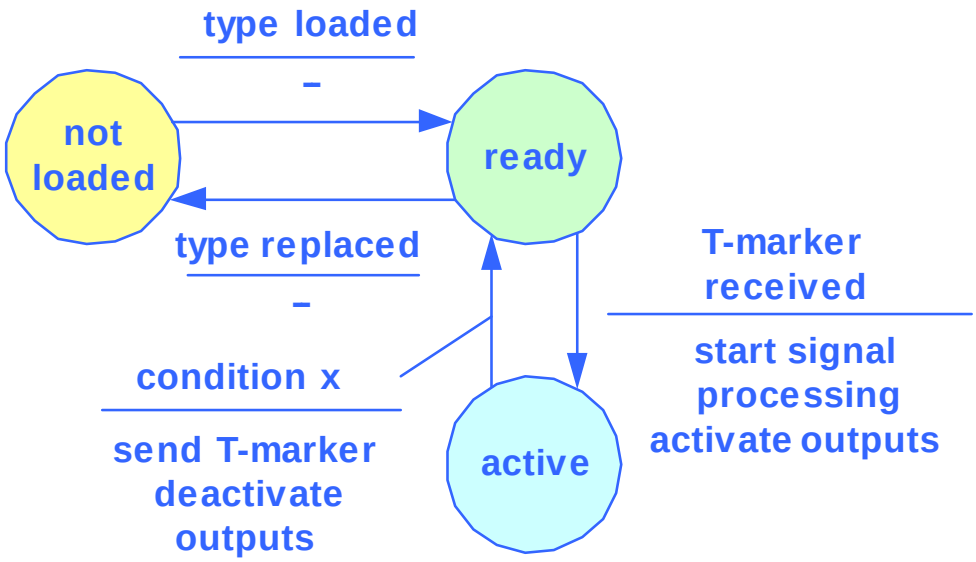
- a **Module-FSM** realizes loading of types depending on process events or physical values
- states of Module-FSMs represent their different types
- **prefetching** of types enables hard real-time operation
- the structure of a Module-FSM is known at compile time
- distributed implementation in the types of a module



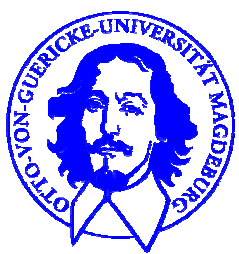
Example of a Module-FSM

Type-FSM

- transfer of the active state of each Module-FSM from one type to the next active state → global **T-Marker** for each type comes into operation
- a **Type-FSM** represents the different states of a type
  - → **not loaded**: stored in external memory, needs no logic
  - → **ready**: prefetched, listening on control input, but not processing data and driving outputs
  - → **active**: loaded, listening on control input, processing data and driving outputs
- the structure of a Type-FSM is known at compile time
- **distributed implementation** of Module-FSM and Type-FSM packed together in the types of a module
- a distributed **communication system** realizes the exchange of the T-Marker between the types/states

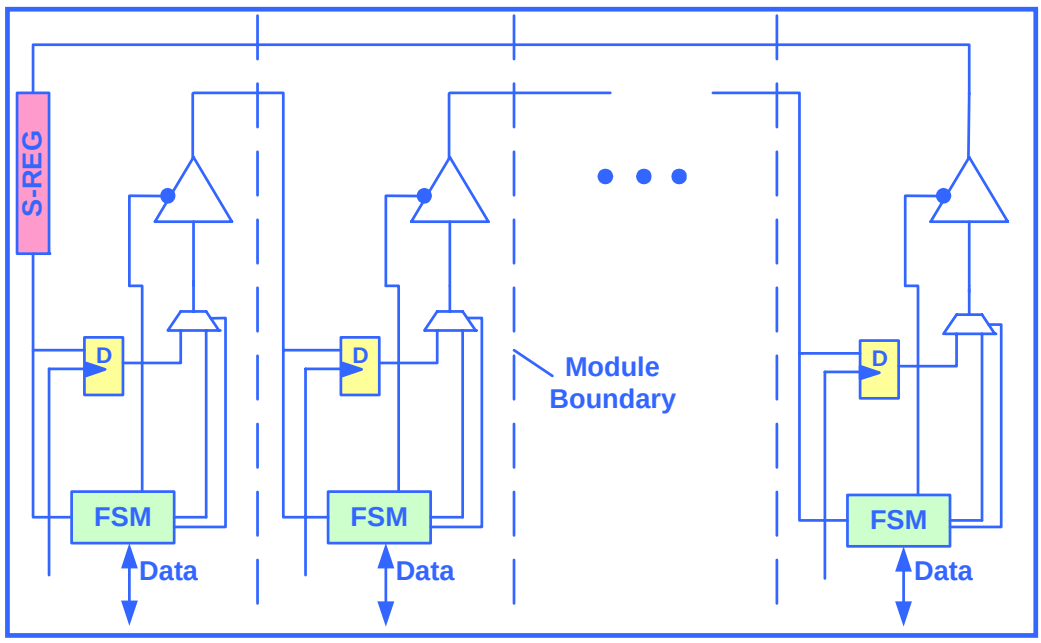


Type-FSM



imat

Implementing dynamic reconfigurable systems III



Communication System

Communication System

- distributed **shift register** and bus macros (long lines)
- all modules build the SR with each storing one bit which is shifted each clock cycle
- local **multiplexers** and registers are implemented inside the bus macros → ***guaranteed communication during reconfiguration***

Control System

- the **General Control Module** (GCM) accesses a static table containing all T-Markers and location and size of types in memory when receiving a T-Marker
- the GCM reads the bit stream to be loaded and sends it to the internal configuration access port **ICAP** → T-Marker is forwarded to activate the new type

Results

- ➔ implementation and test of a **run-up and feed back control system** for an electro motor using a Xilinx FPGA and development tools
- ➔ validation and test of the communication system
- ➔ network interface: **Universal Serial Bus** (USB)
- ➔ process interface: **Delta-Sigma ADCs**
- ➔ **bit serial data processing and transmission** reduces the amount of logic and lines to a minimum → based on a library containing frequently needed processing elements
- ➔ test of reconfiguration via JTAG and ICAP

- ➔ ***direct dynamic reconfiguration of mechatronic hard real-time and control systems***
- ➔ ***reduced logic amount enables more complex algorithms on smaller FPGAs***

OTTO-VON-GUERICKE-UNIVERSITÄT MAGDEBURG

imat