Synchronous Programming with Refinement Types

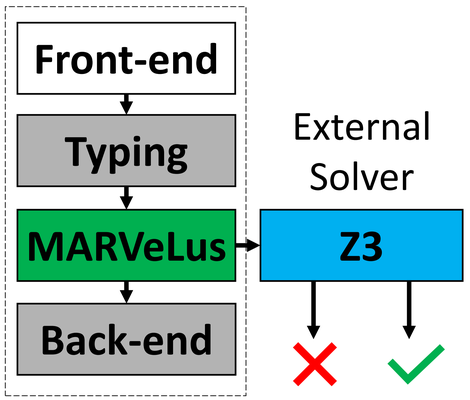
José Luiz Vargas de Mendonça[1] (Honors Capstone), Jiawei Chen[2], Jean-Baptiste Jeannin[3]
1. Aerospace and Computer Engineering Undergraduate, 2. Robotics PhD Student,
3. Assistant Professor of Aerospace Engineering

Introduction

- **Cyber-physical systems (CPSs)** are composed of software that interacts with the environment.

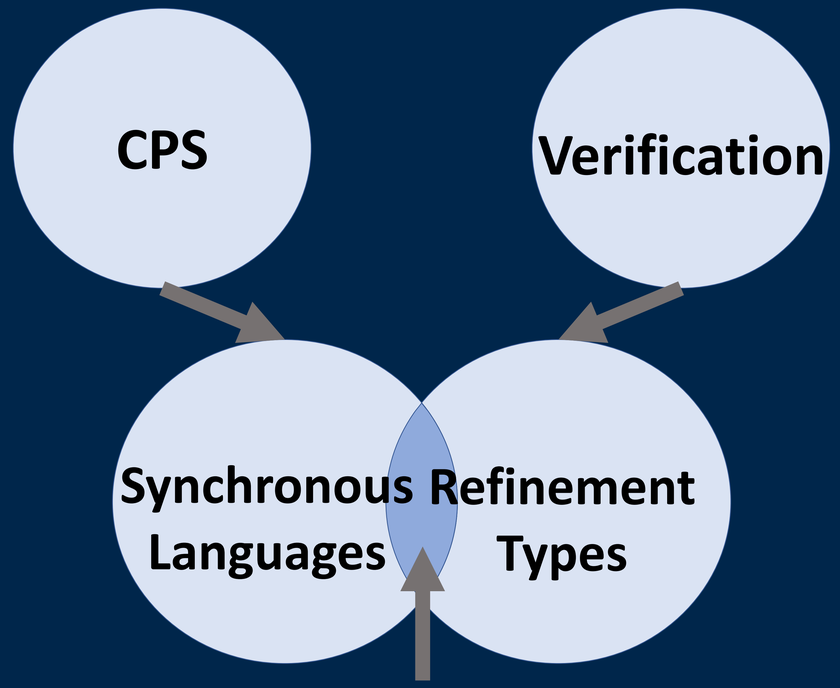
- Unit testing complex software might not cover all scenarios.
- Formal verification provides rigorous tools to prove **software safety**.
- **Synchronous programming languages** have been used in CPSs.
- **Refinement types** have been used for verification.
- This project combines synchronous programming with refinement types into **MARVeLus**: a tool to prove CPSs safety properties.

Algorithm



Unit tests are **not enough** to show that software is safe!

MARVeLus tells you what you need to know to check that a critical software implements its **specifications**.



MORE INFO

Method for Automated Refinement-Type Verification of Lustre (MARVeLus)

ENGINEERING HONORS PROGRAM
UNIVERSITY OF MICHIGAN

Code Example

```
let node main () =
let b = 0 in
let rec (x : {v : int | v >= b})
= 0 fby (x + 1) in ()
```

↓ Generate MARVeLus environment (E)

$$E = (b = 0)$$

↓ Generate verification condition (VC)

$$\frac{\Gamma \vdash s_1 : \tau\ stream \qquad \Gamma \vdash s_2 : \tau\ stream}{\Gamma \vdash s_1\ fby\ s_2 : \tau\ stream}\ \text{(T-FBY)}$$

$$VC \equiv (E \wedge (x = 0) \Rightarrow (x \geq b)) \wedge (E \wedge (x \geq b) \leftarrow \text{from } \textit{let rec} \wedge (x_{next} = x + 1) \Rightarrow (x_{next} \geq b))$$

↓ Check satisfiability with Z3

$$\neg VC$$

Verification condition is satisfiable ✓