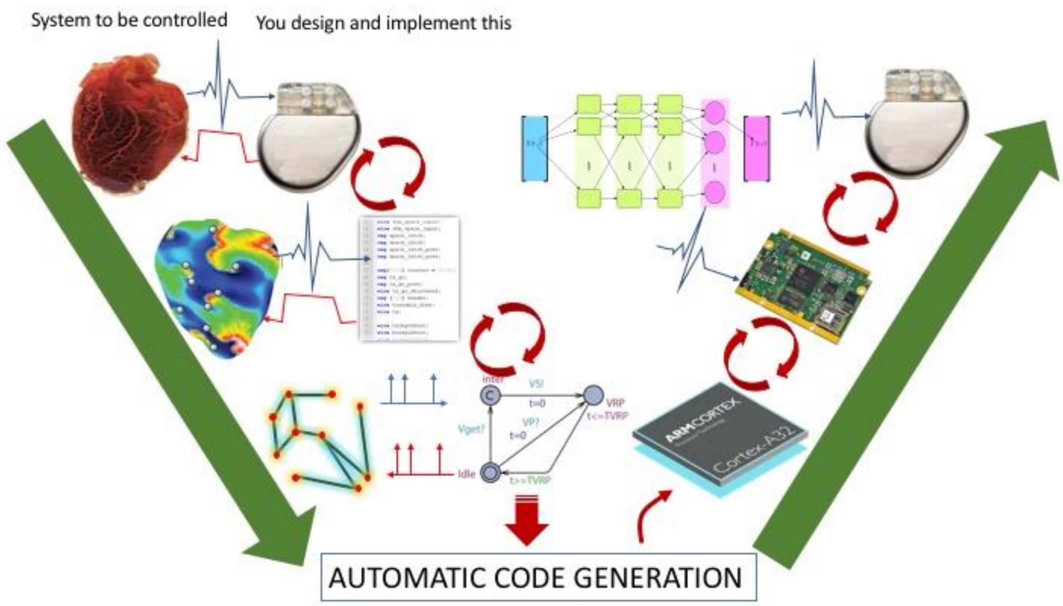
ESE 680

Model-Based Embedded Systems

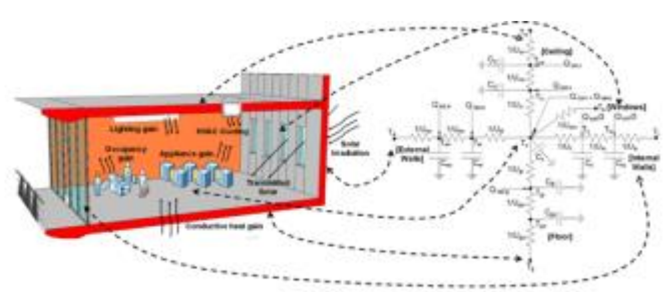
Instructors: Houssam Abbas and Rahul Mangharam



Different models will give you different guarantees about your design's performance and correctness. Starting from a real animal heart on the top left, going to a high-fidelity and expensive PDE model, to a simple but verifiable timed automata model, each stage gives different information and confidence levels in our device design.

What does it take to design and implement a life-saving cardiac defibrillator? The lives of at least 300,000 people worldwide depend on the answer, every year. This course will lead you to the answer, in three projects at the cutting edge of embedded systems design, covering energy-efficient buildings, cardiac medical devices, and automotive controllers. From ODEs to Recurrent Neural Networks, all models are fair game!

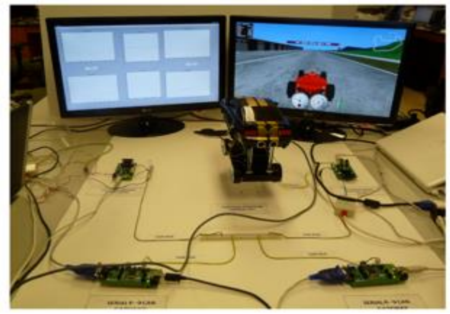
Too Late for Planet Earth



Objectives: *System identification of linear systems, controller design, RLC circuit models of non-electrical phenomena*
Tools: *Matlab/Simulink, MLE+*
Models: *linear systems, first-principles physical modeling*

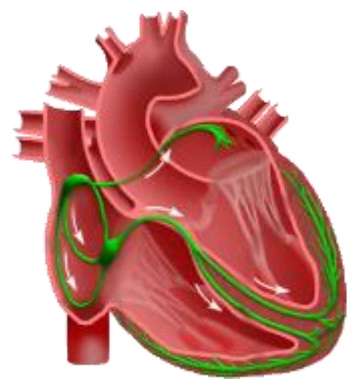
You've been asked to design a network of 1000s of controllers for the air conditioning of large building, and you can't begin to understand the underlying physical models. In this project you will learn how to replace the complex black box model with a simpler linear system using System Identification, how principles of physics can be used to create a 'thermal' RC-network model of the energy-use dynamics of any building, and how to design and test controllers for it. We will also explore alternative data-driven methods for building modeling.

AutoBots, Plug-n-Play!



Objectives: *Hardware-in-the-loop testing, code generation, automated testing*
Tools: *Simulink Coder, SL Design Verifier or Reactis.*
Models: *Simulation black-box model.*
You like to play games, you like to race, and you hate bugs. Luckily, we will be designing controllers, automatically generating C code from them, generating tests to exercise the code, and running our code on ECUs interfaced with the TORCS game engine. Then we play racing games while earning credit.

Your Code is Killing me. Literally.



Objectives: *Finite and Timed automata, appreciating the utility and challenges of formal proofs.*
Tools: *UPPAAL.*
Models: *Timed automata, deterministic automata, Partial Differential Equations.*
Life-saving medical devices, like pacemakers and defibrillators, require a rigorous approach to verifying their safety. Testing, in which the device is fed different inputs and its behavior observed, cannot *guarantee* correctness and freedom from faults. Formal verification, on the other hand, provides such a guarantee.
In this project, we will first get an overview of the fascinating computer models of the human heart out there, and get a feel for their capabilities. Then we will focus on timed automata, which allow us to formally and rigorously prove freedom from faults. The lessons and perspective you learn in this project will be useful to you in almost any embedded systems project you tackle in the future.