UMR IRISA

Institut de Recherche en Informatique
et Systèmes Aléatoires

Capteurs sans batterie ou le mythe de l'autonomie infinie

Comment la variabilité et le vieillissement des composants impacte l'exécution de programmes ?

Hugo Reymond - Equipe PACAP

Greendays 2025

Université de Rennes | rennes | INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES | cnrs | Inria | Inserm | CentraleSupélec | IMT Atlantique Bretagne-Pays de la Loire École Mines-Télécom | Université Bretagne Sud ubs

Capteurs sans batterie

Capteurs sans batterie

Capteurs sans fil

Capteurs sans batterie

Capteurs sans fil

- Collecte de données, traitement et envoi des données

Capteurs sans batterie

Capteurs sans fil

- Collecte de données, traitement et envoi des données
- Déploiement dans des lieux souvent isolés ou inaccessibles

Capteurs sans batterie

Capteurs sans fil

- Collecte de données, traitement et envoi des données
- Déploiement dans des lieux souvent isolés ou inaccessibles

Autonomie des capteurs sans fil

Capteurs sans batterie

Capteurs sans fil

- Collecte de données, traitement et envoi des données
- Déploiement dans des lieux souvent isolés ou inaccessibles

Autonomie des capteurs sans fil

Piles et batteries

Capteurs sans batterie

Capteurs sans fil

- Collecte de données, traitement et envoi des données
- Déploiement dans des lieux souvent isolés ou inaccessibles

Autonomie des capteurs sans fil

Piles et batteries

- Autonomie limitée, remplacement fréquent
- Beaucoup de déchets

Capteurs sans batterie

Capteurs sans fil

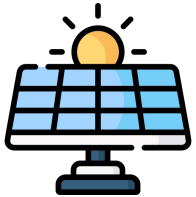
- Collecte de données, traitement et envoi des données
- Déploiement dans des lieux souvent isolés ou inaccessibles

Autonomie des capteurs sans fil

Piles et batteries

- Autonomie limitée, remplacement fréquent
- Beaucoup de déchets

Récolte d'énergie



Capteurs sans batterie

Capteurs sans fil

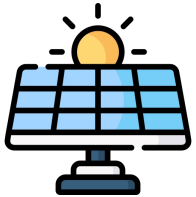
- Collecte de données, traitement et envoi des données
- Déploiement dans des lieux souvent isolés ou inaccessibles

Autonomie des capteurs sans fil

Piles et batteries

- Autonomie limitée, remplacement fréquent
- Beaucoup de déchets

Récolte d'énergie



Condensateur

Capteurs sans batterie

Capteurs sans fil

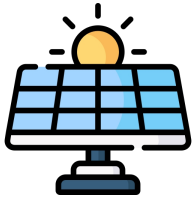
- Collecte de données, traitement et envoi des données
- Déploiement dans des lieux souvent isolés ou inaccessibles

Autonomie des capteurs sans fil

Piles et batteries

- Autonomie limitée, remplacement fréquent
- Beaucoup de déchets

Récolte d'énergie



Condensateur

- Autonomie "infinie"
- Empreinte écologique réduite

Capteurs sans batterie

Capteurs sans fil

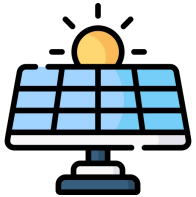
- Collecte de données, traitement et envoi des données
- Déploiement dans des lieux souvent isolés ou inaccessibles

Autonomie des capteurs sans fil

Piles et batteries

- Autonomie limitée, remplacement fréquent
- Beaucoup de déchets

Récolte d'énergie



Condensateur

- Autonomie "infinie"
- Empreinte écologique réduite

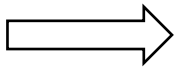
Comment cette autonomie "infinie" se confronte aux réalités matérielles ?

Variabilité des composants

Les caractéristiques des composants varient selon :

- Conditions d'opérations 
- Vieillissement des matériaux 

Exemple : Condensateur

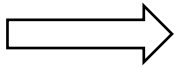
 Vieillissement  Capacité réduite, Résistance série (ESR) augmentée

Variabilité des composants

Les caractéristiques des composants varient selon :

- Conditions d'opérations 
- Vieillissement des matériaux 

Exemple : Condensateur

 Vieillissement  Capacité réduite, Résistance série (ESR) augmentée

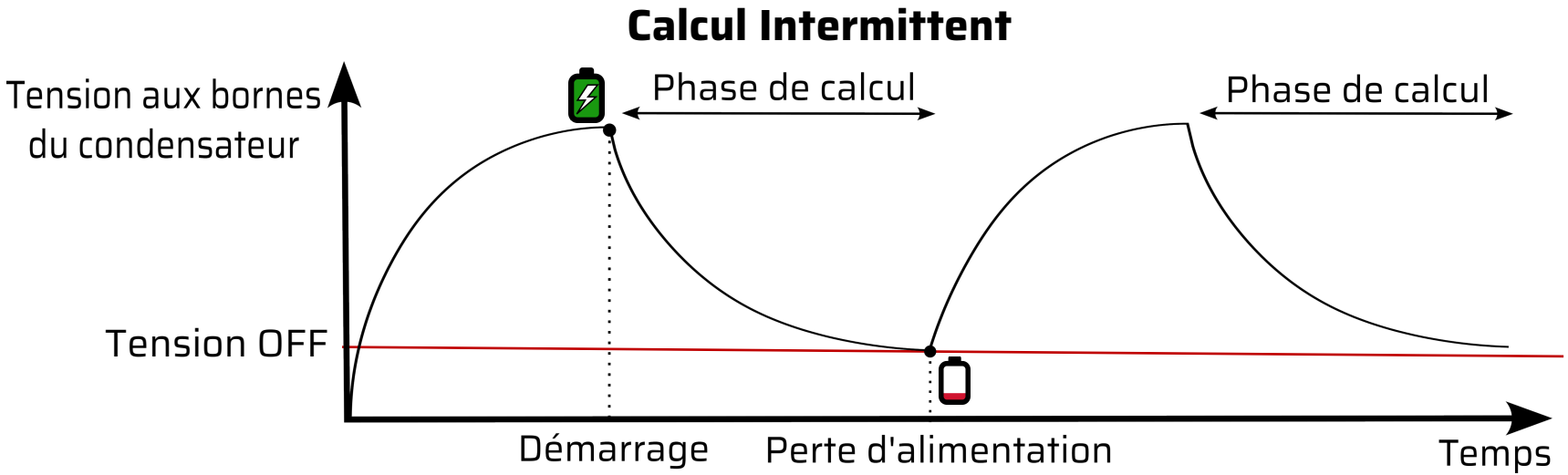
Quel est l'impact de cette variabilité sur l'exécution d'un programme ?

Un peu de contexte : le calcul intermittent

Absence de batteries ✚ Faible énergie récoltée ⇨ Calcul en continu impossible

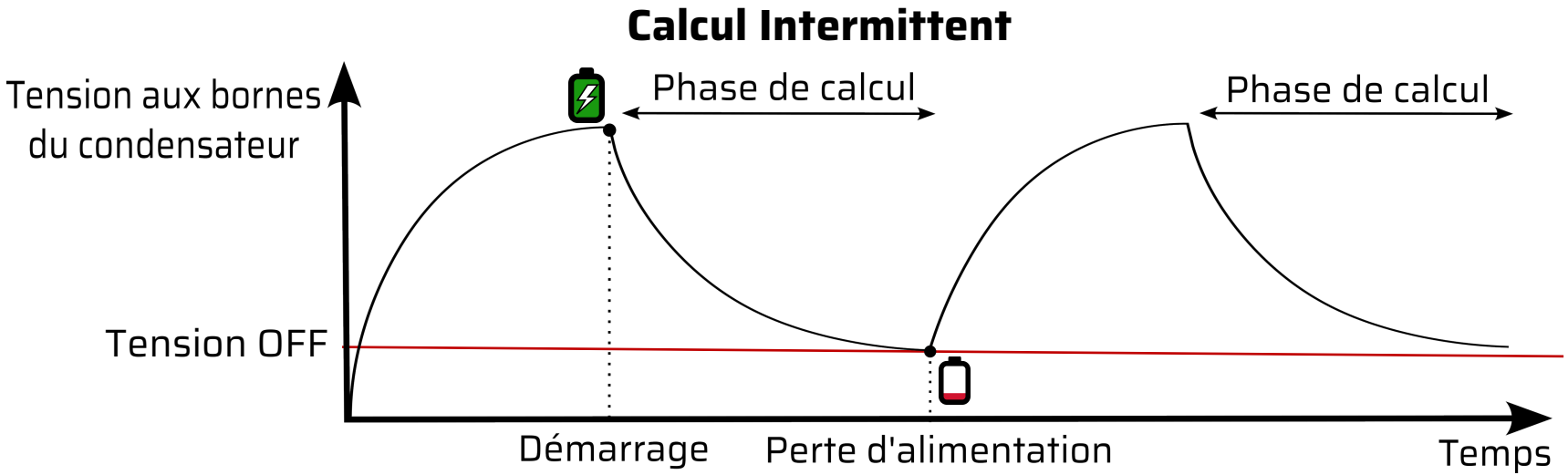
Un peu de contexte : le calcul intermittent

Absence de batteries ✚ Faible énergie récoltée ⇨ Calcul en continu impossible



Un peu de contexte : le calcul intermittent

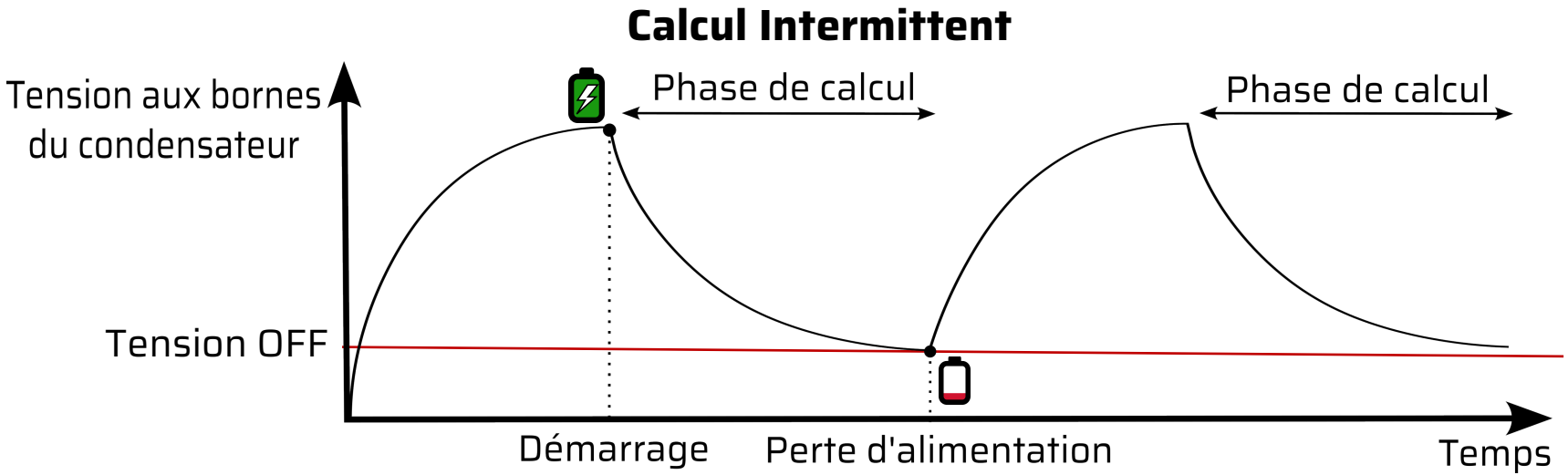
Absence de batteries ✚ Faible énergie récoltée ⇨ Calcul en continu impossible



Perte d’alimentation

Un peu de contexte : le calcul intermittent

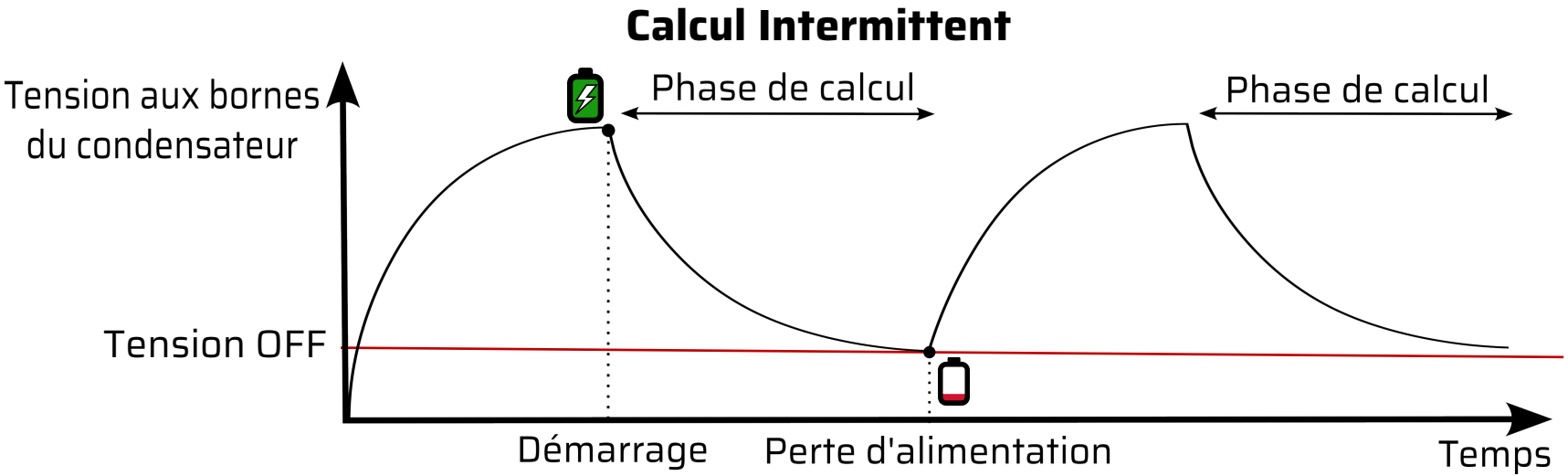
Absence de batteries ✚ Faible énergie récoltée ⇨ Calcul en continu impossible



Perte d'alimentation ⇨ Perte des données volatiles (mémoire et registres)

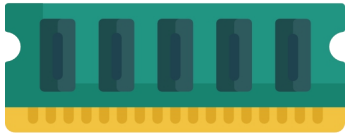
Un peu de contexte : le calcul intermittent

Absence de batteries ✚ Faible énergie récoltée ⇨ Calcul en continu impossible

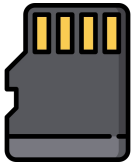


Perte d'alimentation ⇨ Perte des données volatiles (mémoire et registres) ⇨ Progression perdue

Checkpointing

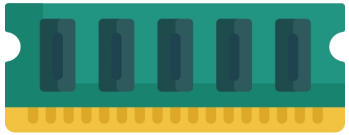


VM : Mémoire volatile



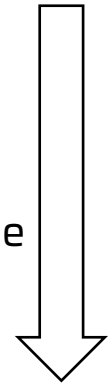
NVM : Mémoire non volatile

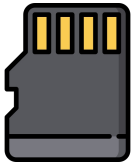
Checkpointing



VM : Mémoire volatile

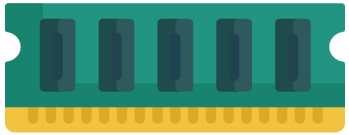
Sauvegarde





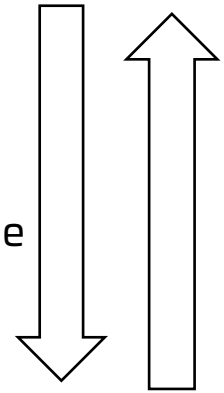
NVM : Mémoire non volatile

Checkpointing

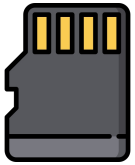


VM : Mémoire volatile

Sauvegarde

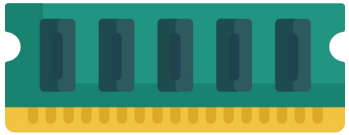


Reprise de l'exécution



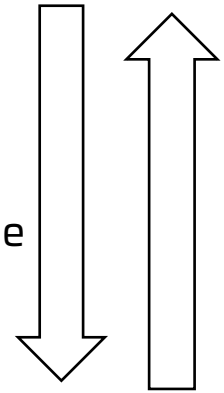
NVM : Mémoire non volatile

Checkpointing

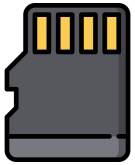


VM : Mémoire volatile

Sauvegarde



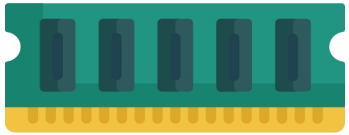
Reprise de l'exécution



NVM : Mémoire non volatile

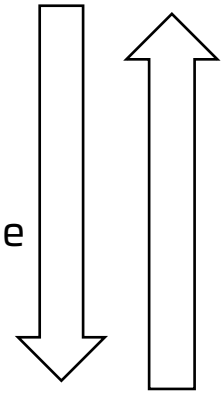
```
int sum = 0;
for(int i = 0; i<4; i++){

    sum += array[i]
}

send(sum);
```

Checkpointing

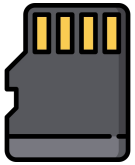


VM : Mémoire volatile

Sauvegarde



Reprise de l'exécution



NVM : Mémoire non volatile

```
int sum = 0;
for(int i = 0; i<4; i++){
    checkpoint();
    sum += array[i]
}
checkpoint();
send(sum);
checkpoint();
```

Impact de la variabilité sur l'exécution

Placement de point de sauvegarde dimensionné par rapport à la taille du condensateur

Impact de la variabilité sur l'exécution

Placement de point de sauvegarde dimensionné par rapport à la taille du condensateur

```
int sum = 0;
for(int i = 0; i<4; i++){
    checkpoint();
    sum += array[i]
}
checkpoint();
send(sum);
checkpoint();
```

Impact de la variabilité sur l'exécution

Placement de point de sauvegarde dimensionné par rapport à la taille du condensateur

```
int sum = 0;
for(int i = 0; i<4; i++){
    checkpoint();
    sum += array[i]
}
checkpoint();
send(sum);
checkpoint();
```

$$E = \frac{1}{2} C V^2$$

Impact de la variabilité sur l'exécution

Placement de point de sauvegarde dimensionné par rapport à la taille du condensateur

```
int sum = 0;
for(int i = 0; i<4; i++){
    checkpoint();
    sum += array[i]
}
checkpoint();
send(sum);
checkpoint();
```

$$E = \frac{1}{2}CV^2$$

Si la capacité diminue, il devient impossible d'atteindre le prochain point de sauvegarde

Réexécution à l'infini : PLUS DE PROGRÈS

Impact de la variabilité sur l'exécution

Placement de point de sauvegarde dimensionné par rapport à la taille du condensateur

```
int sum = 0;
for(int i = 0; i<4; i++){
    checkpoint();
    sum += array[i]
}
checkpoint();
send(sum);
checkpoint();
```

$$E = \frac{1}{2}CV^2$$

Si la capacité diminue, il devient impossible d'atteindre le prochain point de sauvegarde

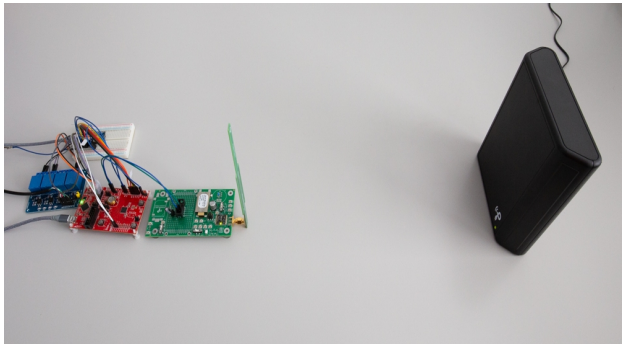
Réexécution à l'infini : PLUS DE PROGRÈS

L'observe-t-on expérimentalement ?

Dispositif expérimental

Matériel

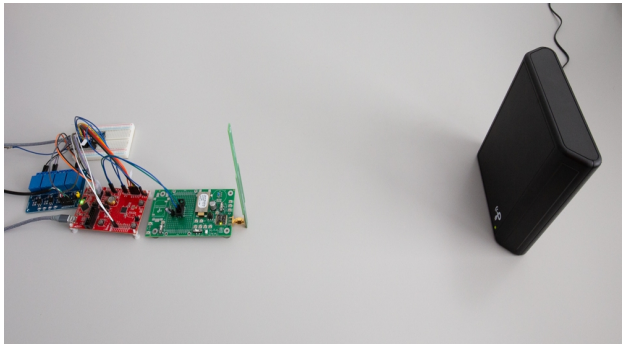
Noir : Source d'énergie (Émetteur RF)
Vert : Récolte d'énergie (P2110 RF harvester)
Rouge : Cible (Microcontrôleur MSP430FR5969)



Dispositif expérimental

Matériel

Noir : Source d'énergie (Émetteur RF)
Vert : Récolte d'énergie (P2110 RF harvester)
Rouge : Cible (Microcontrôleur MSP430FR5969)



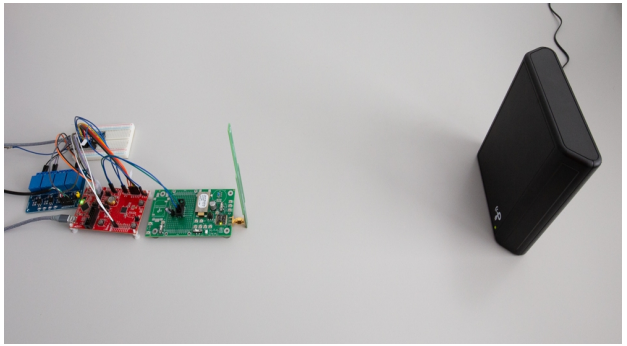
Logiciel

- Trois benchmarks (aes, crc, rc4)
- Placement de point de sauvegarde statique conservatif

Dispositif expérimental

Matériel

Noir : Source d'énergie (Émetteur RF)
Vert : Récolte d'énergie (P2110 RF harvester)
Rouge : Cible (Microcontrôleur MSP430FR5969)



Logiciel

- Trois benchmarks (aes, crc, rc4)
- Placement de point de sauvegarde statique conservatif

Réserve d'énergie

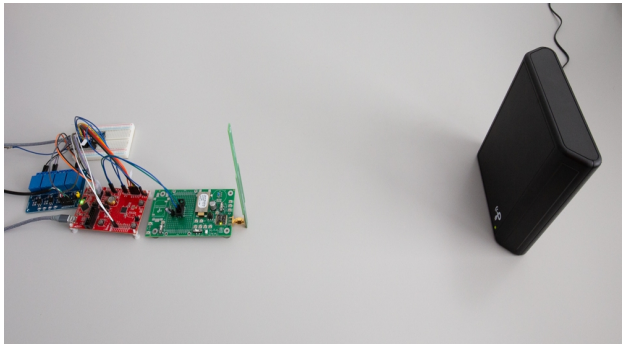
Capacité (en µF)	105	75	42	22
Dégradation	0%	29%	60%	80%

⚠ On ne simule pas la dégradation de l'ESR

Dispositif expérimental

Matériel

Noir : Source d'énergie (Émetteur RF)
Vert : Récolte d'énergie (P2110 RF harvester)
Rouge : Cible (Microcontrôleur MSP430FR5969)



Logiciel

- Trois benchmarks (aes, crc, rc4)
- Placement de point de sauvegarde statique conservatif

Réserve d'énergie

Capacité (en µF)	105	75	42	22
Dégradation	0%	29%	60%	80%

⚠ On ne simule pas la dégradation de l'ESR

Protocole

- La puissance récoltée ne permet pas d'alimenter la cible en continu
- La cible exécute un programme en boucle pendant 1 min
- On compte le nombre de programmes exécutés pendant les 1 min

Progression des calculs

	Benchmarks exécutés (moyenne)		
Capacité(µF)	aes	crc	rc4
105	26	61	34
75	30	61	38
42	0	73	7
22	0	0	0

- Si le condensateur est trop dégradé, pas de progrès
  - Aucun benchmark complété

Progression des calculs

	Benchmarks exécutés (moyenne)		
Capacité(µF)	aes	crc	rc4
105	26	61	34
75	30	61	38
42	0	73	7
22	0	0	0

	Pourcentage de réexécution		
Capacité(µF)	aes	crc	rc4
105	0%	0%	0%
75	0%	0%	0%
42	96%	0%	95%
22	80%	99%	99%

- Si le condensateur est trop dégradé, pas de progrès
  - Aucun benchmark complété
  - Principalement de la réexécution

Progression des calculs

	Benchmarks exécutés (moyenne)		
Capacité(µF)	aes	crc	rc4
105	26	61	34
75	30	61	38
42	0	73	7
22	0	0	0

	Pourcentage de réexécution		
Capacité(µF)	aes	crc	rc4
105	0%	0%	0%
75	0%	0%	0%
42	96%	0%	95%
22	80%	99%	99%

- Si le condensateur est trop dégradé, pas de progrès
  - Aucun benchmark complété
  - Principalement de la réexécution
- En réalité, pas notre principal problème

Progression des calculs

	Benchmarks exécutés (moyenne)		
Capacité(µF)	aes	crc	rc4
105	26	61	34
75	30	61	38
42	0	73	7
22	0	0	0

	Pourcentage de réexécution		
Capacité(µF)	aes	crc	rc4
105	0%	0%	0%
75	0%	0%	0%
42	96%	0%	95%
22	80%	99%	99%

- Si le condensateur est trop dégradé, pas de progrès
  - Aucun benchmark complété
  - Principalement de la réexécution
- En réalité, pas notre principal problème : par moment, on récupèrera suffisament d'énergie pour s'exécuter en continu

Progression des calculs

	Benchmarks exécutés (moyenne)		
Capacité(µF)	aes	crc	rc4
105	26	61	34
75	30	61	38
42	0	73	7
22	0	0	0

	Pourcentage de réexécution		
Capacité(µF)	aes	crc	rc4
105	0%	0%	0%
75	0%	0%	0%
42	96%	0%	95%
22	80%	99%	99%

- Si le condensateur est trop dégradé, pas de progrès
  - Aucun benchmark complété
  - Principalement de la réexécution
- En réalité, pas notre principal problème : par moment, on récupèrera suffisament d'énergie pour s'exécuter en continu
- Problème : la corruption mémoire.

Anomalies mémoire

Accès mémoire VM et NVM ✚ ré-exécution ⟹ corruption mémoire

Anomalies mémoire

Accès mémoire VM et NVM ✚ ré-exécution ⟹ corruption mémoire

```
int array[2];
int i = 0;
while(i<2) {
    checkpoint();
    array[i] = 0;
    i++;
}
```

Anomalies mémoire

Accès mémoire VM et NVM ✚ ré-exécution ⟹ corruption mémoire

```
int array[2];
int i = 0;
while(i<2) {
    checkpoint();
    array[i] = 0;
    i++;
}
```

Avec `i` en NVM

Anomalies mémoire

Accès mémoire VM et NVM ✚ ré-exécution ⟹ corruption mémoire

```
int array[2];
int i = 0;
while(i<2) {
    checkpoint();
    array[i] = 0;
    i++;
}
```

⟹ Exécution Intermittente

```
array[0] = 0
i++
checkpoint()
array[1] = 0
i++
```

Avec `i` en NVM

Anomalies mémoire

Accès mémoire VM et NVM ✚ ré-exécution ⟹ corruption mémoire

```
int array[2];
int i = 0;
while(i<2) {
    checkpoint();
    array[i] = 0;
    i++;
}
```

⟹ Exécution Intermittente

```
array[0] = 0
i++
checkpoint()
array[1] = 0
i++
```

Avec `i` en NVM

Anomalies mémoire

Accès mémoire VM et NVM ✚ ré-exécution ⟹ corruption mémoire

```
int array[2];
int i = 0;
while(i<2) {
    checkpoint();
    array[i] = 0;
    i++;
}
```

Avec `i` en NVM

⟹ Exécution Intermittente

```
array[0] = 0
i++
checkpoint()
array[1] = 0
i++
⚡
array[2] = 0 ⚠
```

Accès out of bound
= corruption mémoire

Anomalies mémoire

Accès mémoire VM et NVM ✚ ré-exécution ⟹ corruption mémoire

```
int array[2];
int i = 0;
while(i<2) {
    checkpoint();
    array[i] = 0;
    i++;
}
```

Avec `i` en NVM

⟹ Exécution Intermittente

```
array[0] = 0
i++
checkpoint()
array[1] = 0
i++
⚡
array[2] = 0  ⚠
```

Accès out of bound = corruption mémoire

Une seule réexécution peut corrompre tout le système !

Anomalies mémoire

Accès mémoire VM et NVM ✚ ré-exécution ⟹ corruption mémoire

```
int array[2];
int i = 0;
while(i<2) {
    checkpoint();
    array[i] = 0;
    i++;
}
```

Avec `i` en NVM

⟹ Exécution Intermittente

```
array[0] = 0
i++
checkpoint()
array[1] = 0
i++
⚡
array[2] = 0  ⚠
```

Accès out of bound = corruption mémoire

Une seule réexécution peut corrompre tout le système !

Certaines approches “pire cas” garantissent l’absence de réexécution en se basant sur la taille du condensateur

Anomalies mémoire

Accès mémoire VM et NVM ✚ ré-exécution ⟹ corruption mémoire

```
int array[2];
int i = 0;
while(i<2) {
    checkpoint();
    array[i] = 0;
    i++;
}
```

Avec `i` en NVM

⟹ Exécution Intermittente

```
array[0] = 0
i++
checkpoint()
array[1] = 0
i++
⚡
array[2] = 0  ⚠
```

Accès out of bound = corruption mémoire

Une seule réexécution peut corrompre tout le système !

Certaines approches “pire cas” garantissent l’absence de réexécution en se basant sur la taille du condensateur ⟹ Sa dégradation casse ces garanties

Réexécutions

	Apparition d'une ré-exécution (sur 50 expériences)		
Capacité(µF)	aes	crc	rc4
105	0	0	0
75	0	0	0
42	50	0	50
22	50	50	50

- La dégradation du condensateur entraine des réexécutions

Réexécutions

	Apparition d'une ré-exécution (sur 50 expériences)		
Capacité(µF)	aes	crc	rc4
105	0	0	0
75	0	0	0
42	50	0	50
22	50	50	50

- La dégradation du condensateur entraine des réexécutions
- Ces réexécutions peuvent entrainer des anomalies mémoires

Réexécutions

	Apparition d'une ré-exécution (sur 50 expériences)		
Capacité(µF)	aes	crc	rc4
105	0	0	0
75	0	0	0
42	50	0	50
22	50	50	50

- La dégradation du condensateur entraine des réexécutions
- Ces réexécutions peuvent entrainer des anomalies mémoires
  - Meilleur cas : Capteur HS
  - Pire cas : Mesures corrompues/érronées, comportements nuisibles

Réexécutions

	Apparition d'une ré-exécution (sur 50 expériences)		
Capacité(µF)	aes	crc	rc4
105	0	0	0
75	0	0	0
42	50	0	50
22	50	50	50

- La dégradation du condensateur entraine des réexécutions
- Ces réexécutions peuvent entrainer des anomalies mémoires
  - Meilleur cas : Capteur HS
  - Pire cas : Mesures corrompues/érronées, comportements nuisibles
- Effets dommageables et irréversibles sur le réseau de capteur

Conclusion

- Autonomie “infinie” ?
- Fonctionnement des capteurs sans batteries impacté par
    - Conditions opératoires (température, humidité)
    - Vieillissement des composants matériels

Conclusion

- Autonomie “infinie” ?
- Fonctionnement des capteurs sans batteries impacté par
  - Conditions opératoires (température, humidité)
  - Vieillissement des composants matériels
- Dégradation du condensateur ⇨ Absence de progrès, Anomalies mémoire
- Avec une dégradation de 60%, possibilité d’effets irréversibles
- Nécessité d’intervention manuelle

Conclusion

- Autonomie “infinie” ?
- Fonctionnement des capteurs sans batteries impacté par
  - Conditions opératoires (température, humidité)
  - Vieillissement des composants matériels
- Dégradation du condensateur ⇒ Absence de progrès, Anomalies mémoire
- Avec une dégradation de 60%, possibilité d’effets irréversibles
- Nécessité d’intervention manuelle

Autonomie menacée par la dégradation des composants

Travaux futurs

- Comment détecter la dégradation du condensateur ?
- Comment s'adapter aux dégradations du condensateur ?
- Focus : Condensateur
    - Est-ce le composant le plus faillible ?
    - Analyse des autres composants
- Quelle est la durée de vie d'un capteur sans batterie ?

Merci de votre attention
Avez-vous des questions ?

This work has received a French government support granted to the Labex CominLabs excellence laboratory and managed by the National Research Agency in the "Investing for the Future" program under reference ANR-10-LABX-07-01

Dégradation du condensateur

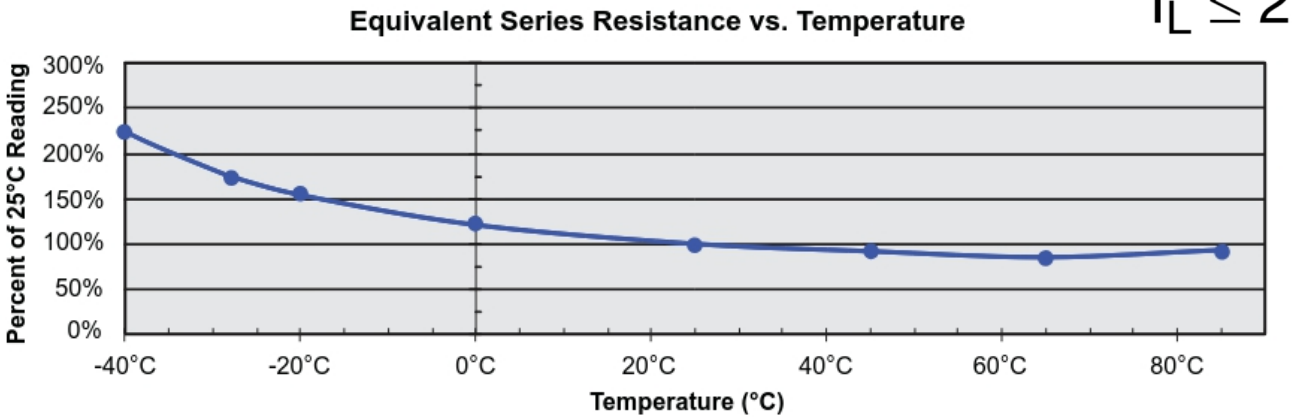
- Durée de vie “utile” d’un condensateur conditions standard
  - ≈1 an à 40° (8 000h)
  - ≈3 ans à 25° (25 000h)

Définition de la durée de vie utile

$\Delta C/C$: ± 30 %

$R_I \leq 4$ x spec. limit

$I_L \leq 2$ x spec. limit



Sources :

“196DLC series - Energy Storage Double Layer Capacitors” datasheet, Vishay Components, rev 2018

“SCM series - Series-Connected SuperCapacitor Modules” datasheet, AVX