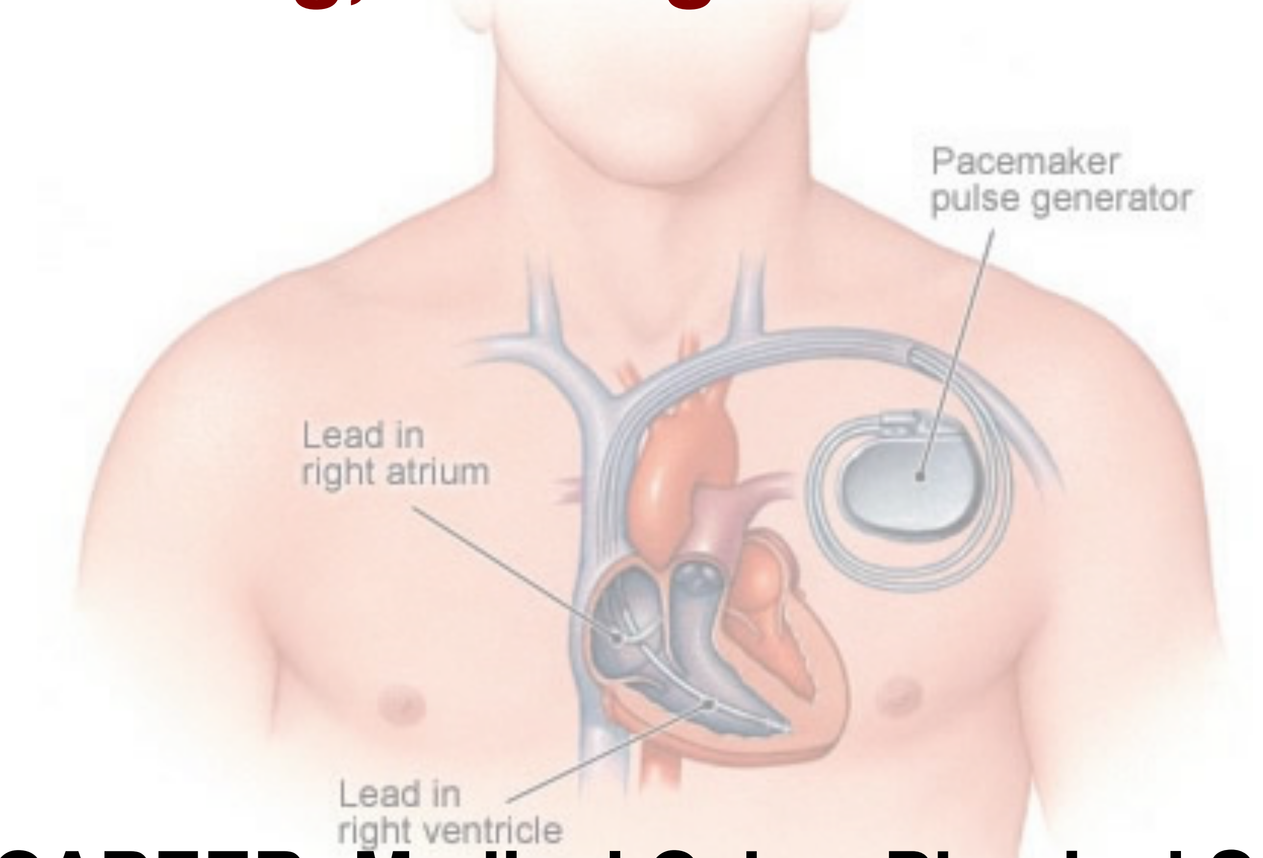
Closing the loop with Medical Device Modeling, Testing and Verification

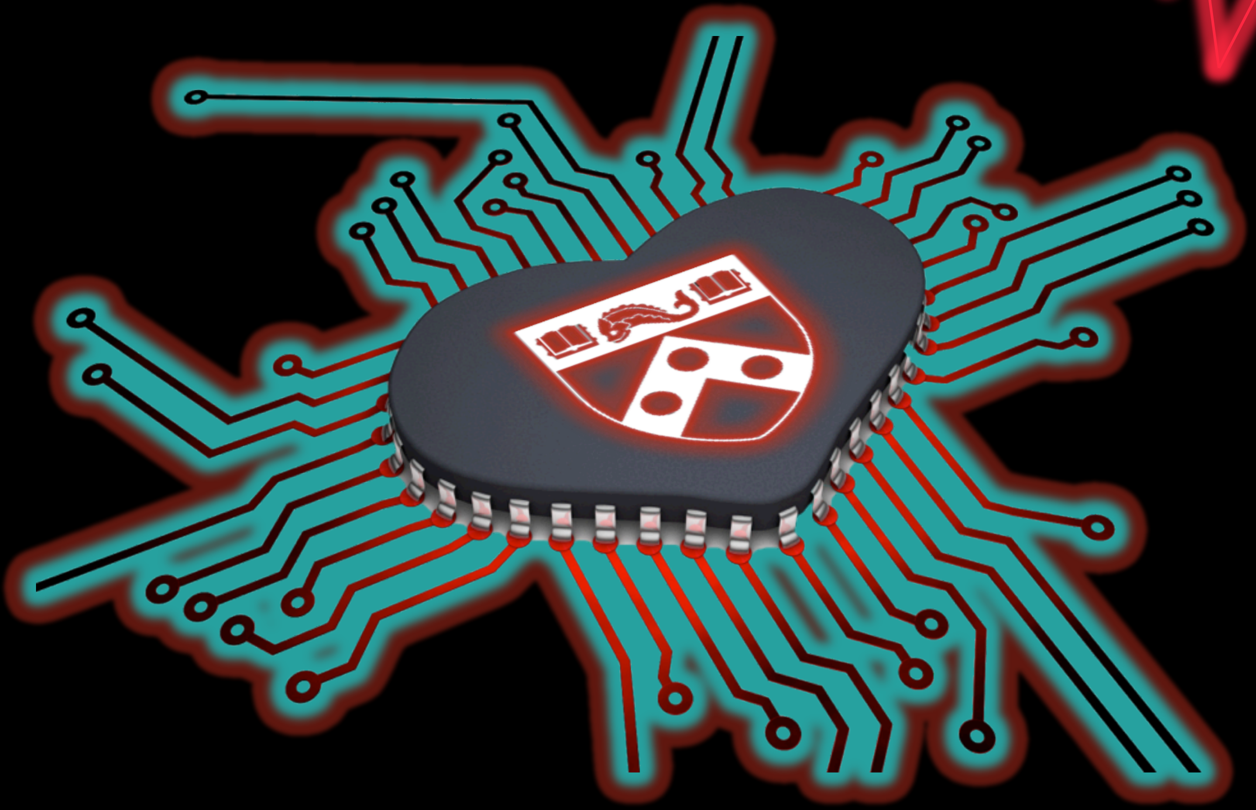


NSF CAREER: Medical Cyber-Physical Systems

PI: Rahul Mangharam

Heart-on-a-Chip Platform

for medical device software verification, testing and certification



Rahul Mangharam, Zhihao Jiang and Miroslav Pajic

University of Pennsylvania

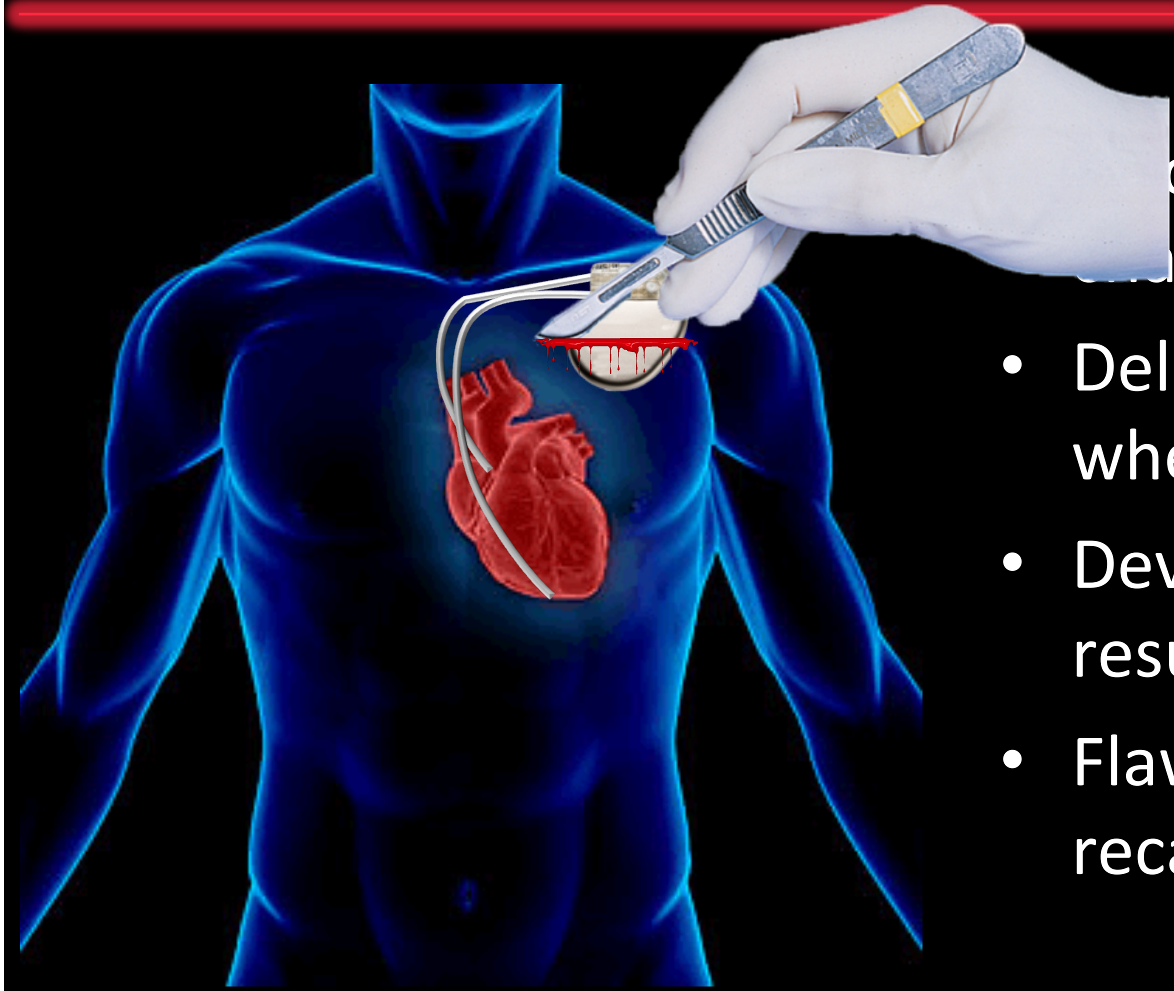
First, the bad news

- Over 600,000 cardiac medical devices recalled from 1990-2000
- 40% of recent recalls were due to software issues

Implantable Pacemaker

Implantable Cardioverter-Defibrillator (ICD)

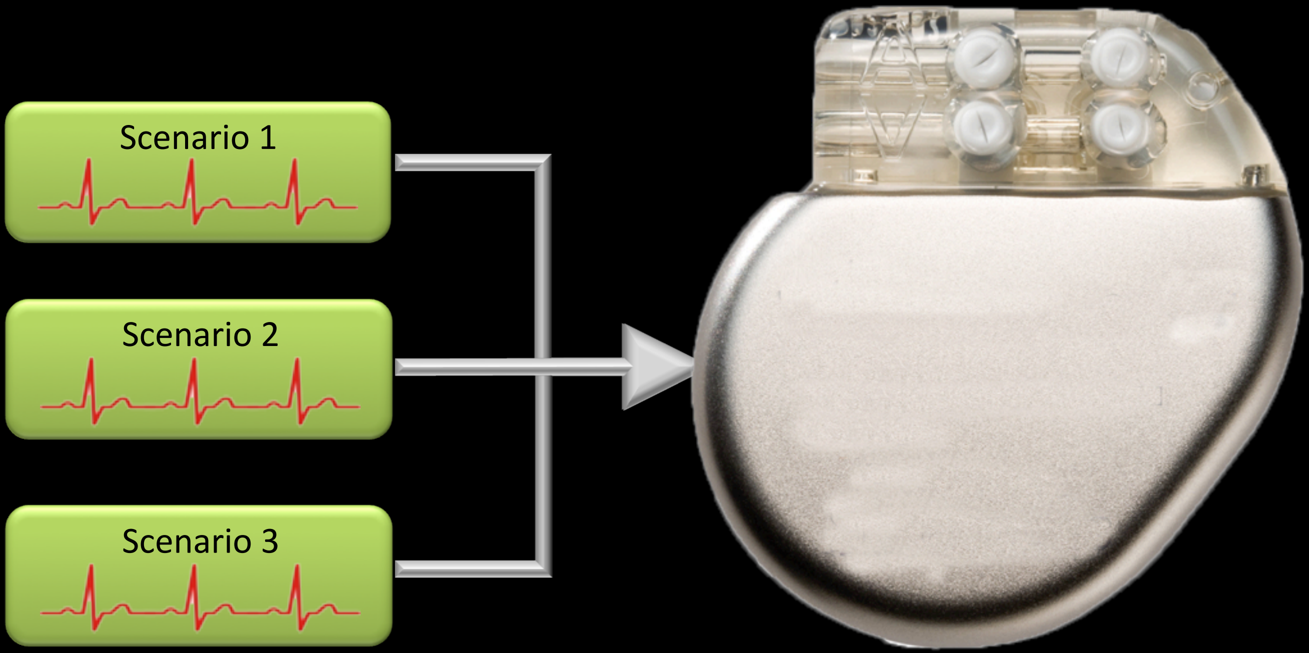
Implantable Pacemaker



o leads in heart
ambers

- Deliver electrical signals when heart rate is low
- Device malfunction may result in injury or death
- Flawed devices are recalled

Open-Loop Testing



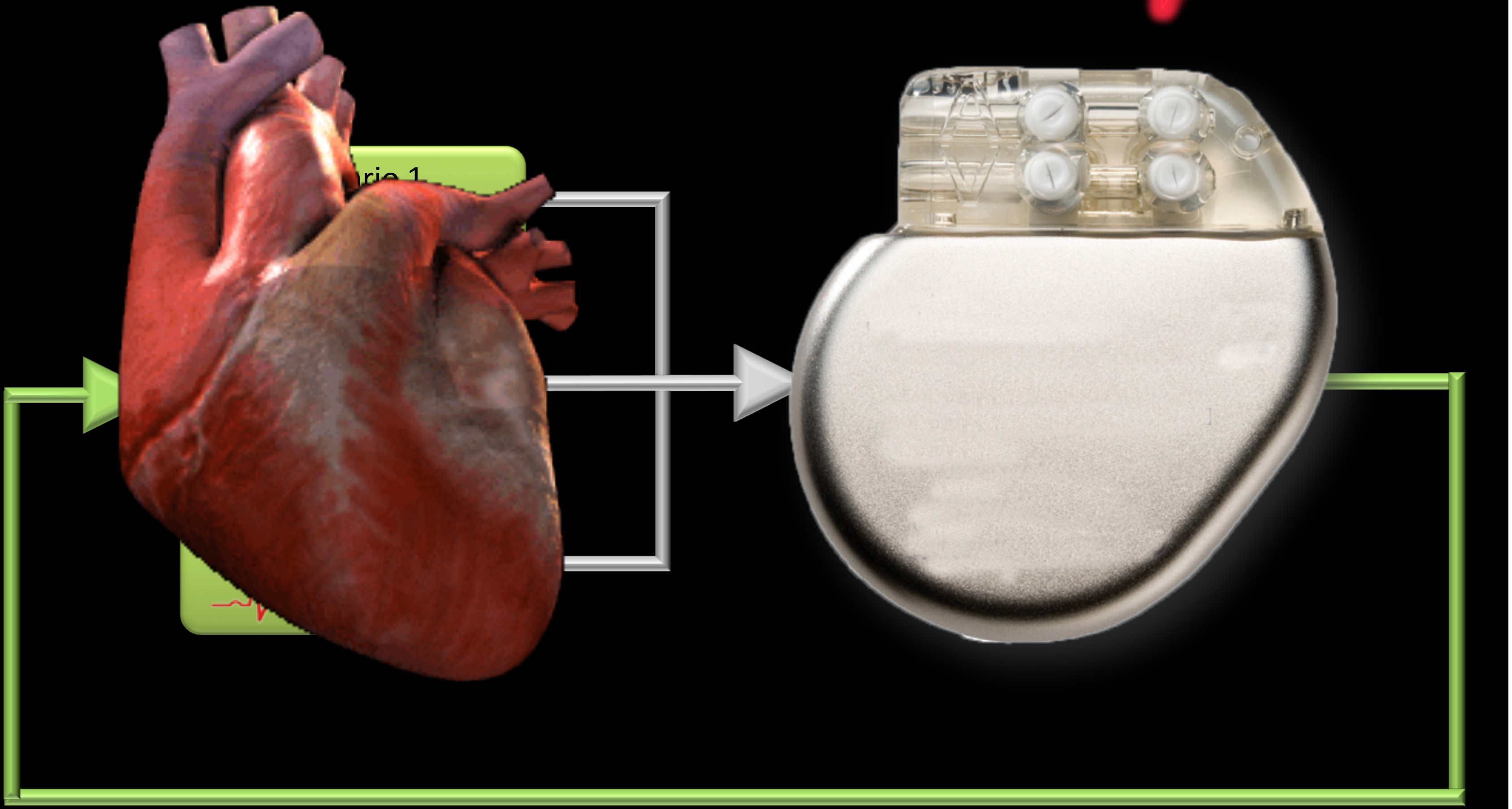
OUR GOAL

Verify and test the device software

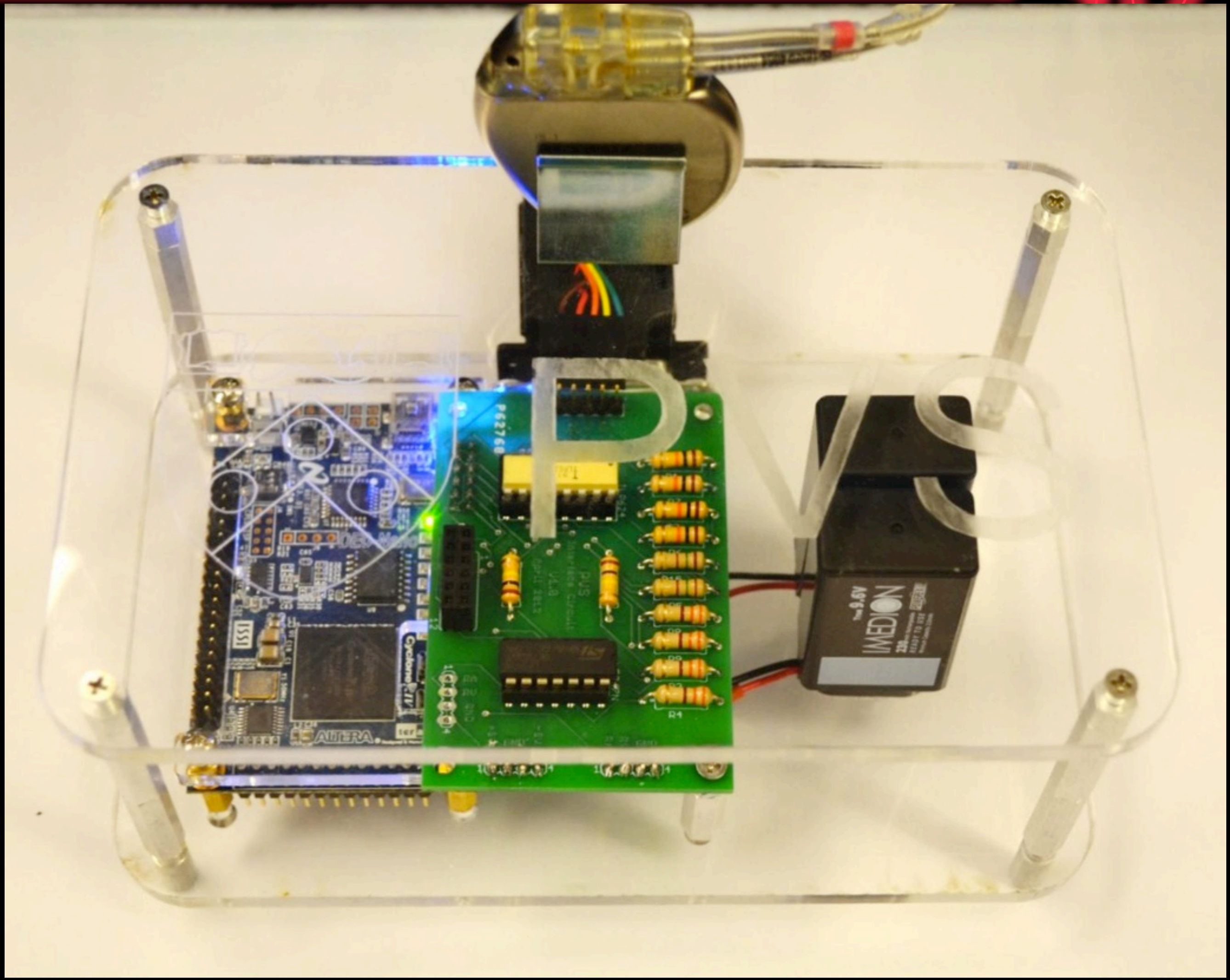
in closed-loop

with its physical environment

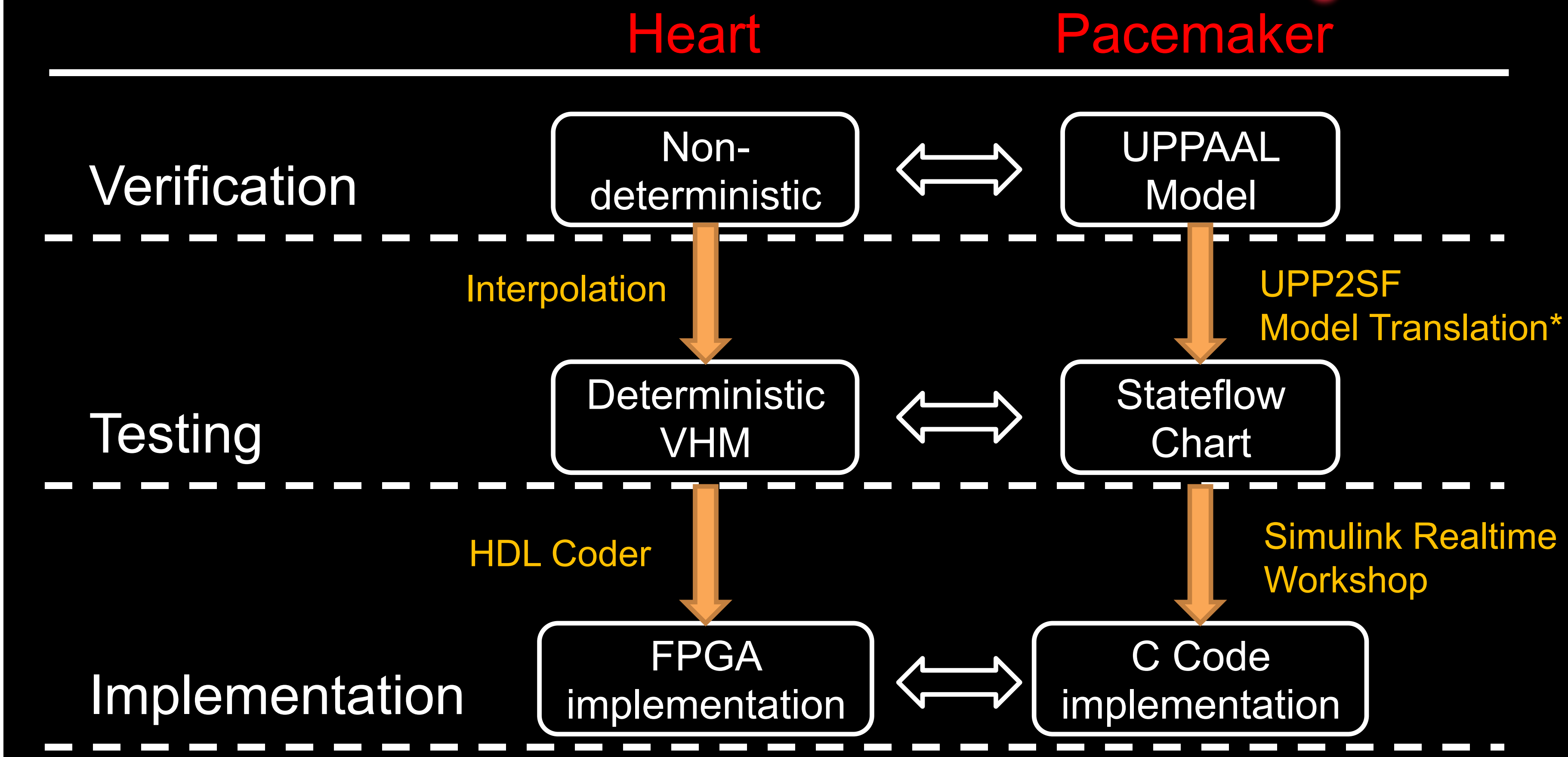
Closed-Loop Evaluation



Demo of Configurable Heart



Model-based Pacemaker Design



Heart

Pacemaker

Verification

Non-deterministic ⟺ UPPAAL Model

Interpolation

UPP2SF Model Translation*

Testing

Deterministic VHM ⟺ Stateflow Chart

HDL Coder

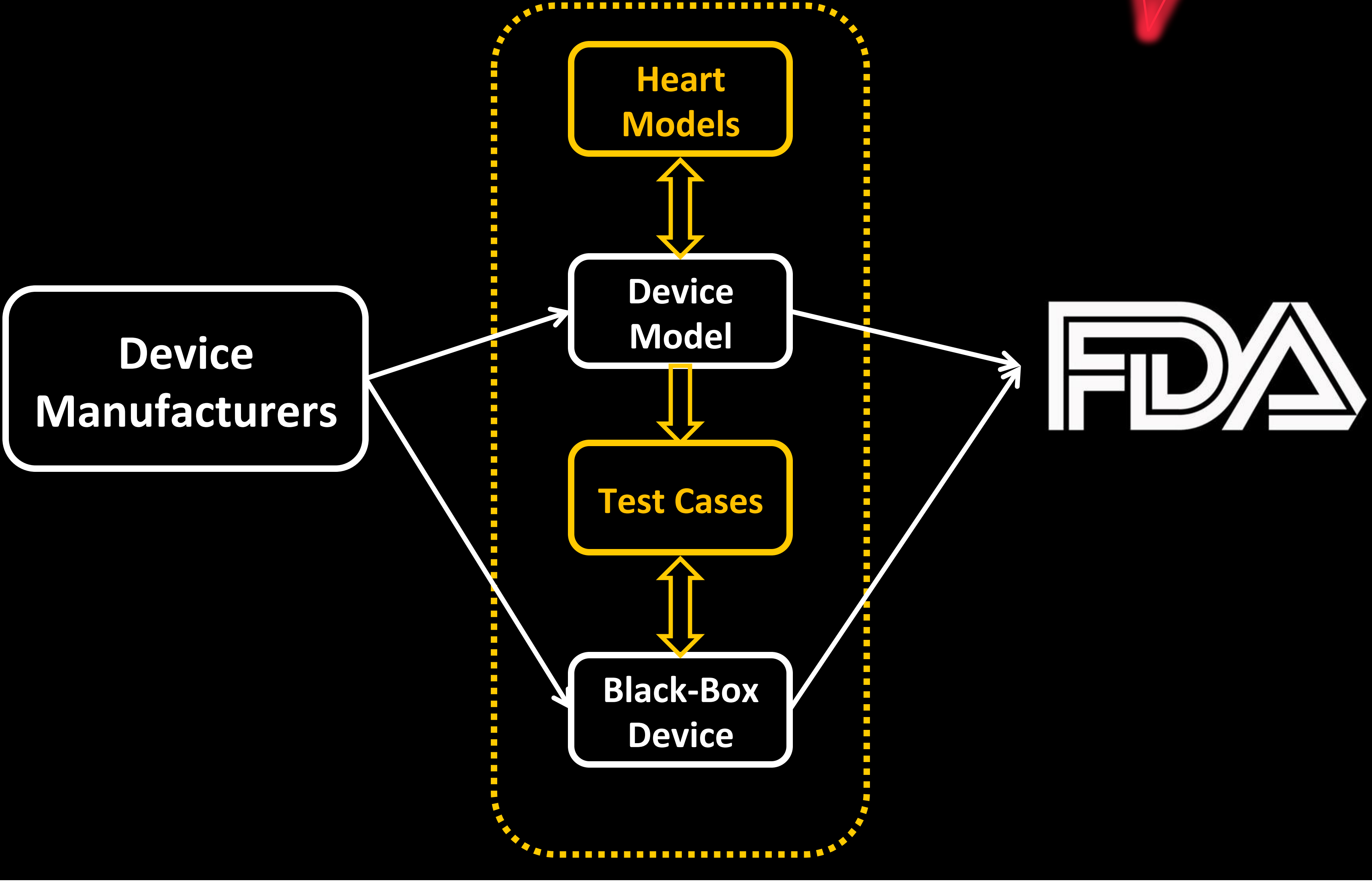
Simulink Realtime Workshop

Implementation

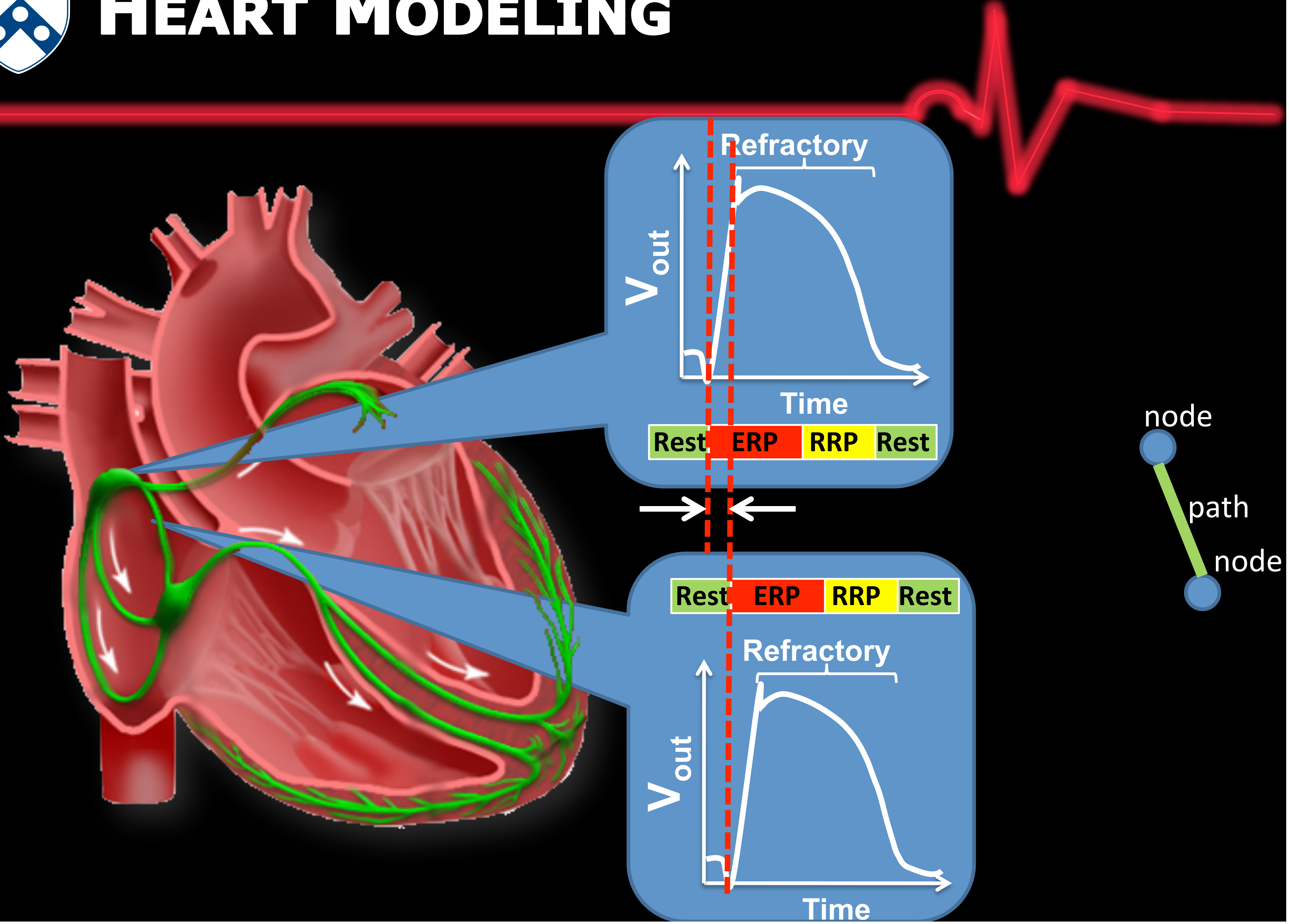
FPGA implementation ⟺ C Code implementation

Published in: ECRTS'10, EMBC'10, EMBC'11, ICCPS'11, IEEE Proceedings'11, TACAS'12, STTT'13, RTAS'12 Best Student Paper Award

Model-Based Software Certification



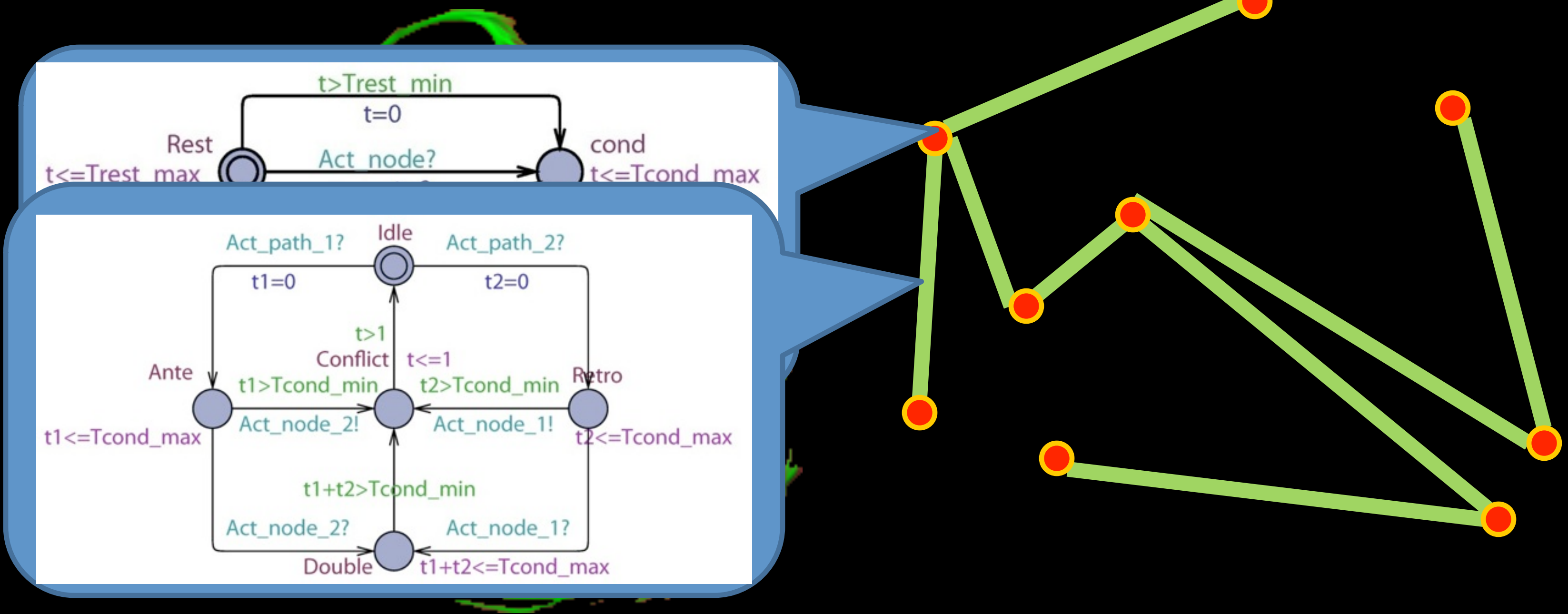
Heart Modeling



Heart Modeling

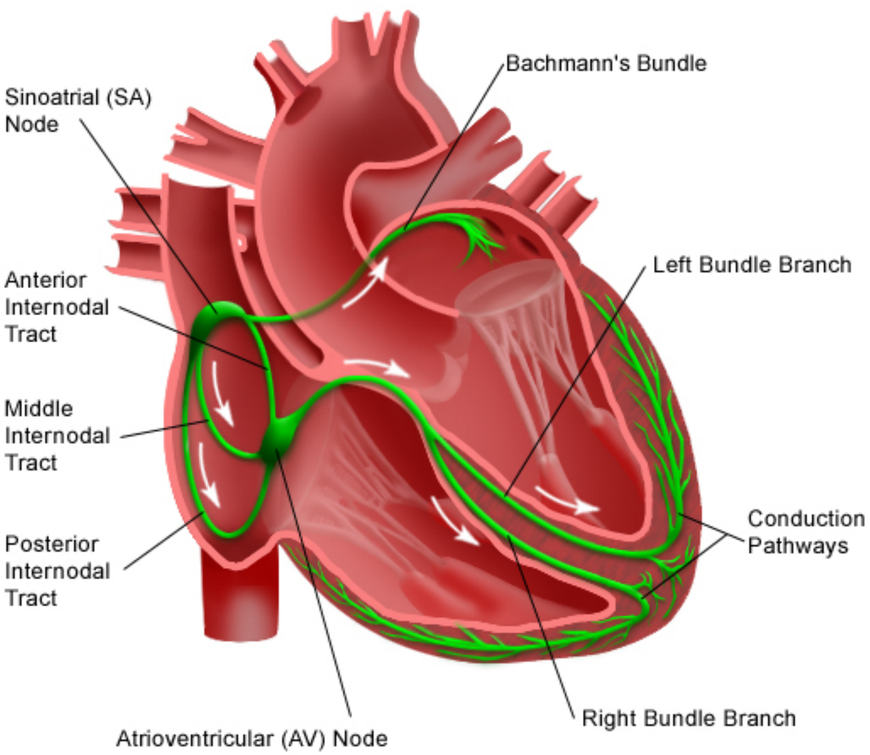
Node Automata

Path Automata

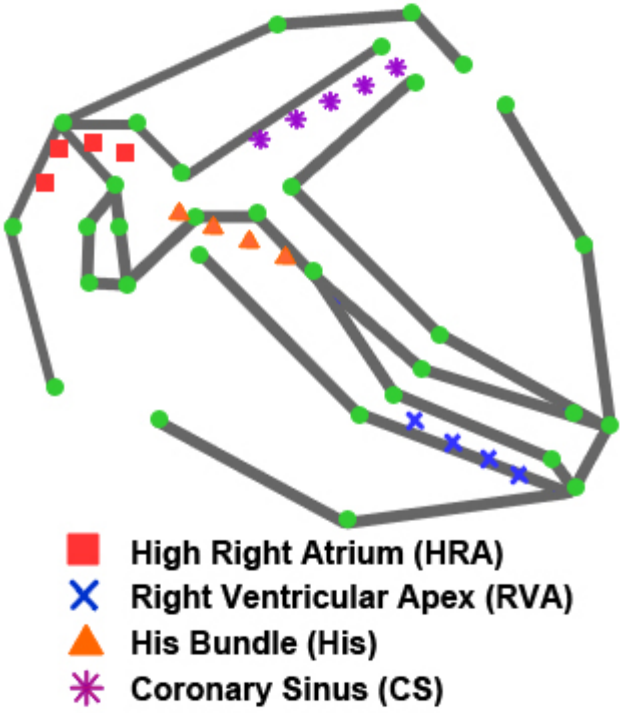


Conduction System
Timed Automata Model

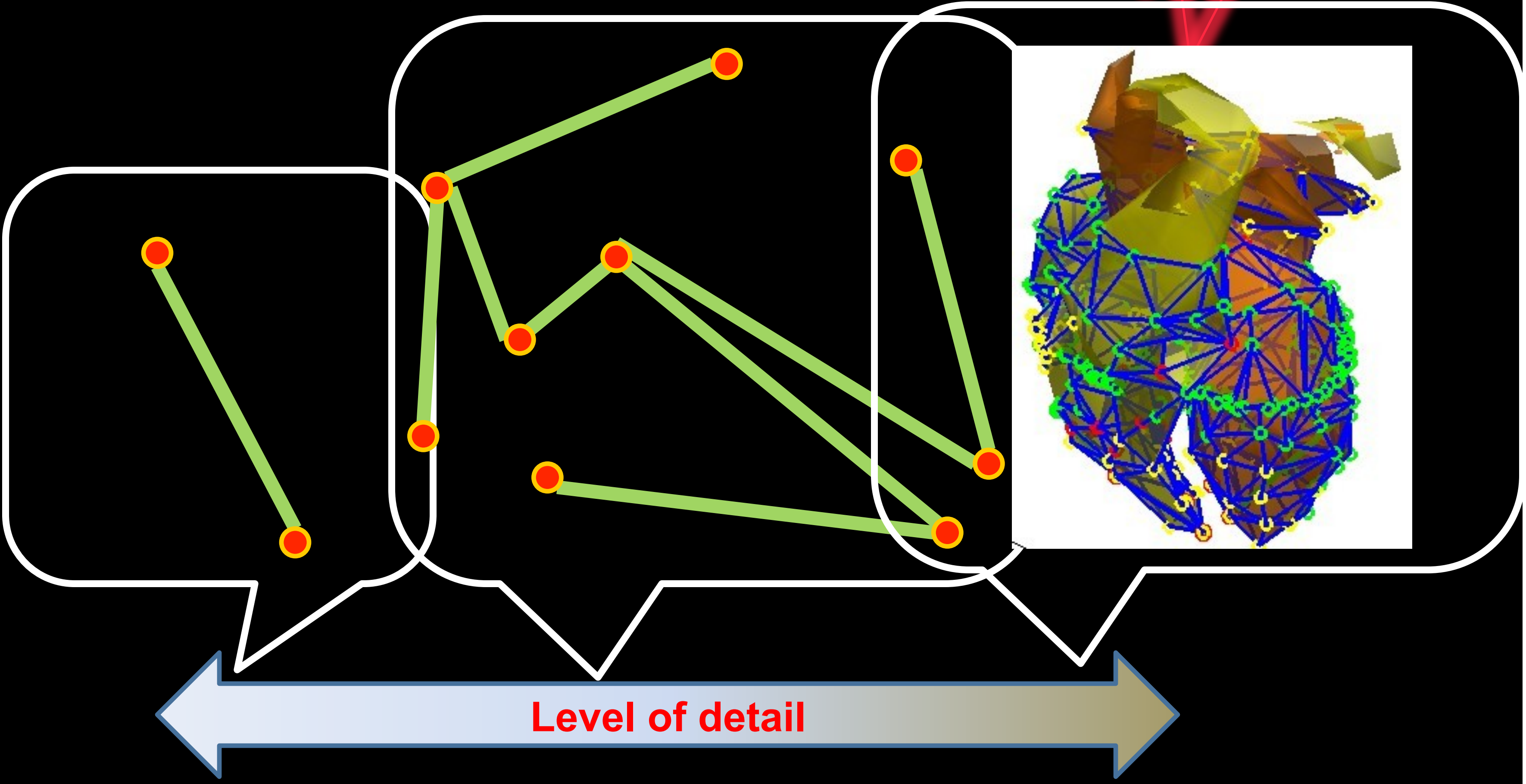
- 32 nodes
- 33 paths



Electrical System of the Heart



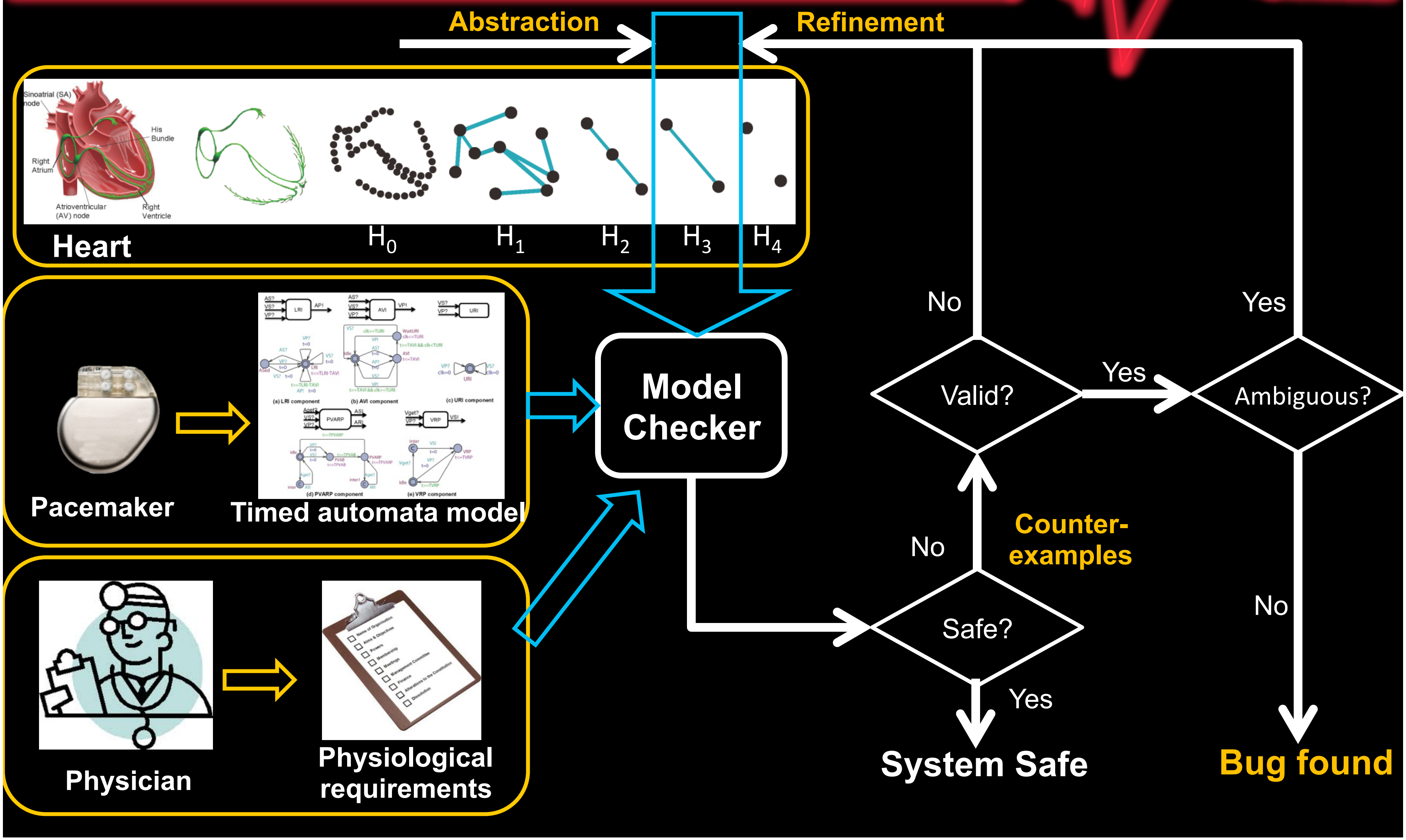
GRANULARITY



Abstraction

Refinement

Counter-Example-Guided Abstraction & Refinement



Abstraction

Refinement

Heart

Pacemaker

Timed automata model

Physician

Physiological requirements

Model Checker

Safe?

Yes

System Safe

No

Counter-examples

Valid?

No

Yes

Ambiguous?

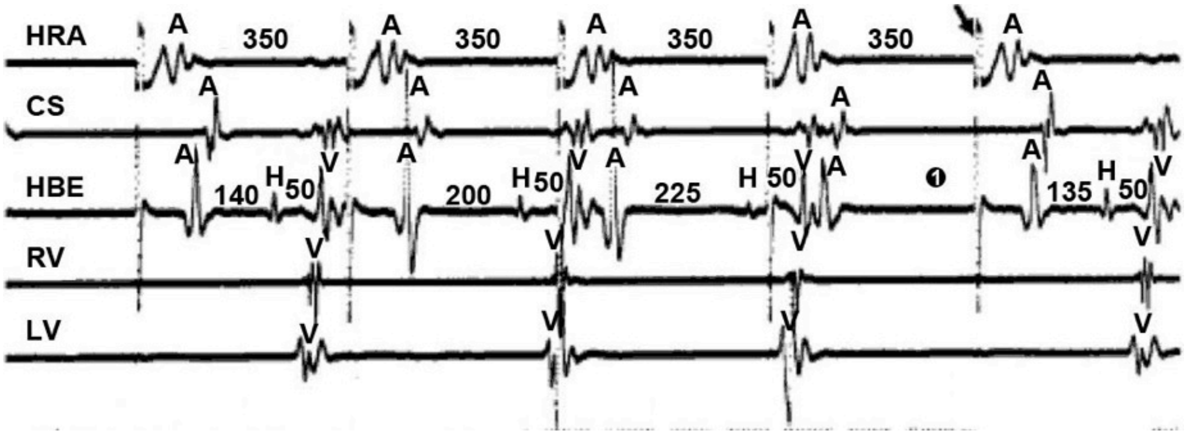
Yes

No

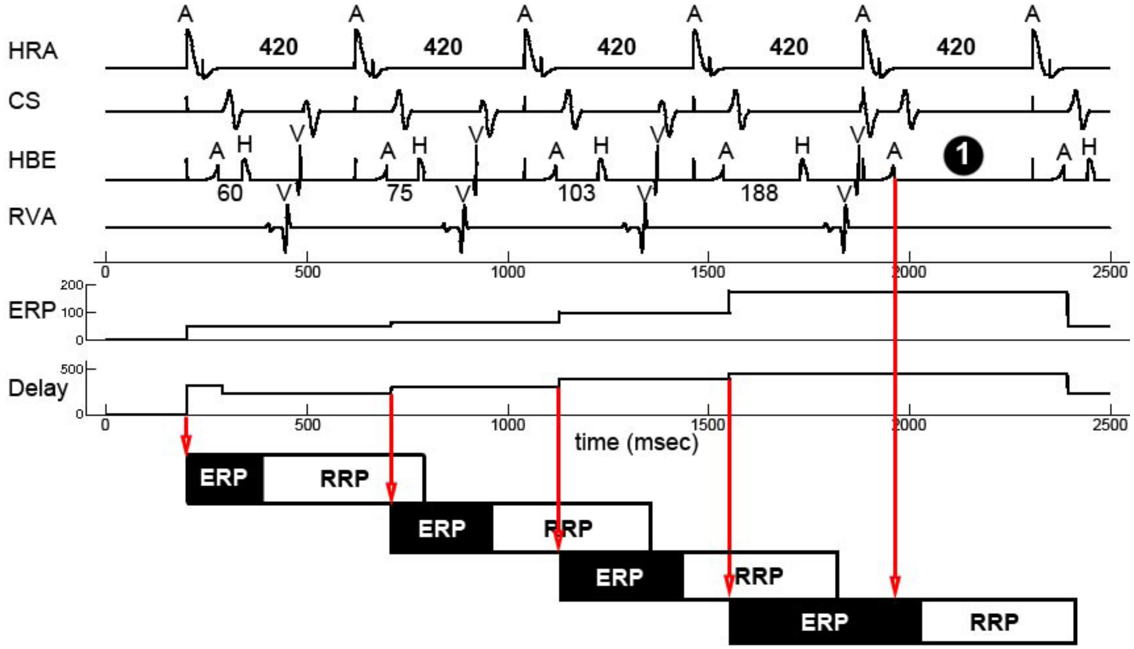
Bug found

Validation Cases:
Wenckebach AV nodal response

Real Patient



Heart Model



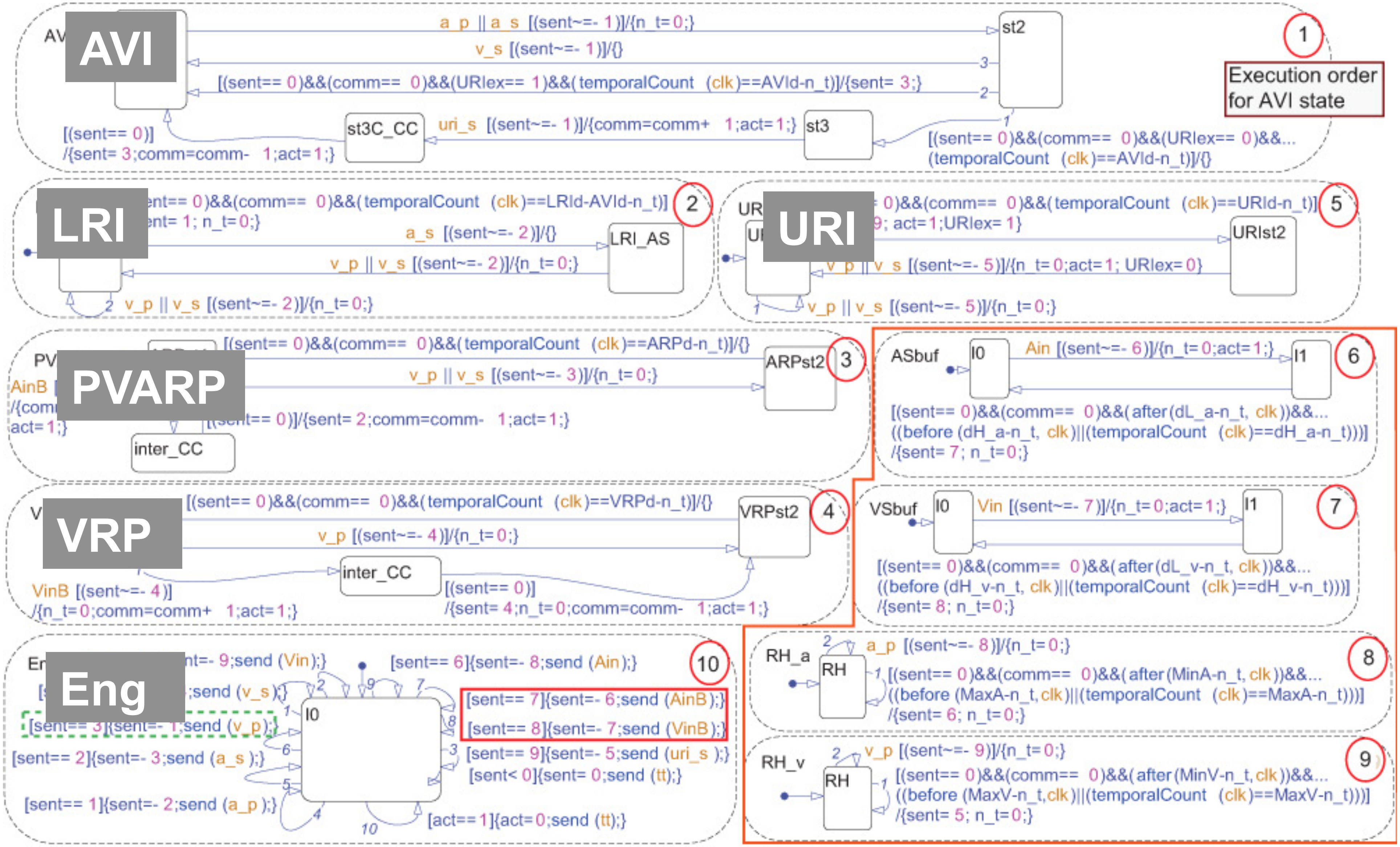
This case shows that our model is not a signal generator.
It responds to a sequence of stimuli interactively

Heart Modeling Coverage

1. Normal Sinus Rhythm
2. Bradycardia
3. Heart block
4. Supraventricular Tachycardia
5. Lead displacement
6. Lead Cross-talk and race conditions
7. Pacemaker Mediated Tachycardia
8. Endless-loop Tachycardia
9. Pacemaker Mode-switch

Pacemaker Case Study – Extracted Stateflow Model

of the Closed-loop Systems



AVI
AV
a_p || a_s [(sent~=- 1)]/{n_t=0;}
st2
v_s [(sent~=- 1)]/{}
3
[(sent== 0)&&(comm== 0)&&(URIex== 1)&&(temporalCount (clk)==AVId-n_t)]/{sent= 3;}
2
1
Execution order for AVI state
[(sent== 0)]
/{sent= 3;comm=comm- 1;act=1;}
st3C_CC
uri_s [(sent~=- 1)]/{comm=comm+ 1;act=1;}
st3
[(sent== 0)&&(comm== 0)&&(URIex== 0)&&...
(temporalCount (clk)==AVId-n_t)]/{}
LRI
ent== 0)&&(comm== 0)&&(temporalCount (clk)==LRId-AVId-n_t)]
ent= 1; n_t=0;}
2
a_s [(sent~=- 2)]/{}
LRI_AS
v_p || v_s [(sent~=- 2)]/{n_t=0;}
2 v_p || v_s [(sent~=- 2)]/{n_t=0;}
URI
UR
UI
0)&&(comm== 0)&&(temporalCount (clk)==URId-n_t)]
9; act=1;URIex= 1}
5
URIst2
v_p || v_s [(sent~=- 5)]/{n_t=0;act=1; URIex= 0}
1 v_p || v_s [(sent~=- 5)]/{n_t=0;}
PVARP
PV
[(sent== 0)&&(comm== 0)&&(temporalCount (clk)==ARPd-n_t)]/{}
ARPst2
3
v_p || v_s [(sent~=- 3)]/{n_t=0;}
AinB [
/{comm
act=1;}
[(sent== 0)]/{sent= 2;comm=comm- 1;act=1;}
inter_CC
ASbuf
I0
Ain [(sent~=- 6)]/{n_t=0;act=1;}
I1
6
[(sent== 0)&&(comm== 0)&&(after(dL_a-n_t, clk))&&...
((before (dH_a-n_t, clk)||(temporalCount (clk)==dH_a-n_t))]
/{sent= 7; n_t=0;}
VRP
V
[(sent== 0)&&(comm== 0)&&(temporalCount (clk)==VRPd-n_t)]/{}
VRPst2
4
v_p [(sent~=- 4)]/{n_t=0;}
inter_CC
VinB [(sent~=- 4)]
/{n_t=0;comm=comm+ 1;act=1;}
[(sent== 0)]
/{sent= 4;n_t=0;comm=comm- 1;act=1;}
VSbuf
I0
Vin [(sent~=- 7)]/{n_t=0;act=1;}
I1
7
[(sent== 0)&&(comm== 0)&&(after(dL_v-n_t, clk))&&...
((before (dH_v-n_t, clk)||(temporalCount (clk)==dH_v-n_t))]
/{sent= 8; n_t=0;}
Eng
En
nt=- 9;send (Vin);}
[sent== 6]{sent=- 8;send (Ain);}
10
;send (v_s);}
[sent== 3]{sent=- 1;send (v_p);}
I0
2
9
7
1
8
6
3
5
4
10
[sent== 7]{sent=- 6;send (AinB);}
[sent== 8]{sent=- 7;send (VinB);}
[sent== 2]{sent=- 3;send (a_s);}
[sent== 9]{sent=- 5;send (uri_s);}
[sent< 0]{sent= 0;send (tt);}
[sent== 1]{sent=- 2;send (a_p);}
[act== 1]{act=0;send (tt);}
RH_a
RH
2 a_p [(sent~=- 8)]/{n_t=0;}
8
1 [(sent== 0)&&(comm== 0)&&(after(MinA-n_t, clk))&&...
((before (MaxA-n_t, clk)||(temporalCount (clk)==MaxA-n_t))]
/{sent= 6; n_t=0;}
RH_v
RH
2 v_p [(sent~=- 9)]/{n_t=0;}
9
1 [(sent== 0)&&(comm== 0)&&(after(MinV-n_t, clk))&&...
((before (MaxV-n_t, clk)||(temporalCount (clk)==MaxV-n_t))]
/{sent= 5; n_t=0;}

Generated C Code

Listing 1. **bitsForTID0** definition

```
struct {
    uint_T is_AVI:3;
    uint_T is_LRI:2;
    uint_T is_PVARP:2;
    uint_T is_VRP:2;
    uint_T is_URI:2;
    uint_T is_active_AVI:1;
    uint_T is_active_LRI:1;
    uint_T is_active_PVARP:1;
    uint_T is_active_VRP:1;
    uint_T is_active_URI:1;
    uint_T is_active_Eng:1;
    uint_T is_Eng:1;
    uint_T URI_ex:1;
  } bitsForTID0;
```

Listing 5. **broadcast_tt()** procedure

```
static void broadcast_tt(void) {
  int16_T sf_previousEvent;
  sf_previousEvent = _sfEvent_;
  _sfEvent_ = event_tt;
  c1_ChartName();
  _sfEvent_ = sf_previousEvent;
}
```

Listing 2. **Rt_OneStep** procedure

```
detect active inputs;
for each of the input events {
  if EventName is active   {
    sf_previousEvent = _sfEvent_;
    _sfEvent_ = EventName;
    c1_ChartName();
    _sfEvent_ = sf_previousEvent;
  }
}

update the outputs;
update the input events states;
```

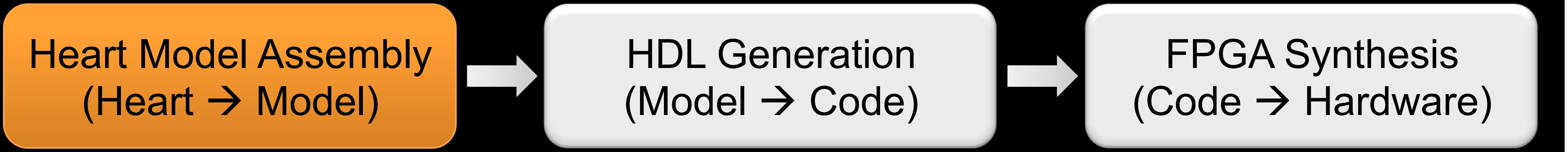
Listing 3. **cl_ChartName()** procedure

```
increase counters for _sfEvent_;
for each parallel state {
  processState();
}
```

Listing 4. **processState()** procedure

```
if (rtDWork.bitsForTID0.is_active_NAME != 0){
  switch (rtDWork.bitsForTID0.is_NAME) {
    case SubStateName1:
      /* the loop below is - checkTrans();*/
      for all transitions in ex. order {
       if transition enabled {
         execution transition actions;
         reset corresponding temporal counters;
         update rtDWork.bitsForTID0.is_NAME;
         break for
       }
      }
      break;
    case SubStateName2:
      checkTrans();
      break;
    ...
    default:
     rtDWork.bitsForTID0.is_NAME=NoActiveChild;
     break;
   }
  }
```

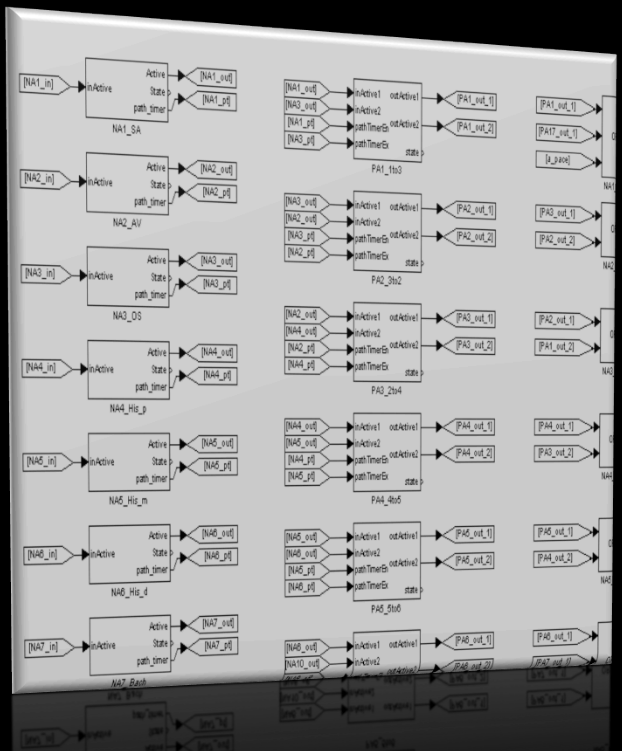
Heart Implementation

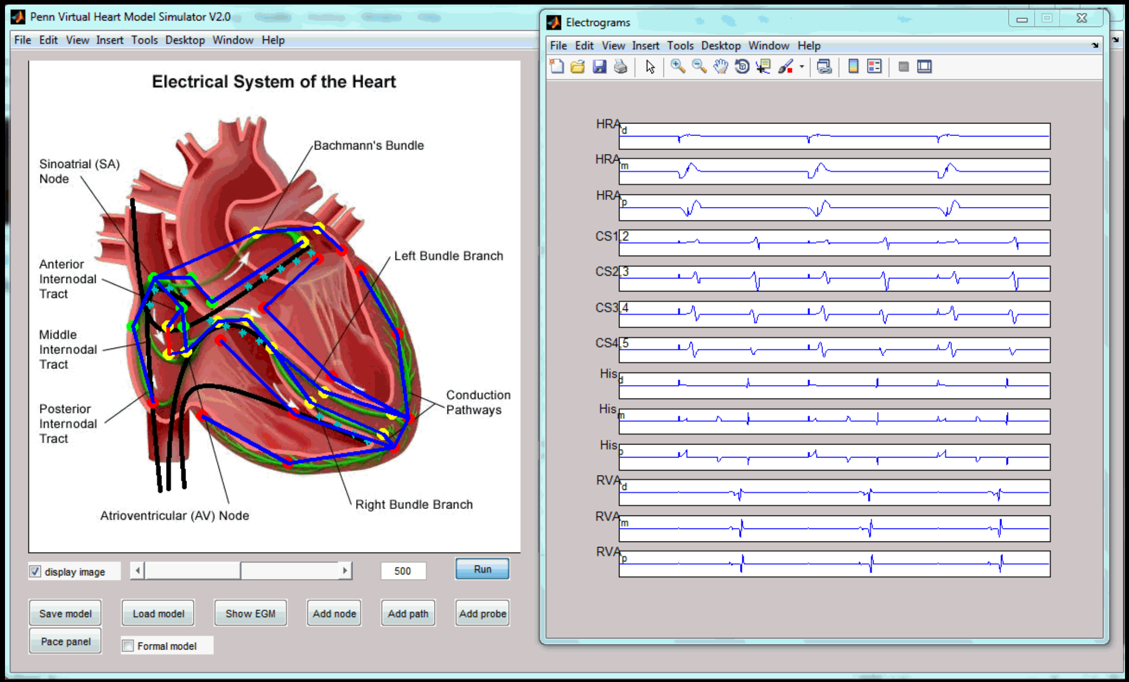


Heart Model Assembly (Heart → Model)

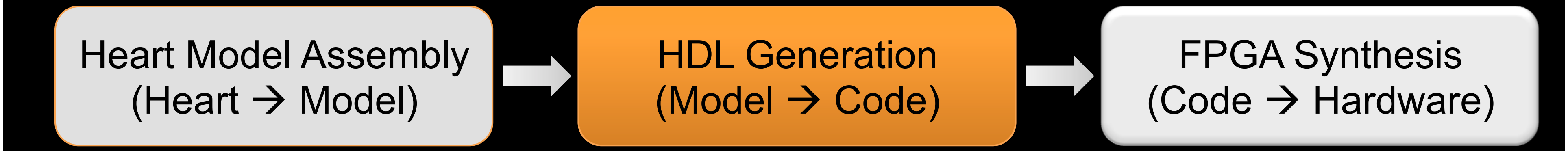
HDL Generation (Model → Code)

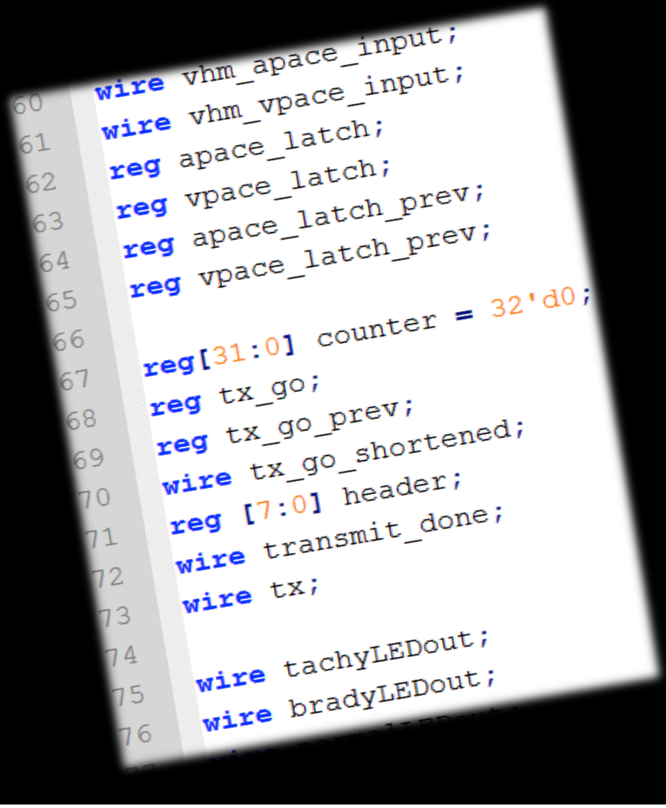
FPGA Synthesis (Code → Hardware)





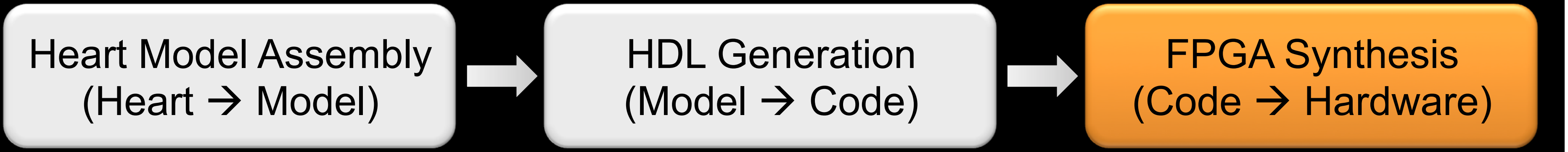
Heart Implementation

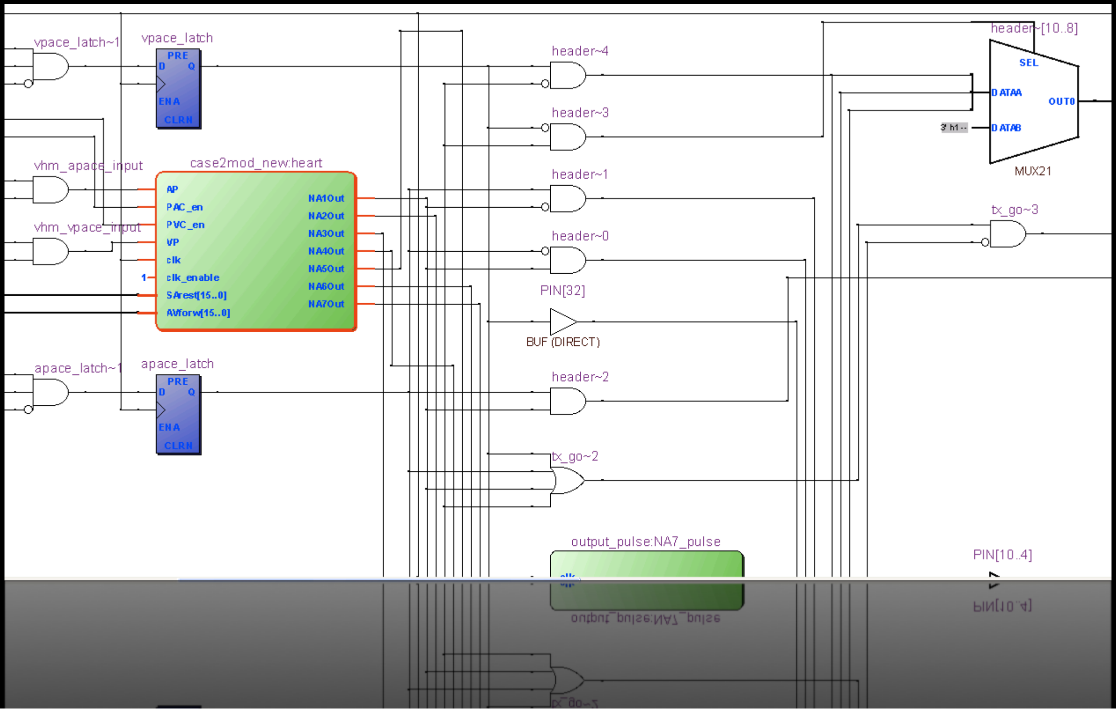


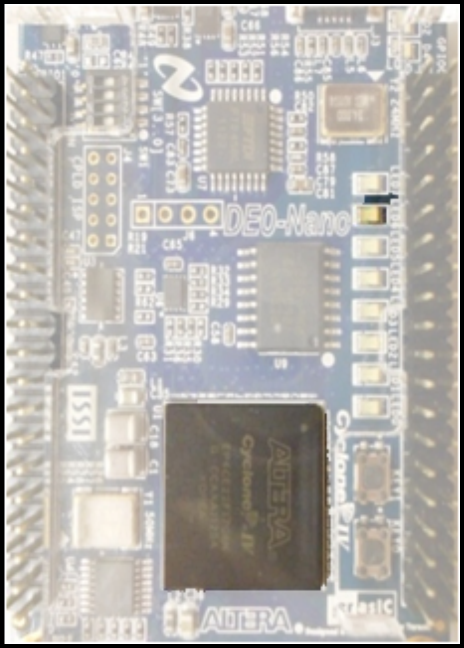


```
wire vhm_apace_input;
wire vhm_vpace_input;
reg apace_latch;
reg vpace_latch;
reg apace_latch_prev;
reg vpace_latch_prev;

reg[31:0] counter = 32'd0;
reg tx_go;
reg tx_go_prev;
wire tx_go_shortened;
reg [7:0] header;
wire transmit_done;
wire tx;

wire tachyLEDout;
wire bradyLEDout;
```

Heart Implementation

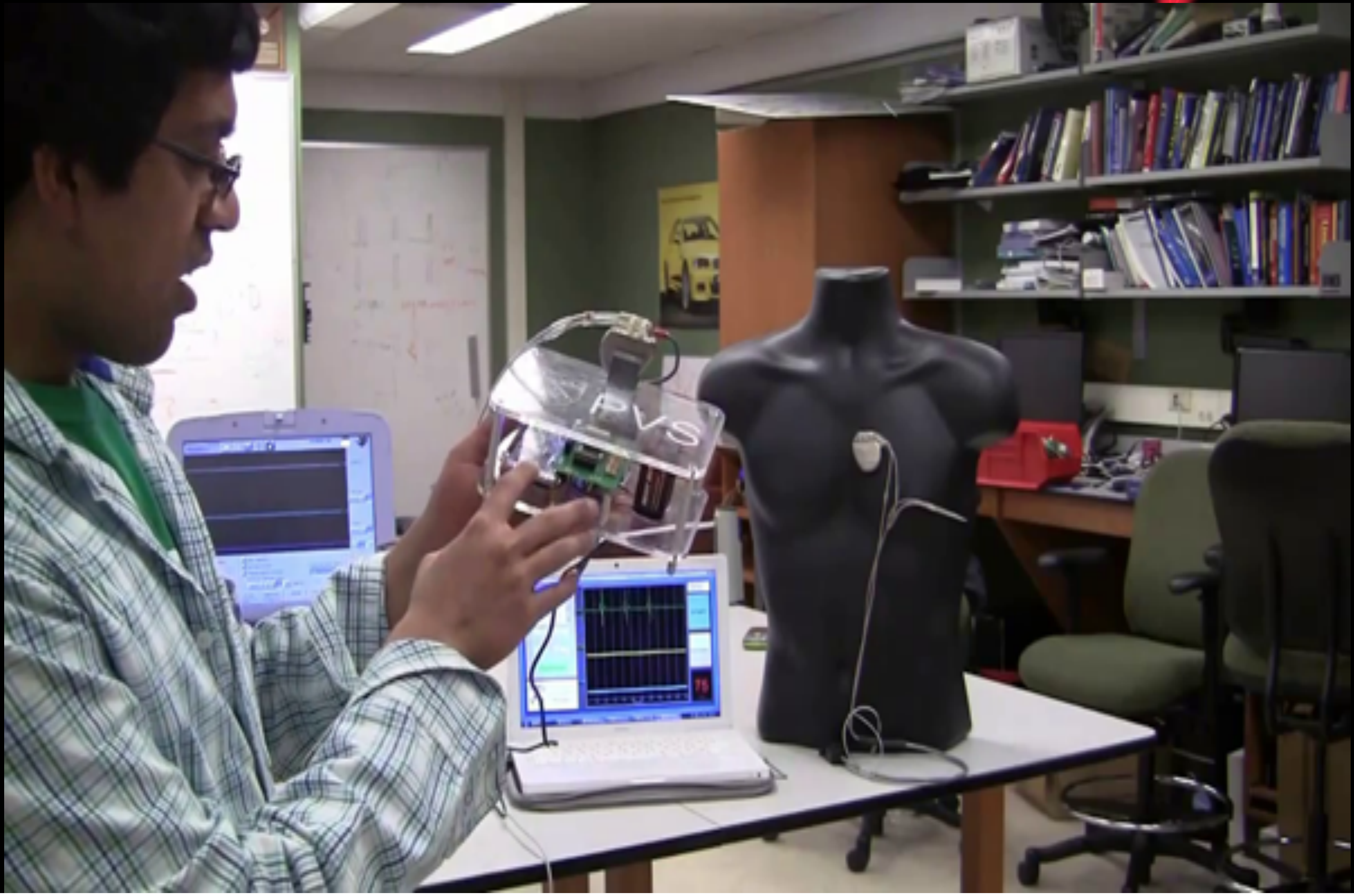






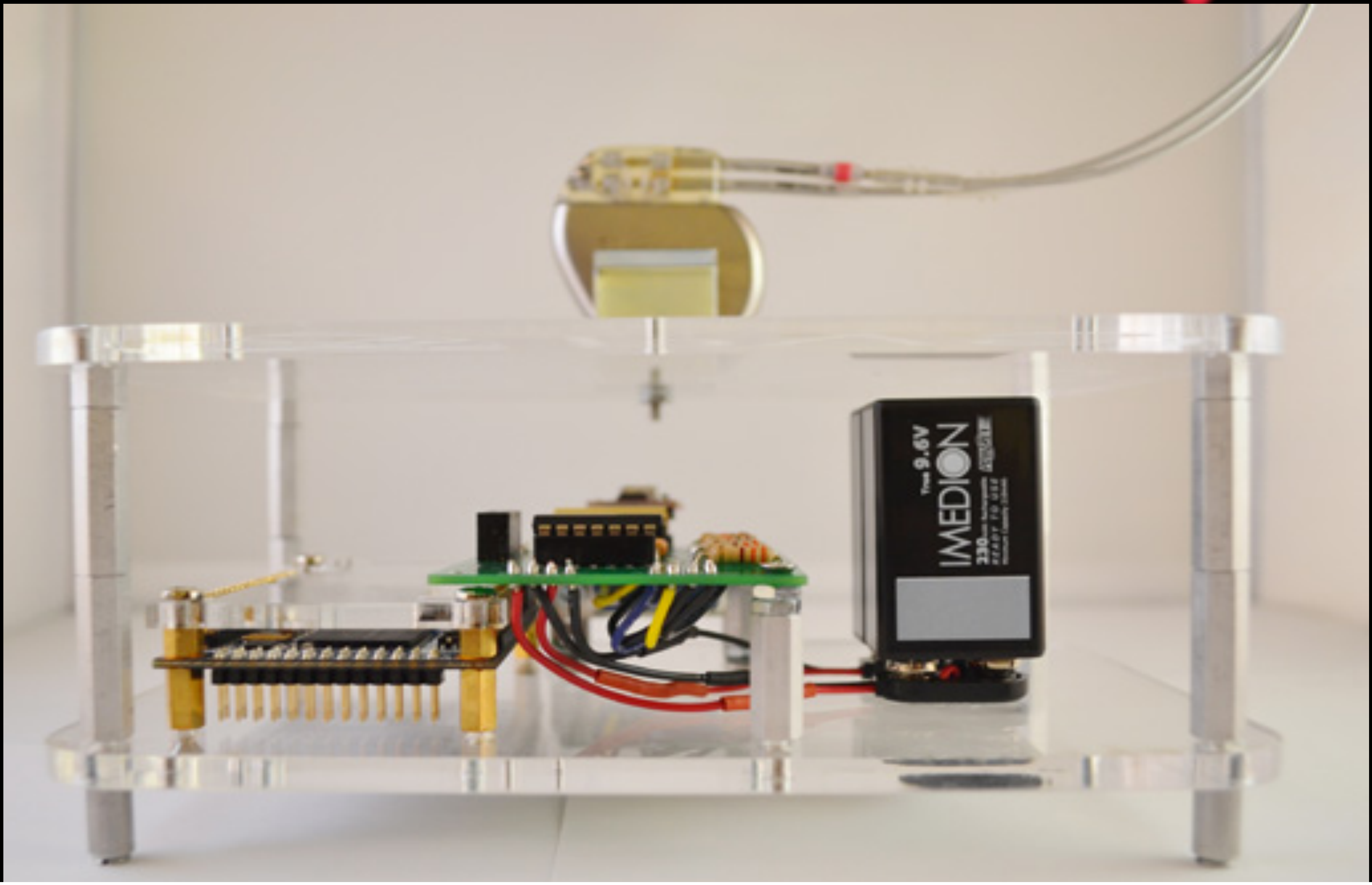
Heart-on-a-Chip Platform

for closed-loop implantable device testing and verification



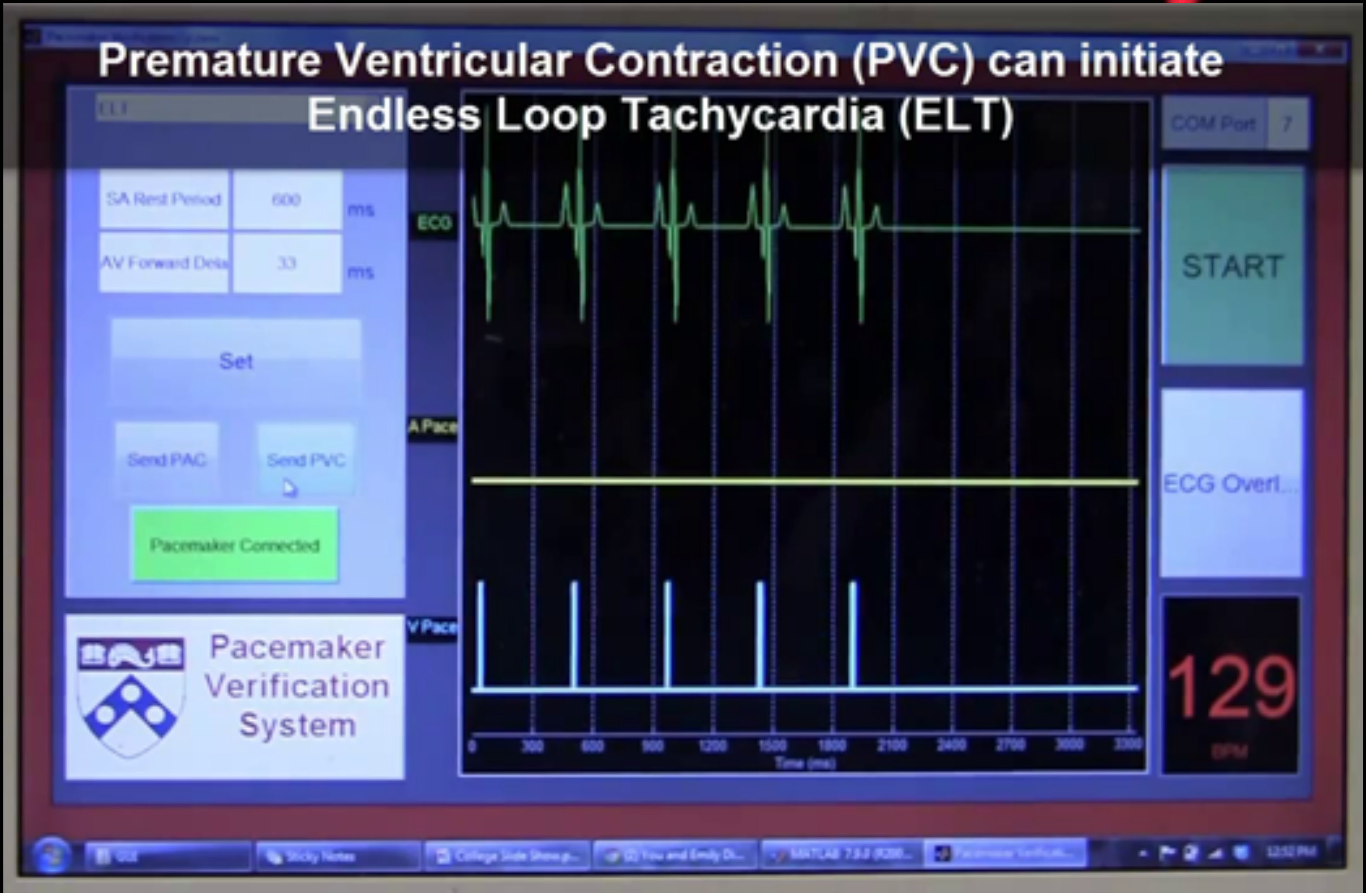
Heart-on-a-Chip Platform

for closed-loop implantable device testing and verification



Heart-on-a-Chip Platform

for closed-loop implantable device testing and verification



Premature Ventricular Contraction (PVC) can initiate Endless Loop Tachycardia (ELT)

SA Rest Period 600 ms

AV Forward Dela 33 ms

COM Port 7

WHO CAN USE THIS PLATFORM?

FDA

Boston Scientific

ST. JUDE MEDICAL™
MORE CONTROL. LESS RISK.

Medtronic

“Let our heart catch bugs before your heart does.”