System Modelling using Event-B

Neeraj Kumar Singh

McMaster Centre for Software Certification

March 25, 2014

Outline

1. Model Development
2. Machines and Contexts
3. Event
4. Action
5. Proof Obligation (POs)
6. Refinement
7. Tools and Books
8. Case Studies

Model Development with Event-B

Event-B

- is designed by J.-R. Abrial in 1988.
- is not a programming language.
- notations are based-on set-theory and first-order logic.
- notations are used to develop the mathematical models of discrete transition systems (system behaviour).
- supports refinement based development.
- models can be developed using the Rodin Platform.

Modelling Systems

1. Problem Understanding
2. Identifying and organizing the requirements and properties
3. To specify a very abstract model
4. Adding new requirements using small refinement steps
5. Stop the refinement if model is enough concrete

Set-theoretical Notations

Name	Syntax	Definition
Binary relation	$s \leftrightarrow t$	$(\mathcal{P})(s \times t)$
Composition of relations	$r_1 \ ; \ r_2$	$\{x, y \mid x \in a \ \wedge \ y \in b \wedge$ $\exists z.(z \in c \ \wedge \ x, z \in r_1 \ \wedge \ z, y \in r_2\}$
Inverse relation	r^{-1}	$\{x, y \mid x \in \mathcal{P}(a) \ \wedge \ y \in \mathcal{D}(b) \ \wedge \ a \mapsto b \in r)\}$
Domain	$dom(r)$	$\{a \mid a \in s \ \wedge \ \exists b.(b \in t \ \wedge \ a \mapsto b \in r)\}$
Range	$ran(r)$	$dom(r^{-1})$
Identity	$id(s)$	$\{x, y \mid x \in s \ \wedge \ y \in s \ \wedge \ x = y\}$
Restriction	$s \lhd r$	$id(s); r$
Co-restriction	$r \rhd s$	$r; id(s)$
Anti-restriction	$s \lhd\!\!\!- r$	$(dom(r) - s) \lhd r$
Anti-co-restriction	$r \rhd\!\!\!- s$	$r \rhd (ran(r) - s)$
Image	$r[w]$	$ran(w \lhd r)$
Overriding	$q \lhd\!\!\!+ r$	$(\mathrm{dom}(r) \lhd q) \cup r$
Partial function	$s \nrightarrow t$	$\{r \mid r \in s \leftrightarrow t \ \wedge \ (r^{-1}; r) \subseteq id(t)\}$

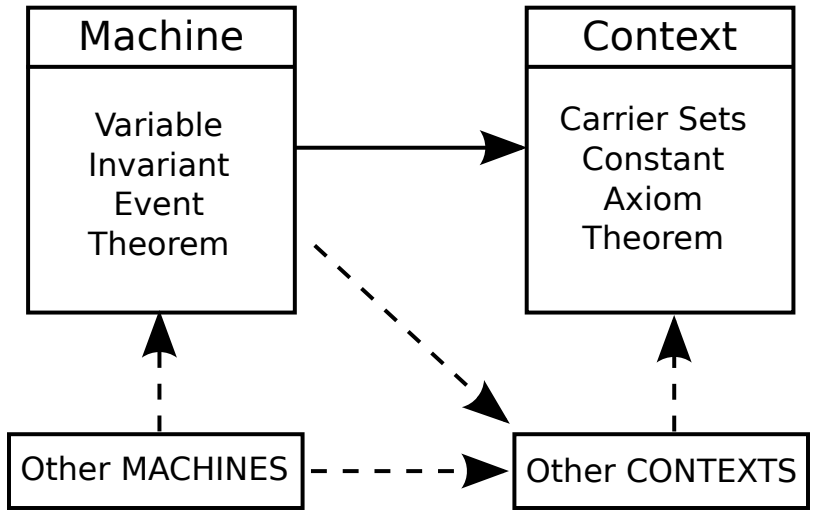
Machines and Contexts

Contexts

Contexts are used to formalize the static structure of a discrete system (constants and axioms)

Machines

Machines are used to formalize the dynamic structure of a discrete system (variables, invariants, and events).



Context Structure

context
< *context identifier* >
extendes
< *context identifier* >
sets
< *set identifier* >
constants
< *constant identifier* >
axioms
< *label* >:< *predicate* >
theorems
< *label* >:< *predicate* >
end

context
c1
sets
S
constants
x
y
axioms
axm1: $x \in \mathbb{N}$
axm2: $y \in 1..x \rightarrow S$
theorems
thm1: $x \in \mathbb{N}1$
end

Machine Structure

machine
< *machine identifier* >
refines
< *machine identifier* >
sees
< *context identifier* >
variables
< *variable identifier* >
invariants
< *label* >:< *predicate* >
theorems
< *label* >:< *predicate* >
events
< *initialisation* >< *evn* 1...*evn n* >
variant
< *variant* >
end

machine
m1
sees
c1
variables
i
invariants
axm1: $i \in 1..x$
events
...
end

Event Structure

$< \textit{event identifier} > \triangleq$
status
$\{\textit{ordinary}, \textit{convergent}, \textit{anticipated}\}$
refines
$< \textit{event identifier} >$
any
$< \textit{parameter identifier} >$
where *or* **when**
$< \textit{label} >: < \textit{predicate} >$
with
$< \textit{label} >: < \textit{witness} >$
then
$< \textit{label} >: < \textit{action} >$
end

Event *Update*
refines
U*pdate*
any
a
where
grd1: $a \in 1..x$
then
act1: $i := a$
end

Events

Event: E	**Before-After Predicate**
BEGIN $x : \mid P(x, x')$ END	$P(x, x')$
WHEN $G(x)$ THEN $x : \mid P(x, x')$ END	$G(x) \wedge P(x, x')$
ANY t WHERE $G(t, x)$ THEN $x : \mid P(t, x, x')$ END	$\exists t.(G(t, x) \wedge P(t, x, x'))$

Event: E	**Guard:** $grd(E)$
BEGIN $x : \mid P(x, x')$ END	$TRUE$
WHEN $G(x)$ THEN $x : \mid P(x, x')$ END	$G(x)$
ANY t WHERE $G(t, x)$ THEN $x : \mid P(t, x, x')$ END	$\exists t.G(t, x)$

Actions

Actions

An action modifies one or more state variables. There are two types of actions: deterministic and non-deterministic

Deterministic Action

$x := p + q$

$y := max(p, q)$

Non-Deterministic Action

$x : |x' = p + q$

$x, y : |x' > x \wedge y' < x'$

$x :\in \{p + 1, q + 3, r + 4\}$

$x : |x' \in \{p + 1, q + 3, r + 4\}$

Proof Obligation

The POs are automatically generated by a Rodin Platform tool called the Proof Obligation Generator. A list of POs is given as follows:

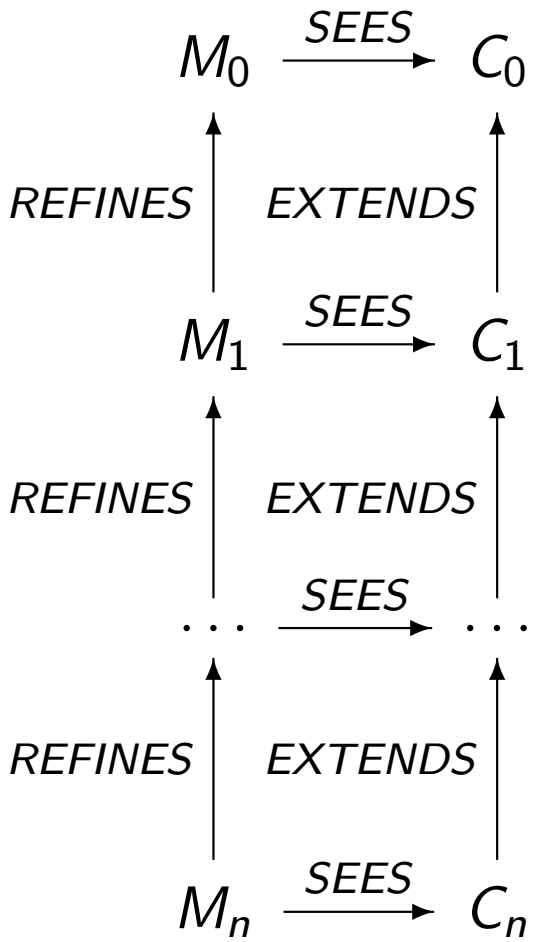
- Well-definedness (WD)
- Invariant preservation (INV)
- Non-deterministic action feasibility (FIS)
- Guard strengthening(GRD)
- Simulation (SIM)
- Numeric variant (NAT)
- Proving theorems (THM)

Proof Obligations

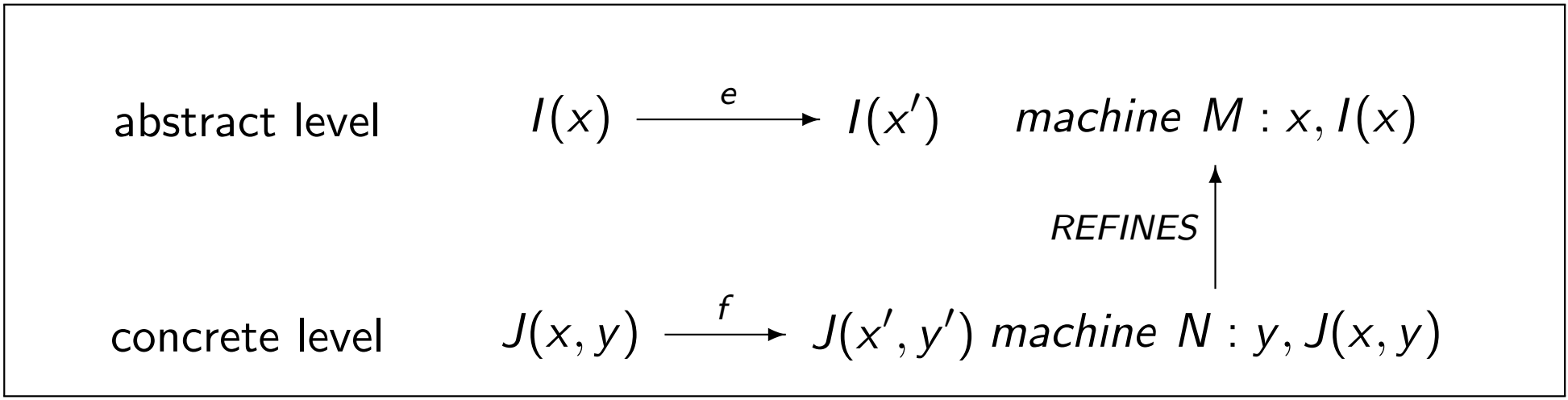
INV1	$A(s,c) \wedge G(x,s,c,v) \wedge BA(e)(x,s,c,v,v')$ $\vdash$ $I(s,c,v')$
INV2	$A(s,c) \wedge I(s,c,v) \wedge G(x,s,c,v) \wedge BA(e)(x,s,c,v,v')$ $\vdash$ $I(s,c,v')$
FIS	$A(s,c) \wedge I(s,c,v) \wedge G(x,s,c,v)$ $\vdash$ $\exists v'.BA(e)(x,s,c,v,v')$
GRD	$A(s,c) \wedge I(s,c,v) \wedge J(s,c,v,w) \wedge H(y,s,c,w) \wedge Wit(x,y,s,c,w)$ $\vdash$ $grd(x,s,c,v)$
DEAD	$A(s,c) \wedge I(s,c,v)$ $\vdash$ $(grd(g_1) \vee ... \vee grd(g_n))$

Refinement

- To add more details (like superposition).
- To introduce a set of new events, and safety properties.
- To prove that the concrete behaviors are abstract ones.
- Each new event refines SKIP.
- No deadlock



Key Steps of the Refinement



- An event f simulates an event e.
- f modifies y
- e modifies x
- If e preserves $I(x)$ and f simulates e, then f preserves $I(x)$

$$I(x) \wedge J(x, y) \wedge BA(f)(y, y') \Rightarrow \exists x'.(BA(e)(x, x') \wedge J(x', y'))$$

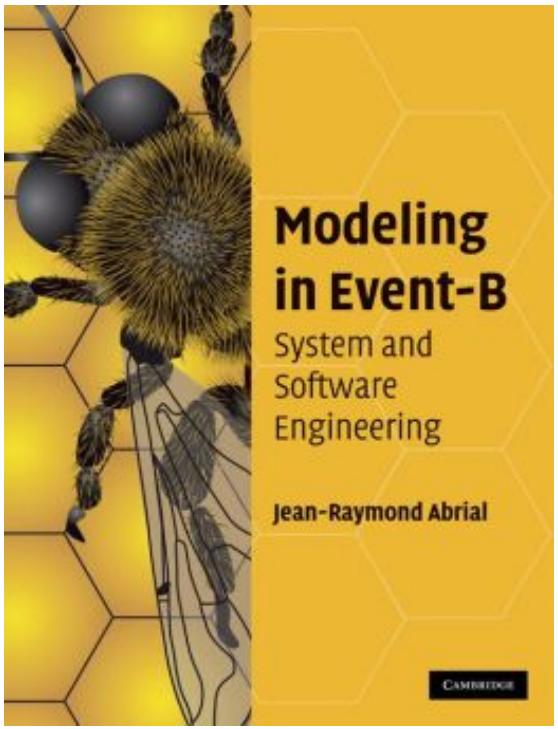
The invariant in the refinement model is preserved by the refined event and the activation of the refined event triggers the corresponding abstract event.

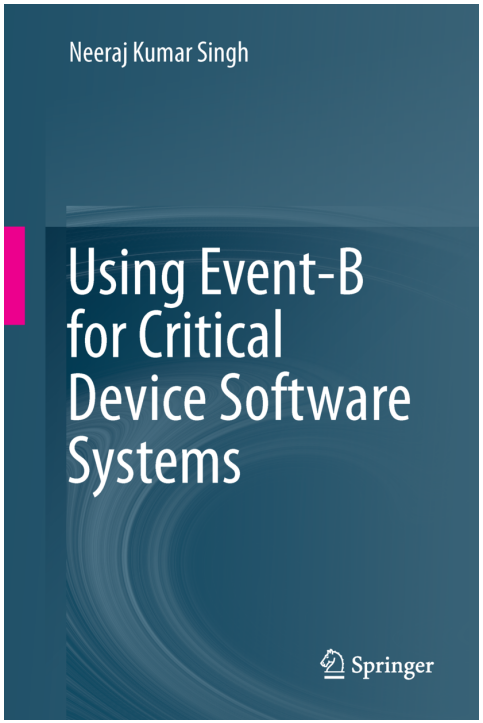
Proof Obligations for Refinement

REF1	$InitC(y)$ $\Rightarrow$ $\exists x.(InitA(x) \wedge J(x, y)$
REF2	$I(x) \wedge J(x, y) \wedge BAC(y, y')$ $\Rightarrow$ $\exists x.(BAA(x, x') \wedge J(x', y')$
REF3	$I(x) \wedge J(x, y) \wedge BAC(y, y')$ $\Rightarrow$ $J(x, y')$
REF3	$I(x) \wedge J(x, y) \wedge (G_1(x) \vee ... \vee G_n(x))$ $\Rightarrow$ $H_1(y) \vee ... \vee H_k(y)$

Tools and Books

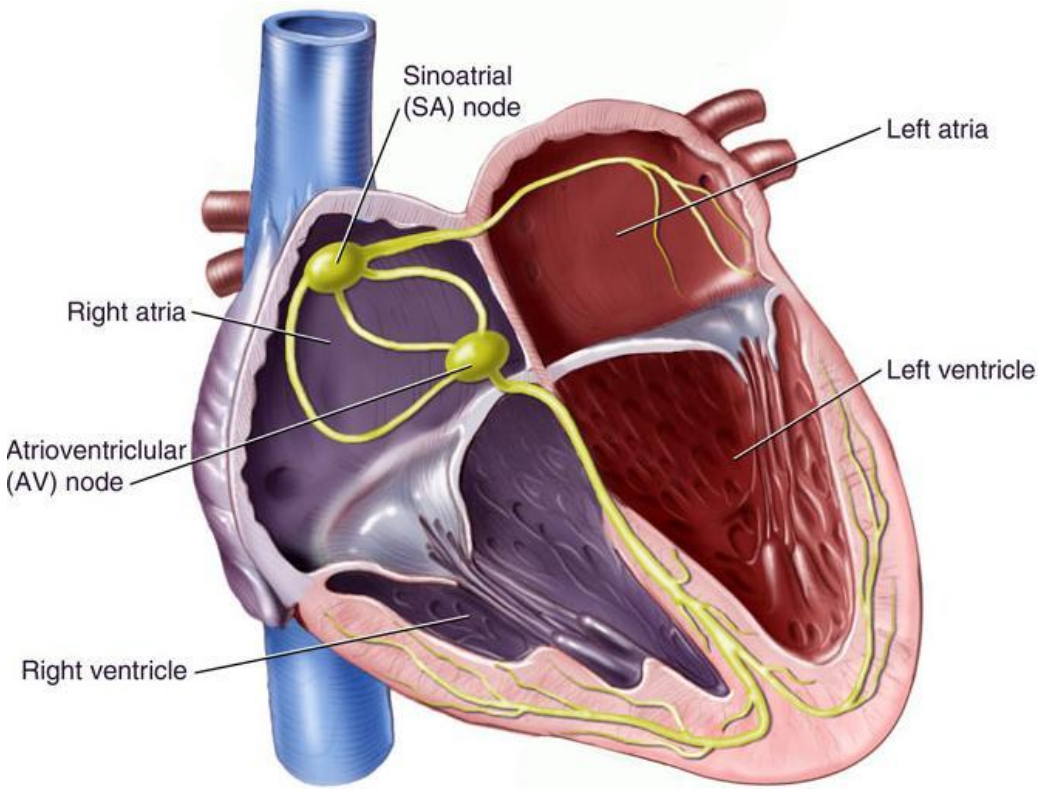
- RODIN Platform: http://www.event-b.org/platform.html
- Event B: http://www.event-b.org
- ProB :http://www.stups.uni-duesseldorf.de/ProB
- EB2ALL Toolset: http://eb2all.loria.fr



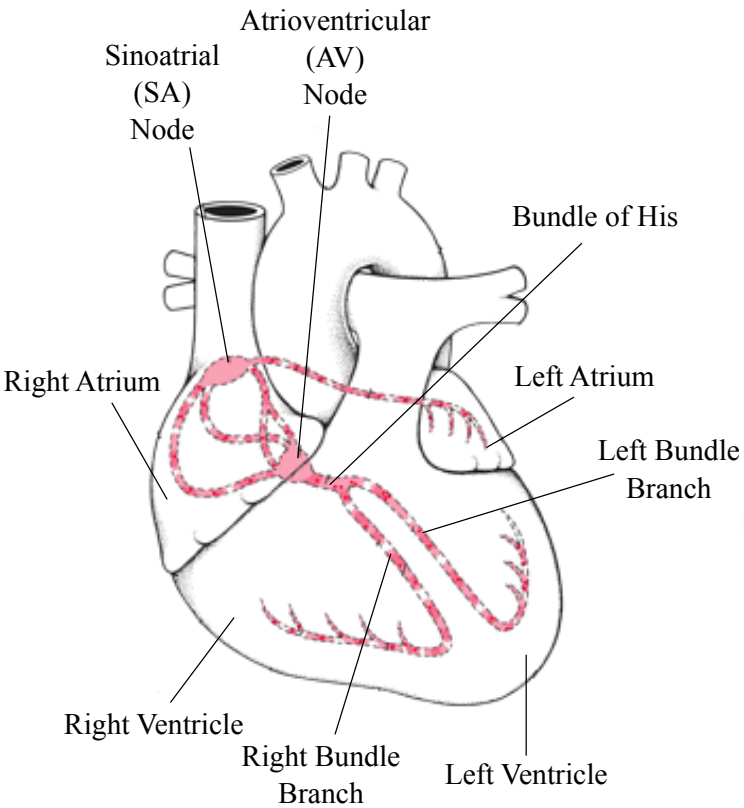


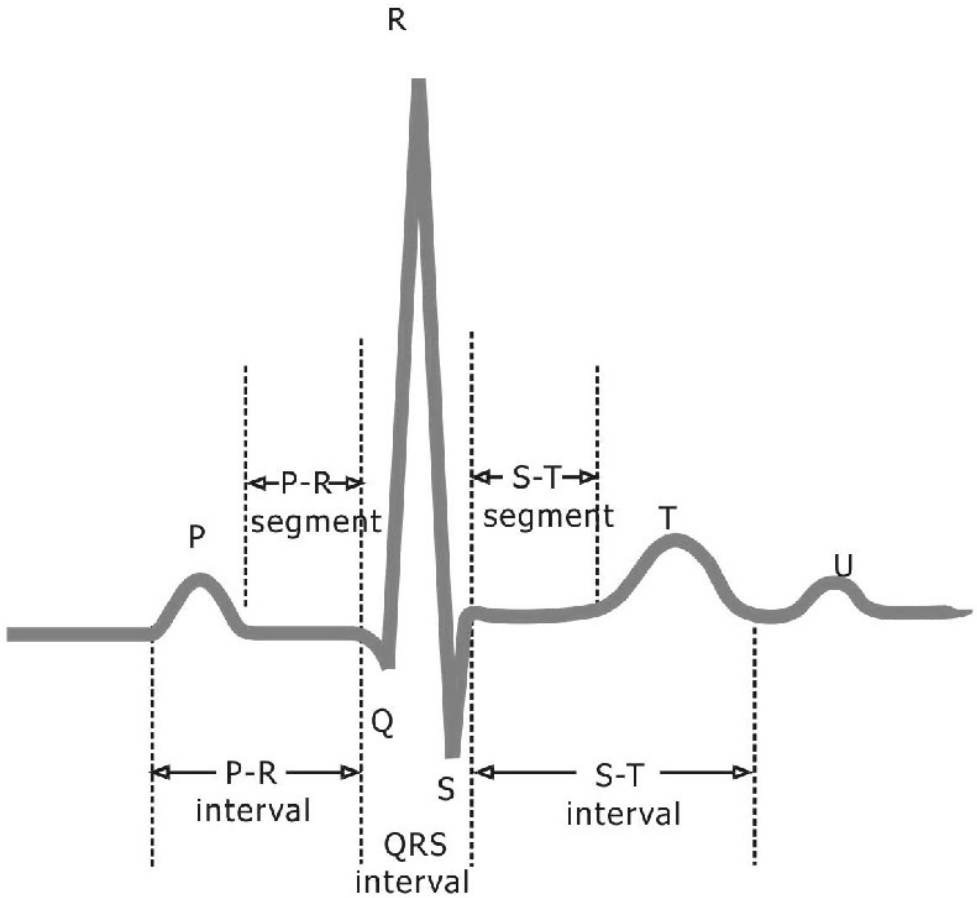
Cardiac Pacemaker

Heart System



Electrical Signal of the Heart





The Cardiac Pacemaker

Pacemaker

A pacemaker is an electronic device implanted in the body to regulate the abnormal heart rhythm (bradycardia).
Type of pacemakers:1,2 and 3-Electrodes.

The Cardiac Pacemaker

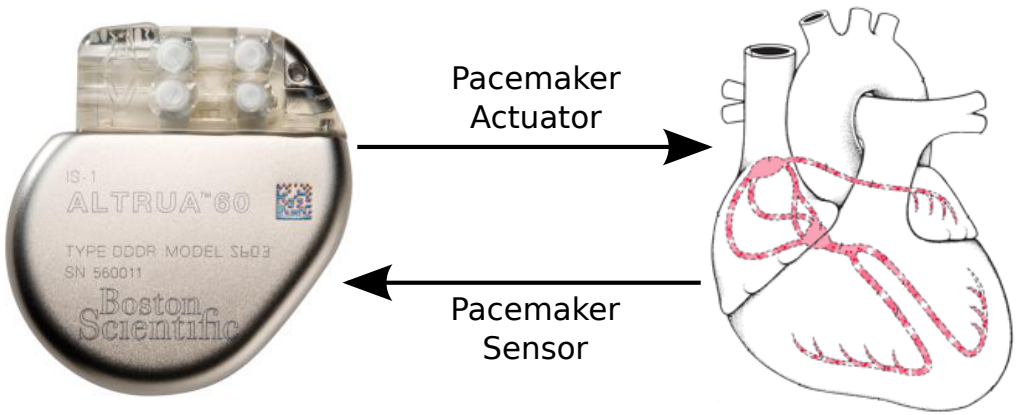
Operating Modes : NASPE/BPEG Generic Code

Category	**Chambers Paced**	**Chambers Sensed**	**Response to Sensing**	**Rate Modulation**
Letters	**O**-None **A**-Atrium **V**-Ventricle **D**-Dual(A+V)	**O**-None **A**-Atrium **V**-Ventricle **D**-Dual(A+V)	**O**-None **T**-Triggered **I**-Inhibited **D**-Dual(T+I)	**R**-Rate Modulation

i.e. AOO, VOO, AAI, AAT, VVI, VVT, AATR,VVTR, AOOR etc. . .
Periodic stimuli : (AOO, VOO and DOO)
Aperiodic stimuli : (AAI, VVI, DDD, DDI, etc.)

The Cardiac Pacemaker

- Simple Pacemaker Model
- Closed-loop Model



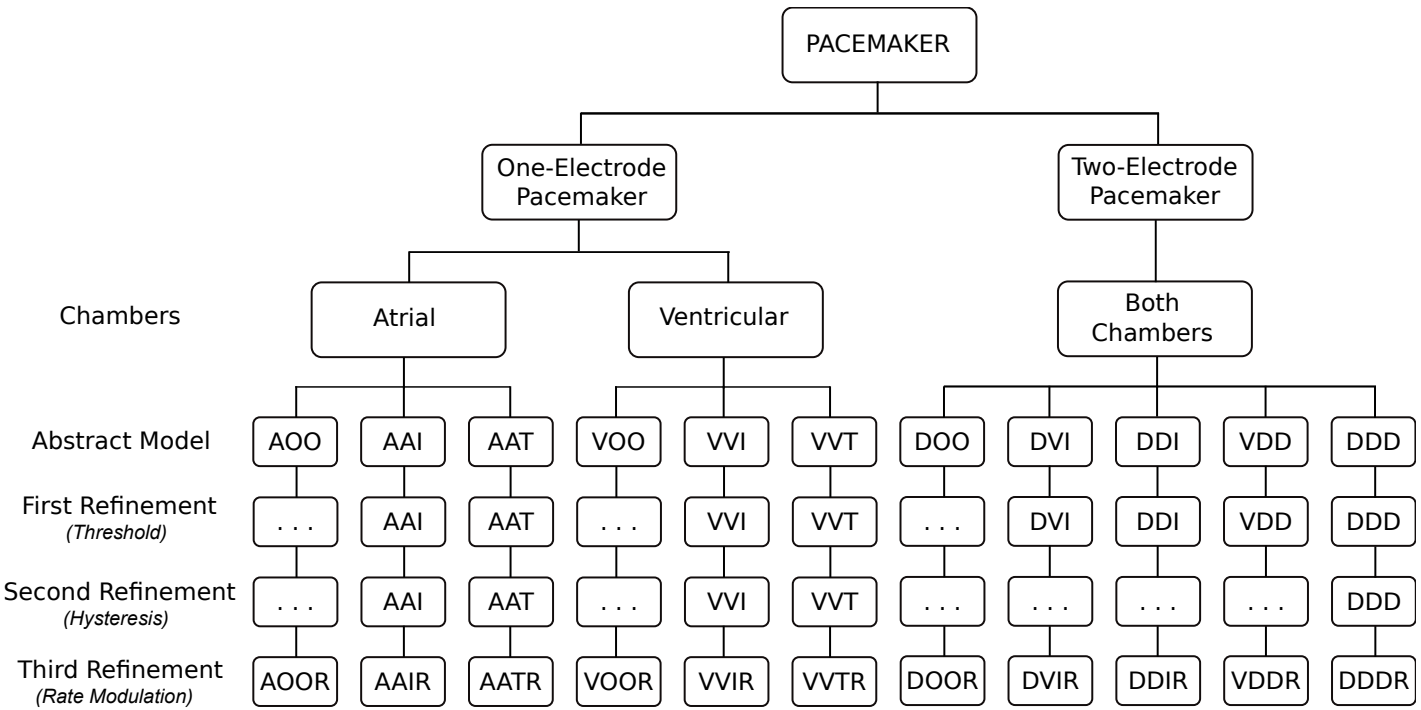
The Cardiac Pacemaker

Design Patterns

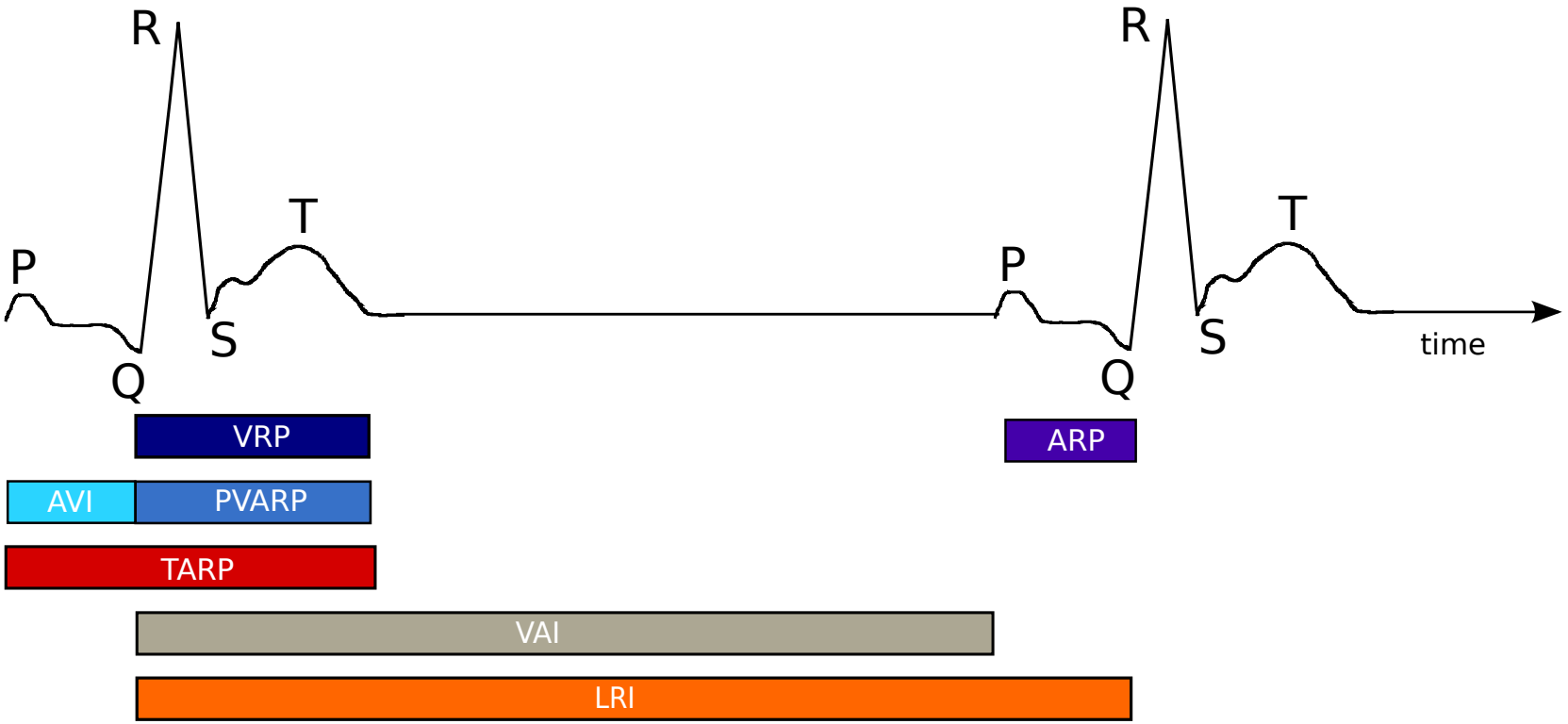
- Action Reaction Patterns
- Real Time Patterns

Development of Cardiac Pacemaker

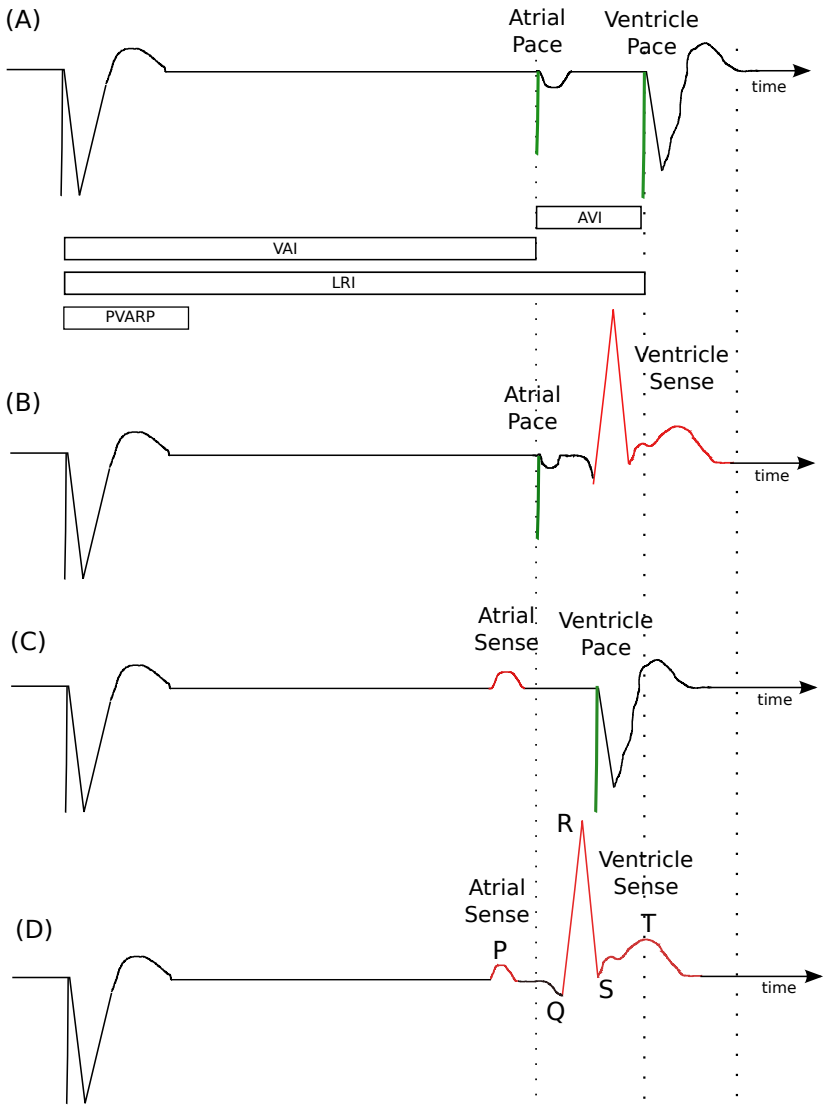
- Formalization of 1 and 2-electrode cardiac pacemaker



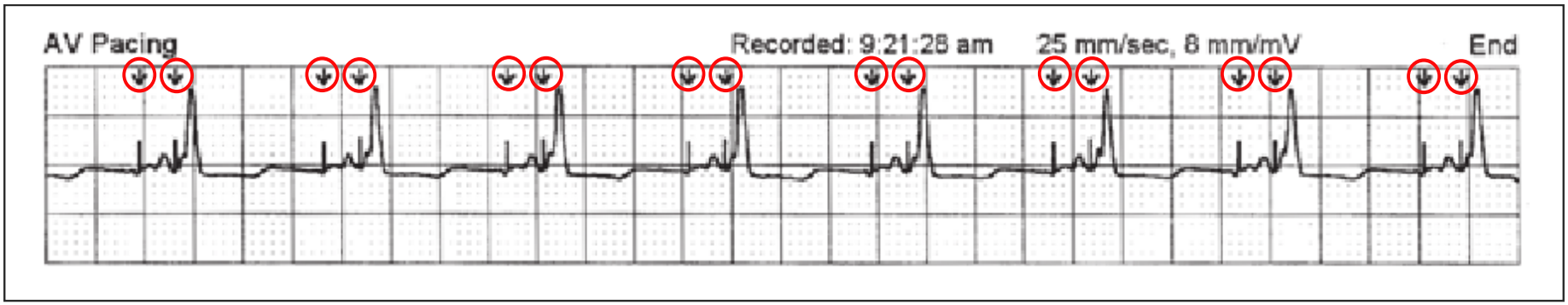
Timing Cycles



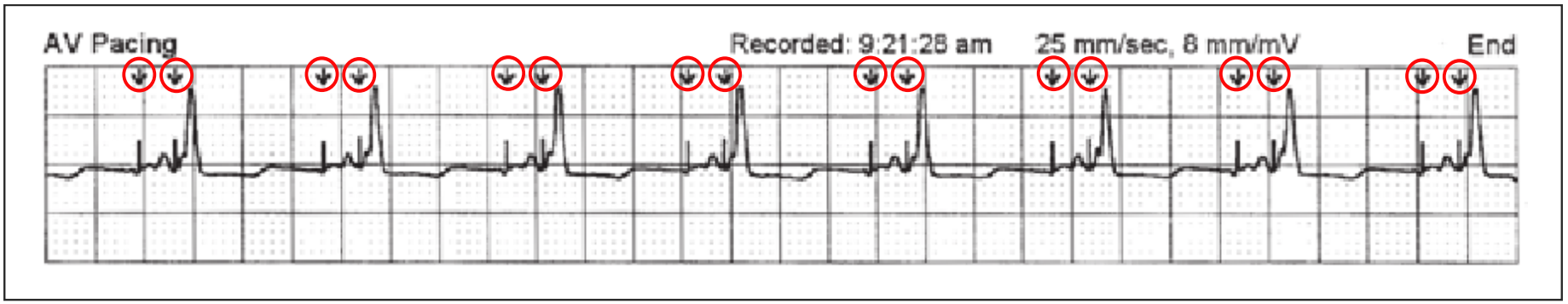
The DDD Pacing Scenarios



DDD Pacing Modes

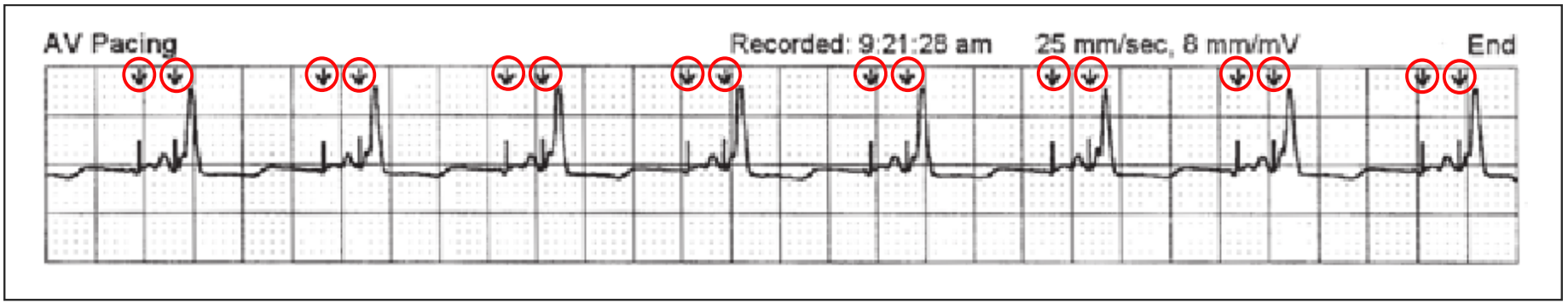


DDD Pacing Modes



$axm1 : LRL \in 30 \mathinner{..} 175$
$axm2 : URL \in 50 \mathinner{..} 175$
$axm3 : URI \in \mathbb{N}_1 \ \wedge URI = 60000/URL$
$axm4 : LRI \in \mathbb{N}_1 \ \wedge LRI = 60000/LRL$
$axm5 : status = \{ON, OFF\}$
$axm6 : FixedAV \in 70 \mathinner{..} 300$
$axm7 : ARP \in 150 \mathinner{..} 500$
$axm8 : VRP \in 150 \mathinner{..} 500$
$axm9 : PVARP \in 150 \mathinner{..} 500$
$axm10 : V\_Blank \in 30 \mathinner{..} 60$
$\ldots$

DDD Pacing Modes



$axm1 : LRL \in 30 \mathinner{..} 175$
$axm2 : URL \in 50 \mathinner{..} 175$
$axm3 : URI \in \mathbb{N}_1 \ \wedge URI = 60000 / URL$
$axm4 : LRI \in \mathbb{N}_1 \ \wedge LRI = 60000 / LRL$
$axm5 : status = \{ON, OFF\}$
$axm6 : FixedAV \in 70 \mathinner{..} 300$
$axm7 : ARP \in 150 \mathinner{..} 500$
$axm8 : VRP \in 150 \mathinner{..} 500$
$axm9 : PVARP \in 150 \mathinner{..} 500$
$axm10 : V\_Blank \in 30 \mathinner{..} 60$
. . .

$inv1 : PM\_Actuator\_A \in status$
$inv2 : PM\_Sensor\_A \in status$
$inv5 : Pace\_Int \in URI \mathinner{..} LRI$
$inv6 : sp \in 1 \mathinner{..} Pace\_Int$
$inv7 : last\_sp \geq PVARP \wedge last\_sp \leq Pace\_Int$

$inv11 : sp < VRP \wedge AV\_Count\_STATE = FALSE \Rightarrow$
$PM\_Actuator\_V = OFF \wedge PM\_Sensor\_A = OFF \wedge$
$PM\_Sensor\_V = OFF \wedge PM\_Actuator\_A = OFF$

$inv12 : Pace\_Int\_flag = FALSE \wedge PM\_Actuator\_V = ON \Rightarrow$
$sp = Pace\_Int \vee (sp < Pace\_Int \wedge$
$AV\_Count > V\_Blank \wedge AV\_Count \geq FixedAV)$

$inv13 : Pace\_Int\_flag = FALSE \wedge PM\_Actuator\_A = ON \Rightarrow$
$(sp \geq Pace\_Int - FixedAV)$
. . .

DDD Pacing Modes

EVENT Actuator_ON_V
WHEN
grd1 : $PM\_Actuator\_V = OFF$
grd2 : $(sp = Pace\_Int)$
$\lor$
$(sp < Pace\_Int \land AV\_Count > V\_Blank \land AV\_Count \geq FixedAV)$
grd3 : $sp \geq VRP \land sp \geq PVARP$
THEN
act1 : $PM\_Actuator\_V := ON$
act2 : $last\_sp := sp$
END

EVENT Actuator_OFF_V
WHEN
grd1 : $PM\_Actuator\_V = ON$
grd2 : $(sp = Pace\_Int)$
$\lor$
$(sp < Pace\_Int \land AV\_Count > V\_Blank \land AV\_Count \geq FixedAV$
grd3 : $AV\_Count\_STATE = TRUE$
grd4 : $PM\_Actuator\_A = OFF$
grd5 : $PM\_Sensor\_A = OFF$
THEN
act1 : $PM\_Actuator\_V := OFF$
act2 : $AV\_Count := 0$
act3 : $AV\_Count\_STATE := FALSE$
act4 : $PM\_Sensor\_V := OFF$
act5 : $sp := 1$
END

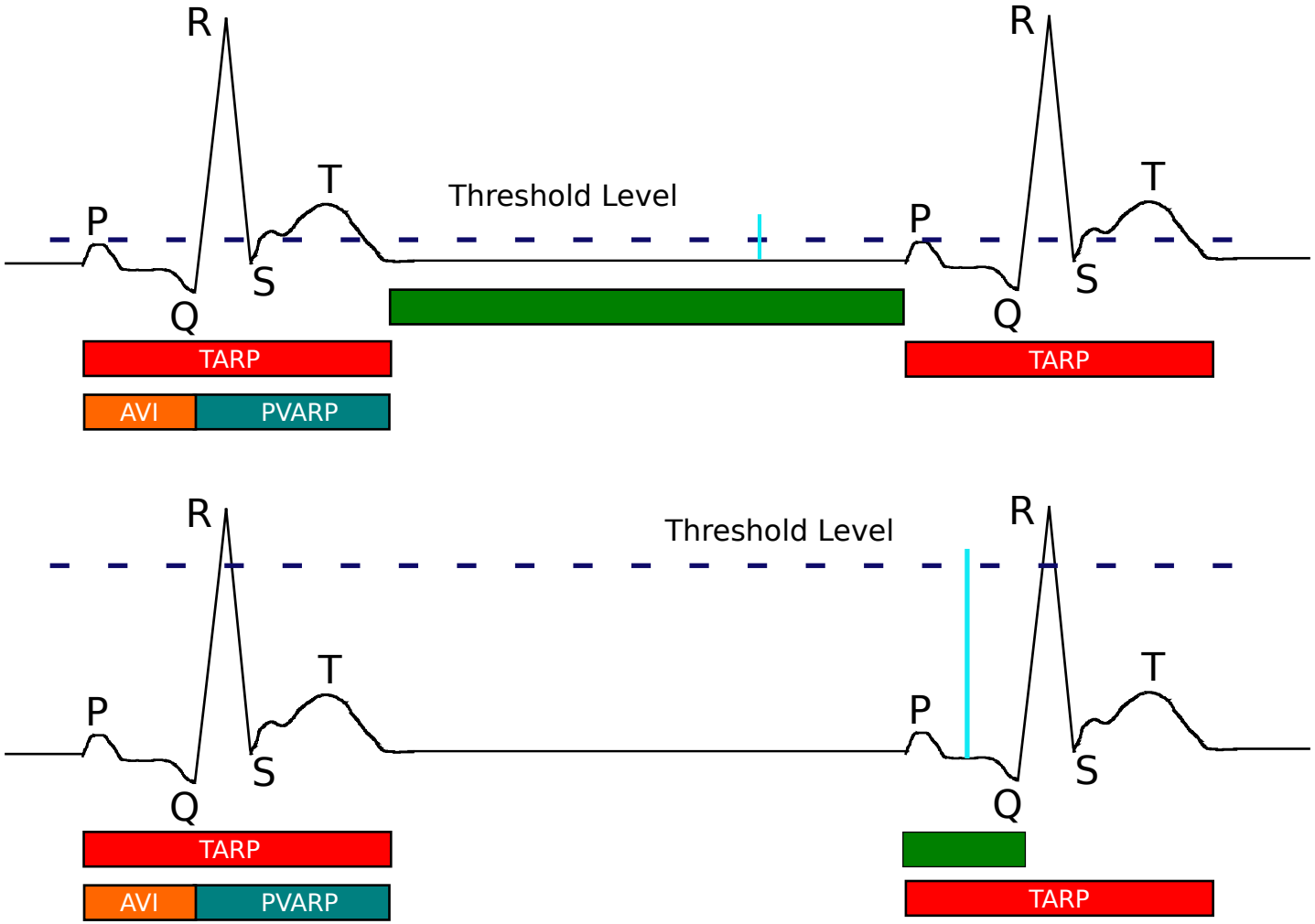
EVENT tic
WHEN
grd1 : $(sp < VRP)$
$\lor$
$(sp \geq VRP \land sp < Pace\_Int - FixedAV \land PM\_Sensor\_A = ON \land PM\_Sensor\_V = ON$
THEN
act1 : $sp := sp + 1$
END

DDD Pacing Modes

EVENT Sensor_ON_V
WHEN
grd1 : $PM\_Sensor\_V = OFF$
grd2 : $(sp \geq VRP \wedge sp < Pace\_Int - FixedAV \wedge PM\_Sensor\_A = ON)$
$\vee$
$(sp \geq Pace\_Int - FixedAV \wedge AV\_Count\_STATE = TRUE)$
grd4 : $PM\_Actuator\_A = OFF$
THEN
act1 : $PM\_Sensor\_V := ON$
END

EVENT Sensor_OFF_V
WHEN
grd1 : $PM\_Sensor\_V = ON$
grd2 : $(sp < Pace\_Int - FixedAV)$
$\vee$
$(sp \geq Pace\_Int - FixedAV \wedge sp < Pace\_Int)$
grd4 : $PM\_Actuator\_V = OFF$
grd5 : $PM\_Actuator\_A = OFF$
grd6 : $sp \geq VRP \wedge sp \geq PVARP$
THEN
act1 : $PM\_Sensor\_V := OFF$
act2 : $last\_sp := sp$
act3 : $sp := 1$
act5 : $PM\_Sensor\_A := OFF$
act6 : $AV\_Count := 0$
act7 : $AV\_Count\_STATE := FALSE$
END

First Refinement (Threshold): Sensor Activity in DDD



First Refinement (Threshold): Sensor Activity in DDD

$inv1 : Thr\_A \in \mathbb{N}_1 \land Thr\_V \in \mathbb{N}_1$

$inv2 : Pace\_Int\_flag = FALSE \land sp > VRP \land sp < Pace\_Int - FixedAV \Rightarrow PM\_Sensor\_V = ON$

$inv3 : Pace\_Int\_flag = FALSE \land sp > Pace\_Int - FixedAV \land sp < Pace\_Int \land AV\_Count\_STATE = TRUE \Rightarrow$
$PM\_Sensor\_A = OFF \land PM\_Sensor\_V = ON \land PM\_Actuator\_A = OFF$

EVENT Thr_Value_A
ANY
Thr_A_val
WHERE
grd1 : $PM\_Sensor\_A = ON$
grd2 : $Thr\_A\_val \in \mathbb{N}$
grd3 : $Thr\_A\_State = TRUE$
grd5 $Thr\_A < STA\_THR\_A$
grd6 $(sp \geq VRP \land sp < Pace\_Int - FixedAV)$
THEN
act1 : $Thr\_A := Thr\_A\_val$
act2 : $Thr\_A\_State := FALSE$
END

Second Refinement: Hysteresis

What is Hysteresis?

EVENT Hyt_Pace_Updating Refines Change_Pace_Int
ANY
Hyt_Pace_Int
WHERE
grd1 : $Pace\_Int\_flag = TRUE$
grd2 : $Hyt\_Pace\_Int\_flag = TRUE$
grd3 : $Hyt\_Pace\_Int \in Pace\_Int \mathinner{..} LRI$
THEN
act1 : $Pace\_Int := Hyt\_Pace\_Int$
act2 : $Hyt\_Pace\_Int\_flag := FALSE$
act3 : $HYT\_State := TRUE$
END

Third Refinement: Rate Modulation

What is Rate Modulation?

Increase_Interval Refines Change_Pace_Int
WHEN
grd1 : $grd1 : Pace\_Int\_flag = TRUE$
grd1 : $acler\_sensed \geq threshold$
grd1 : $HYT\_State = FALSE$
THEN
act1 : $Pace\_Int := 60000/MSR$
act1 : $acler\_sensed\_flag := TRUE$
END

inv3 : $acler\_sensed < acc\_thr \wedge acler\_sensed\_flag = TRUE \Rightarrow Pace\_Int = 60000/LRL$
inv4 : $acler\_sensed \geq acc\_thr \wedge acler\_sensed\_flag = TRUE \Rightarrow Pace\_Int = 60000/MSR$

Validation, Proof Statistics and Code Generation

ProB

Used to check an expected behavior of each operating mode.

Proof Statistics

Model	Total number of POs	Automatic Proof	Interactive Proof
One-electrode pacemaker			
Abstract Model	203	199(98%)	4(2%)
First Refinement	48	44(91%)	4(9%)
Second Refinement	12	8(66%)	4(34%)
Third Refinement	105	99(94%)	6(6%)
Two-electrode pacemaker			
Abstract Model	204	195(95%)	9(5%)
First Refinement	234	223(95%)	11(5%)
Second Refinement	3	3(100%)	0(0%)
Third Refinement	83	74(89%)	9(11%)
Total	892	845(94%)	47(6%)

Code Generation using EB2ALL.