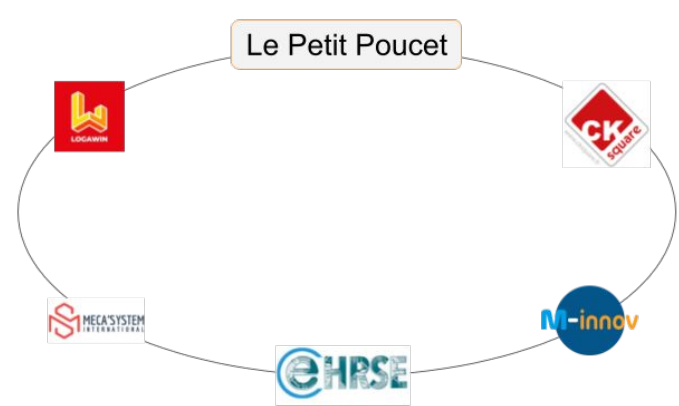


Création d'un outil d'intégration continue pour tester du code embarqué

Stage de 2ème année

CKsquare

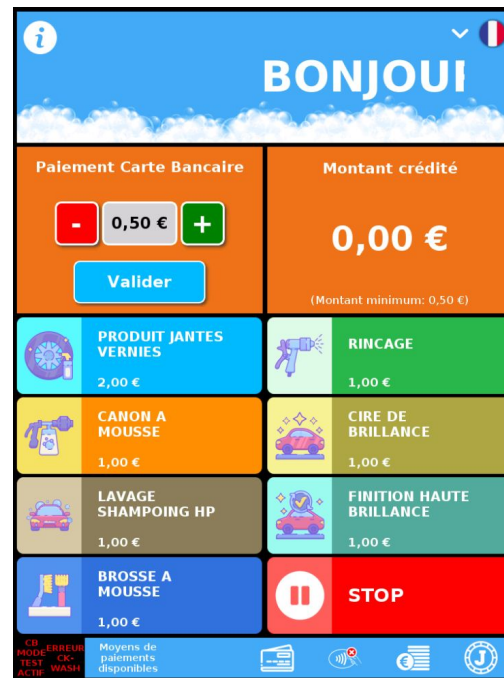
- Créée en 2003, Emmanuel Bertrand
- Groupe “*Le Petit Poucet*”
- Entreprise d'ingénierie, d'étude et de conseil
- Systèmes de paiements automatisés
- Clients : laveries et stations de lavage auto
- Plus de 30 employés
- > 40 000 stations de lavages auto



CKsquare



Myosis TAP



Programme pupitouch

Introduction



- Le projet
 - 1 seul développeur
 - durée: 5 mois
 - démarrage de 0
- Objectifs
 - réalisation d'un outil d'intégration continue sur du code embarqué
 - compiler sur Linux (pipeline)
 - simulation des composants électroniques
 - documentation => outil modifiable
 - projets C et Qt

Plan



- Mise en place du projet
 - Choix du framework de test
 - CMake
- La simulation
 - Horloges
 - Stockage
 - Cctalk & MDB
- La suite du projet
 - Qt
 - La pipeline
 - Développement durable



Choix du framework de test

	Check	CUnit	criterion	minunit	munit	Unity	Tau
compatible c++	non	non	oui	oui	non	non	non
macros (tests)	oui	non	oui	oui	non	non	oui
main généré	non	non	oui	non	non	non	oui
compatible gitlab	oui	non	oui	oui	oui	oui	oui
dépendances	oui	non	non	oui	non	non	non
portable	oui	oui	oui	oui	oui	oui	oui
processus séparés	oui	non	oui	non	non	non	non
+ embarqué	oui	non	oui	non	non	oui	oui
PRNG	non	non	non	non	oui	non	non
TAP	oui	non	oui	non	non	non	oui
star github/gitlab	941	9	1.7k	477	436	3k	121
contributeurs	8	1	32	14	9	111	5
dernière maj	18/10/2021	2/04/23	30/03/23	4/04/2020	12/05/20	15/03/23	21/05/22





Criterion

- Moderne
- Simple d'utilisation
- Compatible C++
- Fonctionnalités avancées
 - fixtures / tests paramétrés / capture de signaux
 - **isolation des tests**
- Défauts
 - complexe
 - macros

```
#include <riterion/criterion.h>
#include <stdio.h>

void setup(void) {
    puts("Runs before the test");
}

void teardown(void) {
    puts("Runs after the test");
}

Test(simple, fixtures, .init = setup,
    .fini = teardown) {
    cr_assert(1);
}
```

CMake

- Makefile
 - flexible
 - complexe sur de gros projets
- CMake
 - très simple à mettre en place
 - plus lisible
 - CTest
 - pratique avec Qt



GNU Make



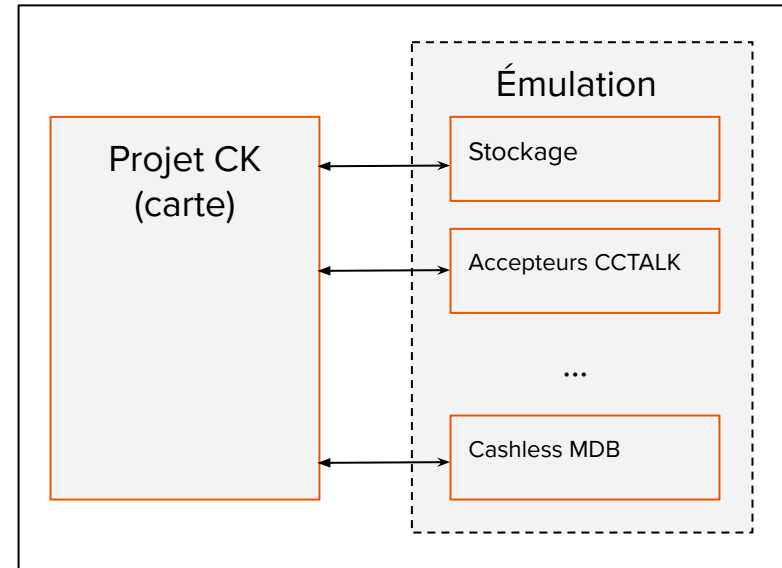
Simulation



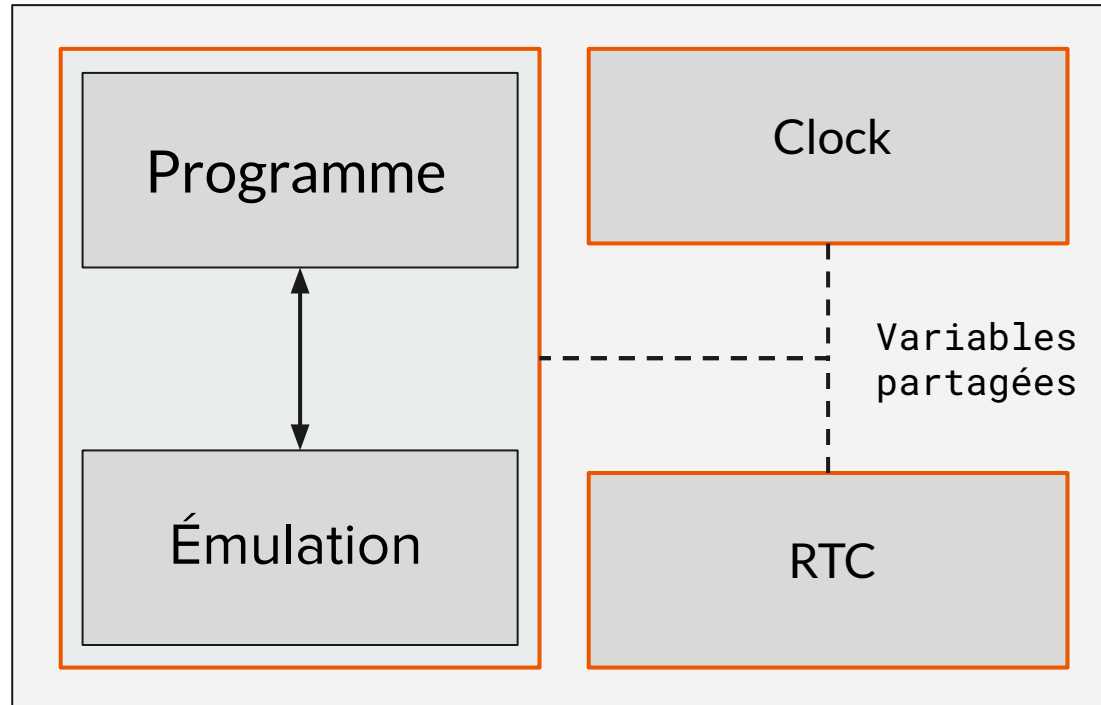
- Objectif
 - isoler le code testé
 - exécuter le code dans la pipeline
 - vérifier les interactions composants / carte
- Composants
 - Horloges
 - Stockage
 - Interfaces Cctalk & MDB

Threads

- Première solution
 - Simulation très réaliste
 - Synchronisation ? Sémaphores ?
 - Mémoire partagée
 - trop complexe
- Threads utiles ?
 - timers

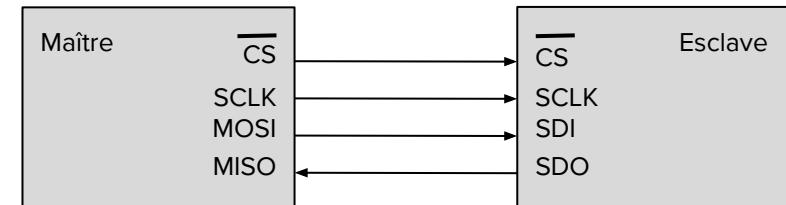
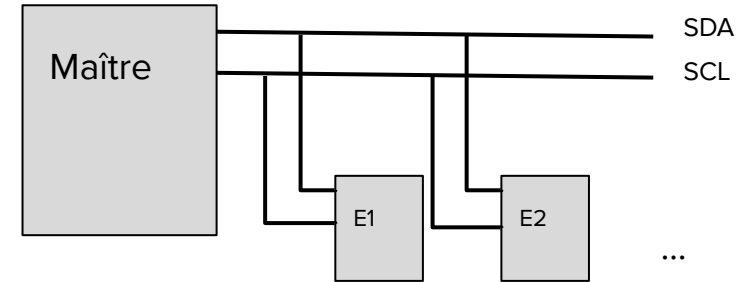


Les horloges




Stockage

- Mémoire => tableaux
 - registre
 - eeprom
 - flash
- 2 interfaces
 - I2C
 - SPI
- Simulation
 - machines à états
 - fichiers modifiés
 - ~~Qt~~ (fichier)



Stockage - Exemple



```
void TESTMEMSPI_Init(void)
{
    TESTMEMSPI_CONTROL.DeviceNumber = 0;
    TESTMEMSPI_DeviceAdd(EEPROM,0,0); // TESTMEM_EEPROMM[0], CS: 0
    TESTMEMSPI_DeviceAdd(FLASH,0,1); // TESTMEM_FLASHM[0], CS: 1
    TESTMEMSPI_DeviceAdd(FLASH,1,2); // TESTMEM_FLASHM[1], CS: 2
}
```

Configuration

Stockage - Exemple

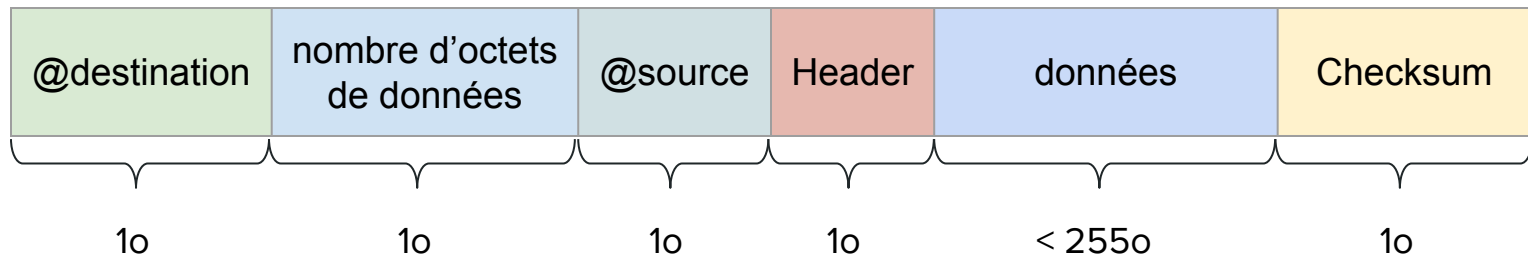


```
unsigned char Buff[5] = { 1, 2, 3, 4, 5 };  
W25Q_BufferWrite(0, 10, Buff, 5);  
  
// les données sont bien écrites sur l'eeeprom 0 ?  
cr_assert(eq(u8[5], TESTMEM_EEPROM[0] + 10, Buff));
```

Test

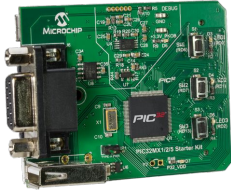
Le protocole cctalk

- Protocole
 - monétique
 - maître / esclave
 - headers
 - 256 headers
 - 20 ... 99 réservés aux applications
- Affichage + communication (TCP/IP)
- Trame:



Cctalk - exemple

Affiche:
"Hello, World!"



2

14

1

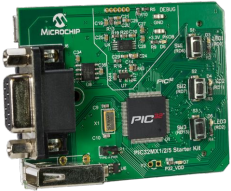
203

3Hello, World!

184

Display Control + sous header 3

Cctalk - exemple



1	0	2	0	253
---	---	---	---	-----

ACK

Hello, World!

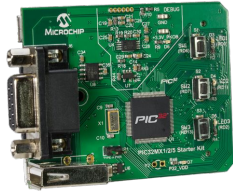


Le protocole MDB

- Protocole
 - monétique
 - maître / esclave
- Trame (9 bits)
 - 8 bits de données
 - 1 bit de mode:
 - 0 => données
 - 1 => adresse / ACK / checksum
 - commandes + sous-commandes



MDB - exemple



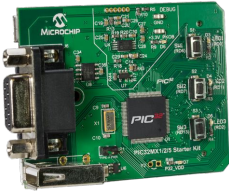
0x113	0x00	0x113
-------	------	-------

Vend request

$$0x013 = 0x10 + 0x03$$

Adresse Vend

MDB - exemple



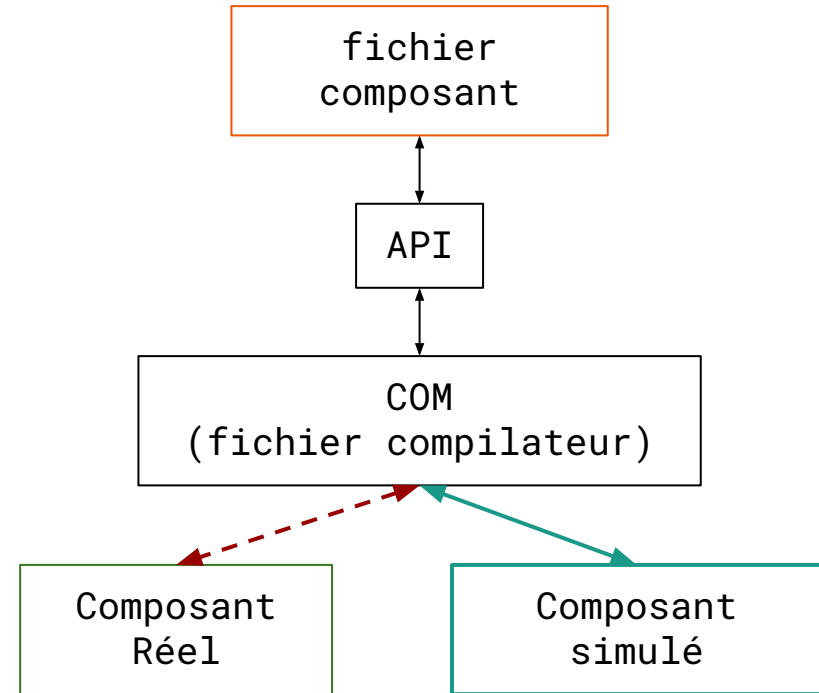
0x05	0x105
------	-------

Vend approved



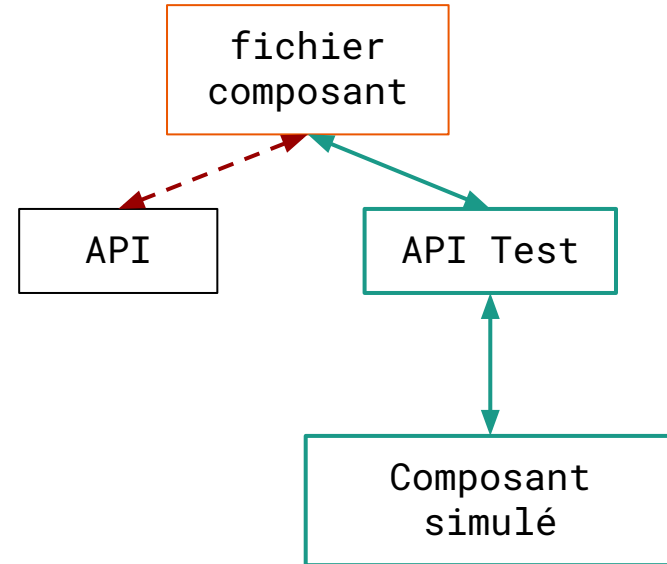
Simulation bas niveau

- Problème = placer la simulation
- Objectif = tester tout le code
- Situation réaliste
- Solution complexe
 - mise en oeuvre
 - utilisation
 - localisation des erreurs
- Tests d'intégration



Simulation haut niveau

- Plus simple
- Plus précis
- Fichier testé isolé
- Moins réaliste
- Test unitaire



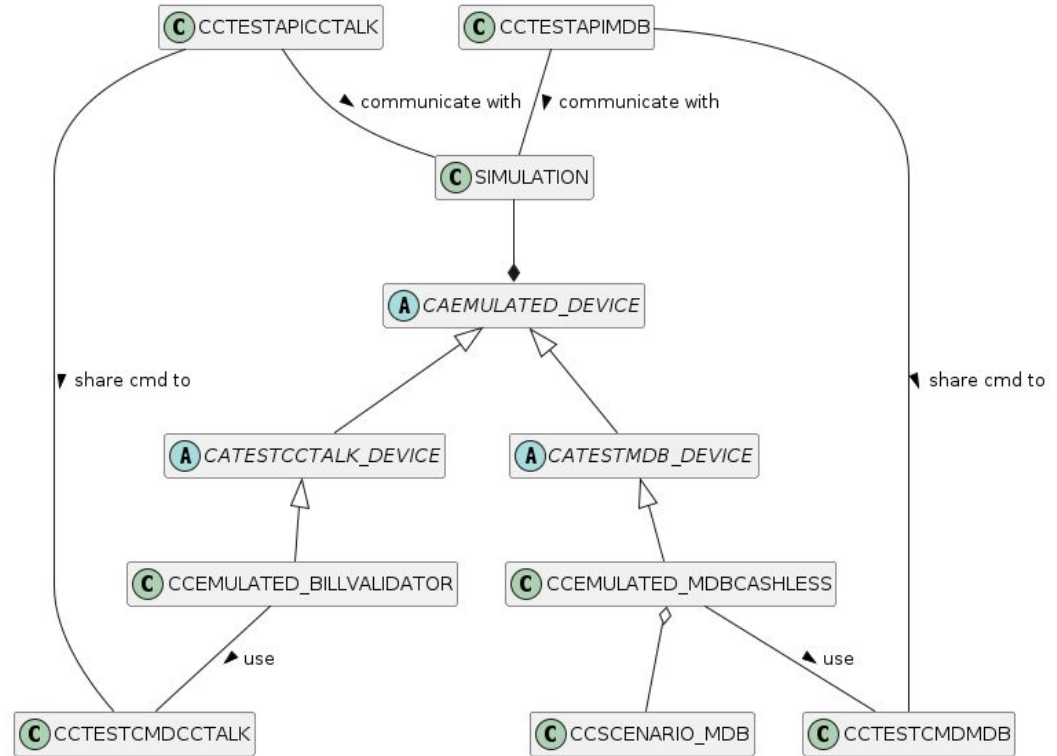
Les capacités de la simulation



- Contrôle des horloges
- Stockage
 - périphérique
 - interfaces
- Cctalk & MDB
 - tests unitaires / intégration
 - vision des communications
 - réponses configurables & automatiques
 - Scénarios (MDB)

Qt

- Qt Test
 - classes
 - compatible Qt
- Objet
- Mocks



La pipeline

- Fonctionnalités
 - Automatisation des tests
 - Code coverage (gcov)
 - Memcheck (C++)
 - Construction d'images docker (Qt)



Développement durable

- Effet indirecte
- Détection des bugs
 - moins de transport
 - moins de mises à jour
- Optimisation du code
 - diminuer les consommations
- Création de l'outil
 - économie des ressources



Conclusion



- Cahier des charges
 - framework de test
 - simulation
 - pipeline (test, coverage, memcheck)
 - documentation (wiki, test exemples)
- Perspectives
 - pipeline
 - praticité d'utilisation
- Outil utile !
- Au delà du projet
 - découverte d'outils
 - le monde de l'embarqué
 - le monde de l'entreprise

Merci de votre attention.