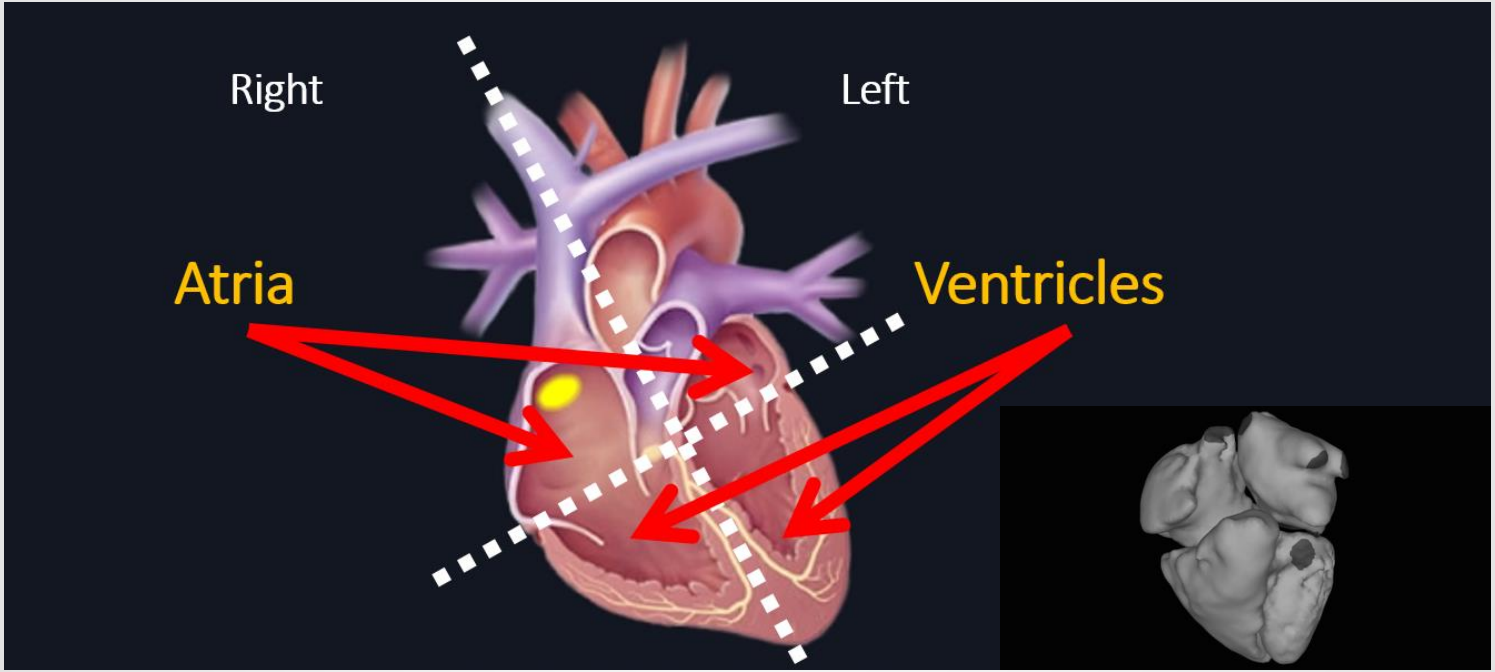
Software Design of a DDD Pacemaker

Vera Zhang

Background

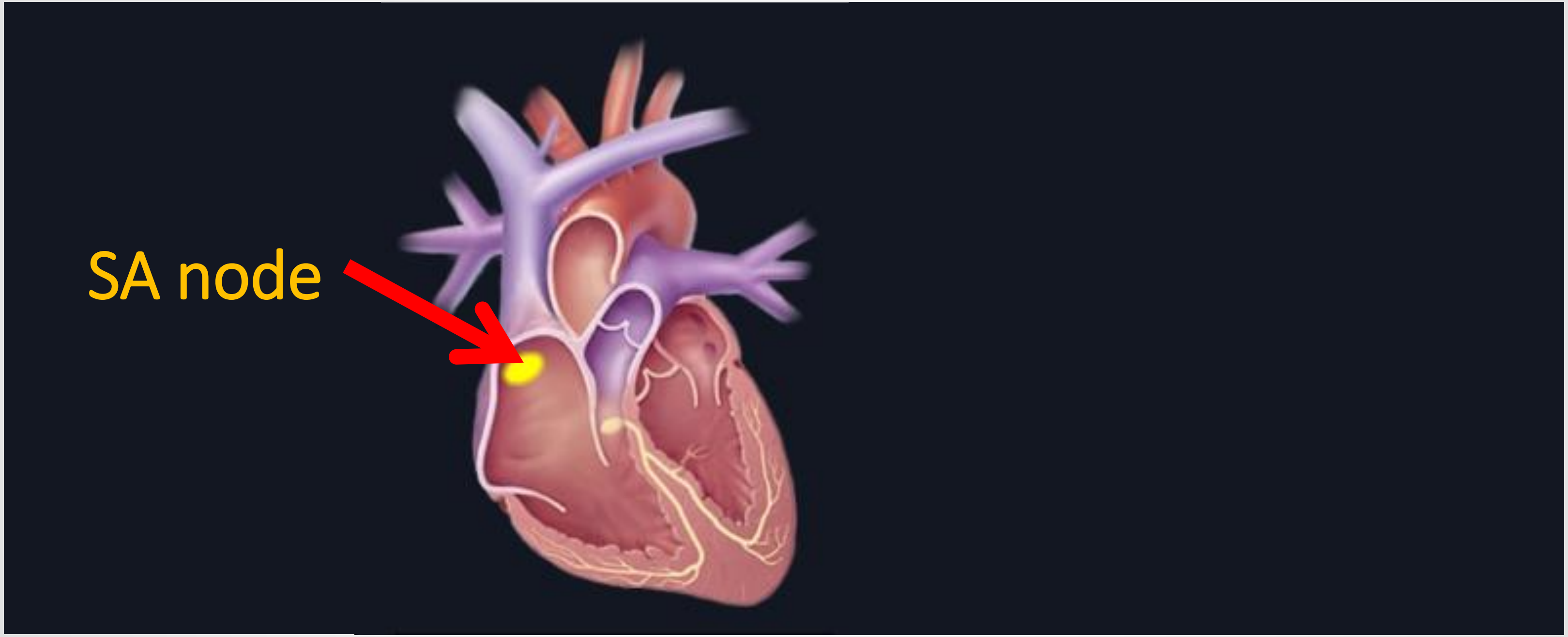
Electrophysiology of the Heart



Background

Impulse Formation

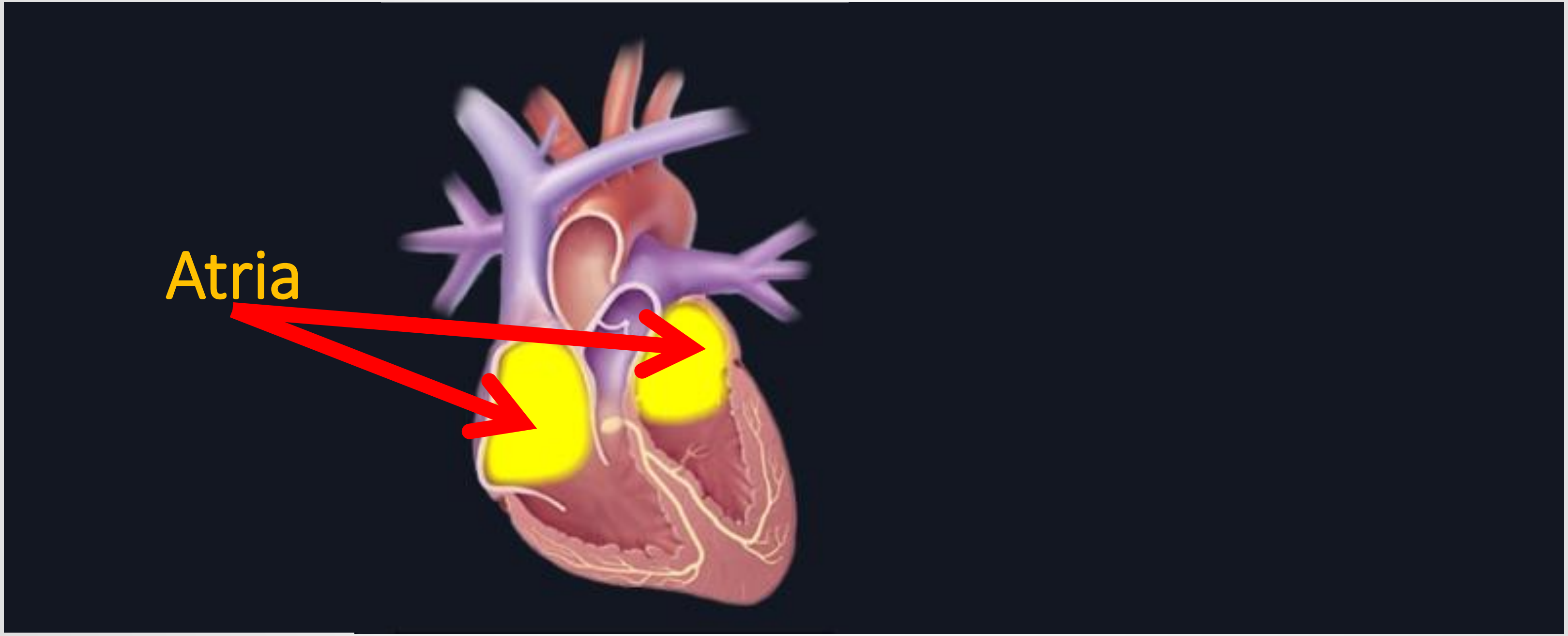
Step1. Natural pacemaker: Periodically generates electrical impulses to initialize heart beats



Background

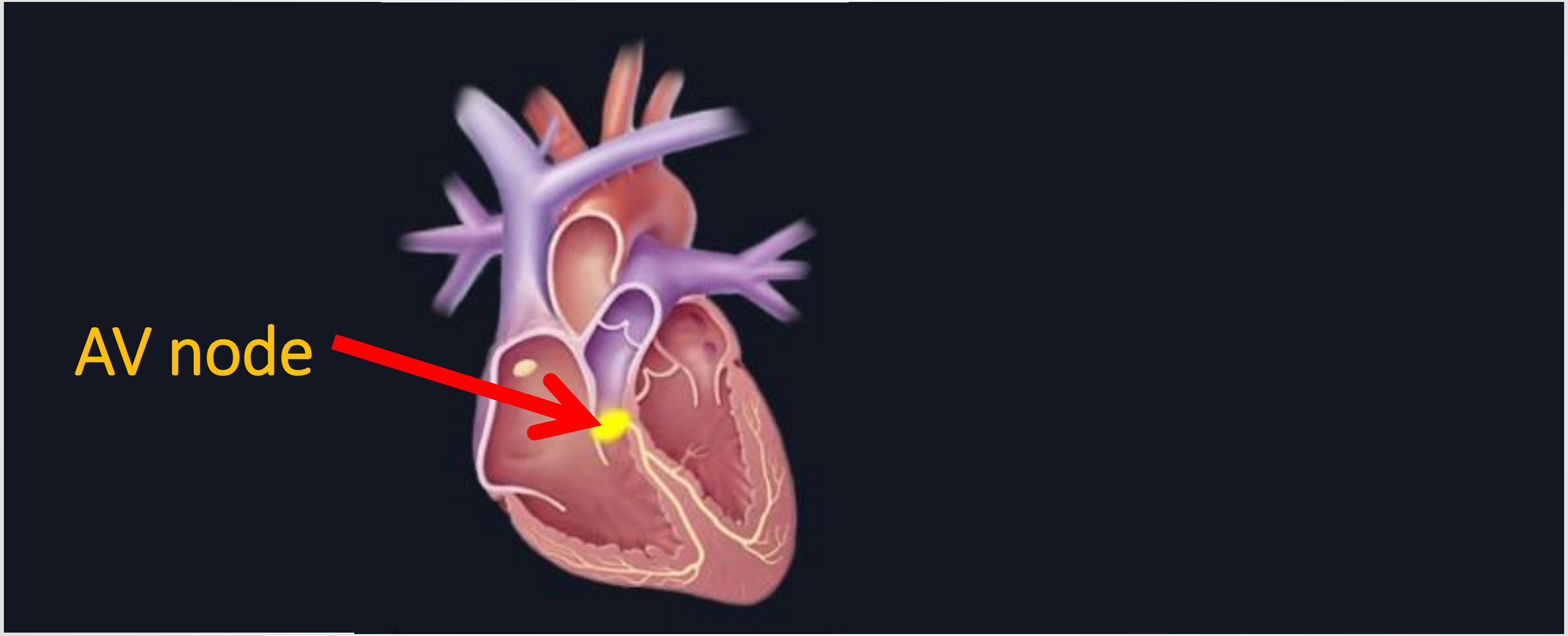
Impulse Formation

Step2. An impulse first triggers muscle contractions in the atria, pushing blood into the ventricles



Background

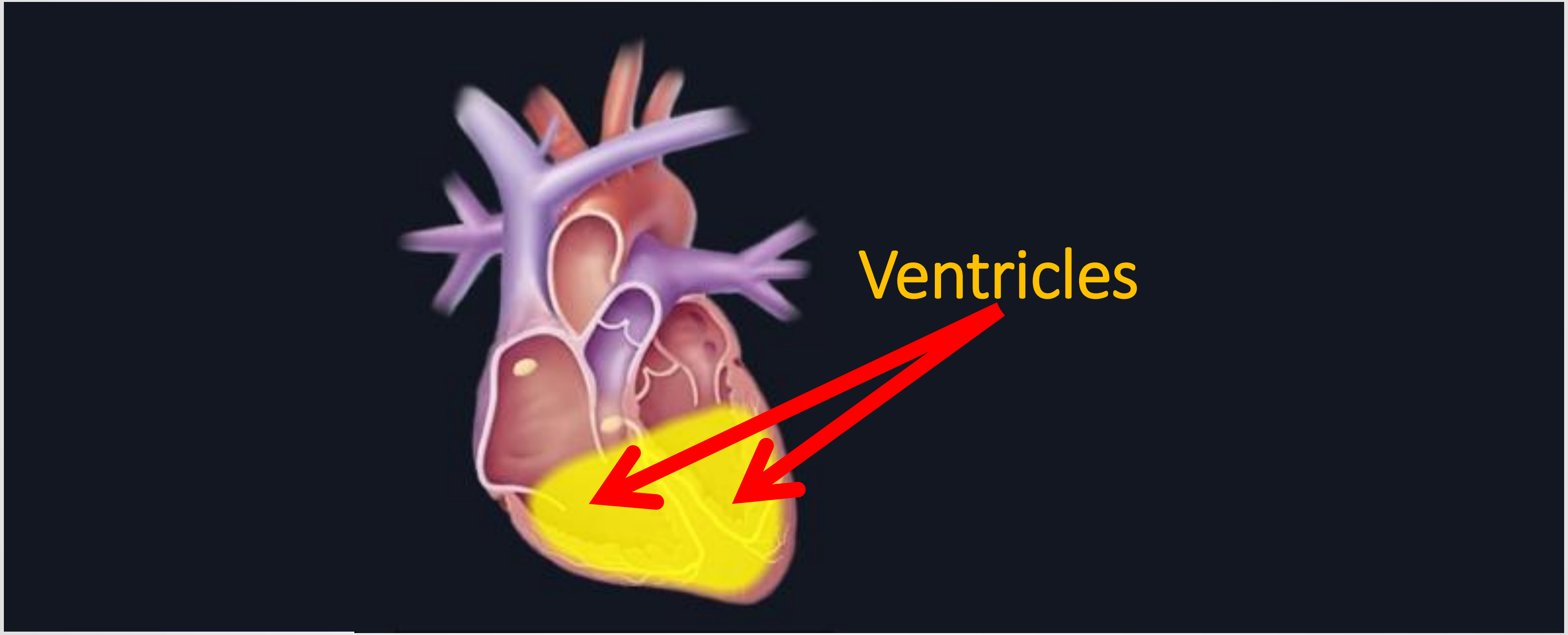
Step3. Delay at AV node allows the blood to fill fully in ventricles



Background

Impulse Formation

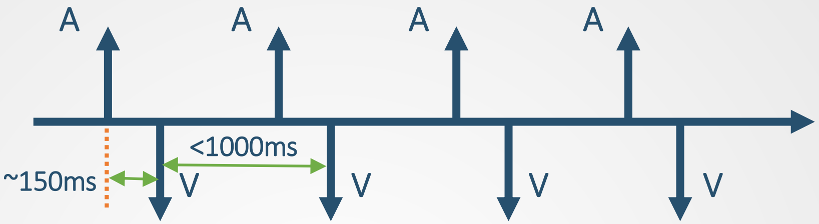
Step4. Strong muscle contractions pump blood out of the ventricles



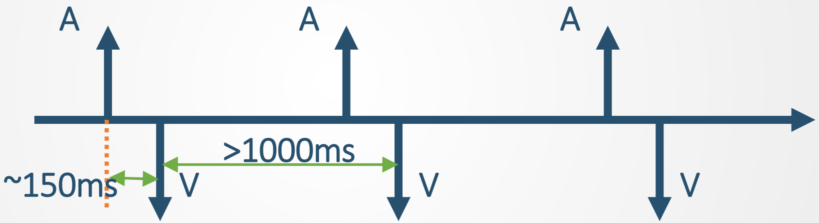
Background

Heart Conditions

Normal Sinus Rhythm

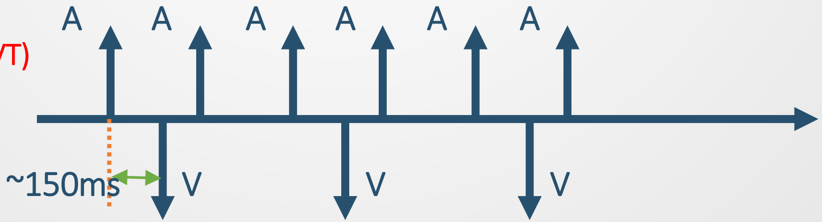


Bradycardia



Supraventricular tachycardia (SVT)

Tachycardia



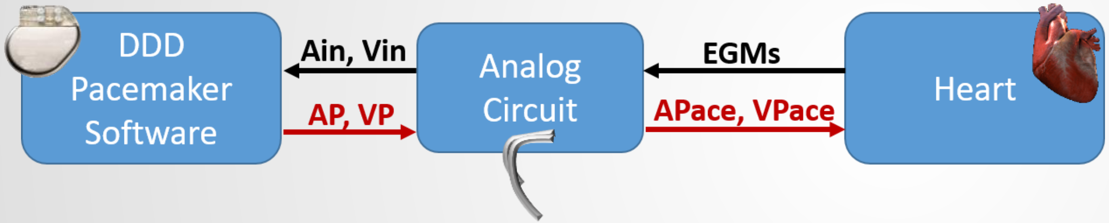
Motivation

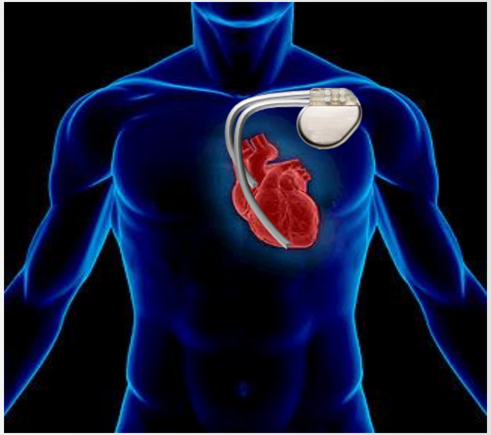
Heart contraction can be triggered by external electrical events

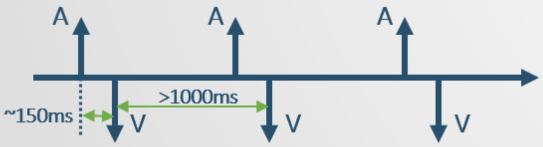
Problem

Design a dual-chamber pacemaker giving electrical pacing whenever need to:

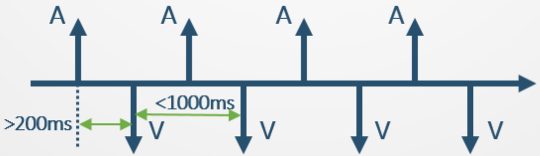
- Treat Bradycardia safely
- Not make Tachycardia worse (SVT ↛ VT)



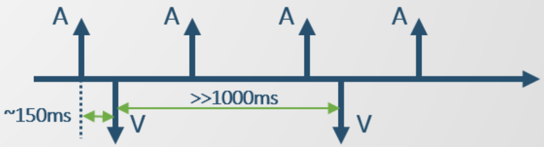




Slow generation

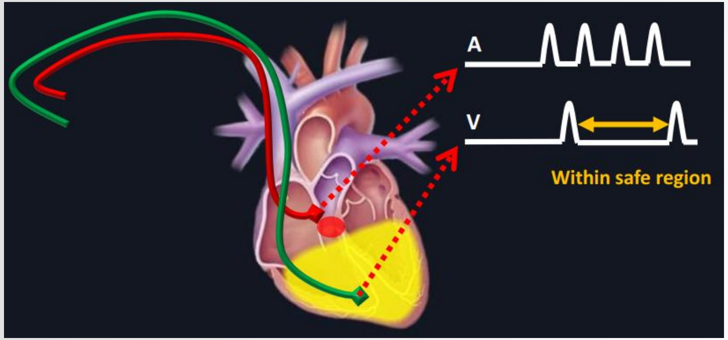


Delayed conduction

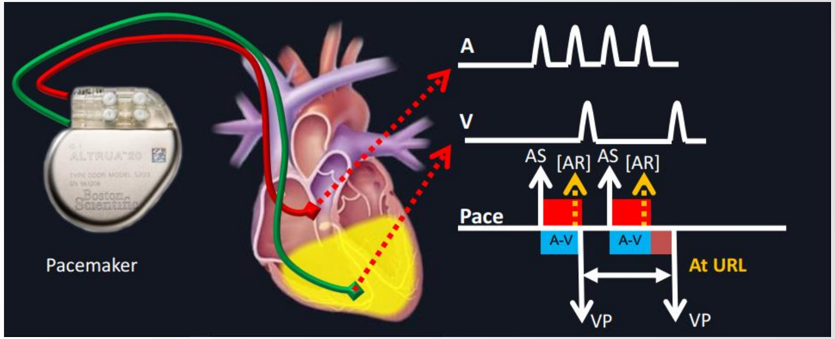


Blocked conduction

Atrial Tachycardia Response



Thanks to **ERP** of **AV** Node, ventricle beats at a **safe** rate.



ERP of AV Node **won't work**, and Pacemaker gives **VP** at rate of **URL**

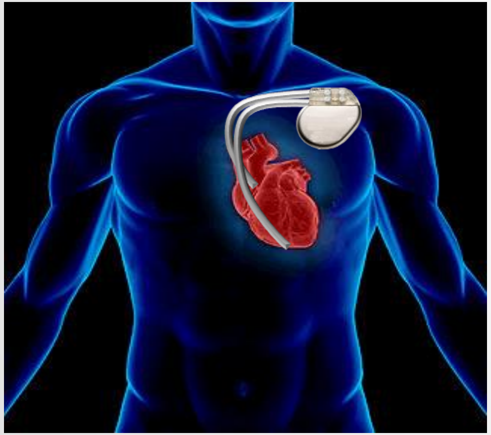
SVT → VT

Forbidden!

Problem

Design a dual-chamber pacemaker giving electrical pacing whenever need to:

- Treat Bradycardia safely
- Not make Tachycardia worse (SVT ↛ VT)



Basic requirements for any heart condition:

1. No deadlock
2. Ventricular rate no less than 60 bpm
3. Ventricular rate no more than 150bpm

Additional requirement for ATR:

4. The pacemaker should not convert SVT into VT.

Challenge

Serious Facts:

- 600,000 pacemaker recalled during 1990-2000
- 15% medical device recalls due to software errors

How to ensure the software design is safe and effective?

- How to validate the software design at an early development stage?
- How to ensure the software works as intended?
- Whether identified hazards have been mitigated?

Challenge

Serious Facts:

- 600,000 pacemaker recalled during 1990-2000
- 15% medical device recalls due to software errors

How to ensure the software design is safe and effective?

- How to validate the software design at an early development stage?
- How to ensure the software works as intended?
- Whether identified hazards have been mitigated?

Challenge

Serious Facts:

- 600,000 pacemaker recalled during 1990-2000
- 15% medical device recalls due to software errors

How to ensure the software design is safe and effective?

- How to validate the software design at an early development stage?
- How to ensure the software works as intended?
- Whether identified hazards have been mitigated?

Challenge

Serious Facts:

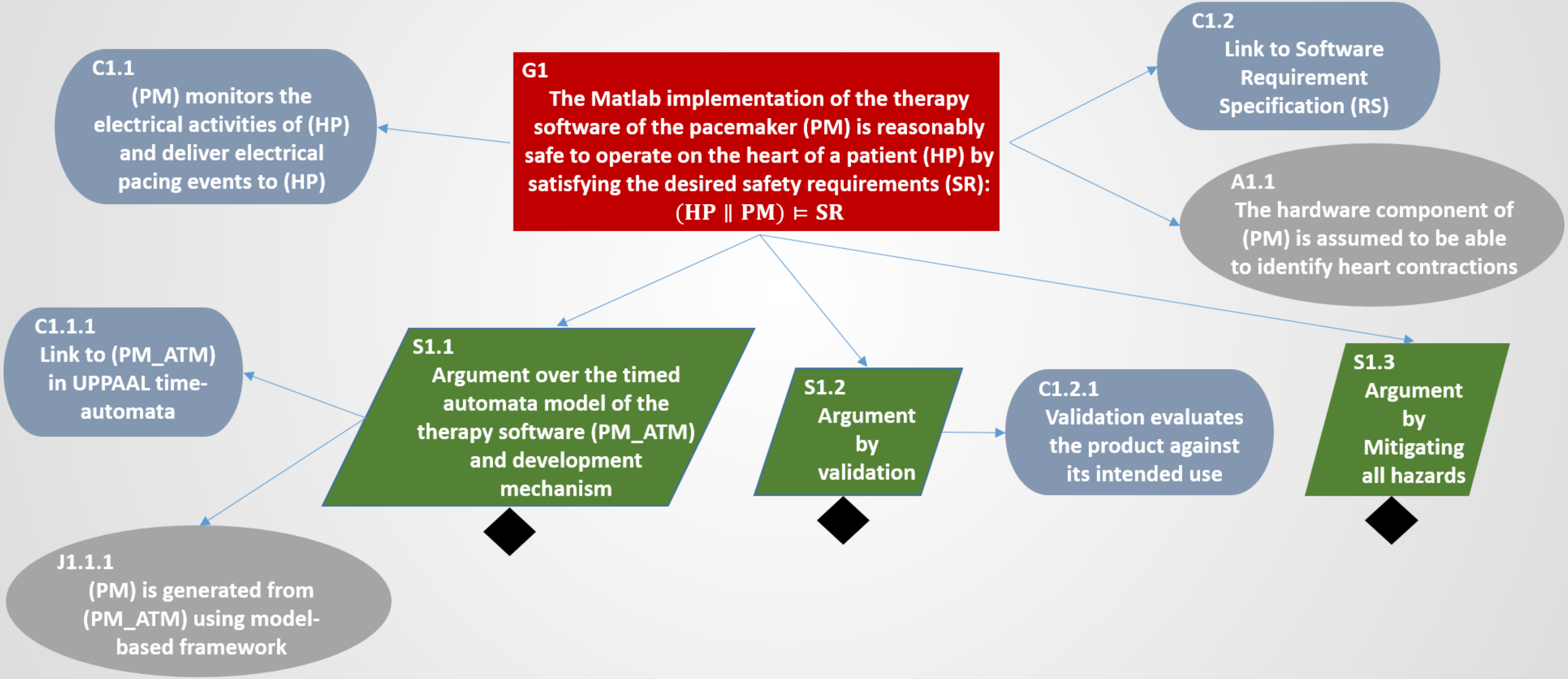
- 600,000 pacemaker recalled during 1990-2000
- 15% medical device recalls due to software errors

How to ensure the software design is safe and effective?

- How to validate the software design at an early development stage?
- How to ensure the software works as intended?
- Whether identified hazards have been mitigated?

Solution Overview

Safety Assurance: A Top Level



C1.1
(PM) monitors the electrical activities of (HP) and deliver electrical pacing events to (HP)

G1
The Matlab implementation of the therapy software of the pacemaker (PM) is reasonably safe to operate on the heart of a patient (HP) by satisfying the desired safety requirements (SR):
$(\mathbf{HP} \parallel \mathbf{PM}) \vDash \mathbf{SR}$

C1.2
Link to Software Requirement Specification (RS)

A1.1
The hardware component of (PM) is assumed to be able to identify heart contractions

C1.1.1
Link to (PM_ATM) in UPPAAL time-automata

S1.1
Argument over the timed automata model of the therapy software (PM_ATM) and development mechanism

S1.2
Argument by validation

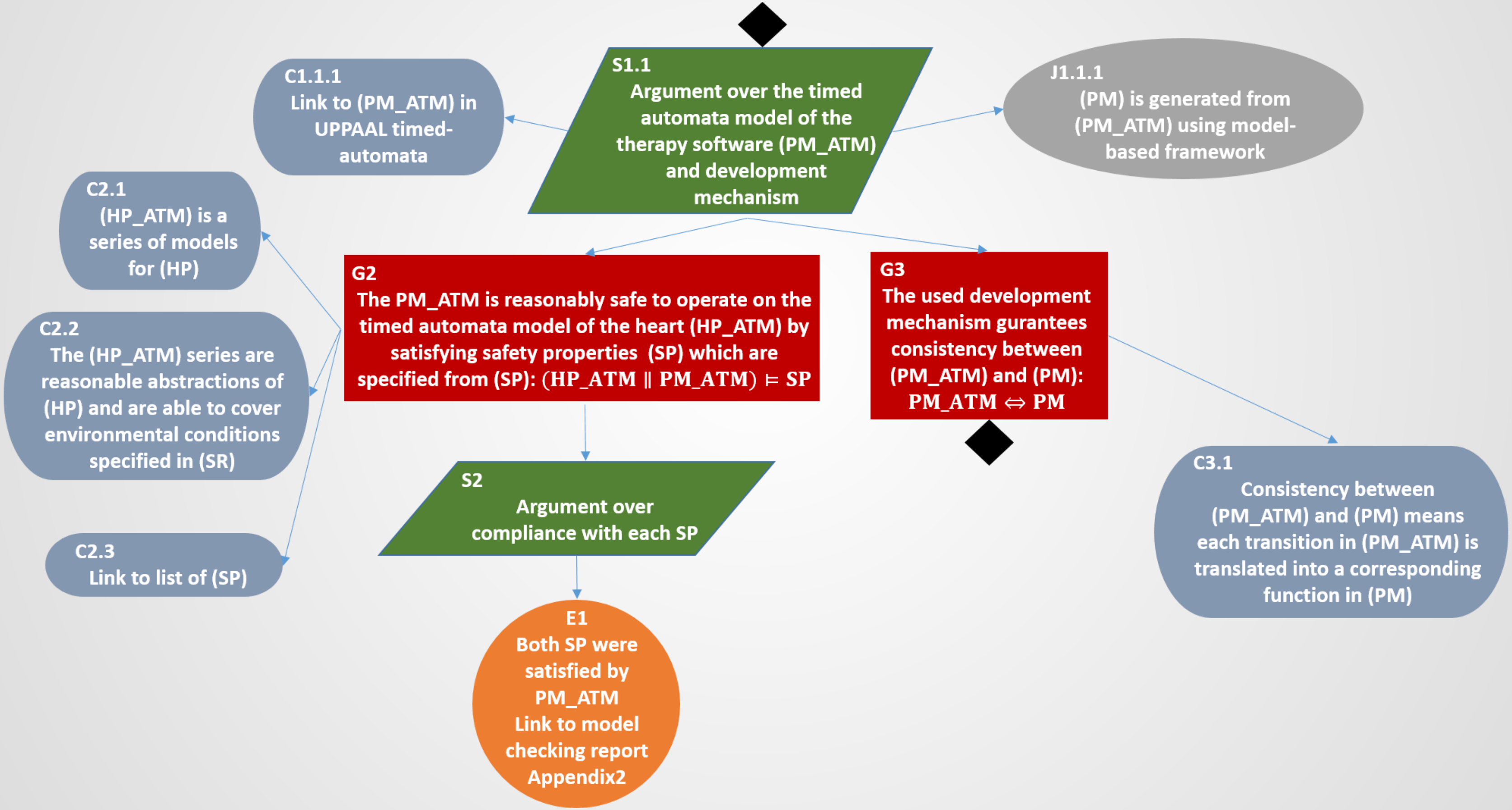
C1.2.1
Validation evaluates the product against its intended use

S1.3
Argument by Mitigating all hazards

J1.1.1
(PM) is generated from (PM_ATM) using model-based framework

Solution Overview *cont.*

Safety Assurance: A Top Level



S1.1
Argument over the timed automata model of the therapy software (PM_ATM) and development mechanism

C1.1.1
Link to (PM_ATM) in UPPAAL timed-automata

J1.1.1
(PM) is generated from (PM_ATM) using model-based framework

C2.1
(HP_ATM) is a series of models for (HP)

C2.2
The (HP_ATM) series are reasonable abstractions of (HP) and are able to cover environmental conditions specified in (SR)

C2.3
Link to list of (SP)

G2
The PM_ATM is reasonably safe to operate on the timed automata model of the heart (HP_ATM) by satisfying safety properties (SP) which are specified from (SP): $(\mathrm{HP\_ATM} \parallel \mathrm{PM\_ATM}) \models \mathrm{SP}$

G3
The used development mechanism gurantees consistency between (PM_ATM) and (PM): $\mathrm{PM\_ATM} \Leftrightarrow \mathrm{PM}$

C3.1
Consistency between (PM_ATM) and (PM) means each transition in (PM_ATM) is translated into a corresponding function in (PM)

S2
Argument over compliance with each SP

E1
Both SP were satisfied by PM_ATM
Link to model checking report Appendix2

Solution Overview *cont.*

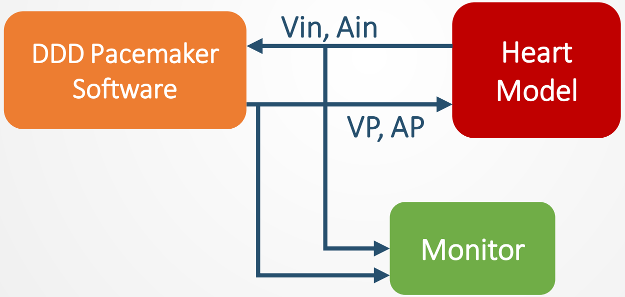
E1:Model Checking

Basic requirements for any heart condition:

1. No deadlock
2. Ventricular rate no less than 60 bpm
3. Ventricular rate no more than 150bpm

Additional requirement for ATR:

4. The pacemaker should not convert SVT into VT.

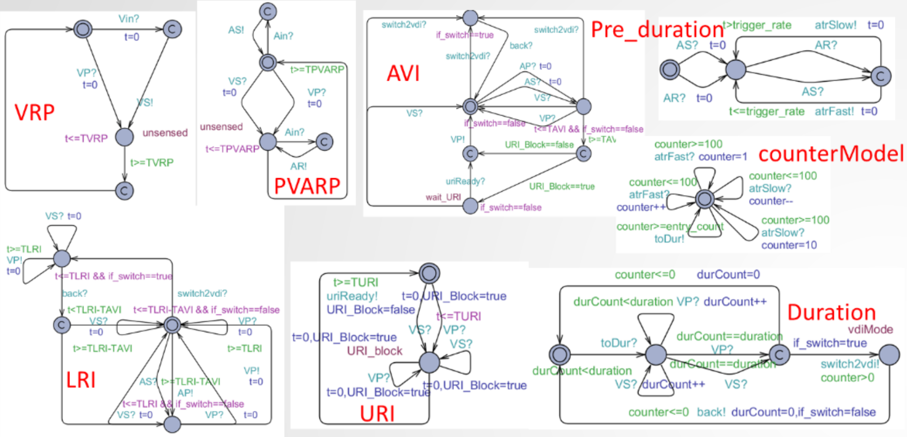


1. A[] (not deadlock)
2. A[] (PLRL.two_v imply PLRL.t<=TLRI)
3. A[] (PURL.interval imply PURL.t==TURI)

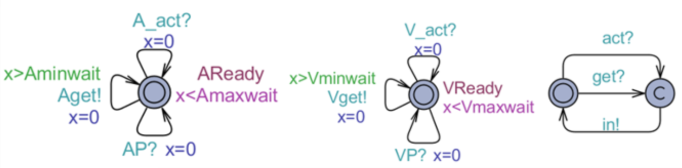
4. A[] (not PPersist.err)

Solution Overview *cont.*

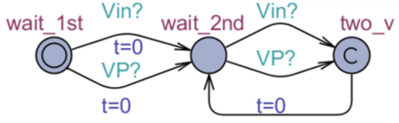
E1:Model Checking



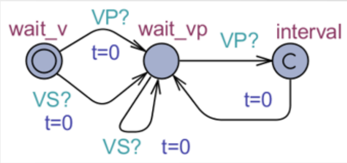
DDD Pacemaker



Heart Model

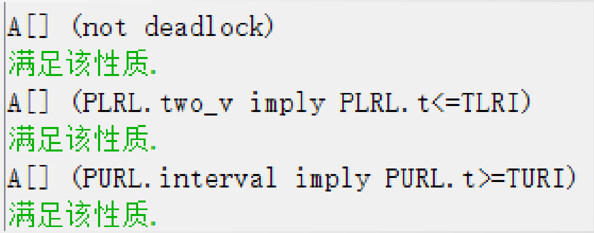


Monitor for Requirement 2



Monitor for Requirement 3

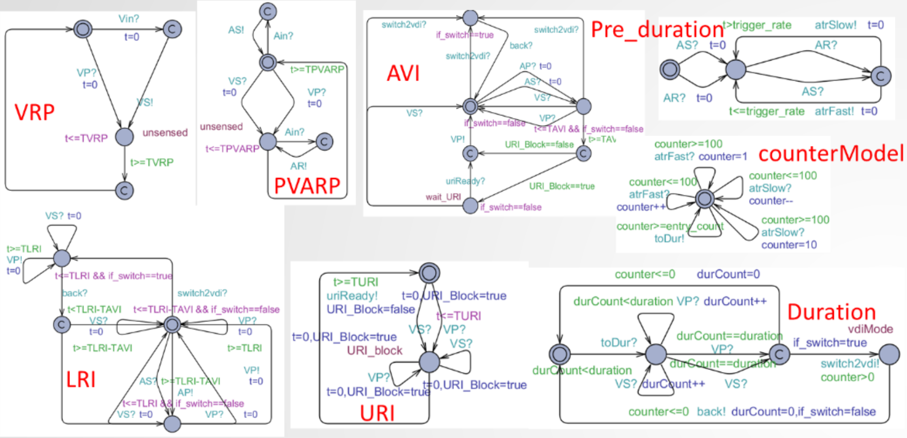
basic requirements for all the possible heart conditions



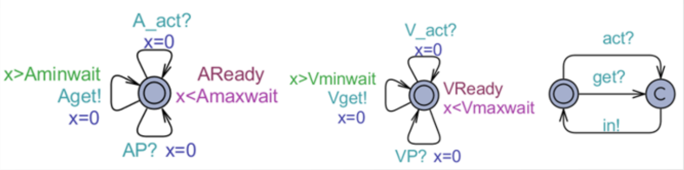
```
A[] (not deadlock)
满足该性质.
A[] (PLRL.two_v imply PLRL.t<=TLRI)
满足该性质.
A[] (PURL.interval imply PURL.t>=TURI)
满足该性质.
```

Solution Overview *cont.*

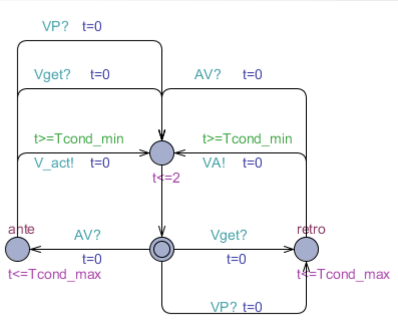
E1:Model Checking



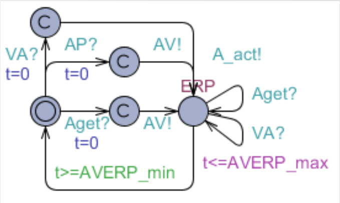
DDD Pacemaker



Heart Model

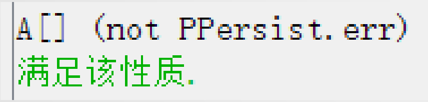


Conduction Path Model

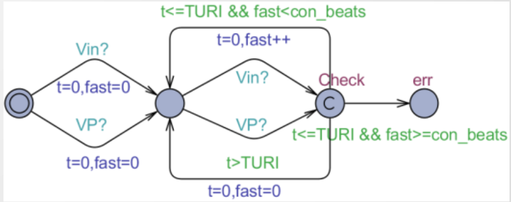


AV Node Model

specific requirement for ATR



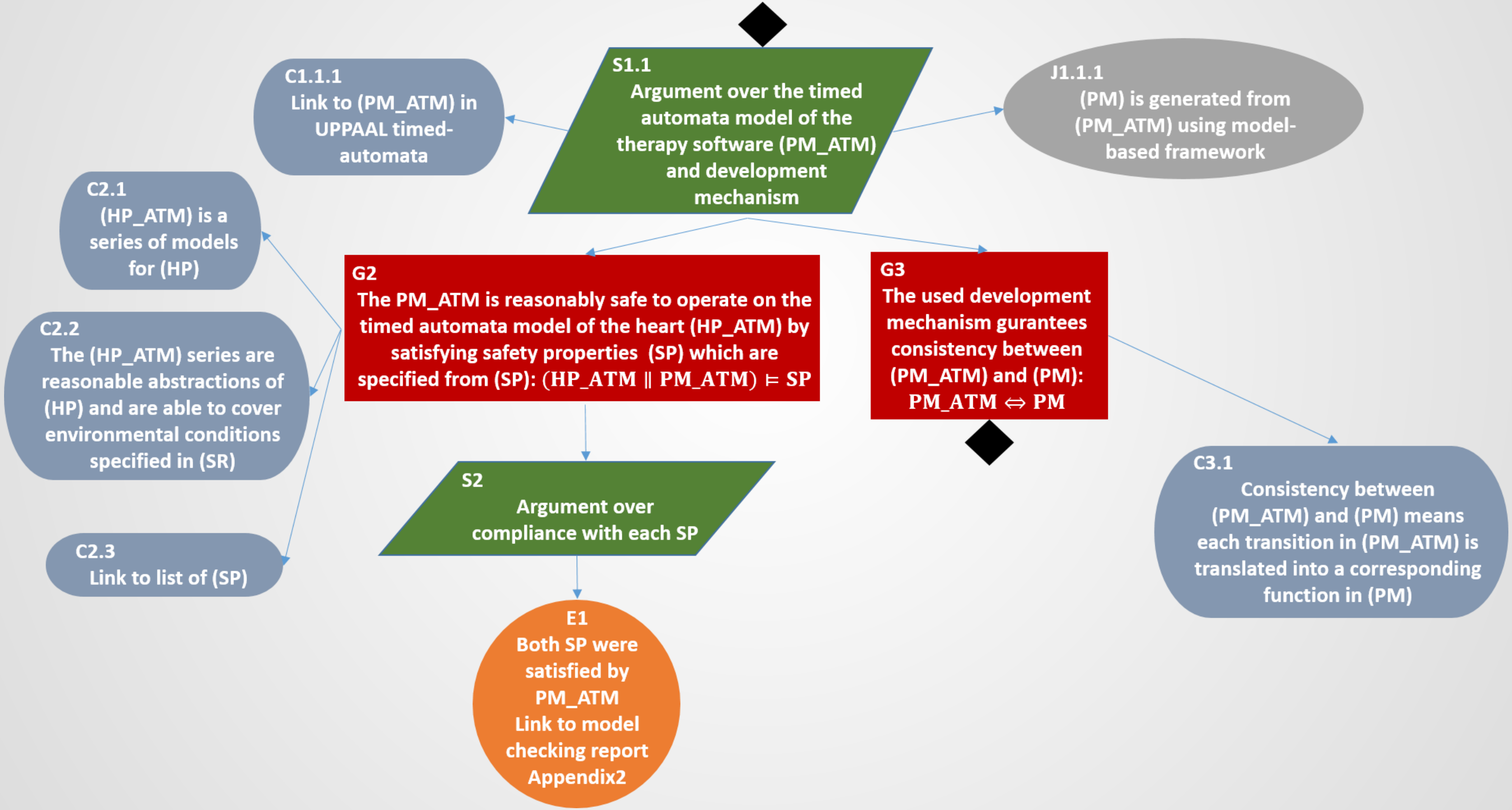
```
A[] (not PPersist.err)
满足该性质.
```



Monitor for Requirement 4

Solution Overview *cont.*

Safety Assurance: A Top Level



S1.1
Argument over the timed automata model of the therapy software (PM_ATM) and development mechanism

C1.1.1
Link to (PM_ATM) in UPPAAL timed-automata

J1.1.1
(PM) is generated from (PM_ATM) using model-based framework

G2
The PM_ATM is reasonably safe to operate on the timed automata model of the heart (HP_ATM) by satisfying safety properties (SP) which are specified from (SP): $(\mathrm{HP\_ATM} \parallel \mathrm{PM\_ATM}) \vDash \mathrm{SP}$

C2.1
(HP_ATM) is a series of models for (HP)

C2.2
The (HP_ATM) series are reasonable abstractions of (HP) and are able to cover environmental conditions specified in (SR)

C2.3
Link to list of (SP)

S2
Argument over compliance with each SP

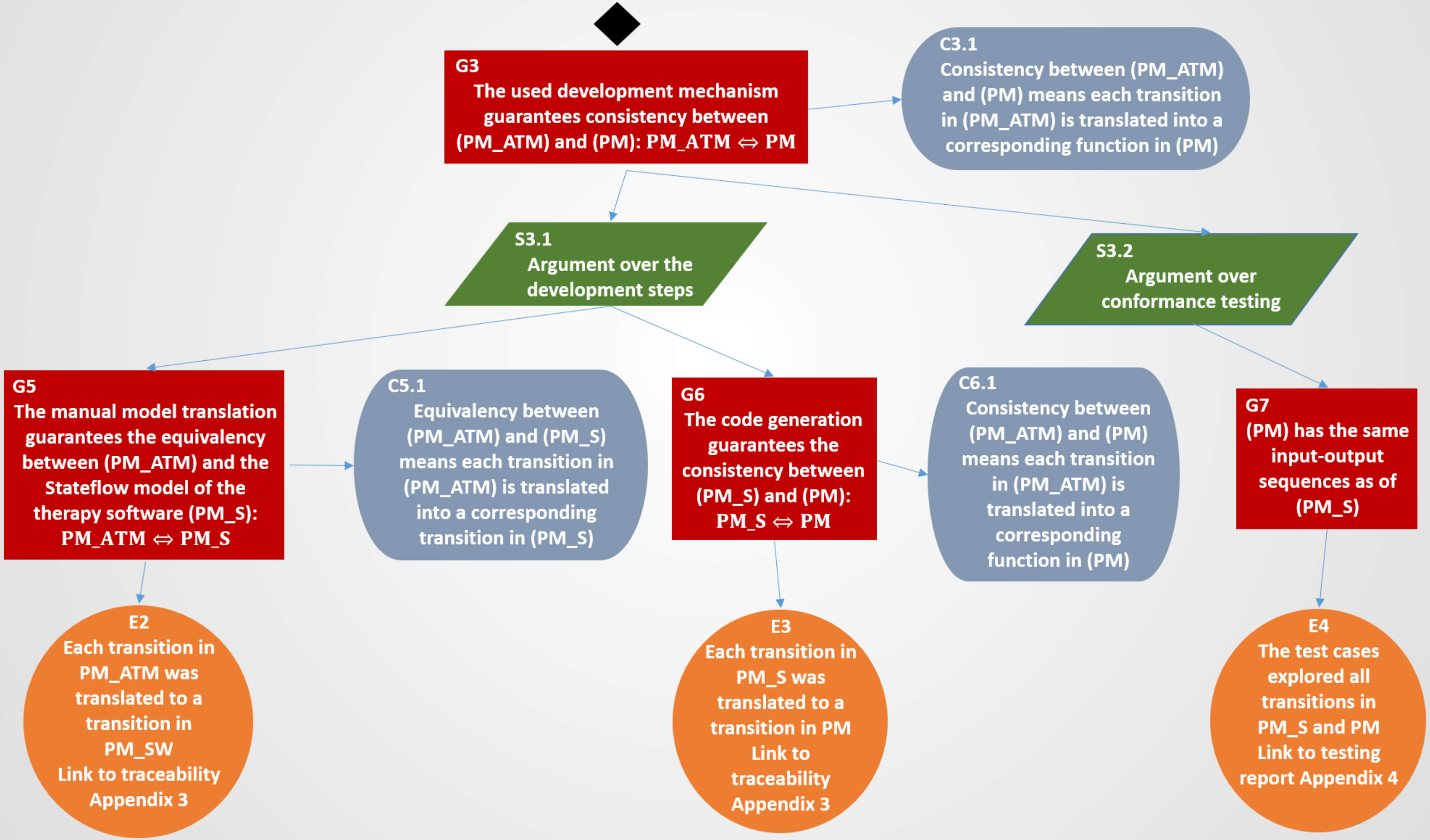
E1
Both SP were satisfied by PM_ATM
Link to model checking report Appendix2

G3
The used development mechanism gurantees consistency between (PM_ATM) and (PM): $\mathrm{PM\_ATM} \Leftrightarrow \mathrm{PM}$

C3.1
Consistency between (PM_ATM) and (PM) means each transition in (PM_ATM) is translated into a corresponding function in (PM)

Solution Overview *cont.*

Safety Assurance: A Top Level



G3
The used development mechanism guarantees consistency between (PM_ATM) and (PM): PM_ATM ⇔ PM

C3.1
Consistency between (PM_ATM) and (PM) means each transition in (PM_ATM) is translated into a corresponding function in (PM)

S3.1
Argument over the development steps

S3.2
Argument over conformance testing

G5
The manual model translation guarantees the equivalency between (PM_ATM) and the Stateflow model of the therapy software (PM_S): PM_ATM ⇔ PM_S

C5.1
Equivalency between (PM_ATM) and (PM_S) means each transition in (PM_ATM) is translated into a corresponding transition in (PM_S)

G6
The code generation guarantees the consistency between (PM_S) and (PM): PM_S ⇔ PM

C6.1
Consistency between (PM_ATM) and (PM) means each transition in (PM_ATM) is translated into a corresponding function in (PM)

G7
(PM) has the same input-output sequences as of (PM_S)

E2
Each transition in PM_ATM was translated to a transition in PM_SW
Link to traceability Appendix 3

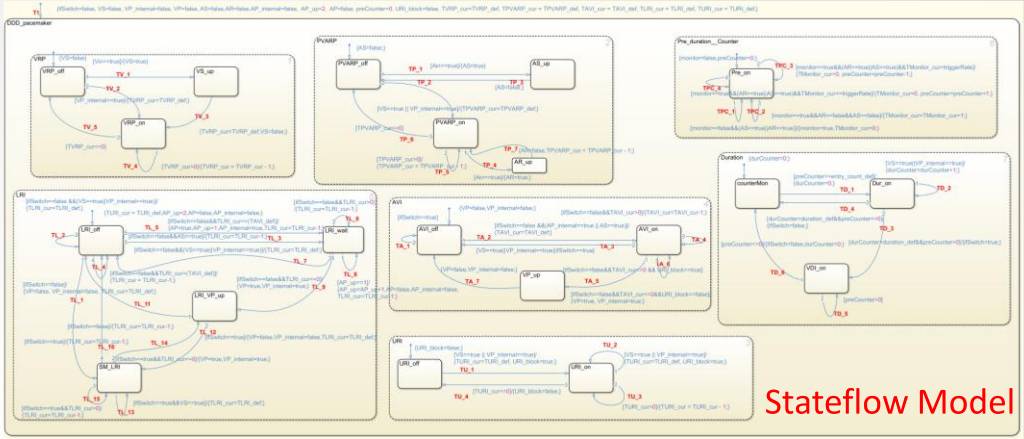
E3
Each transition in PM_S was translated to a transition in PM
Link to traceability Appendix 3

E4
The test cases explored all transitions in PM_S and PM
Link to testing report Appendix 4

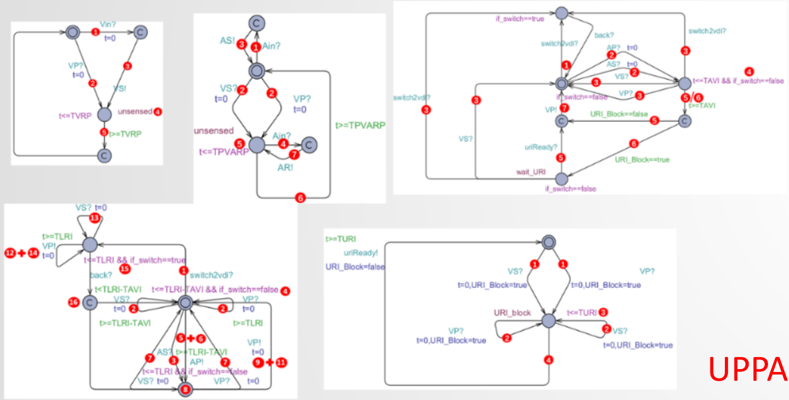
Solution Overview ***cont.***

E2:UPPAAL to Stateflow Model

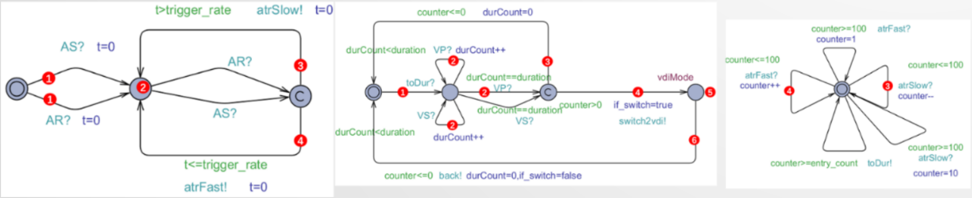
- Clocks were translated into counters
- Events were set to True for 1 time cycle



Stateflow Model

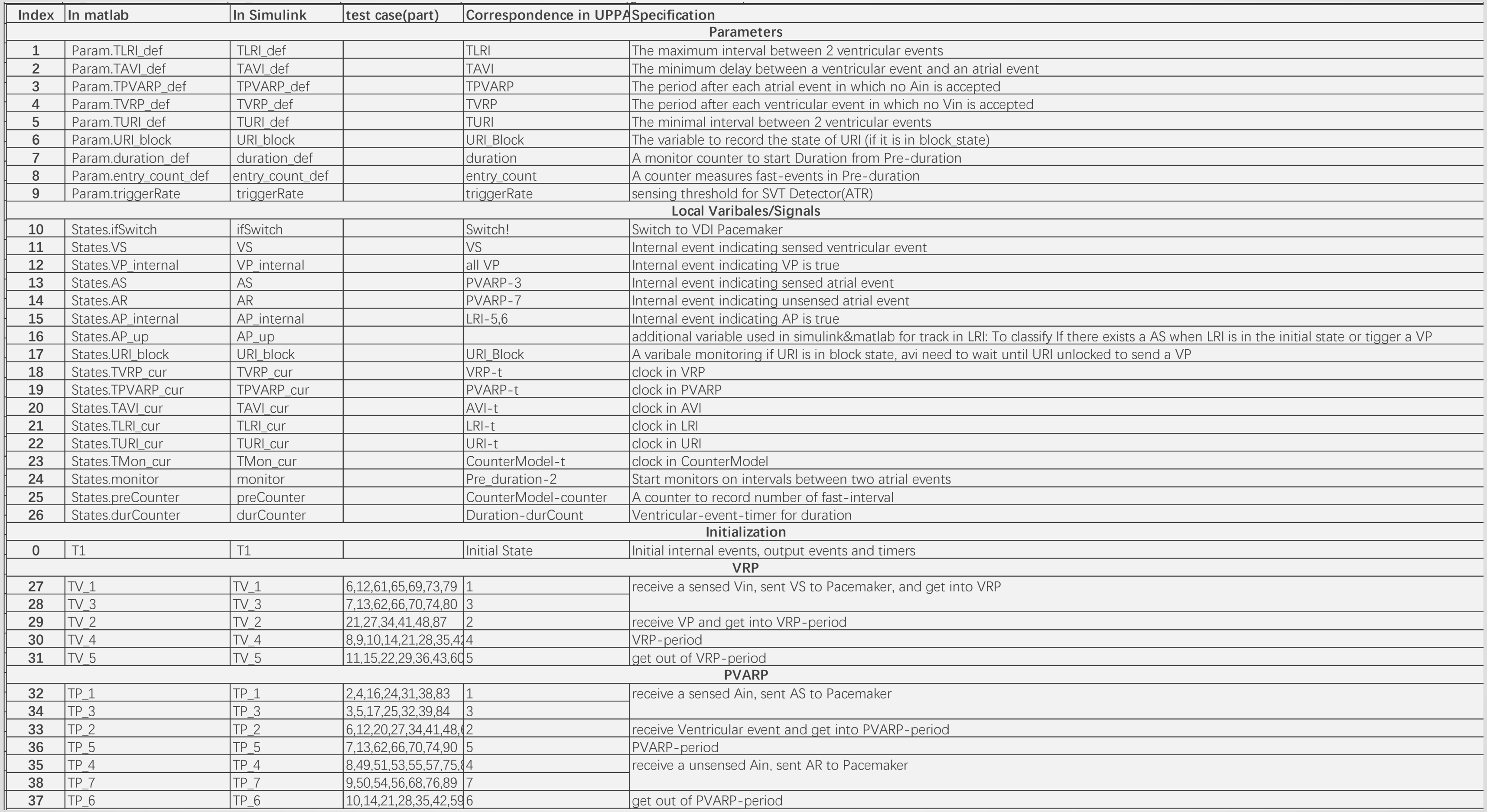


UPPAAL Model



Solution Overview *cont.*

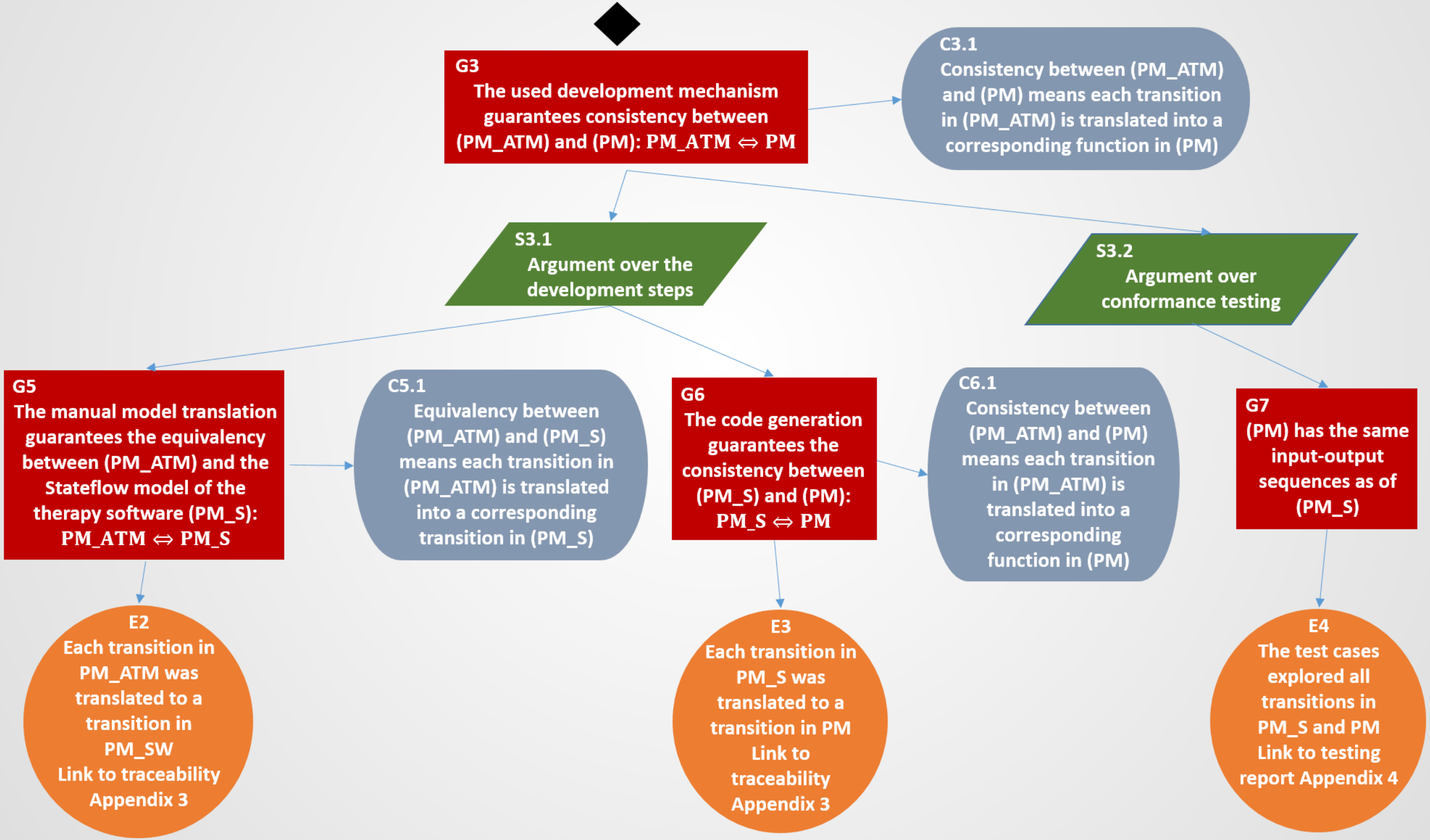
E2:UPPAAL to Stateflow Model



Index	In matlab	In Simulink	test case(part)	Correspondence in UPPA	Specification
Parameters					
1	Param.TLRI_def	TLRI_def		TLRI	The maximum interval between 2 ventricular events
2	Param.TAVI_def	TAVI_def		TAVI	The minimum delay between a ventricular event and an atrial event
3	Param.TPVARP_def	TPVARP_def		TPVARP	The period after each atrial event in which no Ain is accepted
4	Param.TVRP_def	TVRP_def		TVRP	The period after each ventricular event in which no Vin is accepted
5	Param.TURI_def	TURI_def		TURI	The minimal interval between 2 ventricular events
6	Param.URI_block	URI_block		URI_Block	The variable to record the state of URI (if it is in block_state)
7	Param.duration_def	duration_def		duration	A monitor counter to start Duration from Pre-duration
8	Param.entry_count_def	entry_count_def		entry_count	A counter measures fast-events in Pre-duration
9	Param.triggerRate	triggerRate		triggerRate	sensing threshold for SVT Detector(ATR)
Local Varibales/Signals					
10	States.ifSwitch	ifSwitch		Switch!	Switch to VDI Pacemaker
11	States.VS	VS		VS	Internal event indicating sensed ventricular event
12	States.VP_internal	VP_internal		all VP	Internal event indicating VP is true
13	States.AS	AS		PVARP-3	Internal event indicating sensed atrial event
14	States.AR	AR		PVARP-7	Internal event indicating unsensed atrial event
15	States.AP_internal	AP_internal		LRI-5,6	Internal event indicating AP is true
16	States.AP_up	AP_up			additional variable used in simulink&matlab for track in LRI: To classify If there exists a AS when LRI is in the initial state or tigger a VP
17	States.URI_block	URI_block		URI_Block	A varibale monitoring if URI is in block state, avi need to wait until URI unlocked to send a VP
18	States.TVRP_cur	TVRP_cur		VRP-t	clock in VRP
19	States.TPVARP_cur	TPVARP_cur		PVARP-t	clock in PVARP
20	States.TAVI_cur	TAVI_cur		AVI-t	clock in AVI
21	States.TLRI_cur	TLRI_cur		LRI-t	clock in LRI
22	States.TURI_cur	TURI_cur		URI-t	clock in URI
23	States.TMon_cur	TMon_cur		CounterModel-t	clock in CounterModel
24	States.monitor	monitor		Pre_duration-2	Start monitors on intervals between two atrial events
25	States.preCounter	preCounter		CounterModel-counter	A counter to record number of fast-interval
26	States.durCounter	durCounter		Duration-durCount	Ventricular-event-timer for duration
Initialization					
0	T1	T1		Initial State	Initial internal events, output events and timers
VRP					
27	TV_1	TV_1	6,12,61,65,69,73,79	1	receive a sensed Vin, sent VS to Pacemaker, and get into VRP
28	TV_3	TV_3	7,13,62,66,70,74,80	3	
29	TV_2	TV_2	21,27,34,41,48,87	2	receive VP and get into VRP-period
30	TV_4	TV_4	8,9,10,14,21,28,35,4	4	VRP-period
31	TV_5	TV_5	11,15,22,29,36,43,60	5	get out of VRP-period
PVARP					
32	TP_1	TP_1	2,4,16,24,31,38,83	1	receive a sensed Ain, sent AS to Pacemaker
34	TP_3	TP_3	3,5,17,25,32,39,84	3	
33	TP_2	TP_2	6,12,20,27,34,41,48,(	2	receive Ventricular event and get into PVARP-period
36	TP_5	TP_5	7,13,62,66,70,74,90	5	PVARP-period
35	TP_4	TP_4	8,49,51,53,55,57,75,8	4	receive a unsensed Ain, sent AR to Pacemaker
38	TP_7	TP_7	9,50,54,56,68,76,89	7	
37	TP_6	TP_6	10,14,21,28,35,42,59	6	get out of PVARP-period

Solution Overview *cont.*

Safety Assurance: A Top Level



G3
The used development mechanism guarantees consistency between (PM_ATM) and (PM): PM_ATM ⇔ PM

C3.1
Consistency between (PM_ATM) and (PM) means each transition in (PM_ATM) is translated into a corresponding function in (PM)

S3.1
Argument over the development steps

S3.2
Argument over conformance testing

G5
The manual model translation guarantees the equivalency between (PM_ATM) and the Stateflow model of the therapy software (PM_S): PM_ATM ⇔ PM_S

C5.1
Equivalency between (PM_ATM) and (PM_S) means each transition in (PM_ATM) is translated into a corresponding transition in (PM_S)

G6
The code generation guarantees the consistency between (PM_S) and (PM): PM_S ⇔ PM

C6.1
Consistency between (PM_ATM) and (PM) means each transition in (PM_ATM) is translated into a corresponding function in (PM)

G7
(PM) has the same input-output sequences as of (PM_S)

E2
Each transition in PM_ATM was translated to a transition in PM_SW
Link to traceability Appendix 3

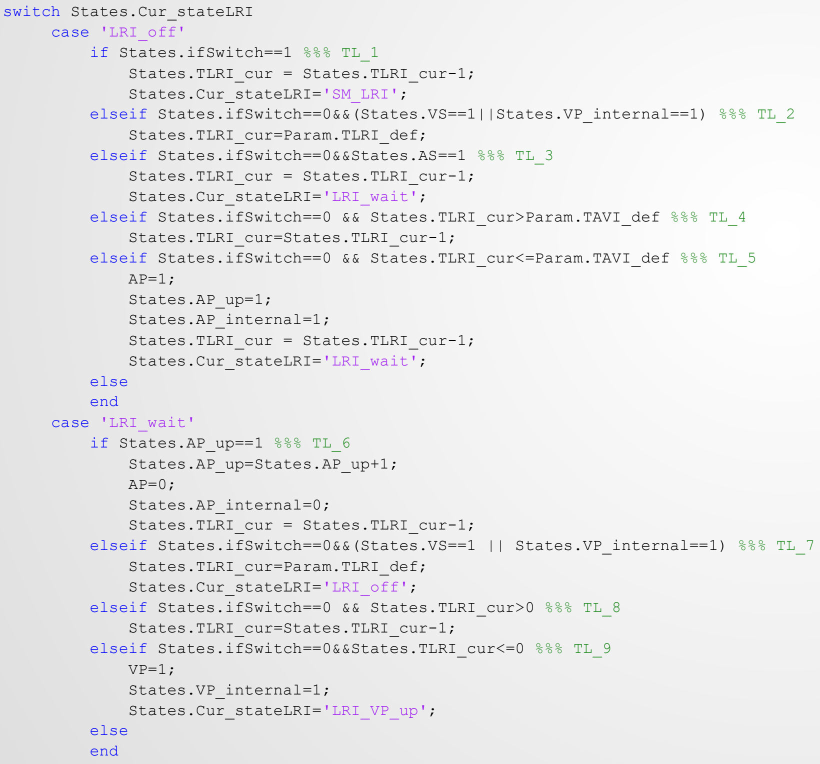
E3
Each transition in PM_S was translated to a transition in PM
Link to traceability Appendix 3

E4
The test cases explored all transitions in PM_S and PM
Link to testing report Appendix 4

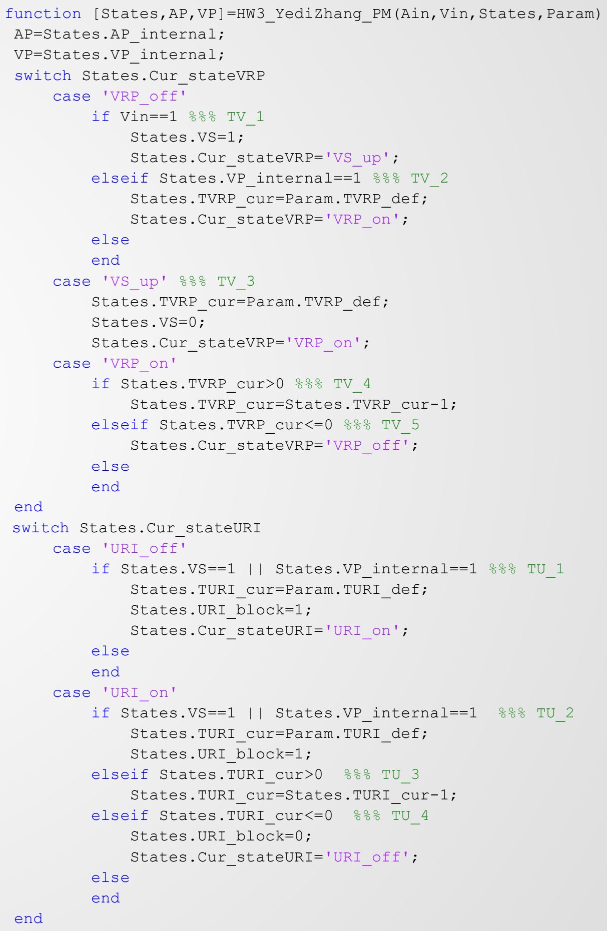
Solution Overview *cont.*

E3:Stateflow Model to Matlab Code

- 1-to-1 translation from the Stateflow model
- Function called every 1ms



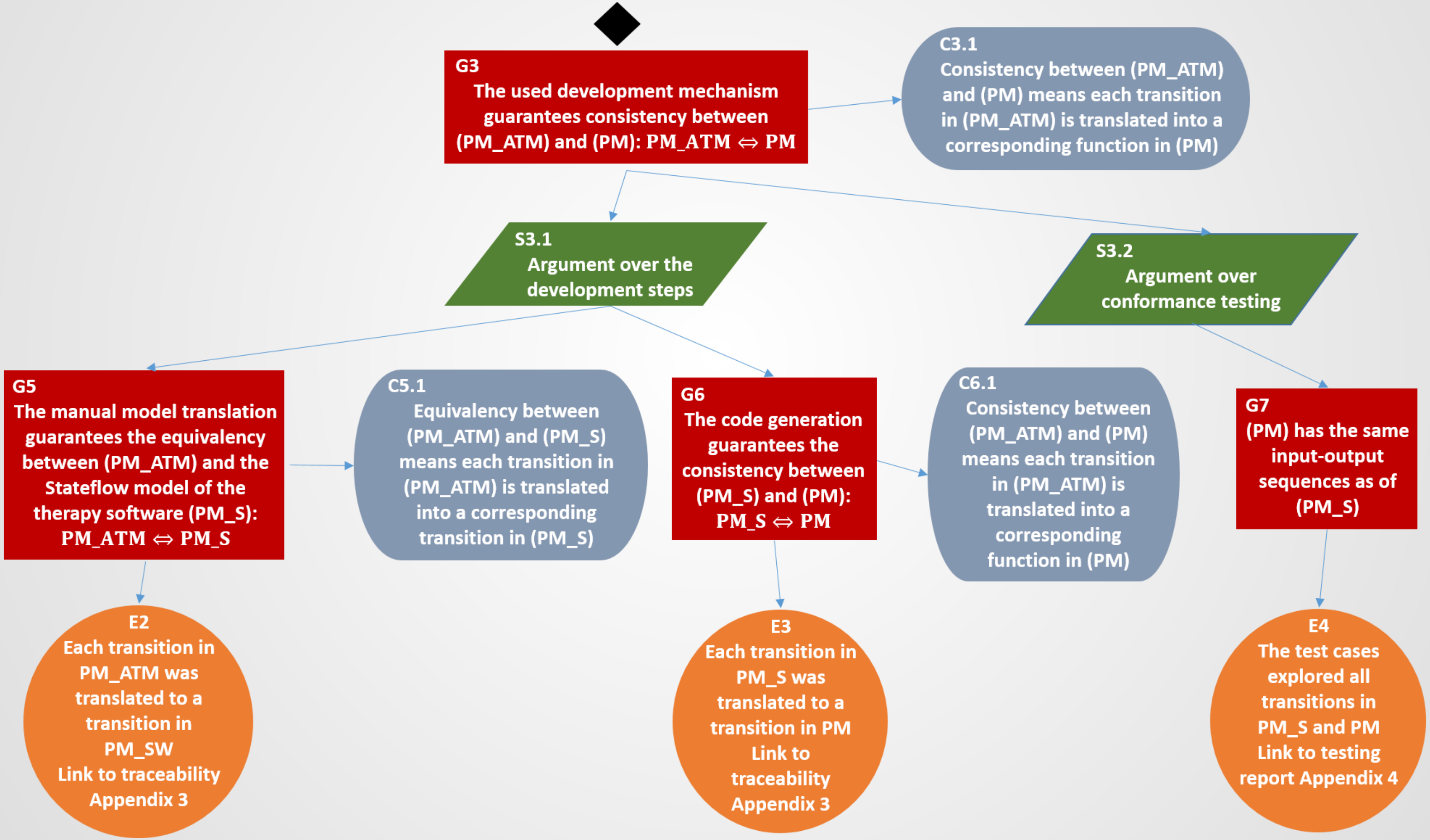
```
switch States.Cur_stateLRI
    case 'LRI_off'
        if States.ifSwitch==1 %%% TL_1
            States.TLRI_cur = States.TLRI_cur-1;
            States.Cur_stateLRI='SM_LRI';
        elseif States.ifSwitch==0&&(States.VS==1||States.VP_internal==1) %%% TL_2
            States.TLRI_cur=Param.TLRI_def;
        elseif States.ifSwitch==0&&States.AS==1 %%% TL_3
            States.TLRI_cur = States.TLRI_cur-1;
            States.Cur_stateLRI='LRI_wait';
        elseif States.ifSwitch==0 && States.TLRI_cur>Param.TAVI_def %%% TL_4
            States.TLRI_cur=States.TLRI_cur-1;
        elseif States.ifSwitch==0 && States.TLRI_cur<=Param.TAVI_def %%% TL_5
            AP=1;
            States.AP_up=1;
            States.AP_internal=1;
            States.TLRI_cur = States.TLRI_cur-1;
            States.Cur_stateLRI='LRI_wait';
        else
        end
    case 'LRI_wait'
        if States.AP_up==1 %%% TL_6
            States.AP_up=States.AP_up+1;
            AP=0;
            States.AP_internal=0;
            States.TLRI_cur = States.TLRI_cur-1;
        elseif States.ifSwitch==0&&(States.VS==1 || States.VP_internal==1) %%% TL_7
            States.TLRI_cur=Param.TLRI_def;
            States.Cur_stateLRI='LRI_off';
        elseif States.ifSwitch==0 && States.TLRI_cur>0 %%% TL_8
            States.TLRI_cur=States.TLRI_cur-1;
        elseif States.ifSwitch==0&&States.TLRI_cur<=0 %%% TL_9
            VP=1;
            States.VP_internal=1;
            States.Cur_stateLRI='LRI_VP_up';
        else
        end
```



```
function [States,AP,VP]=HW3_YediZhang_PM(Ain,Vin,States,Param)
 AP=States.AP_internal;
 VP=States.VP_internal;
 switch States.Cur_stateVRP
     case 'VRP_off'
         if Vin==1 %%% TV_1
             States.VS=1;
             States.Cur_stateVRP='VS_up';
         elseif States.VP_internal==1 %%% TV_2
             States.TVRP_cur=Param.TVRP_def;
             States.Cur_stateVRP='VRP_on';
         else
         end
     case 'VS_up' %%% TV_3
         States.TVRP_cur=Param.TVRP_def;
         States.VS=0;
         States.Cur_stateVRP='VRP_on';
     case 'VRP_on'
         if States.TVRP_cur>0 %%% TV_4
             States.TVRP_cur=States.TVRP_cur-1;
         elseif States.TVRP_cur<=0 %%% TV_5
             States.Cur_stateVRP='VRP_off';
         else
         end
 end
 switch States.Cur_stateURI
     case 'URI_off'
         if States.VS==1 || States.VP_internal==1 %%% TU_1
             States.TURI_cur=Param.TURI_def;
             States.URI_block=1;
             States.Cur_stateURI='URI_on';
         else
         end
     case 'URI_on'
         if States.VS==1 || States.VP_internal==1  %%% TU_2
             States.TURI_cur=Param.TURI_def;
             States.URI_block=1;
         elseif States.TURI_cur>0  %%% TU_3
             States.TURI_cur=States.TURI_cur-1;
         elseif States.TURI_cur<=0  %%% TU_4
             States.URI_block=0;
             States.Cur_stateURI='URI_off';
         else
         end
 end
```

Solution Overview *cont.*

Safety Assurance: A Top Level



G3
The used development mechanism guarantees consistency between (PM_ATM) and (PM): PM_ATM ⇔ PM

C3.1
Consistency between (PM_ATM) and (PM) means each transition in (PM_ATM) is translated into a corresponding function in (PM)

S3.1
Argument over the development steps

S3.2
Argument over conformance testing

G5
The manual model translation guarantees the equivalency between (PM_ATM) and the Stateflow model of the therapy software (PM_S): PM_ATM ⇔ PM_S

C5.1
Equivalency between (PM_ATM) and (PM_S) means each transition in (PM_ATM) is translated into a corresponding transition in (PM_S)

G6
The code generation guarantees the consistency between (PM_S) and (PM): PM_S ⇔ PM

C6.1
Consistency between (PM_ATM) and (PM) means each transition in (PM_ATM) is translated into a corresponding function in (PM)

G7
(PM) has the same input-output sequences as of (PM_S)

E2
Each transition in PM_ATM was translated to a transition in PM_SW
Link to traceability Appendix 3

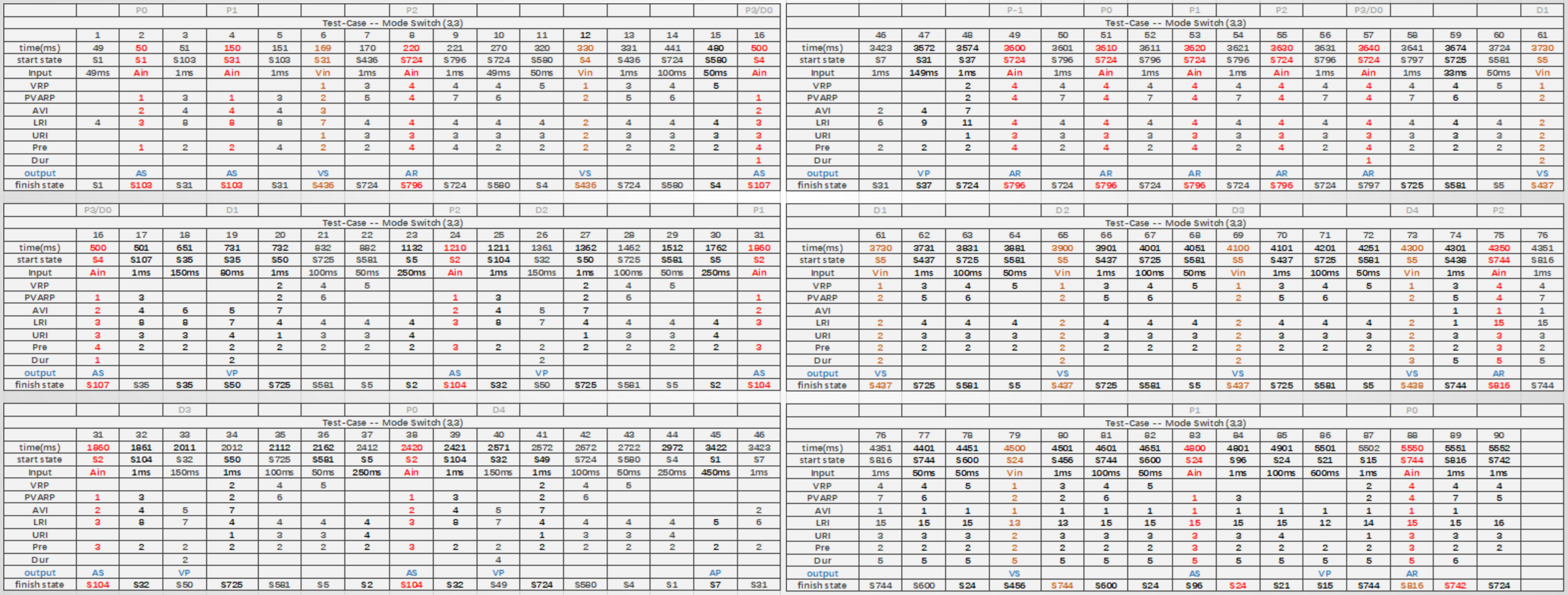
E3
Each transition in PM_S was translated to a transition in PM
Link to traceability Appendix 3

E4
The test cases explored all transitions in PM_S and PM
Link to testing report Appendix 4

Solution Overview *cont.*

E4:Conformance Testing

- Test case generation (90’)



		P0		P1				P2								P3/D0
	Test-Case -- Mode Switch (3,3)															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
time(ms)	49	50	51	150	151	169	170	220	221	270	320	330	331	441	480	500
start state	S1	S1	S103	S31	S103	S31	S436	S724	S796	S724	S580	S4	S436	S724	S580	S4
Input	49ms	Ain	1ms	Ain	1ms	Vin	1ms	Ain	1ms	49ms	50ms	Vin	1ms	100ms	50ms	Ain
VRP						1	3	4	4	4	5	1	3	4	5	
PVARP		1	3	1	3	2	5	4	7	6		2	5	6		1
AVI		2	4	4	4	3										2
LRI	4	3	8	8	8	7	4	4	4	4	4	2	4	4	4	3
URI						1	3	3	3	3	3	2	3	3	3	3
Pre		1	2	2	4	2	2	4	4	2	2	2	2	2	2	4
Dur																1
output		AS		AS		VS		AR				VS				AS
finish state	S1	S103	S31	S103	S31	S436	S724	S796	S724	S580	S4	S436	S724	S580	S4	S107

	P3/D0			D1					P2		D2					P1
	Test-Case -- Mode Switch (3,3)															
	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
time(ms)	500	501	651	731	732	832	882	1132	1210	1211	1361	1362	1462	1512	1762	1860
start state	S4	S107	S35	S35	S50	S725	S581	S5	S2	S104	S32	S50	S725	S581	S5	S2
Input	Ain	1ms	150ms	80ms	1ms	100ms	50ms	250ms	Ain	1ms	150ms	1ms	100ms	50ms	250ms	Ain
VRP					2	4	5					2	4	5		
PVARP	1	3			2	6			1	3		2	6			1
AVI	2	4	6	5	7				2	4	5	7				2
LRI	3	8	8	7	4	4	4	4	3	8	7	4	4	4	4	3
URI	3	3	3	4	1	3	3	4				1	3	3	4	
Pre	4	2	2	2	2	2	2	2	3	2	2	2	2	2	2	3
Dur	1			2							2					
output	AS			VP					AS		VP					AS
finish state	S107	S35	S35	S50	S725	S581	S5	S2	S104	S32	S50	S725	S581	S5	S2	S104

			D3					P0		D4						
	Test-Case -- Mode Switch (3,3)															
	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46
time(ms)	1860	1861	2011	2012	2112	2162	2412	2420	2421	2571	2572	2672	2722	2972	3422	3423
start state	S2	S104	S32	S50	S725	S581	S5	S2	S104	S32	S49	S724	S580	S4	S1	S7
Input	Ain	1ms	150ms	1ms	100ms	50ms	250ms	Ain	1ms	150ms	1ms	100ms	50ms	250ms	450ms	1ms
VRP				2	4	5					2	4	5			
PVARP	1	3		2	6			1	3		2	6				
AVI	2	4	5	7				2	4	5	7					2
LRI	3	8	7	4	4	4	4	3	8	7	4	4	4	4	5	6
URI				1	3	3	4				1	3	3	4		
Pre	3	2	2	2	2	2	2	3	2	2	2	2	2	2	2	2
Dur			2							4						
output	AS		VP					AS		VP					AP	
finish state	S104	S32	S50	S725	S581	S5	S2	S104	S32	S49	S724	S580	S4	S1	S7	S31

				P-1		P0		P1		P2		P3/D0				D1
	Test-Case -- Mode Switch (3,3)															
	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61
time(ms)	3423	3572	3574	3600	3601	3610	3611	3620	3621	3630	3631	3640	3641	3674	3724	3730
start state	S7	S31	S37	S724	S796	S724	S796	S724	S796	S724	S796	S724	S797	S725	S581	S5
Input	1ms	149ms	1ms	Ain	1ms	Ain	1ms	Ain	1ms	Ain	1ms	Ain	1ms	33ms	50ms	Vin
VRP			2	4	4	4	4	4	4	4	4	4	4	4	5	1
PVARP			2	4	7	4	7	4	7	4	7	4	7	6		2
AVI	2	4	7													
LRI	6	9	11	4	4	4	4	4	4	4	4	4	4	4	4	2
URI			1	3	3	3	3	3	3	3	3	3	3	3	3	2
Pre	2	2	2	4	2	4	2	4	2	4	2	4	2	2	2	2
Dur												1				2
output		VP		AR		AR		AR		AR		AR				VS
finish state	S31	S37	S724	S796	S724	S796	S724	S796	S724	S796	S724	S797	S725	S581	S5	S437

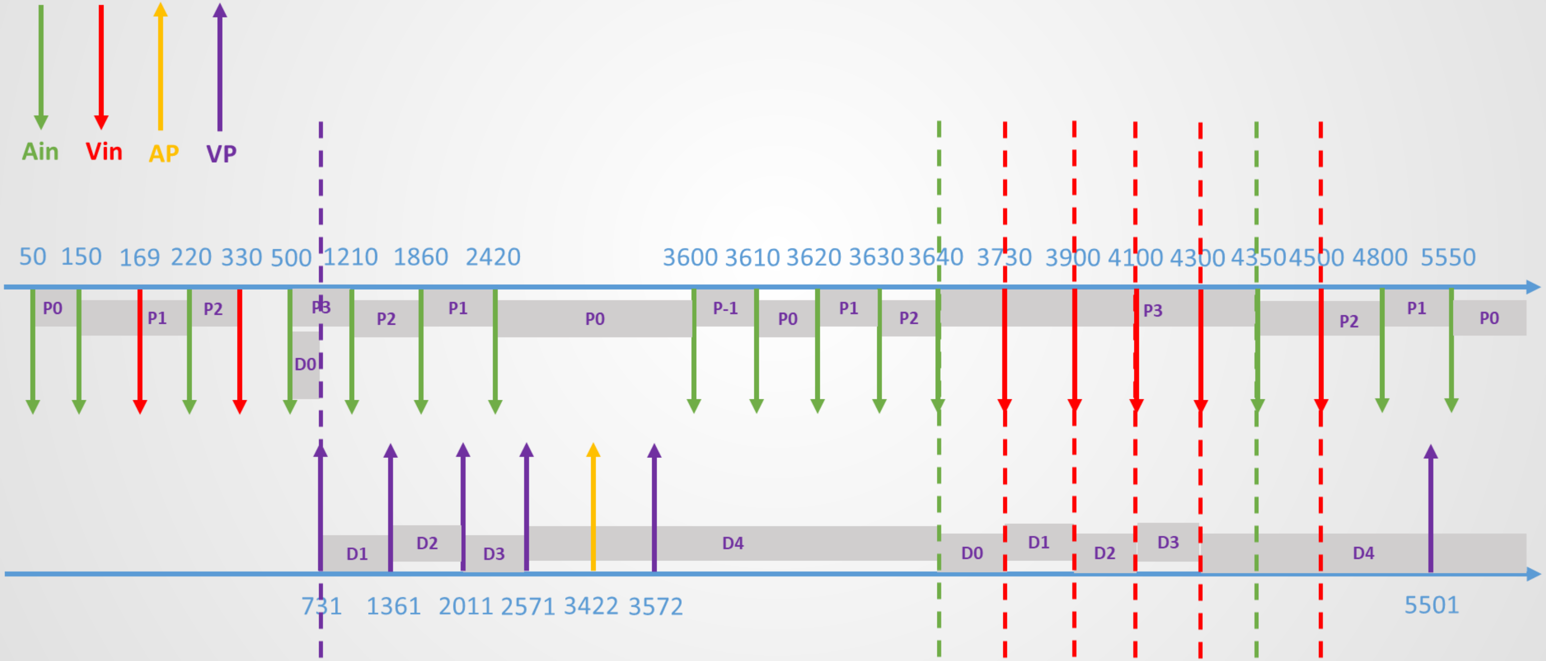
	D1				D2				D3				D4		P2	
	Test-Case -- Mode Switch (3,3)															
	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76
time(ms)	3730	3731	3831	3881	3900	3901	4001	4051	4100	4101	4201	4251	4300	4301	4350	4351
start state	S5	S437	S725	S581	S5	S437	S725	S581	S5	S437	S725	S581	S5	S438	S744	S816
Input	Vin	1ms	100ms	50ms	Vin	1ms	100ms	50ms	Vin	1ms	100ms	50ms	Vin	1ms	Ain	1ms
VRP	1	3	4	5	1	3	4	5	1	3	4	5	1	3	4	4
PVARP	2	5	6		2	5	6		2	5	6		2	5	4	7
AVI														1	1	1
LRI	2	4	4	4	2	4	4	4	2	4	4	4	2	1	15	15
URI	2	3	3	3	2	3	3	3	2	3	3	3	2	3	3	3
Pre	2	2	2	2	2	2	2	2	2	2	2	2	2	2	3	2
Dur	2				2				2				3	5	5	5
output	VS				VS				VS				VS		AR	
finish state	S437	S725	S581	S5	S437	S725	S581	S5	S437	S725	S581	S5	S438	S744	S816	S744

								P1					P0			
	Test-Case -- Mode Switch (3,3)															
	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	
time(ms)	4351	4401	4451	4500	4501	4601	4651	4800	4801	4901	5501	5502	5550	5551	5552	
start state	S816	S744	S600	S24	S456	S744	S600	S24	S96	S24	S21	S15	S744	S816	S742	
Input	1ms	50ms	50ms	Vin	1ms	100ms	50ms	Ain	1ms	100ms	600ms	1ms	Ain	1ms	1ms	
VRP	4	4	5	1	3	4	5					2	4	4	4	
PVARP	7	6		2	2	6		1	3			2	4	7	5	
AVI	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
LRI	15	15	15	13	13	15	15	15	15	15	12	14	15	15	16	
URI	3	3	3	2	3	3	3	3	3	4		1	3	3	3	
Pre	2	2	2	2	2	2	2	3	2	2	2	2	3	2	2	
Dur	5	5	5	5	5	5	5	5	5	5	5	5	5	6		
output				VS				AS			VP		AR			
finish state	S744	S600	S24	S456	S744	S600	S24	S96	S24	S21	S15	S744	S816	S742	S724	

Solution Overview *cont.*

E4:Conformance Testing

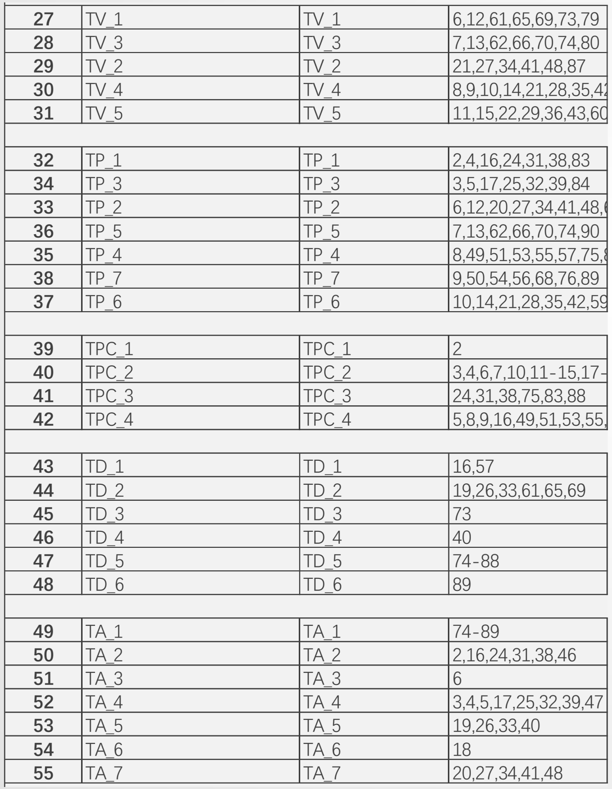
- Test case generation



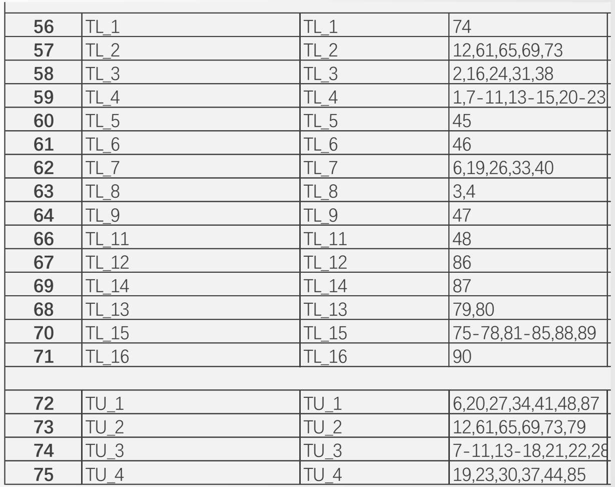
Solution Overview *cont.*

E4:Conformance Testing

- Test case coverage criteria: 100%



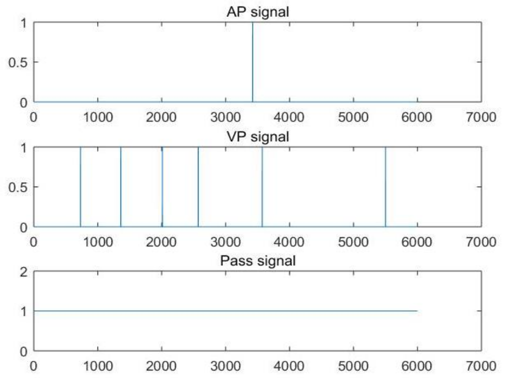
27	TV_1	TV_1	6,12,61,65,69,73,79
28	TV_3	TV_3	7,13,62,66,70,74,80
29	TV_2	TV_2	21,27,34,41,48,87
30	TV_4	TV_4	8,9,10,14,21,28,35,42
31	TV_5	TV_5	11,15,22,29,36,43,60
32	TP_1	TP_1	2,4,16,24,31,38,83
34	TP_3	TP_3	3,5,17,25,32,39,84
33	TP_2	TP_2	6,12,20,27,34,41,48,6
36	TP_5	TP_5	7,13,62,66,70,74,90
35	TP_4	TP_4	8,49,51,53,55,57,75,8
38	TP_7	TP_7	9,50,54,56,68,76,89
37	TP_6	TP_6	10,14,21,28,35,42,59
39	TPC_1	TPC_1	2
40	TPC_2	TPC_2	3,4,6,7,10,11-15,17-
41	TPC_3	TPC_3	24,31,38,75,83,88
42	TPC_4	TPC_4	5,8,9,16,49,51,53,55,
43	TD_1	TD_1	16,57
44	TD_2	TD_2	19,26,33,61,65,69
45	TD_3	TD_3	73
46	TD_4	TD_4	40
47	TD_5	TD_5	74-88
48	TD_6	TD_6	89
49	TA_1	TA_1	74-89
50	TA_2	TA_2	2,16,24,31,38,46
51	TA_3	TA_3	6
52	TA_4	TA_4	3,4,5,17,25,32,39,47
53	TA_5	TA_5	19,26,33,40
54	TA_6	TA_6	18
55	TA_7	TA_7	20,27,34,41,48

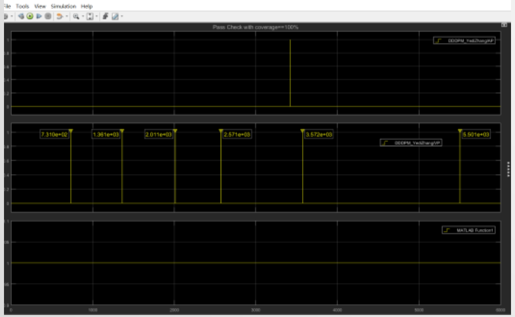


56	TL_1	TL_1	74
57	TL_2	TL_2	12,61,65,69,73
58	TL_3	TL_3	2,16,24,31,38
59	TL_4	TL_4	1,7-11,13-15,20-23
60	TL_5	TL_5	45
61	TL_6	TL_6	46
62	TL_7	TL_7	6,19,26,33,40
63	TL_8	TL_8	3,4
64	TL_9	TL_9	47
66	TL_11	TL_11	48
67	TL_12	TL_12	86
69	TL_14	TL_14	87
68	TL_13	TL_13	79,80
70	TL_15	TL_15	75-78,81-85,88,89
71	TL_16	TL_16	90
72	TU_1	TU_1	6,20,27,34,41,48,87
73	TU_2	TU_2	12,61,65,69,73,79
74	TU_3	TU_3	7-11,13-18,21,22,28
75	TU_4	TU_4	19,23,30,37,44,85

Solution Overview *cont.* E4:Conformance Testing

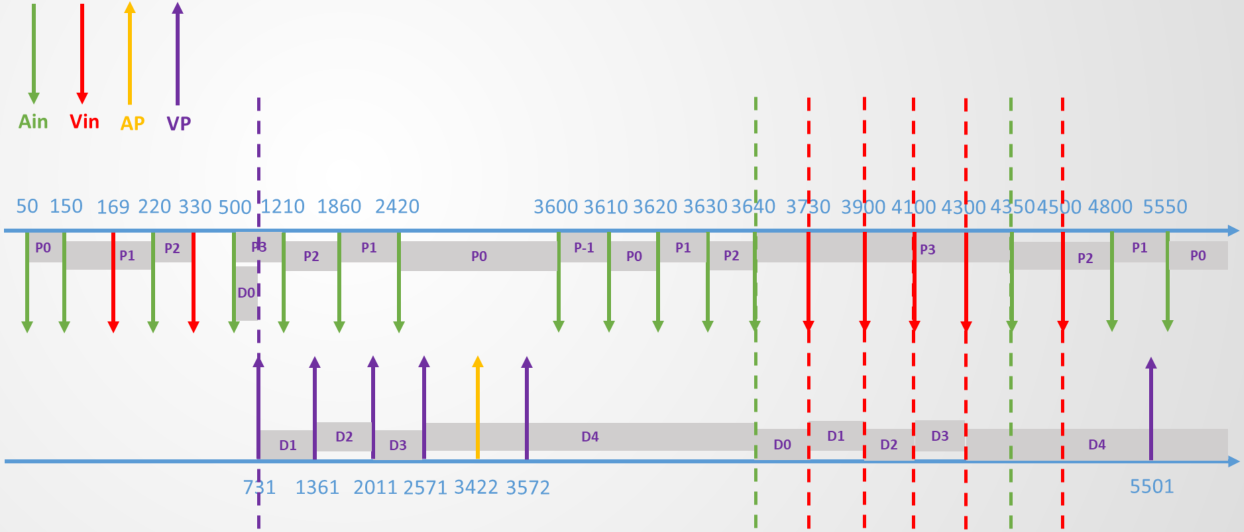
- Evaluation the conformance: the Stateflow Model and the Matlab Code





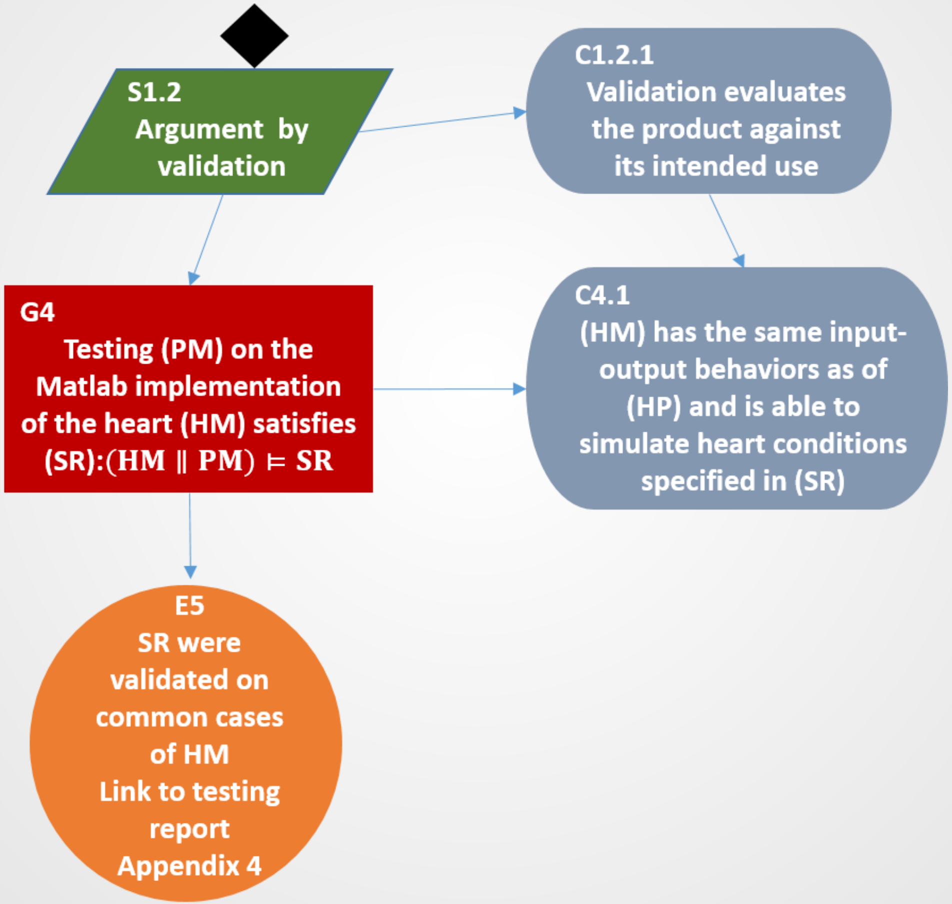
Pass Check with coverage==100%

7.310e+02 1.361e+03 2.011e+03 2.571e+03 3.572e+03 5.501e+03



Solution Overview *cont.*

Safety Assurance: A Top Level



S1.2
Argument by validation

C1.2.1
Validation evaluates the product against its intended use

G4
Testing (PM) on the Matlab implementation of the heart (HM) satisfies (SR):$(\mathbf{HM} \parallel \mathbf{PM}) \vDash \mathbf{SR}$

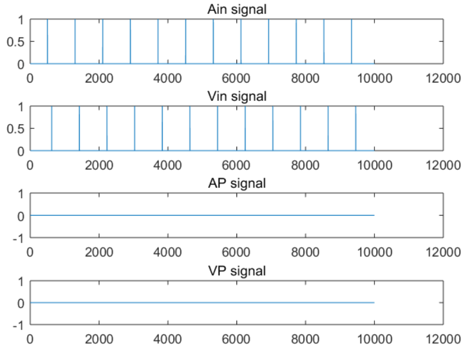
C4.1
(HM) has the same input-output behaviors as of (HP) and is able to simulate heart conditions specified in (SR)

E5
SR were validated on common cases of HM
Link to testing report
Appendix 4

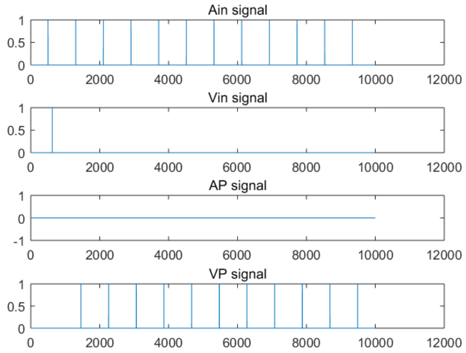
Solution Overview *cont.*

E5:Validation on common cases

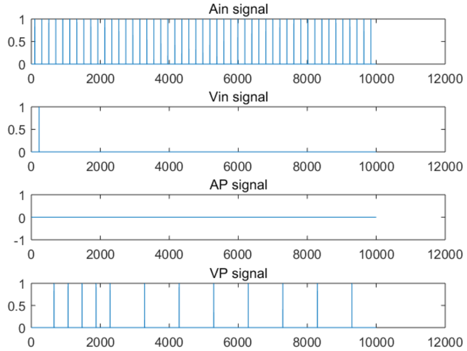
- Validate on **Matlab Code**, using common Physiological Heart Models



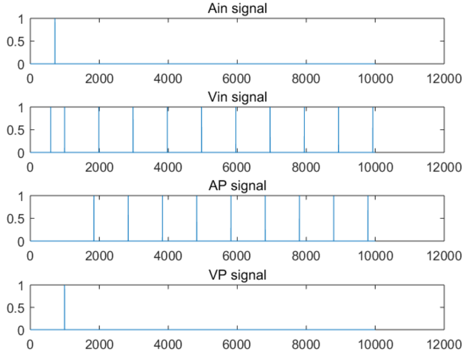
Normal Sinus Rhythm



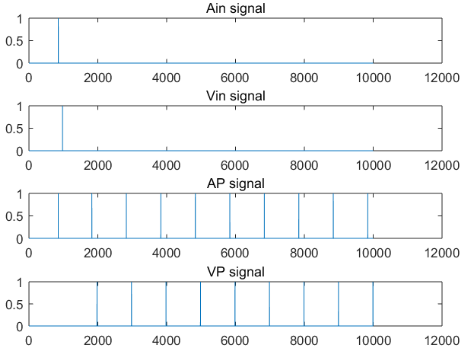
Normal Sinus Rhythm with AV delay/block



Atrial Flutter/SVT (Mode Switch)



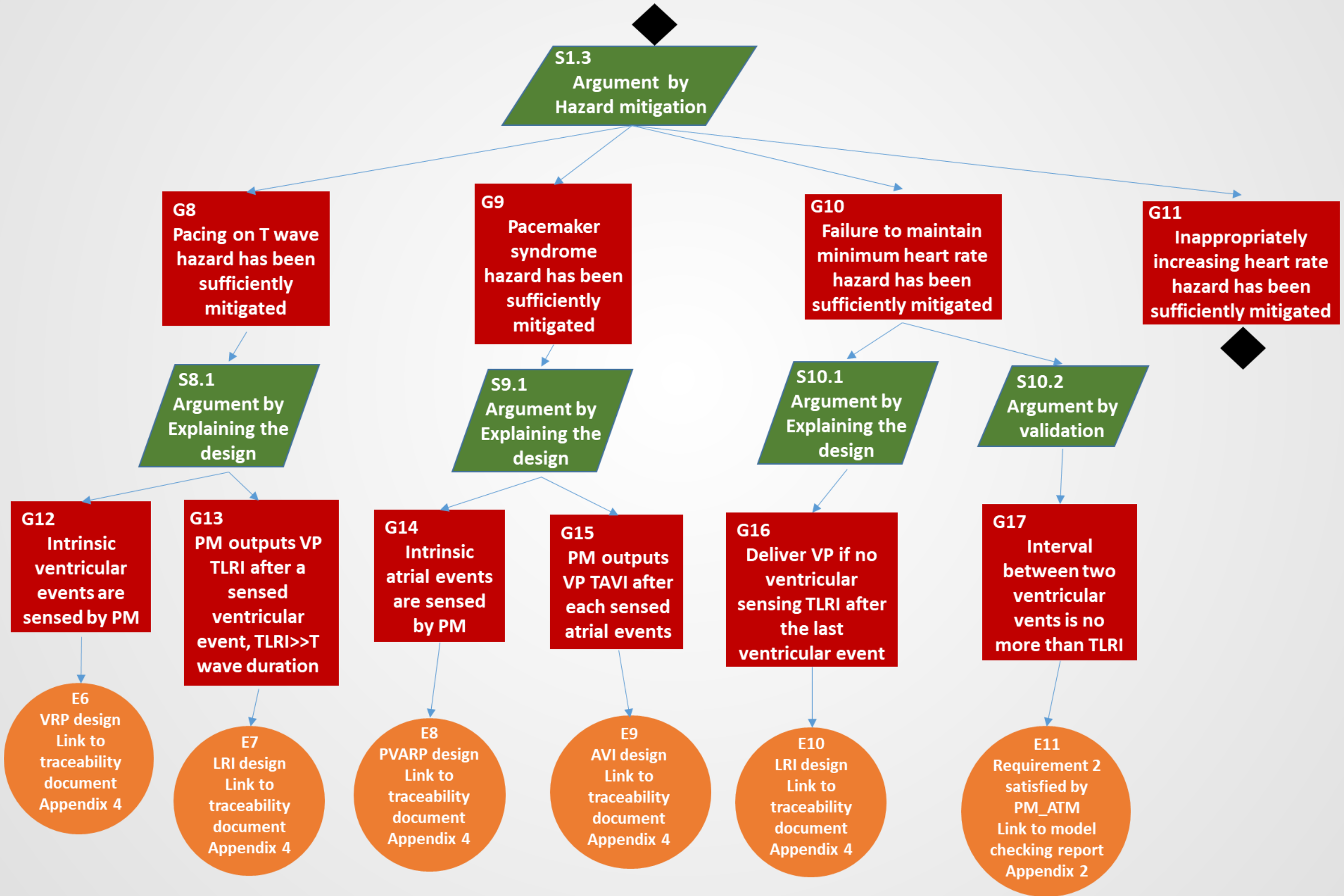
Bradycardia



Bradycardia with AV delay/block

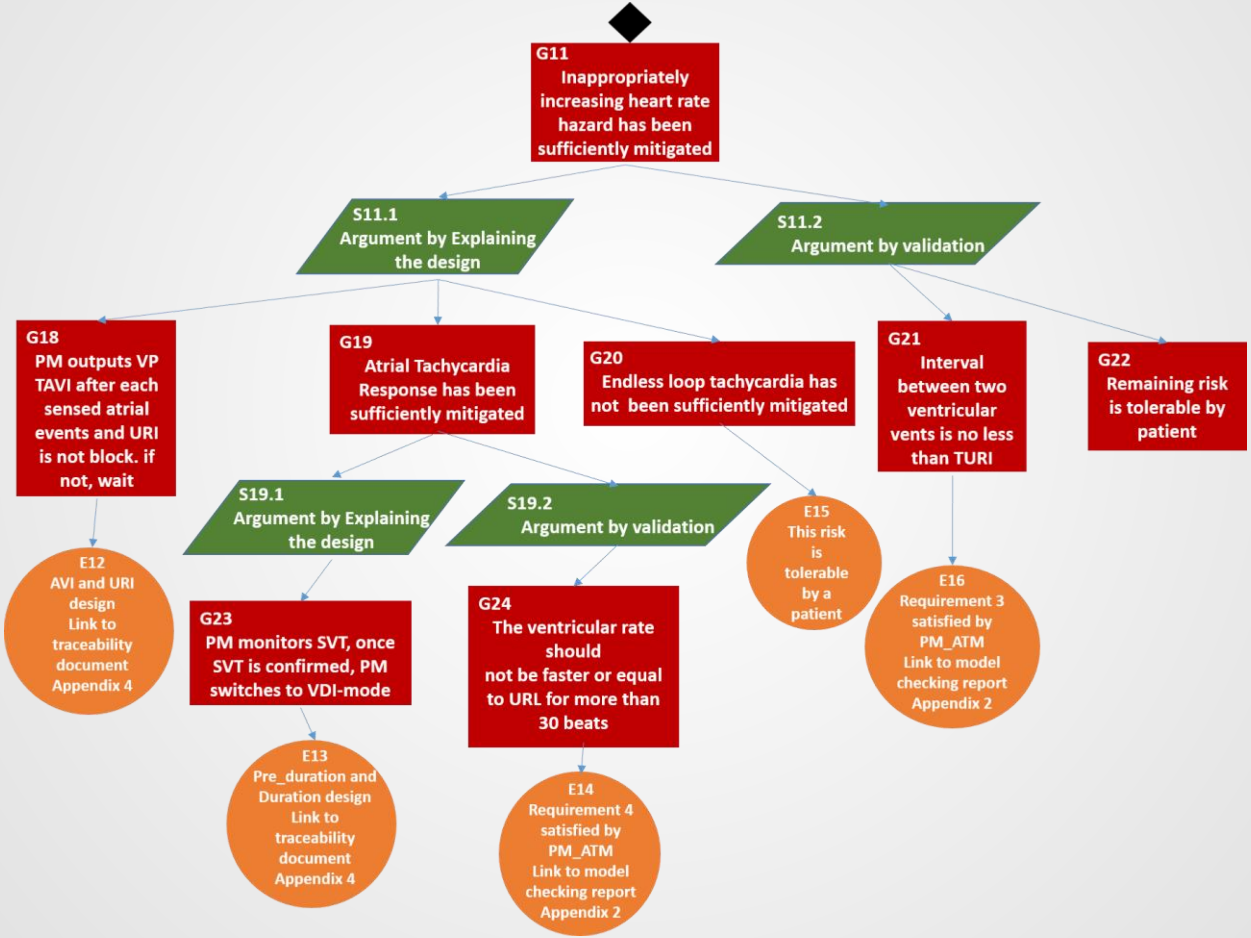
Solution Overview *cont.*

Safety Assurance: A Top Level


S1.3 Argument by Hazard mitigation
G8 Pacing on T wave hazard has been sufficiently mitigated
G9 Pacemaker syndrome hazard has been sufficiently mitigated
G10 Failure to maintain minimum heart rate hazard has been sufficiently mitigated
G11 Inappropriately increasing heart rate hazard has been sufficiently mitigated
S8.1 Argument by Explaining the design
S9.1 Argument by Explaining the design
S10.1 Argument by Explaining the design
S10.2 Argument by validation
G12 Intrinsic ventricular events are sensed by PM
G13 PM outputs VP TLRI after a sensed ventricular event, TLRI>>T wave duration
G14 Intrinsic atrial events are sensed by PM
G15 PM outputs VP TAVI after each sensed atrial events
G16 Deliver VP if no ventricular sensing TLRI after the last ventricular event
G17 Interval between two ventricular vents is no more than TLRI
E6 VRP design Link to traceability document Appendix 4
E7 LRI design Link to traceability document Appendix 4
E8 PVARP design Link to traceability document Appendix 4
E9 AVI design Link to traceability document Appendix 4
E10 LRI design Link to traceability document Appendix 4
E11 Requirement 2 satisfied by PM_ATM Link to model checking report Appendix 2

Solution Overview *cont.*

Safety Assurance: A Top Level



G11
Inappropriately increasing heart rate hazard has been sufficiently mitigated

S11.1
Argument by Explaining the design

S11.2
Argument by validation

G18
PM outputs VP TAVI after each sensed atrial events and URI is not block. if not, wait

G19
Atrial Tachycardia Response has been sufficiently mitigated

G20
Endless loop tachycardia has not been sufficiently mitigated

G21
Interval between two ventricular vents is no less than TURI

G22
Remaining risk is tolerable by patient

S19.1
Argument by Explaining the design

S19.2
Argument by validation

E15
This risk is tolerable by a patient

E12
AVI and URI design Link to traceability document Appendix 4

G23
PM monitors SVT, once SVT is confirmed, PM switches to VDI-mode

G24
The ventricular rate should not be faster or equal to URL for more than 30 beats

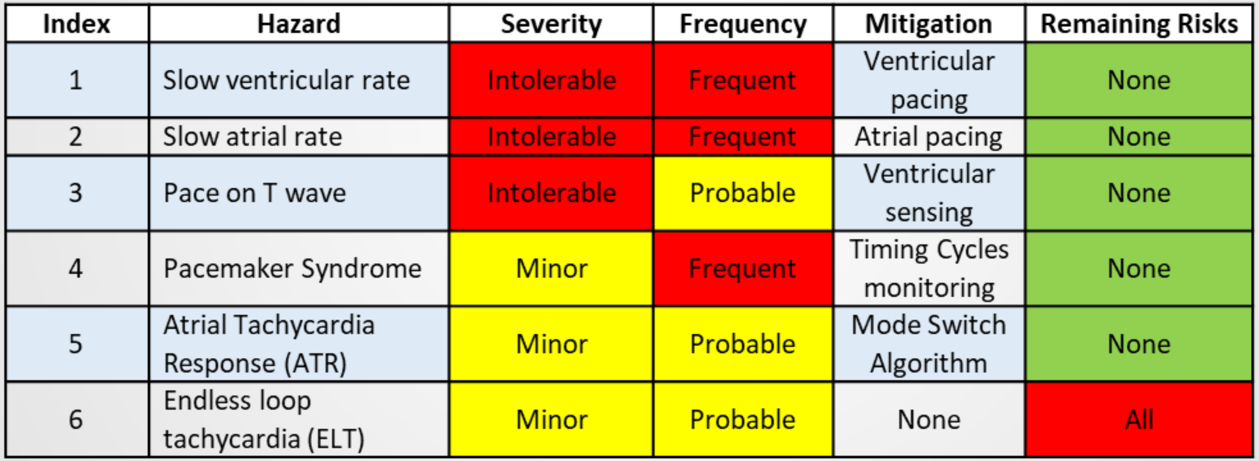
E16
Requirement 3 satisfied by PM_ATM Link to model checking report Appendix 2

E13
Pre_duration and Duration design Link to traceability document Appendix 4

E14
Requirement 4 satisfied by PM_ATM Link to model checking report Appendix 2

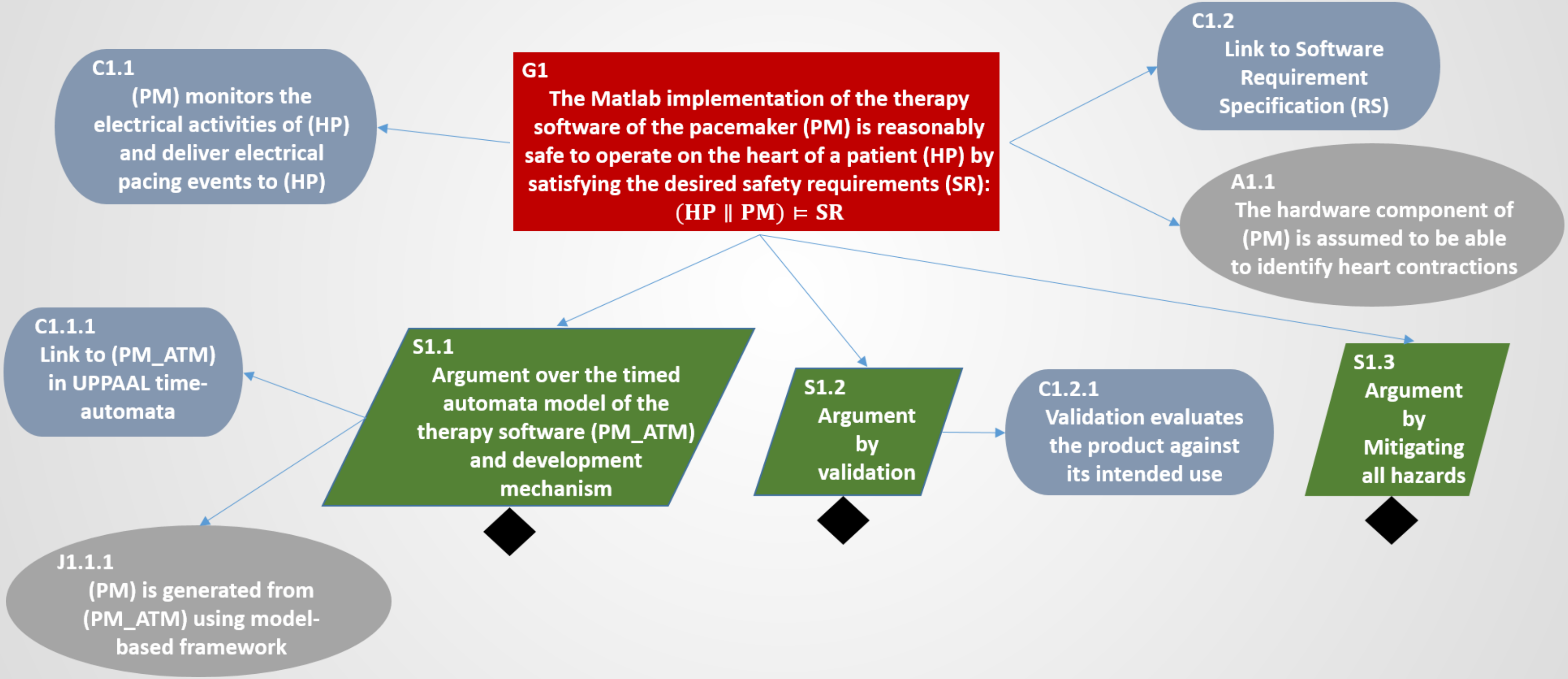
S1.3: Hazard Analysis

- Hazard 1~5 were sufficiently mitigated
- Hazard 6 cannot be addressed with the current system architecture
- Hazard 6 was deemed tolerable



Index	Hazard	Severity	Frequency	Mitigation	Remaining Risks
1	Slow ventricular rate	Intolerable	Frequent	Ventricular pacing	None
2	Slow atrial rate	Intolerable	Frequent	Atrial pacing	None
3	Pace on T wave	Intolerable	Probable	Ventricular sensing	None
4	Pacemaker Syndrome	Minor	Frequent	Timing Cycles monitoring	None
5	Atrial Tachycardia Response (ATR)	Minor	Probable	Mode Switch Algorithm	None
6	Endless loop tachycardia (ELT)	Minor	Probable	None	All

Summary



G1
The Matlab implementation of the therapy software of the pacemaker (PM) is reasonably safe to operate on the heart of a patient (HP) by satisfying the desired safety requirements (SR): $(\mathrm{HP} \parallel \mathrm{PM}) \vDash \mathrm{SR}$

C1.1
(PM) monitors the electrical activities of (HP) and deliver electrical pacing events to (HP)

C1.2
Link to Software Requirement Specification (RS)

A1.1
The hardware component of (PM) is assumed to be able to identify heart contractions

C1.1.1
Link to (PM_ATM) in UPPAAL time-automata

S1.1
Argument over the timed automata model of the therapy software (PM_ATM) and development mechanism

S1.2
Argument by validation

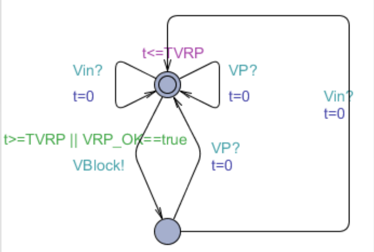
C1.2.1
Validation evaluates the product against its intended use

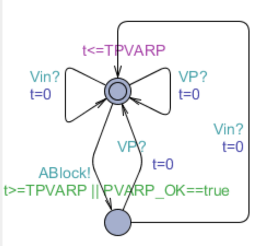
S1.3
Argument by Mitigating all hazards

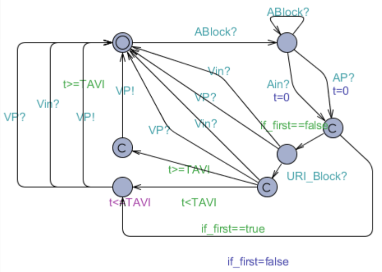
J1.1.1
(PM) is generated from (PM_ATM) using model-based framework

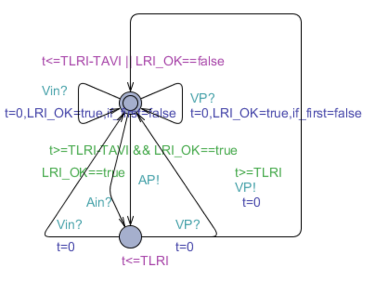
Revision History

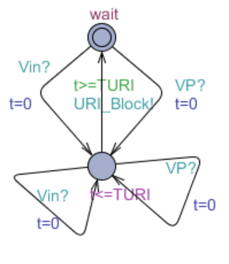
- Remove a lot of redundant transitions and invariants safely without losing any functionality when designing DDD pacemaker in UPPAAL.








t<=TLRI-TAVI || LRI_OK==false
Vin?
t=0,LRI_OK=true,if_first=false
VP?
t=0,LRI_OK=true,if_first=false
t>=TLRI-TAVI && LRI_OK==true
LRI_OK==true
AP!
t>=TLRI
VP!
t=0
Ain?
Vin?
t=0
VP?
t=0
t<=TLRI



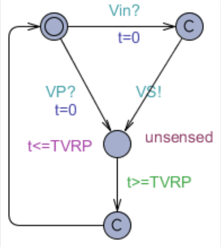
VRP

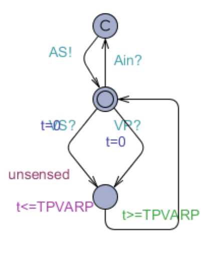
PVARP

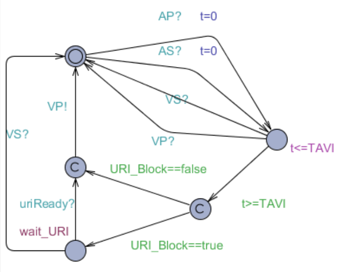
AVI

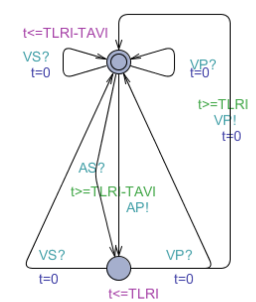
LRI

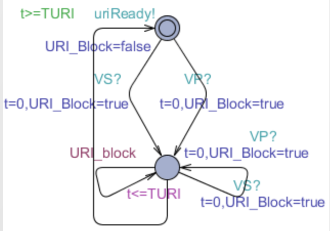
URI





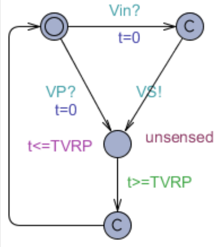




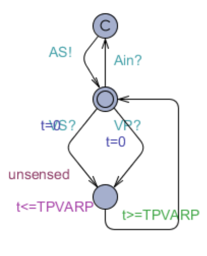


Revision History

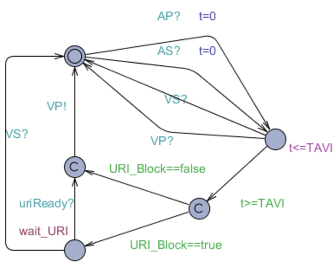
- Add Mode Switch functionality
- Remove one transition again when doing conformance testing with test cases safely without losing any functionality



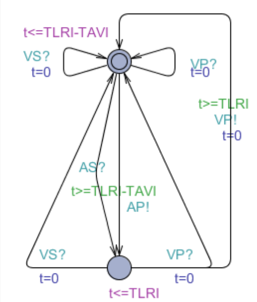
VRP



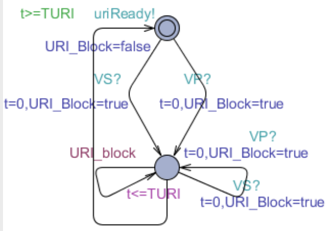
PVARP



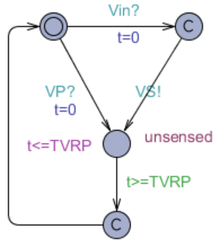
AVI

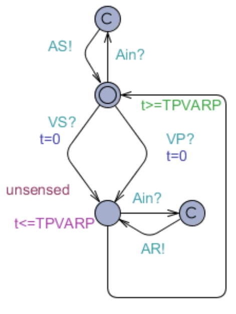


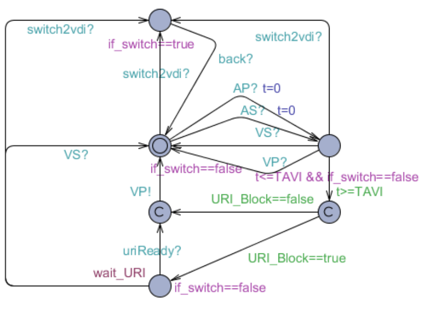
LRI

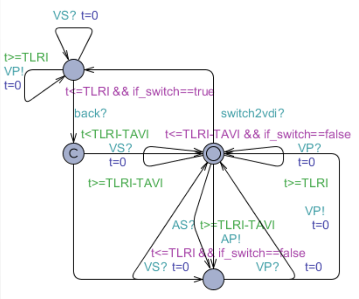


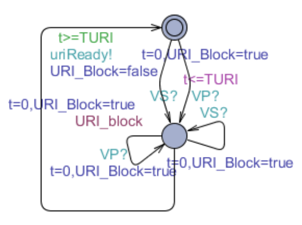
URI



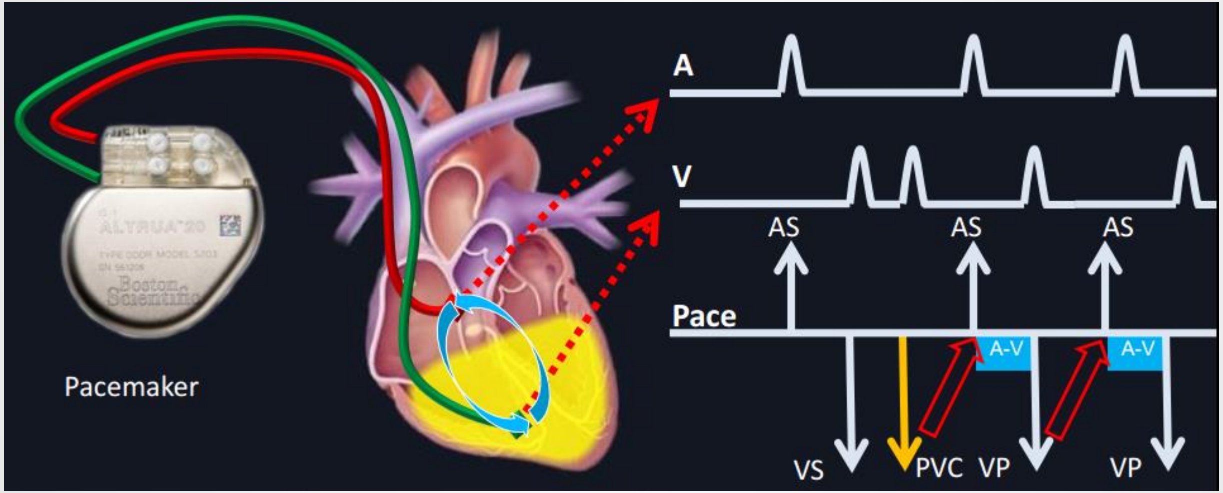




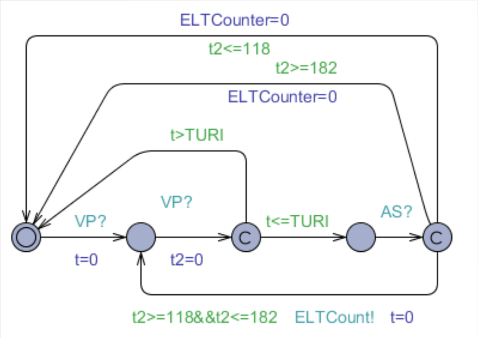


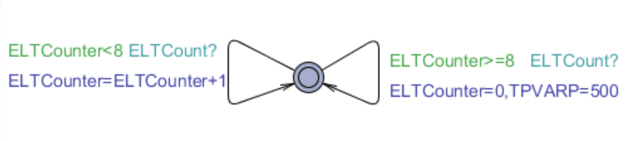


Remaining Problems -- ELT



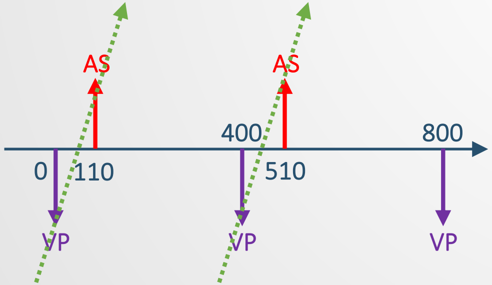
Remaining Problems -- ELT

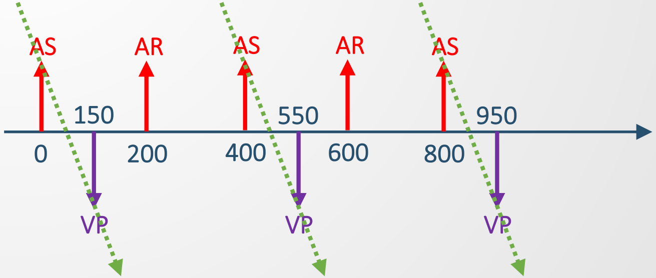




- For above solution

  Not robust → False negative when conduction delay $T_{cond} \in (\mathbf{100ms}, \mathbf{118ms}) \cup (\mathbf{182ms}, \mathbf{250ms})$

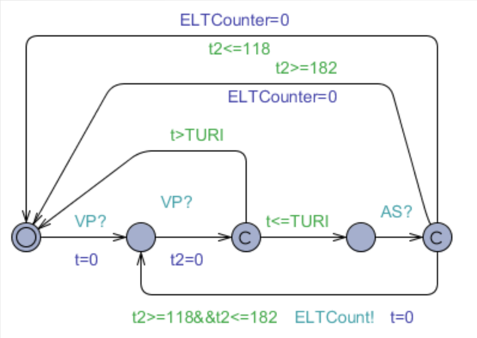


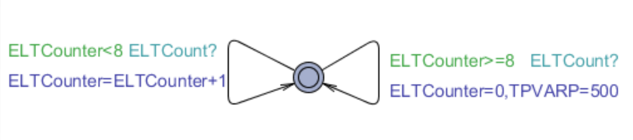


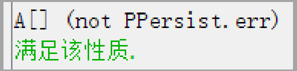
- More Generally Speaking

  Direction undistinguishable → False positive when Atrial event is from SA node with AV block

Remaining Problems -- ELT

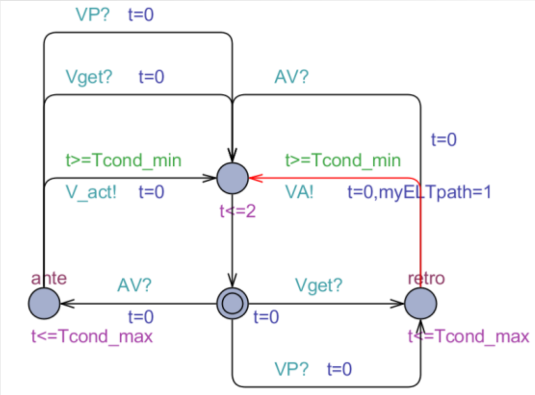


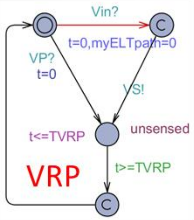


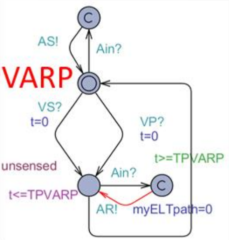


A[] (not PPersist.err)
满足该性质.

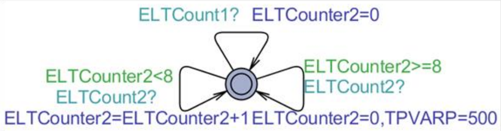
Add another lead → monitor path， instead of time

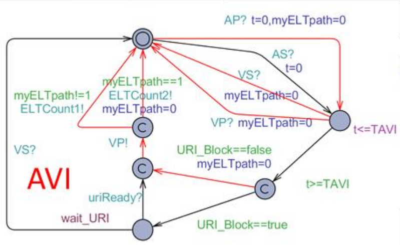


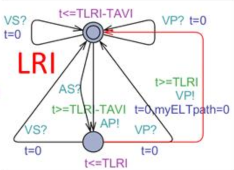




ELT







Thank you!

Any Question?