Lorentz center
Workshop @Oort

Safety of Future Systems
Science meets Industry
9 - 13 April 2018, Leiden, the Netherlands

Concurrency Theory meets IoT

Hubert Garavel

Inria Grenoble – LIG

Université Grenoble Alpes

http://convecs.inria.fr

CONVECS

Inria informatics mathematics

LIG

Security issues in the Internet of Things

The Internet of Things (IoT)

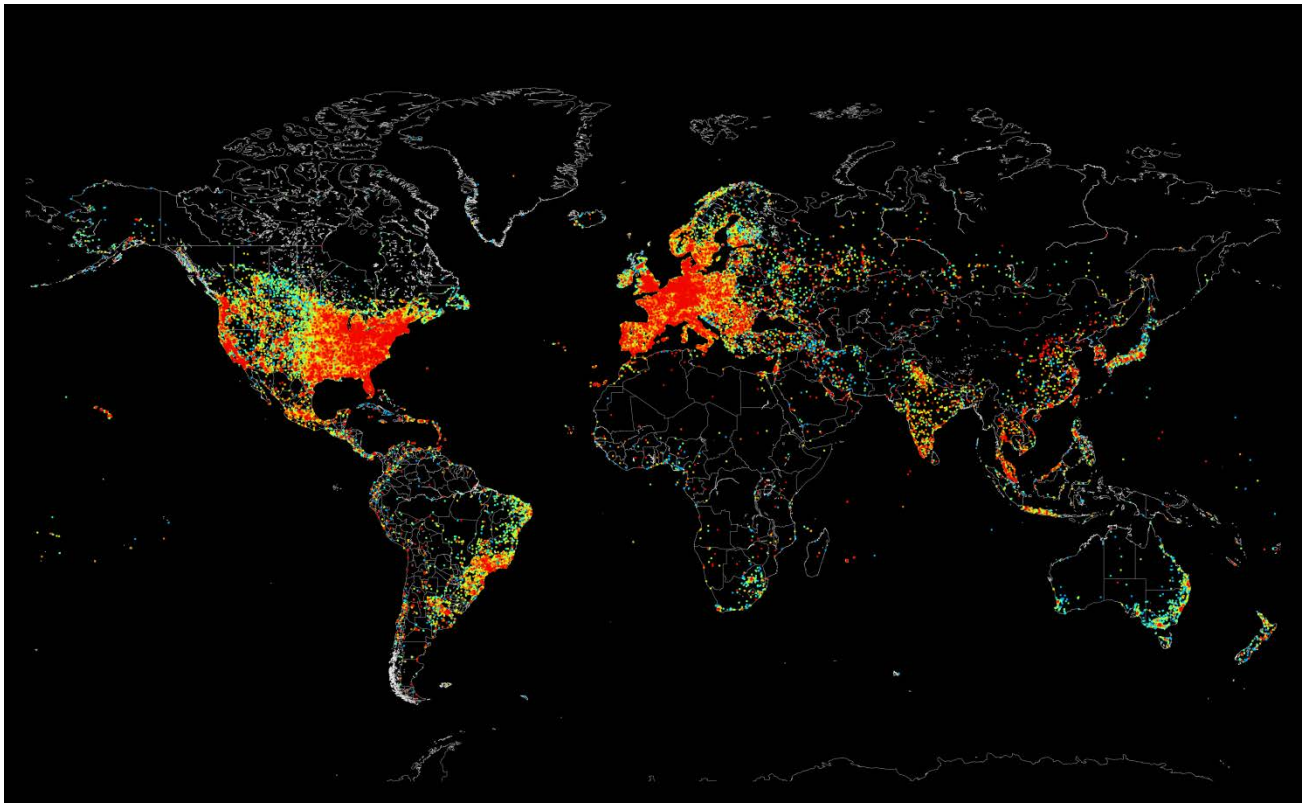
- 2017: 8.4 billion connected objects
- 2020: estimated to 20-50 billion

- Many types of devices:
  - networks
  - smart grids
  - transportation
  - smart homes
  - smart cities
  - etc.

IoT: a nightmare for security

- Same threats as for computers and smartphones but IoT devices have less computing resources:
  - no firewalls
  - no anti-malwares
- Low-cost IoT devices are not well protected:
  - primarily designed to send data, not to be secure
  - not properly maintened (no security updates)
- 87% of the active IoT devices are vulnerable

The "shodan.io" map



- shodan.io: free and commercial web site
- IP addresses of "open" IoT devices

Just a few horror stories

- USA, 2008:
  Researchers take remote control of pacemakers
- Finland, winter 2016:
  Attackers disable heating in two buildings (DDoS)
- IoT also threats the traditional Internet:
  - The Mirai botnet hijacked 500,000 DVR and IP cameras, crashed Dyn (DNS), causing major web sites (Twitter, Netflix, Spotify, etc.) to become unavailable
  - Over the last 3 years, 20% of companies have been attacked from the IoT

Securing the Internet of Things

Two complementary approaches

- "Endpoint" security
  - secure each IoT device
- "Gateway" security
  - secure the network

Worldwide IoT Security Spending Forecast (Millions of Dollars)

	2016	2017	2018	2019	2020	2021
Endpoint Security	240	302	373	459	541	631
Gateway Security	102	138	186	251	327	415
Professional Services	570	734	946	1,221	1,589	2,071
Total	**912**	**1,174**	**1,506**	**1,931**	**2,457**	**3,118**

Source: Gartner (March 2018)

- In this talk: $\Rightarrow$ Endpoint security

The concept of "secure element"

- An IoT device:
  - may have to encrypt its communications
  - should accept security patches (software updates)
  - but only from a trusted source

  $\Rightarrow$ authentication and integrity issues
- Secure element:
  - a tamper-proof processor (or microcontroller)
  - that can perform cryptography
  - that can store secret data (e.g., cryptographic keys)

  Examples: credit cards, SIM cards, NFC devices, etc.

Attacks against secure elements (1/2)

- Attacker's goal: steal cryptographic keys
  - then upload a corrupted firmware

- Brute-force attacks
  - try all possible keys until finding the right one
  - countermeasures: long keys, maximal number of trials
- "Active" attacks
  - flip memory bits using a laser to alter execution
  - countermeasures: circuit shield, redundancy

Attacks against secure elements (2/2)

- "Passive" attacks (side-channel analyses)
  - infer the secret key by measuring:
    - power consumption
    - electromagnetic radiations
    - response time
  - such "template" attacks are efficient
  - machine learning makes them automated and effortless
  - countermeasures:
    - circuit shield
    - randomness: noise, desynchronized traces (random jitter)
    - useless calculations ($\Rightarrow$ increased power consumption)

Industrial case study:

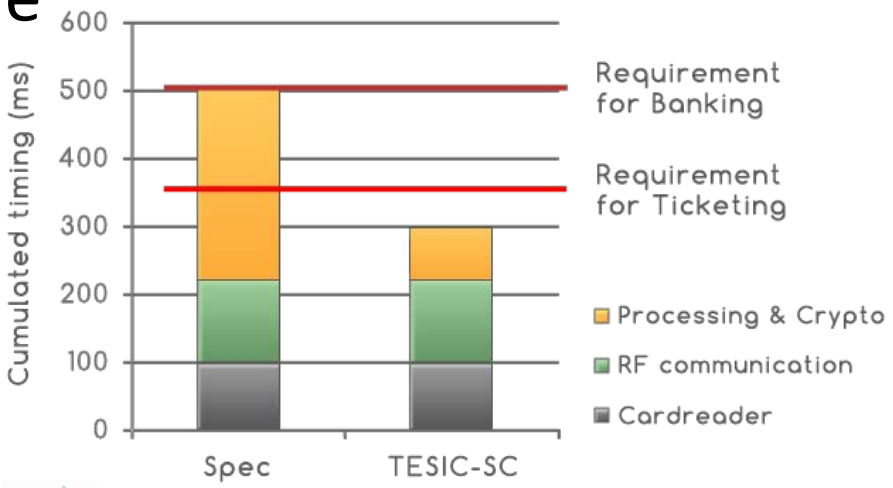
secure elements

The TESIC family of secure slements

- Several chips: TESIC-SC 300, TESIC-SC 500, TESIC-SE
- 16-bit microcontrollers with 32-bit numeric ops
- 256-bit crypto co-processors (AES, DES, ECC, CRC)
- Secure storage / secret file system
- Dual interface: contact and contactless (NFC)
- Markets:
  - banking
  - transportation (open-loop transit fare)
  - e-government (passports, identity documents)
  - Internet of Things (targeted by TESIC-SE)

Tiempo's key technology (1/2)

- Asynchronous logic
  - no central clock
  - handshake communications (~ rendezvous)
- Higher speed
  - each part of the circuit computes as fast as possible
  - no need to wait for the central clock ticks
  - fast switching between active and sleep modes



Tiempo's key technology (2/2)

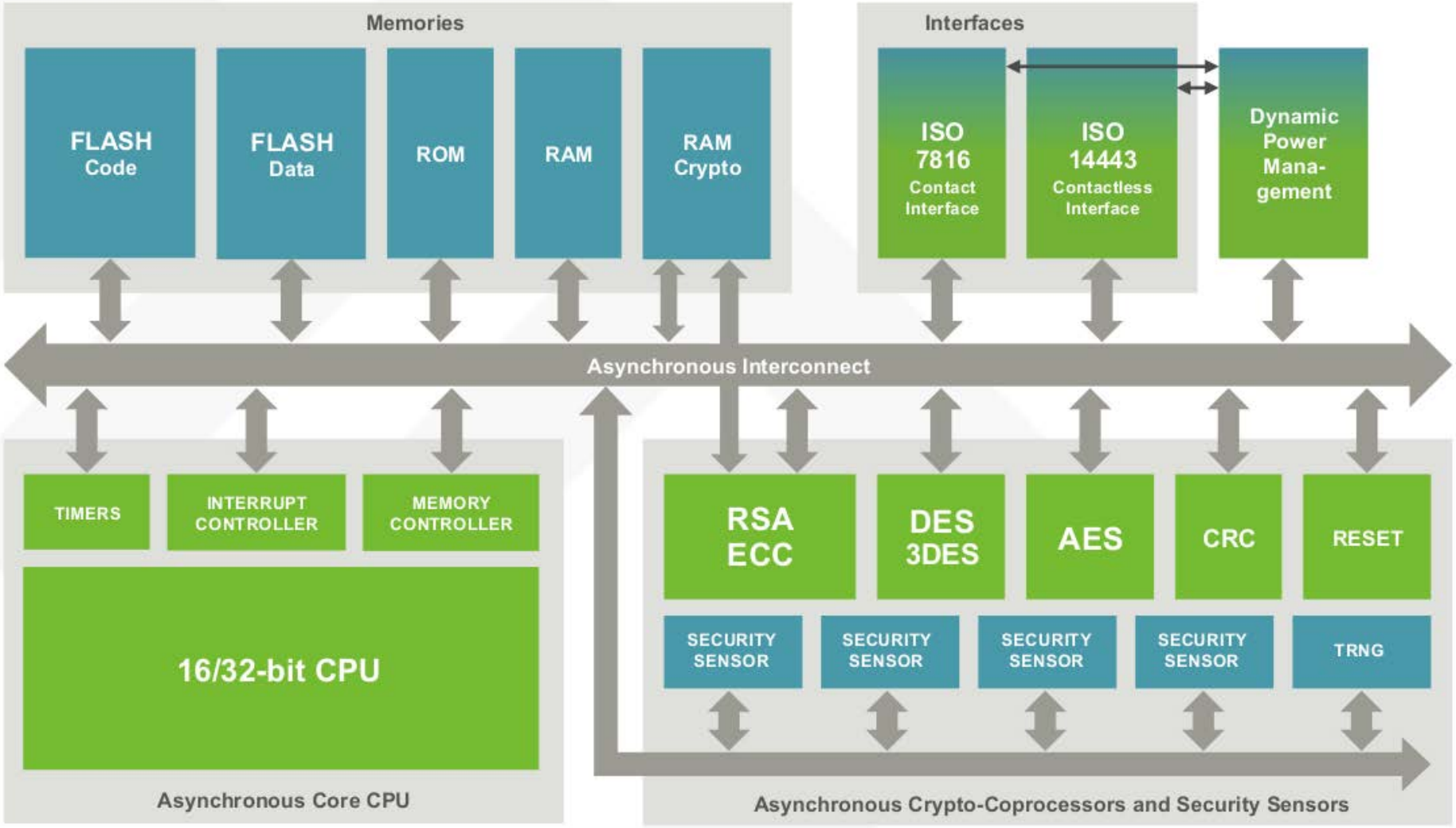
- Lower consumption
  - a central clock needs energy (> 30% of total power)
  - calculations are done only if needed (no idling)
  - battery life expectancy: over 10 years
- Better security
  - a central clock is easy to observe
  - asynchronous logic makes attacks harder
- Third-party certification
  - EMVCo Product Approval
  - EAL 5+ (Common Criteria, ANSSI)

Formal methods for the TESIC

- Why?
  - asychronous circuits are harder to design
  - no commercial verification tools
  - formal methods are required to go above EAL 5+

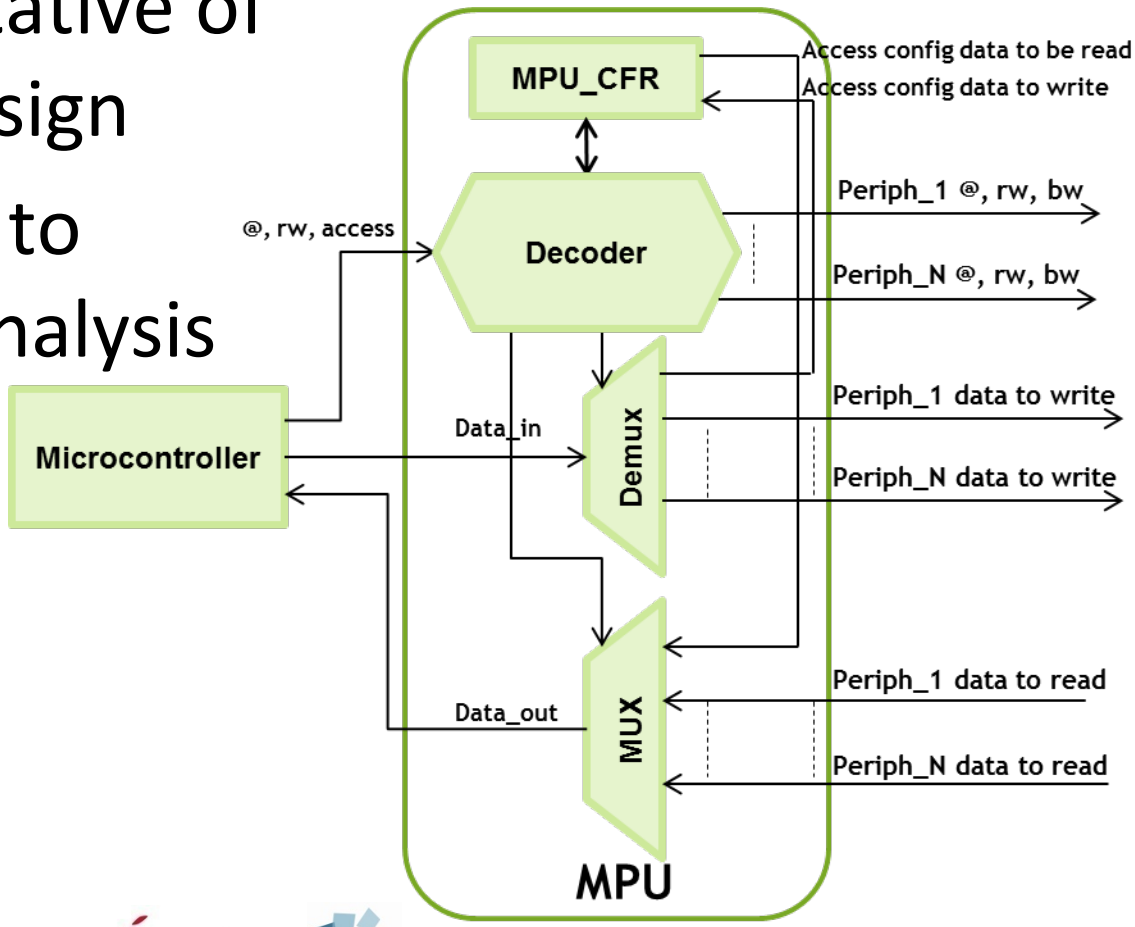
- The SECURIOT-2 French national project:
  - security for the Internet of Things
  - supported by four French innovation clusters
  - leader: Tiempo
  - budget: 5.4 M€

Block diagram of the TESIC chip



TESIC Memory Protection Unit (MPU)

- A crucial block for security
- A good representative of asynchronous design
- Complex enough to deserve formal analysis



Complexity of the TESIC MPU

- 4400 lines of SystemVerilog
  - SystemVerilog: a standard HDL [IEEE 1800-2012]
  - with language extensions for asynchronous circuits
- 8950 lines of LNT
  - LNT: a modern language for replacing LOTOS [ISO 8807:1989]
  - derived from E-LOTOS [ISO 15437:2001]
  - imperative / functional programming style
- MPU: high degree of internal concurrency
  - 146 "main" concurrent processes (themselves concurrent)
  - 250 internal channels
  - 660 tokens in the underlying Petri net

From SystemVerilog to LNT

- SystemVerilog and LNT have been independently designed, but have common features
- Translation done manually, but easy to automate

```
-- main SV module
module address_decoder (
  ch_bit.in add_in,
  ch_data_t.in d_in,
  ch_data_t.out d_out0,
  ch_data_t.out d_out1
);
  always begin
    bit address;
    data_t data;
    fork
      add_in.BeginRead(address);
      d_in.BeginRead(data);
    join
    case (address)
      1'b0: d_out0.Write(data);
      1'b1: d_out1.Write(data);
    end case
    fork
      add_in.EndRead();
      d_in.EndRead();
    join
  end
end module
```

```
-- main LNT process
process main[
  add_in : ch_bit,
  d_in,
  d_out0,
  d_out1 : ch_data_t]
 is
  loop var
    address : bit,
    data : data_t in
      par
        add_in(?address)
      || d_in(?data)
      end par;
      case address in
        0 -> d_out0(data); d_out0
      | 1 -> d_out1(data); d_out1
      end case;
      par
        add_in
      || d_in
      end par
  end var end loop
end process
```

Fighting MPU state-space explosion

- Direct (brute-force) state-space generation fails
- But refined strategies succeed
  - abstraction based on data independence [Wolper-86]
  - compositional minimisation [Fernandez-88]
  - projection and interfaces [Krimm-Mounier-97]

Acces type	Number of inter-mediate LTSs	Largest intermediate LTS		Final LTS			Time
		States	Transitions	States	Transitions	File size	
Co-processor	20	6.6 M	53 M	5.5 M	42 M	92 MB	13 min
MPU_CFR	20	27 M	355 M	27 M	355 M	692 MB	4h33
NVM	20	117 M	862 M	21 M	144 M	296 MB	3h34

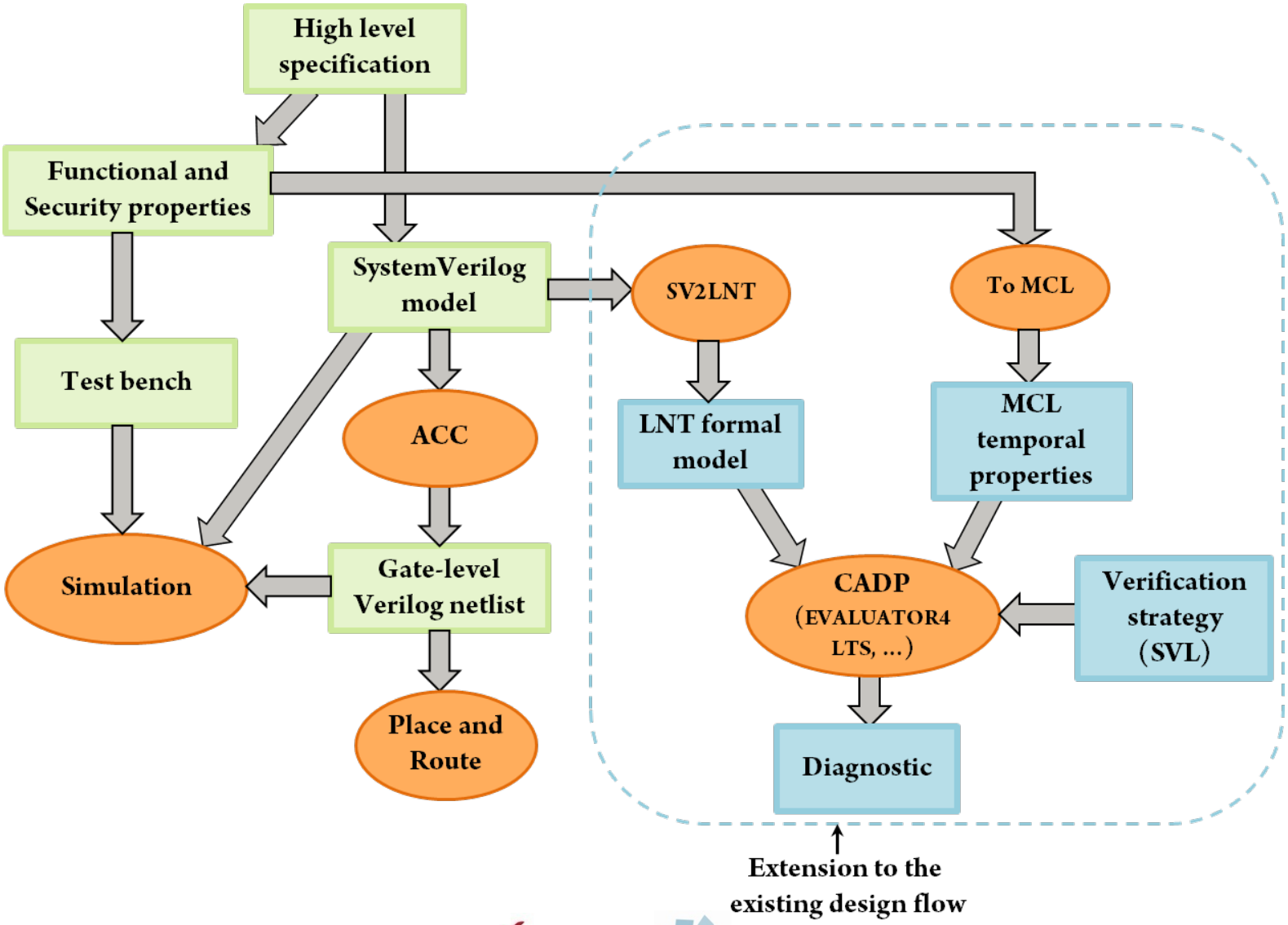
Model checking MPU properties

- 184 properties specified in MCL [Mateescu-97]
- Functional properties:
  - absence of deadlocks
  - absence of livelocks
  - mutual exclusion of reads and writes
  - stimulus-response properties
- Security properties:
  - access-control policies
- Verified by the Evaluator4 model checker of CADP

Beyond the TESIC MPU

- The MPU verification is not a one-shot attempt
- More designs are being verified by Tiempo:
  - Asynchronous Serial Link -- see model at [MCC'2018]
  - DES crypto-processor -- see also [Serwe MARS'2015]
  - etc.
- Integration of CADP in Tiempo's design flow

Tiempo's design flow with verification



Conclusion

Conclusion

- Security of IoT: a major challenge ahead
- The SECURIOT-2 project addresses this problem:
  - Secure elements based on asynchronous logic
  - Formal methods are an enabling technology
- Growing industrial acceptance of formal methods

- More info:

A. Bouzafour, M. Renaudin, H. Garavel, R. Mateescu, W. Serwe. **Model-checking Synthesizable SystemVerilog Descriptions of Asynchronous Circuits**. Proc. IEEE ASYNC'18, Vienna, May 2018