Modélisation de systèmes RFID en vue du test et du diagnostic

Laboratoire de Conception et d'Intégration des Systèmes (LCIS - Valence)

Gilles FRITZ
Vincent BEROULLE
Oum-El-Kheir AKTOUF

prenom.nom@lcis.grenoble-inp.fr

Introduction

- 2004/2009 : Grenoble-INP – ESISAR
  - Electronique, Informatique et Système
    - Informatique des Systèmes Embarqués
- 2008/2009 : Master 2R Micro et Nano Electronique
- 1er année de thèse
  - Sûreté de fonctionnement des systèmes RFID
  - Allocation recherche ministérielle de l'ED EEATS de Grenoble

Plan

- Présentation du LCIS
- Sûreté de fonctionnement des systèmes RFID
- Simulation d'un système RFID HF
- Conclusion

Plan

- **Présentation du LCIS**
- Sûreté de fonctionnement des systèmes RFID
- Simulation d'un système RFID HF
- Conclusion

Présentation du LCIS

- Laboratoire de Conception et d'Intégration des Systèmes
- 4 groupes de recherche
  - Systèmes complexes coopérants (COSY)
  - Modélisation, analyse et commande des sytèmes dynamique (MACSY)
  - Systèmes optoélectroniques et radiofréquence (ORSYS)
  - **Conception et test des systèmes embarqués (CTSYS)**

Présentation de CTSYS

- Test et validation de conception des systèmes matériels
  - Sûreté de fonctionnement des systèmes RFID
- Testabilité et test des systèmes logiciels critiques
- Diagnostic en ligne dans les systèmes embarqués

Plan

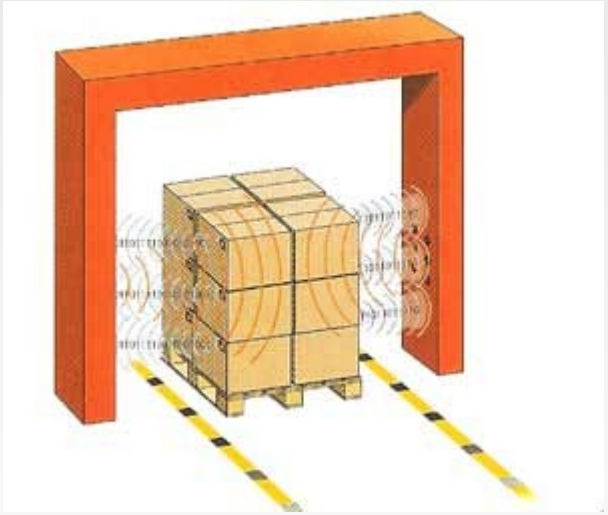
- Présentation du LCIS
- **Sûreté de fonctionnement des systèmes RFID**
- Simulation d'un système RFID HF
- Conclusion

Plan

- Présentation du LCIS
- **Sûreté de fonctionnement des systèmes RFID**
- Simulation d'un système RFID HF
- Conclusion

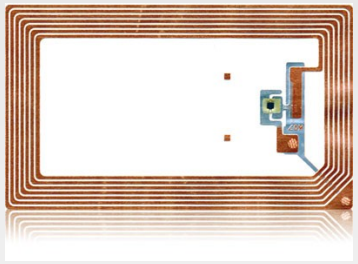
Sûreté de fonctionnement des systèmes RFID

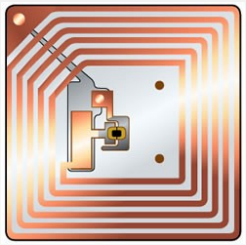
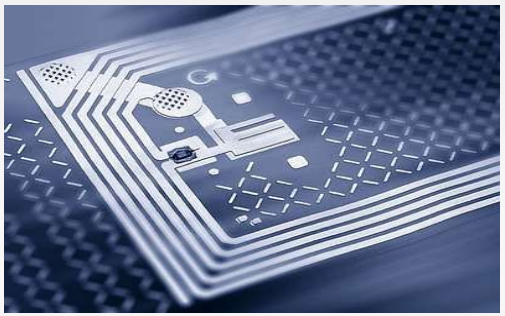
- Concept et applications
  - Identifier un objet par radiofréquence
  - Sans contact ni vision directe



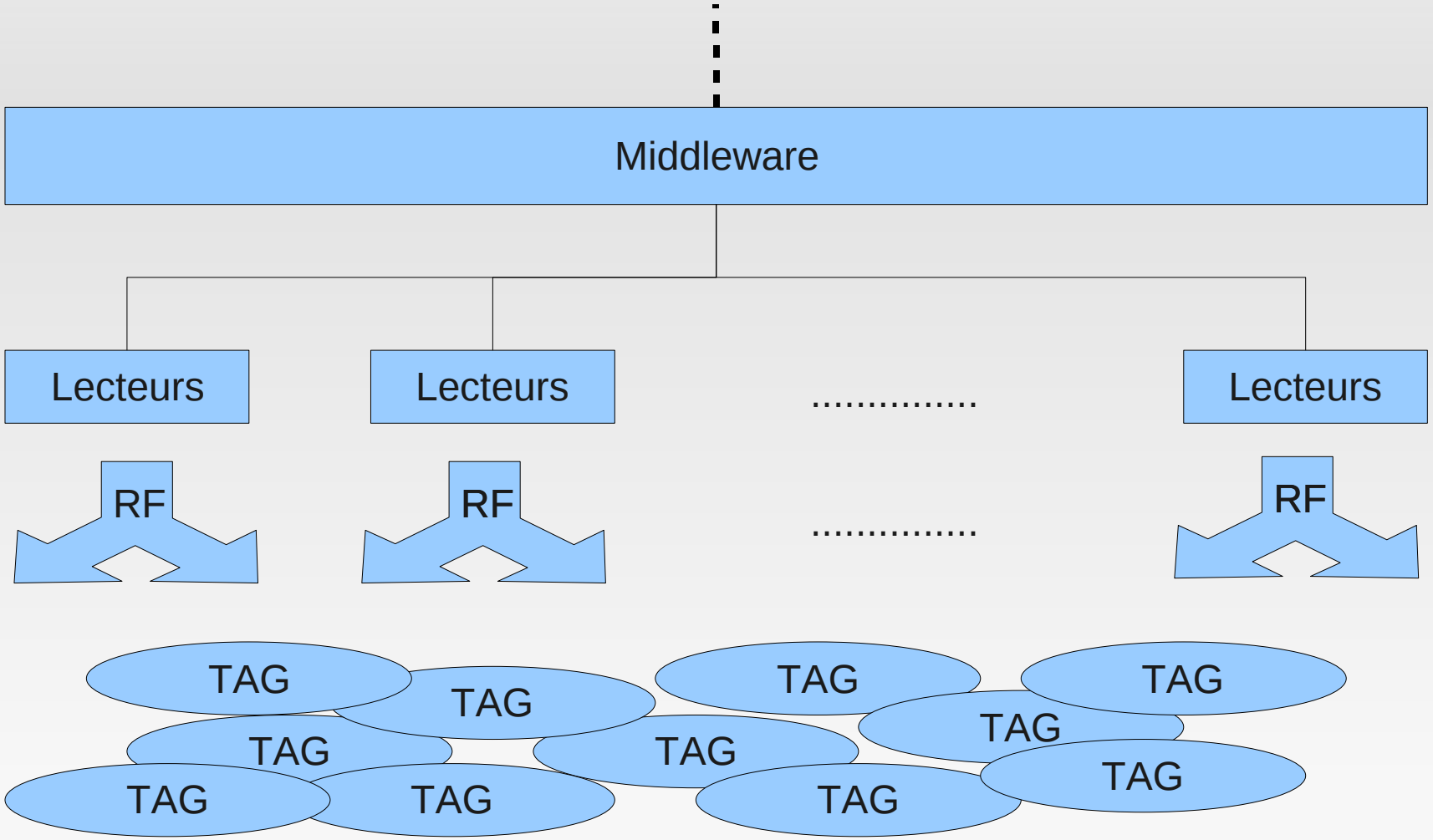
Sûreté de fonctionnement des systèmes RFID

- Deux composants matériels principaux
  - *Transpondeur*, *étiquette* ou plus souvent *tag*
  - *Interrogateur*, *base station* ou *lecteur*

Sûreté de fonctionnement des systèmes RFID



Sûreté de fonctionnement des systèmes RFID

- Intergiciel ou Middleware
  - Lien entre les couches basses logicielles et l'application
    - Abstraction matérielle
    - Gestion des lecteurs
    - Filtrage
    - Base de données

Sûreté de fonctionnement des systèmes RFID

- Beaucoup d'études sur la sécurité-confidentialité
- Peu sur la fiabilité, la disponibilité

Pourquoi un simulateur?

- Automatiser l'analyse des dysfonctionnements
- Formaliser le modèle
- Faciliter l'observation des dysfonctionnements locaux sur le système complet

Simulateur RFID

- RIFIDI (HF,UHF) [Pramari]
- Fosstrak (UHF) [Auto-ID Labs]
- RFIDSim (UHF) [MIT]
- « Flexible simulation and prototyping for rfid designs » (HF, UHF) [Vienna University of Technology]

Simulateur RFID

- Simulation du protocole de communication lecteur/tag
- Simulation fonctionnelle
- Simulation HF et/ou UHF
- Simulation des signaux analogiques
- Simulation des composants numériques

Simulateur RFID

- Simulateurs existants inadéquats
  - Pas de simulation de la partie analogique et numérique en même temps
  - Pas d'injection de fautes
  - Non interfaçable avec un middleware

Plan

- Présentation du LCIS
- Sûreté de fonctionnement des systèmes RFID
- **Simulation d'un système RFID HF**
- Conclusion

Démarche

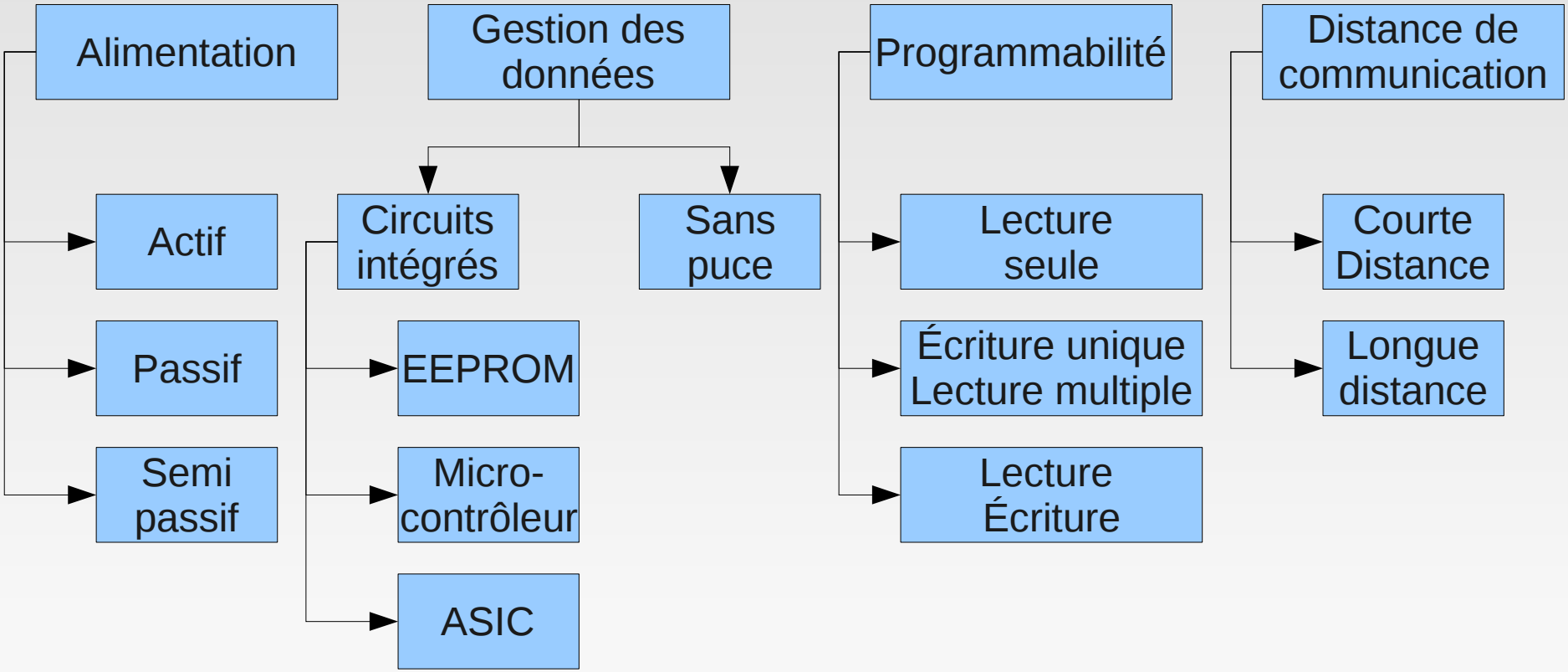
- Spécification fonctionnelle des différents composants [norme ISO-15693]
- Analyse des modes de défaillances
- Développement d'un simulateur

Démarche

- **Spécification fonctionnelle des différents composants [norme ISO-15693]**
- Analyse des modes de défaillances
- Développement d'un simulateur

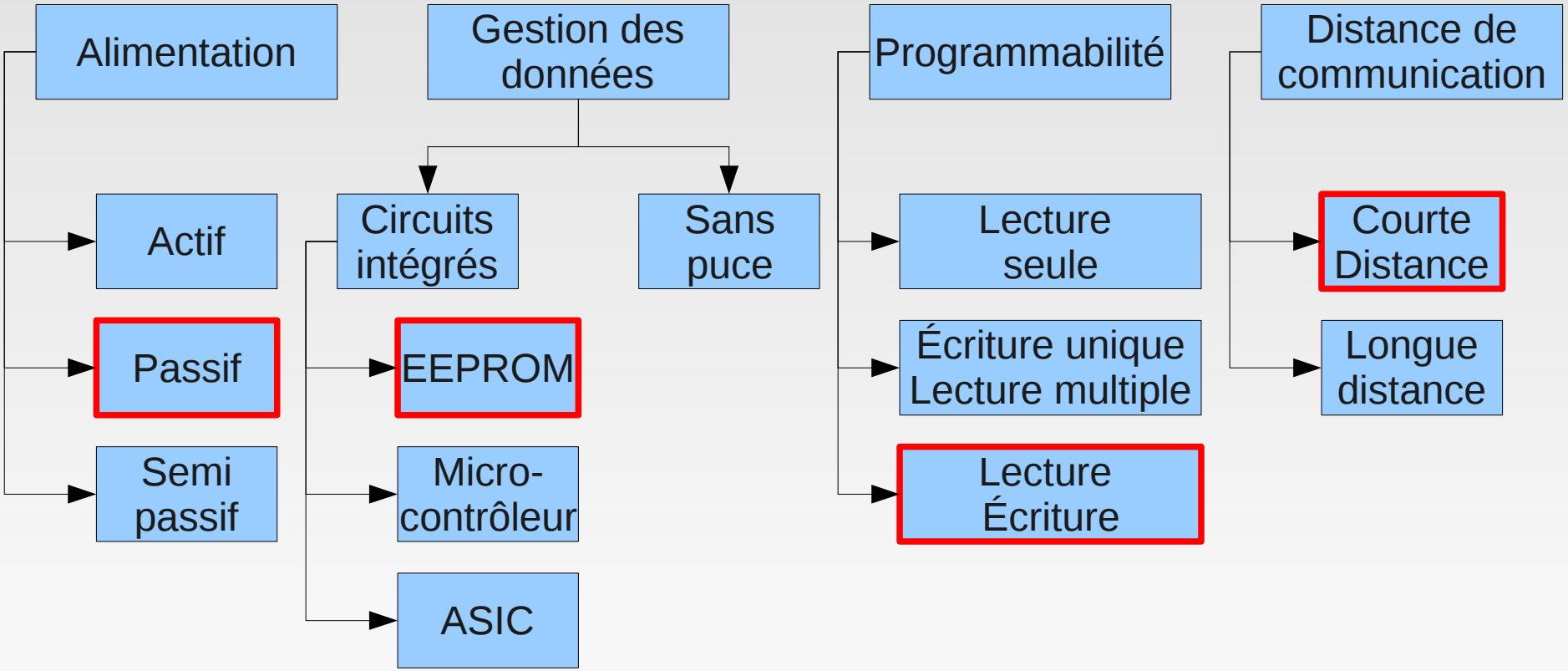
Spécification fonctionnelle des différents composants

Tag



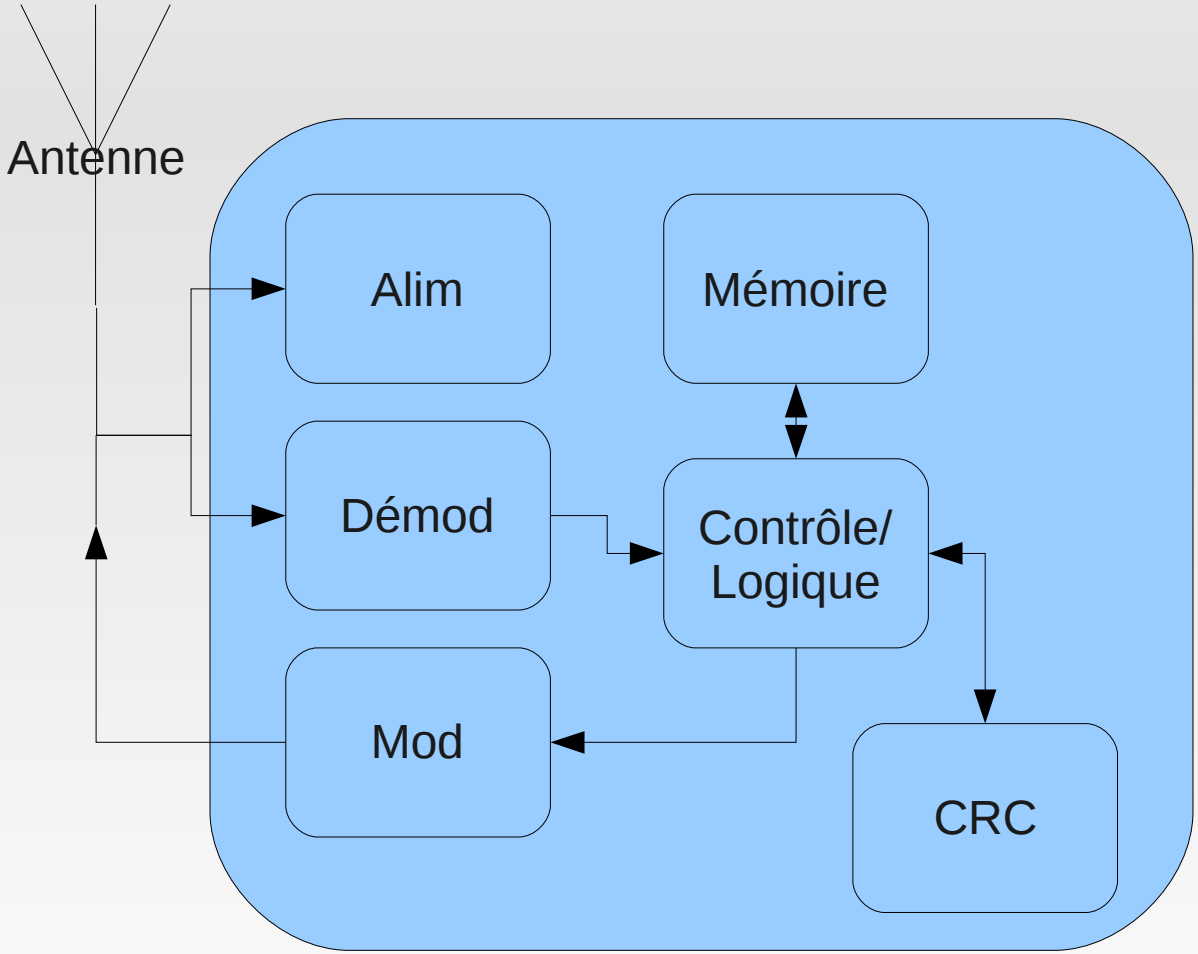
Spécification fonctionnelle des différents composants

Tag



Spécification fonctionnelle des différents composants

Tag



Spécification fonctionnelle des différents composants

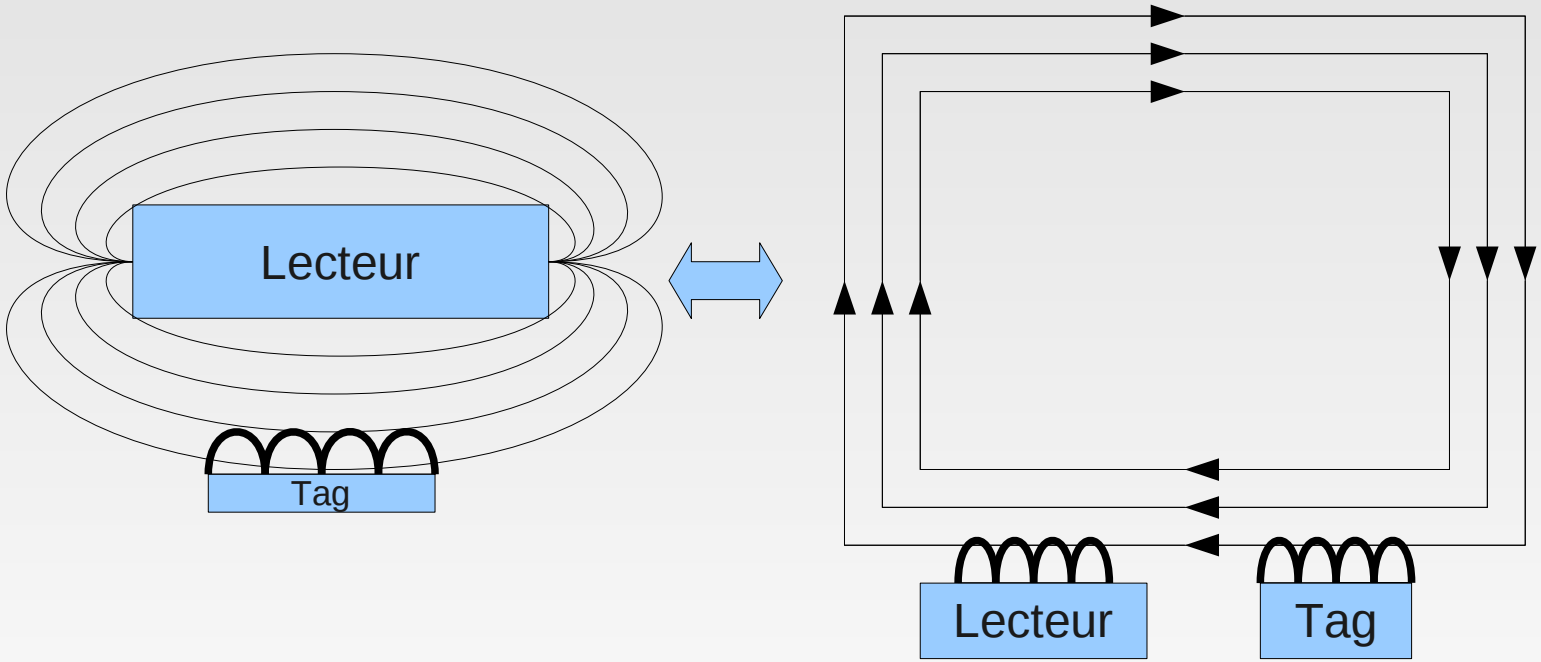
- Lien RF
  - Permet
    - d'alimenter le tag lorsqu'il est passif
    - d'échanger des informations
  - Dépend
    - de la fréquence de fonctionnement

Spécification fonctionnelle des différents composants

- Lien RF
  - HF
    - 13,56MHz
    - Longueur d'onde < distance tag-lecteur
    - Fonctionnement similaire à un transformateur
    - Le tag répond en changeant de charge

Spécification fonctionnelle des différents composants

- Lien RF
  - HF



Spécification fonctionnelle des différents composants

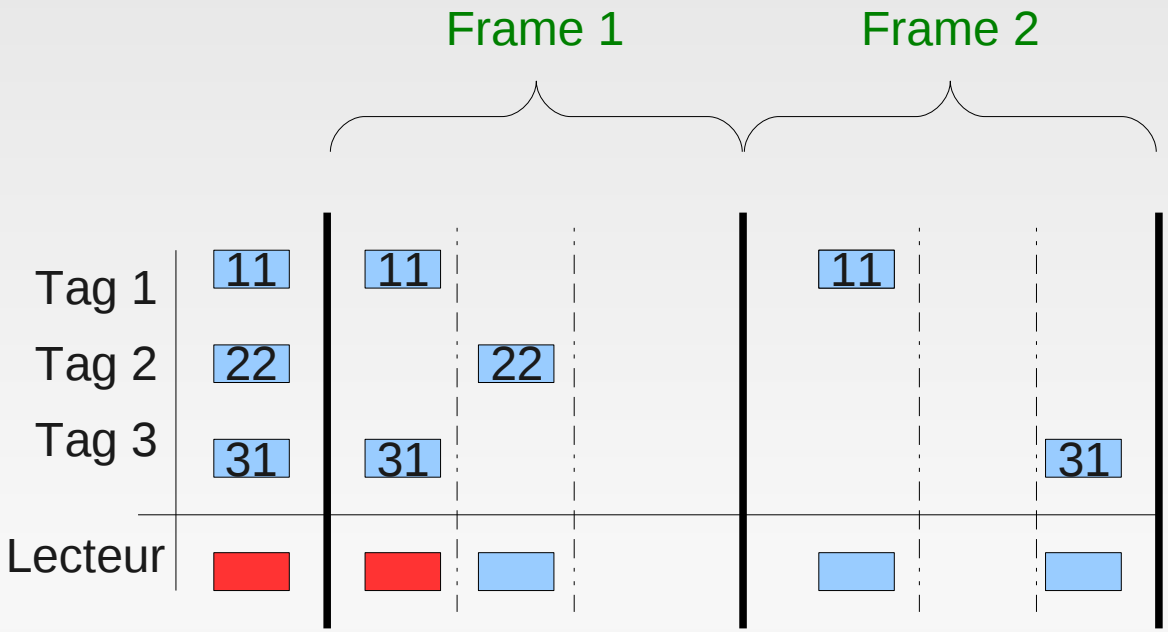
- Lien RF
  - Collision
    - Plusieurs tags ou lecteurs émettent en même temps
    - Rend le transfert d'informations impossible
    - Mise en place de protocoles anti-collisions

Spécification fonctionnelle des différents composants

- Lien RF
  - Collision
    - Cas d'étude : Frame Slotted Aloha
      - Algorithme probabiliste
      - Discrimination des tags par leur ID

Spécification fonctionnelle des différents composants

- Lien RF
  - Collision
    - Exemple de protocole : Frame Slotted Aloha

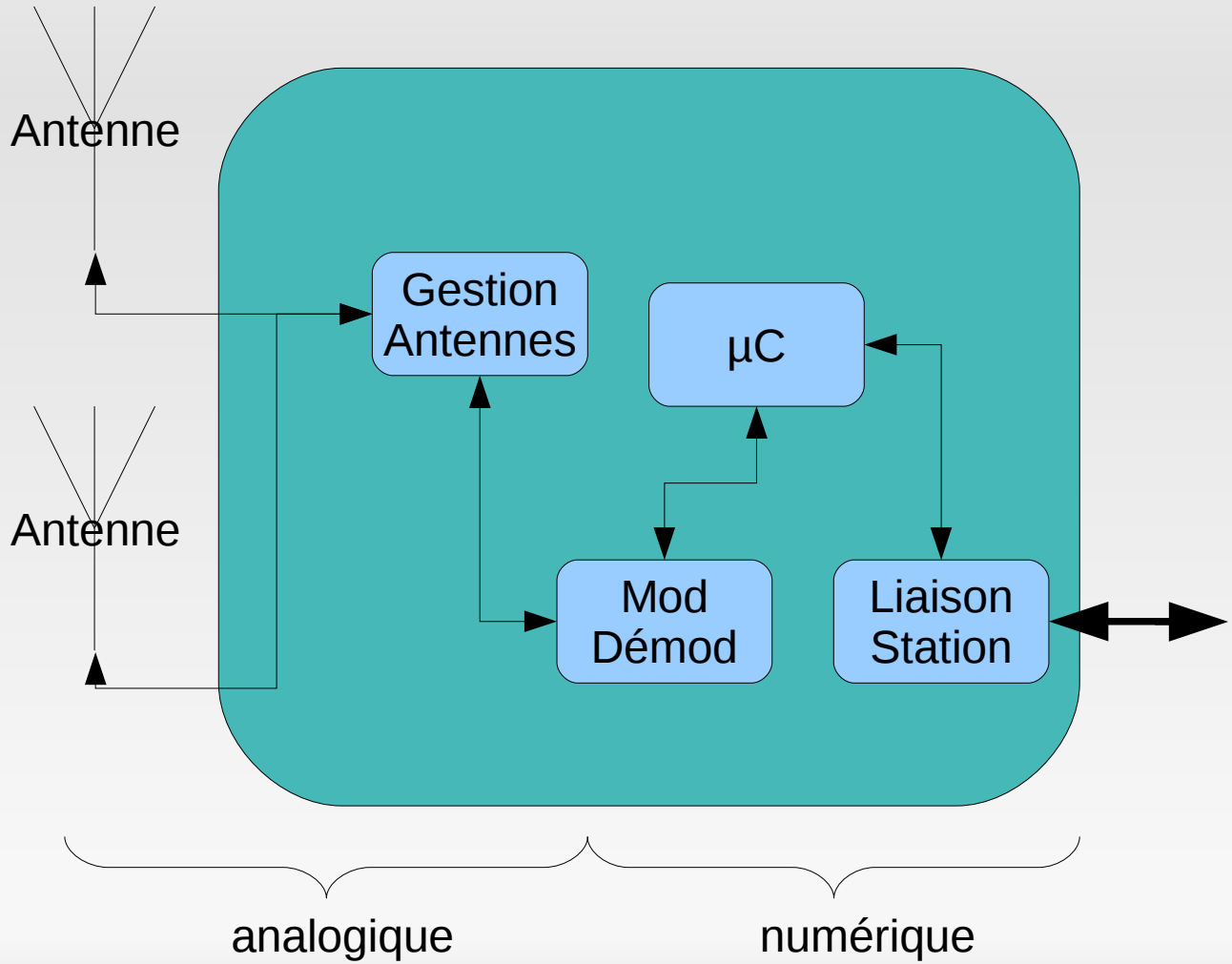


Spécification fonctionnelle des différents composants

- Lecteur
  - Même démarche que pour le tag : analyse des composants standards d'un lecteur

Spécification fonctionnelle des différents composants

- Lecteur



Démarche

- Analyse des différents composants
- **Analyse des modes de défaillances**
- Développement d'un simulateur

Analyse des dysfonctionnements

- Pour chaque composant
  - Identifier les dysfonctionnements possibles
  - Impact sur le système

Analyse des dysfonctionnements

- Exemple : Mémoire ROM
  - Tableau de blocs
  - Blocs = n bits
  - 1 action : lire un champ

Analyse des dysfonctionnements

- Exemple : Mémoire ROM
  - Lecture fausse
    - Permanent
    - Intermittent
  - Aucune lecture
    - Permanent
    - Intermittent

Analyse des dysfonctionnements

- Exemple : Mémoire ROM
  - Lecture fausse :
    - 1 bit faux
    - 1 bloc faux
    - 1 bloc juste mais pas le bon
    - Tout faux

Analyse des dysfonctionnements

- Classification
  - Logique
  - Communication
  - Gestion des données

Démarche

- Analyse des différents composants
- Analyse des modes de défaillances
- **Développement d'un simulateur**

Conception du modèle

- Choix des outils
  - Java, C, C++, etc.
    - Permet de simuler le côté fonctionnel
    - Ne permet pas de simuler les différents composants et leurs interactions
  - VHDL, VHDL-AMS
    - Permet de simuler les composants électroniques
    - Permet de simuler les composants analogiques
    - Impossible de simuler les parties logicielles

Conception du modèle

- Choix des outils
  - SystemC
    - Co-design
    - Permet de simuler des composants numériques
    - Permet de simuler des composants logiciels
    - Permettra de simuler des composants analogiques (extension SystemC-AMS)

Conception du modèle

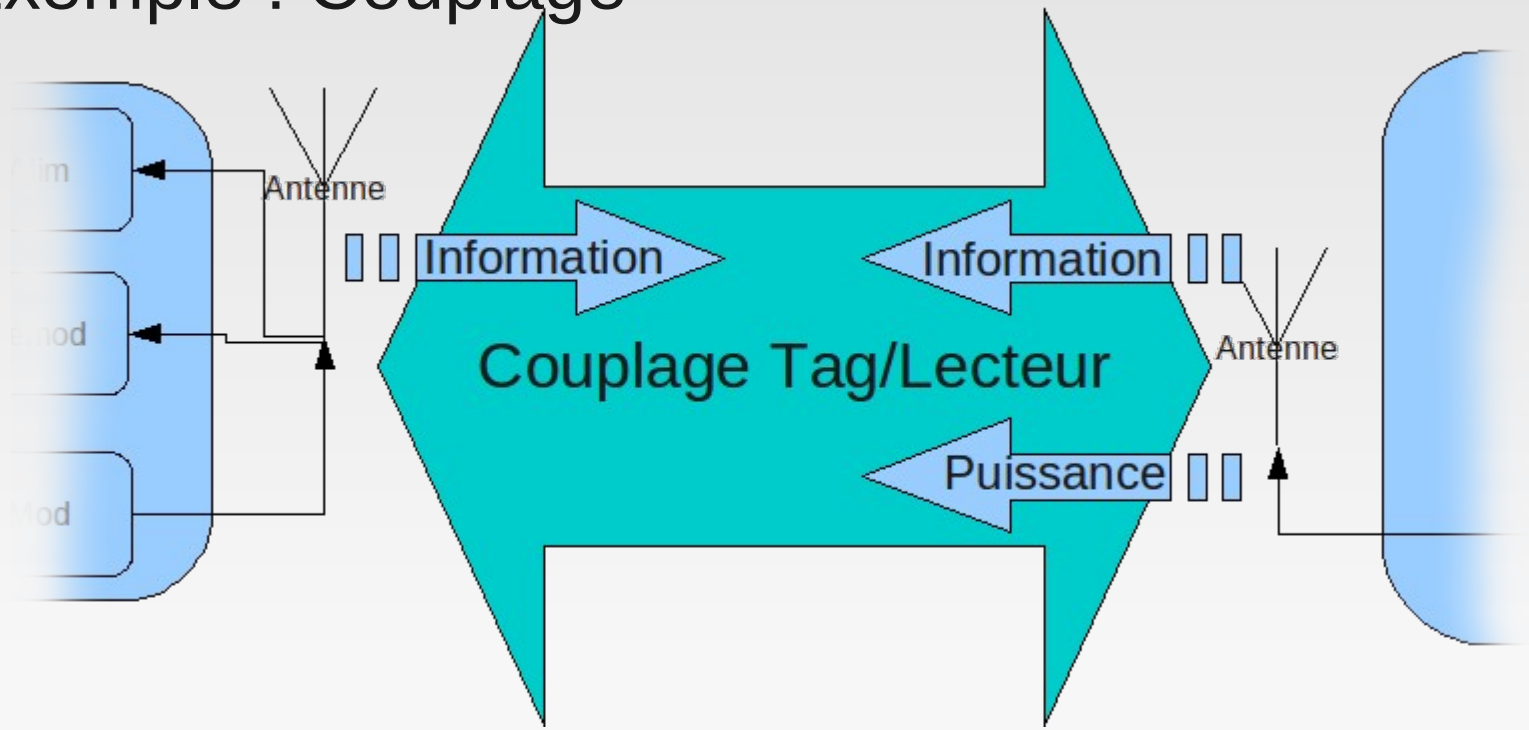
- Spécifications détaillées
  - Découpage hiérarchique du modèle
    - 3 composants principaux
      - Tag
      - Lecteur
      - Couplage
    - Sous-composants
      - Mémoire
      - CRC
      - Logique de contrôle
      - ...

Conception du modèle

- Spécifications détaillées
  - Pour chaque composant et sous-composant
    - Fonction
    - Interactions avec les autres composants
      - Entrées
      - Sorties
    - Scénarios de tests unitaires
  - Scénarios de tests d'intégrations partielles
  - Scénarios de tests d'intégration complète

Conception du modèle

- Spécifications détaillées
  - Exemple : Couplage



Conception du modèle

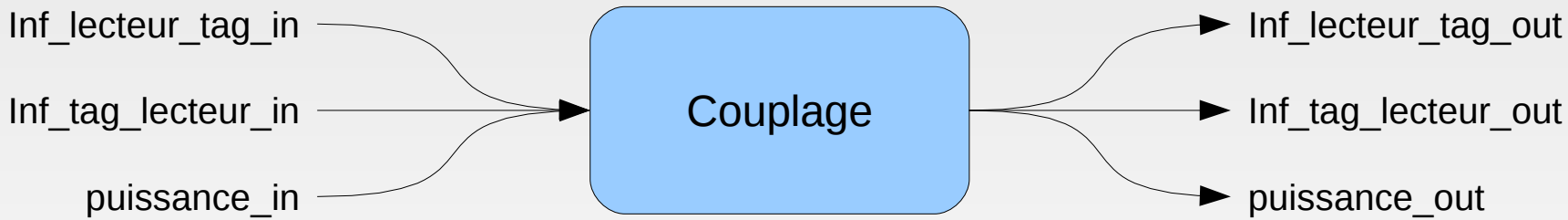
- Spécifications détaillées
  - Exemple : Couplage
    - Informations
      - Données binaires brutes
      - Modulation
      - Codage bit
      - Débit
      - Start of frame
      - End of frame

Conception du modèle

- Spécifications détaillées
  - Exemple : Couplage
    - Fonctions
      - Transmettre la puissance du lecteur vers le tag
      - Transmettre les informations du lecteur vers le tag
      - Transmettre les informations du tag vers le lecteur

Conception du modèle

- Spécifications détaillées
  - Exemple : Couplage



Validation du modèle

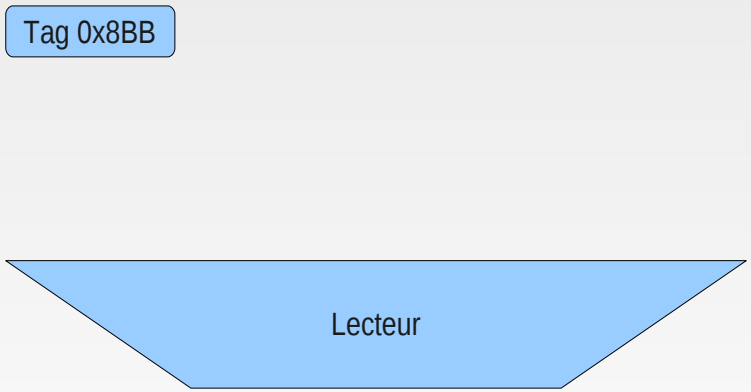
- Validation fonctionnelle
- Validation de certains signaux internes
- Scénario de validation
  - 3 tags qui apparaissent et disparaissent en fonction du temps
  - 2 tags avec début d'identifiant identique
  - 1 lecteur effectuant une lecture toutes les *ms*

Validation du modèle

Scénario de validation

$0ms \rightarrow 4{,}5ms$

```
1000021 ns 1 tag détecté : 0x8BB
2000021 ns 1 tag détecté : 0x8BB
3000021 ns 1 tag détecté : 0x8BB
4000021 ns 1 tag détecté : 0x8BB
```



Validation du modèle

Scénario de validation

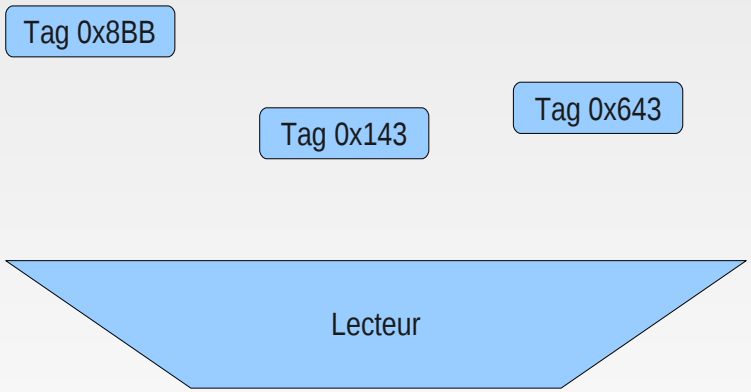
4,5ms → 6,5*ms*


```
1000021 ns 1 tag détecté : 0x8BB
2000021 ns 1 tag détecté : 0x8BB
3000021 ns 1 tag détecté : 0x8BB
4000021 ns 1 tag détecté : 0x8BB
5000021 ns Collision
5000116 ns   1 tag détecté : 0x143
5000335 ns   1 tag détecté : 0x8BB
6000021 ns Collision
6000116 ns   1 tag détecté : 0x143
6000335 ns   1 tag détecté : 0x8BB
```

Validation du modèle

Scénario de validation

6,5ms → 8,5*ms*




```
5000335 ns   1 tag détecté : 0x8BB
6000021 ns Collision
6000116 ns   1 tag détecté : 0x143
6000335 ns   1 tag détecté : 0x8BB
7000021 ns Collision
7000116 ns   Collision
7000335 ns   1 tag détecté : 0x8BB
7000525 ns     Collision
7000963 ns       1 tag détecté : 0x143
7001089 ns       1 tag détecté : 0x643
8000021 ns Collision
8000116 ns   Collision
8000335 ns   1 tag détecté : 0x8BB
8000525 ns     Collision
8000963 ns       1 tag détecté : 0x143
8001089 ns       1 tag détecté : 0x643
```

Validation du modèle

Scénario de validation

8,5ms → 10*ms*


```
6000116 ns   1 tag détecté : 0x143
6000335 ns   1 tag détecté : 0x8BB
7000021 ns Collision
7000116 ns   Collision
7000335 ns   1 tag détecté : 0x8BB
7000525 ns     Collision
7000963 ns       1 tag détecté : 0x143
7001089 ns       1 tag détecté : 0x643
8000021 ns Collision
8000116 ns   Collision
8000335 ns   1 tag détecté : 0x8BB
8000525 ns     Collision
8000963 ns       1 tag détecté : 0x143
8001089 ns       1 tag détecté : 0x643
9000031 ns Pas de tag
10000031 ns Pas de tag
```

Validation du modèle

Fichier log

Messages envoyés et reçus par le lecteur

```
<-- at 5 ms :: (1, [(36, 1, {0, }, {}, 0xbf4e, )], 1, 2, 0, 2, 100, )
--> at 5000020 ns :: Collision
<-- at 5000021 ns :: (1, [(4, 1, {0, }, {}, 0xbc75, )], 1, 2, 0, 2, 100, )
<-- at 5000052 ns :: (0, [(0, 0, {}, {}, 0, )], 1, 0, 0, 0, 100, )
<-- at 5000083 ns :: (0, [(0, 0, {}, {}, 0, )], 1, 0, 0, 0, 100, )
<-- at 5000114 ns :: (0, [(0, 0, {}, {}, 0, )], 1, 0, 0, 0, 100, )
--> at 5000115 ns :: (3, [(0, 0, {0, },
                {224, 171, 137, 171, 205, 239, 1, 35, }, 0x9c0f, )], 2, 1, 1, 3, 40, )
```

Validation du modèle

Fichier log

Messages envoyés et reçus par le lecteur

Start of Frame

<-- at 5 ms :: (**1**, [(36, 1, {0, }, {}, 0xbf4e,)], 1, 2, 0, 2, 100,)
--> at 5000020 ns :: Collision
<-- at 5000021 ns :: (1, [(4, 1, {0, }, {}, 0xbc75,)], 1, 2, 0, 2, 100,)
<-- at 5000052 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000083 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000114 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
--> at 5000115 ns :: (3, [(0, 0, {0, },
{224, 171, 137, 171, 205, 239, 1, 35, }, 0x9c0f,)], 2, 1, 1, 3, 40,)

Validation du modèle

Fichier log

Messages envoyés et reçus par le lecteur

Données

<-- at 5 ms :: (1, **[(36, 1, {0, }, {}, 0xbf4e,)]**, 1, 2, 0, 2, 100,)
--> at 5000020 ns :: Collision
<-- at 5000021 ns :: (1, [(4, 1, {0, }, {}, 0xbc75,)], 1, 2, 0, 2, 100,)
<-- at 5000052 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000083 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000114 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
--> at 5000115 ns :: (3, [(0, 0, {0, },
{224, 171, 137, 171, 205, 239, 1, 35, }, 0x9c0f,)], 2, 1, 1, 3, 40,)

Validation du modèle

Fichier log

Messages envoyés et reçus par le lecteur

Flags

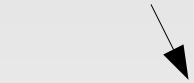
<-- at 5 ms :: (1, **[(36, 1, {0, }, {}, 0xbf4e,)]**, 1, 2, 0, 2, 100,)
--> at 5000020 ns :: Collision
<-- at 5000021 ns :: (1, [(4, 1, {0, }, {}, 0xbc75,)], 1, 2, 0, 2, 100,)
<-- at 5000052 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000083 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000114 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
--> at 5000115 ns :: (3, [(0, 0, {0, },
{224, 171, 137, 171, 205, 239, 1, 35, }, 0x9c0f,)], 2, 1, 1, 3, 40,)

Validation du modèle

Fichier log

Messages envoyés et reçus par le lecteur

Commande



<-- at 5 ms :: (1, **[(36, 1, {0, }, {}, 0xbf4e,)]**, 1, 2, 0, 2, 100,)
--> at 5000020 ns :: Collision
<-- at 5000021 ns :: (1, [(4, 1, {0, }, {}, 0xbc75,)], 1, 2, 0, 2, 100,)
<-- at 5000052 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000083 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000114 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
--> at 5000115 ns :: (3, [(0, 0, {0, },
{224, 171, 137, 171, 205, 239, 1, 35, }, 0x9c0f,)], 2, 1, 1, 3, 40,)

Validation du modèle

Fichier log

Messages envoyés et reçus par le lecteur

Paramètres

<-- at 5 ms :: (1, **[(36, 1, {0, }, {}, 0xbf4e,)]**, 1, 2, 0, 2, 100,)
--> at 5000020 ns :: Collision
<-- at 5000021 ns :: (1, [(4, 1, {0, }, {}, 0xbc75,)], 1, 2, 0, 2, 100,)
<-- at 5000052 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000083 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000114 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
--> at 5000115 ns :: (3, [(0, 0, {0, },
{224, 171, 137, 171, 205, 239, 1, 35, }, 0x9c0f,)], 2, 1, 1, 3, 40,)

Validation du modèle

Fichier log

Messages envoyés et reçus par le lecteur

Informations

<-- at 5 ms :: (1, **[(36, 1, {0, }, {}, 0xbf4e,)]**, 1, 2, 0, 2, 100,)
--> at 5000020 ns :: Collision
<-- at 5000021 ns :: (1, [(4, 1, {0, }, {}, 0xbc75,)], 1, 2, 0, 2, 100,)
<-- at 5000052 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000083 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000114 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
--> at 5000115 ns :: (3, [(0, 0, {0, },
{224, 171, 137, 171, 205, 239, 1, 35, }, 0x9c0f,)], 2, 1, 1, 3, 40,)

Validation du modèle

Fichier log

Messages envoyés et reçus par le lecteur

CRC

<-- at 5 ms :: (1, **[(36, 1, {0, }, {}, 0xbf4e,)]**, 1, 2, 0, 2, 100,)
--> at 5000020 ns :: Collision
<-- at 5000021 ns :: (1, [(4, 1, {0, }, {}, 0xbc75,)], 1, 2, 0, 2, 100,)
<-- at 5000052 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000083 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000114 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
--> at 5000115 ns :: (3, [(0, 0, {0, },
{224, 171, 137, 171, 205, 239, 1, 35, }, 0x9c0f,)], 2, 1, 1, 3, 40,)

Validation du modèle

Fichier log

Messages envoyés et reçus par le lecteur

End of Frame

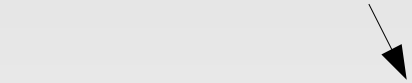
<-- at 5 ms :: (1, [(36, 1, {0, }, {}, 0xbf4e,)], **1**, 2, 0, 2, 100,)
--> at 5000020 ns :: Collision
<-- at 5000021 ns :: (1, [(4, 1, {0, }, {}, 0xbc75,)], 1, 2, 0, 2, 100,)
<-- at 5000052 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000083 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000114 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
--> at 5000115 ns :: (3, [(0, 0, {0, },
{224, 171, 137, 171, 205, 239, 1, 35, }, 0x9c0f,)], 2, 1, 1, 3, 40,)

Validation du modèle

Fichier log

Messages envoyés et reçus par le lecteur

Codage bits



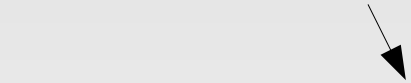
<-- at 5 ms :: (1, [(36, 1, {0, }, {}, 0xbf4e,)], 1, **2**, 0, 2, 100,)
--> at 5000020 ns :: Collision
<-- at 5000021 ns :: (1, [(4, 1, {0, }, {}, 0xbc75,)], 1, 2, 0, 2, 100,)
<-- at 5000052 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000083 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000114 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
--> at 5000115 ns :: (3, [(0, 0, {0, },
{224, 171, 137, 171, 205, 239, 1, 35, }, 0x9c0f,)], 2, 1, 1, 3, 40,)

Validation du modèle

Fichier log

Messages envoyés et reçus par le lecteur

Débit



<-- at 5 ms :: (1, [(36, 1, {0, }, {}, 0xbf4e,)], 1, 2, **0**, 2, 100,)
--> at 5000020 ns :: Collision
<-- at 5000021 ns :: (1, [(4, 1, {0, }, {}, 0xbc75,)], 1, 2, 0, 2, 100,)
<-- at 5000052 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000083 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000114 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
--> at 5000115 ns :: (3, [(0, 0, {0, },
{224, 171, 137, 171, 205, 239, 1, 35, }, 0x9c0f,)], 2, 1, 1, 3, 40,)

Validation du modèle

Fichier log

Messages envoyés et reçus par le lecteur

Modulation

<-- at 5 ms :: (1, [(36, 1, {0, }, {}, 0xbf4e,)], 1, 2, 0, **2**, 100,)
--> at 5000020 ns :: Collision
<-- at 5000021 ns :: (1, [(4, 1, {0, }, {}, 0xbc75,)], 1, 2, 0, 2, 100,)
<-- at 5000052 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000083 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000114 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
--> at 5000115 ns :: (3, [(0, 0, {0, },
{224, 171, 137, 171, 205, 239, 1, 35, }, 0x9c0f,)], 2, 1, 1, 3, 40,)

Validation du modèle

Fichier log

Messages envoyés et reçus par le lecteur

Qualité

<-- at 5 ms :: (1, [(36, 1, {0, }, {}, 0xbf4e,)], 1, 2, 0, 2, **100**,)
--> at 5000020 ns :: Collision
<-- at 5000021 ns :: (1, [(4, 1, {0, }, {}, 0xbc75,)], 1, 2, 0, 2, 100,)
<-- at 5000052 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000083 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
<-- at 5000114 ns :: (0, [(0, 0, {}, {}, 0,)], 1, 0, 0, 0, 100,)
--> at 5000115 ns :: (3, [(0, 0, {0, },
{224, 171, 137, 171, 205, 239, 1, 35, }, 0x9c0f,)], 2, 1, 1, 3, 40,)

Plan

- Présentation du LCIS
- Sûreté de fonctionnement des systèmes RFID
- Simulation d'un système RFID HF
- **Conclusion**

Conclusion

- État d'avancement
  - Analyse des différents composants
  - Analyse des modes de défaillances
  - **Développement d'un simulateur**
    - **Développement du modèle**
    - Injection de fautes

Conclusion

- Étude de la RFID
  - Composants
  - Fonctionnement
  - Dysfonctionnement
- Développement d'un modèle RFID
  - Co-design
  - Simulation du système complet (10 000 lignes de code)

Conclusion

- Perspectives
  - Utiliser le modèle pour injecter des fautes
  - Simuler des stratégies de sûreté de fonctionnement
  - Interfacer avec un middleware

Conclusion

- Injection de faute
  - Exemple : couplage
    - Transmission impossible de message
    - Forte diminution de la puissance
  - Exemple : mémoire
    - Bits faux
    - Blocs illisibles

Merci pour votre attention